Complexity of Total Search Problems: Equilibria and Fair Division

Kasper Høgh

PhD Dissertation

Department of Computer Science
Aarhus University
Denmark

Complexity of Total Search Problems: Equilibria and Fair Division

A Dissertation
Presented to the Faculty of Natural Sciences
of Aarhus University
in Partial Fulfillment of the Requirements
for the PhD Degree

by
Kasper Høgh
May 31, 2023

Abstract

In this thesis, we study the computational complexity of various total search problems. A search problem being *total* means that for any instance of the problem a solution is guaranteed to exist. Totality typically follows from some underlying principle, such as the existence of a sink in a directed graph or the existence of a fixed point of a continuous map from the unit disk to itself. One approach to studying the difficulty of computational problems is to classify them into a hierarchy of complexity classes. In the area of total search problems, these classes correspond to the underlying principles guaranteeing existence of solutions.

In the first part of this thesis, we present a useful technique for proving membership in the complexity classes PPAD and FIXP, both related to Brouwer's fixed point theorem. Applying this tool we greatly simplify the proofs of many known containment results, and we also manage to prove novel ones. Notably, we are able to prove that the exact CakeCutting problem is FIXP-complete.

Secondly we study problems related to the Borsuk-Ulam theorem. Previous work has shown that the ConsensusHalving problem is contained in the class BU, corresponding to the aforementioned theorem, but the question of hardness remained open. We present a hardness result for the problem of computing a strong approximate consensus halving.

Finally, we study the EFX problem from fair division of indivisible items. In this problem one has to divide a set of indivisible goods among a set of agents in a fair manner. We study this problem in the case of two agents and characterize its difficulty by identifying the classes of utility functions for which the problem is tractable. Specifically, we give a polynomial-time algorithm that works for a large class of utility functions, including gross substitutes and budget-additive valuations, and we prove that the problem is PLS-hard for submodular valuations.

Resumé

I denne afhandling studerer vi kompleksiteten af forskellige totale søgeproblemer. Et søgeproblem siges at være *totalt* hvis enhver instans af problemet har en løsning. En sådan garanti følger ofte af et mere fundamentalt eksistensresultat: for eksempel at enhver orienteret acyklisk graf har et dræn eller at enhver kontinuert afbildning fra enhedskuglen til sig selv har et fikspunkt. En fremgangsmåde i projektet om at beskrive den relative sværhedsgrad af forskellige beregningsproblemer er at klassificere disse i et hierarki af kompleksitetsklasser. I studiet af totale søgeproblemer svarer disse klasser til de underliggende principper, som sikrer eksistens af løsninger.

I den første del af denne afhandling præsenterer vi en ny teknik der har vist sig nyttig i beviser for medlemskab i klasserne PPAD og FIXP, der begge er relateret til Brouwer's fikspunktssætning. Ved at anvende dette redskab giver vi væsentligt enklere beviser for kendte resultater, og vi formår også at klassificere problemer hvis kompleksitet tidligere var ukendt. Her kan for eksempel nævnes at den eksakte version af KageDelings problemet er FIXP-fuldstændigt.

Dernæst studerer vi problemer, der relaterer sig til Borsuk-Ulams sætning. Tidligere arbejde har placeret KonsensusHalverings problemet i klassen BU, som ikke overraskende svarer til Borsuk-Ulams sætning. De lod imidlertid spørgsmålet om hårdheden af problemet stå uløst hen. Vi præsenterer et hårdhedsresultat for problemet at beregne en såkaldt stærk approksimation af en konsensus halvering.

Endeligt studerer vi EFX-problemet der omhandler retfærdig fordeling af udelelige goder blandt en grupper personer. Vi studerer dette problem i tilfældet hvor der er to personer, og vi formår at karakterisere dets sværhedsgrad i forhold til hvor komplicerede agenternes nyttefunktioner er. Mere specifikt giver vi en effektiv algoritme som beregner en EFX-allokering for en bred klasse af nyttefunktioner, og vi viser at problemet er PLS-fuldstændigt for submodulære valuationer.

Acknowledgments

First of all, I would like to thank my advisor Kristoffer Arnsfelt Hansen for sharing his ideas with me and always being available for having discussions and answering questions. Had it not been for him guiding me in a sensible direction, this thesis would never have materialized.

I would also like to thank my other collaborators Aris Filos-Ratsikas, Alexandros Hollender, and Paul Goldberg. I started working with Aris and Alex soon after starting my PhD. The many meetings we had were a pleasure and have resulted in work making up a large portion of this thesis. Paul and Alex also hosted me for a very enjoyable and inspiring stay in Oxford during the spring of 2022. I would also like to thank everyone in the algorithms group and complexity group here in Aarhus for creating a good environment.

Finally, I thank my parents and my brother for their love and support throughout the years.

Kasper Høgh,
Aarhus, May 31, 2023.

Contents

Part I

Overview

Chapter 1

Introduction

In Economics and Game Theory, one studies the interaction between rational agents. A fundamental question that arises in these settings is whether the behavior of the agents will eventually converge to some stable state. For instance, in the case of games, one may ask if each player can pick a strategy in such a way that no player has an incentive to deviate if given the opportunity to do so unilaterally; in the case of markets, one may wonder if it is possible to price the available goods in such a way that supply equals demand. Such balanced states, which the game or the market should eventually converge toward, are known as *equilibria*. Several major results in the study of Economics and Game Theory amounts to establishing the existence of equilibria in various different settings. A common theme in the proofs of these and many similar results is the application of general mathematical existence theorems. For instance,

- Nash's theorem [175] that every finite game admits an equilibrium was initially proved by applying Kakutani's fixed point theorem [149] (and, soon thereafter, the simpler fixed-point theorem of Brouwer [39]).
- Arrow and Debreu [8] established the existence of an equilibrium in their market model by applying a fixed point theorem of Debreu [68].
- Several problems in fair division, such as the problem of cutting a cake, divvying up the rent for a shared apartment, or dividing an inheritance between two families, can also be attacked using some of the fixed point theorems mentioned above and others such as the Borsuk-Ulam theorem [34].

One goal of computer science is to study the inherent difficulty of interesting computational problems. One strong indication that a given problem is hard is that it is NP-complete. However, the theory of NP-completeness is phrased in terms of decision problems: does a Boolean formula have a satisfying assignment? If a given game has an equilibrium is not a very interesting decision problem, because we know that this is always the case by Nash's theorem. Of course, one can make the question more interesting by asking if a game has an equilibrium satisfying some further conditions.

Instead we will study search variants of the questions above: given some game, compute an equilibrium! In this thesis, we study the difficulty of various search problems by placing them in appropriate complexity classes.

1.1 Outline of Thesis

In this thesis, we study various total search problems. In Part I we survey the wider research area and describe how our results contribute to the field. Part II consists of the following publications:

- Chapter 3

  FIXP-Membership via Convex Optimization: Games, Cakes, and Markets [97]
  A. Filos-Ratsikas, K. A. Hansen, K. Høgh, and A. Hollender.
  62nd IEEE Symposium on Foundations of Computer Science (FOCS 2021)

- Chapter 4

  PPAD-membership for Problems with Exact Rational Solutions: A General Approach via Convex Optimization.
  A. Filos-Ratsikas, K. A. Hansen, K. Høgh, and A. Hollender.
  Manuscript

- Chapter 5

  Strong Approximate Consensus Halving and the Borsuk-Ulam Theorem [22]
  E. Batziou, K. A. Hansen, and K. Høgh.
  48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)

- Chapter 6

  The Frontier of Intractability for EFX with Two Agents [130]
  P. Goldberg, K. Høgh, and A. Hollender.
  Manuscript

The papers are all included in a largely unedited state except for containing minor corrections and changes to formatting. For the two articles that have been published, we include the full arXiv versions. I have had a proportional impact on the development of arguments in all of the articles. In chapter 3, I have written most of section 5 on cake cutting and the part of section 6 about the Hylland-Zeckhauser mechanism. In chapter 4, I wrote preliminary versions of the sections on markets. The presentation in the current paper is slightly changed, but the proofs are essentially the same. This is also the case for the section on pacing equilibria. In chapter 5, I wrote parts of section 4 (except subsection 4.2), and sections 5 and 6. In chapter 6, I wrote a large portion of all sections except for the introduction.

Chapter 3 and Chapter 4 are closely related. In the former, we develop a novel technique for proving membership results in the complexity class FIXP, originally introduced by Etessami and Yannakakis [85] in order to capture problems for which a solution is guaranteed to exist by Brouwer's fixed point theorem. Our main contribution is the construction of a so-called *pseudogate* for computing solutions to, for example, linear optimization problems. Applying this general tool, we obtain FIXP-membership results for a wide range of problems, including the *envy-free cake cutting problem* and the problem of computing an *equilibrium in an Arrow-Debreu market*. In the latter chapter, we present a similar technique for proving membership results in the class PPAD. One may view PPAD as the linear fragment LinearFIXP of FIXP. However, the class LinearFIXP is more restrictive than FIXP, and so more care has to be taken in the constructions. In Chapter 5, we study total search problems related to the Borsuk-Ulam theorem. The Borsuk-Ulam theorem has several equivalent formulations. We introduce a complexity class related to one of these formulations and show how this class relates to the computational problems related to the other formulations studied in previous work. Further, we show that the problem of computing a *strong approximation* of a solution to the *Consensus Halving* problem is complete for the class BU_a. In Chapter 6, we study the problem of dividing a set of indivisible goods among identical agents such that the allocation is *envy-free up to any good (EFX)*. We characterize the hardness of the problem in terms of well-known valuation classes. In the positive direction, we introduce the class of weakly well-layered valuation functions and present a simple greedy algorithm that solves the problem for this class. On the other hand, we show that the problem is PLS-complete for submodular valuations. The main technical contribution is showing that the problem of computing a local optimum on a Kneser graph is PLS-hard.

Chapter 2

Preliminaries

In the first section of this chapter, we describe some preliminaries on search problems common to the remaining sections in which we describe our results. The main objectives of this chapter is to (1) present our results and put them into context, and (2) give an intuitive overview of our arguments.

2.1 Search Problems

A *search problem* Π has a set of instances I given as strings over some finite alphabet Σ. It is assumed that when confronted with some string I over Σ, one may decide whether I is a valid instance of Σ in polynomial time in the input length $|I|$. Every valid instance I has a *domain* D_I in which one has to search for a solution. In this thesis we consider both *discrete* search problems, where $D_I \subseteq \{0,1\}^{d_I}$, and *real-valued* search problems, where $D_I \subseteq \mathbb{R}^{d_I}$. In either case, it is assumed that d_I can be computed in polynomial time in the length of the instance. We denote the *set of solutions* to the instance by $\mathrm{Sol}(I) \subseteq D_I$. The complexity class NP consisting of decision problems that can be solved by a polynomial-time nondeterministic Turing machine has a search analogue called FNP. This class consists of all discrete search problems Π, where there exists a polynomial time Turing machine that given I and $w \in \{0,1\}^{d_I}$ checks whether $w \in \mathrm{Sol}(I)$. In analogy with P, the class FP consists of all NP search problems for which there exists a polynomial-time Turing machine that given an instance I computes a solution w from $\mathrm{Sol}(I)$ if this set is non-empty and rejects otherwise. Because of the self-reducibility of SAT (meaning that search reduces to decision), it holds that P = NP if and only if FP = FNP.

In this thesis, however, we will consider *total search problems* exclusively, meaning that $\mathrm{Sol}(I) \neq \emptyset$ for any valid instance I. Such problems are typically based on some general existence result, for instance Brouwer's fixed point theorem or the fact that any directed acyclic graph has a sink.

2.1.1 TFNP

The class TFNP, short for *total function nondeterministic polynomial*, consists of all problems Π in FNP that are total, meaning that $\text{Sol}(I) \neq \emptyset$ for every instance I of Π. The study of TFNP is related to the question of P versus NP. Indeed, on the one hand TFNP contains problems that are widely believed to not be solvable in polynomial time. For instance, because of the fundamental law of arithmetic, TFNP contains the problem of computing prime factorizations. On the other hand, Megiddo and Papadimitriou [170] showed that TFNP is unlikely to contain any FNP-hard problems. Indeed, a reduction from the search version of SAT to a total NP search problem would imply the existence of certificates of unsatisfiability for unsatisfiable Boolean formulas, which would imply that NP = coNP.

Reductions. Let Π_1 and Π_2 be two TFNP problems. One says that Π_1 *many-one reduces* to Π_2 if there exists a pair of polynomial-time computable functions f and g such that (i) the function $f\colon \Sigma^* \to \Sigma^*$ maps any instance I of Π_1 to an instance $f(I)$ of Π_2, and (ii) the function $g\colon \Sigma^* \times \Sigma^* \to \Sigma^*$ maps solutions $y \in \text{Sol}(f(I))$ to the instance $f(I)$ of Π_2 to a solution $g(I,y) \in \text{Sol}(I)$ to the instance I of Π_1.

PPAD

Papadimitriou [182] introduced the class PPAD to capture computational problems associated with path-following algorithms. Before providing the formal definition of PPAD, we illustrate the underlying existence principle through a proof Sperner's lemma in two dimensions [204]. This result concerns triangulations of the 2-simplex where every vertex has one of three possible labels subject to the following rules: (1) the corners of the triangle have distinct labels, and (2) every vertex on the boundary has a label that is different from that of the opposite corner. If a labeled triangulation of Δ^2 satisfies these properties, then there exists a fully labeled sub-triangle. For an example of a valid labeling, see the figure below:

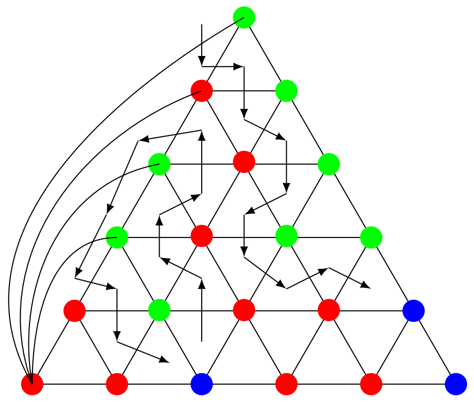


Figure 2.1: Proof of Sperner's lemma.

In the figure above, we have added a number of edges between the vertex corresponding to the first standard basis vector and all vertices on the red/green face. This does not introduce new fully labeled triangles, and it amounts to assuming that there are no vertices on the red/green face. We now initiate a walk, moving from the exterior into the triangle via the red/green edge. If the triangle we just entered is not fully labeled, then it must have another red/green edge. In this case, we walk through it and continue this procedure until a fully labeled triangle is reached. Note that this walk cannot exit the large triangle, because of the auxiliary edges we added in the beginning. Hence, in order to argue that this procedure succeeds in finding a fully labeled triangle, it suffices to prove that this walk does not enter a cycle. If this were the case, consider the *first* triangle which is entered twice. The second time around, we must enter through the unused edge, because otherwise this would not be the first triangle to be entered twice. However, this would require the triangle to have three red/green edges, which is obviously not possible. In conclusion, the described path must reach a fully labeled triangle. We note that there are two other fully labeled triangles in the figure above, corresponding to the source and sink of another path.

The proof above indicates that Sperner's lemma may be thought of as a general statement concerning graphs. Consider namely a graph that has one vertex for every sub triangle (including the exterior one) and where there is a directed edge between two triangles if they share a red/green edge (the direction is fixed by stipulating that one must cross the edge while having the green vertex on one's left). Such a graph consists of simple paths and cycles. Importantly, the vertex corresponding to the exterior has no incoming edge, so there exists at least one simple path, leading to the existence of at least one sink. In analogy with the proof above, the vertices that we wish to find are the sink of the path originating in the exterior vertex or any other source or sink. This general structure leads to the definition of PPAD. Formally, PPAD consists of all TFNP problems that many-one reduce to the following search problem: End-of-Line. A valid instance I of this problem encodes two Boolean circuits $S_I, P_I \colon \{0,1\}^n \to \{0,1\}^n$, where n is a natural number that is polynomial in the instance size, such that $P_I(0^n) = 0^n \neq S(0^n)$. These two Boolean circuits define a directed graph where the successor and predecessor of every vertex is computed efficiently by the circuits S_I and P_I. Specifically, the vertex set of the graph is $\{0,1\}^n$ and (u,v) is a directed edge if and only if $S_I(u) = v$ and $P_I(v) = u$. Note that the in- and out-degree of every vertex is at most one. Hence, the graph consists of cycles and simple paths. Furthermore, 0^n has in-degree zero, meaning that it is a source. Hence, if we start at vertex 0^n and repeatedly apply the successor function S_I, then we will eventually end up in a sink, i.e., a vertex that has out-degree zero. This sink could potentially be 0^n itself. A solution to the instance I is a sink in this graph or a source different from 0^n, that is, $\mathrm{Sol}(I) = \{x \in \{0,1\}^n \colon P_I(S_I(x)) \neq x \text{ or } S_I(P_I(x)) \neq x \neq 0^n\}$. It is clear that a candidate solution may be verified efficiently, and the problem is total by the previous discussion. Hence, End-of-Line is a TFNP problem.

In this thesis, we will mainly be proving PPAD-membership results. Instead of reducing to End-of-Line, it may be more convenient to reduce to some other problem contained in PPAD. The previous discussion shows that the computational problem associated with Sperner's lemma is in PPAD, and it is, in fact, complete [52, 182]. In section 2.2.1, we describe how Sperner's lemma can be thought of as a discrete version of Brouwer's fixed point theorem. Using this Daskalakis et al. [67] showed that computing an ε-Nash equilibrium is PPAD-complete for ≥ 3 players, and Chen et al. [55] proved that computing exact Nash equilibria in bimatrix games is PPAD-complete. Other notable results include PPAD-completeness of computing an ε-approximate envy-free cut of a cake [76], and problems related to computing equilibria in various different market settings [53, 56, 59, 118, 119, 220].

Polynomial Local Search (PLS)

Johnson et al. [147] introduced the class PLS to capture local search problems in settings where local improvements, if they exist, can be computed in polynomial time. More formally, PLS consists of all TFNP problems that reduce to certain basic PLS problems Π. An instance I of such a basic problem is specified by a domain $D_I \subseteq \{0,1\}^{p(|I|)}$, an initial solution $s_0 \in D_I$ that can be computed in polynomial time, and two polynomial-time Turing machines, C_I and N_I, for computing the objective value $C_I(x) \in \mathbb{N}$ and the neighborhood $N_I(x) \subseteq D_I$ of any feasible solution $x \in D_I$. It is assumed that membership in D_I can be decided in polynomial time. The solution set of I then consists of all feasible solutions that are locally optimal. For instance, in the case of a minimization problem, $\mathrm{Sol}(I) = \{x \in D_I \colon C_I(x) \leq C_I(y) \text{ for all } y \in N_I(x)\}$.

2.2 Fixed Point Computation

In this section we discuss our contributions to proving FIXP- and PPAD-membership results as described in the papers [96, 97]. Before moving on to a discussion of these papers and their relation to the literature, we begin this section with a description of these complexity classes, motivated through the computational problem Nash. Both FIXP and PPAD are related to the following well-known existence theorem:

Theorem 2.2.1 (Brouwer's fixed point theorem [39])**.** *Let $X \subseteq \mathbb{R}^n$ be a compact and convex set. If $f\colon X \to X$ is a continuous map, then there exists a point $x^* \in X$ such that $f(x^*) = x^*$. We say that x^* is a fixed point of f.*

Using this theorem Nash [175] gave a proof that any finite normal form game admits an equilibrium. In a finite normal form game Γ with n players, every player i has a finite set of pure strategies $S_i = \{1, 2, \ldots, m_i\}$, and for every tuple of pure strategies $(j_1, \ldots, j_n) \in S_1 \times \cdots \times S_n$ it has a payoff $u_{i,j_1,\ldots,j_n} \in \mathbb{R}_{\geq 0}$. While there might not exist a stable state where the players use only pure strategies, Nash proved that such a state is guaranteed to exist if the players are allowed to play according to *mixed strategies*, i.e., probability distributions over their pure strategies. In this case, the payoff of the players is given as the expected payoff, denoted $\overline{u}$, when the agents play according to their mixed strategies. Let $\Delta(S_i) = \{x \in \mathbb{R}^{m_i}_{\geq 0} \colon \sum_{j=1}^{m_i} x_j = 1\}$ denote the set of mixed strategies of player i, and let $D = \prod_{i=1}^{n} \Delta(S_i)$. A tuple of mixed strategies $(x_1,\dots,x_n) \in D$ is an *equilibrium* of the game if, for every agent i, it holds that $x_i \in \arg\max_{x \in \Delta(S_i)} \overline{u}_i(x, x_{-i})$, where $(x, x_{-i}) := (x_1,\dots,x_{i-1},x,x_{i+1},\dots,x_n)$. In this case, we say that x_i is a *best response* to the mixed strategies of the other agents. As mentioned previously, Nash proved that an equilibrium point is guaranteed to exist by constructing a continuous map $D \to D$ whose fixed points are in one-to-one correspondence with equilibria of the game and then appealing to Brouwer's fixed point theorem. As we shall see soon, this proof immediately implies that the problem of computing an exact mixed Nash equilibrium is contained in FIXP.

Nash, however, gave a considerably simpler proof for the existence of equilibria in finite games in a previous paper of his [177]. This proof, which we describe below, makes use of a different fixed point theorem due to Kakutani [149]. However, as we shall see, it is not immediately clear how this proof can be converted into a FIXP-membership result. Kakutani's fixed point theorem, which is a generalization of that of Brouwer, concerns correspondences $X \rightrightarrows X$, that is, maps $X \to \mathcal{P}(X)$ from X to its powerset, where X is a convex and compact subset of Euclidean space. Generally, one says that the graph of a correspondence $f\colon X \rightrightarrows Y$ is closed if $\{(x,y) \in X \times Y : y \in f(x)\}$ is a closed set. We can now state the theorem:

Theorem 2.2.2 (Kakutani's fixed point theorem [149])**.** *Let $X \subseteq \mathbb{R}^n$ be a compact and convex set. Let $f\colon X \rightrightarrows X$ be a nonempty- and convex-valued correspondence whose graph is closed. Then there exists a point $x^* \in X$ such that $x^* \in f(x^*)$.*

With notation as above, Nash defined a correspondence $f\colon D \rightrightarrows D$ by $(x_1,\dots,x_n) \mapsto \times_{i=1}^n \arg\max_{x \in \Delta(S_i)} \overline{u}_i(x, x_{-i})$. Note that, by definition, if x is a fixed point of f, then the mixed strategy x_i is a best response to x_{-i} for every player i. Hence, the fixed points of f are exactly the equilibria of the game. As $\arg\max_{x \in \Delta(S_i)} \overline{u}_i(x, x_{-i})$ is the set of optimal solutions for a feasible linear program, it is easy to verify that f is nonempty- and convex-valued. Furthermore, by continuity of the payoff functions, the graph of f is closed. As such, Kakutani's fixed point theorem applies to prove the existence of a fixed point, and therefore of an equilibrium.

The main result of our papers, which we shall describe in the remainder of this section, gives a method for converting this proof into a FIXP-membership result. Before we get to that, however, we introduce the relevant complexity classes.

2.2.1 The classes FIXP and LinearFIXP

Etessami and Yannakakis [86] introduced the class FIXP as a class of real-valued search problems where solutions can be realized as fixed points of suitably defined functions. They define the class by (1) introducing a family of canonical FIXP-problems, and (2) closing the class under a suitable type of reductions. A search problem Π is one of these canonical, or in our terms *basic*, FIXP-problems if (i) for a given instance I of Π one may in polynomial time compute a description of a linear feasibility problem with rational coefficients defining a domain $D_I \subseteq \mathbb{R}^{d_I}$ and a description of a circuit C_I with rational coefficients and gates from $\{+,-,*,\div,\max,\min\}$ defining a continuous function $F_I\colon D_I \to D_I$, and (ii) the set of solutions of I is the fixed points of F_I, i.e., $\mathrm{Sol}(I) = \{x \in D_I \colon F_I(x) = x\}$. We note that Etessami and Yannakakis defined FIXP using a different set of gates, but they proved that the one chosen here defines the same class.

The class is then closed under *SL-reductions*, that is, FIXP consists of all real-valued search problems that SL-reduce to a basic FIXP-problem. An SL-reduction from Π_1 to Π_2 is a many-one reduction (f, g) where the solution mapping g is *separable linear*. This means that for an instance I of Π_1 one may compute an map of indices $\pi\colon \{1, \dots, d_I\} \to \{1, \dots, d_{f(I)}\}$ and rational coefficients a_i, b_i such that for $y \in \mathrm{Sol}(f(I))$ the solution mapping is given as follows: $g(I, y) = x$, where $x_i = a_i y_{\pi(i)} + b_i$ for $i = 1 \dots, d_I$. In particular, by choosing $a_i = 1$ and $b_i = 0$, it is possible to simply project away some of the coordinates of y.

LinearFIXP. Etessami and Yannakakis [85] also considered the subclass LinearFIXP consisting of problems that reduce to basic FIXP problems where the circuits C_I are restricted to only use gates from the set $\{+, \max, *\zeta\}$. They gave the following characterization: PPAD = LinearFIXP. For us, it is the inclusion from right to left which is useful. The basic idea of this inclusion is that given some circuit computing a continuous map F, one may use Scarf's algorithm to find a point x^* in the domain such that $\|F(x^*) - x^*\|_\infty \leq \varepsilon$ for some sufficiently small $\varepsilon > 0$. Because F is represented by a piecewise linear circuit, one may then round x^* to an exact fixed point by solving a linear program. The upshot is that in order to show that a problem is in PPAD, it suffices to produce piecewise linear circuits whose fixed points correspond to solutions.

2.2.2 Pseudogates and optimization

Note that in the proof of Nash's theorem via Kakutani, every agent finds a best response by solving a linear program where the coefficients of the objective function are parameterized by the mixed strategies of the other players. In order to convert this proof into a FIXP-membership statement, we would therefore seemingly have to construct an algebraic circuit computing an optimal solution to a linear program parameterized by its input. However, this is impossible even for very simple problems. Consider for instance the following LP parameterized by $c \in \mathbb{R}$:

$$\begin{array}{ll} \text{maximize} & cx \\ \text{subject to} & x \in [0,1] \end{array} \tag{2.1}$$

The optimal solution makes a discontinuous jump from being $x = 0$ when $c < 0$ to being $x = 1$ when $c > 0$. As well-defined algebraic circuits compute continuous functions, it is therefore impossible to construct an algebraic circuit which on input $c \in \mathbb{R}$ computes an optimal solution to LP 2.1. Note that when $c = 0$, any $x \in [0, 1]$ is an optimal solution. Hence, the set of optimal solutions defines a correspondence

$H\colon \mathbb{R} \rightrightarrows [0,1]$ reminiscent of the Heaviside function:

$$H(c) = \begin{cases} \{0\} & \text{if } c < 0, \\ [0,1] & \text{if } c = 0, \\ \{1\} & \text{if } c > 0. \end{cases}$$

The realization that makes it possible for us to circumvent this problem of discontinuity is the following:

Observation 1. *In order to prove that problem is contained in* FIXP*, it suffices to construct a well-defined algebraic circuit C whose fixed points correspond to solutions of the problem. Hence, given that C is well-defined, one only has to reason about its behavior in fixed points.*

This observation leads us to the following central definition. Let $f\colon \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ be some correspondence. We say that an algebraic circuit G_f defining a map $\mathbb{R}^n \times [0,1]^\ell \to \mathbb{R}^m \times [0,1]^\ell$ is a *pseudogate computing* f if for any $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$, and $y \in [0,1]^\ell$, the following holds:

$$G_f(x,y) = (z,y) \Rightarrow z \in f(x) \tag{2.2}$$

The content of this implication is that if the auxiliary y-variables are fixed then G_f correctly computes the correspondence f. Figure 2.2 below illustrates how such a pseudogate can be used when constructing a FIXP-circuit. If at some point one has to compute $f(a)$, where a is some vector depending on the input x, then one simply adds auxiliary variables y to the circuit and inserts the construction of G_f. One then uses the output z of G_f in all the places where one would want to use $f(a)$. If (x,y) is a fixed point of the entire circuit, then one has that $z \in f(a)$ by definition. One can then use this fact to reason about the behavior of the circuit in fixed points. Finally, because of the choice of reductions, one can project away the auxiliary variables.

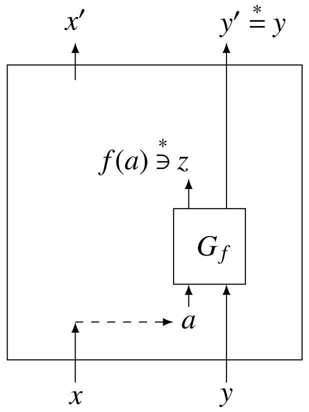


Figure 2.2: Pseudogate.

In order to illustrate this notion, we give a construction of a pseudogate computing the Heaviside function. Consider a map $G\colon \mathbb{R}\times[0,1]\to\mathbb{R}\times[0,1]$ with one auxiliary variable given by $G(x,y) = (y, \min\{1, \max\{0, x+y\}\})$. Clearly, we may define an algebraic circuit computing G. Furthermore, one may verify that G is a pseudogate computing the Heaviside correspondence. Suppose that the auxiliary variable is fixed, that is, $y = \min\{1, \max\{0, x+y\}\}$, and note that:

- If $x < 0$ then $x+y \notin \{1, y\}$, and so the assumption implies that $y = 0$.
- If $x = 0$, then clearly any $y \in [0,1]$ is fixed.
- If $x > 0$ then $x+y \notin \{0, y\}$, and so the assumption implies that $y = 1$.

In any case, it holds that $y \in H(x)$. We conclude that G is a pseudogate computing the Heaviside correspondence. This gate is instrumental in our construction of a pseudogate for solving convex optimization problems, which we sketch below in the case of linear programs. Suppose that we are constructing a FIXP-circuit with input x from which we have obtained parameters A, b, c, R specifying a linear program:

$$\begin{aligned} \text{maximize} \quad & c^T y \\ \text{subject to} \quad & Ay \leq b \\ & y \in [-R, R]^n \end{aligned} \tag{2.3}$$

For instance, in the case of markets, the matrix A could specify prices of goods and b could be the budgets of the agents. To compute a solution, we add auxiliary variables from $[0,1]^n$ and scale it to a vector $x \in [-R,R]^n$. In order to check if z is feasible, we add m auxiliary variables to compute $\mu_i := H(a_i^T x - b_i)$ using our Heaviside gate. Subtracting $\sum_{i=1}^m \mu_i a_i$ from x has the effect of moving x closer to the set of feasible solutions. To ensure that an optimal solution is computed, we also add $\mu_0 c$ to x, where $\mu_0 = 1 - \max\{\mu_i \colon 1 \leq i \leq m\}$, which has the effect of increasing the objective value. One then projects x back onto the cube $z := \Pi_R\big(x + \mu_0 c - \sum_{i=1}^m \mu_i a_i\big) \in [-R,R]^n$ and scales it back into $y' = (z+R)/2R \in [0,1]^n$. The auxiliary variables being fixed essentially implies that $\mu_0 c - \sum_{i=1}^m \mu_i a_i = 0$, which under certain assumptions on the feasible domain implies that z is an optimal solution.

A similar construction works for LinearFIXP. However, there is one major obstacle which is the lack of a multiplication gate, meaning that we cannot compute the products $\mu_0 c$ and $\mu_i a_i$. For the constraints we handle this by assuming that A is constant. This, however, would be too restrictive for the objective function, and circumventing this problem requires more work. We refer to chapter 4 for the details.

2.2.3 Applications and comparison to previous work

Applying our pseudogate for computing optimal solutions to optimization problems, we are able to recover, and in some cases extend, a wide array of FIXP- and PPAD-membership results. Especially in the case of games and markets, it is clear how to apply the OPT-gate. For instance, in the case of games, one may under certain conditions construct a circuit which when given a strategy profile as input computes a best response mixed strategy for each individual player. The circuit then outputs this tuple of best responses, ensuring that fixed points correspond to equilibria. This construction illustrates a typical feature of applying the OPT-gate in that it essentially mimics Nash's original simple proof of his theorem using Kakutani's theorem [177].

Our construction also has less obvious applications. Note that a network flow problem may be formulated as a linear program. Note also that in the cake cutting problem, a division of the cake is envy-free precisely if the bipartite graph with edges between agents and preferred pieces admit a perfect matching. Using these two observations in conjunction, one may, again under various conditions, construct circuits whose fixed points correspond to envy-free divisions of CakeCutting instances. Our proof makes crucial use of a theorem due to Hall [134], also used by Woodall [228] in his proof of the existence of an envy-free division of a cake.

However, there are notable limitations to our approach, especially for PPAD-membership results. While we do recover the PPAD-membership of 2-player Nash, this is not the case for the 3-player case. Indeed, in 3-player Nash, the coefficients in the best-response LP for any one agent will be quadratic expressions in the mixed strategies of the remaining agents. Hence, a Linear-FIXP-circuit cannot make the computations required by the construction described in Section 3.3.2. This is no accident: whereas 2-player Nash is guaranteed to have rational solutions (the input describes a game where all the utilities are rational numbers), all solutions of a 3-player game may be irrational [175]. Instead Daskalakis et al. [67] proved that ε-approximate n-player Nash, which is the problem of computing a strategy profile where no player can increase his payoff by more than ε by unilaterally changing strategy, is PPAD-complete for $n \geq 3$. Similar concessions have to be made for many problems where the existence of a rational solution is not guaranteed. An interesting questions is whether one may construct some sort of approximate OPT-gate capturing such problems.

2.3 Consensus Halving and Borsuk-Ulam

This section contains a description of our paper [22]. We consider the consensus halving problem of fairly dividing a resource, modeled by the unit interval $A = [0, 1]$, into two equally valued pieces. Specifically, each of n agents has a continuous measure μ_i on A, and the goal is to partition A into two pieces $A = A^+ \sqcup A^-$ such that $\mu_i(A^+) = \mu_i(A^-)$ for all i. Using the Borsuk-Ulam theorem Simmons and Su [201] proved that a consensus halving exists. Recall:

Theorem 2.3.1 (Borsuk-Ulam [34])**.** *Let S^n and B^n denote the unit sphere and unit ball with respect to some norm p-norm, $p \geq 1$. The following statements hold:*

1. *If $F\colon S^n \to \mathbb{R}^n$ is continuous there exists $x \in S^n$ such that $F(x) = F(-x)$.*
2. *If $H\colon B^n \to \mathbb{R}^n$ is continuous and odd on ∂B^n there exists $x \in B^n$ with $H(x) = 0$.*

More than just establishing existence of a consensus halving, their proof also shows that only n cuts is required, dividing the resource into $n+1$ pieces each having a label indicating if it belongs to A^+ or A^-. In their proof, it is useful to restrict our attention to spheres with respect to the 1-norm $S_1^n = \{x \in \mathbb{R}^{n+1} : \sum_{i=1}^{n+1} |x_i| = 1\}$. They interpret a point $x \in S_1^n$ as a partitioning of A into two parts by letting $|x_i|$ be the length and $\mathrm{sgn}(x_i)$ the label of the ith piece. For instance, the point $x = (0.2, -0.7, 0.1)$ corresponds to a partitioning into the pieces $A^+(x) = [0, 0.2) \cup [0.9, 1]$ and $A^-(x) = [0.2, 0.9)$. Observe that $-x$ corresponds to the same partitioning, except for the labels being reversed. That is, $A^+(-x) = A^-(x)$. Upon defining $f : S_1^n \to \mathbb{R}^n$ by $f(x) = (\mu_1(A^+(x)), \dots, \mu_n(A^+(x)))$, one now sees that a point x satisfying the equality $f(x) = f(-x)$ defines a consensus halving. Indeed, if $f(x) = f(-x)$, then it holds that $\mu_i(A^+(x)) = \mu_i(A^+(-x)) = \mu_i(A^-(x))$ for all i. Hence, the Borsuk-Ulam theorem applies to show the existence of a consensus halving using only n cuts. Considering the case where agent i only has positive measure in the interval $((i-1)/n, i/n)$, $i = 1, \dots, n$, one sees that there does exist examples where n cuts are required.

As with Brouwer's fixed point theorem having a discrete analogue in Sperner's lemma, the Borsuk-Ulam theorem also has an analogue known as Tucker's lemma. The latter gives rise to a computational problem which is complete for the complexity class PPA contained in TFNP. Using Tucker's lemma, one may compute an approximate solution to the consensus halving problem where $|\mu_i(A^+) - \mu_i(A^-)| \leq \varepsilon$ for some small $\varepsilon > 0$. Filos-Ratsikas and Goldberg [94] and Filos-Ratsikas and Goldberg [95] proved that the problem of computing such an approximate consensus halving is a complete problem for PPA. They also used this to show PPA-completeness of other problems such as, for instance, the ham sandwich theorem, which can be proven by an application of Borsuk-Ulam. Deligkas et al. [71] introduced the complexity class BU in order to capture total real-valued search problems whose totality is guaranteed by the Borsuk-Ulam theorem. They showed that their class contains the exact version of the consensus halving problem CH. Inspired by the argument of Filos-Ratsikas et al. [98] that the ε-CH problem is PPAD-hard, they also proved that CH is hard for FIXP, but left open the question whether it is complete for BU. The main result of our paper is that the problem of finding a *strong approximate* solution to the CH problem is complete for the strong approximate version of the class BU. We describe the meaning of this in the following subsection.

2.3.1 Borsuk-Ulam, Consensus-Halving and Strong Approximation

In this section we briefly introduce the computational classes, problems and solution concepts central to our results. We define two complexity classes BU and BBU corresponding to the two formulations of the Borsuk-Ulam theorem. As is the case for FIXP, the classes are defined as consisting of all real-valued search problems that reduce to canonical problems under some suitable choice of reduction.

Definition 2.3.1. A real-valued search problem Π is a *basic* BU-problem if an instance I describes an algebraic circuit computing a continuous map $F_I : S_1^{d_I} \to \mathbb{R}^{d_I}$ such that the solution set $\mathrm{Sol}(I) = \{x \in S_1^{d_I} \colon F_I(x) = F_I(-x)\}$. Similarly, Π is a *basic* BBU-problem if an instance I describes an algebraic circuit forming a continuous map $F_I \colon B_1^{d_I} \to \mathbb{R}^{d_I}$ that is odd on $\partial B_1^{d_I}$ such that $\mathrm{Sol}(I) = \{x \in B_1^{d_I} \colon F_I(x) = 0\}$.

Before specifying our choice of reductions, let us define the computational search problem CH of finding a consensus halving. An instance I consists of a list of $\{+,-,*,\div,\max,\min\}$-circuits $C_1,\dots,C_n$ computing cumulative distribution functions $F_1,\dots,F_n$ on A. Inspired by Simmons and Su, the search space is $D_I = S_1^n$, and the set of solutions $\mathrm{Sol}(I)$ consists of all $x \in S_1^n$ satisfying, for every i,

$$\sum_{j\colon x_j>0} F_i(t_j) - F_i(t_{j-1}) = \sum_{j\colon x_j<0} F_i(t_j) - F_i(t_{j-1})$$

where $t_0 = 0$ and $t_j = t_{j-1} + |x_j|$ for $j = 1,\dots,n+1$. Because we have chosen the domains of the basic BU-problems to be spheres with respect to the 1-norm, it is now easy to see that CH belongs to BU by following the proof of Simmons and Su [201], even for a very limited class of reductions. Deligkas et al. [71] defined the consensus halving problem slightly differently. For them a solution to the CH problem is given as a vector $(t_1,\dots,t_k)$ of cuts points dividing A into pieces $[t_i,t_{i+1})$ of alternating labels. Thus, because successive coordinates in a solution to the Borsuk-Ulam problem may have equal sign, one has to be able to detect the sign of a real number when recovering a solution to the consensus halving problem in their phrasing. As such, they allowed for the use of a discontinuous comparison gate, which outputs 1 if its input is strictly positive and 0 otherwise, in their solution mapping. In our phrasing, this type of gate is not needed. Instead, we consider what we call *PL-reductions*. A PL-reduction from Π to Γ is a many-one reduction (f,g) from Π to Γ satisfying the added property that for any instance I of Π one may in polynomial time in $|I|$ construct a circuit with gates in $\{+,*\zeta,\max\}$ (where ζ represents rational constants) that computes the function $g(I,\cdot)$. This notion of reductions is stronger than SL-reductions used for the class FIXP. However, it maintains the property that for any instance I of Π the solution mapping $g(I,\cdot)$ is Lipschitz continuous with a Lipschitz constant that is of polynomial bit length in the size of I. The class BU (BBU) now consists of all real valued search problems that PL-reduce to a basic BU (BBU) problem.

A real valued search problem Π has an associated *strong approximation* version Π_a. An instance of Π_a consists of a pair (I,ε) where I is an instance of Π and $\varepsilon > 0$ is a rational number. The search space remains D_I, but the set of solutions becomes $\mathrm{Sol}(I,\varepsilon) = \{x \in D_I \colon \exists x^* \in \mathrm{Sol}(I), \|x - x^*\|_\infty \le \varepsilon\}$. We may consider Π_a to be a discrete search problem by letting the search space be given by the vertices of a sufficiently fine triangulation of D_I. Note that if $\Pi \le \Gamma$ under PL-reductions then $\Pi_a \le \Gamma_a$ under PL-reductions. Indeed, let (f,g) be the PL-reduction from Π to Γ. As we noted above, for any instance I of Π, there exists a Lipschitz constant L of polynomial bit size in $|I|$. Define the instance mapping from Π_a to Γ_a by setting $\tilde{f}(I,\varepsilon) = (f(I),\varepsilon/L)$ and the solution mapping by $\tilde{g}((I,\varepsilon),y) = g(I,y)$. Now if $y \in \mathrm{Sol}(\tilde{f}(I)) = \mathrm{Sol}(f(I),\varepsilon/L)$, then there exists a $y^* \in \mathrm{Sol}(f(I))$ such that $\|y - y^*\|_\infty \le \varepsilon/L$. Because $y^* \in \mathrm{Sol}(f(I))$, we have that $g(I, y^*) \in \mathrm{Sol}(I)$. Further, $\|g(I,y) - g(I,y^*)\| \leq L\|y - y^*\|_\infty \leq L \cdot \frac{\varepsilon}{L} = \varepsilon$. We conclude that $g(I,y) \in \mathrm{Sol}(I,\varepsilon)$. Hence, $(\tilde{f}, \tilde{g})$ constitutes a reduction from Π_a to Γ_a.

We may now state our main result:

Theorem 2.3.2. CH_a *is* BU_a*-complete.*

By the discussion above, the membership result follows immediately from the containment of CH in BU. Now, we will give a brief overview of the proof of hardness.

2.3.2 Description of the Reduction

Instead of reducing from a basic ℓ_1-BU_a problem, it suffices to reduce from a basic ℓ_∞-BBU_a problem. The proof of this can be found in the full paper. Suppose then that we are given an instance of an ℓ_∞-BBU_a problem as a circuit computing a continuous map $H \colon B^n_\infty \to \mathbb{R}^n$ that is odd on the boundary ∂B^n_∞ and an approximation parameter $\varepsilon > 0$. One now has to map this to a CH_a-problem whose solutions correspond to strong approximations of zeros of H. Before delving further into the details, we give a brief overview of the reduction:

1. Construct a consensus halving instance such that an exact solution z^* encodes a vector $x^* \in B^n_\infty$ such that $\|H(x^*)\|_\infty \leq \delta$ for some very small $\delta > 0$. The number δ will be small enough that the inequality $\|H(x^*)\|_\infty \leq \delta$ implies the existence of some $x^{**} \in B^n_\infty$ such that $H(x^{**}) = 0$ and $\|x^* - x^{**}\|_\infty \leq \varepsilon/2$.

2. Pick $\varepsilon' > 0$ such that if some cuts $z, z^* \in S^m_1$ satisfy the inequality $\|z - z^*\|_\infty \leq \varepsilon'$, then the vectors $x, x^* \in B^n_\infty$ that they encode satisfy $\|x - x^*\|_\infty \leq \varepsilon/2$.

3. Apply the triangle inequality to show that a solution z to the CH_a instance just described encodes a vector x that is an ε-strong approximation of a zero of H.

We now describe the consensus halving instance of step (i) that should, in some sense, simulate the map H. The paper of Filos-Ratsikas et al. [99] gives a simplified proof of PPA-hardness for the problem of computing a weak approximate consensus halving, and our construction is inspired by theirs.

We describe the consensus halving instance as being defined on an interval $[0, M]$, where M is a polynomially large integer, which may then be scaled to $[0,1]$. At the very left of the interval $[0, M]$ are placed n unit intervals which for any cut encodes a vector $x \in B^n_\infty$. Specifically, given some cut z, the interval $I_i := [i-1, i]$ encodes the ith coordinate of a vector x as $x_i := \mu(I_i^+) - \mu(I_i^-)$, where μ denotes the Lebesgue measure and $I_i^\pm$ denotes the subset of I_i with the corresponding label. We call this the *label encoding*. After that we will have $2n + 1$ *circuit simulation regions*. Each of these simulators will first read in x or $-x$, depending on the label immediately before the simulator, and then simulate the computation of H on this input. The output of every gate will be represented by a cut in an interval.

Let us briefly describe figure 2.3 below. Assume that the value encoded in the coordinate encoding region is $x_1 = 0.8$, meaning that $\mu(I_1^+) = 0.9$ and $\mu(I_1^-) = 0.1$.

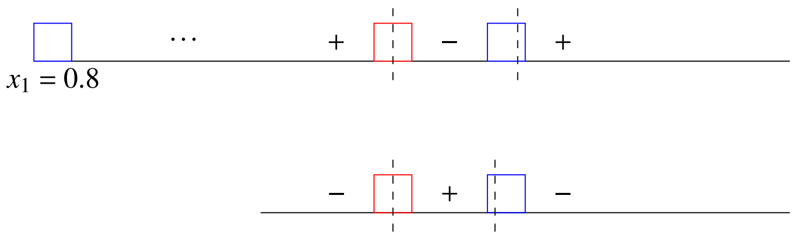


Figure 2.3: Reading in a value from the coordinate encoding region.

Every circuit simulator has one agent that reads in this value. We simulate this by constructing an agent that has a measure with density indicated by the blue blocks above. In front of every circuit simulator, we place an agent that has positive measure corresponding to the red block in the figure. The purpose of this agent is flipping the label. Now, if the label before the circuit simulator is $+$, the agent with the blue blocks will force a cut in the point 0.9 of the second blue block. We interpret this as encoding the value $0.9 - 0.1 = 0.8 = x_1$, the length of the interval before the cut minus the length of the interval after the cut. On the other hand, if the label in front of the circuit simulator is $-$, then the blue agent will force a cut in the point 0.1 of the second blue block. We interpret this as encoding the value $0.1 - 0.9 = -0.8 = -x_1$. We call this the *position encoding*. Using this encoding, we may construct agents that simulate all gates required for computing H. Here it is important that for every agent the labeling sequence of the input and output interval are identical. This may be ensured by placing agents with measures consisting of a single block. For the construction of agents implementing the required gates, we refer to the full paper.

However, it does not suffice that the CH-instance we construct is able to simulate H. Recall that we want an exact consensus halving z^* to encode a vector $x^* \in B^n_\infty$ such that x^* is $\varepsilon/2$-close to a zero of H. For this it was sufficient that $\|H(x)\|_\infty \leq \delta$ for some very small $\delta > 0$. In fact, one may show that it is sufficient that $\delta \leq \varepsilon^{-2^{p(|H|)}}$ where $p(|H|)$ is some polynomial in the size of the circuit computing H. Hence, a sufficiently small δ may be computed by a circuit of size polynomial in the bit length of ε and the size of H via repeated squaring. Thus, for every coordinate i, we may construct a *feedback agent* f_i that has a very narrow *Dirac block* of width δ centered in the ith output interval of every circuit simulator. All other agents will force a cut in a particular interval, but this is not the case for the feedback agents. Ideally, all of them will place their cut in the coordinate encoding region. In this case, all the circuit simulators will read in the same x^* and output $H(x^*)$ into the feedback region. In order for the feedback agents to agree that A^+ and A^- have the same value, this forces the cuts to be in the centers of the output intervals, meaning that $H_i(x^*) = 1/2 - 1/2 = 0$ for all i. However, some circuit simulators may produce *stray cuts*, i.e., cuts not in the coordinate encoding region. Each such stray cut may corrupt one circuit simulator, meaning that at most n circuit simulators can become corrupted, leaving $n+1$ that work as intended. Furthermore, if there is a stray cut, then at least one of the unit intervals in the coordinate encoding region does not contain a cut, meaning that $\|x^*\|_\infty = 1$.

Hence, the boundary condition on H implies that $H(x^*) = -H(-x^*)$. Suppose toward contradiction that there exists an i such that $|H_i(x^*)| > \delta$. If we now consider some uncorrupted circuit simulator where the preceding label is positive, then it will read in x^* and output the value $H_i(x^*)$ into the feedback region corresponding to the ith output. Because $H_i(x^*) > \delta$, all of the Dirac block will be contained in the part with negative label. This is also the case if the label preceding the circuit simulator is negative. Hence, in all of the at least $n + 1$ uncorrupted circuit simulators all of the Dirac block corresponding to the ith feedback agent will receive negative label. However, this implies that z^* cannot possibly be a consensus halving. Hence, if z^* is a consensus halving, then $||H(x^*)||_\infty \leq \delta$. This finishes the argument that CH$_a$ is BU$_a$-complete.

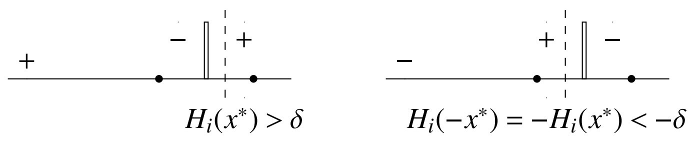


Figure 2.4: Illustration of the argument above. Taken from Batziou et al. [23].

2.3.3 Related work

As mentioned, Deligkas et al. [72] prove that CH is FIXP-hard. Central to their proof is also the construction of a consensus halving instance simulating an algebraic circuit. For FIXP, however, a solution is an input vector which is fixed by the circuit. As such, the feedback agents are constructed so as to force a cut in the coordinate encoding region. In the case of BU we are not looking for a fixed point, but instead a zero that is seemingly unrelated to the input. As such we cannot force the feedback agents to place their cuts in the coordinate encoding region, leading to a possibility of them corrupting the circuit simulators. As such it would seem that a different approach is required. Another possible direction would be to consider other search problems related to the Borsuk-Ulam theorem. For instance, Filos-Ratsikas and Goldberg [95] prove that a discrete version of the ham sandwich problem is complete for PPA, and Deligkas et al. [75] proved that problems related to sharing a pizza is contained in BU and PPA-hard, leaving open the question of BU-hardness for the exact version.

2.4 Fair Division of Indivisible Items

In this section we introduce the background and discuss the contributions of our paper [130]. In contrast with problems discussed previously, such as cake cutting and consensus halving, we will now be considering fair division of *indivisible* items.

Model. The problem that we consider is given by a set of indivisible goods M and a set of agents N. Every agent $i \in N$ has a utility function $v_i : 2^M \to \mathbb{R}_{\geq 0}$ that is assumed to be normalized, i.e., $v_i(\emptyset) = 0$, and monotone, i.e., $v_i(S) \leq v_i(T)$ for every pair of subsets such that $S \subseteq T \subseteq M$. Our results classify the the hardness of computing a fair allocation in terms of the strength of the utility functions. We recall some of the, for our purposes, most relevant valuation classes below:

1. A utility function $v\colon 2^M \to \mathbb{R}_{\geq 0}$ is *additive* if for every subset $S \subseteq M$ the following equality holds: $v(S) = \sum_{g \in S} v(\{g\})$.

2. Before defining the notion of *gross substitutes*, we introduce some notation. For a utility function $v\colon 2^M \to \mathbb{R}_{\geq 0}$ and any *price vector* $p \in \mathbb{R}^M_{\geq 0}$ define the quasi-linear utility function v_p by $v_p(S) = v(S) - \sum_{g \in S} p_g$ and define the demand set $D(v, p) = \arg\max_{S \subseteq M} v_p(S)$. The utility function v satisfies the *gross substitutes property* if for all $p \leq p'$ (the inequality being coordinate-wise), it holds that if $S \in D(v, p)$, then there exists $S' \in D(v, p')$ such that $\{g \in S : p_g = p_{g'}\} \subseteq S'$.

3. A utility function $v\colon 2^M \to \mathbb{R}_{\geq 0}$ is *submodular* if for every pair of subsets $S \subseteq T \subseteq M$ and all $g \in M$ it holds that $v(T \cup \{g\})) - v(T) \leq v(S \cup \{g\})) - v(S)$.

Submodular valuations may be thought of as satisfying the law of diminishing returns. Gross substitutes were introduced by Kelso and Crawford [152] in their study of matching firms and workers. One interesting feature of GS valuations is that they ensure the existence of a Walrasian equilibrium. The basic idea is to run the tâtonnement process due to Walras [225]. At first, all prices are set to zero and all the goods are allocated to one person. The agents are then allowed to take turns choosing a bundle from their demand set, and the prices of overdemanded goods are increased. The GS property is defined so as to ensure that all goods will be allocated at the end of this process. We refer to Paes Leme [180] for a nice exposition. It should also be noted that the GS property is in some sense necessary for ensuring the existence of a Walrasian equilibrium by theorem 2 of Gul and Stacchetti [133].

Fairness notions. An *allocation* is a partitioning $M = A_1 \sqcup A_2 \sqcup \cdots \sqcup A_n$ of the goods into n subsets, where player i receives the *bundle* A_i. One notion of fairness, analogous to the cake cutting setting, is the following: an allocation $(A_1, \ldots, A_n)$ is *envy free* if for every $i, j \in N$ it holds that $v_i(A_i) \geq v_i(A_j)$. However, it is easy to see that an envy-free division need not exist. For instance, in the case of two agents and a single good, it is impossible to devise an envy free division under the assumption that both agents value the good positively. Bouveret and Lang [36] proved that the problem of deciding if a given instance admits an envy free division is NP-complete. Indeed, one may reduce the Partition problem to the problem of deciding whether an envy free division exists for two agents with identical additive valuations.

As a result, various weaker notions of fairness have been studied. For instance, Budish [40] introduced the notion of *envy being bounded by a single good*, now commonly referred to as *envy-freeness up to one good (EF1)*. An allocation is said to be EF1 if for any pair of agents i and j, if i envies the bundle of j, then there should exist a $g \in A_j$ such that $v_i(A_i) \geq v_i(A_j \setminus \{g\})$. Such an allocation is guaranteed to exist, and it can be computed in polynomial time. Namely, using the envy-cycle elimination algorithm of Lipton et al. [163] one may maintain a partial EF1 allocation such that at least one agent is not envied by anyone. One then adds a good to the bundle of this agent, eliminates potential envy cycles, and continues. The fairness notion which we will consider is similar to but stronger than EF1. Instead of envy being bounded by one item (which could potentially be very valuable), we study a fairness notion where envy is required to be bounded by the removal of *any* item. Gourvès et al. [131] introduced the notion of *near jealousy-freeness*, which is identical to the following fairness notion defined by Caragiannis et al. [43]. An allocation $(A_1, \dots, A_n)$ is *envy-free up to any good (EFX)* if for any pair of players i and j it holds that $v_i(A_i) \geq v_i(A_j \setminus \{g\})$ for all $g \in A_j$. If this is not satisfied for some agents i and j, then we say that i EFX-envies agent j. In this paper, we study the computational hardness of finding an EFX allocation. We characterize the difficulty of computing EFX allocations for two agents in terms of the generality of their valuation functions by establishing the point at which the problem becomes hard in the standard complement-free hierarchy

$$\textit{additive} \subseteq \textit{OXS} \subseteq \textit{gross substitutes} \subseteq \textit{submodular} \subseteq \textit{XOS} \subseteq \textit{subadditive}$$

of Lehmann et al. [159]. Specifically, we provide a greedy polynomial-time algorithm for GS utility functions and prove that the problem is PLS-hard for submodular valuations. These results are described in the following two subsections.

2.4.1 A polynomial time algorithm

Plaut and Roughgarden [185] introduced a total ordering, the *leximin++* ordering, on the set of allocations and proved that when all the agents have identical monotone valuations, the maximum with respect to this ordering is an EFX-allocation. However, this does not provide an efficient algorithm for finding an EFX-allocation, because computing the optimal leximin++ allocation is NP-hard. Again, one may see this by reducing from the Partition problem.

We now describe our algorithm in the case of identical additive valuation functions. Assume that the goods $g_1, \dots, g_m$ are sorted in order of decreasing utility, and initialize a partial allocation with $A_i = \emptyset$ for all $i \in N$. The algorithm proceeds in m rounds, where in the jth round, the good g_j is allocated to one of the agents that currently has minimal utility. One may show that after any round, the allocation is a partial EFX-allocation. Indeed, this holds true in the beginning as the bundle of every agent is empty. Suppose then that the current partial allocation is EFX and that agent i receives good g_j in the current round. By definition of the algorithm, this means that no agent envies agent i after removing g_j from its new bundle. Furthermore, if another good from the bundle of agent i is removed, then this good must be at least as valuable as g_j due to the goods being allocated in order of decreasing utility. We conclude that the allocation obtained after adding g_j to the bundle A_i is still EFX. Hence, after the m rounds have concluded, the algorithm will output an EFX-allocation. Essentially, the important property of additive valuations that makes this proof work is that the problem of computing an optimal bundle of a given size can be solved by the greedy algorithm. In the full paper we prove the following result:

Definition 2.4.1. A valuation function $v\colon 2^M \to \mathbb{R}_{\geq 0}$ is *weakly well-layered* if for any $M' \subseteq M$ the sets $S_0, S_1, S_2, \ldots$ obtained via the greedy algorithm (that is, $S_0 = \emptyset$ and $S_i = S_{i-1} \cup \{x_i\}$ where $x_i \in \arg\max_{x \in M' \setminus S_{i-1}} v(S_{i-1} \cup \{x\})$ for $1 \leq i \leq M$) are optimal in the sense that $v(S_i) = \max_{S \subseteq M'\colon |S|=i} v(S)$ for every $i \leq |M'|$.

Theorem 2.4.1. *If the agents have a common valuation function v which is weakly well-layered, then one may compute an EFX-allocation in polynomial time.*

Note that our algorithm does not produce the leximin++ solution. Indeed, consider the following example of an additive valuation with five goods a,b,c,d,e having values $5,4,3,2,2$. Our greedy algorithm could produce the allocation $(\{a,d,e\},\{b,c\})$, whereas the maximum leximin++ allocation is $(\{b,d,e\},\{a,c\})$. The exact definition of the leximin++ ordering is given in the next subsection.

Earlier, we mentioned that gross substitutes is the largest class of valuation functions guaranteeing the existence of a Walrasian equilibrium. One might wonder if a similar statement can be made in terms of weakly well-layered valuations and EFX-allocations. However, it is unclear exactly how this should be phrased, seeing as an EFX allocation is automatically guaranteed to exist when the number of agents exceeds the number of goods.

2.4.2 PLS-completeness

The leximin++ ordering of Plaut and Roughgarden [185] is defined as follows. For any allocation $A = (A_1, \ldots, A_n)$, let π_A be a permutation on $\{1, \ldots, n\}$ ordering the bundles by increasing utility (if bundle i and j have the same utility and $i < j$ then choose π_A such that $\pi_A(i) < \pi_A(j)$). Define

$$\mathrm{L}(A) = (v(\pi_A(1)), |X_{\pi_A(1)}|, \ldots, v(\pi_A(n)), |X_{\pi_A(n)}|) \tag{2.4}$$

The leximin++ ordering is now defined by saying that $A \prec_{++} A'$ if $\mathrm{L}(A) \prec \mathrm{L}(A')$ where $\prec$ refers to the lexicographic ordering on $\mathbb{R}^{2n}_{\geq 0}$. That is to say, when comparing two allocations one first compares the utility of their least valued bundles, then the size of their least valued bundles, then the utility of their second to least valued bundles, then the size of their second to least valued bundles, and so on. Plaut and Roughgarden show that if A is not an EFX-allocation, then it is possible to construct an allocation A' from A by moving a single good from one bundle to another such that $A \prec_{++} A'$. More than just proving existence, this essentially shows that the problem of computing an EFX-allocation for identical agents is contained in PLS.

Our proof that this problem is also PLS-hard is more involved. Plaut and Roughgarden gave an exponential query lower bound by reducing the problem of local optimization on an odd Kneser graph to the problem of finding an EFX allocation for two identical agents with a submodular valuation. Therefore, it suffices to argue that local optimization on an odd Kneser graph is PLS-hard. This is shown by reducing from the Min-Circuit/Flip problem. An instance of this problem consists of a Boolean circuit C with n input-nodes and m output-nodes defining a function

$f: \{0,1\}^n \to \mathbb{N}$ by interpreting the output-nodes as the bit representation of a number. The domain of such an instance is $D = \{0,1\}^n$, and the neighborhood of any $x \in D$ consists of all the Boolean vectors whose Hamming distance to x is one, i.e., $N(x) = \{x' \in D\colon \Delta(x,x') = 1\}$ where $\Delta(x,x') = \#\{i \in \{1,\ldots,n\}\colon x_i \neq x'_i\}$. That is, as indicated by the name Min-Circuit/Flip, $x' \in N(x)$ if x' can be obtained from x by flipping a single bit. A solution to the instance given by C is then an $x \in \{0,1\}^n$ such that $f(x) \leq f(x')$ for all $x' \in N(x)$. The Kneser problem is defined by a Boolean circuit C' having $2k+1$ input-nodes for some $k \in \mathbb{N}$ defining a function as above. The domain $D' = \{x \in \{0,1\}^{2k+1}\colon \Delta(x,\mathbf{0}) = k\}$ consists of Boolean vectors of Hamming weight k, and the neighborhood of any $x \in D$ is the set $N'(x) = \{x' \in D'\colon \forall i, x_i \cdot x'_i = 0\}$. The domain and neighborhood corresponds exactly to the odd Kneser graph $K(2k+1,k)$, which is a graph with one vertex for every size k subset of $\{1,2,\ldots,2k+1\}$ and edges between disjoint subsets.

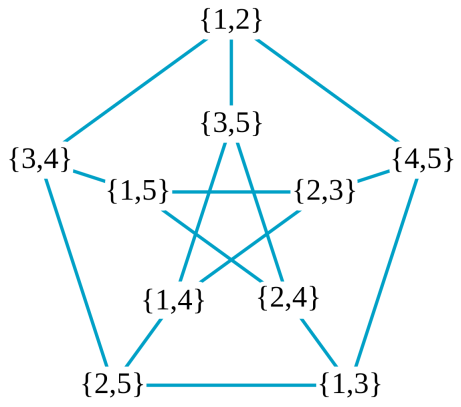


Figure 2.5: $K(5,2)$ taken from wikipedia.org/wiki/File:Kneser-5-2.svg.

When reducing from Min-Circuit/Flip to Kneser, it is natural to think of some $x \in \{0,1\}^n$ as the vertex $x\overline{x}0$ in the Kneser graph (where $\overline{x}$ is the bitwise complement). Note that if x and y are neighbors in the Min-Circuit/Flip instance, then there is a simple path from $x\overline{x}0$ to $y\overline{y}0$ in the corresponding Kneser instance. First, if we are in the case where y can be obtained from x by flipping one bit from 1 to 0, then the path is $x\overline{x}0 \to \overline{x}y1 \to y\overline{y}0$. In the other case x could be obtained from y by flipping a single 1, and a valid path would be the reverse of $y\overline{y}0 \to \overline{y}x1 \to x\overline{x}0$. For instance, supposing that $x = 111$ and $y = 110$, we have that $1110000 \to 0001101 \to 1100010$ is a valid path. One now has to define a cost function on the Kneser graph such that if y is an improving neighbor of x in the Min-Circuit/Flip instance, then there is an improving path from $x\overline{x}0$ to $y\overline{y}0$ in the Kneser instance. We refer to the full paper for the details.

2.4.3 Further related work

As explained above, Plaut and Roughgarden [185] proved existence of an EFX allocation in the case that all the agents share a common utility function. Later, Amanatidis et al. [6] gave a polynomial time algorithm for computing an EFX allocations for agents with additive valuations in two cases: (1) when the agents can only have one of only two possible values for any good, and (2) when for every agent, the utilities of all the goods are in the interval $[R_i, 2R_i]$ for some $R_i \geq 0$. Garg and Murhekar [111] later extended the former result to obtain an allocation that is both EFX and Pareto optimal in the case of bi-valued additive utility functions. Moving beyond two agents, Chaudhury et al. [45] proved that an EFX allocations exists for three agents with additive valuations. Their proof goes via a potential function, so it might be possible to obtain a PLS-membership result in this case. Of course, an intriguing open problem is proving existence of an EFX allocation for any number of agents, even for additive utility functions.

Part II

Publications

Chapter 3

FIXP-membership via Convex Optimization: Games, Cakes, and Markets

Abstract

We introduce a new technique for proving membership of problems in FIXP – the class capturing the complexity of computing a fixed-point of an algebraic circuit. Our technique constructs a “pseudogate” which can be used *as a black box* when building FIXP circuits. This pseudogate, which we term the “OPT-gate”, can solve most convex optimization problems. Using the OPT-gate, we prove *new* FIXP-membership results, and we *generalize* and *simplify* several known results from the literature on fair division, game theory and competitive markets.

In particular, we prove complexity results for two classic problems: computing a market equilibrium in the Arrow-Debreu model with general concave utilities is in FIXP, and computing an envy-free division of a cake with very general valuations is FIXP-complete. We further showcase the wide applicability of our technique, by using it to obtain simplified proofs and extensions of known FIXP-membership results for equilibrium computation for various types of strategic games, as well as the pseudomarket mechanism of Hylland and Zeckhauser.

3.1 Introduction

Equilibria, i.e., stable states of some dynamic process or environment [230], appear in several classic applications in economics and computer science. Prominent examples include the Nash equilibrium [177], which captures the stable outcome of deliberation between strategic agents, as well as the competitive equilibrium [8], which corresponds to a market-clearing outcome after the adjustment of prices based on demand and supply. These equilibria can most often be captured by fixed points of functions, i.e., points x for which $f(x) = x$. For instance, Nash’s existence theorem, i.e., that every strategic game has a mixed Nash equilibrium, was famously proven using Brouwer's fixed point theorem [39].

The computational class FIXP was defined by Etessami and Yannakakis [85] to capture the complexity of fixed point problems, and in particular those related to Brouwer's fixed point theorem. These problems are *total search problems*, i.e., problems for which a solution is guaranteed to exist via Brouwer's (or some other) fixed point theorem, and for which we aim to find such a solution. Indeed, the class has been successful in that regard, with interesting problems related to game theory [85] and competitive markets [56, 85, 118] among others, being either members of FIXP, or complete for the class.

At the heart of the definition of FIXP lies the notion of an algebraic circuit, used to represent a continuous function mapping a domain to itself. This representation effectively allows for the study of exact fixed points of the function, including irrational ones, and therefore can be used to capture the *exact* complexity of these types of equilibrium problems. In contrast, in the usual Turing model of computation, sometimes the best one can hope for is approximate solutions (e.g., ε-Nash equilibria). The counterpart of FIXP in the Turing model is the class PPAD of Papadimitriou [182] which famously captures the complexity of computing an ε-Nash equilibrium in strategic games [54, 67]. Indeed, for several of the aforementioned problems, computing approximate equilibria is in PPAD, whereas computing exact equilibria is in FIXP. Another interpretation of FIXP in the Turing model of computation is in terms of *strong approximations*, i.e., computing points that are close in the sense of distance (e.g., in the max norm) to equilibrium points. In contrast, PPAD typically captures weak approximations, i.e., points that are approximately equilibrium points, but not necessarily close to an exact equilibrium point in the geometric sense.

Contrary to the case of decision problems in NP, for which the membership in the class is often immediate, proving membership of a total search problem in the corresponding computational class is typically much more involved, and often requires "transforming" an existence proof into a computational reduction. This poses certain challenges, but it has been largely successful for problems in PPAD. For example, the PPAD-membership of Nash equilibrium computation incorporates Nash's existence proof (e.g., see [127, Section 3.2]), and the PPAD-membership of the approximately envy-free cake cutting problem [38, 76, 190] is essentially a modification of an existence proof due to Simmons [212].

In the case of FIXP however, the aforementioned challenges are much more pronounced; for an existence proof to be used as a basis for a membership result, it has to display several characteristics. First, it has to go via Brouwer's fixed point theorem, and more importantly, it has to avoid using any "discontinuous" components, precluding the use of several types of discrete steps and limit arguments. For this reason, FIXP-membership results tend to be much more ad-hoc, using inventive but often rather involved techniques, which do not necessarily follow the known existence proofs. Even worse, for certain problems like the envy-free cake cutting problem for instance, the literature has not managed to produce any FIXP-membership result for the reasons mentioned above.

A closer inspection into the several proofs of existence for versions of strategic games or competitive markets reveals that they often exhibit a common characteristic: they all include one or multiple optimization problems as subroutines. For example, at the heart of the Nash equilibrium notion is an agent's utility maximization problem, which can be expressed as a linear program (see Equation (3.1) in Section 3.3.1). Another example comes from competitive markets, where the market equilibrium notion includes convex optimization programs for maximizing the utilities of consumers and producers given a set of prices. This offers a possible explanation as to why the literature has fallen short of producing a systematic and unified approach for proving FIXP-membership results: Up until now, it was not known how to actually compute these optimization programs in FIXP, or more specifically, how to incorporate these programs as part of a FIXP circuit, as required for a membership result.

Our paper remedies this situation: We show how to compute convex optimization programs, which can be used as black-box components of FIXP circuits. Simply put, under some mild assumptions, whenever such an optimization program is encountered in an existence proof, it can be effectively substituted by such a component in the FIXP-membership proof. Using our newly introduced technique, we manage to generalize and simplify several FIXP-membership proofs in the literature of game theory and competitive markets, as well as prove for the first time the seemingly elusive FIXP-completeness result for the envy-free cake cutting problem. We present our contributions in more detail below.

3.1.1 Our Contribution

Our main contribution is the introduction of the *OPT-gate*, a new "plug and play" component which can be used as a black-box in FIXP-membership proofs for computing Linear Programs or more general convex optimization programs. The OPT-gate is a special kind of gate, with the following crucial property:

> *The OPT-gate is a "pseudogate", in the sense that its correct operation is only ensured at a fixed point of the function encoded by the algebraic circuit; with regards to a* FIXP*-membership proof, it operates as a normal gate for all intents and purposes.*

More specifically, the OPT-gate can solve any convex program with convex inequality constraints, explicit equality constraints and an explicit bound on its feasible region, as long as it satisfies a "FIXP-appropriate" variant of the well-known *Slater condition* [202] for convex programs (see Sections 3.3.3 and 3.3.4, and Definitions 3.3.3 and 3.3.4). Having programs of this form is in fact necessary (see the discussion in Section 3.3), but at the same time it is sufficient for capturing the rather general optimization problems that appear in the existence proofs mentioned above.

To demonstrate the effectiveness of our technique, we present a host of different applications related to the *envy-free cake cutting problem*, to computing different types of equilibria in various *strategic games*, and to computing competitive equilibria in *markets*. Our results advance the state of the art in three different ways: (a) we provide results for problems for which the complexity was previously entirely unknown, (b) we provide results that generalize known special cases in the literature to domains which are as general as possible, and (c) we provide proofs which are conceptually simpler and reminiscent of the known proofs of existence for those problems.

Applications to Game Theory

First, we discuss the application of our technique to the problem of computing exact equilibria in strategic games. Already in Section 3.3.1, we use the case of normal form games as a motivating example to demonstrate the strength of the OPT-gate. The FIXP-completeness of the problem was established by Etessami and Yannakakis [85], in the same paper where they defined the class FIXP. We show that via the employment of our technique, the membership problem essentially boils down to simply writing the standard utility-maximization linear programs for the players and substituting them by the OPT-gate in the FIXP circuit, making the proof entirely straightforward.

Then, in Section 3.4, we move on to present more general classes of games and different equilibrium concepts, for which we also obtain FIXP-completeness or FIXP-membership results. In particular:

- **FIXP-completeness of concave games.** In Section 3.4.1, we prove the FIXP-completeness of *concave games* [192], a class of games which generalizes the class of normal-form games. Rosen [192] showed via the employment of Kakutani's fixed point theorem [149] that a Nash equilibrium of these games always exists. Our FIXP-membership proof defines a Brouwer function that uses the agent's utility-maximization program, now a convex program, as a subroutine, substituted by the OPT-gate. Similarly to the case of normal form games described above, our proof is very simple and intuitive.

- **FIXP-membership of ε-proper equilibria.** In Section 3.4.2, we consider a Nash equilibrium refinement notion due to Myerson [174], that of an *ε-proper equilibrium.*[1] Hansen and Lund [135] showed that approximating a proper equilibrium (i.e., a limit point of ε-proper equilibria) is complete for FIXP$_a$ [85], the class of *discrete* total search problems that reduce to (strong) approximate Brouwer fixed points. We show that computing an ε-proper equilibrium is in FIXP. To obtain the result, we first develop a more general method based on solving *systems of conditional convex constraints* (see Section 3.4.2), making use of our OPT-gate, which might have applications beyond the ε-proper equilibrium result.

- **FIXP-completeness of *n*-player Stochastic Games**. In Section 3.4.3, we consider n-player stochastic games, which generalize the classic 2-player stochastic games of Shapley [200]. The existence of a *stationary λ-discounted equilibrium* for any discount factor λ was proven by Takahashi [214] and Fink [102] using a

[1]We remark here that the ε parameter is not the same type of approximation as in an ε-Nash equilibrium mentioned earlier; see Definition 3.4.1 and the "approximate" vs "almost" discussion in [84, Section 2].

generalization of Kakutani's fixed point theorem. For 2-player zero-sum games, Etessami and Yannakakis [85] showed that computing a *stationary λ-discounted equilibrium* is in FIXP. We generalize this membership result to n-player general stochastic games. Our proof is based on an enlarged domain of triples consisting of valuation profiles and pairs of stationary strategies, and constructs a Brouwer function from this domain to itself, for which the fixed points "contain" fixed points of the correspondence defined in Takahashi's proof on the original domain. The FIXP-hardness follows from [85], by noting that a normal form game may simply be viewed as a stochastic game with a single state.

Applications to Cake Cutting

Next, in Section 3.5, we prove our main result for the well-known envy-free cake cutting problem [108] (see also [38, 188, 190]). In this problem, the cake serves as a metaphor for a divisible resource, which needs to be divided fairly among a set of agents. The agents have different preferences over how to divide the resource, and an envy-free division is one which guarantees that each agent would rather have their own piece than any other agent's piece. The existence of an envy-free division was proven by Stromquist [209], even for the case where each agent receives a single piece (known as the *contiguous* version or the version with *connected pieces*). An alternative proof was provided by Simmons (cited in [212]). Both proofs employ a discretization of the space of possible divisions and then apply some topological lemma (either a variant of the K-K-M lemma [156] or Sperner's lemma [204]), together with a limit argument.

In terms of the complexity of the problem, results were only known for the approximate version of the problem: Deng et al. [76] proved that for agents with very general valuations, computing a contiguous envy-free division of the cake is PPAD-complete. Deng et al.'s proof closely follows Simmons' proof [212], which, without the limit argument, obtains the existence of an approximately envy-free division. However, before our paper, the complexity of the *exact* envy-free cake cutting problem was not known. To this end, we provide the following result.

The (contiguous) envy-free cake cutting problem with very general valuations is FIXP-*complete.*

By "very general valuations" we mean valuations that are not necessarily additive measures or even monotone over subsets of the cake, and which can assign different values to different divisions for an agent, even if the agent receives the same piece in all of those. The aforementioned existence proofs apply to this very general case as well, and therefore our FIXP-membership result is as strong as possible. We discuss this in more detail in Section 3.5 (see Remark 4).

In order to obtain the FIXP-membership result, we develop a new proof of existence for envy-free cake cutting, one which is not based on discretizations and limit arguments. Our proof constructs a bipartite graph between agents and preferred pieces and computes a maximum flow on this graph. This computation can be immediately substituted by our OPT-gate, effectively turning this new existence proof into a FIXP-membership result. This proof is somehow reminiscent of another existence proof by Woodall [229], but as we explain in Section 3.5, Woodall's proof uses discontinuous steps and therefore cannot conceivably be "turned" into a FIXP-membership proof.

For the FIXP-hardness, we construct a very simple reduction from a generalization of Brouwer's fixed point problem due to Bapat [19]. The very same reduction also shows that Bapat's Brouwer fixed point problem is in FIXP. This in turn has implications for the *rainbow* K-K-M problem [105], a generalization of the K-K-M problem [156], which we show to be FIXP-complete via reductions from and to Brouwer's fixed point problem. These results, which are included in Section 3.5.2, develop a potentially useful machinery for proving FIXP-completeness results for more general cake cutting and fair division problems. For example Aharoni et al. [3] establish the relation between K-K-M-type theorems and envy-free divisions of multiple cakes; whether these can yield FIXP-membership results for those problems as well is something to be explored in the future.

Applications to Markets

Our last application domain is that of competitive markets. Here we provide results for general *Arrow-Debreu markets* [8], as well as for the *pseudomarket mechanism* of Hylland and Zeckhauser [142].

- **Arrow-Debreu markets.** In Section 3.6.1 we prove a very general result, namely that computing competitive equilibria in Arrow-Debreu markets with concave utilities is in FIXP. The Arrow-Debreu market is the most fundamental market model, proposed and studied by Arrow and Debreu [8]. It consists of a set of consumers with utilities, consumption sets and endowments, and a set of producers or firms with production sets. A *competitive* or *market* equilibrium is a stable state in which supply equals demand, and all participants maximize their utilities or profits at the current set of prices. Arrow and Debreu [8] proved that under mild assumptions, every market has a competitive equilibrium.

  FIXP-membership results were only previously known for special cases of Arrow-Debreu markets. Etessami and Yannakakis [85] in their original paper already proved the FIXP-membership of a setting where there are no explicit utilities, and the aggregate demand is a given function, rather than a correspondence which is typically the case in these markets. Garg et al. [117] proved a FIXP-membership result for markets with *Piecewise Linear Concave (PLC)* utilities, straightforward consumption sets (i.e., where consumption is only constrained to be non-negative), and production sets that are also given by PLC functions.

  Our result for Arrow-Debreu markets generalizes[2] the aforementioned results as it considers (a) more general utility functions (i.e., general concave functions)

[2]To be precise, our result applies to any class of concave utility functions, as long as we have access to the supergradients of those functions or when we can compute them given access to the functions.

and (b) more general consumption and production sets (i.e., general convex sets). Additionally, compared to the proofs in these papers, our membership proof is arguably simpler and follows rather easily from the original existence proof of Arrow and Debreu. Essentially, the only difference is that we "organically" devise a Brouwer function rather than a fixed point correspondence, and we substitute the various convex optimization programs that appear in the proof (for the consumers' and producers' optimality) by our OPT-gate.

- **The pseudomarket mechanism of Hylland and Zeckhauser [142].** In Section 3.6.2 we consider the problem of computing equilibria of the pseudomarket mechanism of Hylland and Zeckhauser [142]. This mechanism solves the *random assignment problem* (e.g., see [32]) by allocating to each agent a unit of artificial currency, and by then setting up a "pseudomarket" where agents buy probability shares of the different items. The Hylland and Zeckhauser pseudomarket is not a special case of the Arrow-Debreu market, because of additional allocation constraints that ensure that each agent receives exactly one item in expectation. Hylland and Zeckhauser employed Kakutani's fixed point theorem to prove that an equilibrium of this market is always guaranteed to exist.

  The complexity of computing a Hylland and Zeckhauser equilibrium was an open problem since the definition of the mechanism in 1979 and certainly since the introduction of the relevant complexity classes for equilibrium computation problems. Very recently, Vazirani and Yannakakis [221] showed that the problem lies in FIXP, leaving the FIXP-hardness as an open question. We employ our OPT-gate to obtain the same membership result, via, what we believe to be, an easier proof. Again, like most of our results, the proof resembles strongly the existence proof of Hylland and Zeckhauser [142], except that it constructs a Brouwer fixed point function (rather than a Kakutani fixed point correspondence) and substitutes the agents' utility maximization Linear Programs by instances of the OPT-gate.

3.1.2 Related Work

Below we present some further related work related to our applications, as well as to fixed point computation problems.

Strategic games. The field of game theory was developed in the late 1920s by the works of von Neumann [223, 224] and then notably in the 1950s with the concept of Nash equilibrium, guaranteed to exist by Nash's theorem [177]. The theorem can be proven by either using Brouwer's fixed point theorem [176] or Kakutani's fixed point theorem [177]. The complexity of Nash equilibrium computation was firstly considered by Papadimitriou [182], who actually defined the class PPAD with this problem as the central consideration. More than a decade later, the celebrated results

This is possible for the PLC utilities of [117] as we explain in Section 3.7, but not for the CES utilities of [56], since these are non-superdifferentiable at 0 coordinates. See Remark 5 in Section 3.6.1 for more details.

of Daskalakis et al. [67] and Chen et al. [54] showed the PPAD-completeness of the approximate version of the problem, followed by the definition of FIXP and the FIXP-completeness result of Etessami and Yannakakis [85] for exact equilibria. Since then, several variants of the main normal form game setting and several refinements of the standard equilibrium notions have been considered, with corresponding complexity results being obtained (e.g., see [69, 70, 138, 194]). Out of these refinements, the most relevant to us is the notion of proper equilibria defined by [174]. These equilibria were studied by Hansen and Lund [135] as we explained above, and it was shown that approximating them is complete for the class FIXP$_a$, a discrete variant of FIXP also defined by Etessami and Yannakakis [85].

Stochastic games were defined by Shapley [200] in the early 1950s, and they constitute one of the most fundamental models of repeated games in the literature, which can capture very general scenarios; we refer the reader to [171, 178] for more details on different types of stochastic games and their definitions. Our FIXP-membership result establishes the FIXP-completeness of the problem for n-player games; for 2-player games, besides the FIXP-membership shown in [85], the authors also show that the problem is at least as hard as the Square Root Sum problem, defined therein; whether the 2-player problem is FIXP-complete is still an open question. Exponential or superexponential time algorithms for the 2-player problem were developed by Hansen et al. [139] and Oliu-Barton [179].

Cake cutting. The cake cutting problem was introduced by Steinhaus [206] in the late 1940s and has since been studied extensively in the literature of mathematics, economics and computer science. The problem of finding an envy-free division was introduced by Gamow and Stern [108] about a decade later. The existence of an envy-free division was shown in several proofs, but perhaps the most famous are those by Stromquist [209] and Simmons [212] that we mentioned earlier, which in fact guarantee the existence of contiguous divisions. The computational complexity of the approximate problem was considered by Deng et al. [76] who proved a PPAD-completeness result for the case of general continuous preferences, which is equivalent to the setting of very general valuations we consider here. As we explained earlier, our paper provides the first computational complexity results for the problem of finding an exact envy-free division. For the usual case of additive valuation functions (e.g., see [38, 190]), the FIXP-hardness for exact equilibria, or even the PPAD-hardness for approximate equilibria is still a major open problem.

In a related, but, in a sense, orthogonal line of work, several discrete protocols for finding an envy-free solution were proposed over the years, starting from the cut-and-choose protocol for 2 agents and the Selfridge-Conway protocol for 3 agents (e.g., see [190]), leading to recent breakthrough results from the literature of computer science [11]. These protocols interact with the agents via a set of queries, in the so-called Robertson-Webb (RW) model (see [227]). The RW model is not inherently a computational model, and RW queries can in fact return irrational points as answers. In that regime, the goal is to come up with a protocol that finds an envy-free solution using the smallest number of such queries possible. Even for the non-contiguous version, the discrepancy between the lower bound of Procaccia [187] and the upper bound of Aziz and Mackenzie [11] is astronomical.

Competitive Markets. The fundamental principles of competitive markets and equilibrium theory date back to the 1870s and the works of Walras [225]. Walras described a process of adjusting the market prices based on supply and demand, the so-called “tâtonnement process”, which would eventually lead to the stable outcome that was later known as the competitive equilibrium. Foundational in the establishment of the associated equilibrium theory were the contributions of Arrow and Debreu [8] and McKenzie [168],[3] who proved the existence of an equilibrium. The proof of Arrow and Debreu uses a fixed point theorem due to Debreu [68], whereas McKenzie used Kakutani’s fixed point theorem to obtain the result. An alternative proof via Brouwer’s fixed point theorem was given by Geanakoplos [122].

In computer science, much work has been devoted to the question of computing exact or approximate equilibria of different markets, which are special cases of the general Arrow-Debreu market that we study. There are several works that developed polynomial-time algorithms for finding or approximating equilibria for some classes of utility functions, e.g., see [77–79, 114, 116, 121, 143, 144]. For more complex utility functions, besides the results that we mentioned earlier, the approximate equilibrium computation problem for additively separable piecewise linear concave (SPLC) functions was shown to be PPAD-complete by [53, 220], where the approximation notion is a “weak approximation” in the market clearing and utility-maximization conditions, see [196]. For exact equilibria, Garg and Vazirani [113] showed the PPAD-completeness of Arrow-Debreu markets with linear utility functions and SPLC production sets; in this case, it turns out that there always exist rational exact equilibria, and they can be computed in PPAD. An interesting class of utility functions is that of Leontief utilities, which are simultaneously subcases of the PLC utilities studied in [117, 118] and limit cases of the CES utilities studied in [56]. For this class, Codenotti et al. [59] showed a PPAD-hardness result. Garg et al. [118] showed hardness results for a market model with Leontief utilities, but their FIXP-hardness does not quite yield a FIXP-completeness result together with our membership proof, or even the membership proof of Garg et al. [117], because it is obtained for markets with different sufficiency conditions for equilibrium existence, and not for the market model as presented by Arrow and Debreu [8] that we study in Section 4.6.

Fixed point computation. Besides the applications above, the class FIXP also captures the complexity of other problems, such as branching process and context-free grammars [85], equilibrium refinements [84, 87], and more recently the complexity of computing a Bayes-Nash equilibrium in the first-price auction with subjective priors [100]. Besides FIXP, there are some other computational classes that capture the

[3]In fact, sometimes the fundamental market model is referred to as the “Arrow-Debreu-McKenzie” market.

complexity of different fixed point problems, namely the classes BU [74] and BBU [23] which correspond to the Borsuk-Ulam theorem [34], and the class HB [128], which corresponds to the Hairy Ball theorem [186].

3.2 Preliminaries

In this section, we provide some definitions and theorems that we will use or reference throughout the paper, as well as the formal definition of the class FIXP.

3.2.1 Fixed Point Theorems

We start with the definition of *Brouwer's fixed point theorem* [39], one of the most widely used fixed point theorems in economic applications, such as game theory or market theory.

Theorem 3.2.1 (Brouwer's Fixed Point Theorem [39])**.** *Let $A \subseteq \mathbb{R}^n$ be a nonempty, compact, and convex set. Let $f\colon A \to A$ be continuous. Then there is $x \in A$ such that $f(x) = x$.*

The next fixed point theorem that we will present is *Kakutani's fixed point theorem* [149], a generalization of Brouwer's fixed point theorem. Importantly, this fixed point theorem applies to *correspondences* rather than functions; we provide the definition of a correspondence below.

Definition 3.2.1 (Correspondence)**.** A correspondence f (or multi-valued function) between sets A and B is a function $f\colon A \to \mathcal{P}(B)$, where $\mathcal{P}(B)$ denotes the powerset of B. We denote this by $f\colon A \rightrightarrows B$. In case $f(a) = \{b\}$ we use the function notation $f(a) = b$ for notational simplicity. In a similar way, if for all $a \in A$ we have $|f(a)| = 1$, we may think of f simply as a function $f\colon A \to B$.

For the statement of the theorem, we need the definitions of *upper* and *lower hemi-continuous* correspondences.

Definition 3.2.2 (Upper and lower hemicontinuous correspondence)**.** Let $A \subseteq \mathbb{R}^n$, $B \subseteq \mathbb{R}^m$, and $f\colon A \rightrightarrows B$.

1. f is upper hemicontinuous (uhc) at $a \in A$ if and only if for all open sets $V \subseteq B$ for which $f(a) \subseteq V$ there is an open set $U \subseteq A$ with $a \in U$ such that $f(x) \subseteq V$ for all $x \in U$.

2. f is lower hemicontinuous (lhc) at $a \in A$ if and only if for all open sets $V \subseteq B$ for which $f(a) \cap V \neq \emptyset$ there is an open set $U \subseteq A$ with $a \in U$ such that $f(x) \cap V \neq \emptyset$ for all $x \in U$.

We say that f is uhc (lhc) if f is uhc (lhc) at every $a \in A$. If f is both uhc and lhc we simply say that f is continuous.

We are now ready to state the fixed point theorem.

Theorem 3.2.2 (Kakutani's Fixed Point Theorem [149]). *Let $A \subseteq \mathbb{R}^n$ be a nonempty, compact, and convex set. Let $f\colon A \rightrightarrows A$ be uhc as well as nonempty-, compact- and convex-valued. Then there is $x \in A$ such that $x \in f(x)$.*

Kakutani's fixed point theorem is often used in conjunction with the following theorem, called *the maximum theorem*, proven in 1963 by Berge [25]. For this to be possible, it is additionally needed that the maximizer-correspondence g^* is convex-valued. This is in particular ensured if f is quasi-concave in its second variable and g is convex-valued.

Theorem 3.2.3 (Berge's Maximum Theorem [25]). *Let $A \subseteq \mathbb{R}^n$ and $B \subseteq \mathbb{R}^m$. Let $f\colon A \times B \to \mathbb{R}$ be continuous and $g\colon A \rightrightarrows B$ continuous as well as nonempty- and compact-valued. Define $f^*\colon A \to \mathbb{R}$ and $g^*\colon A \rightrightarrows B$ by $f^*(a) = \max_{b \in g(a)} f(a,b)$ and $g^*(a) = argmax_{b \in g(a)} f(a,b)$. Then f^* is continuous and g^* is uhc as well as nonempty- and compact-valued.*

3.2.2 The Class FIXP

As we said in the introduction, the class FIXP captures the complexity of real-valued search problems associated with Brouwer's fixed point theorem. We provide the formal definition of the class below.

A search problem Π with real-valued search space is defined by associating to any input instance I (encoded as a string over a finite alphabet Σ) a search space $D_I \subseteq \mathbb{R}^{d_I}$ and a set of solutions $\mathrm{Sol}(I)$. We assume there is a polynomial time algorithm that given I computes a description of D_I. In order to define FIXP, we first introduce a set of basic FIXP-problems corresponding to the formulation of Brouwer's fixed point theorem. Afterwards, we explain how the class is closed with respect to a certain type of reductions. We start with the definition of an algebraic circuit.

Definition 3.2.3 (Algebraic Circuit). An algebraic circuit C is a circuit using gates in $\{+,-,*,\div,\max,\min\}$ as well as rational constants. We let $\mathrm{size}(C)$ denote the size of the circuit, including the description of the rational constants.

Next, we define the notion of a *basic* FIXP *problem.*

Definition 3.2.4 (Basic FIXP problem). A search problem Π is a basic FIXP problem if every instance I describes a nonempty compact convex domain D_I described by a set of linear inequalities with rational coefficients and a continuous map $F_I\colon D_I \to D_I$ given by an algebraic circuit C_I,[4] and the solution set is $\mathrm{Sol}(I) = \{x \in D_I \mid F_I(x) = x\}$.

[4]Note that given an algebraic circuit, it is not clear how to check that it is indeed well-defined (i.e., does not divide by zero), and that it indeed represents a function F_I with $F_I(D_I) \subseteq D_I$. For that reason, we assume that it is promised that the algebraic circuit C_I indeed satisfies these two properties (in other words, we only consider instances where this is the case). Furthermore, note that as long as the circuit is well-defined, the function will be continuous, since all gates perform continuous operations.

We now discuss reductions between search problems. Let Π and Γ be search problems with real-valued search space. A *many-one reduction* from Π to Γ is a pair of maps (f,g). The instance mapping f maps instances I of Π to instances $f(I)$ of Γ, and for any solution $y \in \mathrm{Sol}(f(I))$ the solution mapping g maps the pair (I,y) to a solution $g(I,y) \in \mathrm{Sol}(I)$ of Π. In order to avoid meaningless reductions, it is required that $\mathrm{Sol}(f(I)) \neq \emptyset$ if $\mathrm{Sol}(I) \neq \emptyset$. We require that the instance mapping f is computable in polynomial time. Etessami and Yannakakis [85] defined the notion of *SL*-reductions where the solution mapping g is *separable linear*. This means there exists a map $\pi\colon \{1,\dots,d_I\} \to \{1,\dots,d_{f(I)}\}$ and rational constants a_i, b_i, $i = 1,\dots,d_I$, such that for $y \in \mathrm{Sol}(f(I))$ one has that $x = g(I,y)$ is given by $x_i = a_i y_{\pi(i)} + b_i$ for all i.[5] The map π and the constants a_i, b_i should be computable from I in polynomial time.

We are now ready to define the class FIXP.

Definition 3.2.5 (FIXP)**.** The class FIXP consists of all search problems with real-valued search space that SL-reduce to a basic FIXP problem for which the domain D_I is a convex polytope described by a set of linear inequalities with rational coefficients and the function F_I is defined by an algebraic circuit C_I.

In our definition of basic FIXP problems, we have assumed that the domains of the functions are polytopes. However, the definition of the class FIXP is robust to modifications of the definition of basic FIXP problems [85]. For example, the basic FIXP problems could have been defined as having for instance unit-balls B_p^d as domains, $p \in [1,\infty]$, and one could have allowed the functions F_I to be computed by circuits over $\{+,-,*,\div,\max,\min,\sqrt[k]{}\}$.

Note that if we only allow the gates $\{+,-,*c,\max,\min\}$ (where "$*c$" denotes multiplication by a constant) in the definition of FIXP, then we instead obtain the class Linear-FIXP. It is known that Linear-FIXP = PPAD [85]. In other words, the difference between FIXP and PPAD comes from the added power of the general multiplication gate.

3.3 The OPT-gate

In this section, we present a new technique for proving membership in FIXP. Namely, we introduce the OPT-gate: a gate that can essentially solve Linear Programs, under some minor conditions. This gate can be used like any other gate for the purpose of proving FIXP-membership. We begin with a motivating example in Section 3.3.1 and then present some obstacles to constructing the OPT-gate, which lead us to define the notion of a pseudogate in Section 3.3.2. Using this notion we show how to construct an OPT-gate that solves LPs (Section 3.3.3) and even more general convex programs (Section 3.3.4).

[5]This allows one, for example, to project away auxiliary variables.

3.3.1 A motivating example: Nash equilibrium computation

To provide an example of how the OPT-gate could be used, we consider the classical problem of computing a Nash equilibrium in a normal form game.

Normal form game. There are n players and every player $i \in [n] := \{1,2,\dots,n\}$ has a finite set of pure strategies $S_i = [m_i]$ and a payoff function $u_i \colon S \to \mathbb{R}$, where $S := S_1 \times \cdots \times S_n$. A *mixed strategy* of player i is a probability distribution on S_i. We let $\Sigma_i := \Delta(S_i)$ denote the set of all such distributions, i.e., $\Sigma_i := \{y \in \mathbb{R}^{m_i}_{\geq 0} : \sum_j y_j = 1\}$. In other words, Σ_i is the $(m_i - 1)$-dimensional unit simplex. A *mixed strategy profile* is a vector $x \in \Sigma := \Sigma_1 \times \cdots \times \Sigma_n$, where $x_i \in \Sigma_i$ is the mixed strategy played by player i in the strategy profile x. For $j \in S_i = [m_i]$, the jth coordinate of x_i is denoted $x_{i,j}$ and it corresponds to the probability that player i plays its pure strategy j. The payoff function u_i can be extended to all mixed strategy profiles to obtain the *expected* payoff function $\tilde{u}_i \colon \Sigma \to \mathbb{R}$ where

$$\tilde{u}_i(x) = \mathop{\mathrm{E}}_{j_i \sim x_i}[u_i(j_1,\dots,j_n)] = \sum_{(j_1,\dots,j_n)\in S} x_{1,j_1} \cdots x_{n,j_n} u_i(j_1,\dots,j_n).$$

For a mixed strategy profile $x \in \Sigma$ and a mixed strategy $x'_i \in \Sigma_i$ for player i, we let $(x'_i, x_{-i}) \in \Sigma$ denote the mixed strategy profile where x_i has been replaced by x'_i. A mixed strategy profile $x \in \Sigma$ is a *Nash equilibrium* if for every player i and every mixed strategy $x'_i \in \Sigma_i$ it holds that $\tilde{u}_i(x) \geq \tilde{u}_i(x'_i, x_{-i})$. In other words, no player can improve its expected utility by unilaterally modifying its strategy.

Computational problem. We consider the problem of computing a Nash equilibrium of a normal form game, where the payoff functions u_i are given explicitly, i.e., for every $i \in [n]$ and every $(j_1,\dots,j_n) \in S$, the value $u_i(j_1,\dots,j_n)$ is provided as a rational number. By Nash's theorem such an equilibrium always exists [177].

The simplest proof of Nash's theorem uses Kakutani's fixed point theorem (Theorem 3.2.2). However, prior to our work, it was not known how to use this proof to prove membership of the computational problem in FIXP. Instead, the membership in FIXP was shown by relying on an alternative proof of Nash's theorem that uses Brouwer's fixed point theorem [85].

Now assume for an instant that we allow an extra gate in the definition of FIXP, namely the OPT-gate: a gate that can solve a Linear Program (LP). To be more precise, assume that the gate takes as input the description of an LP of the form

$$\begin{array}{ll} \text{maximize} & c^{\mathsf{T}} x \\ \text{subject to} & Ax \leq b \end{array}$$

namely, it takes as input c, A, b and it outputs an optimal solution of the LP.

If such a gate was allowed in the construction of a FIXP-circuit, then the FIXP-membership of the Nash problem would essentially follow immediately. Indeed, note that at a Nash equilibrium x, every player maximizes its utility given the mixed strategies chosen by the other players. In other words, the mixed strategy x_i played by player i is an optimal solution of the following LP, where the variables are $y \in \mathbb{R}^{m_i}$:

$$\begin{aligned} \text{maximize} \quad & \tilde{u}_i(y, x_{-i}) \\ \text{subject to} \quad & \sum_{j=1}^{m_i} y_j = 1 \\ & y_j \geq 0 \qquad j = 1, \ldots, m_i \end{aligned} \tag{3.1}$$

Note that this is indeed an LP, since $\tilde{u}_i(y, x_{-i})$ is linear in y.

In more detail, the FIXP-circuit F for this problem would be constructed as follows. On input $x = (x_1, \ldots, x_n)$, it outputs $F(x) = (F_1(x), \ldots, F_n(x))$, where $F_i(x) \in \mathbb{R}^{m_i}$ is an optimal solution of the corresponding LP (3.1). Since the LP (3.1) can be put in the form needed for the hypothetical OPT-gate above by replacing the equality constraint by two inequality constraints, we can use the hypothetical gate for solving an LP for this. In more detail, the inputs to the gate will be A and b that encode the inequality constraints, and the vector c for the objective function, which will depend on the values of the other inputs x_{-i}. Clearly, any fixed point of F is a Nash equilibrium of the game.

This simple example already shows how such a gate could make some FIXP-membership results very easy to prove. Importantly, the technique also feels very "natural", because it can be applied almost immediately given the description of the problem, without the need to reformulate the problem in any way. Indeed, in this example, the FIXP-membership is essentially immediately obtained from the simple proof of Nash's theorem based on Kakutani's fixed point theorem.

Unfortunately, this "ideal" gate described above is in fact too good to be true. Indeed, there are some fundamental obstacles to constructing such a gate using the standard gates allowed in FIXP-circuits.

3.3.2 Pseudogates: Circumventing obstacles to the construction of an OPT-gate

We consider the task of constructing a gate that solves LPs. To be more precise, we would like to use the standard algebraic gates allowed in a FIXP-circuit to construct, for any $n \in \mathbb{N}$ and $m \in \mathbb{N}_0$ (where $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$), a new gate $G_{n,m}$ that takes as input $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, and outputs an optimal solution to the following LP:

$$\begin{aligned} \text{maximize} \quad & c^{\mathsf{T}} x \\ \text{subject to} \quad & Ax \leq b \end{aligned} \tag{3.2}$$

Obstacle 1. Any function mapping the description of an LP to an optimal solution of the LP cannot be continuous everywhere. This holds even for very simple LPs.

As an example for this obstacle, consider the following very simple LP:

$$\begin{aligned} \text{maximize} \quad & x_1 v_1 + x_2 v_2 \\ \text{subject to} \quad & x_1 + x_2 = 1 \\ & x_1, x_2 \geq 0 \end{aligned} \tag{3.3}$$

where the variables are x_1, x_2, and $v_1, v_2 \in \mathbb{R}$ are external parameters. Clearly, our gate should be able to solve the following task: given v_1, v_2 as input, output any optimal solution of the LP. However, note that this function is *not* continuous: when $v_1 > v_2$ it outputs $(x_1, x_2) = (1, 0)$, but when $v_1 < v_2$ it outputs $(x_1, x_2) = (0, 1)$. Thus, there is no hope of implementing a gate computing this function by using the gates allowed in a FIXP-circuit, which are all continuous.

Pseudogates: The power of fixed point computation. The crucial observation that allows us to go beyond this impossibility result is the following: when the gate is used inside a FIXP-circuit F, it does not have to work correctly for all inputs x to F; it suffices if it works correctly whenever the input to F is a fixed-point x^* of F. Indeed, in order to prove the membership of some problem in FIXP using F, we have to show that any fixed point x^* of F yields a solution to the problem. Thus, we only care about the behavior of the gate when the input to F is some fixed point x^*. Of course, the gate should remain well-defined for all inputs x, namely not divide by zero, etc.

This observation essentially allows us to use an additional—very powerful—tool in the construction of the gate: fixed point computation. In order to illustrate this point, we show how this tool can be used to construct a “gate” that computes the so-called *Heaviside step function*. For our purposes, we define the Heaviside function as the correspondence $\mathrm{H}\colon \mathbb{R} \rightrightarrows [0,1]$ with

$$\mathrm{H}(x) = \begin{cases} 1 & \text{if } x > 0 \\ [0,1] & \text{if } x = 0 \\ 0 & \text{if } x < 0 \end{cases} .$$

We would like a gate that on input x, outputs any $y \in \mathrm{H}(x)$. Clearly, a gate that is constructed using only the standard FIXP-gates cannot compute H, which is discontinuous at $x = 0$. Indeed, note that the Heaviside function is closely related to the example LP (3.3) above. If we had a gate computing the Heaviside function, then by computing $y \in H(v_1 - v_2)$ and then outputting $(y, 1 - y)$, we would simulate a gate solving (3.3). Similarly, if we had a gate solving the LP (3.3), then by computing a solution (x_1, x_2) to the LP (3.3) with parameters $(v_1, v_2) = (x, 0)$, and outputting x_1, we would simulate a gate for H.

Let us now see how we can construct a “gate” computing the Heaviside function H. Consider the function $G\colon \mathbb{R} \times [0,1] \to [0,1]$ given by $G(x,y) = \min(1, \max(0, x + y))$. Let us examine the fixed points of G, where we think of x as being fixed or an external parameter. If $x > 0$, then the fixed point condition $G(x,y) = y$ implies that y must be equal to 1. If $x < 0$, then the fixed point condition implies that $y = 0$. Finally, when

$x = 0$, the fixed point condition implies that y can take any value in $[0,1]$. In particular, note that we always have $y \in \mathrm{H}(x)$.

How can we use this to prove membership in FIXP? Imagine that we can reduce our problem of interest to the problem of finding a fixed point of a correspondence $F\colon D \rightrightarrows D$, i.e., a point $x \in D$ with $x \in F(x)$. Imagine, further, that we can construct a circuit computing F that uses the standard gates, but also a gate computing H. Then, we can construct a FIXP-circuit $\tilde{F}$ for this problem by replacing the gate for H in F by the function G defined above. In more detail, if we want to use a gate computing H with some input x, we instead compute $G(x,y)$, where y is an additional input to $\tilde{F}$. We also add this value $G(x,y)$ as an additional output to $\tilde{F}$ (namely, the output corresponding to the new input y). As a result, we obtain a FIXP-circuit $\tilde{F}\colon D \times [0,1] \to D \times [0,1]$ that only uses the standard gates and is such that any fixed point (z,y) of $\tilde{F}$ satisfies $z \in F(z)$. In other words, finding a fixed point of the correspondence F reduces to finding a fixed point of the function $\tilde{F}$, which is a standard FIXP-circuit. In the case where F makes use of multiple gates computing H, every occurrence of the gate will be replaced by the construction above using G. In particular, if the gate for H is used ℓ times, then we will obtain $\tilde{F}\colon D \times [0,1]^\ell \to D \times [0,1]^\ell$ such that $\tilde{F}(z, y_1, \dots, y_\ell) = (z, y_1, \dots, y_\ell) \implies z \in F(z)$.

As a result, when constructing a FIXP-circuit for some problem, we can assume that we also have access to a gate computing H. However, one should keep in mind that the gate is only guaranteed to work correctly at a fixed point of the circuit. In order to stress this limitation, we say that we have a *pseudogate* computing H. Note that for the purpose of proving membership in FIXP, a pseudogate is just as good as a normal gate. We now present these ideas more formally.

Definition 3.3.1. Let $A \subseteq \mathbb{R}^n$, and let $B \subset \mathbb{R}^\ell$ be a nonempty, compact, and convex set. For any continuous function $G\colon A \times B \to \mathbb{R}^m \times B$ we let $\mathsf{Fix}_B[G]$ denote the correspondence induced by G with fixed-point constraints on B. Formally, the correspondence $\mathsf{Fix}_B[G]\colon A \rightrightarrows \mathbb{R}^m$ is defined as

$$x \mapsto \{z \in \mathbb{R}^m : \exists y \in B \quad G(x,y) = (z,y)\}.$$

When $f = \mathsf{Fix}_B[G]$ we will say that G is a fixed-point representation of f. We will often have $B = [0,1]^\ell$ for some $\ell \in \mathbb{N}$, in which case we will use Fix_ℓ as an abbreviation for $\mathsf{Fix}_{[0,1]^\ell}$.

Example 1. The function $G_\mathrm{H}\colon \mathbb{R} \times [0,1] \to \mathbb{R} \times [0,1]$, $(x,y) \mapsto (y, \min(1, \max(0, x+y)))$, is a fixed-point representation of the Heaviside function H, i.e., $\mathsf{Fix}_1[G_\mathrm{H}] = \mathrm{H}$. Now let us consider a function similar to the Heaviside function, but which will require us to have the first output of G be something other than just y itself. Let $f\colon \mathbb{R} \rightrightarrows \mathbb{R}$ be defined by

$$f(x) = \begin{cases} x+1 & \text{if } x > 0 \\ [0,1] & \text{if } x = 0 \\ x & \text{if } x < 0 \end{cases}.$$

Then a fixed-point representation of f is given by

$$G_f\colon \mathbb{R}\times[0,1]\to\mathbb{R}\times[0,1], \quad (x,y)\mapsto(x+y,\min(1,\max(0,x+y)))$$

i.e., $\mathsf{Fix}_1[G_f]=f$.

Definition 3.3.2. Let $A\subseteq\mathbb{R}^n$ and let $f\colon A\rightrightarrows\mathbb{R}^m$ be a correspondence. We say that there is a pseudogate computing f if there exists $\ell\in\mathbb{N}_0$ and an algebraic circuit computing $G\colon A\times[0,1]^\ell\to\mathbb{R}^m\times[0,1]^\ell$ such that for all $x\in A$, $\mathsf{Fix}_\ell[G](x)\subseteq f(x)$.

The algebraic circuit computing G can use any of the standard gates allowed in FIXP-circuits, and should be well-defined, in the sense that it never divides by zero, never takes the square root of a negative number, etc. Note that by Brouwer's fixed point theorem, $\mathsf{Fix}_\ell[G](x)$ is never empty. Thus, if there is a pseudogate computing some correspondence f, then f must be nonempty-valued. For a discussion about why Definition 3.3.2 uses "$\mathsf{Fix}_\ell[G](x)\subseteq f(x)$" instead of "$\mathsf{Fix}_\ell[G](x)=f(x)$" see Remark 1 at the end of the section.

Using this terminology we can now formally state:

Lemma 1. *There exists a pseudogate computing the Heaviside function* $\mathrm{H}\colon\mathbb{R}\rightrightarrows[0,1]$.

Proof. The function $G_{\mathrm{H}}\colon\mathbb{R}\times[0,1]\to\mathbb{R}\times[0,1]$, $(x,y)\mapsto(y,\min(1,\max(0,x+y)))$, can be represented by an algebraic circuit using the gates $+,\max,\min$ and rational constants 0 and 1. Furthermore, it is easy to see that $\mathsf{Fix}_1[G_{\mathrm{H}}]=H$. □

The pseudogate for the Heaviside function will be a crucial building block for the construction of the OPT-gate. In fact, as mentioned above, the pseudogate for H essentially immediately yields a pseudogate solving the simple LP (3.3).

Note that a (potential) pseudogate for our general LP (3.2) will necessarily depend on n and m, namely the number of variables and constraints, respectively. As a result, we say that a pseudogate solves such an LP, if, given n,m we can, in polynomial time in n and m, construct a pseudogate solving the LP for fixed n and m.

Can we construct a pseudogate for the general LP (3.2)? Unfortunately, there are a few more obstacles.

Obstacle 2. A pseudogate cannot solve a general LP without some explicit bound on the feasible region.

As an example, consider the following LP, which corresponds to letting $n=1$, $m=2$, $c=1$, $A=(a,-1)^\mathsf{T}$ and $b=(1,0)$ in (3.2):

$$\begin{array}{rl} \text{maximize} & x \\ \text{subject to} & ax\le 1 \\ & x\ge 0 \end{array}$$

where the variable is $x\in\mathbb{R}$. The solution to this LP is $x=1/a$ when $a>0$, and the LP is unbounded when $a\le 0$.

Assume that we have a pseudogate solving the LP (3.2) and we use it to solve the LP above. It is reasonable to only demand that the pseudogate solve the LP correctly when $a > 0$. However, the pseudogate—or, to be more precise, the continuous function G implementing it—should be well-defined for *all* $a \in \mathbb{R}$. In particular, it should never divide by zero or take a square root of a negative number. This is to ensure that the pseudogate can really be used like a normal gate without a second thought.

Unfortunately, this means that no pseudogate can be constructed for LP (3.2). Indeed, by Definition 3.3.2, the existence of such a pseudogate would imply the existence of an algebraic circuit $G\colon \mathbb{R}\times[0,1]^\ell \to \mathbb{R}\times[0,1]^\ell$ such that $\mathsf{Fix}_\ell[G](a) = \{1/a\}$ for all $a > 0$. In particular, $\mathsf{Fix}_\ell[G](a)$ would be unbounded when a tends to 0 from above. However, this is a contradiction to the continuity of G, which says that $G([0,1]\times[0,1]^\ell)$ must be compact and thus, in particular, bounded.

Explicitly bounded domain. This issue can be resolved by introducing an explicit bound on the feasible region, namely by replacing (3.2) by:

$$\begin{aligned} \text{maximize} \quad & c^\mathsf{T} x \\ \text{subject to} \quad & Ax \le b \\ & x \in [-R,R]^n \end{aligned} \tag{3.4}$$

where $R \in \mathbb{R}_{>0}$. Note that the notation "$x \in [-R,R]^n$" is used for convenience here; this constraint can equivalently be rewritten as "$-R \le x_i \le R, \forall i$".

Importantly, the parameter R is *not* fixed, but is just another input to the gate, like c, A and b. As a result, this explicit bound is not a significant limitation, since in most applications it is straightforward—or even trivial—to provide such a bound. For example, in the problem of computing a Nash equilibrium, the LP (3.1) that we used is clearly bounded with $R = 1$.

If we are only interested in finding a feasible point of the LP (3.4), or equivalently in solving the LP when $c = 0$, then indeed there exists a pseudogate for that! However, there is still one last obstacle to constructing a pseudogate that *solves* (3.4).

Obstacle 3. A pseudogate cannot solve a general LP without some constraint qualification.

A constraint qualification is some property that the constraints must satisfy. Importantly, it is a property of the constraints and not of the feasible region. In other words, when a feasible region can be represented by various different sets of constraints, some of them may satisfy the constraint qualification, and others not.

As an example for this obstacle, consider the following LP:

$$\begin{aligned} \text{maximize} \quad & x_2 \\ \text{subject to} \quad & x_1 + a x_2 \le 0 \\ & x_1 \ge 0 \\ & x \in [-1,1]^n \end{aligned}$$

Note that for $a = 0$ the optimal solution is $(x_1, x_2) = (0, 1)$, while for $a > 0$ it is $(x_1, x_2) = (0, 0)$. Clearly, this LP can be expressed in the form (3.4) by letting $n = 2$, $m = 2$, and

$$c = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad A = \begin{pmatrix} 1 & a \\ -1 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad R = 1.$$

Thus, a pseudogate for (3.4) should in particular correctly solve this LP for any $a \in [0, 1]$. According to Definition 3.3.2, this would mean that there exists an algebraic circuit $G\colon [0, 1] \times [0, 1]^\ell \to \mathbb{R}^2 \times [0, 1]^\ell$ such that $\mathsf{Fix}_\ell[G](a) = \{(0, 0)\}$ for all $a \in (0, 1]$, and $\mathsf{Fix}_\ell[G](0) = \{(0, 1)\}$. However, this contradicts the continuity of G.

Indeed, consider the sequence $(a_n)_n$ where $a_n = 1/n > 0$. For any $n \in \mathbb{N}$, let $y_n \in [0, 1]^\ell$ be a fixed point of the function $h\colon [0, 1]^\ell \to [0, 1]^\ell$, $y \mapsto G_2(a_n, y)$, where $G_2(a_n, y) \in [0, 1]^\ell$ denotes the second output of G on input (a_n, y). Recall that such a fixed point must exist by Brouwer's fixed point theorem. Since $(y_n)_n$ is a sequence in the compact set $[0, 1]^\ell$, it has a subsequence $(y_{n_k})_k$ that converges to some $y \in [0, 1]^\ell$. Note that for all $k \in \mathbb{N}$ we have $G(a_{n_k}, y_{n_k}) = ((0, 0), y_{n_k})$, because $\mathsf{Fix}_\ell[G](a_{n_k}) = \{(0, 0)\}$. Now, since $a_{n_k} \to 0$, $y_{n_k} \to y$, and by the continuity of G, it follows that $G(0, y) = ((0, 0), y)$. However, this implies that $(0, 0) \in \mathsf{Fix}_\ell[G](0)$, a contradiction to $\mathsf{Fix}_\ell[G](0) = \{(0, 1)\}$.

The issue in this example essentially stems from the fact that, when $a = 0$, the two inequality constraints are equivalent to a single equality constraint. In fact, it is possible to construct a pseudogate for (3.4) that works as long as this does not happen, i.e., as long as the constraint qualification $\{x \in (-R, R)^n : Ax < b\} \neq \emptyset$ holds (where $<$ is componentwise). However, this rules out equality constraints, which we would clearly like our pseudogate to be able to handle, in particular for the Nash problem. To address this issue, in the next section we consider a modified formulation of our LP that allows explicit equality constraints and we show that we can construct a pseudogate that solves it as long as a well-known constraint qualification holds.

In particular, our constraint qualification will require the equality constraints to be linearly independent. As an example for why the linear independence of the equality constraints is needed, consider the following LP:

$$\begin{aligned} \text{maximize} \quad & x_2 \\ \text{subject to} \quad & x_1 + a x_2 = 0 \\ & x_1 = 0 \\ & x \in [-1, 1]^n \end{aligned}$$

By the same arguments as above, it can be shown that a pseudogate cannot solve this LP correctly for all $a \in [0, 1]$.

Remark 1. The attentive reader might look at the definition of a pseudogate (Definition 3.3.2) and wonder why the condition "$\mathsf{Fix}_\ell[G](x) \subseteq f(x)$" is not simply replaced by "$\mathsf{Fix}_\ell[G](x) = f(x)$". Indeed, the pseudogate presented above for the Heaviside function does satisfy the condition with equality. In fact, using Brouwer's fixed point theorem, it is not too hard to show that *any* pseudogate computing H will satisfy the condition with equality. However, consider now the following modification of the Heaviside function:

$$\widehat{\mathrm{H}}(x) = \begin{cases} 1 & \text{if } x > 1 \\ [0,1] & \text{if } x \in [0,1] \\ 0 & \text{if } x < 0 \end{cases} .$$

Note that the pseudogate G_{H} we provided above for H is also a pseudogate for $\widehat{\mathrm{H}}$, but we now have $\mathsf{Fix}_1[G_{\mathrm{H}}](1) \subsetneq \widehat{\mathrm{H}}(1)$. This begs the question of whether we lose anything by allowing the pseudogate to only compute a subset of the output of the initial correspondence. For the purpose of proving membership in FIXP the answer is no. Ultimately, we only want to show that any fixed point of the circuit that we construct satisfies some conditions (e.g., is a Nash equilibrium). Using a pseudogate that enforces a stronger condition than actually intended will not make this any harder. The important thing to note is that the constructed circuit will always have a fixed point, and thus, even if we use pseudogates that enforce stronger conditions than intended, there is no risk of the conditions being "too strong." Thus, in this context, there is no reason to require equality in Definition 3.3.2, since we only really care about the containment in one direction. Is there any setting where we would care about having equality? The only setting that comes to mind is if one is not only interested in proving FIXP-membership, but wants to construct a circuit such that there is a one-to-one correspondence between its fixed points (perhaps after projecting away some coordinates) and the solutions of the problem that is studied.

3.3.3 The OPT-gate for Linear Programming

We consider the following LP formulation, which includes explicit equality constraints and an explicit bound on the feasible region:

$$\begin{aligned} \text{maximize} \quad & c^{\mathsf{T}} x \\ \text{subject to} \quad & Ax = b \\ & Cx \le d \\ & x \in [-R,R]^n \end{aligned} \tag{3.5}$$

where $x \in \mathbb{R}^n$ is the vector of unknown variables, $c \in \mathbb{R}^n$ defines the objective function, and the constraints are given by $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $C \in \mathbb{R}^{k \times n}$, $d \in \mathbb{R}^k$, and $R \in \mathbb{R}_{>0}$.

We introduce the following constraint qualification for our LP formulation.

Definition 3.3.3. We say that the *explicit Slater condition* is satisfied by the LP (3.5) if the following two conditions hold:

1. *non-empty interior:* there exists $x \in (-R,R)^n$ with $Ax = b$ and $Cx < d$ (componentwise),

2. *linear independence:* the rows of A are linearly independent.

The Slater condition [202] is very popular in convex optimization, where it is usually defined using only the first condition. This is without loss, because the second condition can always be enforced with some additional preprocessing (namely, eliminating redundant equality constraints). For our purpose, however, the second condition is required because we cannot perform the usual preprocessing inside an algebraic circuit. To avoid any confusion, we thus refer to the two conditions above as the *explicit* Slater condition.

The main result of this section can now informally be stated as:

The OPT-gate for Linear Programs

There exists a pseudogate for the LP formulation (3.5). This pseudogate has the following guarantees:

- when the feasible region of the LP is non-empty, it outputs a feasible point.
- when the LP satisfies the explicit Slater condition, it outputs an optimal solution.

This pseudogate can be used like any other algebraic gate for the purpose of proving membership in FIXP.

The informal statement above is formally stated in Theorem 3.3.1 below. Note that with the OPT-gate we can in particular directly prove the FIXP-membership of the Nash problem (Section 3.3.1), since the explicit Slater condition is trivially satisfied by all the LPs in question.

To formalize the statement, we think of the LP (3.5) as being parameterized by the tuple (c, A, b, C, d, R). Thus, after fixing $n \in \mathbb{N}$ and $m, k \in \mathbb{N}_0$, we can define the parameter space

$$P_{n,m,k} = \mathbb{R}^n \times \mathbb{R}^{m \times n} \times \mathbb{R}^m \times \mathbb{R}^{k \times n} \times \mathbb{R}^k \times \mathbb{R}_{>0}.$$

For any choice of parameters $p = (c, A, b, C, d, R) \in P_{n,m,k}$ we let $\mathrm{LP}(p)$ denote the corresponding LP formulated in (3.5). We will use $\mathrm{Feas}(\mathrm{LP}(p))$ and $\mathrm{Opt}(\mathrm{LP}(p))$ to denote its set of feasible and optimal solutions, respectively. The main result of this section can be stated formally as follows.

Theorem 3.3.1. *Given $n \in \mathbb{N}$ and $m, k \in \mathbb{N}_0$ we can construct an algebraic circuit $G\colon P_{n,m,k} \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$ in time $\mathrm{poly}(n,m,k)$ such that for any parameters $p = (c, A, b, C, d, R) \in P_{n,m,k}$ it holds:*

- *if the feasible region of $\mathrm{LP}(p)$ is non-empty, i.e., $\mathrm{Feas}(\mathrm{LP}(p)) \neq \emptyset$, then*

$$\mathsf{Fix}_\ell[G](p) \subseteq \mathrm{Feas}(\mathrm{LP}(p)).$$

- *if* LP(p) *satisfies the explicit Slater condition (Definition 3.3.3), then*

$$\mathsf{Fix}_\ell[G](p) \subseteq \mathrm{Opt}(\mathrm{LP}(p)).$$

Theorem 3.3.1 follows from the more general Theorem 3.3.2, which is stated and proved in the next section.

3.3.4 The OPT-gate for Convex Optimization

We can apply the approach presented above to the more general setting of convex optimization. Consider a Convex Program (CP) of the following form:

$$\begin{array}{rll} \text{minimize} & f(x) & \\ \text{subject to} & Ax = b & \\ & g_i(x) \leq 0 & i = 1, \dots, k \\ & x \in [-R, R]^n & \end{array}$$

where $f\colon \mathbb{R}^n \to \mathbb{R}$ and $g_i\colon \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, k$, are convex functions, and, as before, the remaining constraints are given by $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $R \in \mathbb{R}_{>0}$.

For this setting we can again define the appropriate explicit Slater condition.

Definition 3.3.4. We say that the *explicit Slater condition* is satisfied by the Convex Program (3.6) if the following two conditions hold:

1. *non-empty interior:* there exists $x \in (-R, R)^n$ with $Ax = b$ and $g_i(x) < 0$ for $i = 1, \dots, k$,
2. *linear independence:* the rows of A are linearly independent.

The main result of this section can now informally be stated as follows:

The OPT-gate for Convex Optimization

There exists a pseudogate for the Convex Program (CP) (3.5). This pseudogate has the following guarantees:

- when the feasible region of the CP is non-empty, it outputs a feasible point.
- when the CP satisfies the explicit Slater condition, it outputs an optimal solution.

This pseudogate can be used like any other algebraic gate for the purpose of proving membership in FIXP.

For a formal statement, see Theorem 3.3.2 below.

Parameters. On the way to making this statement formal, we need to allow various parts of the optimization problem to depend on a set of parameters (which are going to be the inputs to our pseudogate). Clearly, A, b and R are such parameters—as before—but we would also like the objective function f and the inequality constraints g_i to be parameterized (even just to be able to encode the LP (3.5) from the previous section). To address this, we introduce an additional parameter $w \in \mathbb{R}^s$ and reformulate the optimization problem as follows:

$$\begin{aligned} \text{minimize} \quad & f(x;w) \\ \text{subject to} \quad & Ax = b \\ & g_i(x;w) \le 0 \qquad i = 1,\dots,k \\ & x \in [-R,R]^n \end{aligned} \tag{3.6}$$

where $f\colon \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}$ and $g_i\colon \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}$, $i = 1,\dots,k$, are continuous functions such that $f(\cdot\,;w)$ and $g_i(\cdot\,;w)$ are convex functions for any $w \in \mathbb{R}^s$. Note that $x \in \mathbb{R}^n$ is still the vector of unknown variables and w is simply an additional *external* parameter, just like A, b and R, and is thus treated as completely fixed when optimizing.

After fixing $n \in \mathbb{N}$ and $m,k,s \in \mathbb{N}_0$, as well as the functions f and g_i, $i = 1,\dots,k$, we can define the parameter space $P_{n,m,k,s,f,g} = \mathbb{R}^s \times \mathbb{R}^{m\times n} \times \mathbb{R}^m \times \mathbb{R}_{>0}$. To simplify notation we write $P_{n,m,f,g}$ to mean $P_{n,m,k,s,f,g}$, since k and s are, in a certain sense, also implicitly given by f and $g = (g_1,\dots,g_k)$. For any choice of parameters $p = (w,A,b,R) \in P_{n,m,f,g}$ we let $\mathrm{CP}(p)$ denote the corresponding CP formulated in (3.6). As before, we will use $\mathrm{Feas}(\mathrm{CP}(p))$ and $\mathrm{Opt}(\mathrm{CP}(p))$ to denote its set of feasible and optimal solutions, respectively. As above, the parameters $p = (w,A,b,R) \in P_{n,m,f,g}$ will be the inputs of the pseudogate we construct.

Representation of functions and subgradients. For computational purposes, we assume that the functions f and g_i are given as algebraic circuits. However, we will also need access to *subgradients* of these functions.

Definition 3.3.5. Let $A \subseteq \mathbb{R}^n$ be a convex set and let $f\colon A \to \mathbb{R}$ be a convex function. A vector $v \in \mathbb{R}^n$ is a *subgradient* of f at the point $x \in A$ if, for all $y \in A$,

$$f(y) - f(x) \ge v \cdot (y - x).$$

We let $\partial f(x)$ denote the *subdifferential* of f at x, namely the set of all subgradients of f at x.

If f is a concave function instead, then the *superdifferential* of f at x is given by $\partial f(x) := -\partial(-f)(x)$. In that case, the elements of $\partial f(x)$ are called *supergradients*.

The subdifferential has the following well-known properties (see, e.g., [191]).

Lemma 2. *Let $A \subseteq \mathbb{R}^n$ be a convex set and let $f\colon A \to \mathbb{R}$ be a convex function. Then it holds that:*

- *$\partial f(x)$ is a closed convex set for all $x \in A$,*

- *$\partial f(x)$ is nonempty for all $x \in \operatorname{rel\,int} A$,*
- *if f is differentiable at x, then $\partial f(x) = \{\nabla f(x)\}$,*
- *$x^\star \in A$ is a global minimum of f on A, if and only if $0 \in \partial f(x^\star)$.*

Note that since $f(\cdot;w)$ and $g_i(\cdot;w)$, $i = 1,\dots,k$, are convex functions defined over $\mathbb{R}^n$, the subdifferentials $\partial f(\cdot;w)$, $\partial g_i(\cdot;w)\colon \mathbb{R}^n \rightrightarrows \mathbb{R}^n$, $i = 1,\dots,k$, are guaranteed to exist and be nonempty. For our purposes we will assume that we are given pseudogates computing these subgradients. In other words, we assume that we are given algebraic circuits $G_{\partial f}, G_{\partial g_i}\colon \mathbb{R}^n \times \mathbb{R}^s \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$, $i = 1,\dots,k$, such that $\mathsf{Fix}_\ell[G_{\partial f}](x,w) \subseteq \partial f(x;w)$ and $\mathsf{Fix}_\ell[G_{\partial g_i}](x,w) \subseteq \partial g_i(x;w)$, $i = 1,\dots,k$, for all $x, w \in \mathbb{R}^n \times \mathbb{R}^s$. See Section 3.7 for an example of how such pseudogates can be constructed.

Example 2. As an example, let us see why our convex optimization setting (3.6) is indeed a generalization of our LP setting (3.5). To go from (3.5) to (3.6) we set $s = n + kn + k$ and decompose $w = (c, C, d) \in \mathbb{R}^n \times \mathbb{R}^{k\times n} \times \mathbb{R}^k$ accordingly. Then we let $f(x;w) = -c^\mathsf{T} x$ and $g_i(x;w) = C_i^\mathsf{T} x - d_i$ for $i = 1,\dots,k$, where C_i denotes the ith row of C. As a result, it is easy to see that the subdifferentials are in fact gradients, namely $\nabla f(x;w) = -c$ and $\nabla g_i(x;w) = C_i$. Clearly, the functions are convex for any fixed value of w, and both the functions and their subdifferentials can easily be expressed as algebraic circuits. In particular, this means that Theorem 3.3.2 below implies Theorem 3.3.1.

We can now formally state the main result of this section.

Theorem 3.3.2. *Given $n \in \mathbb{N}$, $m,k,s \in \mathbb{N}_0$ and $G_{\partial f}$, g_i, $G_{\partial g_i}$, $i = 1,\dots,k$, we can construct an algebraic circuit $G\colon P_{n,m,f,g} \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$ in polynomial time in the input length $\operatorname{poly}(n,m,k,s,\operatorname{size}(G_{\partial f}),\operatorname{size}(g),\operatorname{size}(G_{\partial g_i}))$ such that for any parameters $p = (w,A,b,R)$ in $P_{n,m,f,g}$ it holds:*

- *if the feasible region of $\mathrm{CP}(p)$ is non-empty, i.e., $\mathrm{Feas}(\mathrm{CP}(p)) \neq \emptyset$, then*
$$\mathsf{Fix}_\ell[G](p) \subseteq \mathrm{Feas}(\mathrm{CP}(p)).$$
- *if $\mathrm{CP}(p)$ satisfies the explicit Slater condition, then*
$$\mathsf{Fix}_\ell[G](p) \subseteq \mathrm{Opt}(\mathrm{CP}(p)).$$

Remark 2. First of all, note that according to the statement of Theorem 3.3.2, the construction does not actually need access to an algebraic circuit computing f, but only to a pseudogate for ∂f. However, it requires both a circuit for g_i and a pseudogate for ∂g_i. Furthermore, a careful examination of the proof of Theorem 3.3.2 below reveals that the construction also works if the functions f and g_i are pseudoconvex, instead of convex. A differentiable function $f\colon A \to \mathbb{R}$, where $A \subseteq \mathbb{R}^n$ is an open convex set, is said to be *pseudoconvex* if for all $x, y \in A$ it holds that

$$f(y) < f(x) \implies \nabla f(x) \cdot (y - x) < 0.$$

Any differentiable convex function is pseudoconvex, but every pseudoconvex function is not necessarily convex. Furthermore, every convex function is not necessarily pseudoconvex, because it might not be differentiable. It is possible to define a notion that generalizes both convexity and pseudoconvexity, and to state Theorem 3.3.2 with this notion, but for simplicity we have stated it only for convex functions above.

3.3.5 Proof of Theorem 3.3.2

High-level idea. We want to construct a circuit G that takes as input $(w, A, b, R) \in P_{n,m,f,g}$ and $y \in [0,1]^n$, and outputs $z \in \mathbb{R}^n$ and $\overline{y} \in [0,1]^n$, such that if $y = \overline{y}$, then z is an optimal solution of CP(w, A, b, R) (when CP(w, A, b, R) satisfies the explicit Slater condition). The circuit G will roughly perform the following computations:

1. Compute $x := 2Ry - R$, i.e., scale $y \in [0,1]^n$ into a point x in $[-R, R]^n$.

2. Compute $\mu_i \in \mathrm{H}(g_i(x; w))$ and $\lambda_j \in 2\,\mathrm{H}(a_j \cdot x - b_j) - 1$ for $i = 1, \ldots, k$, and $j = 1, \ldots, m$, where H denotes the Heaviside function.

3. Compute $\mu_0 := 1 - \max(\mu_1, \ldots, \mu_k, |\lambda_1|, \ldots, |\lambda_m|)$.

4. Compute $v_0 \in \partial f(x; w)$ and $v_i \in \partial g_i(x; w)$ for $i = 1, \ldots, k$.

5. Compute
$$z := \Pi_R\left(x - \mu_0 v_0 - \sum_{i=1}^{k} \mu_i v_i - \sum_{j=1}^{m} \lambda_j a_j\right)$$
where Π_R denotes projection to $[-R, R]^n$.

6. Compute $\overline{y} := (z + R)/2R$, i.e., scale $z \in [-R, R]^n$ back into a point $\overline{y}$ in $[0,1]^n$.

7. Output $(z, \overline{y})$.

Note that steps 2 and 4 compute correspondences and will actually be implemented by pseudogates, which will require us to add more auxiliary variables (which we denote by y' in the full construction).

Now assume that $y = \overline{y}$. By construction of the circuit, it follows that $x = z$. For simplicity, assume that $x \in (-R, R)^n$ (the case where x lies on the boundary is handled in the full proof below). Then, from $x = z$ it follows that

$$\mu_0 v_0 - \sum_{i=1}^{k} \mu_i v_i - \sum_{j=1}^{m} \lambda_j a_j = 0.$$

Using this equation, we show that by construction of μ and λ, it must be that $x \in \text{Feas}(\text{CP}((w,A,b,R)))$, if this feasible region is non-empty. Furthermore, again by construction, we additionally have that $(\mu,\lambda) \neq (0,0)$, $\mu \geq 0$ (componentwise), and $g_i(x;w) < 0 \implies \mu_i = 0$ for all $i = 1,\dots,k$. Taking all these conditions together, it follows that x satisfies the so-called Fritz John conditions [146, 164], which are necessary conditions for optimality of x. Now, by using the explicit Slater condition, we can show that $\mu_0 = 0$. In that case, the Fritz John conditions become the well-known Karush-Kuhn-Tucker (KKT) conditions [151, 158]. Given that we have a convex program, the KKT conditions are also sufficient for optimality. Thus, it follows that x is an optimal solution, i.e., $x \in \text{Opt}(\text{CP}((w,A,b,R)))$. The full proof that we present below does not assume any knowledge of the various optimality conditions mentioned here.

Notation. Let $G_{\text{H}} \colon \mathbb{R} \times [0,1] \to \mathbb{R} \times [0,1]$ denote the algebraic circuit which implements the pseudogate computing the Heaviside function H, as given by Lemma 1, i.e., such that $\mathsf{Fix}_1[G_{\text{H}}](x) \subseteq \text{H}(x)$ for all $x \in \mathbb{R}$.

For $i = 1,\dots,k$ let $g_i \colon \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}$ denote the algebraic circuits computing the functions for the inequality constraints. For $i = 1,\dots,k$ let $G_{\partial g_i} \colon \mathbb{R}^n \times \mathbb{R}^s \times [0,1]^t \to \mathbb{R}^n \times [0,1]^t$ denote an algebraic circuit which implements a pseudogate computing the subdifferential ∂g_i, i.e., such that $\mathsf{Fix}_t[G_{\partial g_i}](x,w) \subseteq \partial g_i(x;w)$ for all $(x,w) \in \mathbb{R}^n \times \mathbb{R}^s$. Similarly, let $G_{\partial f} \colon \mathbb{R}^n \times \mathbb{R}^s \times [0,1]^t \to \mathbb{R}^n \times [0,1]^t$ denote an algebraic circuit which implements a pseudogate computing the subdifferential ∂f, i.e., such that $\mathsf{Fix}_t[G_{\partial f}](x,w) \subseteq \partial f(x;w)$ for all $(x,w) \in \mathbb{R}^n \times \mathbb{R}^s$. Note that we have assumed that $G_{\partial f}$, $G_{\partial g_1},\dots,G_{\partial g_k}$ all use the same number t of auxiliary inputs/outputs. This is without loss of generality, because additional auxiliary inputs/outputs can be added to such a circuit without altering the represented correspondence.

Construction of the algebraic circuit *G*. We now describe in detail the construction of the algebraic circuit $G \colon P_{n,m,f,g} \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$. The circuit G has exactly $\ell = n+k+m+t(k+1)$ auxiliary inputs/outputs. We denote the input to circuit G by (w,A,b,R,y,y') where $(w,A,b,R) \in P_{n,m,f,g}$ are the parameters for the convex program (i.e., the inputs to the pseudogate we are constructing), $y \in [0,1]^n$ are the first n auxiliary inputs, and $y' \in [0,1]^{\ell-n}$ are the remaining $\ell - n = k+m+t(k+1)$ auxiliary inputs. The output of the circuit is denoted by $(z,\overline{y},\overline{y}')$, where $z \in \mathbb{R}^n$ is the primary output (i.e., the actual output of the pseudogate we are constructing), $\overline{y} \in [0,1]^n$ are the first n auxiliary outputs, and $\overline{y}' \in [0,1]^{\ell-n}$ are the remaining $n-\ell$ auxiliary outputs. We now describe how the outputs of the circuit are computed using the inputs and standard algebraic gates.

The circuit G begins by computing the vector $x \in [-R,R]^n$ as $x := 2Ry - R$. This simply corresponds to scaling $y \in [0,1]^n$ to a vector in $[-R,R]^n$, and can clearly be computed using the standard algebraic gates. Next, G uses the given algebraic circuits g_i to compute $g_1(x;w),\dots,g_k(x;w)$. Then, for each $i = 1,\dots,k$, the circuit computes $\mu_i \in \text{H}(g_i(x;w))$ by using the pseudogate computing H. To be more precise, the circuit computes $(\mu_i, \overline{y}'_i) := G_{\mathrm{H}}(g_i(x;w), y'_i)$, using the algebraic circuit G_{H}. Note that when $y'_i = \overline{y}'_i$, we indeed have $\mu_i \in \mathrm{H}(g_i(x;w))$, as desired.

For $j = 1, \ldots, m$ let $a_j \in \mathbb{R}^n$ denote the jth row of the matrix A. In particular, the jth equality constraint can be written as $a_j \cdot x = b_j$. The next step is to compute $\lambda_j \in 2\mathrm{H}(a_j \cdot x - b_j) - 1$ for each $j = 1, \ldots, m$, again by using the pseudogate computing H. Formally, this means that the circuit sets $(\lambda'_j, \overline{y}'_{k+j}) := G_{\mathrm{H}}(a_j \cdot x - b_j, y'_{k+j})$ and then $\lambda_j := 2\lambda'_j - 1$. Note that the computation of the μ_i's and the λ_j's has used up exactly $k+m$ coordinates of the auxiliary inputs/outputs $y', \overline{y}'$, which means that $t(k+1)$ are still available at this point.

Next, the circuit computes $v_0 \in \partial f(x;w)$ and $v_i \in \partial g_i(x;w)$ for $i = 1, \ldots, k$. Formally, this is achieved by setting $(v_0, \overline{y}'_{(0)}) := G_{\partial f}(x, w, y'_{(0)})$ and $(v_i, \overline{y}'_{(i)}) := G_{\partial g_i}(x, w, y'_{(i)})$ for $i = 1, \ldots, k$, where $y'_{(i)} = (y'_{k+m+it+1}, \ldots, y'_{k+m+it+t})$ for $i = 0, 1, \ldots, k$, and $\overline{y}'_{(i)}$ is defined analogously.

We summarize some properties of the construction up to this point in the following claim.

Claim 1. *If $y' = \overline{y}'$, then we have:*

- $\mu_i \in \mathrm{H}(g_i(x;w))$ *for* $i = 1, \ldots, k$,
- $\lambda_j \in 2\mathrm{H}(a_j \cdot x - b_j) - 1$ *for* $j = 1, \ldots, m$,
- $v_0 \in \partial f(x;w)$,
- $v_i \in \partial g_i(x;w)$ *for* $i = 1, \ldots, k$.

We are now ready to finish the construction of G. The circuit computes

$$\mu_0 := 1 - \max(\mu_1, \ldots, \mu_k, |\lambda_1|, \ldots, |\lambda_m|).$$

Note that $|\lambda_j|$ can simply be computed as $\max(\lambda_j, -\lambda_j)$. We let $\Pi_R : \mathbb{R}^n \to [-R, R]^n$ denote the projection onto $[-R, R]^n$. The function Π_R can easily be computed using algebraic gates, since it suffices to apply the function $\alpha \mapsto \max(-R, \min(R, \alpha))$ to each coordinate separately. The primary output z of G is computed as

$$z := \Pi_R\left(x - \mu_0 v_0 - \sum_{i=1}^{k} \mu_i v_i - \sum_{j=1}^{m} \lambda_j a_j\right) \tag{3.7}$$

and the auxiliary output $\overline{y} \in [0,1]^n$ of G is then computed as

$$\overline{y} := \frac{z + R}{2R}$$

which, in particular, implies that $\overline{y} \in [0,1]^n$. Note that here it is important that we always have $R > 0$.

This completes the construction of the circuit G. Clearly, the construction can be performed in time $\mathrm{poly}(n, m, k, s, \mathrm{size}(G_{\partial f}), \mathrm{size}(g_1), \ldots, \mathrm{size}(g_k), \mathrm{size}(G_{\partial g_1}), \ldots, \mathrm{size}(G_{\partial g_k}))$. Note that, in particular, we have not used a circuit computing f at any point in the construction.

Fixed-point properties. In Claim 1 we have already noted some properties that must hold when $y' = \overline{y}'$. Now we consider the implications of $y = \overline{y}$. First of all, when $y = \overline{y}$, it follows that $x = z$, since $x = 2Ry - R$ and $z = 2R\overline{y} - R$. From (3.7) we then obtain:

Claim 2. *If $y = \overline{y}$, then $x = z$ and the vector*

$$v := \mu_0 v_0 + \sum_{i=1}^{k} \mu_i v_i + \sum_{j=1}^{m} \lambda_j a_j \tag{3.8}$$

satisfies, for $r = 1, \ldots, n$,

$$v_r > 0 \implies x_r = -R$$

and

$$v_r < 0 \implies x_r = R.$$

Next, we prove the following technical result, which will be useful for the remainder of the proof.

Claim 3. *If $(y, y') = (\overline{y}, \overline{y}')$ and u is a feasible point, i.e., $u \in \mathrm{Feas}(\mathrm{CP}(w, A, b, R))$, then*

- $v_r(u_r - x_r) \geq 0$ *for $r = 1, \ldots, n$,*
- $\mu_i v_i \cdot (u - x) \leq 0$ *for $i = 1, \ldots, k$,*
- $\lambda_j a_j \cdot (u - x) \leq 0$ *for $j = 1, \ldots, m$.*

Furthermore, if $\mu_0 = 0$, then all these terms are equal to zero.

Proof. Since $u \in \mathrm{Feas}(\mathrm{CP}(w, A, b, R))$, it holds that $u \in [-R, R]^n$, $Au = b$ and $g_i(u; w) \leq 0$ for $i = 1, \ldots, k$. It follows that $v_r \cdot (u_r - x_r) \geq 0$ for $r = 1, \ldots, n$, because

$$v_r > 0 \implies x_r = -R \implies u_r \geq x_r$$

and

$$v_r < 0 \implies x_r = R \implies u_r \leq x_r$$

where we used Claim 2 and the fact that $u \in [-R, R]^n$.

By Claim 1 we know that $\mu_i \geq 0$ for $i = 1, \ldots, k$. Now, if $\mu_i > 0$ for some i, then it must be that $g_i(x; w) \geq 0$. But this means that $g_i(x; w) \geq g_i(u; w)$, because u is feasible. Since $v_i \in \partial g_i(x; w)$ (Claim 1), and by the definition of subgradients (Definition 3.3.5), it follows that $v_i \cdot (u - x) \leq g_i(u; w) - g_i(x; w) \leq 0$. As a result, we obtain that $\mu_i v_i \cdot (u - x) \leq 0$ for all $i = 1, \ldots, k$.

If $\lambda_j > 0$ for some j, then by Claim 1 we have $a_j \cdot x - b_j \geq 0$. Since u is feasible, we have $a_j \cdot u - b_j = 0$ and thus $a_j \cdot (u - x) \leq 0$. Similarly, if $\lambda_j < 0$ for some j, then by Claim 1 we have $a_j \cdot x - b_j \leq 0$, which by feasibility of u yields $a_j \cdot (u - x) \geq 0$. As a result, we obtain that $\lambda_j a_j \cdot (u - x) \leq 0$ for all $j = 1, \ldots, m$.

Finally, consider the case where $\mu_0 = 0$. Taking the inner product of (3.8) with $(u-x)$, we obtain

$$\sum_{r=1}^{n} \nu_r(u_r - x_r) = \sum_{i=1}^{k} \mu_i v_i \cdot (u-x) + \sum_{j=1}^{m} \lambda_j a_j \cdot (u-x)$$

which, together with the above, implies that all the terms must be zero. □

We are now ready to prove the desired properties of G in the following two claims. Recall that z is the primary output of the circuit G, i.e., the output of the pseudogate it computes.

Claim 4. *If* $(y,y') = (\overline{y},\overline{y}')$ *and* $\mathrm{Feas}(\mathrm{CP}(w,A,b,R)) \neq \emptyset$*, then* $z \in \mathrm{Feas}(\mathrm{CP}(w,A,b,R))$.

Proof. We will show that $x \in \mathrm{Feas}(\mathrm{CP}(w,A,b,R))$, which suffices to prove the claim since $x = z$ by Claim 2. Since $\mathrm{Feas}(\mathrm{CP}(w,A,b,R)) \neq \emptyset$, there exists a feasible vector u, i.e., $u \in [-R,R]^n$ such that $Au = b$ and $g_i(u;w) \leq 0$ for $i = 1,\dots,k$.

Now, towards a contradiction, let us assume that $x \notin \mathrm{Feas}(\mathrm{CP}(w,A,b,R))$. Since $x \in [-R,R]^n$, this means that there exists $i^\star$ with $g_{i^\star}(x;w) > 0$, or $j^\star$ with $a_{j^\star} \cdot x \neq b_{j^\star}$. In both cases, it follows that $\mu_0 = 0$, since by Claim 1, $\mu_{i^\star} = 1$ or $\lambda_{j^\star} \in \{-1,1\}$, respectively. By Claim 3, it follows that $\mu_i v_i \cdot (u-x) = 0$ for all i, and $\lambda_j a_j \cdot (u-x) = 0$ for all j.

If there exists $i^\star$ with $g_{i^\star}(x;w) > 0$, then by Claim 1 we have that $\mu_{i^\star} = 1 > 0$. Furthermore, since $v_{i^\star} \in \partial g_{i^\star}(x;w)$ (Claim 1), it follows by the definition of subgradients (Definition 3.3.5) that $v_{i^\star} \cdot (u-x) \leq g_{i^\star}(u;w) - g_{i^\star}(x;w) < 0$, since u is feasible. But this means that $\mu_{i^\star} v_{i^\star} \cdot (u-x) < 0$, a contradiction.

It remains to consider the case where there exists $j^\star$ with $a_{j^\star} \cdot x \neq b_{j^\star}$. If $a_{j^\star} \cdot x > b_{j^\star}$, then $\lambda_{j^\star} = 1 > 0$ (Claim 1), and $a_{j^\star} \cdot (u-x) < 0$, since u is feasible. On the other hand, if $a_{j^\star} \cdot x < b_{j^\star}$, then $\lambda_{j^\star} = -1 < 0$ (Claim 1), and $a_{j^\star} \cdot (u-x) > 0$, since u is feasible. As a result, in both cases we obtain that $\lambda_{j^\star} a_{j^\star} \cdot (u-x) < 0$, a contradiction.

Since we have obtained a contradiction in all cases, it must be the case that x is in $\mathrm{Feas}(\mathrm{CP}(w,A,b,R))$. □

Claim 5. *If* $(y,y') = (\overline{y},\overline{y}')$ *and* $\mathrm{CP}(w,A,b,R)$ *satisfies the explicit Slater condition, then we have* $z \in \mathrm{Opt}(\mathrm{CP}(w,A,b,R))$.

Proof. By Claim 2, $x = z$, and thus it suffices to show that $x \in \mathrm{Opt}(\mathrm{CP}(w,A,b,R))$. By Claim 4, we already know that x is feasible for $\mathrm{CP}(w,A,b,R)$.

By Claim 1, we always have $\mu_0 \geq 0$. Let us first consider the case where $\mu_0 > 0$. Let $u \in [-R,R]^n$ be any feasible point. Taking the inner product of (3.8) with $(u-x)$ we obtain

$$\mu_0 v_0 \cdot (u-x) = \sum_{r=1}^{n} \nu_r(u_r - x_r) - \sum_{i=1}^{k} \mu_i v_i \cdot (u-x) - \sum_{j=1}^{m} \lambda_j a_j \cdot (u-x).$$

By Claim 3, all the terms on the right hand side are non-negative. This implies that $\mu_0 v_0 \cdot (u-x) \geq 0$ and thus $v_0 \cdot (u-x) \geq 0$. Since $v_0 \in \partial f(x;w)$ (Claim 1), by the definition of subgradients (Definition 3.3.5), it follows that $f(u) - f(x) \geq v_0 \cdot (u-x) \geq 0$. Since this holds for any feasible point u, this means that x is an optimal solution, i.e., $x \in \mathrm{Opt}(\mathrm{CP}(w,A,b,R))$.

It remains to handle the case where $\mu_0 = 0$. We will show that this case cannot occur. Towards a contradiction, assume that indeed $\mu_0 = 0$. Since $\mathrm{CP}(w,A,b,R)$ satisfies the explicit Slater condition, there exists $u \in (-R,R)^n$ with $Au = b$ and $g_i(u;w) < 0$ for $i = 1,\ldots,k$. In particular, u is feasible and since $\mu_0 = 0$, by Claim 3 we obtain that $\nu_r(u_r - x_r) = 0$ for all r, $\mu_i v_i \cdot (u-x) = 0$ for all i, and $\lambda_j a_j \cdot (u-x) = 0$ for all j.

If $\nu_r > 0$ for some r, then, by Claim 2, we have $x_r = -R$. Since $u_r \in (-R,R)$, it follows that $u_r - x_r > 0$, and thus $\nu_r(u_r - x_r) > 0$, a contradiction. If $\nu_r < 0$ for some r, then, by Claim 2, we have $x_r = R$, and thus again $\nu_r(u_r - x_r) > 0$, a contradiction. As a result, we obtain that $\nu_r = 0$ for all $r = 1,\ldots,n$.

If $\mu_i > 0$ for some i, then by Claim 1 it must be that $g_i(x;w) \geq 0$. Since $v_i \in \partial g_i(x;w)$ (Claim 1), it follows by the definition of subgradients (Definition 3.3.5) that $v_i \cdot (u-x) \leq g_i(u;w) - g_i(x;w) < 0$, because $g_i(u;w) < 0$. Thus, we obtain that $\mu_i v_i \cdot (u-x) < 0$, a contradiction. As a result, we have $\mu_i = 0$ for all $i = 1,\ldots,k$.

Now, since $\mu_0 = \mu_i = \nu_r = 0$, the equation in Claim 2 just yields $\sum_{j=1}^m \lambda_j a_j = 0$. But $\mathrm{CP}(w,A,b,R)$ satisfies the explicit Slater condition, so the vectors a_j, $j = 1,\ldots,m$, are linearly independent. It follows that $\lambda_j = 0$ for all $j = 1,\ldots,m$. However, note that this is a contradiction, because according to the construction of μ_0, if $\mu_i = 0$, for all $i = 1,\ldots,k$, and $\lambda_j = 0$, for all $j = 1,\ldots,m$, then $\mu_0 = 1$. □

The proof of Theorem 3.3.2 is thus completed.

3.4 Applications to Game Theory

In this section, we discuss further applications of our technique to equilibrium computation in strategic games. In Section 3.3.1, we already demonstrated how the employment of our OPT-gate can make the FIXP-membership proof of normal form games essentially straightforward. In this section, we provide further FIXP-membership results, namely:

- Computing equilibria in *concave n-player games* [192]. In these games, which generalize the normal form games mentioned above, the players have continuous strategy spaces and continuous payoff functions. Again via a relatively simple proof based on convex programs rather than Linear Programs, we show that computing equilibria of these games is in FIXP; the FIXP-completeness follows from the FIXP-hardness of normal form games due to Etessami and Yannakakis [85].

- Computing *ε-proper equilibria* in normal form games, an equilibrium refinement due to Myerson [174]. We show that the corresponding problem is in FIXP. Our proof first shows how to compute solutions to *systems of conditional convex constraints* (Section 3.4.2) using our OPT-gate, and then employs this to show the FIXP-membership result for ε-proper equilibria.

- Computing *stationary λ-discounted equilibria* in n-player stochastic games [200]. We show that this problem is FIXP-complete. The FIXP-membership could technically already be achieved via the machinery used by Etessami and Yannakakis [85] to achieve the FIXP-membership of the 2-player problem, but our proof uses the OPT-gate here as well. The FIXP-hardness follows from [85] by viewing a normal form game as a stochastic game consisting of a single state.

3.4.1 Concave *n*-player games

In this section, we generalize the FIXP-membership result from normal form games to concave games, a class of games studied by Rosen [192], which we define below. Together with the FIXP-hardness result for normal form games [85], we obtain the following result.

Theorem 3.4.1. *Computing an equilibrium of a concave n-player game is* FIXP-*complete.*

An n-player game G consists of n players, with each player $i \in [n]$ having a compact and convex strategy space $\Sigma_i \subseteq \mathbb{R}^{m_i}$ and a continuous payoff function $u_i \colon \Sigma \to \mathbb{R}$, where $\Sigma := \Sigma_1 \times \cdots \times \Sigma_n$. For a strategy profile $x \in \Sigma$, let $(y, x_{-i}) = (x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n)$ denote the strategy profile where player i unilaterally changes strategy. The game G is *concave* if for any player i and fixed $x \in \Sigma$, the function $u_i(y, x_{-i})$ is concave in y.

A strategy profile $x \in \Sigma$ is an *equilibrium* if $u_i(x) = \max_{y \in \Sigma_i} u_i(y, x_{-i})$ for every player i, that is, no player can increase its payoff by a unilateral change of strategy. Rosen [192] provides a proof that every concave game admits an equilibrium point by constructing an upper hemicontinuous correspondence $F \colon \Sigma \to 2^{\Sigma}$ mapping any $x \in \Sigma$ to the set $\{y \in \Sigma \mid \forall i \colon y_i \text{ is a best response to } x_{-i}\}$. One then applies Kakutani's fixed point theorem to show the existence of some strategy profile $x \in \Sigma$ with $x \in F(x)$. As x_i is a best response to x_{-i} for all players by construction, x is an equilibrium.

Computational problem. In the computational problem, we assume that each strategy space Σ_i is given as the set of all $x \in [-R_i, R_i]^{m_i}$ satisfying equality constraints $A_i x = b_i$ and inequality constraints $g_{ij}(x) \leq 0$ for some $R_i > 0$, matrix $A_i \in \mathbb{R}^{d_i \times m_i}$, vector $b_i \in \mathbb{R}^{d_i}$, and convex functions g_{ij}, $j = 1, \ldots, k_i$. As A_i is given as input, we may apply preprocessing to eliminate linear dependence among the rows, so we just have to assume that the constraints satisfy the general Slater condition. We also assume that we are given algebraic circuits for g_{ij} and pseudogates computing their subdifferentials ∂g_{ij}, as well as the superdifferentials ∂u_i.

In order to prove FIXP-membership, we construct a circuit $F \colon D \to D$, where $D = \times_i [-R_i, R_i]^{m_i}$. On input x, the ith output of the circuit F will be a best response of player i to x_{-i}. Namely, the i-th output of F is simply set as the output of the OPT-gate for the following convex program:

$$\begin{aligned} \text{minimize} \quad & -u_i(y, x_{-i}) \\ \text{subject to} \quad & A_i y = b_i \\ & g_{ij}(y) \leq 0 \qquad j = 1, \ldots, k_i \\ & y \in [-R_i, R_i]^{m_i} \end{aligned}$$

Note that the explicit Slater condition is satisfied by assumption. Thus, the OPT-gate correctly solves the convex program. As a result, if x is a fixed point of F, then $x \in \Sigma$ and x_i is a best response to x_{-i} for every player i, meaning that x is indeed an equilibrium of the game.

Remark 3. Rosen [192] actually considers a more general setting where the space of strategy profiles Σ is not assumed to be equal to the product space of the players' individual strategy spaces. Rather, he just assumes that the space of strategies is a compact and convex subset $\Sigma \subseteq \Sigma_1 \times \cdots \times \Sigma_n$. Let us assume that Σ is given as the set of all $x \in [-R, R]^m$ that satisfy $Ax = b$ and $g_j(x) \leq 0$, $j = 1, \ldots, k$, where, as per usual, A is a matrix, b a vector and the g_j are convex functions. We may write $A = (A_1 \mid \cdots \mid A_n)$ as a concatenation of block matrices. For fixed $x \in \Sigma$, player i would then maximize its utility $u_i(y, x_{-i})$ subject to the constraints $A_i y = b_i(x_{-i}) := b - \sum_{j \neq i} A_j x_j$ and $g_j(y, x_{-i}) \leq 0$ for all j. As A_i is given in the input we can apply preprocessing to it and remember the linear combinations required to eliminate potential linear dependence in the rows of A_i. Applying these same linear combinations to the A_i and b_i, we obtain constraints $\tilde{A}_i y = \tilde{b}_i$. It suffices to require that the constraints $A_i y = b_i(x_{-i})$ and $g_j(y, x_{-i}) \leq 0$, $j = 1, \ldots, k$, satisfy the general Slater condition for all $i \in [n]$ and $x \in \Sigma$.

3.4.2 Computing an ε-proper equilibrium via systems of conditional convex constraints

In this section we consider the Nash equilibrium refinement of proper equilibrium due to Myerson [174]. First, we define the notion of ε-proper equilibrium and then define a proper equilibrium as a limit point of ε-proper equilibria.

Definition 3.4.1 ([174])**.** Let Γ be a finite n-player game in strategic form. Given $\varepsilon > 0$, a mixed strategy profile x is an *ε-proper equilibrium* in Γ if it is fully mixed and satisfies $x_{ik} \leq \varepsilon x_{i\ell}$ whenever $u_i(k, x_{-i}) < u_i(\ell, x_{-i})$ for all players i and all pairs of actions k, ℓ of player i.

A mixed strategy profile x is a *proper equilibrium* if and only if it is a limit point of a sequence of ε-proper equilibria with $\varepsilon \to 0^+$.

It was proved recently by Hansen and Lund [135] that the task of approximating a proper equilibrium is complete for the class FIXP$_a$ of [85]. This work follows a line of similar results [84, 87] for approximating other notions of equilibrium refinements, e.g. Selten's trembling hand perfect equilibrium [199], that are, like proper equilibria, defined as limit points of certain ε-equilibria. These previous results were proved by showing that the problems of computing the ε-equilibria are in FIXP. In fact they can be computed by a reduction to a basic FIXP-problem (Definition 4.2.4) where ε is an input variable of the algebraic circuit. This additional property is exploited to prove that approximating the equilibrium refinement notions is in FIXP$_a$ by using the ability of an algebraic circuit to compute a "virtual infinitesimal" by means of repeated squaring that then takes the place of ε.

Hansen and Lund [135] did not prove that computing an ε-proper equilibrium is in FIXP, but instead proved that computing a so-called δ-almost ε-proper equilibrium is in FIXP. These equilibria can in fact be computed by reducing to a basic FIXP-problem where δ and ε are inputs of the algebraic circuit. It is then shown that approximating a proper equilibrium is in FIXP$_a$ by substituting "virtual infinitesimals" for both δ and ε. The question of whether computing an ε-proper equilibrium is in FIXP was left as an open problem. Using our technique we resolve this question, and thereby significantly simplify the proof of Hansen and Lund [135] that approximating a proper equilibrium is FIXP$_a$-complete.

Theorem 3.4.2. *The problem of computing an ε-proper equilibrium of a given finite game n-player game in normal form is in* FIXP.

To establish existence of ε-proper equilibria, Myerson [174] made use of Kakutani's fixed point theorem (Theorem 3.2.2). Suppose that Γ is a given n-player game in strategic form and $\varepsilon > 0$. Let $S_i = [m_i]$ and u_i be the set of strategies and utility function of Player i. Define $\eta_i(\varepsilon) = \varepsilon^{m_i}/m_i$ and let

$$\Sigma_i^{\eta_i} = \left\{ y \in \mathbb{R}^{m_i} \,\middle|\, \sum_j y_j = 1; \forall j : y_j \geq \eta_i \right\}$$

be the set of η_i-*perturbed* mixed strategies for Player i. Let $\eta = (\eta_1, \ldots, \eta_n)$ and define $\Sigma^\eta = \prod_{i=1}^n \Sigma_i^{\eta_i}$ to be the set of all η-perturbed mixed strategy profiles for Γ. Define the correspondence $F \colon \Sigma^\eta \rightrightarrows \Sigma^\eta$ by $F(x) = \prod_{i=1}^n F_i(x)$, where

$$F_i(x) = \{ y \in \Sigma^\eta \mid \forall k, \ell \in S_i : u_i(k, x_{-i}) < u_i(\ell, x_{-i}) \Rightarrow y_{ik} \leq \varepsilon y_{i\ell} \} \ .$$

Clearly any fixed point of F is an ε-proper equilibrium of Γ. Myerson concluded his proof by showing that F satisfies the conditions of the Kakutani fixed point theorem. In particular, F_i is nonempty since we have $y_i \in F_i(x)$ where

$$y_{ik} = \varepsilon^{\rho_i(k)} / \sum_{\ell \in S_i} \varepsilon^{\rho_i(\ell)}$$

and $\rho_i(k) = |\{\ell \in S_i \mid u_i(k, x_{-i}) < u_i(\ell, x_{-i})\}|$.

Computing a fixed point of F is a special case of the result of the following subsection (see Theorem 3.4.3).

Solving Systems of Conditional Convex Constraints

In this section we consider the task of solving systems of what we refer to as conditional convex constraints by finding fixed points. We make use of the main result of the section (Theorem 3.4.3) to prove Theorem 3.4.2, but it could be applicable to other problems as well, and therefore it could be of independent interest.

Definition 3.4.2. A conditional convex constraint on n variables is a pair (f, g) of a continuous function $f : \mathbb{R}^n \to \mathbb{R}$ and a convex function $g : \mathbb{R}^n \to \mathbb{R}$. A point $x \in \mathbb{R}^n$ satisfies (f, g) if $f(x) > 0 \Rightarrow g(x) \leq 0$.

A system of conditional convex constraints naturally defines a search problem, where the task is to find a point x that satisfies all constraints of the system. A system of conditional convex constraints also defines a correspondence in a natural way. We shall further restrict our attention to correspondences with nonempty, compact, and convex domain.

Definition 3.4.3. Let $D \subseteq \mathbb{R}^n$ be a non-empty, compact and convex set. Let $(f_1, g_1), \dots, (f_m, g_m)$ be conditional convex constraints on n variables. The correspondence $F\colon D \rightrightarrows D$ defined by D and $(f_1, g_1), \dots, (f_m, g_m)$ is given by

$$F(x) = \{y \in D \mid \forall i : f_i(x) > 0 \Rightarrow g_i(y) \leq 0\} \ .$$

Note that there is a one-to-one correspondence between fixed points of F and solutions of the system of constraints contained in D.

Except for the property of nonempty-valued, such correspondences satisfy the conditions of Kakutani's fixed point theorem.

Proposition 1. *Let F be a correspondence defined by a non-empty, compact and convex set $D \subseteq \mathbb{R}^n$ and conditional convex constraints $(f_1, g_1), \dots, (f_m, g_m)$. Then F is uhc as well as compact and convex-valued.*

Proof. Let $x \in D$ and let $V \subseteq D$ be an open set such that $F(x) \subseteq V$. By continuity of the functions f_i we may find an open set U containing x such that if $x' \in U$ and $f_i(x') > 0$ then we have $f_i(x) > 0$ as well. It follows that $F(x') \subseteq F(x) \subseteq V$, which means that F is uhc. We also have that $F(x)$ is an intersection of closed and convex sets, and $F(x)$ is thus closed and convex as well. □

Thus if we had a guarantee that F was nonempty-valued as well, a fixed point would be guaranteed by Kakutani's fixed point theorem. We can associate a total search problem with F where the task is to find $x \in D$ such that either $x \in F(x)$ or $F(x) = \emptyset$. For the computational problem, we assume that D is given as a set of linear constraints $x \in [-R, R]^n$ and $Ax = b$, and convex constraints $h_i(x) \leq 0$, $i = 1, \dots, k$, that satisfy the explicit Slater condition. We also assume that we are given algebraic circuits computing f_i, g_i and h_i, and pseudogates computing the subgradients of g_i and h_i.

The idea is that the function G in the proof below is derived from a system of convex constraints in variables y that are parameterized by variables x. We consider the constraints given by $\max(0, f_i(x))g_i(y) \leq 0$. When $f_i(x) > 0$ this is equivalent to the constraint $g_i(y) \leq 0$. When $f_i(x) \leq 0$, the constraint becomes trivial.

Theorem 3.4.3. *The problem of solving systems of conditional convex constraints is in* FIXP.

Proof. Define a circuit $G\colon [-R,R]^n \times [-R,R]^n \to [-R,R]^n \times [-R,R]^n$, by letting $G(x,y) := (\overline{x}, \overline{y})$. Here $\overline{y}$ is computed as the output of our convex OPT-gate for the following feasibility problem (parameterized by x)

$$\begin{array}{ll} \text{maximize} & 0 \\ \text{subject to} & Az = b \\ & h_i(z) \leq 0 \qquad i = 1,\dots,k \\ & z \in [-R,R]^n \\ & \max(0, f_i(x))g_i(z) \leq 0 \qquad i = 1,\dots,m \end{array}$$

and $\overline{x}$ is the projection of y onto D, which can be obtained by using the OPT-gate to solve

$$\begin{array}{ll} \text{minimize} & \|y - z\|_2^2 \\ \text{subject to} & Az = b \\ & h_i(z) \leq 0 \qquad i = 1,\dots,k \\ & z \in [-R,R]^n \end{array}$$

Suppose that (x,y) is a fixed point of G. We argue that x is a solution to the search problem described above. First of all, note that $x \in D$, because $x = \overline{x}$ is the projection of some point (namely, y) onto D. Now there are two cases. If the set of feasible solutions to the first convex program is empty, then it follows that $F(x) = \emptyset$, and so $x \in D$ is indeed a solution to our search problem. If, on the other hand, the set of feasible solutions is non-empty, then it follows from the first part of Theorem 3.3.2 that $\overline{y} = y$ is a solution to the first convex program. The first three constraints show that $y \in D$, and from the inequalities $\max(0, f_i(x))g_i(y) \leq 0$ it follows that $f_i(x) > 0 \Rightarrow g_i(y) \leq 0$, i.e., $y \in F(x)$. But since x is the projection of y onto D, and $y \in D$, it must be that $x = y$, and thus $x \in F(x)$. □

3.4.3 n-player Stochastic Games

Stochastic games, as first introduced by Shapley in his seminal work [200], model dynamic interaction between players in an environment whose state is changing according to a stochastic process influenced by the actions of the players. We shall here consider discrete-time finite games, where players receive immediate payoffs in each round of play and discount future payoffs. Shapley's model then corresponds to the special case of two-player zero-sum games.

The main result of this section is the following.

Theorem 3.4.4. *Computing a* stationary λ-discounted equilibrium *of an n-player stochastic game is* FIXP*-complete.*

Next, we first define n-player stochastic games formally, as well as the equilibrium notion which appears in the statement of Theorem 3.4.4 above. Then we present the proof of FIXP-membership for the problem; the FIXP-hardness follows from [85], by considering a single-state stochastic game, i.e., a normal form game.

An n-player finite stochastic game Γ is given as follows. The game is played on a finite set of states S. Every player i has a finite set of actions A_i. Let $A = A_1 \times \cdots \times A_n$ denote the set of action profiles and $P = \{(s,a) : s \in [n], a \in A\}$ the pairs of states and action profiles. The immediate payoffs to player i are then given by a function $u_i \colon P \to \mathbb{R}$ and the state transitions are given by a function $q \colon P \to \Delta(S)$. Let $M = \max_{i,(s,a)\in P} |u_i(s,a)|$.

A play of Γ is an infinite sequence $h \in P^\infty$. A finite play up to stage t is a sequence $h_t \in P^{t-1} \times S$. Let $\mathcal{H} = \dot\cup_{t=1}^{\infty} \left(P^{t-1} \times S\right)$ denote the set of all finite plays. A behavioral strategy for player i is a function $\sigma_i \colon \mathcal{H} \to \Delta(A_i)$. A stationary strategy is a behavioral strategy that depends only on the last state of a finite play. A stationary strategy x_i may thus be viewed as a function $x_i \colon S \to \Delta(A_i)$. Behavioral strategies σ_i for each player i form a behavioral strategy profile $\sigma = (\sigma_1, \dots, \sigma_n)$. In the same way, stationary strategies for each player form a stationary strategy profile. A behavioral strategy profile σ and an initial state $s^1 \in S$ define by Kolmogorov's extension theorem a unique probability distribution $\Pr_{s^1,\sigma}$ on plays $(s^1, a^1, s^2, a^2, \dots)$, where the conditional probability of $a^t = a$ given the play up to stage t, $h_t = (s^1, a^1, \dots, s^t)$, is equal to $\prod_{i=1}^n \Pr[\sigma_i(h_t) = a_i]$, and the conditional probability of s^{t+1} given s^t and a^t is equal to $q(s^t, a^t)$. We denote by $\mathrm{E}_{s^1,\sigma}$ the expectation with respect to $\Pr_{s_1,\sigma}$.

For every *discount factor* $0 < \lambda \le 1$, the λ-discounted payoff to player i is defined to be

$$\gamma_i^\lambda(s^1, \sigma) = \mathop{\mathrm{E}}_{s^1,\sigma}\left[\lambda \sum_{t=1}^{\infty} (1-\lambda)^{t-1} u_i(s^t, a^t)\right] . \tag{3.9}$$

A behavioral strategy profile σ is a λ-discounted equilibrium if

$$\gamma_i^\lambda(s^1, \sigma) \ge \gamma_i^\lambda(s^1, (\sigma_i', \sigma_{-i})) , \tag{3.10}$$

for all states $s_1 \in S$, all players $i \in [n]$, and all behavioral strategies σ_i' for player i. It was proved by Fink [102] and Takahashi [214] that any finite discounted stochastic game has a λ-discounted equilibrium in stationary strategies for any discount factor λ. In the case of two-player zero-sum games, Shapley proved existence of the λ-discounted value $v_\lambda \in \mathbb{R}^S$, as well as optimal stationary strategies, for the λ-discounted payoff.

The results of Shapley, Fink, and Takahashi lead to a natural real-valued total search problem. Etessami and Yannakakis [85] proved the FIXP-membership of the problem of finding the λ-discounted values and optimal stationary strategies in two-player zero-sum stochastic games with discounted payoffs.

Now let Γ be an n-player stochastic game. The proofs by Fink and Takahashi of existence of λ-discounted equilibrium in stationary strategies both make use of Kakutani's fixed point theorem (Theorem 3.2.2).[6] Let us now consider the approach of Takahashi, specialized to finite stochastic games.

Valuations of states $v_i \colon S \to \mathbb{R}^n$ by every player i now form a *valuation profile* $v = (v_1, \dots, v_n)$. Given a discount factor λ and a valuation profile v, we can form associated n-player normal form games $\Gamma_{s,\lambda}(v)$, generalizing the case of two players. For every state $s \in S$, the utility function $u_i^{s,\lambda,v} \colon A \to \mathbb{R}$ of player i in $\Gamma_{s,\lambda}(v)$ is given by

$$u_i^{s,\lambda,v}(a) = \lambda u_i(s,a) + (1-\lambda) \sum_{s' \in S} q(s' \mid s,a) v_i(s')$$

A stationary strategy profile $x = (x_1, \dots, x_n)$ in Γ_λ induces strategy profiles $x(s) = (x_1(s), \dots, x_n(s))$ in the games $\Gamma_{s,\lambda}(v)$, and corresponding valuations of states $u_i^{s,\lambda,v}(x(s))$. Let

$$D = ([-M,M]^S)^n \times (\Delta(A_1)^S \times \cdots \times \Delta(A_n)^S)$$

be the set of valuation profiles v and stationary strategy profiles x. Takahashi defines a correspondence $F : D \rightrightarrows D$ whose fixed points are pairs of valuation profiles and stationary strategy profiles such that the stationary strategy profiles are λ-discounted equilibria in Γ_λ. Letting $F(v,x) = (G(v,x), H(v,x))$, Takahashi defines

$$G(v,x)_{i,s} = \max_{y(s)_i \in \Delta(A_i)} u_i^{s,\lambda,v}(y(s)_i; x(s)_{-i})$$

and

$$H(v,x)_{i,s} = \operatorname*{arg\,max}_{y(s)_i \in \Delta(A_i)} u_i^{s,\lambda,v}(y(s)_i; x(s)_{-i})$$

By Berge's maximum theorem (Theorem 3.2.3), the correspondence F satisfies the requirements of Kakutani's fixed point theorem (Theorem 3.2.2) which in turn yields existence of a fixed point (v,x). Takahashi then proves that the stationary strategy profile x is a λ-discounted equilibrium in Γ_λ.

We obtain FIXP-membership by replacing these correspondences by OPT gates, obtaining a circuit computing a function $F' : D' \to D'$, where $D' := ([-M,M]^S)^n \times ([0,1]^{|A_1|})^S \times \cdots \times ([0,1]^{|A_n|})^S$. On input (v,x), simply consider for every player i and every state s the linear program computing best replies for player i in the game $\Gamma_{s,\lambda}$:

$$\begin{aligned} \text{maximize} \quad & u_i^{s,\lambda,v}(z, x(s)_{-i}) \\ \text{subject to} \quad & \sum_{j=1}^{|A_i|} z_j = 1 \\ & z \in [0,1]^{|A_i|} \end{aligned}$$

[6]More accurately, the proof by Takahashi [214] applies to stochastic games with infinite action spaces, and as a consequence uses a generalization of Kakutani's fixed point to locally convex spaces due to Fan [90] and Glicksberg [123].

Denote by $\overline{x}(s)_i$ the output of the OPT gate. Then $\overline{v}(s)_i = u_i^{s,\lambda,v}(\overline{x}(s)_i, x(s)_{-i})$ is computed and we let $F'(v,x) = (\overline{v}, \overline{x})$. The set of fixed points of F and F' clearly coincide and the result follows.

3.5 Applications to Cake Cutting

In this section, we discuss the applications of our main technique to the area of fair division, and in particular to the complexity of the well-known *envy-free cake cutting problem* [108] (see also [38, 188, 190]). In this problem, the cake serves as a metaphor for a divisible resource—represented without loss of generality by the interval $[0,1]$—which needs to be divided among a set of n agents, such that every agent receives a piece of cake she prefers compared to any piece assigned to any other agent. In the general formulation of the problem, a "piece" can be a collection of possibly disconnected subintervals. In the *contiguous* version, each piece is a single interval (and hence the cake is divided using $n-1$ cuts), and in that case any division of the cake can be represented as a point x in the simplex Δ^{n-1}, with x_j denoting the j-th coordinate.

Formally, the valuation of agent i for the j-th piece is given by a map $u_{ij}\colon \Delta^{n-1} \to \mathbb{R}_{\geq 0}$. Given a division x, we say that agent i prefers the j-th piece if $u_{ij}(x) \geq u_{ik}(x)$ for every k. We say that a division x is *envy-free* if there exists a permutation π of $\{1,\dots,n\}$ such that for every i, agent i prefers piece $\pi(i)$. For the computational version of the problem, we will assume that the valuations are given by means of an algebraic circuit as defined in Definition 3.2.3.

Stromquist [209] proved that an envy-free division of the cake is guaranteed to exist, even for the contiguous version, as long as the valuations u_{ij} are continuous functions and the agents are *hungry*, that is no agent prefers an empty piece of cake.

Theorem 3.5.1 (Stromquist [209]). *When the valuations are continuous and the agents are hungry, an envy-free division of the cake always exists.*

Stromquist's proof considers the simplex of divisions as described above, and applies a variant of the K-K-M lemma of Knaster et al. [156] (see Lemma 5 in Section 3.5.2), which regards the covering of the simplex by sets. While this lemma is defined on a continuous domain (the unit-simplex), Stromquist [209] actually applied it after constructing a subdivision of the simplex into "cells", first obtaining the existence of an *approximately* envy-free division, and then invoked a limit argument to prove the existence for *exact* envy-freeness. Another proof of existence was developed independently by Simmons in 1980 (published for the first time in [212] and attributed to Simmons as "private communication to Michael Starbird"). This proof made use of the well-known Sperner's lemma from topology [204], and therefore also works on a subdivision (a "triangulation") of the unit-simplex. Similarly to the proof of Stromquist [209], the proof first establishes the existence of an approximately envy-free division, and then applies a limit argument to obtain the existence for the exact version.

Given their nature as described above, these existence proofs cannot be turned into a membership in FIXP. Indeed, the related literature has only gone as far as proving the membership of the approximate version of the problem in the class PPAD of Papadimitriou [182], a result due to Deng et al. [76]. PPAD is fundamentally related to an approximate computational version of Brouwer's fixed point theorem, and in that sense can be seen as a discrete, approximate analogue of FIXP. Before our paper, the complexity of the *exact* envy-free cake-cutting problem was not known. To this end, we provide the following theorem.

Theorem 3.5.2. *The envy-free cake cutting problem with very general valuations is* FIXP*-complete.*

In order to obtain the FIXP-membership result above, in the process we (implicitly) develop a new existence proof for the envy-free cake cutting problem, one which is not based on discrete subdivisions of the unit-simplex and limit arguments. Our proof is based on maximum flow computation on a network given by a bipartite graph with the agents on one side and the pieces of cake on the other side, as given by a division x. Using our OPT-gate from Section 3.3, we can turn this existence proof into a FIXP-membership result rather easily. Our approach is somewhat reminiscent of the only markedly different proof for envy-free cake cutting that we know of, that of Woodall [229]. This proof constructs a similar bipartite graph and uses Brouwer's fixed point theorem to prove the result, but crucially, it uses a discontinuous step (see Stage 2 of the construction in [229]) which impedes its applicability as a potential argument for a FIXP-membership proof.

Remark 4 (Very general valuations)**.** The statement of Theorem 4.8.2 regards the case of *very general* agents' valuations. Note that according to the definition of the problem in the beginning of the section, an agent has a possibly different valuation for each possible division of the cake. In this generality, the setting captures scenarios in which an agents' value for a piece is not necessarily the sum of her values for the subpieces that comprise the piece (i.e., the valuations are not necessarily *additive*), or even where an agent's value for a piece could be smaller for her value for a subset of that piece (i.e., the valuations are not necessarily *monotone*), and also scenarios that exhibit *externalities*, as an agent might value differently two allocations that assign her the same piece. In several fair division textbooks (e.g., see [38, 190]), the problem is typically presented for the case where the valuations are simply additive measures. On the other hand, the aforementioned existence proofs apply to the case of very general valuations.

Ideally of course, we would like to obtain a FIXP-hardness result for the version of the problem with additive valuations, and this is in fact a major open problem in the literature of computational fair division. That being said, our FIXP-hardness result is in fact in line with the PPAD-hardness result of Deng et al. [76] for the approximate version of the problem, which only holds for very general valuations, in the same generality as we have defined them here. Besides, our focus in this paper is primarily establishing the FIXP-membership of interesting problems via our newly introduced technique in Section 3.3, and clearly the FIXP-membership result is stronger for the case that we consider, i.e., that of very general valuations.

Before we proceed with the proof of Theorem 4.8.2, we remark that in Section 3.5.2, we prove that the computational versions of several well-known topological lemmas and theorems, very much related to the cake cutting problem and its generalizations are also FIXP-complete. In particular, we show that the computational versions of the K-K-M lemma of Knaster et al. [156] that we mentioned above, its "rainbow" generalization due to Gale [105], as well as a "rainbow" generalization of Brouwer's fixed point theorem due to Bapat [19] are also FIXP-complete. These results could be useful for showing FIXP-completeness for more general versions of the envy-free cake cutting problem, or other "multi-label" problems in fair division (e.g., see [3]).

3.5.1 Envy-free cake cutting is FIXP-complete – The proof of Theorem 4.8.2

FIXP-membership

We start with showing the membership of the problem in FIXP. Given a division $x \in \Delta^{n-1}$ of the cake, consider the bipartite graph with agents on the left and pieces of cake on the right and an edge between agent i and piece j if and only if agent i prefers piece j. It is clear that x is an envy-free division if and only if this bipartite graph admits a perfect matching.

Construction of the circuit. Using the above idea we construct a circuit whose fixed points are envy-free divisions. The domain of the circuit is Δ^{n-1}, namely the set of all possible divisions of the cake into n pieces.

We now construct $F\colon \Delta^{n-1} \to \Delta^{n-1}$. On input $x \in \Delta^{n-1}$, the circuit F outputs $\overline{x} \in \Delta^{n-1}$, which is computed as follows. First, for each agent i, we compute capacities $c_i \in \Delta^{n-1}$ for the edges incident on i in the bipartite graph by solving the following LP using the OPT-gate:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} z_j \cdot u_{ij}(x) \\ \text{subject to} \quad & \sum_{j=1}^{n} z_j = 1 \\ & z \geq 0 \end{aligned}$$

Next, we compute a maximum flow $y \in ([0,1]^n)^n$ in the bipartite graph with edge-capacities given by the previously computed c_i's by solving the following LP using the OPT-gate:

$$\begin{aligned} \text{maximize} \quad & \sum_{1 \le i,j \le n} z'_{ij} \\ \text{subject to} \quad & 0 \le z'_{ij} \le c_{ij} + \frac{1}{n^3}, \forall i,j \\ & \sum_{i=1}^{n} z'_{ij} \le 1, \forall j \\ & \sum_{j=1}^{n} z'_{ij} \le 1, \forall i \end{aligned}$$

Finally, the circuit computes $r_k = \max(0, 1 - \sum_{i=1}^{n} y_{ik})$ for every k. Then the output $\overline{x} \in \Delta^{n-1}$ of the circuit is computed as follows

$$\overline{x} := \Big(\frac{x_j + r_j}{1 + \sum_{k=1}^{n} r_k} \Big)_{1 \le j \le n}$$

We note that both linear programs satisfy the explicit Slater condition (Definition 3.3.3) that is required for the OPT-gate to output an optimal solution by Theorem 3.3.1. In particular, for the second LP this is ensured by the nonzero term "$1/n^3$" in the first constraint. Furthermore, the term $1/n^3$ is picked to be sufficiently small so that the arguments in the proof go through (namely, Lemma 3). Any sufficiently small nonzero quantity would also work.

Fixed points. Suppose that x is a fixed point of F, i.e., $x = \overline{x}$, where $\overline{x}$ is computed as described above. In the following we argue that x is an envy-free division. First, we argue that it is sufficient to show that the total flow y is n.

Lemma 3. *If the total flow $(y_{ij})_{i,j}$ is n, then x is an envy-free division.*

Proof. We argue by contradiction, so suppose that x is not an envy-free division. This means that there is no perfect matching in the bipartite graph described at the beginning of this subsection. For a set of agents A, let $N(A)$ denote the set of pieces that is preferred by some agent $i \in A$. By Hall's theorem [134], there must exist some set of agents A with $|N(A)| < |A|$. In particular the y-flow from A to $N(A)$ is at most $|A| - 1$.

Now note that the first linear program implies that $c_{ij} = 0$ for any $i \in A$ and $j \notin N(A)$, because piece j is not preferred by agent i (i.e., there exists piece j' such that $u_{ij'}(x) > u_{ij}(x)$), and hence we always have $z_j = 0$ at any optimal solution of this LP. As a result, by the first constraint of the second linear program, $y_{ij} \le 1/n^3$ for any $i \in A$ and $j \notin N(A)$. We conclude that the total flow from A to $\{1,\dots,n\} \setminus N(A)$ is bounded by $\frac{1}{n^3} \cdot |A| \cdot |\{1,\dots,n\} \setminus N(A)| \le \frac{n^2}{n^3} < 1$. Since, as shown above, the flow from A to $N(A)$ is at most $|A| - 1$, it follows that the total flow out of A is less than $|A|$. We conclude that the total flow is strictly less than n. □

By Lemma 3, it now suffices to show that the total flow is n. This is the same as saying that $r_j = 0$ for all j. Since $x = \overline{x}$, we have that $x_j = (x_j + r_j)/(1 + \sum_{k=1}^{n} r_k)$ for all j. This implies that

$$x_j \cdot \sum_{k=1}^{n} r_k = r_j \text{ for all } j. \tag{3.11}$$

We have the following lemma.

Lemma 4. *There exists some j with $r_j = 0$ and $x_j > 0$.*

Proof. Suppose towards a contradiction that for all j we have that $r_j > 0$ or $x_j = 0$. Because $x_j > 0$ for some j, this means that the total flow is less than n. Therefore, there exists some agent i whose out-flow is less than 1. We claim that there exists some $j \in N(i)$ with $r_j = 0$. Suppose towards a contradiction that $r_j > 0$ for all $j \in N(i)$. Then we can send more flow along the edge (i, j) for some $j \in N(i)$, because $\sum_{k=1}^{n} y_{ik} < 1$ and $\sum_{k \in N(i)} c_{ik} = 1$. This contradicts the fact that y is an optimal solution to the second second linear program. Hence, there exists some $j \in N(i)$ with $r_j = 0$. This concludes the proof, since $x_j > 0$ holds by the hungriness-condition. □

Combining Lemma 4 and Equation (3.11), we conclude that $r_k = 0$ for every k. Hence, the total flow is n, and it follows from Lemma 3 that x is an envy-free division. Hence, we have shown the following.

Proposition 2. *The envy-free cake cutting problem with very general valuations is in* FIXP.

FIXP-hardness

We now turn our attention to showing FIXP-hardness. Since we are considering very general valuations, the FIXP-hardness result is rather straightforward. For the proof, we will use the following generalization of Brouwer's fixed point theorem due to Bapat [19], which we refer to as *Bapat's Brouwer fixed point theorem.*

Theorem 3.5.3 (Bapat's Brouwer fixed point theorem [19])**.** *Let $f_i \colon \Delta^{n-1} \to \Delta^{n-1}$ be continuous functions for $i = 1, \ldots, n$. There exists some $z \in \Delta^{n-1}$ and a permutation π of $\{1, \ldots, n\}$ such that $z_{\pi(i)} \geq f_{i,\pi(i)}(z)$ for all i.*

Theorem 3.5.3 implies Brouwer's fixed point theorem by choosing all the f_i's to be equal. The computational version of the problem is defined similarly to the computational version of Brouwer's fixed point theorem (see Definition 4.2.5). Since Bapat's version is a generalization of Brouwer's fixed point problem, its FIXP-hardness is immediate.

Let continuous functions $f_1, \ldots, f_n \colon \Delta^{n-1} \to \Delta^{n-1}$ be given, represented by algebraic circuits. To show FIXP-hardness of the cake cutting problem, we define an instance with n players such that an envy-free division corresponds to a solution for the Bapat problem given by $f_1,\dots,f_n$. For our definition of the valuations of the agents to satisfy the hungriness-condition, we need to make an assumption about the f_i, namely that $f_{ij}(x) \geq \frac{1}{2n}$ for all $i, j \in [n]$ and $x \in \Delta^{n-1}$. We first show that we may assume this without loss of generality.

Preprocessing. We now show how to make new functions $g_1,\dots,g_n\colon \Delta^{n-1} \to \Delta^{n-1}$ such that 1) a solution to the Bapat problem for $g_1,\dots,g_n$ easily translates into a solution for $f_1,\dots,f_n$, and 2) the functions satisfy $g_{ij}(x) \geq \frac{1}{2n}$ for all $i, j \in [n]$ and $x \in \Delta^{n-1}$. Let $x \in \Delta^{n-1}$ be given. As in [85], we may use a sorting network to compute a value $t \geq 0$ such that $\sum_{i=1}^{n} \max(x_i - t, \frac{1}{2n}) = 1$. We now define $\pi(x)$ by $\pi_j(x) = \max(x_j - t, \frac{1}{2n})$ and $g_{ij}(x) = \frac{1}{2} f_{ij}(2(\pi(x) - \frac{1}{2n})) + \frac{1}{2n}$. Note the following:

(i) For all $i, j \in [n]$, $g_{ij}(x) \geq \frac{1}{2n}$.

(ii) If x is a Bapat solution for $g_1,\dots,g_n$, then $2(x - \frac{1}{2n})$ is a solution for $f_1,\dots,f_n$. Suppose that x is a solution for $g_1,\dots,g_n$. Then there is a permutation π such that $x_{\pi(i)} \geq g_{i,\pi(i)}(x)$. In particular, we get from (i) that $x_{\pi(i)} \geq \frac{1}{2n}$ for all i. This implies that the value t computed by the sorting network is $t = 0$, which implies that $\pi(x) = x$. Now the inequalities

$$x_{\pi(i)} \geq g_{i,\pi(i)} = \frac{1}{2} f_{i,\pi(i)}\left(2\left(\pi(x) - \frac{1}{2n}\right)\right) + \frac{1}{2n} = \frac{1}{2} f_{i,\pi(i)}\left(2\left(x - \frac{1}{2n}\right)\right) + \frac{1}{2n}$$

for all i, imply that $2(x_{\pi(i)} - \frac{1}{2n}) \geq f_{i,\pi(i)}(2(x - \frac{1}{2n}))$ for all i. This says that $2(x - \frac{1}{2n})$ is a solution to the Bapat problem for $f_1,\dots,f_n$.

Now (i) says that the g_i satisfy the required inequalities and (ii) says that given any solution for $g_1,\dots,g_n$, we may (very easily) compute a solution for $f_1,\dots,f_n$.

Reduction to cake cutting. Due to the preprocessing step, we may assume that the functions $f_1,\dots,f_n$ satisfy $f_{ij}(x) \geq \frac{1}{2n}$ for all $i, j \in [n]$ and $x \in \Delta^{n-1}$. The valuation of player i for the j-th piece is defined to be $u_{ij}(x) = \max(0, x_j - f_{ij}(x))$ for any point $x \in \Delta^{n-1}$. We now show that the agents are hungry. Suppose that $x \in \Delta^{n-1}$ with $x_j = 0$ for some j. This implies that $u_{ij}(x) = 0$ for all j. As $x_j = 0 < \frac{1}{2n} \leq f_{ij}(x)$ and $x, f_i(x) \in \Delta^{n-1}$, we get that there exists some k with $f_{ik}(x) < x_k$. This implies that $u_{ik}(x) > 0 = u_{ij}(x)$. In particular, no agent prefers an empty piece.

If x is an envy-free division and π is the corresponding permutation, we obtain for any i that

$$\max\left(0, x_{\pi(i)} - f_{i,\pi(i)}(x)\right) \geq \max\Big(0, x_j - f_{ij}(x)\Big) \text{ for all } j, \tag{3.12}$$

because of the inequalities $u_{i,\pi(i)}(x) \geq u_{ij}(x)$ for all i, j. We claim that $x_{\pi(i)} \geq f_{i,\pi(i)}(x)$ for all i. Suppose that there exists some i for which this is not the case, i.e., $x_{\pi(i)} < f_{i,\pi(i)}(x)$. This implies that the left-hand side of **??** 3.12 is equal to 0. However, we also have that

$$x_j, f_{ij}(x) \geq 0, \quad x_{\pi(i)} < f_{i,\pi(i)}(x), \quad \text{and} \quad \sum_{j=1}^{n} x_j = \sum_{j=1}^{n} f_{ij}(x) = 1,$$

which implies that $x_j > f_{ij}(x)$ for some j. This implies that the right-hand side of **??** 3.12 is strictly positive, leading to a contradiction. From this, we obtain the following.

Proposition 3. *The envy-free cake cutting problem with very general valuations is* FIXP-*hard.*

3.5.2 The K-K-M Lemma

As we mentioned in the beginning of the section, our technique also has implications to the FIXP-membership of some known topological problems which are related to the cake cutting problem, in the sense that they have been, or can be used to prove the existence of an envy-free division. Here, we present those results.

The *Knaster–Kuratowski–Mazurkiewicz (K-K-M) lemma* [156] is a basic result concerning certain covers of the unit simplex by closed sets, related to Brouwer's fixed point theorem. First we present the definition of a *K-K-M covering* and then the statement of the lemma.

Definition 3.5.1 (K-K-M covering)**.** Let $T_1, \ldots, T_n \subseteq \mathbb{R}^n$ be a collection of closed sets. We say that $T_1, \ldots, T_n$ form a *K-K-M covering* of Δ_{n-1} if $conv(\{e_i\colon i \in S\}) \subseteq \cup_{i \in S} T_i$, for any set $S \subseteq [n]$.

Lemma 5 (K-K-M lemma [156])**.** *Let $T_1, \ldots, T_n$ be a K-K-M covering of Δ_{n-1}. Then $\cap_{i=1}^{n} T_i \neq \emptyset$.*

We can derive a corresponding total search problem that applies to any collection $T_1, \ldots, T_n$ of closed sets as follows.

Corollary 1. *Let $T_1, \ldots, T_n \subseteq \mathbb{R}^n$ be a collection of closed sets. Then there exists $x \in \Delta_{n-1}$ such that exactly one of the following conditions holds.*

1. $x \in \cap_{i=1}^{m} T_i$
2. $x \notin \cup_{i \in \mathrm{Supp}(x)} T_i$

In case the second condition never holds we say that the sets $T_1, \ldots, T_n$ satisfy the K-K-M condition.

Proof. Let us note that for any $x \in \Delta_{n-1}$ we have that $x = \sum_{i \in \mathrm{Supp}(x)} x_i e_i \in \mathrm{Conv}(\{e_i\colon i \in \mathrm{Supp}(x)\})$. Thus, if the second condition does not hold for any $x \in \Delta_{n-1}$ it follows that $T_1, \ldots, T_n$ form a K-K-M covering. By the K-K-M lemma we have $\cap_{i=1}^{n} T_i \neq \emptyset$, which means that there exist $x \in \Delta_{n-1}$ satisfying the first condition. □

Knaster et al. [156] gave a proof of Brouwer's fixed point theorem from the K-K-M lemma, whereas Gale [106] conversely gave a proof of the K-K-M lemma from Brouwer's fixed point theorem. To adapt these proofs to a computational setting, we need to settle on the generality of closed sets we would like to consider and how to represent those. It is natural to restrict attention to *closed semi-algebraic sets* definable in the first order theory of the reals using polynomials with integer coefficients.

Definition 3.5.2 (Basic closed semi-algebraic set)**.** A set $S \in \mathbb{R}^n$ is a *basic closed semi-algebraic set* if there exists polynomials $P_1, \ldots, P_k$ such that $S = \{x \in \mathbb{R}^n \mid \wedge_{i=1}^k P_i(x) \geq 0\}$.

Any closed semialgebraic set S is a finite union of basic closed semialgebraic sets, see e.g. [31], and if S is definable using polynomials with integer coefficients, this holds also for the basic closed semialgebraic sets whose union is S. Disregarding complexity concerns, all such sets can be represented as an inverse image $F^{-1}(0)$ of a nonnegative function F computed by an algebraic circuit as shown in the following lemma.

Lemma 6. *Let $S \subseteq \mathbb{R}^n$ be a closed semialgebraic set definable using polynomials with integer coefficients. Then there exists an algebraic circuit C computing a nonnegative function F such that $S = \{x \in \mathbb{R}^n \mid F(x) = 0\}$.*

Proof. We may write S as a union $S = \cup_{i=1}^s S_i$ of basic closed semialgebraic sets $S_i = \{x \in \mathbb{R}^n \mid \wedge_{j=1}^{k_i} P_{ij}(x) \geq 0\}$, where each polynomial P_{ij} have integer coefficients.

Note that $S_{ij} = \{x \in \mathbb{R}^n \mid P_{ij}(x) \geq 0\} = \{x \in \mathbb{R}^n \mid \max(0, -P_{ij}(x)) = 0\}$, which means in particular that S_{ij} may be represented by as the inverse image $F_{ij}^{-1}(0)$ of a nonnegative function F_{ij} computed by an algebraic circuit. We complete the proof by showing that the collection of sets representable in this way is closed under intersection and union.

Suppose F_1 and F_2 are nonnegative functions computed by algebraic circuits and let $S_1 = F_1^{-1}(0)$ and $S_2 = F_1^{-1}(0)$. Then the two functions $\max(F_1(x), F_2(x))$ and $\min(F_1(x), F_2(x))$ are nonnegative functions computed by algebraic circuits that represents $S_1 \cap S_2$ and $S_1 \cup S_2$, respectively. □

We note that if S is any closed set, then the distance $d(x, S)$ is a nonnegative continuous function that assumes the value 0 precisely in the set S. We may thus think of F as a stand-in for the distance $d(x, S)$ from x to S.

We can now define a search problem based on Corollary 1 and the representation of closed semialgebraic sets used in Lemma 6.

Definition 3.5.3 (K-K-M problem)**.** Given algebraic circuits $C_1, \ldots, C_n$ computing non-negative functions $F_1, \ldots, F_n : \mathbb{R}^n \to \mathbb{R}$, a point $x \in \Delta_{n-1}$ is a solution of the associated K-K-M problem if one of the following conditions holds.

1. $\forall i : F_i(x) = 0$.
2. $\forall i : x_i > 0 \Rightarrow F_i(x) > 0$.

In case the second condition never holds we say that the functions $F_1, \ldots, F_n$ satisfy the K-K-M condition.

Let us note that while the condition that the functions F_i are non-negative is a semantic condition, it can be enforced syntactically when computed by algebraic circuits by considering either $(F_i(x))^2$ or $\max(F_i(x), 0)$ depending on whether we wish to represent the set $F_i^{-1}(0)$ or the set $\{x \in \mathbb{R}^n \mid F_i(x) \leq 0\}$.

We can now adapt the proof by Gale [106] to show that the K-K-M problem is in FIXP, and the proof of Knaster et al. [156] to show that the K-K-M problem is also FIXP-hard. We have the following theorem.

Theorem 3.5.4. *The K-K-M problem is* FIXP-*complete.*

Proof. We first prove FIXP-membership of the K-K-M problem. Define the function $G : \Delta_{n-1} \to \Delta_{n-1}$ by

$$G(x)_i = \frac{x_i + F_i(x)}{1 + \sum_{j=1}^n F_j(x)}$$

Given algebraic circuits for all functions F_i, an algebraic circuit computing G can clearly be constructed in polynomial time. Suppose now that $x \in \Delta_{n-1}$ such that $G(x) = x$. It follows that $x_i \sum_{j=1}^n F_j(x) = F_i(x)$ for all i. If $F_i(x) = 0$ for all i, x is a solution satisfying the first condition of the K-K-M problem. Otherwise, suppose there is i such that $F_i(x) > 0$. Then $\sum_{j=1}^n F_j(x) > 0$ as well and it follows that

$$x_i = \frac{F_i(x)}{\sum_{j=1}^n F_j(x)}$$

for all i. Hence $x_i > 0$ implies $F_i(x) > 0$, for all i, and x is a solution satisfying the second condition of the K-K-M problem.

To prove FIXP-hardness we reduce from the basic FIXP problem with domain Δ_{n-1}. Suppose that $G : \Delta_{n-1} \to \Delta_{n-1}$ is a continuous function computed by an algebraic circuit. We may then in polynomial time construct algebraic circuits computing the functions $F_i(x) = \max(0, G(x)_i - x_i)$, for $i = 1, \dots, n$. We claim that these functions satisfy the K-K-M property. Suppose for the contrary that $x \in \Delta_{n-1}$ such that $F_i(x) > 0$ whenever $x_i > 0$. Letting $S = \operatorname{Supp} x$ we then have $G(x)_i > x_i$ for all $i \in S$, leading to the contradiction

$$1 = \sum_{i \in S} x_i < \sum_{i \in S} G(x)_i \leq \sum_{i=1}^n G(x)_i = 1 \quad .$$

Thus if $x \in \Delta_{n-1}$ is a solution of the K-K-M problem it satisfies that $F_i(x) = 0$, for all i, which implies that $x_i \geq G(x)_i$, for all i. Since also $G(x) \in \Delta_{n-1}$ it follows that in fact $x_i = G(x)_i$, for all i, which means that x is a fixed point of G. □

The rainbow K-K-M lemma and Bapat's Brouwer fixed point generalization

Several years after his proof for the K-K-M lemma in [106], Gale [105] also proved a generalization of the K-K-M lemma, commonly referred to as the *rainbow K-K-M lemma*.[7]

[7]Sometimes in the literature the term "colorful" is used instead of "rainbow".

Lemma 7 (Rainbow K-K-M lemma [105]). *Let $T^n_{i,j=1} \subseteq \mathbb{R}^n$ be a collection of closed sets such that for every i, the collection $T_{i,1}, \ldots, T_{i,n}$ form a K-K-M covering of Δ_{n-1}. Then there exists a permutation π of $[n]$ such that $\cap_{i=1}^n T_{i,\pi(i)} \neq \emptyset$.*

Bapat [19] provided a proof of the rainbow K-K-M lemma via the introduction of a generalization of Sperner's lemma. In the same paper, he also provided his generalization of Brouwer's fixed point theorem (Theorem 3.5.3), which we used in the proof of Proposition 3. Our reduction in Section 3.5.1 also proves the membership of the computational version of Bapat's Brouwer fixed point theorem in FIXP (henceforth *Bapat's Brouwer fixed point problem*), since envy-free cake cutting is in FIXP by Proposition 2.

The reduction of Theorem 3.5.4 from the K-K-M problem to the computational version of Brouwer's fixed point theorem generalizes immediately to the case of reducing from the rainbow K-K-M version (henceforth the *rainbow K-K-M problem*) to Bapat's Brouwer fixed point problem, thus establishing the FIXP-membership of the rainbow K-K-M problem as well. In fact Bapat's proof uses the rainbow K-K-M lemma in the same way as Knaster et al. [156] do to prove Brouwer's fixed point theorem from the K-K-M lemma. Therefore, we have the following theorem.

Theorem 3.5.5. *The rainbow K-K-M problem and Bapat's Brouwer fixed point problem are* FIXP*-complete.*

3.6 Applications to Markets

In this section, we show how our main technique can be used to prove the FIXP-membership of computing equilibria in competitive markets. Our first result here is a rather general one, namely that the problem of computing a market equilibrium in Arrow-Debreu markets with concave utilities is in FIXP. This generalizes previously known results, on markets with specific utility functions [117] but at the same time is conceptually easier, as long as our OPT-gate is used as a black box. We state the main theorem below.

Theorem 3.6.1. *The problem of computing a market equilibrium in an Arrow-Debreu market with concave utilities is in FIXP.*

Our second result regards the problem of computing an equilibrium in the pseudomarket mechanism of Hylland and Zeckhauser [142], which was shown to be in FIXP quite recently by Vazirani and Yannakakis [221]. We obtain the same result, via a conceptually simpler proof based on our general technique.

Theorem 3.6.2. *The problem of computing an equilibrium of the Hylland-Zeckhauser mechanism is in* FIXP.

The crucial ingredient that allows us to obtain proofs which are conceptually simpler and often more general is in the utility-maximizing optimization program of the agents, which appears in the corresponding proofs of existence. Generally speaking, given a set of prices, an agent computes a utility-maximizing allocation given a set of constraints on consumption and endowment. Then, market clearing is ensured by the fixed point condition on the prices. Using our OPT-gate, we can directly "simulate" the utility-maximization program in these proofs, which makes our membership results look very similar to the existence proofs themselves, which rely on some fixed point correspondence theorem, most often Kakutani's fixed point theorem (Theorem 3.2.2). Note however that proving FIXP-membership using the OPT-gate implicitly also yields a proof of existence based on Brouwer's fixed point theorem (Theorem 3.2.1).

3.6.1 Arrow-Debreu Markets

The fundamental model of competitive markets was established in the pioneering work of Arrow and Debreu [8], formalizing some ideas of Walras [225]. Arrow and Debreu showed that under relatively mild assumptions, every market has an equilibrium, i.e., a set of prices and allocations such that supply equals demand, and every market participant is maximally satisfied with their assigned commodities at the given prices. We present the formal setting of the Arrow-Debreu market below, following closely their original paper. When $<$ and $\leq$ are used for vectors, they are applied componentwise.

Arrow-Debreu Market. An Arrow-Debreu market consists of

- A set L of *commodities* (i.e., divisible resources or items); let $\ell = |L|$.
- A set N of *production units* or *firms*; let $n = |N|$.

  For each firm $j \in N$, there is a set Y_j of possible *production plans*. An element $y_j \in Y_j$ is a vector in $\mathbb{R}^\ell$, i.e., $y_j = (y_{j,1}, y_{j,2}, \dots, y_{j,\ell})$, where $y_{j,h}$ is the output (i.e., the produced amount) of commodity h according to plan y_j. The production can also be negative, where $y_{j,h} < 0$ is interpreted as the commodity h being an input to the production plan, rather than an output. Let $Y = \sum_{j=1}^{n} Y_j$.

  The production sets Y_j satisfy the following assumptions:

  I.a. For all $j \in N$, Y_j is a closed convex subset of $\mathbb{R}^\ell$ containing 0, *(non-increasing returns to scale)*

  I.b. $Y \cap \mathbb{R}^\ell_{\geq 0} = \{0\}$, *(no output without input)*

  I.c. $Y \cap (-Y) = \{0\}$. *(irreversible production)*

- A set M of *consumption units* or *consumers*; let $|M| = m$.

  For each consumer $i \in M$, there is a set X_i of possible *consumption vectors*, indicating the set of resources that a consumer could possibly consume, if there were no budgetary constraints. We use $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,\ell})$ to denote the consumption of agent i, where $x_{i,h}$ denotes the consumer's consumption of commodity h. The consumption can also be negative, where $x_{i,h} < 0$ is interpreted as a labor service provided by the consumer to the market. The consumption sets X_i satisfy:

II. For all $i \in M$, X_i is a closed, convex subset of $\mathbb{R}^\ell$ which is bounded from below, i.e., there is a vector ξ_i such that $\xi_i \leq x_i$ for all $x_i \in X_i$.

Each consumer has a *utility function* $u_i \colon X_i \to \mathbb{R}$, that satisfies the following properties:

III.a. $u_i(x_i)$ is a continuous function on X_i, *(continuity)*

III.b. For any $x_i \in X_i$, there is an $x'_i \in X_i$ such that $u_i(x'_i) > u_i(x_i)$, *(non-satiation)*

III.c. If $u_i(x_i) > u_i(x'_i)$ and $0 < t < 1$, then $u_i\left(tx_i + (1-t)x'_i\right) > u_i(x'_i)$. *(convexity of preferences)*

Each consumer $i \in M$ is also endowed with a vector $\zeta_i = (\zeta_{1,i}, \zeta_{2,i}, \dots, \zeta_{\ell,i}) \in \mathbb{R}^\ell$ of initial holdings of the different commodities, which we will refer to as the *endowment* of consumer i. The following assumption is made:

IV.a. For all $i \in M$, there exists some $x_i \in X_i$ with $x_i < \zeta_i$.

Additionally, consumer i has a *share* α_{ij} of the profit of the j-th production unit, for each $j \in N$. Shares are non-negative and the profits are entirely shared among the consumers, i.e.,

IV.b. For all $i \in M$ and $j \in N$, $\alpha_{ij} \geq 0$; for all $j \in N$, $\sum_{i=1}^{m} \alpha_{ij} = 1$.

Before we proceed, we provide a brief discussion on some of the assumptions of the model above.[8] For the production plans, Assumption I.a. corresponds to non-increasing returns to scale, i.e., when all production variables are increased by an amount, the output is increased by an "at-most-proportional" amount. Assumption I.b. states that it is not possible for the production to have output without having some input. Finally, Assumption I.c. asserts that it is not possible to have two production vectors that exactly "cancel" each other, since some labor is necessary for production, and labor cannot be produced. For the consumers' utilities, Assumption III.a. is a standard assumption in consumer and market theory. Assumption III.b. asserts that there is no "saturation point", i.e., a consumption vector that the consumer prefers to all others. A related sufficient condition would be that the utility function is *strictly monotone*, although non-satiation is a weaker condition which is still sufficient for the existence of a market equilibrium by Theorem 3.6.3. Assumption III.c. is a standard assumption on the convexity of the indifference curves. Assumption IV.a. asserts that each consumer could feasibly consume from their endowment and still have leftover stock to trade in the market. Finally, Assumption IV.b. simply states that the profits

[8]The reader is referred to the original paper by Arrow and Debreu [8] for an extensive discussion on the assumptions of the model.

are non-negative and are shared entirely among the consumers, i.e., there are no profits supplied to outside entities.

At the center of competitive markets is the concept of *competitive* or *market equilibrium*, i.e., a set of prices, production plans and consumption vectors such that supply equals demand and all agents maximize their individual utility at the given set of prices. We provide the formal definition below.

Definition 3.6.1 (Arrow-Debreu Market Equilibrium)**.** A tuple of vectors $(x_1^*, x_2^*, \dots, x_m^*, y_1^*, y_2^*, \dots y_n^*, p^*)$ is a (Arrow-Debreu) market equilibrium if the following conditions are satisfied:

1. $y_j^* \in \arg\max_{y_j \in Y_j} p^* \cdot y_j$, for all $j \in N$, *(firm profit maximization)*

2. $x_i^* \in \arg\max_{x_i \in S_i} u_i(x_i)$, where *(consumer utility maximization)* $S_i = \{x_i | x_i \in X_i, p^* \cdot x_i \leq p^* \cdot \zeta_i + \sum_{j=1}^n \alpha_{ij} p^* \cdot y_j^*\}$, for all $i \in M$,

3. $p^* \in P = \{p | p \in \mathbb{R}^\ell, p \geq 0, \sum_{h=1}^\ell p_h = 1\}$, *(non-negative, normalized prices)*

4. $z^* \leq 0$ and $p^* \cdot z^* = 0$, where $z^* = \sum_{i=1}^m x_i^* - \sum_{i=1}^n y_j^* - \sum_{i=1}^m \zeta_i$. *(market clearance)*

Again, we briefly discuss the four conditions of the Arrow-Debreu market equilibrium definition above. Condition 1 requires that at the given set of prices, the firms maximize their profits within their production plans. Condition 2 requires that at the given set of prices, the consumers maximize their utility within their consumption vectors that satisfy their budget constraint. Condition 3 stipulates that the prices are non-negative, and can be normalized to sum to 1 without loss of generality. Finally, Condition 4 is the market clearing, the "supply equals demand" condition. The condition states that (a) the total consumption of each commodity minus the sum of the total production and the consumers' endowment of that commodity has to be non-positive (i.e., consumers cannot consume more than what is available) and (b) this difference is actually zero (i.e., consumers consume exactly what is available), for commodities for which the price is non-zero. Commodities which are priced at zero are allowed to not be entirely consumed. We remark that in the related literature (e.g., see [166, Chapter 10]), Condition 4 is often replaced by the strong condition of "supply equals demand" for all commodities, regardless of the price, namely

$$\sum_{i=1}^m x_{i,h}^* = \sum_{i=1}^n y_{j,h}^* + \sum_{i=1}^m \zeta_{i,h}, \quad \text{for } h = 1, \dots, \ell.$$

Then, in order to guarantee the existence of a market equilibrium as in Theorem 3.6.3, the extra assumption of *free disposability* needs to be added to the list of Assumptions I.a. to I.c. for the production sets (e.g., see [166, Chapter 5]), namely:

I.d. $Y - \mathbb{R}_{\geq 0}^\ell \subseteq Y$. *(free disposability)*

The property asserts that it is always possible to absorb additional inputs without producing any extra output (i.e., the inputs can be "freely disposed"). Our approach still works even if one requires this stronger clearing condition, see Remark 6.

Arrow and Debreu [8] proved that for any market as defined above, a market equilibrium always exists.

Theorem 3.6.3 ([8])**.** *Every instance of the market above admits a market equilibrium.*

One important observation is that Conditions 1 and 2 are essentially convex programs whereas Conditions 3 and 4 are essentially sets of constraints. Using our OPT-gate from Section 3.5.2, we can "substitute" these optimization programs by pseudogates, effectively transforming an existence proof into a FIXP-membership proof. Indeed, the proof of Theorem 3.6.3 as presented in [8] uses these conditions essentially as optimization programs to construct a fixed point correspondence for the consumption, the production and the prices. Then the existence of an equilibrium is proven via a fixed point theorem due to Debreu [68]. Subsequent proofs used Kakutani's fixed point theorem (Theorem 3.2.2) to obtain the same existence result. Our FIXP-membership result essentially devises a proof via Brouwer's fixed point theorem instead, similar in that regard to the proof of Geanakoplos [122].

Our result in Theorem 3.6.1 generalizes some membership results that were already presented in the literature, for markets with specific utility functions, production and consumption sets. Etessami and Yannakakis [85] proved the FIXP-membership of a setting where there are no explicit utilities, and the aggregate demand is a given function, rather than a correspondence which is typically the case in these markets. Garg et al. [117] provided a FIXP-membership result for Arrow-Debreu markets with *Piecewise Linear Concave* (PLC) utility functions, and PLC production sets. The authors consider straightforward consumption sets, namely that the consumption of an agent is non-negative. For the formal definition of PLC functions, we refer the reader to Section 3.7.

Remark 5. Besides the FIXP-membership results mentioned above, Chen et al. [56] proved a FIXP-membership result for markets with CES utilities, only non-negativity constraints on consumption and no production. While these are a special case of the Arrow-Debreu market that we consider, strictly speaking, our result does not generalize theirs. This is exclusively due to technical reasons, which we highlight below.

As we explain in Section 3.6.1 below, we assume that access to the utility functions is given to us via the supergradients of those functions; this is clearly necessary when dealing with general concave functions without any additional structure. In the case of explicit utility functions however, like PLC utilities or CES utilities, ideally we would like to be able to compute the supergradients given the utility functions, rather than assume access to them. For PLC utilities (and actually much more general utility functions) we can do that, and we show how in Section 3.7. In that sense, our FIXP-membership result is a strict generalization of the corresponding membership result proven in [117]. On the other hand, CES utilities are not superdifferentiable when some coordinate is 0, and therefore we cannot claim the same for the results of Chen et al. [56]. We believe that it is possible to adapt our approach in Section 3.6.1 to capture the case of CES utilities, but we leave that for future work.

The computational Arrow-Debreu market problem

In some of the applications that we have presented so far, deriving the corresponding computational problems from the existence theorems has been relatively straightforward, and thus not explicitly stated. In the market domain, due to the generality of the utility functions, as well as the production and consumption sets, some additional discussion is in order.

- For the utility functions u_i, we will assume that we are given a pseudogate computing the supergradients ∂u_i (see Sections 3.3.2 and 3.3.4). This is in a sense necessary, since we are concerned with general concave utility functions, with no further particular structure. In the case of specific utility functions, such as the PLC utilities studied in [117], we can easily construct pseudogates computing the supergradients. In Section 3.7, we explain how to compute the supergradients for a general class of utility functions that subsumes PLC utilities, which we refer to as *Piecewise Differentiable Concave (PDC) functions*, rather than assuming that they are given to us as inputs.

- For the consumption and production sets, we will assume that they are given to us as sets of convex inequalities and linear equations that satisfy the (standard) Slater condition, see Section 3.3.4. Note that here we do not need them to satisfy the explicit Slater condition of Definition 3.3.4; this is because the consumption and production sets are part of the input to the problem, and therefore we can apply preprocessing to the corresponding constraints to eliminate any linear dependence. For the convex inequalities, we also assume that we are given pseudogates computing their subgradients, as explained in Section 3.3.4. We further assume that the endowments ζ_i, the shares α_{ij}, and the lower bounds ξ_i on the consumption are given in the input as rational numbers.

- For the production sets, the assumptions in the Arrow-Debreu model ensure that the set of all $(y_1,\dots,y_n) \in (\mathbb{R}^\ell)^n$ that satisfy

$$y_j \in Y_j, \text{ for } j = 1,\dots,n \quad \text{and} \quad \sum_{i=1}^{m} \xi_i - \sum_{j=1}^{n} y_j - \sum_{i=1}^{m} \zeta_i \leq 0 \tag{3.13}$$

  is bounded. This is proved by Arrow and Debreu [8, pp. 276-277] and, from a technical standpoint, one of the main purposes of the various assumptions on the production sets is to ensure that this property indeed holds. Note that any production plan $(y_1,\dots,y_n)$ that satisfies the market clearing condition must necessarily satisfy the inequality constraint above. Thus, in any market equilibrium, the production plan lies in this bounded set.

For the definition of the computational problem, we assume that the input to the problem contains an explicit bound C on the set above (in the ℓ_∞-norm). This is needed to ensure that solutions can be bounded with some bound that has polynomial bit-complexity. We remark that in the case of the PLC production sets used in [117], the constraint $y \in Y_j$ corresponds to a set of linear equalities and inequalities. As a result, the bound C does not need to be provided in the input in that case, since it can easily be obtained by solving an LP.

Membership in FIXP – the proof of Theorem 3.6.1

Bounding the domain of allocations. Consider any $(x_1,\ldots,x_m) \in (\mathbb{R}^\ell)^m$ and $(y_1,\ldots,y_n) \in (\mathbb{R}^\ell)^n$ that satisfy $x_i \in X_i$ for $i = 1,\ldots,m$, $y_j \in Y_j$ for $j = 1,\ldots,n$, and

$$z := \sum_{i=1}^{m} x_i - \sum_{j=1}^{n} y_j - \sum_{i=1}^{m} \zeta_i \le 0.$$

By Assumption II., we have $\xi_i \le x_i$, and as a result (3.13) is satisfied. It follows that $\|y_j\| \le C$ for $j = 1,\ldots,n$, where $\|\cdot\|$ denotes the ℓ_∞-norm. Now, $z \le 0$ also implies that for any $i \in [m]$

$$x_i \le \sum_{j=1}^{n} y_j + \sum_{k=1}^{m} \zeta_k - \sum_{k \ne i} x_k \le \sum_{j=1}^{n} y_j + \sum_{k=1}^{m} \zeta_k - \sum_{k \ne i} \xi_k$$

where we used $\xi_k \le x_k$ again. Together with $\xi_i \le x_i$, it follows that $\|x_i\| \le nC + m\max_k \|\zeta_k\| + m\max_k \|\xi_k\| =: C'$.

We let $K := C' + 1$. From the above, we have that if $(x_1,\ldots,x_m,y_1,\ldots,y_n)$ satisfies $x_i \in X_i$, $y_j \in Y_j$, and $z \le 0$, then $x_i, y_j \in (-K,K)^\ell$ for all i, j. In particular, this must be satisfied at any equilibrium. Note that K can be computed in polynomial time from the inputs to our problem.

Construction of the circuit. We construct a circuit $F : D \to D$, where $D = ([-K,K]^\ell)^m \times ([-K,K]^\ell)^n \times [0,1]^\ell$. We let $(x_1,\ldots,x_m,y_1,\ldots,y_n,p)$ denote the input to F, and $(\overline{x}_1,\ldots,\overline{x}_m,\overline{y}_1,\ldots,\overline{y}_n,\overline{p})$ denote the output of F.

We set $\overline{y}_j$ as the output of the OPT-gate for the convex optimization problem:

$$\begin{aligned} \text{maximize} \quad & p \cdot v \\ \text{subject to} \quad & v \in Y_j \\ & v \in [-K,K]^\ell \end{aligned} \tag{3.14}$$

We set $\overline{x}_i$ as the output of the OPT-gate for the convex optimization problem:

$$\begin{aligned} \text{maximize} \quad & u_i(v) \\ \text{subject to} \quad & v \in X_i \\ & p \cdot v \le p \cdot \zeta_i + \sum_{j=1}^{m} \alpha_{ij} p \cdot y_j \\ & v \in [-K,K]^\ell \end{aligned} \tag{3.15}$$

The inequality constraint $p \cdot v \leq p \cdot \zeta_i + \sum_{j=1}^m \alpha_{ij} p \cdot y_j$ is called the *budget constraint.*

Finally, we set $\overline{p}$ as the output of the OPT-gate for the LP:

$$\begin{aligned} \text{maximize} \quad & v \cdot z \\ \text{subject to} \quad & \sum_{h=1}^{\ell} v_h = 1 \\ & v \in [0,1]^{\ell} \end{aligned} \tag{3.16}$$

where $z := \sum_{i=1}^m x_i - \sum_{j=1}^n y_j - \sum_{i=1}^m \zeta_i$.

Fixed points. Consider any fixed point of F, i.e., $(x_1, \dots, x_m, y_1, \dots, y_n, p) \in D$ such that

$$(x_1, \dots, x_m, y_1, \dots, y_n, p) = (\overline{x}_1, \dots, \overline{x}_m, \overline{y}_1, \dots, \overline{y}_n, \overline{p}).$$

We begin by showing that the explicit Slater condition holds at $(x_1, \dots, x_m, y_1, \dots, y_n, p)$ for all three optimization problems, and thus they are solved correctly by the OPT-gate. Then, we show that $(x_1, \dots, x_m, y_1, \dots, y_n, p)$ is indeed a market equilibrium, as desired.

Explicit Slater condition. Clearly, the constraints of (3.16) satisfy the explicit Slater condition. As a result, the OPT-gate functions properly and $p = \overline{p}$ is indeed an optimal solution of this LP.

The constraints of the convex optimization problem (3.14) also satisfy the explicit Slater condition. To see this, note that, by assumption, the constraints describing Y_j satisfy the explicit Slater condition, i.e., there exists $v \in Y_j$ that satisfies the inequality constraints of Y_j strictly. Since $0 \in Y_j \cap (-K, K)^{\ell}$ (Assumption I.a.) and by convexity of the inequality constraints of Y_j, there must exist $v' \in Y_j \cap (-K, K)^{\ell}$ that satisfies the inequality constraints of Y_j strictly as well. As a result, the OPT-gate functions properly and $y_j = \overline{y}_j$ is an optimal solution of (3.14). In particular, $y_j \in Y_j$.

By Assumption IV.a., there exists $v \in X_i$ with $v < \zeta_i$ (componentwise). Since $\xi_i \leq v$ (Assumption II.), it also follows that $v \in X_i \cap (-K, K)^{\ell}$, by construction of K. Since y_j is an optimal solution of (3.14), we have $p \cdot y_j \geq 0$, because $0 \in Y_j \cap [-K, K]^{\ell}$ is a feasible point of (3.14). Thus, $\sum_{j=1}^m \alpha_{ij} p \cdot y_j \geq 0$, since $\alpha_{ij} \geq 0$. Now, since $v < \zeta_i$ and since there exists $h \in [\ell]$ with $p_h > 0$ (by optimality of p for (3.16)), it follows that $p \cdot v < p \cdot \zeta_i + \sum_{j=1}^m \alpha_{ij} p \cdot y_j$, i.e., $v \in X_i \cap (-K, K)^{\ell}$ strictly satisfies the budget constraint in (3.15).

Recall that the constraints describing X_i are assumed to satisfy the explicit Slater condition, i.e., there exists $v' \in X_i$ that satisfies the inequality constraints of X_i strictly. By the convexity of the inequality constraints of X_i, it follows that there exists v'' on the segment $[v, v']$ such that $v'' \in X_i \cap (-K, K)^{\ell}$, v'' strictly satisfies the budget constraint in (3.15), and v'' strictly satisfies the inequality constraints of X_i. This means that the constraints of (3.15) satisfy the explicit Slater condition and thus the OPT gate for (3.15) works correctly. As a result, $x_i = \overline{x}_i$ is an optimal solution of (3.15). In particular, $x_i \in X_i$ and x_i satisfies the budget constraint for consumer i.

Market equilibrium. By summing up the budget constraints satisfied by each x_i, and using the fact that $\sum_{i=1}^m \alpha_{ij} = 1$ (Assumption IV.b.) we immediately obtain that $p \cdot z \le 0$. Now, since p is an optimal solution of (3.16), it must be that $z \le 0$. But then, by construction of K, it follows that $x_i, y_j \in (-K,K)^\ell$ for all i and j. As a result, y_j is also an optimal solution of (3.14) without the constraint $v \in [-K,K]^\ell$. Similarly, by *convexity of preferences* (Assumption III.c.), which in particular holds for concave u_i, x_i is also an optimal solution of (3.15) without the constraint $v \in [-K,K]^\ell$. We have thus shown that Conditions 1 and 2 hold.

Recall that we have $z \le 0$ and $p \cdot z \le 0$. In order to prove that Condition 4 holds, it remains now to show that $p \cdot z = 0$. Clearly, if the budget constraint for each consumer i is tight, then indeed $p \cdot z = 0$. Assume that for some i, the budget constraint is not tight, i.e., $p \cdot x_i < p \cdot \zeta_i + \sum_{j=1}^m \alpha_{ij} p \cdot y_j$. By *non-satiation* (Assumption III.b.), there exists $x_i' \in X_i$ with $u_i(x_i') > u_i(x_i)$. By *convexity of preferences* (Assumption III.c.), there exists $x_i'' \in X_i \cap (-K,K)^\ell$ with $u_i(x_i'') > x_i$ and such that x_i'' also satisfies the budget constraint. This is a contradiction to the optimality of x_i for (3.15). Thus, Condition 4 also holds.

Finally, note that Condition 3 trivially holds, since p is a feasible point for (3.16). It follows that any fixed point $(x_1,\dots,x_m,y_1,\dots,y_n,p)$ of F is indeed a market equilibrium.

Remark 6. If we also assume *free disposability* (Assumption I.d.), then there exists a market equilbrium that also satisfies $z = 0$, i.e., supply equals demand for all commodities, even those with zero price. We briefly sketch how F can be modified to yield FIXP-membership for this problem too.

Instead of outputting $(\overline{y}_1,\dots,\overline{y}_n)$, F outputs $(\overline{y}_1',\dots,\overline{y}_n')$, which is set as the output of the OPT-gate for the convex feasibility problem:

$$\begin{aligned}
\text{maximize} \quad & 0 \\
\text{subject to} \quad & \sum_{j=1}^n v_j = \sum_{i=1}^m x_i - \sum_{i=1}^m \zeta_i \\
& p \cdot v_j = p \cdot \overline{y}_j, \forall j \\
& v_j \in Y_j, \forall j \\
& v_j \in [-K,K]^\ell, \forall j
\end{aligned}$$

Furthermore, in (3.15) and (3.16), we replace y_j by $\overline{y}_j$, including in z.

Using the same arguments as above, it follows that $p = \overline{p}$ is an optimal solution of (3.16), $\overline{y}_j$ is an optimal solution of (3.14), and $x_i = \overline{x}_i$ is an optimal solution of (3.15). As before, this implies that $z \le 0$ and $p \cdot z = 0$.

Recall that we have redefined $z := \sum_{i=1}^m x_i - \sum_{j=1}^n \overline{y}_j - \sum_{i=1}^m \zeta_i$. By *free disposability* (Assumption I.d.), and since $z \le 0$, it follows that $w := \sum_{j=1}^n \overline{y}_j + z \in Y$. This means that we can write $w = \sum_{j=1}^n w_j$ where $w_j \in Y_j$, and it holds that $z' := \sum_{i=1}^m x_i - \sum_{j=1}^n w_j -$

$\sum_{i=1}^m \zeta_i = 0$. In particular, $w_j \in (-K,K)^\ell$ by construction of K. Furthermore,

$$\sum_{j=1}^n p \cdot w_j = p \cdot w = \sum_{j=1}^n p \cdot \overline{y}_j + p \cdot z = \sum_{j=1}^n p \cdot \overline{y}_j$$

since $p \cdot z = 0$. This implies that $p \cdot w_j = p \cdot \overline{y}_j$ for all j, because $\overline{y}_j$ is an optimal solution of (3.14), and w_j is feasible for (3.14). As a result, $(w_1,\dots,w_n)$ is feasible for the feasibility problem above. In particular, the feasible region is nonempty, and thus the OPT-gate ensures that $(\overline{y}'_1,\dots,\overline{y}'_n)$ is indeed feasible for the feasibility problem.

Since x_i satisfies the budget constraint with $\overline{y}_j$, it also satisfies it with w_j instead. It follows that $(x_1,\dots,x_m,y_1,\dots,y_n,p)$ is a market equilibrium where supply equals demand for all commodities, even those with zero price.

3.6.2 The Hylland-Zeckhauser pseudomarket mechanism

In 1979, Hylland and Zeckhauser [142] introduced the *random assignment problem* (also known as the randomized *house allocation problem*, e.g., see [1, 32]). In this problem, the goal is to assign a set of indivisible goods $G = \{1,\dots,n\}$ to a set of agents $A = \{1,\dots,n\}$, who have preferences over the goods. These preferences are expressed via cardinal utilities values, i.e., every agent i has a utility vector u_i, where u_{ij} denotes the utility of agent i for good j.

Together with the introduction of the problem, Hylland and Zeckhauser [142] also proposed a mechanism for coming up with an allocation that satisfies several desirable properties, namely *ex-ante envy-freeness* and *ex-ante Pareto efficiency*. The mechanism works by assigning every good a price $p_j \geq 0$ and every agent i, endowed with one unit of artificial currency, buys *probability shares* $x_i = (x_{i1},\dots,x_{in})$ of the goods in a manner that maximizes the agent's expected utility $\sum_{j=1}^n u_{ij}x_{ij}$ subject to a budget constraint $\sum_{j=1}^n p_j x_{ij} \leq 1$ and a feasibility allocation constraint $\sum_{j=1}^n x_{ij} = 1$. We require that every good is entirely assigned to the agents, that is $\sum_{i=1}^n x_{ij} = 1$ for every j, and if this is not the case, the mechanism reacts by adjusting the prices. A set of prices and an allocation for which all the items are entirely allocated (and the artificial budgets are exhausted) is an equilibrium point of the mechanism, which we refer to as a *HZ-equilibrium*.

Definition 3.6.2 (HZ-equilibrium)**.** A pair (p,x) of a price vector p and an allocation matrix x form a *HZ-equilibrium* for the pseudomarket described above if:

i. For every good $j \in G$: $\sum_{i=1}^n x_{ij} = 1$.

ii. For every agent i, the allocation x_i maximizes $\sum_{j=1}^n u_{ij}x_{ij}$ subject to the constraints $\sum_{j=1}^n p_j x_{ij} \leq 1$ and $\sum_{j=1}^n x_{ij} = 1$. Also, if there are multiple allocations for i that satisfy this, then x_i must be the cheapest such allocation.

From the definition above, it is clear why the Hylland-Zeckhauser mechanism is often referred to as a *pseudomarket*. The HZ-equilibrium clearly resembles the

Competitive Equilibrium from Equal Incomes [104, 219], except for the allocation constraints which restrict a buyer's allocation to be a total of one unit. This constraint turns out to be rather crucial, as computing a CEEI can be done in polynomial time, whereas the complexity of computing exact HZ-equilibria is still an open problem.

The reason for requiring that the x_i are the cheapest utility-maximizing allocations is that this ensures that a HZ-equilibrium (p,x) is ex-ante Pareto optimal. We note that $p_j \le n$ for all j, because of the budget constraints. Using Kakutani's fixed point theorem, Hylland and Zeckhauser proved the following result.

Theorem 3.6.4 (Hylland and Zeckhauser [142])**.** *Every instance of the pseudomarket above admits a HZ-equilibrium (p,x). Also, p may be chosen such that $p_j = 0$ for some j.*

Membership in FIXP – the proof of Theorem 3.6.2

Vazirani and Yannakakis [221] recently provided a proof of membership in FIXP for the problem of computing a HZ-equilibrium. Their proof defines a map of prices and allocations $F = (F_p, F_1, \dots, F_n)$. Each of the coordinate functions is defined by a straight-line-program consisting of various steps. For instance, in one of the steps of F_i the allocation x_i is altered in the case that agent i has not exhausted its budget and $x_{ij} > 0$ for some good j with $u_{ij} < \max_k u_{ik}$. Using a potential function argument, they argue that if (p,x) is a fixed point, then none of the steps of the coordinate functions of F will alter (p,x). With this in mind, they then argue that if (p,x) is not an equilibrium, then there must be some step of F that causes a change, implying that (p,x) cannot be a fixed point. Next, we present a different proof which makes use of our general technique and which we believe to be conceptually simpler.

The domain of our map F is $D = [0,n]^n \times ([0,1]^n)^n$. The basic idea of the map is that, on input (p,x), we first compute an allocation x'_i for each agent i that maximizes the expected utility (but not necessarily the cheapest among those) given prices p. Then, using x'_i, we compute an allocation y_i on $n+1$ goods, which achieves the same utility as x'_i, but also minimizes the cost. The extra "dummy" good is carefully constructed in such a way that it is never chosen (i.e., $y_{i,n+1} = 0$) at a fixed point of F. The function F outputs $(\overline{p}, \overline{x})$, where $\overline{x}_i$ is equal to $\pi(y_i)$, namely the projection of y_i onto the first n coordinates, and $\overline{p}$ is obtained as the solution to a simple LP which ensures that overallocated goods get maximum price, and underallocated goods get price 0. We note that the input x is not used in the computation of the circuit; the only reason why it is included in the domain of F is so that a fixed point (p,x) of F also includes an optimal allocation.

Description of the map. On input (p,x), $F : D \to D$ first computes an allocation x'_i as a solution to the Linear Program:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} u_{ij} z_{ij} \\ \text{subject to} \quad & \sum_{j=1}^{n} z_{ij} = 1 \\ & \sum_{j=1}^{n} p_j z_{ij} \leq 1 \\ & z_{ij} \geq 0, \forall j \end{aligned} \tag{3.17}$$

Before describing the Linear Program computing y_i, we introduce the dummy good that we mentioned earlier. For every agent i, we let δ_i denote the difference in utility between its most preferred and second most preferred good (if the agent prefers all goods equally, we let $\delta_i = 1$). The utility of agent i for the dummy good is now defined as $u_{i,n+1} = u_{i\max} + \delta_i$. Furthermore, we let $p_{n+1} = 3n$. Now, y_i is computed as a solution to the Linear Program:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n+1} p_j z_{ij} \\ \text{subject to} \quad & \sum_{j=1}^{n+1} z_{ij} = 1 \\ & \sum_{j=1}^{n+1} u_{ij} z_{ij} \geq \sum_{j=1}^{n} u_{ij} x'_{ij} \\ & z_{ij} \geq 0, \forall j \end{aligned} \tag{3.18}$$

The second output of F is obtained by setting $\overline{x}_i := \pi(y_i)$, where $\pi(\cdot)$ denotes projection onto the first n coordinates. Finally, we compute a solution p^* to the following Linear Program:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} q_j \Big(\sum_{i=1}^{n} y_{ij} - 1\Big) \\ \text{subject to} \quad & 0 \leq q_j \leq n, \forall j \end{aligned} \tag{3.19}$$

and we put $\bar{p} = p^* - (\min_j p^*_j) \cdot e$ where $e = (1, \ldots, 1)$.

Fixed points. Suppose that (p,x) is a fixed point of F, i.e., $(p,x) = (\bar{p}, \bar{x})$. Since by construction $\bar{p} = p^* - (\min_j p^*_j) \cdot e$, there exists some j with $p_j = \bar{p}_j = 0$. As a result, there exists a feasible solution to LP (3.17) that satisfies the inequality constraints strictly: let $z_{ik} = \varepsilon$ for all $k \neq j$ and $z_{ij} = 1 - (n-1)\varepsilon$ for a sufficiently small $\varepsilon > 0$.

Also, the equality constraints are clearly linearly independent. This implies that the explicit Slater condition is satisfied for LP (3.17), and therefore x'_i is indeed an optimal solution to this LP. Similarly, using that $u_{i,n+1} > u_{ij}$ for all $j \leq n$ we may show that there exists a feasible solution to LP (3.18) that satisfies all the inequalities strictly. Again, this means that the explicit Slater condition is satisfied for this LP and thus y_i is an optimal solution.

Lemma 8. *For every i we have $y_{i,n+1} = 0$.*

Proof. Suppose towards a contradiction that $y_{i,n+1} > 0$. Because y_i is a cost-minimizing allocation and $p_{n+1} > p_j$ for all $j \leq n$, there exists some j with $y_{ij} > 0$ and $u_{ij} < u_{i\max}$, where $u_{i\max} = \max_j u_{ij}$. Otherwise, one could achieve a better allocation that consists of only one of the goods that has maximal utility for agent i.

Let k denote a good of maximal utility for agent i, and pick $0 < \varepsilon < y_{ij}, y_{i,n+1}$. Now, define a new allocation y'_i by $y'_{i\ell} = y_{i\ell}$ for $\ell \neq j, k, n+1$, $y'_{ik} = y_{ik} + 2\varepsilon$, $y'_{ij} = y_{ij} - \varepsilon$, and $y'_{i,n+1} = y_{i,n+1} - \varepsilon$. We claim that y'_i is a strictly better allocation. Clearly, y'_i is still a stochastic vector. Also, using the definition of δ_i, we have that $u_{ij} \leq u_{i\max} - \delta_i$, so we may bound the change in utility as

$$\Delta u = 2\varepsilon u_{i\max} - \varepsilon u_{ij} - \varepsilon u_{i,n+1} = \varepsilon(2u_{i\max} - u_{ij} - (u_{i\max} + \delta_i)) \geq 0$$

We conclude that y'_i also satisfies the utility constraint. Using that $p_j, p_k \in [0, n]$ and $p_{n+1} = 3n$, we may bound the change in price of the allocation as

$$\Delta P = 2\varepsilon p_k - \varepsilon p_j - \varepsilon p_{n+1} \leq 2\varepsilon n - 3\varepsilon n < 0$$

We conclude that y'_i is a strictly better solution, contradicting the optimality of y_i. Therefore we have that $y_{i,n+1} = 0$. □

By the lemma above, we obtain that $\pi(y_i)$ is a cheapest utility-maximizing allocation for agent i at prices p. In particular, it satisfies the budget constraint. What remains to show is that every good $j \leq n$ is allocated fully. If that was not the case, we would have that there is some j with $\sum_{i=1}^n y_{ij} < 1$. This means that a solution p^* to LP (3.19) must have $p^*_j = 0$. There must also exist some k with $\sum_{i=1}^n y_{ik} > 1$ which implies that $p^*_k = n$. As $p^*_j = 0$, we obtain that $p_k = \overline{p}_k = n$. However, $p_k = n$ together with $\sum_{i=1}^n y_{ik} > 1$ implies that $\sum_{i=1}^n p_k y_{ik} > n$ which contradicts the budget constraints of the agents. Therefore $\sum_{i=1}^n y_{ij} = 1$ for every good j. We conclude that $(p, x) = (\overline{p}, \pi(y))$ forms a HZ-equilibrium, therefore proving Theorem 3.6.2.

3.7 Conclusion and Future Work

In this paper, we introduced the OPT-gate, a powerful tool that can be used as a black box for FIXP-membership proofs, essentially substituting the various Linear Programs and more general convex optimization programs that often appear in the corresponding existence proofs. We demonstrated the strength of our technique via a set of different applications on quite important and fundamental equilibrium computation problems in game theory, fair division and competitive markets.

We believe that our technique can be used even more broadly in the future, to enable clean and simple FIXP-membership proofs of other interesting problems. For example, one could study the equilibria of the various generalizations of the Hylland and Zeckhauser mechanism, e.g., due to He et al. [140] and Echenique et al. [81]; we believe that our OPT-gate can be used to show membership results for those problems as well, but the details need to be worked out. Independently of our technique but related to our results, some important open problems are whether one can show a FIXP-hardness result for the equilibrium computation problem in Hylland and Zeckhauser pseudomarkets, or in Arrow-Debreu markets. For Hylland and Zeckhauser pseudomarkets, it was recently proved that computing a (weak) approximate equilibrium is PPAD-complete [49], but the complexity of computing exact equilibria remains open. For Arrow-Debreu markets, as we explained in Section 3.1.2, the result of Garg et al. [118] does not quite yield the FIXP-completeness result for the market model as presented in [8]. On the other hand, our FIXP-membership result enables future work to consider rather general markets (with general concave utilities and convex consumption and production sets), in the quest of establishing the desired FIXP-completeness of the problem.

A.1 Piecewise Differentiable Concave Functions

A *piecewise differentiable concave* function is a function that is obtained by taking the lower envelope of differentiable concave functions. Formally, a piecewise differentiable concave function $f\colon \mathbb{R}^n \to \mathbb{R}$ is given by

$$f(x) = \min_{j\in[m]} g_j(x)$$

where for each $j \in [m]$, $g_j\colon \mathbb{R}^n \to \mathbb{R}$ is differentiable and concave. Note that, as a result, f is also concave and, in particular, admits a superdifferential ∂f.

For computational purposes f is represented as follows. For each $j \in [m]$, we are given:

- an algebraic circuit computing g_j, and
- an algebraic circuit computing ∇g_j, the gradient of g_j.

Clearly, given the circuits for g_j we can easily construct an algebraic circuit that computes f.

Lemma 9. *Given f represented as above, we can in polynomial time construct a pseudogate computing the superdifferential ∂f.*

Before proving this Lemma, we provide some notable special cases of piecewise differentiable concave function.

- **Piecewise Linear Concave (PLC):** This corresponds to the special case where the functions g_j are linear affine, i.e., $g_j(x) = a_j \cdot x + b_j$. PLC functions are usually represented by the rationals a_j, b_j for $j = 1, \dots, m$. Circuits for g_j and ∇g_j can easily be constructed from these. This class of functions also contains Leontief utilities.

- **Piecewise Polynomial Concave:** This corresponds to the special case where the functions g_j are concave polynomials. The polynomials are represented explicitly as a list of monomials. Again, circuits for g_j and ∇g_j can easily be constructed from these.

Proof of Lemma 9. We show how to construct a circuit $G_{\partial f}\colon \mathbb{R}^n \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$ such that $\mathsf{Fix}_\ell[G_{\partial f}](x) \subseteq \partial f(x)$ for all $x \in \mathbb{R}^n$.

On input $x \in \mathbb{R}^n$, $G_{\partial f}$ first computes $g_j(x)$ and $\nabla g_j(x)$ for $j = 1, \dots, m$, using the circuits for g_j and ∇g_j. Then using the OPT-gate, it computes $w \in \mathbb{R}^n$ as an optimal solution to the following LP:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{m} v_j g_j(x) \\ \text{subject to} \quad & \sum_{j=1}^{m} v_j = 1 \\ & v_j \geq 0, \forall j \end{aligned}$$

Finally, $G_{\partial f}$ outputs $\sum_{j_1}^{m} w_j \nabla g_j(x)$. Note that the ℓ auxiliary inputs/outputs of $G_{\partial f}$ are used to implement the OPT-gate.

Clearly, the LP above satisfies the explicit Slater condition (Definition 3.3.3), and as a result w is indeed an optimal solution of the LP. In other words, any $z \in \mathsf{Fix}_\ell[G_{\partial f}](x)$ can be written as $z = \sum_{j_1}^{m} w_j \nabla g_j(x)$, where w is an optimal solution to the LP above. By construction, optimality for the LP ensures that $z \in \mathrm{Conv}\{\nabla g_j(x)\colon g_j(x) = \min_k g_k(x)\} \subseteq \partial f(x)$. The last containment follows from standard properties of the superdifferential, see, e.g., [191]. $\square$

Chapter 4

PPAD-membership for Problems with Exact Rational Solutions: A General Approach via Convex Optimization

Abstract

We introduce a general technique for proving membership of search problems with *exact rational* solutions in PPAD, one of the most well-known classes containing total search problems with polynomial-time verifiable solutions. In particular, we construct a "pseudogate", coined the *linear-OPT-gate*, which can be used as a "plug-and-play" component in a linear arithmetic circuit, as an integral component of the "Linear-FIXP" equivalent definition of the class. The linear-OPT-gate can solve several convex optimization programs, including quadratic programs, which often appear organically in the simplest existence proofs for these problems. This effectively transforms existence proofs to PPAD-membership proofs, and consequently establishes the existence of solutions described by rational numbers.

Using the linear-OPT-gate, we are able to significantly simplify and generalize almost all known PPAD-membership proofs for finding exact solutions in the application domains of game theory, competitive markets, auto-bidding auctions, and fair division, as well as to obtain new PPAD-membership results for problems in these domains.

4.1 Introduction

Total search problems, i.e., search problems for which a solution is always guaranteed to exist, have been studied extensively over the better part of the last century, in the intersection of mathematics, economics and computer science. Famous examples of such problems are finding Nash equilibria in games [177], competitive equilibria in markets [8] and envy-free divisions of resources [209]. While the classic works in mathematics and economics have been primarily concerned with establishing the existence as well as desirable properties of these solutions, the literature of computer science over the past 35 years has been instrumental in formulating and answering questions about the computational complexity of finding them.

More precisely, Megiddo and Papadimitriou [170] defined the class TFNP to include all total search problems for which a solution is verifiable in polynomial time. To capture the computational complexity of many problems including the aforementioned ones, several subclasses of TFNP were subsequently defined. Among those, one that has been extremely successful in this regard is the class PPAD of Papadimitriou [182], which was proven to characterize the complexity of computing Nash equilibria in games [55, 67], as well as competitive equilibria for several types of markets [56], among many others.

In reality, when making statements like the above, i.e., general statements of the form, "finding a Nash equilibrium is in PPAD", or similarly for a solution to some other total search problem, it is most often meant that what lies in the class is the problem of finding *approximate* solutions. For strategic games for example, that would mean strategy profiles which are *almost* Nash equilibria, up to some additive parameter ε. This is actually quite often necessary, as it has been shown that for many of these problems, there are cases where all of their solutions can only be described by irrational numbers, and hence we can not hope to compute them exactly on a computer.

Still, there is a large number of important variants of these domains for which *exact rational* solutions exist. For example, several strategic games always have equilibria in rational numbers, and so do certain markets for their competitive equilibria. There are also examples from fair division where rational partitions of the resources can be achieved. In all of those cases, PPAD-membership results for their approximate versions are unsatisfactory; we would like to place the *exact* problems in PPAD instead.

Indeed, coming up with proofs of existence that also guarantee rationality of solutions has been a topic of interest in the area since the very early days, way before the introduction of the relevant computational complexity classes, e.g., see [80, 141, 160, 161]. Driven by those classic results, a significant literature in computer science has attempted, and quite often has succeeded in placing the corresponding computational problems in PPAD, for several of the application domains mentioned above, including games [137, 153, 155, 172, 203], markets [110, 112, 119, 220], as well as the more recent domain of auto-bidding auctions [57].

While these PPAD-membership proofs typically do follow one of a few common approaches, in essence they are rather domain-specific and require reconstructing a set of arguments again for each application at hand (see Section 4.1.2 below for a detailed discussion). Instead, we would like to have *one general technique for proving PPAD-membership of problems with exact solutions*, and ideally one that arises "organically" as the computational equivalent of the standard proofs of existence. To do this, a very promising avenue seems to be via a characterization of PPAD, coined *Linear-FIXP*, due to Etessami and Yannakakis [86], which defines the class in terms of fixed points of problems represented by piecewise-linear arithmetic circuits. This is because a standard existence proof, e.g., via the Kakutani fixed point theorem [149] or via Brouwer's fixed point theorem [39], often obtains the solution as a fixed point of a set of local optimization problems, in which each agent or player is independently maximizing a piecewise utility/payoff function. If we could "embed" these optimization problems into a piecewise-linear circuit, that would essentially translate the existence proof into a PPAD-membership proof. This is crisply captured in the following quote from Vazirani and Yannakakis [220], in the context of proving PPAD-membership for competitive equilibria in certain markets:

> *"There are very few ways for showing membership in PPAD. A promising approach for our case is to use the characterization of PPAD of Etessami and Yannakakis [2010] as the class of exact fixed-point computation problems for piecewise-linear, polynomial time computable Brouwer functions.* [...] *Unfortunately, we do not see how to do this* [...] *it is not clear how to transfer the piecewise-linearity of the utility functions to the Brouwer function." [220].*

Recently, Filos-Ratsikas et al. [101] in fact developed a general technique along those lines: they designed an *optimization gate*, which can be used as part of a circuit to substitute the aforementioned optimization problems and obtain membership results. Crucially however, their membership results are *not* for the class PPAD, but rather for the class FIXP [86], a superclass of Linear-FIXP in which the main computational device is a (general) arithmetic circuit, not a piecewise-linear one. These circuits are particularly powerful and can capture solutions with irrationalities. Using their "OPT-gate for FIXP", Filos-Ratsikas et al. [101] showed the FIXP-membership of several very general problems related to strategic games, markets and fair division.

While FIXP is certainly a natural class, it has not enjoyed the same success as PPAD, even in the context of classifying problems with exact solutions. Besides, in the standard (Turing) model of computation, a FIXP-membership result can be interpreted as finding a point that is *close* to a solution (e.g., in the max norm). This is often a stronger guarantee than an approximate solution as described earlier, but it it still very much only an approximation. Again, this is unsatisfactory for those problems with exact rational solutions that should be in PPAD.

Could we hope to use Filos-Ratsikas et al.'s optimization gate to obtain PPAD-membership? This is actually practically impossible, for reasons which are deeply rooted in the definitions of the classes; we highlight those in Section 4.1.3 below. In short, the power of general arithmetic circuits over piecewise-linear ones lies in their capability to multiply and divide input variables, and this is of vital importance in the design of the OPT-gate for FIXP in [101]. What we really need is *a new gate*, one which avoids such multiplications/divisions and hence can be used in a piecewise-linear arithmetic circuit. Clearly, the gate cannot capture the generality of applications that the OPT-gate for FIXP does, as, as we said earlier, problems with irrational solutions cannot be in PPAD. It should however be general enough to capture any problem for which exact rational solutions are possible.

This is the main technical contribution of our paper. We introduce the linear-OPT-gate,[1] which can be used as a general purpose tool for proving PPAD-membership of problems with exact rational solutions. We demonstrate its strength and generality on a host of different applications in game theory, markets, auctions and fair division. Via its use, we are able to *significantly simplify* or *generalize* virtually all of the PPAD-membership proofs for problems with exact solutions in the literature, as well as to prove *new* membership results for problems for which PPAD-membership was not known; we offer more details in the following subsection.

4.1.1 A Powerful Tool for PPAD-membership: The linear-OPT-gate

We introduce the linear-OPT-gate for proving membership of problems in PPAD. The linear-OPT-gate can be used as a "plug-and-play" component in a linear arithmetic circuit, i.e., similarly to any of the other gates $\{+, -, \max, \min, \times\zeta\}$ of the circuit (see Definition 4.2.3). The gate is guaranteed to work correctly at a fixed point of the function that the circuit encodes, which, for the purposes of proving PPAD-membership of a problem, is equivalent to a standard gate.

The linear-OPT-gate allows us to compute solutions to optimization programs of a certain form, like those shown in the left-hand side of Figure 4.1. In particular, these are optimization programs with a non-empty and bounded feasible domain given by a set of linear inequalities, and the subgradient of the convex objective function (in the variables x) is given by a linear arithmetic circuit. In particular, the linear-OPT-gate can compute the solution to any linear program, but also to more general convex programs, e.g., those with quadratic objective functions. The inherent strength of the technique lies in the fact that these types of optimization programs arise naturally in several of the applications in game theory, competitive markets and fair division. Now, for the purpose of showing membership in PPAD, they may effectively be substituted by linear-OPT-gates.

From the ability of the linear-OPT-gate to solve optimization programs of the form C of Figure 4.1, we can also derive *feasibility programs* with *conditional constraints*, like the program Q on the right-hand side of Figure 4.1. These feasibility programs also often arise naturally in the context of existence proofs, and can be also thought of as being solved in a black-box manner by a gate, which is constructed using the linear-OPT-gate.

Our linear-OPT-gate has a wealth of applications, which we discuss below.

4.1.2 Applications of the linear-OPT-gate

We apply our linear-OPT-gate to a plethora of different domains, and obtain PPAD-membership for finding solutions in several strategic games, competitive markets,

[1]The term "linear" here refers to piecewise-linear functions, and this is why it is typeset differently, see also Remark 7.

Optimization Program C	Feasibility Program Q
$$\begin{aligned} \min \quad & f(x;c) \\ \text{s.t.} \quad & Ax \le b \\ & x \in [-R,R]^n \end{aligned}$$	$$\begin{aligned} h_i(y) > 0 \implies a_i^\mathsf{T} x \le b_i \\ x \in [-R,R]^n \end{aligned}$$

Figure 4.1: The optimization programs and feasibility programs that can be solved by the linear-OPT-gate.

auto-bidding auctions, as well as problems in fair division. We detail those applications in the corresponding sections below. Our results achieve the following three desired objectives simultaneously:

- Proofs of existence of solutions.
- Proofs of rationality of solutions.
- PPAD-membership of the corresponding problems.

For some of these domains, PPAD-membership results for the corresponding problems were known; still, the proofs to establish those were often rather involved. With the employment of our linear-OPT-gate, they become *conceptually and technically significantly simpler*. In essence, the linear-OPT-gate allows us to turn a simple existence proof into a PPAD-membership result. For some of our applications such simple existence proofs already existed, and are transformed to PPAD-membership proofs via the linear-OPT-gate. For others, developing these simpler existence proofs is also part of our contribution; we provide more details in the sections below. The linear-OPT-gate also allows us to *straightforwardly* obtain generalizations of some of the known PPAD-membership results, to cases beyond what was known in the literature. Finally, we also obtain the PPAD-membership of some problems whose complexity had not been studied in the literature before.

We summarize our results in Table 4.1, where we indicate which results were known in the literature before, which are generalizations, and which concern problems for which we did not know any results about their computational complexity.

Before we proceed with the applications, we present the main techniques that have been used previously for proving PPAD-membership results, and highlight the main technical challenges of using those techniques as opposed to the "plug-and-play" nature of our linear-OPT-gate.

Main Previous Approaches

Linear Complementarity Programs and Lemke's Algorithm. The first main approach for establishing rationality of solutions and PPAD-membership is that of *linear complementarity programs (LCPs)* [63, 64]. Given an $n \times n$ matrix $\mathbf{M}$ and a vector $\mathbf{q}$, an LCP seeks to find two vectors $\mathbf{y}$ and $\mathbf{v}$ satisfying:

$$\mathbf{M} \cdot \mathbf{y} + \mathbf{v} = \mathbf{q}, \;\; \mathbf{y} \geq 0, \;\; \mathbf{v} \geq 0, \quad \text{and} \quad \mathbf{y}^{\mathsf{T}} \cdot \mathbf{v} = 0$$

The term "complementarity" stems from the fact that in a solution, we may have either $\mathbf{y}_i > 0$ or $\mathbf{v}_i > 0$, but not both. Lemke [160] designed an algorithm (based on the previously designed Lemke-Howson algorithm [161]) to solve LCPs via a series of *complementary pivoting* steps, i.e., steps in which when a variable enters the basis, a complementary variable exits. Interestingly, the algorithm was designed in the context of computing Nash equilibria in bimatrix games, long before the associated computational complexity classes were defined. LCP-based formulations of equilibria and other fixed point problems have in fact been a subject of study in classic works (e.g, see [80, 141]) as a means to obtain existence proofs that guarantee the rationality of solutions. PPAD membership can be obtained by pairing the algorithm with an appropriate local orientation of its complementarity paths [216].

Quite importantly, Lemke's algorithm terminates with either finding a solution to the LCP, or without finding a solution, in what is referred to as a *secondary ray*. This feature of the algorithm is well-documented (e.g., see [195] for an excellent exposition) and is known as *ray termination*. In terms of proving PPAD-membership, it seems almost inevitable that every PPAD-membership proof that uses this approach has to argue against ray termination. As Garg and Vazirani [112] pointedly remark, in the context of a succession of papers on equilibrium computation in competitive markets:

> *"In the progression of these three works, the LCPs have become more involved and proving the lack of secondary rays has become increasingly harder." [112].*

This is not particular to markets either. For example, in Hansen and Lund's [2018] generalization of the results of Sørensen [203] from bimatrix to polymatrix games, those concerning ε-proper equilibria, a new LCP needs to be devised, together with a new argument for ray termination. Additionally, there are often significant challenges in even appropriately formulating the problems in question as LCPs. In some cases, the naive formulations may lead to inefficient representations, e.g., see [203]. In other cases, all known formulations lead to *nonstandard* LCPs, which cannot be handled by the "vanilla" version of Lemke's algorithm, and require variants of the algorithm to be devised, e.g., see [119, 172]. Finally in some cases, it is not known if the derived LCPs can be solved via any variant of Lemke's algorithm, thus leading to the development of entirely new pivoting algorithms [155]. These characteristics of the LCP approach make it somewhat insufficient as a general purpose PPAD-membership technique.

One advantage of LCP-based approaches is that they have been shown to perform well in practice, e.g., see [119] and references therein. However, for the purpose of proving PPAD-membership, we do not see any general advantage of the LCP method over our linear-OPT-gate.

Approximation and Rounding. The second general technique that has been used in several applications to prove the PPAD-membership of exact solutions is that of *approximation and rounding*. This generally consists of the two following main steps:

- consider an *approximation* or a *relaxation* of the solution (e.g., ε-approximate equilibria) and prove that the approximate version is in PPAD, and
- devise a rounding procedure to transform approximate solutions to exact solutions, while maintaining membership in the class.

This rather indirect approach certainly suffers in terms of elegance. More importantly however, it is very much domain-specific. First, showing the PPAD-membership for the approximate version typically still requires a non-trivial proof, often even a rather involved one, e.g., via some reduction to one of the well-known problems in PPAD, like End-of-Line (see Definition 4.2.1) or the computational version of Sperner's lemma [204]. Also, the rounding procedure itself may be rather complicated, and of an ad hoc nature. For certain applications, there is a general linear programming-based technique developed by Etessami and Yannakakis [86] to transform ε-approximate solutions to exact ones, for sufficiently small values of ε. Still, this does not apply to all problems, and it may need to be used in conjunction with other tailor-made rounding steps, e.g., see [57, 220].

The linear-OPT-gate as a "plug-and-play" component. As we will explain in the following, and as it will be evident via inspection of our proofs throughout the paper, the linear-OPT-gate allows us to develop proofs which are very simple and streamlined, essentially mimicking the easiest proofs of existence. Clearly, most of the technical complications are "hidden" in the "inner workings" of the linear-OPT-gate. This is the advantage of having a "plug-and-play" component readily available for the proofs: one does not need to even be concerned about how the linear-OPT-gate works, but only to understand what kind of optimization programs it can solve. We consider this to be a significant advantage over the two aforementioned techniques, which require to devise application-specific arguments (be it arguments about ray termination or appropriate approximation and rounding). These arguments may be of a standard general nature, but they have to be devised anew for each application, as evidenced by all the different PPAD-membership results that employ these techniques.

Implicit Functions and Correspondences

As a final remark before we present our applications, we point out that, via machinery that we develop in Section 4.3.4, our linear-OPT-gate can be used to show the PPAD-membership of problems for which the inputs (e.g., utilities or latency functions) are given *implicitly* in the input. In particular, we show how we can construct linear arithmetic circuits computing these functions, when those are inputted *succinctly* via Boolean circuits. In terms of the applications, this allows us to effectively consider functions of exponential size (in the size of the circuits), e.g., piecewise-linear utility

Applications to Game Theory	
Games with Linear Best Response Oracles (LBRO)	**[Our Work]**
Linear Concave Games	**[Our Work]**
Bilinear Games	[157], *implicitly*
Extended Digraph Threshold Games	**[Our Work]**
Bimatrix Games	[182] [64], *implicitly*
Polymatrix Games	[141], *implicitly*
Linear Succinct Games	**[Our Work]**
Multi-class Congestion games with piecewise-linear latency functions	
Non-atomic Network Congestion Games	**[Our Work]** linear latencies [172]
Atomic Splittable Network Congestion Games	**[Our Work]** linear latencies [155]
Congestion Games with Malicious Players	**[Our Work]**
Other equilibrium notions	
ε-proper Equilibria in Bimatrix Games	[203]
ε-proper Equilibria in Polymatrix Games	[137]
ε-proper Equilibria in Linear Succinct Games	**[Our Work]**
Personalized Equilibria	[153]
Applications to Competitive Markets	
Exchange Markets with Linear Utilitites	[80], *implicitly*
Arrow-Debreu Markets with SPLC Utilities	[116]
Arrow-Debreu Markets with SPLC Utilities/Productions	[220] [112]
Arrow Debreu Markets with Leontief-free Utilities/Productions [119]	Arrow Debreu markets with *succint* SPLC utilities and and SPLC production **[Our Work]**
Applications to Auto-Bidding Auctions	
Pacing Equilibria in Second-Price Auctions with Budgets	[57]
Applications to Fair Division	
Envy-free Cake Cutting	[126], *implicitly*
Rental Harmony	**[Our Work]**

Table 4.1: A summary of our PPAD-membership results - for other complementary results please see the respective sections/paragraphs in the introduction. Classes of domains that are within the same frame in the table (i.e., not separated by borders) are of increasing generality from top to bottom. Domains that appear in the same row of a frame are incomparable in terms of their generality. For the applications to game theory, all of the domains are special cases of linear concave games which in turn are a special case of LBRO games. For those applications, the PPAD-membership extends to *generalized equilibria*. For all of the results in the table, regardless of whether we obtain entirely new results, generalizations, or simply results which were known in the literature, we obtain *significant simplifications* in the proofs.

functions with exponentially-many pieces. We provide details on how this capability of the linear-OPT-gate is used in each application in the corresponding sections below. We present applications for which the aforementioned techniques of Section 4.1.2 are *inherently insufficient* for obtaining PPAD-membership results for those implicit functions, when these results are in fact enabled by the use of the linear-OPT-gate.

PPAD-membership for Strategic Games

We start our discussion from the applications of the linear-OPT-gate to the problem of computing (exact) equilibria in strategic games. To provide some initial intuition, before the technical sections of the paper, we provide an informal example of the use of the linear-OPT-gate to compute mixed Nash equilibria in bimatrix games; this is exposed in more detail in Section 4.4.1.

An Example: Bimatrix Games

A bimatrix game is a game played between two players, in which the payoffs are given by two matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, one for each player, denoting the payoff of the players when they each choose certain actions. Each player chooses a *mixed strategy*, i.e., a probability distribution over actions in the game, aiming to maximize their expected payoff, against the choice of the opponent. A mixed Nash equilibrium is a pair of mixed strategies for which every player is *best responding*, i.e., she is maximizing her payoff, given the strategy of the other player. The existence of mixed Nash equilibria for bimatrix games follows from Nash's general existence theorem [177]. The proof of the theorem that employs the Kakutani fixed point theorem [149] constructs a fixed point of a function F from the domain of mixed strategies to itself, for which each coordinate F_i is a best response for player i in the game. These best responses can be captured by optimization programs of the form C in Figure 4.1 and in particular for the case of bimatrix games, these are linear programs in which the subgradients of the objective functions are linear functions. The existence proof then immediately yields a PPAD-membership proof if one substitutes those programs with linear-OPT-gates that compute them.

We remark that for bimatrix games, the original PPAD-membership proof of Papadimitriou [182] adopts the "LCP approach" that we mentioned earlier, i.e., it appeals to an alternative proof of Nash equilibrium existence due to Cottle and Dantzig [64] (see also [161]) that formulates the problem as an LCP. This is a good example of what we mentioned earlier; the linear-OPT-gate allows us to organically retrieve PPAD-membership from the standard, textbook existence proof of Nash [177].

Best Response Oracles, Linear Concave Games and Generalized Equilibria

Linear Best Response Oracles. The approach that we highlighted above is not restricted to bimatrix games, but it actually captures a large class of strategic games. In Section 4.4.3 we provide a technical definition for a very general class of games, in which the best response of each agent is given by an oracle that can be computed by a linear arithmetic circuit. We refer to these games as *games with linear best response oracles (LBRO games)*. An equilibrium of any LBRO game can straightforwardly be formulated as a fixed point of a function like the function F above, where each coordinate F_i computes the best response of player i via the oracle. By using linear-OPT-gates as oracles, we immediately obtain PPAD-membership results for a wealth of different games.

Linear Concave Games. The class of *concave games* is a very large class of games, studied notably by Rosen [192] and Debreu [68]. These are games with continuous strategy spaces, for which the existence of an equilibrium is guaranteed under certain continuity and concavity assumptions on the utility functions. This was proven by Rosen [192] but also earlier independently by Debreu [68], [91], and Glicksberg [124], and for that reason the existence result is often referred to as the Debreu-Fan-Glicksberg theorem for continuous games.

In Section 4.4.3 we prove that as long as the supergradient of the (concave) utility function can be computed by a linear arithmetic circuit, concave games are LBRO games, and hence finding an equilibrium is in PPAD. We refer to those games as linear concave games, but emphasize again that the function does not have to be linear; in particular, it could for example be a quadratic function. Bimatrix games are linear concave games, and so are *polymatrix games* [141, 145], *bilinear games* [115], as well as generalizations of *(digraph) threshold games* [181], and thus we obtain membership of finding equilibria in all of these games in PPAD. The latter two games have continuous strategy spaces, and thus the equilibria that we compute are pure, whereas for polymatrix games (and as a result, for bimatrix games) we compute equilibria in mixed strategies.

Linear succinct games. In fact, we define a large class of games, which generalize polymatrix games, one which we coin *linear succinct games*. In these games, the expected utility of a player, given a pure strategy j and a mixed strategy $\mathbf{x}_{-i}$ of the other players, can be computed by a linear arithmetic circuit. These are linear concave games, and the PPAD-membership of finding their mixed Nash equilibria is a corollary of the results mentioned above.

We draw parallels between linear succinct games and those defined in Daskalakis et al. [66] and Papadimitriou and Roughgarden [183]. Those works define classes of succinct games for which there is an oracle for computing the expected utility of the player. In [183], this oracle is referred to as the *polynomial expectation property* and is used to show that correlated equilibria [9] of games with this property can be computed in polynomial time. In [66], it is shown that if the oracle is given by a *bounded division free straight-line program of polynomial length*, then these games are in PPAD. Crucially, this latter result concerns *approximate equilibria*. One could view our result as a complement to those two results, one which concerns *exact* equilibria in *rational* numbers.

Our PPAD-membership result for linear concave games captures the limits of the class of concave games for which rational equilibria exist, and thus membership in PPAD is possible. The only other known complexity results for the general class of concave games are a FIXP-completeness result due to Filos-Ratsikas et al. [101], and a very recent PPAD-membership result for *approximate* equilibria due to [184].

Generalized Equilibrium. Debreu [68] did not only consider concave games, but in fact a more general equilibrium notion, one in which the strategy space of each player is dependent on the set of strategies chosen by the other players. This was coined a "social equilibrium" by Debreu [68] (see also Dasgupta and Maskin [65]) but over the years has been better known by the term *generalized equilibrium*. For our purposes, the dependence on other strategies can be embedded in the constraints of the optimization programs that we use as oracles in LBRO games, in a way that can be handled by the linear-OPT-gate. As a corollary, we obtain all of the aforementioned PPAD-membership results for generalized equilibria (rather than standard equilibria) as well, see Section 4.4.3. To the best of our knowledge, these are the first PPAD-membership results for generalized equilibria in the literature.

Personalized Equilibria

The notion of *personalized equilibrium* was introduced by Kintali et al. [153] in the context of games played on hypergraphs, with an equivalent strategic form. Intuitively speaking, these equilibria allow players to "match" their strategies with those of their opponents, without obeying a product distribution. Kintali et al. [153] showed the PPAD-membership (and as a result, rationality of equilibria) of personalized equilibria via the "relaxation and rounding approach" (see Section 4.1.2). In particular, they first define an approximate version of the problem (the ε-personalized equilibrium), and reduce that problem to End-Of-Line (see Definition 4.2.1 in Section 4.2), via a relatively involved construction. To obtain PPAD-membership for the exact problem (i.e., when $\varepsilon = 0$) Kintali et al. [153] construct an elaborate argument that appeals to linear programming compactness, by first showing that for sufficiently small ε, ε-personalized equilibria "almost satisfy" the constraints of the linear programs, and then carefully rounding the solution to obtain an exact equilibrium.

The use of the linear-OPT-gate allows us to obtain the PPAD-membership of the problem via an extremely simple argument. Essentially, each player computes their best response via a linear program which is computed by the linear-OPT-gate, which reduces the problem to finding an equilibrium of an LBRO game, see Section 4.4.3.

ε-proper Equilibria

We also consider an alternative equilibrium notion, that of *ε-proper equilibria*. This notion was introduced by Myerson [174] to refine the notion of ε-perfect equilibrium of Selten [199], and captures situations in which the players can make small mistakes ("trembles") in the choice of their mixed strategies. The PPAD-membership of computing ε-proper equilibria was known for bimatrix games due to Sørensen [203] and for polymatrix games due to Hansen and Lund [137]. Both of these works adopt the LCP approach, which means that they need to go through the hassle of establishing the properties of Lemke's algorithm, as discussed in Section 4.1.2 above. Additionally, formulating the problem as an LCP in this case is far from trivial, and requires an extended formulation of the generalized permutahedron due to Goemans [125], to make sure that the LCP has polynomially-many constraints.

The use of our linear-OPT-gate distinctly avoids all this labor. We formulate the problem of computing a best response for each player (where the best response is defined with respect to the ε-proper equilibrium notion) as a feasibility program of the form Q in Figure 4.1, which can be solved by the linear-OPT-gate. This essentially renders the game a LBRO game, and the PPAD-membership follows simply as a corollary of our main theorem for LBRO games.

Network Congestion Games

Our last application in the area of game theory is to multi-class congestion games. In particular, we will consider two models, *non-atomic congestion games* and *atomic splittable congestion games*. In the former case, there is a continuum of players who collectively form a class controlling a certain load allocation to different resources. In the latter case, each class is represented by a single (atomic) player, who controls the load and distributes it to the resources. For both of those settings, we will also consider the subclass of *network congestion games*, where the strategies can be represented more succinctly using flows over a directed network.

The existence of equilibria in those games was established in classic works, e.g., see [198] or [173], originally via the employment of the Debreu-Fan-Glicksberg theorem [1952] for continuous games, assuming that the latencies on the resources are concave functions. Relevant to us are the works on their computational complexity, namely [172] (for non-atomic network congestion games) and [155] (for atomic splittable network congestion games). Both papers showed the PPAD-membership of finding pure equilibria in their respective settings, when the latency functions are *linear*. We remark that these games are different from atomic (non-splittable) congestion games, for which finding pure Nash equilibria is known to be in the class PLS defined by Johnson et al. [148].

Meunier and Pradeau obtain their PPAD-membership result via the "LCP approach" mentioned in Section 4.1.2. Interestingly, their LCP formulation turns out to not be amenable to the use of Lemke's algorithm, so they have to devise a "Lemke-like" complementary pivoting algorithm, tailored to their problem. As in the case of Lemke's algorithm, they argue explicitly against ray termination. Klimm and Warode note that in their case, the problem of finding an equilibrium can be formulated as an LCP, however, it is not known or clear whether this LCP can be solved using any known variant of Lemke's algorithm. For that, they devise a rather involved proof, based on a new homotopy method, essentially a new pivoting algorithm. Their algorithm solves the problem of finding a Nash equilibrium as a system of linear equations involving notions such as *excess flows*, *vertex potentials* and *block Laplacians*. At a very high level, the authors use the excess and potentials to define an undirected version of the End-of-Line graph (see Definition 4.2.1 in Section 4.2), and the determinant of the block Laplacians to define a unique orientiation of the edges, effectively reducing the problem to End-of-Line.

The linear-OPT-gate allows us to avoid all of the technical complications of the proofs of Meunier and Pradeau [172] and [155] (which are rather involved, especially the latter), and essentially obtain the PPAD-membership for both of these problems as simple corollaries of our main results for LBRO games or concave games. In fact, we obtain *generalizations* of those PPAD-membership results to games with more general latency functions, notably piecewise-linear latency functions (implicitly or explicitly represented). In exactly the same fashion, we can use the linear-OPT-gate to obtain the PPAD-membership of *congestion games with malicious players*, a setting studied by Babaioff et al. [13], for which computational complexity results had not been previously proven.

All of our results on congestion games are presented in Section 4.5.

PPAD-membership for Competitive Markets

We now move on to the application of our technique to the domain of competitive markets. The standard market model in the literature is that of the Arrow-Debreu market [8], where a set of consumers compete for goods endowed by them and other consumers and goods produced by a set of firms. A *competitive equilibrium* of the market is a set of allocations of goods to the consumers, a set of production quantities and a set of prices, such that at those prices, (a) all consumers maximize their individual utilities, (b) all firms produce optimal amounts, and (c) the market clears, i.e., supply equals demand. The existence of an equilibrium for the general market model was established by Arrow and Debreu [8] via the employment of Debreu's social equilibrium theorem [1952], under some standard assumptions on the utilities of the consumers and the production sets of the firms.

Previous results and proofs. It has been well-known since the early works in the area [80] that in general Arrow-Debreu markets, competitive equilibria may be irrational. A significant literature, starting with the work of Eaves [80] aimed at identifying special cases of the Arrow-Debreu market for which *exact rational* solutions are always possible. When computer science took over in this quest, the related question of establishing the PPAD-membership of finding those exact solutions was also brought forward. Most of the PPAD-membership proofs that were developed through the years followed the "LCP approach", see Section 4.1.2. We present them here in succession:

- Eaves [80] considered the simplest case of exchange markets (no production) with linear utilities for the consumers and devised an LCP that can be solved by Lemke's algorithm. To establish the latter fact, he argued against ray termination, a characterstic of this approach that we emphasized in Section 4.1.2. A PPAD-membership proof is implicit in his result.[2]

- Garg et al. [116] considered exchange markets with *separable piecewise-linear concave (SPLC)* utilities, a generalization of linear utilities in which every agent has a piecewise linear concave utility for the amount of a good j that she receives, and her total utility for her bundle is additive over goods. The authors proved the PPAD-membership of finding competitive equilibria in those markets via devising an LCP that was "quite complex" [116], and naturally had to argue against ray termination, to establish that Lemke's algorithm will terminate on this LCP with a valid solution.

- Garg and Vazirani [112] considered Arrow-Debreu markets with SPLC utilities as well as SPLC production functions. This is in fact the work from which the quote of Section 4.1.2 is taken. The quote highlights the increasing challenge of developing these LCPs and establishing their successful termination. Indeed, for this LCP, Garg and Vazirani [112] devise a set of linear programs, and then use the complementary slackness and their feasibility conditions to develop the LCP needed for production. The non-homogeneity of the resulting LCP for the equilibrium problem is dealt with in a manner which is different from previous works [80, 116] and, naturally, since the developed LCP is different, Garg and Vazirani again need to argue against ray termination.

- The most general class of utility/production functions for which a PPAD-membership of exact competitive equilibria was proven is that of *Leontief-free* functions [119], which generalize SPLC functions. For this, the authors devise yet another LCP formulation, which turns out to be even more complex than those of previous works. This is because it has to differentiate between "normal" and "abnormal" variables, the latter preventing the employment of Lemke's algorithm. To circumvent this, they exploit some additional structure of their *nonstandard* LCP, and then they also *modify* Lemke's algorithm, to account for the possibility of abnormal variables becoming zero. Finally, as they devise a new LCP, they also have to argue once again against ray termination.

Besides those works, the first work in computer science to prove PPAD-membership for markets with SPLC utilities/productions was [220]. The approach in that paper is not the "LCP approach" but the "approximation and rounding approach" (again, see Section 4.1.2). An issue with this method is that very small changes in the prices may result in drastic changes in the optimal bundles of the consumers, which makes the proof quite challenging. To deal with this, Vazirani and Yannakakis [220] devise a set of technical lemmas that allow them to "force" certain allocations over others.

[2]Note that for exchange markets with linear utilities and no production the problem is in fact known to be polynomial-time solvable [**?**].

Our results. Our results in this section are twofold.

- **Simplified proofs.** First, we employ the linear-OPT-gate to recover all of the aforementioned PPAD-membership results via proofs which are conceptually and technically quite simpler. In particular, we formulate the optimal consumption and the optimal production as linear programs similar to program C of Figure 4.1, which can be effectively substituted by linear-OPT-gates in a linear arithmetic circuit. We also apply a standard variable change which was first used by Eaves [80], and which we refer to as *Gale's substitution*, see Remark 22. For the prices, we develop a feasibility program, similar to program Q of Figure 4.1. In a fixed point of the circuit, the optimality of consumption and production follows almost immediately by design. The main technical challenge of the proofs lies in arguing the market clearing of the outputted prices, which however still requires a relatively short proof.

  To introduce the reader gently to our proof technique, we first apply it to the simple setting of exchange markets with linear utilities in Section 4.6.1, then to the setting of Arrow-Debreu markets with linear utilities and productions in Section 4.6.2, and finally to the general case of Arrow-Debreu markets with Leontief-free utilities and productions in Section 4.6.3.

- **PPAD-membership for *Succinct* SPLC (SSPLC) utilities.** In Section 4.6.4 we introduce a new class of utility functions, which we coin *succinct separable piecewise-linear (SSPLC) utilities*. These are SPLC utilities in which the different segments of the utility function need not be given explicitly in the input (as in the case of (explicit) SPLC utilities), but can be accessed implicitly via a boolean circuit. Effectively, this allows us to *succinctly* represent SPLC functions with exponentially many pieces, where the input size is the size of the given circuits. We remark that the "LCP-approach" developed in the aforementioned papers is inherently limited in providing PPAD-membership results for this class. Indeed, one could formulate the problem as a large LCP in exponentially-many variables, and that would establish the existence of rational solutions. However, this formulation will no longer be a polynomial time reduction (since now we do not have explicit input parameters u^i_{jk} for the utiliy of each piece) and hence it does not imply the PPAD-membership of the problem. In contrast, using our machinery from Section 4.3.4 we can make sure that our linear-OPT-gate can be used to obtain PPAD-membership for markets with SSPLC utilities as well. In our result we also add (explicit) SPLC production, which our technique clearly can handle. We provide a discussion on the challenges of extending our results to also capture SSPLC production functions at the end of Section 4.6.4. Note that the SSPLC functions and the Leontief-free functions are of incomparable generality (and hence they appear on the same line of Table 4.1). Whether we can prove PPAD-membership for a class of "succinct Leontief-free functions", which would generalize both settings, is an interesting technical question.

PPAD-membership for Auto-bidding Auctions

Our next application is on the domain of auto-bidding auctions, which has received a lot of attention recently, due to its applicability in real-world scenarios [15–18, 33, 57, 61, 62, 162]. In particular, in Section 4.7 we consider the settings studied by Conitzer et al. [61, 62], Chen et al. [55] and Li and Tang [162], in which buyers participate in several parallel single-item auctions, via scaling their valuations by a chosen parameter α, called the *pacing multiplier*. The buyers do that while facing constraints on their feasible expenditure, typically provided by budgets or return-on-investment (ROI) thresholds. The objective is to find a *pacing equilibrium*, i.e., pacing multipliers and allocations for the buyers that are consistent with the format of the auction run (e.g, first-price or second-price) and satisfy the expenditure constraints of all the buyers simultaneously. Pacing equilibria have a similar flavor to the competitive equilibria discussed earlier, but are sufficiently different, and thus require separate handling.

Our proof vs the previous approach. We prove that computing pacing equilibria in parallel second-price auctions with budgets is in PPAD. The problem was already known to be in PPAD (in fact, PPAD-complete) by the recent results of Chen et al. [57], building on the original existence result of Conitzer et al. [62]. Chen et al.'s proof rather heavily applies the "approximation and rounding" paradigm highlighted in Section 4.1.2. In particular, Chen et al. define a (δ, γ)-approximate variant of the pacing equilibrium, where $\delta, \gamma > 0$ are two approximation parameters. Intuitively, this equilibrium corresponds to an "almost equilibrium" (i.e., the expenditure constraints are "almost" satisfied) of an "almost second-price auction" (i.e., an auction in which the set of winners is those with "almost" the highest bid). The authors prove that finding these approximate equilibria is in PPAD, via a reduction to a computational version of Sperner's lemma [204], and then devise an intricate rounding procedure to convert (δ, γ)-equilibria into γ-equilibria. The final step in their proof applies the aforementioned technique of Etessami and Yannakakis [86] (see Section 4.1.2) to further round these equilibria to pacing equilibria (i.e., where $\gamma = 0$).

Our proof employs the linear-OPT-gate and is conceptually and technically much simpler, without needing to use approximations. Instead, we again apply the standard variable change in Gale's substitution (see Remark 22) which we also used for the case of competitive markets, to work with the expenditures rather than the allocations directly. From there, we can formulate the task of finding the optimal expenditures as a set of linear programs (one for each buyer), and the pacing multipliers will be obtained as a fixed point solution of a single simple equation. These linear programs can be solved by linear-OPT-gates which essentially establishes the PPAD-membership of the problem. The proof is detailed in Section 4.7.1.

ROI-constrained buyers. We observe that the existence proof underlying our PPAD-membership proof in this section can in fact almost straightforwardly be modified to yield the existence of pacing equilibria for a different setting in auto-bidding auctions, that of second-price auctions with average return-on-investment (ROI) constraints, studied by Li and Tang [162]. Li and Tang established the existence of pacing equilibria via a rather indirect proof, which first reduces the problem to a somewhat convoluted concave game and applies the Debreu-Fan-Glicksberg theorem [68] to obtain Nash equilibrium existence, and then recovers a pacing equilibrium as a limit point of such a Nash equilibrium. This proof in fact closely follows the original proof of Conitzer et al. [62] for the budgeted setting, and clearly does not have any implications on the computational complexity of the problem.

Our proof, besides its advantages in terms of simplicity, also allows us for the first time to obtain computational membership results for pacing equilibria in the ROI-constrained buyer case. It turns out that for this setting, all pacing equilibria may be irrational (see Example 7 in Section 4.7.2), and hence membership in PPAD is not possible. Instead, we employ the OPT-gate for FIXP developed by Filos-Ratsikas et al. [101] to easily transform our existence proof into a FIXP-membership proof.

PPAD-membership for Fair Division

The last applications of our linear-OPT-gate are related to the task of fairly partitioning a resource among a set of agents with different preferences over its parts. In particular, we show the PPAD-membership of computing *exact* envy-free solutions in two fundamental problems, namely *envy-free cake cutting* [108] and *rental harmony* [212], when the preferences of the agents ensure the existence of rational partitions.

Envy-free cake cutting. The envy-free division of a continuous resource (metaphorically, a "cake") is one of the most fundamental and well-studied mathematical problems of the last century. The origins of the theory of the problem can be traced back to the pioneering work of [208], with different variants being studied over the years in a large body of literature in mathematics, economics, and computer science; see [38, 188, 190] for some excellent textbooks on the topic. The existence of an envy-free division was established in 1980 independently by Stromquist [209], by Woodall [229], and by Simmons (credited in [212]), even when the division is required to be *contiguous*, i.e., when each agent receives a single, connected piece of the resource. These proofs proceed by first establishing the existence of divisions that are *approximately* envy-free and then obtaining exact solutions as limit points of these approximations.

It is known that in general, envy-free divisions might be irrational (e.g., see [24], or Example 8 for a simpler example), and hence the problem of computing them cannot be in PPAD. Filos-Ratsikas et al. [101] showed that envy-free cake cutting is in the class FIXP, which, recall, is appropriate for capturing the complexity of such problems. Still, there are interesting cases for which rational divisions always exist. This is the case for example when the agents' preferences are captured by piecewise constant density functions [126], a class of functions which is general enough to capture many problems of interest. A FIXP-membership result for these variants is unsatisfactory, and we would like to obtain a PPAD-membership result instead.

Without the convenience of using our linear-OPT-gate, one can establish such a membership result via the "approximation and rounding" technique, see Section 4.1.2. Deng et al. [76] showed that *approximately* envy-free cake cutting is in PPAD, by transforming Simmons' proof into a computational reduction. Goldberg et al. [126] showed how to "round" the approximate solution to obtain an exact envy-free division for preferences captured by piecewise-constant densities, as long as ε is sufficiently small.

Luckily, the linear-OPT-gate allows us to avoid having to do that, and instead directly obtain a PPAD-membership result without any need for approximations. In particular, we revisit the FIXP-membership proof of Filos-Ratsikas et al. [101]; similarly to our approach in this paper, they essentially first construct an existence proof for the problem, one which involves a pair of optimization programs, and then substitute those programs with their OPT-gates for FIXP. One might wonder if, by simply following the steps of the proof and substituting those programs with linear-OPT-gates instead, we can recover the PPAD-membership of the problem, for those classes of preferences for which it is possible. This is almost true, apart from the fact that there is a step in their proof that cannot be done in a linear arithmetic circuit.

Still, we manage to substitute that part by a third optimization program, which is in fact a rather simple linear program, and can effectively be substituted by a linear-OPT-gate. This allows us to obtain the PPAD-membership of the problem for the general class of valuation functions (i.e., functions expressing the preferences via numerical values) that can be computed by a linear arithmetic circuit, see Section 4.8.1, capturing the aforementioned case of valuations with piecewise-constant densities.

Rental harmony. The rental harmony problem, notably studied by Su [212], is concerned with the partition of rent among a set of tenants which have different preferences over combinations of rooms and rent partitions. In the generality studied by Su [212], this problem is in fact equivalent to that of finding an envy-free division of a *chore* among a set of agents. Su's existence proof is inspired by Simmons' proof for envy-free cake cutting, but employs a "dual Sperner labelling" [204]. Similarly to the proofs for cake-cutting, the proof also appeals to limits of approximate solutions. In contrast to cake-cutting however, computational complexity results about this general version of the problem were not known, not even for approximate partitions.

In Section 4.8.2, we prove that the problem of finding a solution to rental harmony is in PPAD, as long as the valuations of the tenants for the rent partition are given by linear arithmetic circuits. Interestingly, this is established via very much the same approach as the proof for envy-free cake cutting, thus providing for the first time a unified proof of existence for those two problems. If one goes beyond the aforementioned valuation functions, all rental harmony solutions may be irrational, as we show in Example 9. For those cases, we explain how the existence proof can be coupled with the OPT-gate for FIXP of Filos-Ratsikas et al. [101] to establish the FIXP-membership of the problem.

Computing Envy-free and Pareto-optimal allocations. We remark that very recently Caragiannis et al. [44] used our linear-OPT-gate to establish that computing probabilistic envy-free and Pareto-optimal allocations of multiple divisible goods is in PPAD.

4.1.3 The linear-OPT-gate vs the OPT-gate for FIXP

As we mentioned in the introduction, Filos-Ratsikas et al. [101] were the first to develop an OPT-gate for the computational class FIXP [86]. FIXP is the class that captures the complexity of computing a fixed point of an arithmetic circuit, i.e., a circuit over the basis $\{+,-,\max,\min,\div,*\}$ with rational constants, see Definition 4.2.2. FIXP is a larger class than Linear-FIXP, due to the fact that we can multiply and divide inside the circuit.

The tools that our linear-OPT-gate provides are conceptually very similar to those of the OPT-gate for FIXP of Filos-Ratsikas et al. [101], in that they can substitute convex optimization programs within existence proofs, when constructing a circuit whose fixed points are the solutions that we are looking for. However, the design of the gate itself is much more challenging.

The reason for this is the absence of the general multiplication gate $*$. While we can multiply any two circuit variables in a general arithmetic circuit, we can only multiply variables by constants in a linear arithmetic circuit. The construction of the OPT-gate for FIXP by Filos-Ratsikas et al. [101] makes extensive usage of the multiplication gate $*$ and can thus not directly be used for creating the linear-OPT-gate. In our case, the constraint matrix A is fixed (i.e., not an input to the linear-OPT-gate) and this does help to eliminate some of the general multiplication gates, but not all of them. At a high level, the construction of Filos-Ratsikas et al. [101] ensures that the output x of the gate satisfies

$$\mu_0 \cdot \partial f(x) + A^{\mathsf{T}}\mu = 0$$

where μ satisfies some standard KKT conditions. If x is feasible and if $\mu_0 > 0$, then it follows that x is an optimal solution by standard arguments (using the convexity of f). The term μ_0 is carefully constructed as a function of μ and x in order to ensure that x must be feasible and that $\mu_0 > 0$ when x is feasible. However, since both μ_0 and $\partial f(x)$ depend on x, in our case we cannot construct the term $\mu_0 \cdot \partial f(x)$, because that would entail multiplying two variables in the circuit. As a result, our construction instead ensures that the output x of the gate satisfies

$$\varepsilon \cdot \partial f(x) + A^{\mathsf{T}}\mu = 0$$

where μ again satisfies some standard KKT conditions, and where $\varepsilon > 0$ is some sufficiently small constant that is picked when constructing the gate. By standard arguments it still holds that if x is feasible, then it is an optimal solution. The challenge however is to ensure that x is indeed feasible. While the argument is relatively straightforward in the work of Filos-Ratsikas et al. [101], because μ_0 can depend on

x, here μ_0 has been replaced by a constant ε. Our main technical contribution in the construction of the linear-OPT-gate is to show that there exists a sufficiently small $\varepsilon > 0$ that forces x to be feasible, and that such ε can be constructed efficiently given the parameters of the gate (but, importantly, not its inputs!). As a bonus, our modified construction and analysis allows us to obtain a linear-OPT-gate that does not require any constraint qualification, whereas the construction of Filos-Ratsikas et al. [101] required an explicit Slater condition (which of course, as they show, is necessary in the case where the matrix A is not fixed).

From the standpoint of applications, the linear-OPT-gate can be used in almost the same direct manner as the OPT-gate for FIXP of Filos-Ratsikas et al. [101]. In some cases, precisely because we cannot multiply within a linear arithmetic circuit, we may have to apply some standard variable changes, to "linearize" certain constraints. Still, the linear-OPT-gate can effectively substitute appropriate optimization programs in the same way that the OPT-gate for FIXP can. In a nutshell, one can view the linear-OPT-gate as a more powerful tool for those applications for which rational exact solutions exist.

4.1.4 Organization of the Paper

In Section 4.2 we provide the main definitions and terminology needed for our paper. In Section 4.3, we detail the construction of the linear-OPT-gate, and prove its correctness. In the same section (Section 4.3.4), we also develop the necessary machinery to show how the linear-OPT-gate can be used to obtain PPAD-membership of problems where certain functions are given implicitly in the input to the problem. In Section 4.4, we provide the first applications of the linear-OPT-gate to several important classes of games and to different equilibrium notions, besides Nash equilibria. In Section 4.5 we explain how to apply the machinery that we develop in Section 4.4 in order to obtain PPAD-membership results for equilibrium computation in nonatomic and atomic splittable congestion games. In Section 4.6, we present the applications of our gate to finding competitive equilibria in Arrow-Debreu markets with different utility and production functions. In Section 4.7 we demonstrate the applicability of the linear-OPT-gate to obtain membership results for the auto-bidding auctions with pacing strategies. In Section 4.8 we obtain membership results for the two fundamental fair division problems of envy-free cake cutting and rental harmony. We offer some discussion and some directions for future work in Section 4.9.

We would like to emphasize that while our paper is very long, this is almost exclusively due to the fact that it covers so many applications, rather than due to the proofs that we develop for those applications, which in reality range from being very short to relatively short. For each of all of the domains that we consider, (a) we provide the appropriate definition and place the setting in context within the rest of the paper, (b) we discuss the related work and possibly the previous PPAD-membership results (if any), (c) we provide detailed comparisons with those previous proofs to demonstrate the effectiveness of our linear-OPT-gate as a general-purpose proof technique, and finally (d) we develop the proofs themselves. In some cases in fact, we first apply the technique to simpler settings for a gentle introduction, and then move on to study those settings in their full generality. We believe that all of our application sections are largely self-contained, and can be read almost in isolation, even after only reading the introduction of the paper, and by referring only to certain clearly referenced parts in other sections.

4.2 Preliminaries

In this section, we introduce the computational class PPAD, as well as the main machinery that will be used throughout the paper. The details for the specific settings that we will consider in our applications will be defined in the corresponding sections. We start with the definitions of the relevant computational complexity classes.

4.2.1 The class PPAD

All of the problems that we will consider in this paper will be *total search problems*. A total search problem is one in which a solution is always guaranteed to exist. For example, finding a Nash equilibrium in a game is a total search problem, by Nash's theorem [177]. Similarly, competitive equilibria in markets always exist (e.g., see [8]). The class TFNP [170] contains all total search problems *in NP*, i.e., those for which a candidate solution can be verified in polynomial time. For example, verifying whether a given set of strategies is a Nash equilibrium in a bimatrix game (see Definition 4.4.1) can be done in polynomial time, and hence the problem of finding Nash equilibria in bimatrix games is in TFNP. For a formal definition of the class TFNP, we refer the reader to [182].

The class PPAD, introduced by [182], is defined with respect to its canonical problem, called End-of-Line, see Definition 4.2.1 below. PPAD is the class of all problems in TFNP that are polynomial-time reducible to End-of-Line.

Definition 4.2.1 (End-of-Line)**.** The End-of-Line problem is defined as: given Boolean circuits P and S with n input bits and n output bits such that $P(0) = 0 \neq S(0)$, find x such that $P(S(x)) \neq x$ or $S(P(x)) \neq x \neq 0$.

Intuitively, PPAD captures the following problem. We are given a directed graph in which every node has indegree and outdegree at most 1 and a source of this graph, and we are asked to find another source or a sink. Such a node exists by the parity argument on the degrees of the nodes, which is the underlying principle of the class PPAD. Importantly, we are not given this graph explicitly in the input (otherwise the problem would be trivially in P), but we can access the predecessor and the successor of a given node via Boolean circuits; these are the circuits P and S of Definition 4.2.1 above. We will be using an alternative definition of the class, via linear arithmetic circuits, which we will define in Section 4.2.2 next.

4.2.2 The classes FIXP and Linear-FIXP

We start by defining arithmetic circuits and linear arithmetic circuits.[3]

Definition 4.2.2 (Arithmetic Circuit)**.** An *arithmetic circuit* is a circuit using gates in $\{+, -, *, \div, \max, \min\}$ as well as rational constants.

A linear arithmetic circuit is simply an arithmetic circuit were multiplication and division are not allowed.

Definition 4.2.3 (Linear Arithmetic Circuit)**.** A linear arithmetic circuit is a circuit using gates in $\{+, -, \max, \min, \times\zeta\}$ as well as rational constants, where $\times\zeta$ denotes multiplication by a rational constant.

We will use linear arithmetic circuits to provide an alternative definition of the class PPAD. First, we state and prove the next simple lemma, which will be useful in Section 4.3. For a rational number a, we let size(a) denote the number of bits needed to describe a in the standard representation, where a is written as an irreducible fraction, and the numerator and denominator are written in binary. We let size(f) denote the number of bits needed to describe a linear arithmetic circuit f (in particular, this includes the length of the description of any constants used in f).

Lemma 10. *For any linear arithmetic circuit $f : \mathbb{R}^n \to \mathbb{R}^m$ and any rational $B \geq 0$ it holds that*

$$\max_{x\in[-B,B]^n} \|f(x)\|_\infty \leq 2^{\mathrm{poly}(\mathrm{size}(B),\mathrm{size}(f))}.$$

Proof. Since a linear arithmetic circuit can be evaluated efficiently (see, e.g., [92, Lemma 3.3]), we have $\|f(0)\|_\infty \leq 2^{\mathrm{poly}(\mathrm{size}(f))}$. Additionally, f is L-Lipschitz-continuous over $\mathbb{R}^n$ with Lipschitz constant $L = 2^{\mathrm{poly}(\mathrm{size}(f))}$, see, e.g., [92, Lemma A.1]. As a result, for any $x \in [-B, B]^n$

$$\|f(x)\|_\infty \leq \|f(0)\|_\infty + L\|x\|_\infty \leq 2^{\mathrm{poly}(\mathrm{size}(B),\mathrm{size}(f))}. \qquad \square$$

We now move on to the definition of the related computational classes, in the context of arithmetic circuits. We mentioned the class FIXP in the introduction; we proceed to formally define it below. The following definitions follow those of [101].

A search problem Π with real-valued search space is defined by associating to any input instance I (encoded as a string over a finite alphabet Σ) a search space $D_I \subseteq \mathbb{R}^{d_I}$ and a set of solutions Sol(I). We assume there is a polynomial time algorithm that given I computes a description of D_I.

Next, we define *basic* FIXP *problems* and *basic linear*-FIXP *problems.*

[3]Sometimes in the literature, these are also referred to as *algebraic circuits*, e.g., see [101]. We also adopt the term "linear arithmetic circuit" for circuits over the basis $\{+, -, \max, \min, \times\zeta\}$ with rational constants, which was not used originally by Etessami and Yannakakis [86], but has been used in some more recent works [73, 92].

Definition 4.2.4 (Basic (linear)-FIXP problem)**.** A search problem Π is a basic (linear)-FIXP problem if every instance I describes a nonempty compact convex domain D_I described by a set of linear inequalities with rational coefficients and a continuous map $F_I \colon D_I \to D_I$ given by a (linear) arithmetic circuit C_I, and the solution set is $\mathrm{Sol}(I) = \{x \in D_I \mid F_I(x) = x\}$. We assume that C_I well-defined, i.e., it does not divide by zero and that it indeed represents a function F_I with $F_I(D_I) \subseteq D_I$.[4]

Next, we define reductions between search problems. Let Π and Γ be search problems with real-valued search space. A *many-one reduction* from Π to Γ is a pair of maps (f,g). The instance mapping f maps instances I of Π to instances $f(I)$ of Γ, and for any solution $y \in \mathrm{Sol}(f(I))$ the solution mapping g maps the pair (I,y) to a solution $g(I,y) \in \mathrm{Sol}(I)$ of Π. In order to avoid meaningless reductions, it is required that $\mathrm{Sol}(f(I)) \neq \emptyset$ if $\mathrm{Sol}(I) \neq \emptyset$. We require that the instance mapping f is computable in polynomial time. Etessami and Yannakakis [86] defined the notion of *SL*-reductions where the solution mapping g is *separable linear*. This means there exists a map $\pi \colon \{1,\dots,d_I\} \to \{1,\dots,d_{f(I)}\}$ and rational constants a_i, b_i, $i = 1,\dots,d_I$, such that for $y \in \mathrm{Sol}(f(I))$ one has that $x = g(I,y)$ is given by $x_i = a_i y_{\pi(i)} + b_i$ for all i. The map π and the constants a_i, b_i should be computable from I in polynomial time.

We now define the class FIXP.

Definition 4.2.5 (FIXP)**.** The class FIXP consists of all search problems with real-valued search space that SL-reduce to a basic FIXP problem for which the domain D_I is a convex polytope described by a set of linear inequalities with rational coefficients and the function F_I is defined by an arithmetic circuit C_I.

The class linear-FIXP is the "piecewise-linear fragment of FIXP" [86], defined below.

Definition 4.2.6 (Linear-FIXP)**.** The class linear-FIXP consists of all search problems with real-valued search space that SL-reduce to a basic linear-FIXP problem for which the domain D_I is a convex polytope described by a set of linear inequalities with rational coefficients and the function F_I is defined by a linear arithmetic circuit C_I.

Above we have formally defined the class Linear-FIXP as a class of search problems with real-valued search space forming a subclass of FIXP. However, Linear-FIXP also naturally defines a subclass of TFNP since for any instance I of a basic Linear-FIXP problem Π, $\mathrm{Sol}(I)$ always contains rational-valued solutions of polynomial bit-length. Following the convention in the literature, we will denote the class of search problems in TFNP reducible to the exact fixed-point computation defined by Π by Linear-FIXP as well.

[4]Given an arithmetic circuit, it is not clear how to check whether it does satisfy these properties, so we assume that it does, i.e., we consider promise problems, see also [101]. For the case of basic linear-FIXP problems, the first condition is trivially satisfied (since division is not allowed), but for the second condition we still require a promise. Note that this means that the problem is formally not a TFNP problem, but instead a promise-TFNP problem. This is however not an issue for proving PPAD-membership, since the problem ultimately reduces to the TFNP problem End-of-Line [86].

With this convention, Etessami and Yannakakis [86] showed the following equivalence result:

Theorem 4.2.1 ([86])**.** *Linear-FIXP = PPAD.*

Theorem 4.2.1 provides an alternative definition of PPAD which we will be using throughout the paper. Roughly speaking, to show that a problem is in PPAD, it suffices to show that it can be reduced to computing a fixed point of a function encoded by a linear arithmetic circuit.

Remark 7 (Linear vs Linear)**.** The term "linear" in "linear-FIXP" and "linear arithmetic circuits" might be a bit misleading, as it may suggest that they only refer to linear functions. From Definition 4.2.3, it should be obvious that they capture piecewise-linear functions instead, and are hence more general. We believe that the term "linear" was used in the related literature rather than "piecewise-linear" for succinctness and brevity. In this paper, we will still call these arithmetic circuits "linear", but we will use the typesetting "linear" instead, to differentiate. We will use this typesetting throughout, also in some of our definitions (e.g., "linear succinct games", see Definition 4.4.6 in Section 4.4.3). We hope that the literature will adopt the same convention in future work.

We conclude the section with a very useful definition, that of *linear pseudo-circuits*.

Definition 4.2.7 (Linear pseudo-circuit)**.** A linear pseudo-circuit with n inputs and m outputs is a linear arithmetic circuit $F : \mathbb{R}^n \times [0,1]^\ell \to \mathbb{R}^m \times [0,1]^\ell$. The output of the linear pseudo-circuit on input $\mathbf{x}$ is any $\mathbf{y}$ that satisfies $F(\mathbf{x},\mathbf{z}) = (\mathbf{y},\mathbf{z})$ for some $\mathbf{z} \in [0,1]^\ell$. Note that a linear pseudo-circuit can have multiple possible outputs.

Intuitively, linear pseudo-circuits are linear arithmetic circuits which are only required to work correctly at a fixed point of the encoded function (or, to be more precise, when its "auxiliary" variables $\mathbf{z}$ satisfy a fixed point condition). In particular, linear arithmetic circuits are linear pseudo-circuits, and in fact, for the purpose of proving membership in PPAD, those two are equivalent. The linear-OPT-gates that we will define in the next section are in fact linear pseudo-circuits that are used as *primitives* or *subroutines* in larger linear pseudo-circuits. The following is an example of a simple linear pseudo-circuit, that was already used as an important primitive by Filos-Ratsikas et al. [101].

Example 3 (Linear pseudo-circuit computing the Heaviside function)**.** The Heaviside function is the following correspondence

$$\mathrm{H}(x) = \begin{cases} 1 & \text{if } x > 0 \\ [0,1] & \text{if } x = 0 \\ 0 & \text{if } x < 0 \end{cases} .$$

We can construct a linear pseudo-circuit $F : \mathbb{R} \times [0,1] \to \mathbb{R} \times [0,1]$ computing H by letting

$$F(x,z) := (z, \min\{1, \max\{0, z + x\}\}).$$

It is easy to check that F indeed computes H, i.e., $F(x,z) = (y,z) \implies y \in \mathrm{H}(x)$.

Computing a fixed point of a linear pseudo-circuit corresponds to computing a fixed point of the linear arithmetic circuit representing it. Thus, a linear pseudo-circuit is guaranteed to have at least one rational fixed point, and the problem of computing such a fixed point lies in PPAD.

4.3 A Powerful Tool for PPAD-membership: The linear-OPT-gate

In this section, we develop our main tool, the linear-OPT-gate.

When constructing a linear arithmetic circuit *for the purpose of proving membership in* PPAD, we can assume without loss of generality that we have access to an additional gate, called the linear-OPT-gate, which:

- is parameterized[5] by $n, m, k \in \mathbb{N}$, a rational matrix $A \in \mathbb{R}^{m \times n}$, and a linear arithmetic circuit $G_{\partial f} : \mathbb{R}^n \times \mathbb{R}^k \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$.
- takes as input $b \in \mathbb{R}^m$, $c \in \mathbb{R}^k$, and $R \in \mathbb{R}$.

The linear-OPT-gate outputs an optimal solution of the following optimization problem (over variables $x \in \mathbb{R}^n$):

<u>Optimization Program C</u>

$$\begin{aligned} \min \quad & f(x;c) \\ \text{s.t.} \quad & Ax \leq b \\ & x \in [-R,R]^n \end{aligned} \tag{4.1}$$

whenever the two following conditions hold for the given inputs $b \in \mathbb{R}^m$, $c \in \mathbb{R}^k$, and $R \in \mathbb{R}$:

1. The feasible domain $\{x \in [-R,R]^n : Ax \leq b\}$ is not empty.
2. The map $x \mapsto f(x;c)$ is a convex function on the feasible domain and its subgradient is given by the linear pseudo-circuit $G_{\partial f}$.

For a more formal statement and the full proof see Section 4.3.3.

[5]The parameters of a gate determine its behavior and must be provided every time a gate of this type is used in a circuit (and thus also count towards the representation size of the circuit). For example, whenever we use a "multiplication by a constant" gate $\times\zeta$, we have to specify the constant parameter ζ of the gate. The same also applies to the linear-OPT-gate, except that it has (many) more parameters.

Remark 8. Note that f does not need to be computable by a linear arithmetic circuit. We only require that its subgradient (on the feasible domain) is given as a linear pseudo-circuit. In particular, f can be a quadratic function.

Remark 9. The linear-OPT-gate can of course also solve maximization problems $\max f(x)$ where the objective function f is concave, since this is equivalent to the problem $\min -f(x)$. In that case we have to provide a linear pseudo-circuit computing the supergradient of f, or equivalently the subgradient of $-f$.

In most of our applications, it will suffice for optimization program C to be a linear program, i.e., for f to be linear function. We will use $\mathcal{P}$ to refer to the general form of this linear program, and we will reference that in our applications.

$$\underline{\text{Linear Program } \mathcal{P}}$$

$$\begin{aligned} \min \quad & c^{\mathsf{T}} x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in [-R,R]^n \end{aligned} \tag{4.2}$$

Note that here $f(x;c) = c^{\mathsf{T}}x$, and its subgradient is simply computed by the linear pseudo-circuit $G_{\partial f} : \mathbb{R}^n \times \mathbb{R}^k \to \mathbb{R}^n$, $(x;c) \mapsto c$. The linear pseudo-circuit $G_{\partial f}$ is in fact just a normal linear arithmetic circuit here, i.e., $\ell = 0$, since no auxiliary fixed point variables are needed to compute the subgradient.

Remark 10. Note that we require that the constraint matrix A be fixed, whereas the right-hand side of the constraints b can be given as an input to the gate. This is in fact necessary, as the following example shows. If the linear-OPT-gate could solve the LP

$$\begin{aligned} \min \quad & x \\ \text{s.t.} \quad & a \cdot x \geq 1 \\ & x \in [-2,2] \end{aligned}$$

where a is not fixed, then we would obtain a linear pseudo-circuit computing $1/a$ for $a \in [1,2]$. But then, we would be able to construct a linear pseudo-circuit $F : [1,2] \to [1,2]$ computing $y \mapsto \min\{2, \max\{1, 2/y\}\}$. The only fixed point of F is at $y = \sqrt{2}$, which is a contradiction, since linear pseudo-circuits always have at least one rational fixed point.

4.3.1 Feasibility Program with Conditional Constraints

Using the linear-OPT-gate, we can also solve feasibility programs, which will be very useful throughout our applications. In particular, when constructing a linear arithmetic circuit *for the purpose of proving membership in* PPAD, we can assume without loss of generality that we have access to an additional gate solving feasibility programs with conditional constraints, which:

- is parameterized by $n, m, k \in \mathbb{N}$, a rational matrix $A \in \mathbb{R}^{m \times n}$, and linear arithmetic circuits $h_i : \mathbb{R}^k \to \mathbb{R}$ for $i = 1, \dots, m$.
- takes as input $b \in \mathbb{R}^m$, $y \in \mathbb{R}^k$, and $R \in \mathbb{R}$.

The gate outputs a feasible solution of the following feasibility problem (over variables $x \in \mathbb{R}^n$):

$$\underline{\text{Feasibility Program } Q}$$

$$\begin{aligned} h_i(y) > 0 \implies a_i^{\mathsf{T}} x &\le b_i \\ x &\in [-R, R]^n \end{aligned} \tag{4.3}$$

whenever it is feasible. Note that we may add *unconditional* constraints to feasibility program Q above by simply setting $h_i(y) = 1$ in the conditional constraints above.

Construction. We can solve the feasibility problem Q in (4.3) by solving the following optimization problem (over variables $x \in \mathbb{R}^n$)

$$\begin{aligned} \min \quad & \sum_{i=1}^{m} \max\{0, h_i(y)\} \cdot \max\{0, a_i^{\mathsf{T}} x - b_i\} \\ \text{s.t.} \quad & x \in [-R, R]^n \end{aligned} \tag{4.4}$$

Note that if x is an optimal solution for (4.4) with objective function value 0, then x is feasible for Q in (4.3). Furthermore, if Q is feasible, then the optimal value of (4.4) is 0. Thus, if Q is feasible, then any optimal solution to (4.4) will also be a feasible solution to Q in (4.3).

As a result, it suffices to show that we can use the linear-OPT-gate to solve (4.4). Clearly, the feasible domain of (4.4) is nonempty. Thus, it remains to show that we can construct a linear pseudo-circuit computing the subgradient of $x \mapsto f(x; y, b)$, where

$$f(x; y, b) = \sum_{i=1}^{m} \max\{0, h_i(y)\} \cdot \max\{0, a_i^{\mathsf{T}} x - b_i\}.$$

Note that this function is indeed convex in x.

The subgradient of $x \mapsto \max\{0, a_i^{\mathsf{T}} x - b_i\}$ can be expressed as $\mathrm{H}(a_i^{\mathsf{T}} x - b_i) \cdot a_i$, where we recall that H is the Heaviside function defined as

$$\mathrm{H}(z) = \begin{cases} 1 & \text{if } z > 0 \\ [0,1] & \text{if } z = 0 \\ 0 & \text{if } z < 0 \end{cases} .$$

Thus, the subgradient of $x \mapsto f(x; y, b)$ can be written as $\sum_{i=1}^{m} \max\{0, h_i(y)\} \cdot \mathrm{H}(a_i^{\mathsf{T}} x - b_i) \cdot a_i$. For each $i \in [m]$, we can compute the term $\max\{0, h_i(y)\} \cdot \mathrm{H}(a_i^{\mathsf{T}} x - b_i)$ by using Lemma 11 below. Since the vectors a_i are fixed, we can then compute the product $\max\{0, h_i(y)\} \cdot \mathrm{H}(a_i^{\mathsf{T}} x - b_i) \cdot a_i$. Doing this for every $i \in [m]$ and then summing up yields an element in the subgradient of $x \mapsto f(x; y, b)$. Thus, we have successfully constructed a linear pseudo-circuit computing this subgradient. It remains to prove the following lemma we used, which will also be useful later.

Lemma 11. *For the purpose of proving PPAD-membership, we can construct a* linear *pseudo-circuit computing* $(x,y) \mapsto H(x) \cdot y$.

Proof. Note that $H(x) \cdot y$ can be obtained by computing $H(x) \cdot \max\{0, y\} - H(x) \cdot \max\{0, -y\}$. Thus, it suffices to prove that we can compute $H(x) \cdot \max\{0, y\}$. This can indeed be achieved by using the linear-OPT-gate to solve the following LP (in variable $v \in \mathbb{R}$):

$$\begin{aligned} \max \quad & v \cdot x \\ \text{s.t.} \quad & 0 \le v \le \max\{0, y\} \end{aligned}$$

Note that the feasible domain is nonempty, and the gradient of the objective function is x, which can trivially be computed by a linear arithmetic circuit. It is straightforward to verify that any optimal solution v satisfies $v \in H(x) \cdot \max\{0, y\}$, as desired. □

4.3.2 Using the linear-OPT-gate in applications

In our applications in subsequent sections we will be constructing linear arithmetic circuits containing several linear-OPT-gates, corresponding to multiple optimization programs like the program $\mathcal{C}$ above, as well as feasibility programs like the program $\mathcal{Q}$. It will be helpful to be able to reference the inputs to those linear-OPT-gates as opposed to the variables of the corresponding programs, particularly because variables for one program would be inputs to the linear-OPT-gate corresponding to another program and vice versa. We will use the term *gate inputs* to refer to those inputs.

Definition 4.3.1 (Gate Inputs)**.** Consider an optimization program in the form of $\mathcal{C}$ or a feasibility program in the form of $\mathcal{Q}$ and let C be its corresponding linear-OPT-gate. We will refer to the inputs of C as *gate inputs* of the program $\mathcal{C}$ or $\mathcal{Q}$.

Using this terminology, we can argue that a specific program can be solved by the linear-OPT-gate as follows.

For optimization programs of the form $\mathcal{C}$ we need to argue

- Conditions 1 and 2 in the definition of the linear-OPT-gate for $\mathcal{C}$ above, namely that the domain is non-empty and that the subgradient of the convex objective function is given by a linear pseudo-circuit $G_{\partial f}$,
- that gate inputs appear only on the right-hand side of the constraints, but not on the left-hand side.

For feasibility programs of the form $\mathcal{Q}$ we need to argue that

- the feasibility program is *solvable* (i.e., feasible),

- the gate inputs appear only on the right-hand side of the constraints $a_i^\top x \leq b_i$,
- only gate inputs appear on the left-hand side of the conditional constraints, i.e., in the function $h_i(y) > 0$.

The conditions above are obviously equivalent to the optimization and feasibility programs having the form of (4.1) and (4.3), since the gate inputs are the inputs to the linear-OPT-gate. Here the conditions are simply "spelled-out", because it is easier to refer to them in subsequent sections. What is not obvious is how one may argue about the solvability of a feasibility program Q.

Solvability of Q. The feasibility programs that will appear in most of our applications will have the following same general form; it will be easy to argue the solvability of those that do not.

Feasibility Program Q_{app}

$$h_k(y) - h_{k'}(y) > 0 \implies w_k \leq \rho \cdot w_{k'} \quad \text{for all } k, k' \in [m]$$
$$\sum_{j=1}^{m} w_j = 1, \quad w_i \geq \frac{\rho^m}{m}, \quad \text{for all } i \in [m]$$

for some $0 < \rho \leq 1$, where $w \in \mathbb{R}^m$ are the variables. For this type of feasibility program Q_{app}, we can define the notion of a *feasibility graph.*

Definition 4.3.2 (The feasibility graph G_Q)**.** Consider a feasibility program of the form Q_{app}. Let $G_{Q_{\text{app}}}$ be the graph that has nodes for each $k \in [m]$, and a directed edge (k, k') if and only if $h_k(y) - h_{k'}(y) > 0$. We will refer to $G_{Q_{\text{app}}}$ as the *feasibility graph* of Q_{app}.

The following lemma provides a general condition for solvability of Q_{app}.

Lemma 12 (Solvability of Q_{app})**.** *A feasibility program of the form Q_{app} is solvable as long as its feasibility graph $G_{Q_{app}}$ is acyclic.*

Proof. Assume that $G_{Q_{\text{app}}}$ is acyclic, and let d_k be the length of the *longest* path from node k to a sink node in Q_{app}. Let

$$w_k = \frac{\rho^{d_k}}{\sum_{j=1}^{m} \rho^{d_j}}, \quad \text{for all } k \in [m].$$

We will argue that these values of w_k, for $k \in [m]$, satisfy the constraints of the feasibility program Q_{app}. Obviously, $\sum_{k=1}^{m} w_k = 1$ by definition. Since the graph has m nodes, it holds that $d_k \leq m$. Since $\rho \leq 1$, this implies that $\rho^{d_k} \geq \rho^m$ and that $\sum_{j=1}^{n} \rho^{d_j} \leq m$. Therefore, we obtain that $w_k \geq \frac{\rho^m}{m}$. It remains to show these values of w_k satisfy the conditional constraints. Indeed, consider an edge (k, k') in the feasibility graph Q_{app}, which, recall, corresponds to a constraint where $h_k(y) - h_{k'}(y) > 0$. Since (k, k') is an edge in Q_{app}, we have that $d_k \geq d_{k'} + 1$, as there is a path from k to a sink of Q_{app} that starts with the edge (k, k'). This implies that

$$w_k = \frac{\rho^{d_k}}{\sum_{j=1}^{m} \rho^{d_j}} \leq \frac{\rho^{d'_k+1}}{\sum_{j=1}^{m} \rho^{d_j}} = \frac{\rho \cdot \rho^{d'_k}}{\sum_{j=1}^{m} \rho^{d_j}} = \rho \cdot w_{k'},$$

and hence the corresponding conditional constraint is satisfied. □

Thus, in our applications in which the feasibility programs that we construct are of the form Q_{app} above, it suffices to show that their corresponding feasibility graph Q_{app} is acyclic, in order to establish their solvability by Lemma 12.

4.3.3 Construction and proof for the linear-OPT-gate

In this section, we provide the formal construction and the proof of correctness for the linear-OPT-gate. We state the main theorem below.

Theorem 4.3.1. *Given $n, m, k \in \mathbb{N}$, a rational matrix $A \in \mathbb{R}^{m \times n}$, rational bounds $R > 0$ and $C > 0$, as well as a linear arithmetic circuit $G_{\partial f} : \mathbb{R}^n \times \mathbb{R}^k \times [0,1]^\ell \to \mathbb{R}^n \times [0,1]^\ell$, we can construct a linear arithmetic circuit $F : \mathbb{R}^m \times \mathbb{R}^k \times [0,1]^t \to [-R,R]^n \times [0,1]^t$ in time*

$$\text{poly}(n, m, k, \text{size}(A), \text{size}(R), \text{size}(C), \text{size}(G_{\partial f}))$$

which satisfies

$F(b, c, \alpha) = (x, \alpha) \implies$ *x is an optimal solution to optimization problem C in* (4.1) *at* (b, c)

whenever b and c satisfy the following three conditions:

1. *The feasible domain $\{x \in [-R,R]^n : Ax \leq b\}$ is not empty.*
2. *The map $x \mapsto f(x; c)$ is a convex function on the feasible domain and its subgradient is given by the linear pseudo-circuit $G_{\partial f}$.*
3. *$\|c\|_\infty \leq C$.*

Remark 11. The attentive reader might have noticed that the statement of Theorem 4.3.1 does not completely correspond to what was claimed in the presentation of the linear-OPT-gate at the beginning of Section 4.3. Namely, the theorem assumes that we are given R as a parameter when we construct the gadget, whereas the earlier description allowed R to be an input to the linear-OPT-gate. Furthermore, the theorem also asks for an upper bound C on the length of c to be given as a parameter when constructing the linear-OPT-gate, whereas no such C was mentioned earlier.

Instead of assuming that R is a fixed parameter of the linear-OPT-gate, we can assume that R is an input, but that we are also given an upper bound R' on R. This can easily be achieved by applying Theorem 4.3.1 using R' as the value of the parameter

R, and explicitly adding constraints $x_i \le R$, $-x_i \le R$ (where R can now indeed be given as an input, since it only appears on the right hand side of constraints). Note that as long as we indeed have $R \le R'$, the new optimization problem is equivalent to the previous one, and the linear-OPT-gate will correctly solve it.

We still have to provide upper bounds R' and C when constructing the linear-OPT-gate. The crucial observation here is that the original description explicitly mentions that the linear-OPT-gate, as described there, can only be used *for the purpose of proving membership in* PPAD. More formally, this should be interpreted as saying: as long as we are ultimately only using the linear-OPT-gate inside a linear arithmetic circuit with bounded domain, we can assume that we do not need to explicitly provide upper bounds R' and C. The reason for this stems from Lemma 10, which states that we can bound the magnitude of any value inside a linear arithmetic circuit with bounded domain, in terms of the size of the description of the circuit, and the bound on the domain. Thus, whenever we use a linear-OPT-gate in a linear arithmetic circuit with bounded domain, since the inputs R and c are computed by a linear arithmetic circuit with bounded domain, we can compute corresponding upper bounds R' and C for the actual construction of the linear-OPT-gate gadget according to Theorem 4.3.1. An important but subtle point is that the linear-OPT-gate is always guaranteed to output an element in $[-R',R']^n$. As a result, even in a linear arithmetic circuit that uses multiple linear-OPT-gates, we can efficiently compute upper bounds on the magnitudes of numbers given as input to the linear-OPT-gate, even before replacing the linear-OPT-gates by the actual gadgets implementing them (namely the construction of Theorem 4.3.1).

Construction of the linear arithmetic circuit F

We begin with the description of how the circuit F is constructed. For this we use a sufficiently small value $\varepsilon > 0$, which we construct in the next section. We let $t := n + \ell + m$ and write $[0,1]^t = [0,1]^n \times [0,1]^{\ell+m}$.

On input $(b,c,\alpha,\beta) \in \mathbb{R}^m \times \mathbb{R}^k \times [0,1]^n \times [0,1]^{\ell+m}$, the circuit F proceeds as follows:

1. Compute $x := 2R\alpha - R$. In other words, we scale $\alpha \in [0,1]^n$ into a point $x \in [-R,R]^n$.

2. Compute $(v,\overline{\beta}_1,\dots,\overline{\beta}_\ell) := G_{\partial f}(x,c,\beta_1,\dots,\beta_\ell)$.

3. Compute $\mu_i \in \mathrm{H}(a_i^\mathsf{T} x - b_i)$ for $i = 1,\dots,m$, using auxiliary variables $\beta_{\ell+1},\dots,\beta_{\ell+m}$ and corresponding outputs $\overline{\beta}_{\ell+1},\dots,\overline{\beta}_{\ell+m}$. More formally, compute $\overline{\beta}_{\ell+i} := \min\{1,\max\{0,\beta_{\ell+i} + a_i^\mathsf{T} x - b_i\}\}$ and $\mu_i := \beta_{\ell+i}$ for $i = 1,\dots,m$.

4. Compute
$$\overline{x} := \Pi_R\left(x - \varepsilon \cdot v - A^\mathsf{T}\mu\right)$$
where Π_R denotes projection to $[-R,R]^n$.

5. Compute $\overline{\alpha} := (\overline{x} + R)/2R$ (i.e., scale $\overline{x} \in [-R,R]^n$ into a point $\overline{\alpha} \in [0,1]^n$).

6. Output $(\overline{x},\overline{\alpha},\overline{\beta})$.

Here $a_i \in \mathbb{R}^n$ denotes the ith row of matrix A.

Note that we can construct a linear arithmetic circuit computing F in time

$$\text{poly}(n,m,k,\text{size}(A),\text{size}(R),\text{size}(C),\text{size}(G_{\partial f})),$$

assuming that ε can computed in polynomial time in these quantities (which we argue in the section). In particular, the projection $\Pi_R(y)$ of some vector y can be obtained by computing $\min\{R,\max\{-R,y_i\}\}$ for each coordinate of y. Furthermore, note that the first output $\overline{x}$ of F always satisfies $\overline{x} \in [-R,R]^n$, even when the fixed point constraints $\alpha = \overline{\alpha}$ and $\beta = \overline{\beta}$ are not satisfied.

Construction of ε

We now describe how ε is constructed. Since $G_{\partial f}$ is a linear arithmetic circuit, by Lemma 10 we can compute in time $\text{poly}(\text{size}(R),\text{size}(C),\text{size}(G_{\partial f}))$ a rational $K > 0$ such that

$$\max_{(x,c,\beta)\in[-R,R]^n\times[-C,C]^n\times[0,1]^\ell} \|G_{\partial f}(x,c,\beta)\|_2 \leq K. \tag{4.5}$$

For the construction of ε we will also require the following notation. We define $\widetilde{A} \in \mathbb{R}^{(m+2n)\times n}$ and $\widetilde{b} \in \mathbb{R}^{m+2n}$ such that the system "$\widetilde{A}x \leq \widetilde{b}$" corresponds to the system "$Ax \leq b$" with the additional constraints "$x_i \leq R$" and "$-x_i \leq R$" for $i = 1,\dots,n$. In particular, the first m rows of $\widetilde{A}$ correspond to A and the first m entries in $\widetilde{b}$ correspond to b. We use $\widetilde{a}_i$ to denote the ith row of $\widetilde{A}$.

We set $\varepsilon := \gamma^*/K$, where $\gamma^* > 0$ is computed in time poly(size(A)) using the following lemma.

Lemma 13. *Given $A \in \mathbb{R}^{m\times n}$, we can construct in polynomial time a sufficiently small number $\gamma^* > 0$ such that for any nonempty $I \subseteq [m+2n]$ and for every partition of I into I_0 and I_1, such that $I_1 \neq \emptyset$, if the following optimization problem (in variables $u \in \mathbb{R}^n$ and $\lambda \in \mathbb{R}^{|I|}$)*

$$\begin{aligned} \min \quad & \sum_{i\in I_1} \widetilde{a}_i^\mathsf{T} u \\ s.t. \quad & \widetilde{a}_i^\mathsf{T} u = 0 \qquad \forall i \in I_0 \\ & \widetilde{a}_i^\mathsf{T} u \geq 0 \qquad \forall i \in I_1 \\ & \|u\|_2 = 1 \\ & u = \sum_{i\in I} \lambda_i \widetilde{a}_i \\ & \lambda_i \geq 0 \qquad \forall i \in I \end{aligned} \tag{4.6}$$

is feasible, then its optimal value γ satisfies $\gamma > \gamma^*$.

Proof. Let I, I_0, and I_1 be as specified in the statement of the lemma, and such that optimization problem (4.6) is feasible. Letting γ denote the optimal value of (4.6), note that $\gamma \geq \gamma' / \sqrt{n}$, where γ' is the optimal value of the same optimization problem, except that we replace the constraint "$\|u\|_2 = 1$" by "$\|u\|_\infty = 1$", namely:

$$\begin{aligned} \min \quad & \sum_{i \in I_1} \widetilde{a}_i^\mathsf{T} u \\ \text{s.t.} \quad & \widetilde{a}_i^\mathsf{T} u = 0 \qquad \forall i \in I_0 \\ & \widetilde{a}_i^\mathsf{T} u \geq 0 \qquad \forall i \in I_1 \\ & \|u\|_\infty = 1 \\ & u = \sum_{i \in I} \lambda_i \widetilde{a}_i \\ & \lambda_i \geq 0 \qquad \forall i \in I \end{aligned} \tag{4.7}$$

This is due to the fact that all other constraints are invariant to scaling of (u, λ). In particular, note that (4.7) is also feasible.

The optimal value γ' of (4.7) satisfies $\gamma' = \min_{k \in I, s \in \{+1,-1\}} \gamma'_{k,s}$, where $\gamma'_{k,s}$ is the optimal value of the following LP

$$\begin{aligned} \min \quad & \sum_{i \in I_1} \widetilde{a}_i^\mathsf{T} u \\ \text{s.t.} \quad & \widetilde{a}_i^\mathsf{T} u = 0 \qquad \forall i \in I_0 \\ & \widetilde{a}_i^\mathsf{T} u \geq 0 \qquad \forall i \in I_1 \\ & -1 \leq u_j \leq 1 \quad \forall j \in [n] \\ & u_k = s \\ & u = \sum_{i \in I} \lambda_i \widetilde{a}_i \\ & \lambda_i \geq 0 \qquad \forall i \in I \end{aligned}$$

if it is feasible, and $\gamma'_{k,s} := +\infty$ otherwise.

Next, we show that $\gamma'_{k,s} > 0$. This clearly holds if the LP is infeasible. Assume towards a contradiction that the LP is feasible, but $\gamma'_{k,s} \leq 0$. Then, $\gamma'_{k,s} = 0$ and it is achieved by a feasible u with $\widetilde{a}_i^\mathsf{T} u = 0$ for all $i \in I$. Since u is feasible, there exists $\lambda \geq 0$ with $u = \sum_{i \in I} \lambda_i \widetilde{a}_i$. But then $\|u\|_2^2 = u^\mathsf{T} u = \sum_{i \in I} \lambda_i \widetilde{a}_i^\mathsf{T} u = 0$, which contradicts $|u_k| = 1$.

Finally, note that the bit-complexity of the optimal value of any LP of this form is polynomially bounded by the bit-representation of matrix A. Indeed, the number of bits needed to write down such an LP for any choice of $k \in I, s \in \{+1, -1\}$, and for any I, I_0, and I_1 is bounded by some polynomial quantity in size(A). In particular, given A, we can compute in polynomial time a rational value $\gamma^* > 0$ such that $\gamma^* < \gamma'_{k,s} / \sqrt{n}$ for all $k \in I, s \in \{+1, -1\}$, and for all possible I, I_0, and I_1. By the arguments above, it then follows that $\gamma^* < \gamma$. □

Analysis: Fixed point constraints

In this section, we state and prove some simple properties that follow from the fixed point constraints $(\alpha,\beta) = (\overline{\alpha},\overline{\beta})$. Recall that the system "$\widetilde{A}x \leq \widetilde{b}$" corresponds to the system "$Ax \leq b$" with the additional constraints "$x_i \leq R$" and "$-x_i \leq R$" for $i = 1,\dots,n$. In particular, the first m rows of $\widetilde{A}$ correspond to A and the first m entries in $\widetilde{b}$ correspond to b.

The circuit F has been constructed in order to ensure that the following properties hold.

Claim 6. *If $(\alpha,\beta) = (\overline{\alpha},\overline{\beta})$, then:*

1. $\mu_i \in \mathrm{H}(a_i^{\mathsf{T}} x - b_i)$ *for* $i = 1,\dots,m$.
2. $\|v\|_2 \leq K$ *and, if* $Ax \leq b$, *then* $v \in \partial f(x;c)$.
3. $x = \overline{x}$ *and*
$$\varepsilon \cdot v + \widetilde{A}^{\mathsf{T}}\widetilde{\mu} = 0 \tag{4.8}$$
where $\widetilde{\mu} \in \mathbb{R}^{m+2n}$ *satisfies* $\widetilde{\mu} \geq 0$, *as well as*
$$\widetilde{\mu}_i > 0 \implies \widetilde{a}_i^{\mathsf{T}} x \geq \widetilde{b}_i$$
for all $i = 1,\dots,m+2n$. *Furthermore, for* $i \in [m]$ *we also have* $\widetilde{\mu}_i = \mu_i$.

Proof. The first statement follows from the construction of the Heaviside linear pseudo-circuit. The fact that $\|v\|_2 \leq K$ follows from the definition of K (see (4.5)) and the assumption that $\|c\|_\infty \leq C$. The fact that $v \in \partial f(x;c)$ when $Ax \leq b$ follows from the assumption that $G_{\partial f}$ is a linear pseudo-circuit computing $\partial f(x;c)$, whenever x lies in feasible domain.

Since $\alpha = \overline{\alpha}$, it follows that $x = \overline{x}$, i.e., $x = \Pi_R(x - \varepsilon \cdot v - A^{\mathsf{T}}\mu)$. This implies that
$$x - \varepsilon \cdot v - A^{\mathsf{T}}\mu - x = I_n\lambda^+ - I_n\lambda^-$$
where $\lambda^+, \lambda^- \in \mathbb{R}^n$ are nonnegative, and additionally satisfy $\lambda_j^+ > 0 \implies x_j = R$, and $\lambda_j^- > 0 \implies x_j = -R$. Here $I_n \in \mathbb{R}^{n\times n}$ denotes the identity matrix. Finally, noting that
$$\widetilde{A} = \begin{bmatrix} A \\ I_n \\ -I_n \end{bmatrix}$$
we can rewrite the equation as
$$\varepsilon \cdot v + \widetilde{A}^{\mathsf{T}}\widetilde{\mu} = 0$$
where we let
$$\widetilde{\mu} = \begin{bmatrix} \mu \\ \lambda^+ \\ \lambda^- \end{bmatrix}$$
Note that we indeed have $\widetilde{\mu} \geq 0$ and $\widetilde{\mu}_i > 0 \implies \widetilde{a}_i^{\mathsf{T}} x \geq \widetilde{b}_i$. In particular, for $i \in [m]$ this follows from $\widetilde{\mu}_i = \mu_i \in \mathrm{H}(a_i^{\mathsf{T}} x - b_i)$. □

In the remainder of the proof, we show that x must necessarily be an optimal solution to the optimization problem. We first show that if x is feasible, it is necessarily optimal. We complete the proof by proving that x must necessarily be feasible.

Analysis: Feasibility implies optimality

Consider the case where x is feasible, i.e., $Ax \leq b$. We will show that x must be an optimal solution. Since $x \in [-R,R]^n$ and $Ax \leq b$, it follows that $\widetilde{A}x \leq \widetilde{b}$. Thus, the third statement in Claim 6 yields that

$$\varepsilon \cdot v + \widetilde{A}^{\mathsf{T}}\widetilde{\mu} = 0 \tag{4.9}$$

where $\widetilde{\mu} \geq 0$ and where

$$\widetilde{\mu}_i > 0 \implies \widetilde{a}_i^{\mathsf{T}} x \geq \widetilde{b}_i \implies \widetilde{a}_i^{\mathsf{T}} x = \widetilde{b}_i.$$

In other words, $\widetilde{\mu}_i$ can only be strictly positive if the ith constraint is tight. Furthermore, since x is feasible, the second statement in Claim 6 yields that $v \in \partial f(x;c)$. As a result, given that $\varepsilon > 0$, the equality (4.9) can be interpreted as saying that the Karush-Kuhn-Tucker (KKT) conditions hold at point x for the following constrained minimization problem

$$\begin{aligned} \min \quad & f(x;c) \\ \text{s.t.} \quad & \widetilde{A}x \leq \widetilde{b} \end{aligned}$$

which is the same as our optimization problem C in (4.1). Since, by assumption, f is convex on the feasible domain, the KKT conditions are also sufficient for optimality, and thus x is an optimal solution.

Formally, consider any feasible point z. We will show that $f(z) \geq f(x)$. Taking the inner product of (4.9) with $z - x$ yields

$$\varepsilon \cdot v^{\mathsf{T}}(z-x) = (\widetilde{A}^{\mathsf{T}}\widetilde{\mu})^{\mathsf{T}}(x-z) = \widetilde{\mu}^{\mathsf{T}}\widetilde{A}(x-z) \geq 0$$

since $\widetilde{\mu} \geq 0$ and $\widetilde{\mu}_i > 0 \implies \widetilde{a}_i^{\mathsf{T}} x = \widetilde{b}_i \geq \widetilde{a}_i^{\mathsf{T}} z$. Given that $\varepsilon > 0$, it follows that $v^{\mathsf{T}}(z-x) \geq 0$. Finally, using the fact that $v \in \partial f(x;c)$, i.e., v is a subgradient of $f(\cdot;c)$ at x, together with the definition of subgradients, we obtain

$$f(z) \geq f(x) + v^{\mathsf{T}}(z-x) \geq f(x)$$

i.e., x is a global minimum of f on the feasible domain.

Analysis: Feasibility

We show that x must necessarily be feasible, i.e., $Ax \leq b$. Assume towards a contradiction that x is not feasible.

Let $J := \{i \in [m+2n] : \widetilde{a}_i^{\mathsf{T}} x \geq \widetilde{b}_i\}$ and note that J is nonempty, since x is infeasible. By Claim 6 we have $\widetilde{\mu}_i > 0 \implies \widetilde{a}_i^{\mathsf{T}} x \geq \widetilde{b}_i \implies i \in J$, and thus $\widetilde{\mu}_i = 0$ for all $i \in [m+2n] \setminus J$.

In particular, $\widetilde{A}^\mathsf{T}\widetilde{\mu} = \widetilde{A}_J^\mathsf{T}\widetilde{\mu}_J$, where $\widetilde{A}_J$ denotes the restriction of $\widetilde{A}$ to the subset of rows J, and similarly for $\widetilde{\mu}_J$. As a result, we can rewrite (4.8) from Claim 6 as

$$\varepsilon \cdot v + \widetilde{A}_J^\mathsf{T}\widetilde{\mu}_J = 0. \tag{4.10}$$

Let z^* be the projection of x onto the convex set $D_J := \{y \in \mathbb{R}^n : \widetilde{A}_J y \leq \widetilde{b}_J\}$, which is nonempty, since the feasible region $D := \{y \in \mathbb{R}^n : \widetilde{A}y \leq \widetilde{b}\}$ is nonempty by assumption. Note that $z^* \neq x$, because if $x \in D_J$, then we would also have $x \in D$ by the definition of J.

Let $I := \{i \in J : \widetilde{a}_i^\mathsf{T} z^* = \widetilde{b}_i\}$, i.e., the set of all constraints in J that are tight at z^*. Note that z^*, which is the projection of x onto D_J, is also the projection of x onto $D_I := \{y \in \mathbb{R}^n : \widetilde{A}_I y \leq \widetilde{b}_I\}$. Indeed, assume that z^* is not the projection of x onto D_I. Then there exists $y \in D_I$ with $(x - z^*)^\mathsf{T}(y - z^*) > 0$. But since all constraints in $J \setminus I$ are strictly satisfied at z^*, there exists a point y^* on the segment $[z^*, y]$ that satisfies $y^* \in D_J$ and $(x - z^*)^\mathsf{T}(y^* - z^*) > 0$. This is a contradiction to the fact that z^* is the projection of x onto D_J. Thus, z^* is indeed the projection of x onto D_I.

Next, we partition I into two sets: $I_0 := \{i \in I : \widetilde{a}_i^\mathsf{T} x = \widetilde{b}_i\}$ and $I_1 := \{i \in I : \widetilde{a}_i^\mathsf{T} x > \widetilde{b}_i\}$. Note that by construction $I_0 \cap I_1 = \emptyset$ and $I = I_0 \cup I_1$, since $I \subseteq J$. Furthermore, $I_1 \subseteq [m]$, because we have $x \in [-R, R]^n$, which means that $\widetilde{a}_i^\mathsf{T} x \leq \widetilde{b}_i$ for all $i \in [m+2n] \setminus [m]$. Finally, $I_1 \neq \emptyset$. Indeed, if $I_1 = \emptyset$, then $x \in D_I$, which contradicts $z^* \neq x$.

Taking the inner product of (4.10) with $z^* - x$ yields

$$\varepsilon \cdot v^\mathsf{T}(z^* - x) = (\widetilde{A}_J^\mathsf{T}\widetilde{\mu}_J)^\mathsf{T}(x - z^*) = \widetilde{\mu}_J^\mathsf{T}\widetilde{A}_J(x - z^*) \geq \widetilde{\mu}_{I_1}^\mathsf{T}\widetilde{A}_{I_1}(x - z^*) = \mu_{I_1}^\mathsf{T} A_{I_1}(x - z^*) = \sum_{i \in I_1} a_i^\mathsf{T}(x - z^*)$$

where we used $\mu_i \geq 0$ and $\widetilde{a}_i^\mathsf{T} x \geq \widetilde{b}_i \geq \widetilde{a}_i^\mathsf{T} z^*$ for all $i \in J$, $\widetilde{\mu}_{I_1} = \mu_{I_1}$, $\widetilde{A}_{I_1} = A_{I_1}$, and $\mu_i = 1$ for all $i \in I_1 \subseteq [m]$, since $\mu_i \in \mathrm{H}(a_i^\mathsf{T} x - b_i)$. As a result

$$\sum_{i \in I} a_i^\mathsf{T}(x - z^*) \leq \varepsilon \cdot v^\mathsf{T}(z^* - x) \leq \varepsilon \|v\|_2 \|x - z^*\|_2 \leq \varepsilon K \|x - z^*\|_2 \leq \gamma^* \|x - z^*\|_2 \tag{4.11}$$

where we used $\|v\|_2 \leq K$ (Claim 6) and $\varepsilon = \gamma^*/K$.

We now show that this is a contradiction to our choice of γ^*. Let $u := (x - z^*)/\|x - z^*\|_2$ and note that u is well defined, since $z^* \neq x$. Assume, for now, that u is feasible for optimization problem (4.6) from Lemma 13, which we repeat here for convenience:

$$\begin{array}{lll}
\min & \displaystyle\sum_{i \in I_1} \widetilde{a}_i^\mathsf{T} u & \\
\text{s.t.} & \widetilde{a}_i^\mathsf{T} u = 0 & \forall i \in I_0 \\
& \widetilde{a}_i^\mathsf{T} u \geq 0 & \forall i \in I_1 \\
& \|u\|_2 = 1 & \\
& u = \displaystyle\sum_{i \in I} \lambda_i \widetilde{a}_i & \\
& \lambda_i \geq 0 & \forall i \in I
\end{array}$$

In particular, the optimization problem is feasible, and thus by Lemma 13 its optimal value is (strictly) lower bounded by γ^*. As a result, $\sum_{i\in I_1}\widetilde{a}_i^\mathsf{T}(x-z^*)/\|x-z^*\|_2 > \gamma^*$, which yields $\sum_{i\in I_1}\widetilde{a}_i^\mathsf{T}(x-z^*) > \gamma^*\|x-z^*\|_2$. But this is a contradiction to (4.11).

It remains to show that $u = (x-z^*)/\|x-z^*\|_2$ is indeed feasible for optimization problem (4.6). The first two constraints are satisfied, because $\widetilde{a}_i^\mathsf{T} x = \widetilde{b}_i = \widetilde{a}_i^\mathsf{T} z^*$ for all $i \in I_0$, and $\widetilde{a}_i^\mathsf{T} x > \widetilde{b}_i = \widetilde{a}_i^\mathsf{T} z^*$ for all $i \in I_1$. Clearly, $\|u\|_2 = 1$. Thus, it remains to prove that there exists $\lambda \geq 0$ such that $x - z^* = \widetilde{A}_I^\mathsf{T}\lambda$. Since z^* is the projection of x onto $D_I = \{y \in \mathbb{R}^n : \widetilde{A}_I y \leq \widetilde{b}_I\}$, it follows that $(x-z^*)^\mathsf{T}(y-z^*) \leq 0$ for all $y \in D_I$. Given that $\widetilde{A}_I z^* = \widetilde{b}_I$, we thus obtain that $(x-z^*)^\mathsf{T} y \leq 0$ for all y with $\widetilde{A}_I y \leq 0$. However, by Farkas' lemma [37], we know that exactly one of the following two statements holds:

1. $\exists y \in \mathbb{R}^n : \quad \widetilde{A}_I y \leq 0$ and $(x-z^*)^\mathsf{T} y > 0$
2. $\exists \lambda \in \mathbb{R}^{|I|} : \quad \widetilde{A}_I^\mathsf{T}\lambda = x - z^*$ and $\lambda \geq 0$

Since we have shown that the first statement does not hold, we deduce that there exists $\lambda \geq 0$ such that $\widetilde{A}_I^\mathsf{T}\lambda = x - z^*$, as desired. As a result, $u = (x-z^*)/\|x-z^*\|_2$ is indeed feasible for optimization problem (4.6), and the proof is complete.

4.3.4 Implicit Functions and Correspondences

In this section we will construct linear arithmetic circuits computing univariate piecewise-linear functions and piecewise-constant correspondences, based on a succinct representation given by a Boolean circuit, built from Boolean AND, OR and NOT gates. We remark that conceptually, the ideas around the use of the triangle-wave function (see Definition 4.3.6) and the bit multiplication (see Definition 4.3.7 and the subsequent discussion) are attributed to Fearnley et al. [93] (private communication), who developed them, albeit for a different setting.

By interpreting non-negative integers by their binary representation, a Boolean function may be viewed as computing an integer-valued function. For $b = (b_{n-1}, \ldots, b_0) \in \{0,1\}^n$, we denote by $\mathrm{bitVal}_n(b)$ the number $\mathrm{bitVal}_n(b) = (b_{n-1}, \ldots, b_0)_2 = \sum_{i=0}^{n-1} b_i 2^i$ encoded by b in binary.

Definition 4.3.3. Let C be a Boolean circuit with n inputs and n outputs, thereby computing a function $C\colon \{0,1\}^n \to \{0,1\}^n$, and let $1 \leq N < 2^n$ be an integer. We index the inputs and outputs of C with $\{0, \ldots, n-1\}$ and we consider $b \in \{0,1\}^n$ to represent the integer $\mathrm{bitVal}_n(b)$. The integer-valued function represented by (C, N) is the function $f\colon \{0,1,\ldots,N\} \to \{0,1,\ldots,N\}$ given by

$$f(\mathrm{bitVal}_n(b)) = \min\{N, \mathrm{bitVal}_n(C(b))\} = \min\{N, (C(b)_{n-1}, \ldots, C(x)_0)_2\} \ ,$$

for $b \in \{0,1\}^n$ such that $\mathrm{bitVal}_n(b) \leq N$.

A natural piecewise-linear function is then given by linear interpolation between function values.

Definition 4.3.4. Let C be a Boolean circuit with n inputs and n outputs, and let $1 \le N < 2^n$ be an integer. The piecewise-linear function represented by (C, N) is the function $g \colon \mathbb{R} \to \mathbb{R}$ given by

$$g(x) = \begin{cases} f(0) & \text{for } x < 0 \\ (1-(x-\lfloor x \rfloor))f(\lfloor x \rfloor) + (x - \lfloor x \rfloor) f(\lfloor x \rfloor + 1) & \text{for } 0 \le x < N \\ f(N) & \text{for } N \le x \end{cases},$$

where f is the integer-valued function represented by (C, N).

In a similar way, a natural piecewise-constant correspondence is given by extending function values to the right, until the next function value is defined.

Definition 4.3.5. Let C be a Boolean circuit with n inputs and n outputs, and let $1 \le N < 2^n$ be an integer. The piecewise-constant correspondence represented by (C, N) is the correspondence $g \colon \mathbb{R} \rightrightarrows \mathbb{R}$ given by

$$g(x) = \begin{cases} f(0) & \text{for } x \le 0 \\ [\min(f(x-1), f(x)), \max(f(x-1), f(x))] & \text{for } x \in \{1, 2, \dots, N\} \\ f(\lfloor x \rfloor) & \text{for } x \in [0, N] \setminus \{0, 1, \dots, N\} \\ f(N) & \text{for } N < x \end{cases},$$

where f is the integer-valued function represented by (C, N).

An example illustrating Definition 4.3.4 and Definition 4.3.5 is given in Figure 4.2.

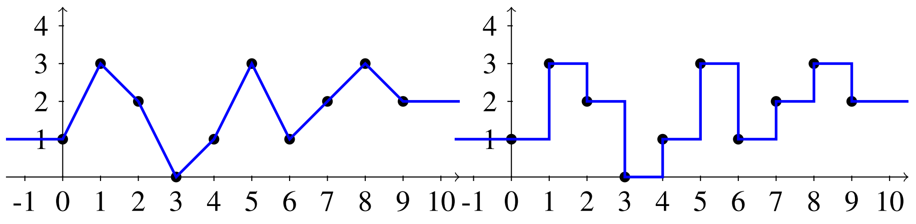


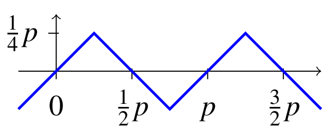
Figure 4.2: The piecewise-linear function (left) and piecewise-constant correspondence (right) defined implicitly by a Boolean circuit C and $N = 9$ computing an integer valued function f as shown with dots.

A function that will be used in both constructions is the familiar square-wave function, restricted to a bounded number of periods, as illustrated in Figure 4.3.

Definition 4.3.6. For a non-negative integer n, the correspondence $S_n \colon \mathbb{R}^2 \rightrightarrows [0, 1]$, consisting of $2^n + 1/2$ periods of length p, in the interval $[0, (2^n + 1/2)p]$, is defined as $\mathrm{H}(T_n(x, p))$, where T_n is the triangle-wave function defined inductively as follows. For $n = 0$, we define

$$T_0(x, p) = \max(\min(x, p/2 - x), \min(x - p, 3p/2 - x)) ,$$

and for $n > 0$ we define $T_n(x, p) = T_{n-1}(\min(x, (2^n + 1/2)p - x), p)$.



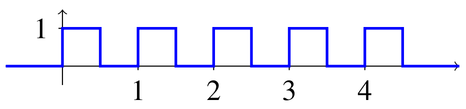


Figure 4.3: The function $T_0(x,p)$ used to define $S_0(x,p)$ (left) and the function $S_2(x,1)$ (right) giving a square wave with 4.5 periods of length 1 in the interval $[0,4.5]$.

Boolean operations may be viewed as special cases of linear functions, namely the Boolean functions $\mathsf{AND}(a,b)$, $\mathsf{OR}(a,b)$, and $\mathsf{NOT}(a)$ correspond to the linear functions $\min(a,b)$, $\max(a,b)$, and $1-a$, for $a,b \in \{0,1\}$. The following is thus immediate.

Lemma 14. *Let C be a Boolean circuit with n inputs and m outputs. Then there is a linear arithmetic circuit with n inputs and m outputs C' with the same number of gates as C such that $C(b) = C'(b)$ for all $b \in \{0,1\}^n$.*

In fact, $\mathsf{AND}(a,b)$ and thus $\min(a,b)$ may also be viewed as the multiplication operation for $a,b \in \{0,1\}$. Even more generally, $\min(a,b)$ may be viewed as a multiplication when just $b \in \{0,1\}$ and $a \in [0,1]$. As long as we ultimately only use this tool to prove PPAD-membership, we can make use of the linear-OPT-gate to construct a stronger multiplication gadget that does not require an a priori bound on the operand a.

Definition 4.3.7. Define the linear pseudo-circuit bitMult with two inputs and one output by $\mathrm{bitMult}(a,b) = \mathrm{H}(b-\frac{1}{2})a$.

Note that bitMult can indeed be computed by a linear pseudo-circuit by Lemma 11 (which uses the linear-OPT-gate internally). From the definition of the Heaviside function H the following is then immediate.

Lemma 15. *We have* $\mathrm{bitMult}(a,b) = ab$*, for any $a \in \mathbb{R}$ and any $b \in \{0,1\}$.*

We will show below that also the piecewise-linear functions and piecewise-constant correspondences represented by Boolean circuits may be computed by linear pseudo-circuits. In fact, for the case of piecewise-constant correspondences we will show a considerably stronger statement allowing the scaling of the input and output by a variable.

An important component of both constructions is *bit extraction*. Informally, bit extraction of a non-negative $x \geq 0$ refers to computing the binary expansion of $\lfloor x \rfloor$. This is clearly an inherently discontinuous operation, but can nevertheless be circumvented in our applications. The bit extraction procedure we define below will actually take 2 inputs x and y, and the intention is to perform bit extraction on the number x/y, when $0 < y \leq 1$.

Definition 4.3.8. For a positive integer n, we inductively define a linear pseudo-circuit bitExt_n with two inputs and n outputs as follows. For $n = 1$, we define $\mathrm{bitExt}_1(x,y) = \mathrm{H}(x-y)$, and for $n > 1$ we define $\mathrm{bitExt}_n(x,y) = (b_{n-1},\dots,b_0)$ where $b_{n-1} = \mathrm{H}(x - 2^{n-1}y)$ and $(b_{n-2},\dots,b_0) = \mathrm{bitExt}_{n-1}(x - \mathrm{bitMult}(2^{n-1}y, b_{n-1}), y)$.

Lemma 16. *Let $n \geq 1$, assume $0 < y \leq 1$, and let $\mathrm{bitExt}_n(x,y) = b = (b_{n-1},\dots,b_0)$. Then, whenever $x/y \in [0,2^n] \setminus \{1,2,3,\dots,2^n\}$ we have $b_i \in \{0,1\}$, for all i and $\mathrm{bitVal}_n(b) = \sum_{i=0}^{n-1} b_i 2^i = \lfloor x/y \rfloor$.*

Proof. Assume $x/y \in [0,2^n] \setminus \{1,2,3,\dots,2^n\}$. Since in particular $x/y \neq 2^{n-1}$, by definition of the Heaviside function H we have that $b_{n-1} = 0$ when $x/y < 2^{n-1}$ and $b_{n-1} = 1$ when $x/y > 2^{n-1}$. This shows the case of $n = 1$. For $n > 1$ we have $x/y - b_{n-1}2^{n-1} \in [0,2^{n-1}] \setminus \{1,2,\dots,2^{n-1}\}$, and by induction we have $\lfloor x/y - b_{n-1}2^{n-1} \rfloor = \sum_{i=0}^{n-2} b_i 2^i$. The result now follows since $\lfloor x/y - b_{n-1}2^{n-1} \rfloor = \lfloor x/y \rfloor - b_{n-1}2^{n-1}$, for $b_{n-1} \in \{0,1\}$. □

Remark 12. The output of bitExt_n is of course well-defined for any value of x and y. In particular we can observe that $(\mathrm{bitExt}_n(x,0))_i = \mathrm{H}(x)$, for all i.

As an extension of Definition 4.3.7 and Lemma 15 we may in a linear arithmetic circuit *multiply* a number given in binary encoding by variables with another variable. This means that we may consider the output of a linear circuit, obtained from a Boolean circuit by Lemma 14, as representing a number given in binary encoding and then multiply that by a variable.

Definition 4.3.9. For an integer $n > 0$, define the linear pseudo-circuit $\mathrm{bitMult}_n$ having $n+1$ inputs and one output by $\mathrm{bitMult}_n(a, b_{n-1},\dots,b_0) = \sum_{i=0}^{n-1} \mathrm{bitMult}(a,b_i)2^i$.

Using Lemma 15 we then have the following.

Lemma 17. *For any positive integer n, $b \in \{0,1\}^n$, and any $a \in \mathbb{R}$ we have $\mathrm{bitMult}_n(a,b_{n-1},\dots,b_0) = a \cdot \mathrm{bitVal}_n(b)$.*

For $b \in \{0,1\}^n$ we shall for brevity also simply write $\mathrm{bitMult}(a,b) = \mathrm{bitMult}_n(a,b_{n-1},\dots,b_0)$.

Piecewise-linear functions

We will now develop a construction of a linear pseudo-circuit computing the piecewise-linear function g represented by (C,N), as defined in Definition 4.3.4, for a circuit C with n inputs and n outputs and an integer N satisfying $1 \leq N < 2^n$. The general idea is to perform bit extraction, evaluate the integer-valued function f represented by (C,N) on two inputs, and then perform linear interpolation. The immediate obstacle is that bit extraction applied to $x \in [0,2^n]$ *fails*, i.e., does not guarantee a Boolean output, when $x \in \{1,\dots,2^n\}$. To overcome this obstacle we perform this procedure on two different translations of the input, and use the square wave function to switch between the two, thereby masking out erroneous values. Let in the following $G = [0,2^n] \setminus \{1,\dots,2^n\}$ be the set of points where bit extraction succeeds.

For simplicity, we first give a description of the computation without referring to Boolean circuits. Suppose $x \in [0,N]$. We consider the following two cases, each with a special case in an endpoint of the interval $[0,N]$.

1. Assume $x \in [\lfloor x \rfloor, \lfloor x \rfloor + \frac{1}{2}]$ and assume first also $x \leq N - \frac{1}{2}$. Let $w_1 = x + \frac{1}{4}$ and $w_2 = x + \frac{5}{4}$. Note that $w_1, w_2 \in G$ and that $\lfloor w_1 \rfloor = \lfloor x \rfloor$ and $\lfloor w_2 \rfloor = \lfloor x \rfloor + 1$. It thus follows that $g(x) = g_1(x)$, where

   $$g_1(x) = (\lfloor w_2 \rfloor - x) f(\lfloor w_1 \rfloor) + (x - \lfloor w_1 \rfloor) f(\lfloor w_2 \rfloor) \ .$$

   To also handle the special case of $x = N$, we proceed as follows. First, let $\delta_1 = \mathrm{H}(x - (N - \frac{1}{4}))$ and define $w_1' = x + \frac{1}{4} - \delta_1$ and $w_2' = x + \frac{5}{4} - \delta_1$. In case $x = N$ we have $\delta_1 = 1$ and thus $w_1' = x - \frac{3}{4}$, $w_2' = w_1 = x + \frac{1}{4}$, and $w_1', w_2' \in G$. We then have $g(x) = g_1'(x)$, where

   $$g_1'(x) = (\lfloor w_2' \rfloor - x) f(\lfloor w_1 \rfloor) + (x - \lfloor w_1' \rfloor) f(\lfloor w_2' \rfloor) \ .$$

   Note that f is evaluated on $\lfloor w_1 \rfloor$ and $\lfloor w_2' \rfloor$ (and not $\lfloor w_1' \rfloor$ and $\lfloor w_2' \rfloor$), which in the case of $x = N$ both equal N, thereby satisfying Definition 4.3.4.

   Finally, note that in the general case, when $x \leq N - \frac{1}{2}$, we have $\delta_1 = 0$, and thus $w_1' = w_1$ and $w_2' = w_2$ which means that $g_1'(x) = g_1(x)$.

2. Assume $x \in [\lceil x \rceil - \frac{1}{2}, \lceil x \rceil]$ and assume first also $x \geq \frac{1}{2}$. Let $w_3 = x - \frac{1}{4}$ and $w_4 = x + \frac{3}{4}$. Note that $w_3, w_4 \in G$. We now have two sub-cases. First, assume that $x < \lceil x \rceil$. Then $\lfloor w_3 \rfloor = \lfloor x \rfloor$ and $\lfloor w_4 \rfloor = \lfloor x \rfloor + 1$. Next assume that $x = \lceil x \rceil$. Then $\lfloor w_3 \rfloor = \lfloor x \rfloor - 1$ and $\lfloor w_4 \rfloor = \lfloor x \rfloor$. In both sub-cases it thus follows that $g(x) = g_2(x)$, where

   $$g_2(x) = (\lfloor w_4 \rfloor - x) f(\lfloor w_3 \rfloor) + (x - \lfloor w_3 \rfloor) f(\lfloor w_4 \rfloor) \ .$$

   To also handle the special case of $x = 0$, we proceed in a similar way as the case of $x = N$ above. First, let $\delta_2 = \mathrm{H}(\frac{1}{4} - x)$ and define $w_3' = x - \frac{1}{4} + \delta_2$ and $w_4' = x + \frac{3}{4} + \delta_2$. In case $x = 0$ we have $\delta_2 = 1$ and thus $w_3' = w_4 = x + \frac{3}{4}$, $w_4' = x + \frac{7}{4}$ and $w_3', w_4' \in G$. We then have $g(x) = g_2'(x)$, where

   $$g_2'(x) = (\lfloor w_4' \rfloor - x) f(\lfloor w_3' \rfloor) + (x - \lfloor w_3' \rfloor) f(\lfloor w_4 \rfloor) \ .$$

   Note that f is evaluated on $\lfloor w_3' \rfloor$ and $\lfloor w_4 \rfloor$ (and not $\lfloor w_3' \rfloor$ and $\lfloor w_4' \rfloor$), which in the case of $x = 0$ both equals 0, thereby satisfying Definition 4.3.4.

   Finally, note that in the general case, when $x \geq \frac{1}{2}$, we have $\delta_2 = 0$, and thus $w_3' = w_3$ and $w_4' = w_4$ which means that $g_2'(x) = g_2(x)$.

We will now simply use the square wave function to select between the two cases. Namely, from the definition of S_n we have that $g(x) = g_2'(x) + S_n(x,1)(g_1'(x) - g_2'(x))$. Since $S_n(x,1)$ is defined as the composition $H(T_n(x,1))$ of the Heaviside function with the linear function $T_n(x,1)$, we can compute the term $S_n(x,1)(g_1'(x) - g_2'(x))$ using the linear-OPT-gate by Lemma 11. We now present the general construction of the linear pseudo-circuit.

Proposition 4. *Let C be a Boolean circuit with n inputs and n outputs, and let $N < 2^n$. Then, for the purpose of proving PPAD-membership, we can construct a linear pseudo-circuit computing the piecewise-linear function g represented by (C, N).*

Proof. Let C' be the linear arithmetic circuit obtained from C by Lemma 14. We construct a linear pseudo-circuit computing $g(x)$, operating as follows.

1. Compute $x' = \max(0, \min(N, x))$. Thus $x' \in [0, N]$ and $g(x) = g(x')$.
2. Compute $\delta_1 = H(x' - (N - \frac{1}{4}))$ and $\delta_2 = \mathrm{H}(\frac{1}{4} - x')$ using Lemma 11.
3. Compute $w_1 = x' + \frac{1}{4}$, $w_1' = x' + \frac{1}{4} - \delta_1$, and $w_2' = x' + \frac{5}{4} - \delta_1$.
4. Compute $\mathbf{b}_1 = \mathrm{bitExt}_n(w_1, 1)$, $\mathbf{b}_1' = \mathrm{bitExt}_n(w_1', 1)$, and $\mathbf{b}_2' = \mathrm{bitExt}_n(w_2', 1)$.
5. Compute $g_1' = \mathrm{bitMult}(\mathrm{bitVal}(\mathbf{b}_2') - x', C'(\mathbf{b}_1)) + \mathrm{bitMult}(x' - \mathrm{bitVal}(\mathbf{b}_1'), C'(\mathbf{b}_2'))$.
6. Compute $w_4 = x' + \frac{3}{4}$, $w_3' = x' - \frac{1}{4} + \delta_2$, and $w_4' = x' + \frac{3}{4} + \delta_2$.
7. Compute $\mathbf{b}_4 = \mathrm{bitExt}_n(w_4, 1)$, $\mathbf{b}_3' = \mathrm{bitExt}_n(w_3', 1)$, and $\mathbf{b}_4' = \mathrm{bitExt}_n(w_4', 1)$.
8. Compute $g_2' = \mathrm{bitMult}(\mathrm{bitVal}(\mathbf{b}_4') - x', C'(\mathbf{b}_3')) + \mathrm{bitMult}(x' - \mathrm{bitVal}(\mathbf{b}_3'), C'(\mathbf{b}_4))$.
9. Output $g_2' + S_n(x', 1)(g_1' - g_2')$, where the last term is computed using Lemma 11.

The correctness of the circuit follows from the preceding case analysis. □

Piecewise-linear correspondences

We will now, in a similar way, develop a construction of a linear pseudocircuit computing the piecewise-constant correspondence g represented by (C, N), as defined in Definition 4.3.5, for a circuit C with n inputs and n outputs and an integer N satisfying $1 \leq N < 2^n$. Again the general idea is to perform bit extraction and evaluate the integer-valued function f represented by (C, N) on two inputs. Since the function we compute is piecewise-constant, we are now able to also scale the output by a variable. Let again $G = [0, 2^n] \setminus \{1, \ldots, 2^n\}$ be the set of points where bit extraction succeeds.

We will be able to compute the correspondence $z \cdot g(x/y)$, but for simplicity we first describe the computation of $g(x)$, also without referring to Boolean circuits. Thus, suppose now $x \in [0, N]$. We consider the same cases as above.

1. Assume $x \in [\lfloor x \rfloor, \lfloor x \rfloor + \frac{1}{2}]$. Let $w_1 = x + \frac{1}{4}$, and note that $w_1 \in G$ and that $\lfloor w_1 \rfloor = \lfloor x \rfloor$. Let $g_1(x) = f(\lfloor w_1 \rfloor)$. It follows that $g_1(x) = f(\lfloor x \rfloor)$.
2. Assume $x \in [\lceil x \rceil - \frac{1}{2}, \lceil x \rceil]$ and assume first also $x \geq \frac{1}{2}$. Let $w_2 = x - \frac{1}{4}$, and note that $w_2 \in G$. Let $g_2(x) = f(\lfloor w_2 \rfloor)$. We now have two subcases. First, assume that $x < \lceil x \rceil$. Then $\lfloor w_2 \rfloor = \lfloor x \rfloor$ and thus $g_2(x) = f(\lfloor x \rfloor)$. Next assume that $x = \lceil x \rceil$. Then $\lfloor w_2 \rfloor = \lfloor x \rfloor - 1$ and thus $g_2(x) = f(\lfloor x \rfloor - 1)$. To also handle the special case of $x = 0$, we proceed as follows. First, let $\delta = \mathrm{H}(\frac{1}{4} - x)$ and define $w_2' = x - \frac{1}{4} + \delta$. Let $g_2'(x) = f(\lfloor w_2' \rfloor)$. In case $x = 0$ we have $\delta = 1$ and thus $w_2' = x + \frac{3}{4}$ and $w_2' \in G$. We thus have $\lfloor w_2' \rfloor = \lfloor x \rfloor$ and thus $g_2'(x) = f(\lfloor x \rfloor)$.

   Finally, note that in the general case, when $x \geq \frac{1}{2}$, we have $\delta = 0$, and thus $w_2' = w_2$ which means that $g_2'(x) = g_2(x)$.

We will now simply use the square wave function to select between the two cases, letting $g(x) = g'_2(x) + S_n(x,1)(g_1(x) - g'_2(x))$.

This will also be what gives the correct result in case $x \in \{1,\dots,N\}$ where we have $g_1(x) = f(x)$ and $g'_2(x) = f(x-1)$ and selecting using the square-wave function yields gives any convex combination of these values. For the special case of $x = 0$ we also obtain the correct result, since $g_1(0) = g'_2(0) = f(0)$.

Since $S_n(x,1)$ is defined as the composition $H(T_n(x,1))$ of the Heaviside function with the linear function $T_n(x,1)$, we can compute the term $S_n(x,1)(g_1(x) - g'_2(x))$ using Lemma 11. We now present the general construction of the linear pseudo-circuit.

Proposition 5. *Let C be a Boolean circuit with n inputs and n outputs, and let $N < 2^n$. Then, for the purpose of proving PPAD-membership, we can construct a linear pseudo-circuit computing a function $h(x,y,z)$ such that when $0 < y \leq 1$ we have $h(x,y,z) = z \cdot g(x/y)$, where g is the piecewise-constant function represented by (C,N).*

Proof. Let C' be the linear arithmetic circuit obtained from C by Lemma 14. We construct a linear pseudo-circuit computing $h(x,y,z)$, operating as follows.

1. Compute $x' = \max(0, \min(yN, x))$. Thus when $y > 0$ we have $\frac{x'}{y} \in [0,N]$ and $g(\frac{x}{y}) = g(\frac{x'}{y})$.
2. Compute $\delta = \mathrm{H}(\frac{1}{4}y - x')y$ using Lemma 11.
3. Compute $w_1 = x' + \frac{1}{4}y$ and $\mathbf{b}_1 = \mathrm{bitExt}_n(w_1, y)$.
4. Compute $g_1 = \mathrm{bitMult}(z, C'(\mathbf{b}_1))$.
5. Compute $w'_2 = x' - \frac{1}{4}y + \delta$ and $\mathbf{b}'_2 = \mathrm{bitExt}_n(w'_2, y)$.
6. Compute $g'_2 = \mathrm{bitMult}(z, C'(\mathbf{b}'_2))$.
7. Output $g'_2 + S_n(x', y)(g_1 - g'_2)$, where the last term is computed using Lemma 11.

The correctness of the circuit follows from the preceding case analysis, using $\frac{x}{y}$ in place of x in the analysis. □

4.4 Concave Games, Nash Equilibria and Other Equilibrium Notions

The first application of our linear-OPT-gate will be to the domain of strategic games, for computing Nash equilibria, as well as other related equilibrium notions. To demonstrate the use of our technique, we first "warm up" with two basic applications, namely (a) computing Nash equilibria in *bimatrix games* (Section 4.4.1) and (b) computing fixed point solutions of *digraph threshold games* (Section 4.4.2). These two applications will be used as examples to demonstrate the use of our linear-OPT-gate via linear programs, similar to linear program $\mathcal{P}$ (see Program (4.2) in Section 4.3) or via feasibility programs, similar to feasibility program Q (see Program (4.3) in Section 4.3) respectively.

Then in Section 4.4.3 we define a very large class of games, which we refer to as *games with* linear *best response oracles (LBRO games)*. This class has a technical definition, and essentially captures every game in which the best response of the agent can be computed by a linear arithmetic circuit (or, to be more precise, a linear pseudo-circuit). The PPAD-membership of LBRO games is then immediate from our results in Section 4.3 via the use of our linear-OPT-gate. This provides us with a very strong unified tool to show the PPAD-membership of a very large class of games, by merely showing that these games are LBRO games.

More precisely, in Section 4.4.3 we consider *generalized equilibria* in *concave games*. Our PPAD-membership proof will capture a large class of concave games that subsumes bimatrix games, polymatrix games and more generally a class of games that we call linear succinct games (recall Remark 7 for the interpretation of the term linear), as well as bilinear games, (extended) digraph threshold games, and essentially all concave games for which Nash equilibria in rational numbers exist. In Section 4.4.3, we consider an alternative equilibrium notion, due to Kintali et al. [153], called *personalized equilibrium*, and in Section 4.4.3 we consider ε-proper equilibria in linear succinct games (including polymatrix games), a well-known equilibrium notion due to Myerson [174].

With the linear-OPT-gate at hand, the proofs that we will develop in this section will be conceptually very simple, basically mimicking the simplest proofs of existence for these settings. More precisely, all of our PPAD-membership results for this section are obtained as special cases of our main theorem for LBRO games (see Theorem 4.4.3). In fact, we also employ the same theorem to obtain PPAD-membership results for congestion games in Section 4.5.

4.4.1 Warm-up: Nash Equilibria in Bimatrix Games via Linear Programs

We define bimatrix games below.

Definition 4.4.1 (Bimatrix Game)**.** A *bimatrix game* is a strategic (normal form) game played between two players. Each player $i \in \{1,2\}$ has a finite set of pure strategies $S_i = \{1,\dots,m_i\}$, and a payoff function $u_i = S_1 \times S_2 \to \mathbb{R}$, mapping pairs of pure strategies to payoffs. The players can also randomize over their pure strategies. A *mixed strategy* $\mathbf{x}_i$ of player i is a probability distribution over pure strategies in S_i. The domain Σ_i of mixed strategies for player i is the $(m_i - 1)$-dimensional unit simplex, i.e., $\Sigma_i := \{y \in \mathbb{R}^{m_i}_{\geq 0} \colon \sum_{j=1}^{m_i} y_j = 1\}$. Given a mixed strategy $\mathbf{x}_i$ for player i, the jth coordinate $x_{i,j}$ denotes the probability that player i is playing pure strategy j. The expected payoff $\tilde{u}_i(\mathbf{x}_1,\mathbf{x}_2)$ of player i given a pair of mixed strategies $(\mathbf{x}_1,\mathbf{x}_2)$ is defined as:

$$\tilde{u}_i(\mathbf{x}_1,\mathbf{x}_2) = \sum_{j_1=1}^{m_1} \sum_{j_2=1}^{m_2} x_{1,j_1} \cdot x_{2,j_2} \cdot u_i(j_1, j_2) \tag{4.12}$$

The name "bimatrix games" comes from the fact that for 2 players, the payoffs can be expressed via two $m_1 \times m_2$ matrices, one for each player.

Definition 4.4.2 (Nash equilibrium)**.** A pair of mixed strategies $(\mathbf{x}_1, \mathbf{x}_2) \in \Sigma_1 \times \Sigma_2$ is a (mixed) *Nash equilibrium* of a bimatrix game if for any player $i \in \{1,2\}$ and any mixed strategy $\mathbf{x}'_i \in \Sigma_i$ of that player, it holds that $\tilde{u}_i(\mathbf{x}_i, \mathbf{x}_{3-i}) \geq \tilde{u}_i(\mathbf{x}'_i, \mathbf{x}_{3-i})$, i.e., no player can improve their expected payoff by unilaterally deviating to any other mixed strategy.

Finding Nash equilibria. We are interested in finding a Nash equilibrium of a bimatrix game, when the payoff functions u_i for each player $i \in \{1,2\}$ are given explicitly as rational numbers. This is a total search problem, since a solution is always guaranteed to exist by Nash's Theorem [177], the proof of which employs the Kakutani fixed point theorem [149]. Papadimitriou [182] showed that the problem lies in PPAD, which also implies the existence of rational solutions, i.e., Nash equilibria in which the mixed strategies of the players are rational numbers. Note that Papadimitriou's proof essentially appeals to an alternative proof of Nash equilibrium existence due to Cottle and Dantzig [64] that formulates the problem as an LCP (see also [161]); we discussed the general challenges of using this approach in Section 4.1.2. Indeed, even in the simple case of bimatrix games, an argument is required against ray termination.[6] In contrast, our linear-OPT-gate allows us to organically obtain PPAD-membership from the standard, textbook existence proof of Nash [177], without any further arguments.

In a celebrated paper, Chen et al. [55] showed that the problem is in fact PPAD-complete (and the hardness holds even when one aims to find approximate equilibria). Of course, Nash's theorem applies to strategic games beyond bimatrix games, i.e., games with more than 2 players. In that case however, it was known already from Nash's original work that with 3 or more players, there are games for which all Nash equilibria are irrational. For those games, it has been shown that the problem of computing exact Nash equilibria is FIXP-complete [86].

We will demonstrate how our linear-OPT-gate, in particular through its capability of solving linear programs of the form $\mathcal{P}$ (see Program (4.2) in Section 4.3), can be used to rather straightforwardly show that computing Nash equilibria in bimatrix games is in PPAD. For ease of notation, let $-i = 3 - i$. We start by observing that given a mixed strategy $\mathbf{x}_{-i}$ for the other player, player i can find an optimal (i.e., an expected payoff-maximizing) mixed strategy $\mathbf{y}_i$ via the solution to the following linear program:

<u>Linear Program $\mathcal{P}$</u>

$$\begin{aligned} \text{maximize} \quad & \tilde{u}_i(\mathbf{y}_i, \mathbf{x}_{-i}) \\ \text{subject to} \quad & \sum_{j=1}^{m_i} y_{ij} = 1 \\ & y_{ij} \geq 0 \qquad j = 1, \ldots, m_i \end{aligned}$$

[6] Such an argument can be found in [63] and also implicitly in [64] and [160].

"Spelling out" the objective function by substituting Equation (4.12) into it, it becomes:

$$\tilde{u}_i(\mathbf{y}_i, \mathbf{x}_{-i}) = \sum_{j_i=1}^{m_i} \sum_{j_{-i}=1}^{m_{-i}} y_{i,j_i} \cdot x_{-i,j_{-i}} \cdot u_i(j_i, j_{-i}) \tag{4.13}$$

Notice that Equation (4.13) is linear in $\mathbf{y}_i$, and hence $\mathcal{P}$ is indeed a linear program.

Now, to prove membership of finding Nash equilibria in bimatrix games in PPAD, all we need to do is to construct a function whose fixed points will be the Nash equilibria of the bimatrix game, and argue that this function can be computed by a linear arithmetic circuit containing linear-OPT-gates. This is essentially the straightforward translation of the proof via the Kakutani fixed point theorem to a PPAD-membership proof.

Theorem 4.4.1. *Computing a Nash equilibrium of a bimatrix game is in PPAD.*

Proof. We construct a function $F : \Sigma_1 \times \Sigma_2 \to \Sigma_1 \times \Sigma_2$, in which $F_i(\mathbf{x}_1, \mathbf{x}_2)$ is the optimal solution of linear program $\mathcal{P}$ for player i. Since each linear program computes a best-response mixed strategy for the corresponding player, the resulting pair of mixed strategies at a fixed point of F is a Nash equilibrium. It remains to show that the linear program $\mathcal{P}$ can be computed by our linear-OPT-gate. Indeed, all the constraints are linear functions of the variables $\mathbf{y}_i$, and the gate inputs $\mathbf{x}_{-i}$ do not appear in the constraints. The feasible domain is non-empty and bounded, and the gradient of the objective function (with respect to the variables $\mathbf{y}_i$) is linear in the variables $\mathbf{y}_i$ and gate inputs $\mathbf{x}_{-i}$ and hence can be computed by a linear pseudo-circuit. □

Before we conclude the section, we remark that for (general) games of 3 or more players, the objective function would be a polynomial of degree at least 3 in the variables and the gate inputs, and its gradient could not be given by a linear pseudo-circuit. This is of course not a coincidence, as, as we said earlier, games with 3 or more players may only have irrational Nash equilibria.

4.4.2 Warm-up: Equilibria of Digraph Threshold Games via Feasibility Programs

Our next "warm-up" application is that of finding equilibria in *digraph threshold games*. A digraph threshold game is played on a directed graph between n players that choose a real number in $[0,1]$, and their payoffs depend on the chosen numbers of their incoming neighbors and a threshold. Digraph Threshold games were defined by Papadimitriou and Peng [181][7] as a tool for proving PPAD-hardness of *public goods*

[7] Papadimitriou and Peng [181] used the term "threshold games" to refer to these games. However, this term is ambiguous, as it has also been used to refer to a specific class of congestion games, see [2]. In fact, both variants have been used in relation to the general class of public goods games [154, 181], which adds to the potential for confusion. For this reason, we have added the term "digraph" in front of the name for distinction.

games on directed graphs, and were later on used to establish the PPAD-hardness of other problems, e.g., see [50, 58][8].

Definition 4.4.3 (Digraph Threshold Game)**.** A digraph threshold game $\mathcal{G}(V, E, t)$ is defined on a directed graph $G = (V, E)$, and $t \in (0, 1)$ is a threshold. The nodes of G correspond to players, and each player $i \in V$ chooses a strategy $x_i \in [0, 1]$. Let $\mathbf{x} = (x_1, \ldots, x_n)$ be the vector of (pure) strategies that we will refer to as a (pure) strategy profile. Let $N(i)$ denote the (in-)*neighborhood* of player $i \in V$, i.e., the set consisting of all the players that have arcs to player i, i.e., $N(i) = \{j : (j, i) \in E\}$. A strategy profile $\mathbf{x}$ is an equilibrium if it satisfies

$$x_i = \begin{cases} 0 & \sum_{j \in N(i)} x_j > t \\ 1 & \sum_{j \in N(i)} x_j < t \\ [0,1] & \sum_{j \in N(i)} x_j = t \end{cases} \tag{4.14}$$

where in the above expression we have abused notation, using $x_i = [0, 1]$ to denote $x_i \in [0, 1]$.

While we did use the terminology of games, digraph threshold games can alternatively be viewed as finding fixed point solutions to a set of constraints. This is why we chose this application to demonstrate the use of our feasibility programs Q (see Program (4.3) in Section 4.3) which we can obtain from our linear-OPT-gate. Indeed, Equation (4.14) can straightforwardly be written as a feasibility program Q as follows:[9]

Feasibility Program Q

$$\sum_{j \in N(i)} x_j > t \Rightarrow x_i = 0$$

$$\sum_{j \in N(i)} x_j < t \Rightarrow x_i = 1$$

$$0 \le x_i \le 1$$

Note that x_i is the only variable of this feasibility program, while the x_j, $j \in N(i)$, are gate inputs.

In turn, membership in PPAD follows rather easily by constructing an appropriate function whose fixed point coordinates are the outcomes of these feasibility programs (one for each player), and arguing that it can be computed by a linear arithmetic circuit containing linear-OPT-gates. We have the following theorem.

[8]To be precise, what is used for PPAD-hardness is their approximate version, which was shown to be PPAD-complete in [181]. Here we show the membership in PPAD for the exact version, strengthening the membership result.

[9]The perceptive reader might observe that digraph threshold games look a lot like the Heaviside function introduced in Example 3. Indeed, one could obtain the PPAD-membership of digraph threshold games as fixed points of functions containing only Heaviside linear pseudo-circuits rather than linear-OPT-gates. We elected to use the feasibility program formulation instead, to introduce the reader to its use in light of further applications to come later. Also note that the feasibility program Q contains equalities; these can easily be modified to inequalities by splitting the constraints into two.

Theorem 4.4.2. *Computing an equilibrium of a digraph threshold game is in PPAD.*

Proof. We construct a function $F : [0,1]^n \to [0,1]^n$ in which for a vector $\mathbf{x} = (x_1, \dots, x_n)$, $F_i(\mathbf{x})$ is the outcome of the feasibility program Q for player $i \in N$. By definition, $\mathbf{x} = F(\mathbf{x})$ satisfies the constraints of Equation (4.14) for every player, and is thus an equilibrium of the digraph threshold game. Clearly, the gate inputs only appear on the left-hand side of the conditional constraints, and Q is trivially always feasible. □

4.4.3 PPAD-membership via Linear Best Response Oracles

In the applications that we presented in Sections 4.4.1 and 4.4.2, what we essentially did was construct a function $F : \times_{i \in N} D_i \to \times_{i \in N} D_i$, where D_i is the strategy space of player i, such that coordinate $F_i(\mathbf{x})$ is a best response of player i to $\mathbf{x}_{-i}$. It is then clear that any fixed point of F must be an equilibrium. This is in fact the application of the Kakutani fixed point theorem to show equilibrium existence. PPAD-membership then followed from the fact that the best responses could be computed via linear-OPT-gates.

In this section we will formulate the aforementioned principle as a general theorem, which will have Theorems 4.4.1 and 4.4.2, as well as several other theorems that we will prove later in this section and in Section 4.5, as corollaries. Below we define the very general notion of a game with linear best response oracles, when the strategy spaces D_i are convex polytopes.

Definition 4.4.4 (Game with Linear Best Response Oracles (LBRO Game))**.** A *game with linear best response oracles (LBRO)* is a game in which a best response of each player $i \in N$ is outputted by an oracle $C_i : D_1 \dots \times \dots D_{i-1} \times D_{i+1} \times \dots \times D_n \to D_i$, which is given by a linear pseudo-circuit.

An equilibrium of a LBRO game is a vector of strategies $(\mathbf{x}_1, \dots, \mathbf{x}_n)$, one for each player, such that each player is choosing a best response. As we will see in the applications later in the section, the notion of "best response" as well as the notion of "equilibrium" can vary, depending on the application at hand. The strength of our linear-OPT-gate makes the following theorem almost obvious.

Theorem 4.4.3. *Computing an equilibrium of a LBRO Game is in PPAD.*

Proof. We construct a function $F : \times_{i \in N} D_i \to \times_{i \in N} D_i$ in which for a vector $\mathbf{x} = (x_1, \dots, x_n)$, $F_i(\mathbf{x})$ is the outcome of the oracle $C_i(\mathbf{x}_{-i})$ for player $i \in N$. By definition, when $\mathbf{x} = F(\mathbf{x})$, $\mathbf{x}$ is an equilibrium. □

In Theorem 4.4.1, C_i was given by a linear-OPT-gate (which is, by definition, a linear pseudo-circuit) computing solutions of linear program $\mathcal{P}$, whereas in Theorem 4.4.2, C_i was given by a linear pseudo-circuit computing solutions of feasibility program Q. In the next subsections, we present several other applications of well-known games that we prove to be LBRO games, thus establishing their PPAD-membership.

Remark 13. We remark that even if one is not necessarily interested in PPAD-membership, our technique also provides an easy approach for establishing the rationality of equilibria, as this is also implied by Theorem 4.4.3.

Remark 14. In Definition 4.4.4, we defined the input domain of the oracle C_i to not include D_i, i.e., the domain of strategies of player i, whose best-response the oracle is calculating. This is natural, since a best-response oracle can intuitively be best understood as a device which computes the best response of a player against the chosen strategies of her opponents only. Still, in our applications in Section 4.5, it will be useful to extend the definition of the oracle to be a function $C_i : \times_j D_j \to D_i$, in order to be able to apply Theorem 4.4.3 above.

Concave Games and Generalized Equilibria

In this section we will generalize the PPAD-membership of Nash equilibria in the previous two sections to a class of more general games, and to a more general equilibrium notion. These will be special cases of *concave games* [192] and *generalized equilibria* [68] that admit equilibria in rational numbers.

Concave Games. Concave games are generalizations of strategic games in which there is a set N of n players, each of which has a strategy space $\Sigma_i \in \mathbb{R}^{m_i}$ which is compact and convex. Let $\Sigma = \Sigma_1 \times \ldots \times \Sigma_n$ be the space of strategy vectors; we will refer to an element $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ of Σ as a *strategy profile*. Each player i also has a payoff function $u_i : \Sigma \to \mathbb{R}$, which is continuous in $\mathbf{x}$ and concave in $\mathbf{x}_i$, assuming that the rest of the strategy profile $\mathbf{x}_{-i}$ is fixed.

Generalized Equilibrium. An *equilibrium* of a concave game is a strategy profile $\mathbf{x}$ in which every player chooses a payoff-maximizing element in her strategy space, i.e., $u_i(\mathbf{x}) = \max_{\mathbf{y}_i \in \Sigma_i} u_i(\mathbf{y}_i, \mathbf{x}_{-i})$ for every player $i \in N$. Debreu [68] defined the notion of a *generalized equilibrium*, in which a strategy profile $\mathbf{x}$ also restricts the choices in the strategy space of the players. In particular, letting $\Sigma_{-i} = \Sigma_1 \times \ldots \Sigma_{i-1} \times \Sigma_{i+1} \ldots \Sigma_n$, there is a correspondence $\gamma_i : \Sigma_{-i} \rightrightarrows \Sigma_i$, which specifies the set $\gamma_i(\mathbf{x}_{-i}) \subseteq \Sigma_i$ that player i is "allowed" to use. A generalized equilibrium of the game is a strategy profile $\mathbf{x}$ in which every player chooses a payoff-maximizing element in $\gamma_i(\mathbf{x}_{-i})$, i.e., $u_i(\mathbf{x}) = \max_{\mathbf{y}_i \in \gamma_i(\mathbf{x}_{-i})} u_i(\mathbf{y}_i, \mathbf{x}_{-i})$. The existence of a generalized equilibrium was established by Debreu [68] in concave games, when γ_i is upper and lower hemicontinuous, convex-valued and non-empty valued. Similarly to Section 4.4.1, given a preference profile $\mathbf{x}_{-i}$ of the other players, the payoff-maximizing (allowable) strategy for player i can be found as the optimal solution to the following convex program:

Convex Program C_{gen}

$$\begin{aligned} \text{maximize} \quad & u_i(\mathbf{y}_i, \mathbf{x}_{-i}) \\ \text{subject to} \quad & \mathbf{y}_i \in \gamma_i(\mathbf{x}_{-i}) \\ & \mathbf{y}_i \geq 0 \end{aligned}$$

Remark 15 (Concave Games and Generalized Equilibria)**.** Concave games were studied by Debreu [68] (in fact, "quasi-concave games", where the payoff functions can be quasi-concave) in the context of his generalized equilibrium result (coined a "Social Equilibrium" there, see also [65]). Interestingly, Rosen [192] also studied concave games independently, without referencing Debreu's work. Seemingly the only difference between Rosen's and Debreu's setting is that the former does not require the strategy profile space Σ to necessarily be the product space of the players' strategy spaces Σ_i, for $i \in N$.

For the standard notion of equilibrium (rather than generalized equilibrium) Debreu's theorem was concurrently and independently proven by Fan [91] and Glicksberg [124], and for this reason it is often referred to as *the Debreu-Fan-Glicksberg theorem for continuous games*. In fact, we will be referring to this theorem again throughout the paper, as it has been used in previous existence proofs in the applications that we consider.

It is not hard to see that concave games generalize n-player strategic games: the set of mixed strategies (i.e., randomizations over the finite set of pure strategies) is compact and convex and the payoff function is continuous in $\mathbf{x}$ and linear in $\mathbf{x}_i$. Thus a mixed Nash equilibrium in a strategic game is a special case of an equilibrium (and hence of a generalized equilibrium) in a concave game.

From the above discussion, it should be obvious that (generalized) equilibria of concave games are not guaranteed to be rational (indeed, the 3-player example of irrationality of Nash [177] still applies, for example). We will consider subclasses of these games (based on the structure of the strategy spaces and the form of the payoff functions) and these equilibria (based on the structure of the sets induced by the functions γ_i) for which rational equilibria always exist; in fact, our proof of PPAD membership will also provide the certificate of rationality. We remark that for general concave games (under the necessary assumptions for the OPT gate for FIXP to work), Filos-Ratsikas et al. [101] proved membership in FIXP; their theorems only apply to equilibria rather than generalized equilibria, but the extension in their setting is almost immediate.

In our setting, we need to consider special cases of convex program C_{gen} which are amenable to the use of the linear-OPT-gate. Specifically, the objective function will be such that its supergradient[10] with respect to $\mathbf{y}_i$ can be given by a linear pseudo-circuit, and the constraints will be linear inequalities with the gate inputs appearing only on the right-hand side of the constraints. We will also impose a bound on the domain of the variables $\mathbf{y}_i$. With this we have the following convex program:

Convex Program C

[10]We consider the supergradient here rather than the subgradient, since u_i is concave rather than convex, see also Remark 9.

$$\begin{aligned} \text{maximize} \quad & u_i(\mathbf{y}_i, \mathbf{x}_{-i}) \\ \text{subject to} \quad & A_i \cdot \mathbf{y}_i \leq b_i(\mathbf{x}_{-i}) \\ & \mathbf{y}_i \in [-R_i, R_i]^{m_i} \end{aligned}$$

In C, we assume that the inputs are provided as rational numbers. For the payoff function u_i, we do not require access to the function itself but rather to its supergradient with respect to $\mathbf{y}_i$, which is provided in the input as a linear pseudo-circuit. Here, the strategy spaces of the players are given by linear inequalities which can depend on the strategies of the other players, and generally lie within a bounded domain $[-R_i, R_i]^{m_i}$, for some $R_i > 0$. The following theorem is a corollary of Theorem 4.4.3.

Theorem 4.4.4. *Computing a generalized equilibrium of a concave game, in which the strategy spaces are given as in the constraints of convex program C and the supergradients of the payoff functions of the players are given by* linear *pseudo-circuits is in PPAD.*

Proof. By the discussion above, it follows directly that convex program C can be computed by a linear-OPT-gate, which can be used as the oracle C_i for each player $i \in N$ in the corresponding LBRO game. □

The class of concave games is very general, and hence captures several games of interest. We provide some examples of the type of equilibrium results that are captured by Theorem 4.4.4 below.

Bimatrix and Polymatrix Games. The theorem establishes that *generalized* Nash equilibria in bimatrix games exist, are rational, and are in PPAD, thus generalizing Theorem 4.4.1. It also establishes the same properties for a natural generalization of bimatrix games, called *polymatrix games* [141, 145]. These are n-player games in which the players' payoffs are additive over several 2-player games.

Definition 4.4.5 (Polymatrix game)**.** A *polymatrix game* consists of a set N of n players, each with a set S_i of pure strategies $S_i = \{1, \dots, m_i\}$ and a domain of mixed strategies $\Sigma_i := \{y \in \mathbb{R}^{m_i}_{\geq 0} \colon \sum_{j=1}^{m_i} y_j = 1\}$. Given a mixed strategy $\mathbf{x}_i$ for player i, the jth coordinate $x_{i,j}$ denotes the probability that player i is playing the pure strategy j. For every pair of players i and i' with $i \neq i'$, there is an associated $(m_i \times m_{i'})$-dimensional *payoff matrix* $\mathbf{A}_{i,i'} \in \mathbb{R}^{m_i \times m_{i'}}$, which determines the payoffs of player i in the bimatrix game against player i'. Given a mixed strategy profile $\mathbf{x} = (\mathbf{x}_1, \dots \mathbf{x}_n)$, the expected payoff of player i is defined as $\tilde{u}_i(\mathbf{x}) = \sum_{i' \in N, i' \neq i} (\mathbf{x}_i)^{\mathsf{T}} \cdot \mathbf{A}_{i,i'} \cdot \mathbf{x}_{i'}$.

Since the gradients of the payoff functions in polymatrix games are linear functions, Theorem 4.4.4 has the following corollary.

Corollary 2. *Computing a generalized mixed Nash equilibrium of a polymatrix game is in PPAD.*

Linear Succinct Games. We observe that the reason which allowed us to use the linear OPT-gate to prove PPAD-membership for the applications above is that we were always able to construct a linear pseudo-circuit computing the following quantity:

$$\tilde{u}_i(j, \mathbf{x}_{-i}) = \mathbb{E}_{\mathbf{s}_{-i} \sim \mathbf{x}_{-i}} u_i(j, \mathbf{s}_{-i}), \tag{4.15}$$

where $\mathbf{s}_{-i}$ denotes a vector of pure strategies (i.e., a pure strategy profile) of all players besides i. The quantity above is the expected payoff of player i when using pure strategy $j \in S_i$ against the mixed strategy $\mathbf{x}_{-i}$ of the other players. As long as we have a linear pseudo-circuit for computing $\tilde{u}_i(j, \mathbf{x}_{-i})$, Theorem 4.4.4 goes through. We will use the term linear succinct games to refer to those games.

Definition 4.4.6 (Linear Succinct Game)**.** A game is a linear succinct game if for any player i and for any pure strategy $j \in S_i$, there exists a linear pseudo-circuit computing the expected utility $\tilde{u}_i(j, \mathbf{x}_{-i})$, for any profile of mixed strategies $\mathbf{x}_{-i}$ of the other players.

We have the following corollary of Theorem 4.4.4.

Corollary 3. *Computing a generalized mixed Nash equilibrium of a linear succinct game is in PPAD.*

We draw parallels between our application and that of Daskalakis et al. [66] and Papadimitriou and Roughgarden [183]. Those works define classes of succinct games for which there is an oracle for computing the expected utility of the player. In [183], this oracle is referred to as the *polynomial expectation property* and is used to show that correlated equilibria [9] of games with this property can be computed in polynomial time. In [66], it is shown that if the oracle is given by a *bounded division free straight-line program of polynomial length*, then these games are in PPAD. Crucially, this latter result concerns *approximate equilibria*. One could view our result as a complement to those two results, one which concerns *exact* equilibria in *rational* numbers.

Extended Digraph Threshold Games. Theorem 4.4.4 also establishes the PPAD-membership of *generalized* equilibria in digraph threshold games (see Section 4.4.2). To see this, we may redefine a player's payoff in a digraph threshold game (Equation (4.14)) as:

$$u_i(x_i, \mathbf{x}_{-i}) = x_i \cdot (t - \sum_{j \in N_i} x_j), \tag{4.16}$$

where $\mathbf{x}_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$. Indeed, when $\sum_{j \in N_i} x_j > t$, the payoff is maximized when $x_i = 1$, when $\sum_{j \in N_i} x_j < t$, the payoff is maximized when $x_i = 0$, and when $\sum_{j \in N_i} x_j = t$, any choice of $x_i \in [0, 1]$ maximizes the payoff. Therefore the equilibria of the game under this utility function are the same as those under the original definition. Additionally, the gradient of the utility function in Equation (4.16) is linear and $x_i \in [0, 1]$ for every player $i \in N$. The digraph threshold game is then a concave game satisfying the conditions of the statement of Theorem 4.4.4. One may also add constraints on the allowable values of x_i chosen by player i, given the chosen strategies $\mathbf{x}_{-i}$ of the other players, as those of the constraints in convex program C. This gives us the following corollary, which generalizes Theorem 4.4.2.

Corollary 4. *Computing a generalized equilibrium of a digraph threshold game is in PPAD.*

Measuring Equation (4.16) against the capabilities of our linear-OPT-gate, one may observe that the term $\sum_{j \in N_i} x_j$ does not have to be a sum; it can in fact be any function such that the supergradient of $u_i(x_i, \mathbf{x}_{-i})$ for x_i can be given by a linear pseudo-circuit. In turn, this naturally allows us to define a more general class of digraph threshold games, which we call *extended digraph threshold games*.

Definition 4.4.7 (Extended Digraph Threshold Game)**.** An extended digraph threshold game $\mathcal{G}(V, E, t)$ is defined on a directed graph $G = (V, E)$, and $t \in (0, 1)$ is a threshold. The nodes of G correspond to players, and each player $i \in V$ chooses a strategy $x_i \in [0, 1]$. Let $\mathbf{x} = (x_1, \dots, x_n)$ be the vector of (pure) strategies that we will refer to as a (pure) strategy profile. A strategy profile $\mathbf{x}$ is an equilibrium if for some function g it satisfies

$$x_i = \begin{cases} 0 & g(\mathbf{x}_{-i}) > t \\ 1 & g(\mathbf{x}_{-i}) < t \\ [0,1] & g(\mathbf{x}_{-i}) = t \end{cases} \tag{4.17}$$

where in the above expression we have abused notation using $x_i = [0, 1]$ to denote $x_i \in [0, 1]$, and where $\mathbf{x}_{-i} = (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$.

To keep the definition close to that of the original digraph threshold games, we may assume that $g_i(\cdot)$ is a function only of the strategies of players in N_i, although this is not necessary. In the alternative payoff representation, the corresponding payoff function is

$$u_i(x_i, \mathbf{x}_{-i}) = x_i \cdot (t - g(\mathbf{x}_{-i})). \tag{4.18}$$

If $g(\mathbf{x}_{-i})$ can be given by a linear pseudo-circuit, then Theorem 4.4.4 applies.

Theorem 4.4.5. *Computing a generalized equilibrium of an extended digraph threshold game is in PPAD, as long as the function $g(\mathbf{x}_{-i})$ is given by a linear pseudo-circuit.*

Some natural extended digraph threshold games for which Theorem 4.4.5 applies can be defined as follows:

- digraph threshold games (Definition 4.4.3), for which $g(\mathbf{x}_{-i}) = \sum_{j \in N_i} x_j$.
- *max digraph threshold games*, for which $g(\mathbf{x}_{-i}) = \max_{j \in N_i} x_j$. In these games, player i chooses $x_i = 0$ if the maximum of the strategies of her incoming neighbors is above the threshold, $x_i = 1$ if it is below, and any value in $[0, 1]$ otherwise.

- *min digraph threshold games*, for which $g(\mathbf{x}_{-i}) = \min_{j \in N_i} x_j$. In these games, player i chooses $x_i = 0$ if the minimum of the strategies of her incoming neighbors is above the threshold, $x_i = 1$ if it is below, and any value in $[0,1]$ otherwise.

Bilinear Games. Theorem 4.4.4 also establishes the rationality and PPAD membership of Nash equilibria in *bilinear games*, introduced by Garg et al. [115]. Intuitively, these generalize bimatrix games in the sense that the strategy space does not have to be the unit simplex $\Sigma_i = \{y \in \mathbb{R}^{m_i}_{\geq 0} : \sum_{j=1}^{m_i} y_j = 1\}$, as in Definition 4.4.1, but it can be any compact polytope.

Definition 4.4.8 (Bilinear Games [115])**.** A *bilinear game* is a two-player game represented by two $m_1 \times m_2$ payoff matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, one for each player, and two compact polytopal strategy sets X_1 and X_2. Let $\mathbf{A}_1 \in \mathbb{R}^{k_1 \times m_1}$ and $\mathbf{A}_2 \in \mathbb{R}^{m_2 \times k_2}$ and let $\mathbf{e}_1 \in \mathbb{R}^{k_1}$ and $\mathbf{e}_2 \in \mathbb{R}^{k_2}$ be two vectors. The strategy spaces of players 1 and 2 respectively are defined as

$$X_1 = \{\mathbf{x} \in \mathbb{R}^{m_1} : \mathbf{A}_1 \cdot \mathbf{x} = \mathbf{e}_1, \mathbf{x} \geq 0\}, \text{ and } X_2 = \{\mathbf{x} \in \mathbb{R}^{m_2} : \mathbf{A}_2 \cdot \mathbf{x} = \mathbf{e}_2, \mathbf{x} \geq 0\}.$$

Given a pair of strategies $(\mathbf{x}_1, \mathbf{x}_2) \in X_1 \times X_2$, the payoff of player 1 is $\mathbf{x}_1^\mathsf{T} \cdot \mathbf{A}_1 \cdot \mathbf{x}_2$ and the payoff of player 2 is $\mathbf{x}_1^\mathsf{T} \cdot \mathbf{A}_2 \cdot \mathbf{x}_2$.

A Nash equilibrium of the game is defined analogously to Definition 4.4.2. From Definition 4.4.8, it follows that bilinear games are concave games in which the gradients of the payoffs of the players are linear functions. Thus Theorem 4.4.4 has the following corollary.

Corollary 5. *Computing a generalized Nash equilibrium of a bilinear game is in PPAD.*

We remark that the PPAD-membership of finding equilibria in bilinear games had not explicitly been proven before our work, but it can be recovered rather implicitly via observing that these games are very much related to the *sequence form* of Koller et al. [157], for which the authors devise an LCP.

The Debreu-Fan-Glicksberg theorem [1952] and Rosen's theorem [1965]. As we mentioned earlier, the Debreu-Fan-Glicksberg theorem for continuous games (or equivalently, Rosen's theorem) is often used in the literature to prove existence of equilibria for various games. It is in a sense stronger than Brouwer's fixed point theorem [39], as it can be used to prove the existence of equilibria in more general games. Our PPAD-membership of this section also provides a tool for proving PPAD-membership of these other problems that apply to the Debreu-Fan-Glicksberg theorem, as long as the games that they consider satisfy the conditions required for Theorem 4.4.4. Generally speaking, if the problem in question can be reduced to that of finding equilibria of a concave game in which the supergradients of the payoff functions can be computed by linear arithmetic circuits, then Theorem 4.4.4 implies its membership in PPAD.

The existence proofs of several of our applications in this section and in Section 4.5 are in fact established via the Debreu-Fan-Glicksberg theorem. One could transform those proofs into computational reductions (and then Theorem 4.4.4 would apply immediately); instead, we obtain the PPAD-membership of those problems directly from the more general Theorem 4.4.3.

Personalized Equilibria

Our next application of Theorem 4.4.3 is to finding *personalized equilibria* in graphical games, a notion introduced by Kintali et al. [153]. Intuitively speaking, these equilibria allow players to "match" their strategies with those of their opponents, without obeying a product distribution.

Definition 4.4.9 (Hypergraph Game [153])**.** Consider a game $\mathcal{G}$ played on a hypergraph $G = (V, E)$ among n different players of a set N. Each player $i \in N$ has a finite strategy set S_i, and $|S_i \cap S_J| = \emptyset$ for all $i, j \in N$ such that $i \neq j$. Let $V = \bigcup_{i \in N} S_i$. For each player $i \in N$, we have a set $E_i \subseteq E$ of hyperedges, which satisfy the following two conditions:

- if $e \in S_i$ then $|e \cap S_j| \leq 1$ for any $j \neq i$,
- if $e, e' \in S_i$, then $e \not\subset e'$.

Lastly, each player $i \in N$ has a utility function $u_i : E_i \to \mathbb{R}$. A *mixed strategy* $\mathbf{x}_i$ for player $i \in N$ is a probability distribution over S_i, and a *weight assignment* $\mathbf{w}_i$ is a probability distribution over E_i.

One way to interpret Definition 4.4.9 above in relation to the "standard" games that we have seen so far, is that the hyperedges e correspond to pure strategy profiles, interpreted as sets of pure strategies, at most one for each player, and each player can choose how much weight $w_i(e)$ to assign to each such profile. A personalized equilibrium of $\mathcal{G}$ is defined as follows:

Definition 4.4.10 (Personalized Equilibrium [153])**.** A personalized equilibrium of $\mathcal{G}$ consists of a vector of mixed strategies $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and a vector of weight assignments $\{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$, such that for every player $i \in N$, $(\mathbf{x}_i, \mathbf{w}_i)$ is a solution to the following linear program (with variables $\mathbf{x}_i$ and $\mathbf{w}_i$):

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E_i} w_i(e) u_i(e) \\ \text{subject to} \quad & \sum_{e: s \in e} w_i(e) \leq x_j(s) \quad \forall s \in S_j, \forall j \neq i \\ & \sum_{e: s \in e} w_i(e) = x_i(s) \quad \forall s \in S_i \\ & w_i(e) \geq 0 \qquad \forall e \in E_i \end{aligned} \tag{4.19}$$

Features of our proof and Kintali et al.'s PPAD-membership result. To establish existence of a personalized equilibrium in every hypergraph game $\mathcal{G}$ as defined above, Kintali et al. [153] essentially reduce the game to a concave game, and then invoke the Debreu-Fan-Glicksberg theorem (1952). This already hints at the fact that we could obtain PPAD-membership of the problem as a corollary of Theorem 4.4.4. We will instead obtain it as corollary of the more general Theorem 4.4.3 rather straightforwardly; we do that in Theorem 4.4.6 below. Additionally, to obtain PPAD-membership (and as a result, rationality of equilibria), Kintali et al. [153] first define an approximate version of the problem (the ε-personalized equilibrium), and reduce that problem to End-Of-Line (see Definition 4.2.1), via a relatively involved construction. To obtain PPAD-membership for the exact problem (i.e., when $\varepsilon = 0$) Kintali et al. [153] construct an elaborate argument that appeals to linear programming compactness, by first showing that for sufficiently small ε, ε-personalized equilibria "almost satisfy" the constraints of the linear programs, and then carefully rounding the solution to obtain an exact equilibrium. Our technique allows us to reduce this whole argument to a few lines.

Theorem 4.4.6. *Computing a personalized equilibrium of a well-behaved hypergraph game (Definition 4.4.9) is in PPAD.*

Proof. We observe that the linear program (4.19) computes the best response of each player $i \in N$ in the game, when the player's strategy is given by the variables $x_i(s)$ and the other players' strategies are given by the parameters $x_j(s)$. The LBRO game oracle C_i for player $i \in N$ will be given by a linear-OPT-gate which inputs the vector $\mathbf{x}_{-i}$ of strategies of the other players and outputs the strategy $\mathbf{x}_i$ of player i. By Theorem 4.4.3, the theorem follows. □

ε-proper Equilibria in Linear Succinct Games

In this section we will consider an alternative equilibrium notion, that of ε-proper equilibrium [174], which refines another well-known notion, that of ε-perfect equilibrium [199]. Both of these notions allow the players to make small mistakes ("trembles") when choosing their optimal mixed strategies, but ensure that these mistakes happen with small probability (related to the parameter ε). Recall the definition of linear succinct games (Definition 4.4.6); we will show that finding ε-proper equilibria of those games is in PPAD.

First, to demonstrate the main ideas, we will consider bimatrix games, which were studied in the past in the context of TFNP and ε-proper equilibria by [203]. Recall that bimatrix games can be expressed by two $m_1 \times m_2$ matrices, which we will henceforth denote by $\mathbf{A}_1$ and $\mathbf{A}_2$. We provide the definition of an ε-proper equilibrium below.

Definition 4.4.11 (ε-proper equilibrium in a bimatrix game [174])**.** Let $\varepsilon > 0$. A pair of mixed strategies $(\mathbf{x}_1, \mathbf{x}_2)$ is an ε-proper equilibrium of a bimatrix game if $\mathbf{x}_1$ and $\mathbf{x}_2$ are *fully mixed* (i.e., $x_{i,j} > 0$ for any pure strategy $j \in S_i$, for $i \in \{1,2\}$), and

$$(\mathbf{A}_1 \cdot \mathbf{x}_2)_j < (\mathbf{A}_1 \cdot \mathbf{x}_2)_{j'} \Rightarrow x_{1,j} \leq \varepsilon \cdot x_{1,j'} \text{ for all } j, j' \in S_1,$$

$$(\mathbf{x}_1^{\mathsf{T}} \cdot \mathbf{A}_2)_j < (\mathbf{x}_1^{\mathsf{T}} \cdot \mathbf{A}_2)_{j'} \Rightarrow x_{2,j} \le \varepsilon \cdot x_{2,j'} \text{ for all } j, j' \in S_2.$$

We remark that the *ε-perfect equilibrium* mentioned earlier is defined analogously, with the only difference being that ε is not multiplied by the mixed strategy $x_{i,j'}$ on the right-hand side of the constraints.

We will obtain that computing ε-proper equilibria of bimatrix games is in PPAD, essentially as a corollary of Theorem 4.4.3. Before we present the theorem and the proof, we discuss the idea and its advantages over the previous PPAD-membership proof due to Sørensen [203].

Features of our proof and the previous PPAD-membership result. Sørensen [203] first provided a PPAD-membership result for computing ε-proper equilibria. His proof proceeds by showing that an ε-proper equilibrium can be recovered as a solution to an LCP, and thus can be found by Lemke's algorithm [160]. As we highlighted in Section 4.1.2, this approach already introduces complications, mainly arguing against ray termination, which is also explicitly done in [203]. Besides that, to make sure that the constructed LCP has polynomial size, Sørensen employs an *extended formulation of the generalized permutahedron* due to Goemans [125].

Our proof is conceptually much simpler: it suffices to embed the conditions of Definition 4.4.11 in a set of feasibility programs Q (see Program (4.3) in Section 4.3), one for each player $i \in \{1,2\}$, see Figure 4.4. Each of those feasibility programs will compute the best response of the corresponding player, where a "best response" here is a mixed strategy $\mathbf{x}_i$ that satisfies the conditions of Definition 4.4.11, i.e., when a pure strategy yields smaller expected payoff, then it is played with probability which is smaller by an ε multiplicative factor. Using these feasibility programs as the oracles C_1 and C_2 of an LBRO game, we obtain the proof of Theorem 4.4.7 as a corollary of Theorem 4.4.3. We state the theorem next.

Theorem 4.4.7. *Computing an ε-proper equilibrium of a bimatrix game is in PPAD.*

Proof. Consider the feasibility programs Q_1 and Q_2 of Figure 4.4, for players 1 and 2 respectively. By Definition 4.4.11, a solution to Q_i, for $i \in \{1,2\}$ is a best response of player i in the bimatrix game. Note that the antecedents (namely, the constraints on the left-hand side of the implications) only contain gate inputs. Hence, these can be computed by linear-OPT-gates, so they can be used as oracles C_1 and C_2 in the corresponding LBRO game (Definition 4.4.4) as long as they are solvable. Then, the theorem follows from Theorem 4.4.3. Solvability of Q_i for $i \in \{1,2\}$ is easy to see, by observing that it is of the form Q_{app}, presented in Section 4.3.2. By Lemma 12, it suffices to argue that the feasibility graph is acyclic. The feasibility graph G_{Q_i} consists of vertices corresponding to the different pure strategies of player i, and an edge (j, j') is only present if the expected utility of player i from j is strictly lower than the expected utility for j'. It is straightforward to verify that G_{Q_i} is acyclic. □

Feasibility Program Q_1	Feasibility Program Q_2
$(\mathbf{A}_1 \cdot \mathbf{x}_2)_j < (\mathbf{A}_1 \cdot \mathbf{x}_2)_{j'} \Rightarrow x_{1,j} \le \varepsilon \cdot x_{1,j'}$ for all $j, j' \in S_1$	$(\mathbf{x}_1^{\mathsf{T}} \cdot \mathbf{A}_2)_j < (\mathbf{x}_1^{\mathsf{T}} \cdot \mathbf{A}_2)_{j'} \Rightarrow x_{2,j} \le \varepsilon \cdot x_{2,j'}$ for all $j, j' \in S_2$
$x_{1,j} \ge \dfrac{\varepsilon^{m_1}}{m_1}$, for all $j \in S_1$, $\displaystyle\sum_{j=1}^{m_1} x_{1,j} = 1$	$x_{2,j} \ge \dfrac{\varepsilon^{m_2}}{m_2}$, for all $j \in S_2$, $\displaystyle\sum_{j=1}^{m_2} x_{2,j} = 1$

Figure 4.4: The feasibility programs Q_1 and Q_2 used in the proof of Theorem 4.4.7.

Theorem 4.4.7 establishes the PPAD-membership of computing ε-proper equilibria in bimatrix games. In the following we will show that the proof can straightforwardly be extended to larger classes of games. We first offer the following remark.

Remark 16 (Proper and Trembling hand perfect equilibria)**.** Myerson [174] defined a *proper equilibrium* to be a limit point of ε-proper equilibria for $\varepsilon \to^+ 0$. Similarly, limits points of ε-perfect equilibria are trembling hand perfect equilibria [199]. With this definition, proper equilibria are *refinements* of trembling hand perfect equilibria, which, in turn, are refinements of Nash equilibria. Intuitively, a refinement is a special class of equilibria characterized by a set of principles that make them more "plausible". Note that ε-proper equilibria are not refinements of (exact) Nash equilibria, and this is why we refer to them as "alternative equilibrium notions". We believe they are still natural, as they can be interpreted as a model of *limited rationality*[11], with the players being "imperfectly rational" in their decisions.

In this context, [203] showed a stronger result, namely that finding *symbolic* ε-proper equilibria is in PPAD. Interestingly, the problem of finding a proper equilibrium is unlikely to be in TFNP [137], and hence Sørensen's result does not provide a PPAD-membership result for all proper equilibria. We refer the reader to [203] for the appropriate definitions and the details.

Linear succinct games. We now explain how Theorem 4.4.7 extends rather straightforwardly to linear succinct games (Definition 4.4.6). The definition of ε-proper equilibria in such games generalizes that of Definition 4.4.11 straightforwardly. Recall the definition of the expected utility of player i when playing the pure strategy j against the mixed strategy profile $\mathbf{x}_{-i}$ of the other players (Equation (4.15)).

Definition 4.4.12 (ε-proper equilibrium (general games) [174])**.** Let $\varepsilon > 0$. A mixed strategy profile $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is an ε-proper equilibrium of a strategic game if for any player $i \in N$, we have that $\mathbf{x}_i$ is *fully mixed* (i.e., $x_{i,j} > 0$ for any pure strategy $j \in S_i$), and for any player $i \in N$

$$\tilde{u}_i(j, \mathbf{x}_{-i}) < \tilde{u}_i(j', \mathbf{x}_{-i}) \Rightarrow x_{i,j} \le \varepsilon \cdot x_{i,j'} \text{ for all } j, j' \in S_i. \tag{4.20}$$

[11]Selten [199] in fact introduces the notion of trembling hand perfect equilibria using a similar narrative.

To extend the feasibility programs Q_i, for $i \in N$, to general games, one only has to substitute the corresponding constraints for each player $i \in N$ with those in Definition 4.4.12 above. In linear succinct games, the quantities $\tilde{u}_i(j, \mathbf{x}_{-i})$ for any player i and pure strategy j can be computed by a linear pseudo-circuit. The proof of the following theorem is then very similar to that of Theorem 4.4.7.

Theorem 4.4.8. *Computing an ε-proper equilibrium of a* linear *succinct game is in PPAD.*

Proof. Again, feasibility program Q_i, for $i \in N$, captures the best response of player $i \in N$. Each Q_i can be computed by a linear-OPT-gate, and is solvable by the same very simple argument used in the proof of Theorem 4.4.7. Thus Q_i can be used as the oracle C_i in the corresponding LBRO game (Definition 4.4.4), and the theorem follows from Theorem 4.4.3. □

Since polymatrix games (Definition 4.4.5) are linear succinct games, we obtain the following corollary.

Corollary 6. *Computing an ε-proper equilibrium of a polymatrix game is in PPAD.*

A proof of Corollary 6 was first provided by Hansen and Lund [137]. Their proof essentially redoes all of the steps of the proof of Sørensen [203] for bimatrix games, extending them to the more general case, stating and proving corresponding lemmas etc. Clearly, our linear-OPT-gate allows us to avoid all this labor and extend the PPAD-membership from bimatrix games to polymatrix games, and even beyond, rather straightforwardly. We conclude our discussion on ε-proper equilibria with the following remark.

Remark 17 (ε-proper equilibria can be irrational)**.** For general multiplayer games (not necessarily linear succinct), Filos-Ratsikas et al. [101] showed a FIXP-membership result via a system of conditional convex constraints, which generalizes the feasibility program Q_i. FIXP is the right class for these games, because it has been shown (see Footnote 1 in [89]) that there are games in which all ε-perfect equilibria (and hence all ε-proper equilibria) are irrational.

4.5 Congestion Games

In this section we consider models of congestion games [193, 226], where players compete for resources. More precisely, we focus our study on *non-atomic* and *atomic splittable* congestion games, and for both we consider in particular the important subclasses of network congestion games. In particular, via the employment of our linear-OPT-gate we will obtain PPAD-membership results for three domains of congestion games, namely for

- finding equilibria in congestion games with rational and malicious players, studied by [13], see Section 4.5.1,

- finding Wardrop equilibria in multi-class non-atomic network congestion games, studied by [172], see Section 4.5.2,
- finding Nash equilibria multi-class atomic splittable network congestion games, studied by [155], see Section 4.5.2.

All of our PPAD-membership results hold when the latency functions (an in some cases possibly their subgradients) are given by linear pseudo-circuits (recall Definition 4.2.7) or simply, by linear arithmetic circuits. In particular, they apply for example to all *piecewise-linear* latency functions that are given explicitly as part of the input. Via the machinery that we develop in Section 4.3.4, we can in fact obtain the PPAD-membership for the more general case where the latency functions and their subgradients are provided in the input *implicitly*, via linear pseudo-circuits. As such, our results are simultaneously *significiant simplifications* over the existing results in the literature, as well as *generalizations*, since the only known PPAD-membership results so far were for *linear* latency functions.[12] In fact, for games with malicious players, complexity results had not been proven before our work.

We remark that all the games mentioned above fall into one of two main categories, either *non-atomic congestion games* or *atomic splittable congestion games*. As such, finding their equilibria does not simply fall in the class PLS [148], which is known to capture the complexity of atomic (non-splittable) congestion games. In fact, finding equilibria for some of the games studied in this section has been proven to be PPAD-complete [155].

Before we proceed with the technical parts of the section, we present the previous works on these domains and compare the previous approaches to our proofs which use the linear-OPT-gate.

Congestion games with malicious players. The study of congestion games with malicious players was initiated by Karakostas and Viglas [150]. Here, we consider the model studied by Babaioff et al. [13], in which there is a continuum of rational players, who are trying to minimize their total load, and a single malicious player, who is trying to maximize the load experienced by the rational players. Babaioff et al. proved the existence of equilibria in such games with non-decreasing latency functions. They also prove that when the latency functions are concave, these games have pure equilibria, by appealing to the Debreu-Fan-Glicksberg theorem [1952]. This already hints that PPAD-membership can be recovered as a corollary of Theorem 4.4.4; we will instead obtain it as a corollary of the more general Theorem 4.4.3. We remark that for these games, a PPAD-membership result was not previously known.[13]

[12]In the context of congestion games, the term "affine" is often used instead of "linear". Boyd and Vandenberghe [37] define a linear function as of the form $f(x) = c^\mathsf{T} x$, and an affine function as the sum of a linear function and a constant. We use the term "linear" to refer to all affine functions.

[13]Babaioff et al. [13] mention in the conclusion of their work that their existence proofs establish membership of the problem in PPAD. This claim is however not straightforward, and there is no proof to support it. In any case, such a claim would most certainly refer only to *approximate* equilibria, as these games do not always have rational equilibria for any concave latency function.

Multi-class non-atomic network congestion games. In multi-class non-atomic network congestion games, there is a continuum of players divided into different classes. Existence of Wardrop equilibria in this setting was first established by Schmeidler [198], via an application of the Debreu-Fan-Glicksberg theorem [1952]. A different proof of existence was provided by Milchtaich [173], who also studied the equilibrium uniqueness. The first computational complexity results on the problem were provided by Meunier and Pradeau [172], who proved that the problem of finding a Wardrop equilibrium lies in PPAD, when the latency functions on the resources of the game are linear. Their proof goes via the "LCP approach" (see Section 4.1.2). It turns out that their LCP formulation cannot be solved by the vanilla version of Lemke's algorithm, and so they devise a similar pivoting algorithm, tailored to their problem. As in the case of Lemke's algorithm, they argue explicitly against ray termination.

Multi-class atomic splittable network congestion games. The existence of Nash equilibria in atomic splittable congestion games follows from the Debreu-Fan-Glicksberg theorem [1952], see [155]. The computational complexity of the problem with player-specific linear latency functions was studied by Klimm and Warode [155], who proved its PPAD-completeness. What is important for us is the most challenging part of that result, which is the PPAD-membership. Klimm and Warode's proof is rather involved, and goes via the development of a homotopy method for tracing an equilibrium given the demand rates of the players. This gives rise to a new pivoting algorithm, similar in spirit to Lemke's algorithm, or more precisely, the Lemke-Howson algorithm (see Section 4.1.2). The method solves the problem of finding a Nash equilibria as a system of linear equations involving *excess flows*, *vertex potentials* and *block Laplacians*. At a very high level, the authors use the excess and potentials to define an undirected version of the End-of-Line graph (see Definition 4.2.1), and the determinant of the block Laplacians to define a unique orientiation of the edges, effectively reducing the problem to End-of-Line. It is interesting to mention that Klimm and Warode do mention that the Nash equilibrium problem can be formulated as an LCP, but it is unclear whether Lemke's algorithm can solve it, motivating the development of their new algorithm.

Our results and proofs. The linear-OPT-gate allows us to avoid any of the technical complications of the proofs of Meunier and Pradeau [172] and [155] (which are rather involved, especially the latter), and essentially obtain the PPAD-membership for all of these problems as simple corollaries of Theorem 4.4.3 (or even Theorem 4.4.4, as the games in this section are linear concave). In fact, as we mentioned earlier, we obtain generalizations of those results, from linear latency functions to any latency function that is given (in some cases together with its subgradient) by a linear arithmetic circuit. In particular, they capture all piecewise-linear latency functions that are given explicitly as part of the input, as well as those where the latencies and their subgradients are given implicitly by the aforementioned circuits (see Section 4.3.4).

For congestion games with malicious players, our PPAD-membership result for this same class of latency functions is the first such complexity result for any version of the problem.

4.5.1 Multi-class Congestion Games

In the models we consider, the players are divided into a finite number k of *classes*, and we refer to these as *multi-class congestion games*. The main difference between the different models we consider arise from how the classes of players are interpreted. Apart from this, the models have many commonalities, and we start by describing these.

Remark 18 (Notation)**.** In this section we shall for convenience make use of the (common) shorthand notation of writing a function argument as a subscript. That is, we may write f_a in place of $f(a)$, for a given function $f\colon A \to B$.

Definition 4.5.1 (Multi-class Congestion Games)**.** A multi-class congestion game with a finite number of k classes is given by a finite set E of *resources* and, for each class $i \in [k]$, a set of *pure strategies* $\Sigma_i \subseteq 2^E$, consisting of subsets of resources. Each resource is equipped with class-dependent continuous *latency functions*, denoted by $\ell^i_e\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ for $e \in E$ and $i \in [k]$. Note that we do not assume that the latency functions are non-decreasing. A *load* on resources is a function $x\colon E \to \mathbb{R}_{\geq 0}$. The latency functions extend additively to latency functions for pure strategies that are functions of loads by letting $\ell^i_S(x) = \sum_{e\in S} \ell^i_e(x_e)$.

We associate to each class $i \in [k]$ a positive *weight demand* d_i and let $d = d_1 + \cdots + d_k$ be the total weight demand. A *load allocation* for class $i \in [k]$ is a function $f^i\colon \Sigma_i \to \mathbb{R}_{\geq 0}$ such that $\sum_{S\in\Sigma_i} f^i_S = d_i$. The load allocation f^i induces a *load* $x^i\colon E \to \mathbb{R}_{\geq 0}$ for class $i \in [k]$ and a resulting total load $\overline{x}\colon E \to \mathbb{R}_{\geq 0}$ given as follows.

$$\begin{aligned} x^i_e &= \sum_{\substack{S\in\Sigma_i\\ e\in S}} f^i_S \quad , \quad e \in E, i = 1,\dots,k \ . \\ \overline{x}_e &= \sum_{i=1}^{k} x^i_e \quad , \quad e \in E \ . \end{aligned} \tag{4.21}$$

Non-atomic congestion games

In the *non-atomic* model, each class of players represents a continuum of players. Formally we consider the set of all players to be given by a bounded real interval I of length d, partitioned into k measurable sets $I_1,\dots,I_k$ with respect to the Lebesgue measure μ, such that $\mu(I_i) = d_i$, for $i \in [k]$, thereby defining the k classes of players. A strategy profile is a measurable function $\rho : I \to 2^E$ such that $\rho(I_i) \subseteq \Sigma_i$ for every $i \in [k]$. The restriction $\rho_i = \rho_{\restriction I_i}$ of ρ to the subset I_i yields the strategy profile $\rho_i\colon I_i \to \Sigma_i$ of the players of class i. The strategy profile induces a load allocation profile $f = (f^1,\dots,f^k)$ by letting $f^i_S = \mu(\rho^{-1}(S))$ for $S \in \Sigma_i$ and $i \in [k]$, which in turn defines a load profile $x = (x^1,\dots,x^k)$ and the total load x on resources.

The central equilibrium notion for non-atomic congestion games is the *Wardrop equilibrium* [226], here defined for multi-class congestion games. The notion is very similar to the Nash equilibrium that we used in the previous sections. In fact, the Wardrop equilibrium coincides with the notion of Nash equilibrium for this formulation in terms of a continuity of players. We will use the term Wardrop equilibrium to refer to equilibria in non-atomic games, and the term *Nash equilibrium* or simply *equilibrium* to refer to atomic splittable games, see Section 4.5.1.

Definition 4.5.2 (Wardrop equilibrium)**.** A strategy profile ρ is a *Wardrop equilibrium* if for all $i \in [k]$ it holds that $\rho(I_i) \subseteq \arg\min_{S \in \Sigma_i} \ell^i_S(\overline{x})$, where $\overline{x}$ is the total load induced by ρ.

We shall also say that the load allocation profile induced by a Wardrop equilibrium ρ *itself* is a Wardrop equilibrium. Note that a different strategy profile ρ' may induce the same load allocation as profile ρ without being a Wardrop equilibrium. In that case, however, there would exist another Wardrop equilibrium ρ'' such that ρ' and ρ'' differ only on a null set with respect to μ.

Definition 4.5.3 (Wardrop equilibrium — load allocation formulation)**.** A load allocation profile $f = (f^1, \dots, f^k)$ is a Wardrop equilibrium if $\ell^i_S(\overline{x}) = \min_{S' \in \Sigma_i} \ell^i_{S'}(\overline{x})$, whenever $f^i_S > 0$, for all i, where $\overline{x}$ is the total load defined by $(f^1, \dots, f^k)$.

Before presenting a fixed point formulation of Wardrop equilibria, we introduce some further notation, which will be used again in the setting of atomic splittable congestion games in Section 4.5.1 below.

Given a load allocation $f = (f^1, \dots, f^k)$ we let $f^{-i} = (f^1, \dots, f^{i-1}, f^{i+1}, f^k)$ denote the load allocation profile for classes different from i. For $i \in [k]$ and $S \in \Sigma_i$, we define the function L^i_S as follows. For a load allocation profile f and a load allocation g^i for class i, we let

$$L^i_S(g^i, f^{-i}) = \sum_{e \in S} \ell^i_e \left(\sum_{\substack{T \in \Sigma_i \\ e \in T}} g^i_T + \sum_{j \neq i} \sum_{\substack{T \in \Sigma_j \\ e \in T}} f^j_T \right) \tag{4.22}$$

Note that if $x^{-i} = (x^1, \dots, x^{i-1}, x^{i+1}, x^k)$ are the induced loads by f^{-i}, and y^i is the induced load by g, then $L^i_S(g^i, f^{-i}) = \sum_{e \in S} \ell^i_e(y^i_e + \sum_{j \neq i} x^j_e)$. In particular, if $\overline{x}$ is the total load induced by f, then $L^i_S(f) = \ell^i_S(\overline{x})$.

Let $i \in [k]$ and consider the following linear program in variables g^i_S.

<u>Linear Program $\mathcal{P}_{\mathrm{WE}}$</u>

$$\begin{aligned} \text{minimize} \quad & \sum_{S \in \Sigma_i} L^i_S(f) g^i_S \\ \text{subject to} \quad & \sum_{S \in \Sigma_i} g^i_S = d_i \\ & g^i_S \geq 0, \quad \text{for all } S \in \Sigma_i \end{aligned}$$

An optimal solution $g^i = (g^i_S)_{S \in \Sigma_i}$ must satisfy that $L^i_S(f) = \min_{S' \in \Sigma_i} L^i_{S'}(f)$ whenever $g^i_S > 0$. This gives rise to a fixed point characterization of Wardrop equilibrium.

Proposition 6. *A load allocation profile $f = (f^1, \dots, f^k)$ is a Wardrop equilibrium if and only if f^i is an optimal solution to the linear program $\mathcal{P}_{\mathrm{WE}}$ for all $i \in [k]$.*

Looking ahead, the characterization of Proposition 6 via linear program $\mathcal{P}_{\mathrm{WE}}$ will be employed to prove PPAD-membership. In particular, each $\mathcal{P}_{\mathrm{WE}}$ will capture the best response of each player, and a linear-OPT-gate will serve as the oracle for computing this best response. This effectively will render the game a LBRO game (Definition 4.4.4), and PPAD-membership will follow from Theorem 4.4.3. We present a concrete application of this in Section 4.5.1 below. First, we define the setting of atomic splittable congestion games.

Atomic splittable congestion games

In the atomic splittable model each class is representing a single player controlling the entire weight demand of the class. We shall thus refer to class i simply as *player i*, and the strategy space of player i is the set of load allocations of class i. Given a load allocation profile f, player i experiences latency $\sum_{S \in \Sigma_i} L^i_S(f) f^i_S$ which becomes the cost function C^i the player wishes to minimize. We thus define $C^i(g^i, f^{-i}) = \sum_{S \in \Sigma_i} L^i_S(g^i, f^{-i}) g^i_S$.

The central equilibrium notion for atomic splittable congestion games we consider is the *Nash equilibrium* with respect to these cost functions.

Definition 4.5.4 (Nash Equilibrium — load allocation formulation)**.** A load allocation profile $f = (f^1, \dots, f^k)$ is a *Nash equilibrium* if and only if $C^i(f) \le C^i(g^i, f^{-i})$ for all $i \in [k]$.

If one imposes an additional convexity assumption involving the latency functions, then the game essentially becomes a concave game, and thus possesses a Nash equilibrium by the Debreu-Fan-Glicksberg theorem [1952]. The following lemma follows immediately from definitions.

Lemma 18 (Convexity assumption for latency functions)**.** *Suppose that for all $i \in [k]$ and $e \in E$ the latency function ℓ^i_e satisfies that the function $x \mapsto \ell^i_e(x + c)x$ is convex for all $c \ge 0$. Then, for all $i \in [k]$ and all load allocation profiles f the cost function $g^i \mapsto C^i(g^i, f^{-i})$ is convex.*

Remark 19. In the literature on congestion games, a common assumption of a latency function ℓ is that the function $x \mapsto \ell(x)x$ is convex. This is in general not sufficient to guarantee that the function $x \mapsto \ell(x + c)x$ for $c \ge 0$ is convex, as required in the model we consider. It is however sufficient if one additionally assumes that ℓ is non-decreasing.

With the assumptions of Lemma 18 we consider the following convex program in variables g^i_S.

Convex Program C_{NE}

$$\begin{aligned}
\text{minimize} \quad & \sum_{S\in\Sigma_i} L^i_S(g^i, f^{-i}) g^i_S \\
\text{subject to} \quad & \sum_{S\in\Sigma_i} g^i_S = d_i \\
& g^i_S \geq 0, \quad \text{for all } S \in \Sigma_i
\end{aligned}$$

Directly from definitions we then have the following.

Proposition 7. *In a congestion game with latency functions satisfying the assumptions of Lemma 18, a load allocation profile $f = (f^1, \ldots, f^k)$ is a Nash equilibrium if and only if f^i is an optimal solution to the convex program C_{NE} for all $i \in [k]$.*

Again, convex program C_{NE} essentially calculates the best response for each player, and hence will be computed by an oracle given by a linear-OPT-gate. We will apply this concretely in Section 4.5.2 in the context of network congestion games, but after we will have reformulated C_{NE} via a more concise description of the strategies in terms of flows on resources rather than load allocations.

PPAD-membership of congestion games with malicious players

A model of non-atomic congestion games with a *malicious* player was introduced by [13]. In this model there are two classes of players, the *rational* players and the *malicious* player. We shall use the index set $\{R, M\}$ for these classes rather than $\{1, 2\}$. The class of rational players represents a continuum of players. As in the basic case of non-atomic congestion games as given in Section 4.5.1, these are given by a bounded real interval I of length d_R and a strategy of the rational players is a measureable function $\rho_R \colon I \to \Sigma_R$. The strategy ρ induces a load allocation f^R, and we will use that to re-express the equilibrium condition in terms of f^R.

The class of the malicious player represents a single player that controls a weight demand d_M, as in the case of atomic splittable games as given in Section 4.5.1. However, unlike the basic model of atomic splittable congestion games, the malicious player can use a *mixed* strategy, i.e., a probability distribution ρ_M over load allocations f^M. Since the malicious player is using a mixed strategy, what the rational players aim to minimize is the expected load of their chosen strategy. The malicious player on the other hand aims to maximize the expected total load experienced by the rational players. We define an equilibrium of this game explicitly below.

Definition 4.5.5. The strategy profile (ρ_R, ρ_M) is an *equilibrium* if the following conditions hold, where x^R is the load induced by ρ^R and x^M is the load induced by f^M.

1. $\rho_R \subseteq \arg\min_{S\in\Sigma_R} \mathbb{E}_{f^M \sim \rho_M}[\ell^R_S(x^R + x^M)]$.
2. $\text{supp}(\rho_M) \subseteq \arg\max_{f^M} \sum_{e\in E} \ell^R_S(x^R_e + x^M_e) x^R_e$.

Linear Program $\mathcal{P}^R_{\mathrm{WE}}$	Convex Program $\mathcal{C}^M_{\mathrm{NE}}$
minimize $\displaystyle\sum_{S\in\Sigma_R} L^R_S(f^R,f^M)g^R_S$	maximize $\displaystyle\sum_{S\in\Sigma_R} L^R_S(f^R,g^M)f^R_S$
subject to $\displaystyle\sum_{S\in\Sigma_R} g^R_S = d_R$	subject to $\displaystyle\sum_{S\in\Sigma_M} g^M_S = d_M$
$g^R_S \geq 0,\ \text{ for all } S\in\Sigma_i$	$g^M_S \geq 0,\ \text{ for all } S\in\Sigma_i$

Figure 4.5: The linear program $\mathcal{P}^R_{\mathrm{WE}}$ for the rational players (left) and the convex program $\mathcal{C}^M_{\mathrm{NE}}$ for the malicious player (right), in the fixed point formulation of the equilibrium.

We reformulate Definition 4.5.5 in terms of load allocations for the rational players, similarly to the case of Wardrop equilibria, and thus view the pair (f^R,ρ_M) as a strategy profile. For convenience we use the function L^i_S defined in Section 4.5.1.

Definition 4.5.6. The strategy profile (f^R,ρ_M) is an equilibrium if

1. $\mathbb{E}_{f^M\sim\rho_M}L^R_S(f^R,f^M) = \min_{S'\in\Sigma_R}\mathbb{E}_{f^M\sim\rho_M}L^R_{S'}(f^R,f^M)$ whenever $f^R_S>0$.
2. $\mathrm{supp}(\rho_M)\subseteq \arg\max_{f^M}\sum_{S\in\Sigma_R}L^R_S(f^R,f^M)f^R_S$.

Babaioff et al. [13] proved that every congestion game with a malicious player with non-decreasing latency functions has an equilibrium. They also proved, employing the Debreu-Fan-Glicksberg theorem, that when the latency functions are concave, such a game has a pure equilibrium, i.e. an equilibrium of the form (f^R,f^M). We will obtain a PPAD-membership result for this case. The result will be based on the following fixed formulation of the problem, very similar to that of linear program $\mathcal{P}_{\mathrm{WE}}$ presented in Section 4.5.1, and convex program $\mathcal{C}_{\mathrm{NE}}$ presented in Section 4.5.1, refined for the problem at hand.

For the rational players, best responses are expressed by linear program $\mathcal{P}^R_{\mathrm{WE}}$ in variables g^R of the left-hand side of Figure 4.5.. For the malicious player, the best response is expressed by convex program $\mathcal{C}^M_{\mathrm{NE}}$ in variables g^M of the right-hand side of Figure 4.5.

From the fixed point formulation in terms of the programs of Figure 4.5, PPAD-membership is almost immediate via arguing that those best response can be computed by linear-OPT-gates, as long as the subgradients of the objective functions can be computed by a linear pseudo-circuit. Hence we have the following theorem.

Theorem 4.5.1. *Computing an equilibrium of a congestion game with a malicious player in which the latency functions are given by linear pseudo-circuits and their subgradients are given by linear pseudo-circuits computing piecewise-constant functions is in PPAD.*

Proof. Linear program $\mathcal{P}^R_{\mathrm{WE}}$ of Figure 4.5 computes the best response of the class of rational players, and C^M_{NE} computes the best response of the malicious players. As long as those optimization programs can be computed by linear-OPT-gates, then the game is a LBRO game (Definition 4.4.4, noting Remark 14 as the oracles also input a players' own strategy) and PPAD-membership follows from Theorem 4.4.3. Indeed, for both programs the feasible domain is non-empty and bounded with no gate inputs appearing in the left-hand side of the constraints. By the statement of the theorem, the latency functions can be computed by linear pseudo-circuits. In the case of linear program $\mathcal{P}^R_{\mathrm{WE}}$, the subgradient is a sum of latency functions (one for each $S \in \Sigma_R$), which by extension can also be computed by a linear pseudo-circuit. For convex program C^M_{NE}, by the definition of the latency function $L^R_S(f^R, g^M)$ in Equation (4.22), we can use the linear pseudo-circuits that compute the latency functions, as well as those that compute their subgradients. Those subgradients are piecewise-constant functions, which we then can multiply by the variables g^M by the machinery developed in Section 4.3.4. In the end, we obtain a function which can be computed by a linear pseudo-circuit, and the whole linear program can be computed by a linear-OPT-gate. $\square$

A direct corollary of Theorem 4.5.1 is to the case when the latency functions are piecewise-linear functions given explicitly as part of the input. Indeed, in that case the objective functions are piecewise-quadratic and their subgradients are piecewise-linear functions, hence computable by a linear pseudo-circuit.

We complement Theorem 4.5.1 above with an example showing the tightness of the class of latency functions considered in the statement of theorem. Indeed, if one moves to more general concave latency functions, then the only equilibria of the game may be irrational.

Example 4 (Only irrational equilibria)**.** Consider a game with only rational players (the game with a malicious player is a generalization). Define the function $f\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ by

$$f(x) = \begin{cases} 3x - x^2 & \text{for } x \leq 1 \\ 1 + x & \text{for } x > 1 \end{cases} .$$

The function f passes through the origin and is continuous, differentiable, increasing, and concave. Consider the non-atomic congestion game with two resources 1 and 2, and latency functions $l_1(x) = f(x)$ and $l_2(x) = 2f(x)$. If $(x, 1-x)$ is an equilibrium, then it must hold that $3x - x^2 = 2(3(1-x) - (1-x)^2$ which as $x \in [0,1]$ implies that $x = (\sqrt{41} - 5)/2$.

We remark that for games with general concave latency functions, a FIXP-membership result is implied by (although not explicitly stated in) [101], as these games are concave games.

4.5.2 Network Congestion Games

We now define an interesting subclass of congestion games, called *network congestion games*. Network congestion games allow the players' strategies to be succinctly described by a flow network.

Multi-commodity flow networks To be able to define multi-class network congestion games we consider a *multi-commodity flow model*. More precisely, consider networks given by a directed graph $G = (V, E)$ on which we consider routing of k commodities. For each commodity $i \in [k]$ we are given a source node $s_i \in N$, a target node $t_i \in N$, and a flow demand $d_i > 0$. A cost function c is in general a function $c\colon E \to \mathbb{R}$, but we will however only consider non-negative cost functions. We assume separate cost functions c^i for each commodity $i \in [k]$.

A flow x is a function $x\colon E \to \mathbb{R}$. To be feasible, the flow x is required to be non-negative. The balance $b(x)$ of x is the function given by

$$b(x)_u = \sum_{uv \in E} x_{uv} - \sum_{vu \in E} x_{vu} \text{ for } u \in N.$$

For every $i \in [k]$, define the vector b^i by

$$b^i_{s_i} = d_i,\ b^i_{t_i} = -d_i, \text{ and } b^i_u = 0, \text{ for every } u \in N \setminus \{s_i, t_i\}.$$

The cost of x with respect to commodity i is $\sum_{uv \in E} c^i_{uv} x_{uv}$.

A multi-commodity flow in G consists of a *flow profile* $x = (x^1, \ldots, x^k)$ of flows for each commodity $i \in [k]$. The individual flow x^i is *feasible* if x^i is nonnegative and satisfies the balance constraint $b(x^i) = b^i$. We will say that the flow profile x is feasible if x^i is feasible for all i. The total flow $\overline{x}$ induced by such a multi-commodity flow $x = (x^1, \ldots, x^k)$ is the sum $\overline{x} = \sum_{i=1}^k x^i$.

For a directed path P in G, we denote by γ^P the unit path flow given by

$$\gamma^P_{uv} = 1 \text{ for } uv \in P \text{ and } \gamma^P_{uv} = 0 \text{ for } uv \notin P.$$

Similarly, for a directed cycle C in G, we denote by γ^C the unit cycle flow given by

$$\gamma^C_{uv} = 1 \text{ for } uv \in C \text{ and } \gamma^C_{uv} = 0 \text{ for } uv \notin C.$$

The costs $c^i(P)$ and $c^i(C)$ of P and C with respect to commodity i are given as

$$c^i(P) = \sum_{uv \in P} c^i_{uv} \quad \text{and} \quad c^i(C) = \sum_{uv \in C} c^i_{uv}.$$

Note that $c^i(P) = c^i(\gamma^P)$ and $c^i(C) = c^i(\gamma^C)$. Let $\mathcal{P}_i$ denote the set of all directed (s_i, t_i) paths in G and let $\mathcal{P} = \cup_{i=1}^k \mathcal{P}_i$. Let C denote the set of all directed cycles in G.

A *routing* of commodity i in G is a flow x^i of the form $x^i = \sum_{P \in \mathcal{P}_i} \alpha_P \gamma^P$, where $\sum_{P \in \mathcal{P}_i} \alpha_P = d_i$ and $\alpha_P \geq 0$ for every $P \in \mathcal{P}_i$. Note that x^i is feasible by definition. A *multi-commodity routing* is a multi-commodity flow $(x^1, \dots, x^k)$ such that x^i is a routing of commodity i for every $i \in [k]$. The network model we consider is *uncapacitated*, i.e. it does not place any upper limits on the flow on arcs. This means that a minimum cost routing must place non-zero flow only on minimum cost paths.

Lemma 19. *Suppose that $x^i = \sum_{P \in \mathcal{P}_i} \alpha_P \gamma^P$ is a routing of commodity i minimizing the cost $c^i(x^i)$. Then $c^i(P) = \min_{P' \in \mathcal{P}_i} c^i(P')$ whenever $\alpha_P > 0$.*

It is well known that any flow may be decomposed path and cycle flows. We shall make use of the following special case.

Lemma 20. *Let x^i be a feasible flow for commodity i. Then there are $\alpha_P \geq 0$ for $P \in \mathcal{P}_i$ and $\beta_C \geq 0$ for $C \in C$ such that $x^i = \sum_{P \in \mathcal{P}_i} \alpha_P \gamma^P + \sum_{C \in C} \beta_C \gamma^C$.*

If $b(x^i) = b^i$ the cost of x^i with respect to commodity i is thus given as $c(x^i) = \sum_{P \in \mathcal{P}_i} \alpha_P c^i(P) + \sum_{C \in C} \beta_C c^i(C)$. We have the following direct consequence.

Corollary 7. *Let x^i be a feasible flow for commodity i minimizing the cost $c^i(x^i)$. If $c^i(C) > 0$ for any cycle C for which there exist $uv \in E$ such that $x^i_{uv} > 0$, then x^i is a routing of commodity i.*

Multi-class network congestion games. From a multi-commodity flow network $G = (V, E)$ we naturally obtain a multi-class congestion game. The set of resources is simply the set E of arcs in the network and the set Σ_i of pure strategies of class i is equal to the set $\mathcal{P}_i$ of directed (s_i, t_i) paths in G. This gives a very succinct description of the strategies that allows the relevant optimization problems to be formulated with variables corresponding to resources rather than strategies. Note that a load allocation of class i in the congestion game corresponds exactly to a routing of commodity i in G.

For the case of Wardrop equilibrium we consider the following linear program in variables y^i_{uv}, where $y^i = (y^i_{uv})_{uv \in E}$, x is a given non-negative multi-commodity flow and $\overline{x}$ the resulting total flow.

<u>Linear Program $\mathcal{P}^G_{\text{WE}}$</u>

$$\begin{aligned} \text{minimize} \quad & \sum_{uv \in E} \ell^i_{uv}(\overline{x}_{uv}) y^i_{uv} \\ \text{subject to} \quad & b(y^i) = b^i \\ & y^i \geq 0 \end{aligned}$$

This linear program is simply expressing minimum cost feasible flows for commodity i with costs given by the latency functions and the total flow $\overline{x}$. In order to identify these minimum cost flows with minimum cost routing , using Corollary 7, we need to impose a very mild restriction on the latency functions.

Definition 4.5.7. We say that the the latency functions l^i_{uv} are without *free cycles* if for every $i \in [k]$, every $C \in \mathcal{C}$ and every feasible flow x, for which $x_{uv} > 0$ for some $uv \in C$, we have $\sum_{uv \in C} \ell^i(x_{uv}) > 0$.

We have the following fixed point characterization of Wardrop equilibrium in network congestion games, using Lemma 19 and Corollary 7.

Proposition 8. *In a network congestion games with latency functions without free cycles, a flow $x = (x^1, \dots, x^k)$ is a Wardrop equilibrium if and only x^i is an optimal solution to the linear program $\mathcal{P}^G_{\mathrm{WE}}$ for all $i \in [k]$.*

For the case of atomic splittable congestion games and with the assumptions of Lemma 18 we consider the following convex program in variables in variables y^i_{uv}, where $y^i = (y^i_{uv})_{uv \in E}$.

Convex Program $\mathcal{C}^G_{\mathrm{NE}}$

$$\begin{aligned} \text{minimize} \quad & \sum_{uv \in E} \ell^i_{uv}(y^i_{uv} + \sum_{j \neq i} x^j_{uv}) y^i_{uv} \\ \text{subject to} \quad & b(y^i) = b^i \\ & y^i \geq 0 \end{aligned}$$

Analogously to the case of the linear program $\mathcal{P}^G_{\mathrm{WE}}$ we would like to ensure that optimal solutions to $\mathcal{C}^G_{\mathrm{NE}}$ are routings of commodity i. Here however, the cost functions of the flow depend on the flow variables, and we thus require a stronger assumption on the latency functions.

Proposition 9. *In a network congestion game with non-decreasing latency functions without free cycles and satisfying the assumptions of Lemma 18, a flow $x = (x^1, \dots, x^k)$ is a Nash equilibrium if and only if x^i is an optimal solution to the convex program $\mathcal{C}^G_{\mathrm{NE}}$ for all $i \in [k]$.*

Proof. We need to prove that each x^i that is an optimal solution to $\mathcal{C}^G_{\mathrm{NE}}$ is a routing of commodity i. By Lemma 20 we have $\alpha_P \geq 0$ for $P \in \mathcal{P}_i$ and $\beta_C \geq 0$ for $C \in \mathcal{C}$ such that $x^i = \sum_{P \in \mathcal{P}_i} \alpha_P \gamma^P + \sum_{C \in \mathcal{C}} \beta_C \gamma^C$. Suppose for contradiction that there exists $C \in \mathcal{C}$ such that $\gamma_C > 0$ and define the new flow $\hat{y}^i = y^i - \beta_C \gamma^C$. Since γ^C is a cycle flow and $\beta_C \gamma^C \leq y^i$ it follows that $\hat{y}^i$ is feasible. Since the latency functions are non-decreasing it follows that $\ell^i_{uv}(\hat{y}^i_{uv} + \sum_{j \neq i} x^j_{uv}) \leq \ell^i_{uv}(\hat{y}^i_{uv} + \sum_{j \neq i} x^j_{uv})$ for all $uv \in E$, and thus also $\ell^i_{uv}(\hat{y}^i_{uv} + \sum_{j \neq i} x^j_{uv}) \hat{y}^i_{uv} \leq \ell^i_{uv}(\hat{y}^i_{uv} + \sum_{j \neq i} x^j_{uv}) y^i_{uv}$ for all $uv \in E$. Now, since the latency functions are without free cycles there exist $uv \in C$ such that $\ell^i(y^i_{uv} + \sum_{j \neq i} x^j_{uv}) > 0$. It follows that $\ell^i(\hat{y}^i_{uv} + \sum_{j \neq i} x^j_{uv}) \hat{y}^i_{uv} \leq \ell^i(y^i_{uv} + \sum_{j \neq i} x^j_{uv}) \hat{y}^i_{uv} < \ell^i(y^i_{uv} + \sum_{j \neq i} x^j_{uv}) y^i_{uv}$. Taken together this contradicts the optimality of y^i. □

PPAD-membership for non-atomic network congestion games

The PPAD-membership result for finding Wardrop equilibria in multi-class network non-atomic congestion games follows almost immediately from the fixed point characterization of equilibria in Proposition 8. The following result generalizes that of Meunier and Pradeau [172].

Theorem 4.5.2. *Finding a Wardrop equilibrium of a multi-class network non-atomic congestion game when the latency functions are without free cycles and they are given by* linear *pseudo-circuits is in PPAD.*

Proof. Linear program $\mathcal{P}^G_{\text{WE}}$ can be interpreted as computing the best response of a player corresponding to class i, for the equilibrium concept of Wardrop equilibrium. $\mathcal{P}^G_{\text{WE}}$ obviously has a non-empty and bounded feasible domain. Since the latency functions ℓ^i_{uv} can be computed by linear pseudo-circuits, the subgradient of the objective function can be computed by a linear pseudo-circuit, and hence the whole linear program can be computed by a linear-OPT-gate. By using this linear-OPT-gate as the oracle, the game becomes a LBRO game (Definition 4.4.4), and the theorem follows as a corollary of Theorem 4.4.3. □

Again, a direct corollary of Theorem 4.5.2 is to the case when the latency functions are piecewise-linear functions given explicitly as part of the input. Indeed, in that case the objective functions are piecewise-quadratic and their subgradients are piecewise-linear functions, hence computable by a linear pseudo-circuit.

Remark 20 (No free cycles)**.** As we mentioned earlier, the no free cycles condition (see Definition 4.5.7 is rather mild one. It is in fact a milder condition that the usual condition in this literature, which requires that if $l_e(x) = 0$, then $x = 0$, i.e., the latencies are 0 only on 0 points. The works on linear latency functions implicitly make this assumption, and we obtain strict generalizations of those via considering more general latency functions.

For completeness, we also provide a simple example in which, if one goes beyond the latency functions captured by Theorem 4.5.2, e.g., to quadratic latency functions, then the game may possess only irrational Wardrop equilibria.

Example 5 (Only irrational Wardrop equilibria)**.** Consider a very simple nonatomic network congestion game where there is only one class given by the interval $[0, 1]$. The node set V of the network G consists of two nodes s and t. For the arc set E, there are two arcs e_1 and e_2 from s to t for the class, with latency functions $l_{e_1}(x) = x^2$ and $l_{e_2}(x) = 2x^2$. Now if $(x, 1 - x)$ is an equilibrium, then it must hold that $x^2 = 2 \cdot (1 - x)^2$ which as $x \in [0, 1]$ implies that $x = 2 - \sqrt{2}$.

Example 5 establishes that the class captured by our Theorem 4.5.2 is tight, as if one moves to larger classes of latency functions, Wardrop equilibria may be irrational. We remark that for these cases, a FIXP-membership result follows from the results of [101] (althought not explicitly stated there), since multi-class non-atomic network congestion games are concave games.

PPAD-membership for atomic splittable network congestion games

In the same fashion as in the previous section, the PPAD-membership result for finding Nash equilibria in multi-class network atomic congestion games follows almost immediately from the fixed point characterization of equilibria in Proposition 9. The following result generalizes that of Klimm and Warode [155]. We also again note the remarkable simplification of the proof that the linear-OPT-gate allows us to achieve, compared to that of Klimm and Warode [155].

Theorem 4.5.3. *Finding a Nash equilibrium of a multi-class network atomic splittable congestion game when the latency functions are without free cycles, they are given by* linear *pseudo-circuits and their subgradients are given by* linear *pseudo-circuits computing piecewise-constant functions is in PPAD.*

Proof. The proof is very similar to that of Theorem 4.5.2 above. Convex program C^G_{NE} can be interpreted as computing the best response of a player corresponding to class i. C^G_{NE} obviously has a non-empty and bounded feasible domain. To compute the subgradient of the objective function, we can use the linear pseudo-circuits that compute the latency functions, as well as those that compute their subgradients. Those subgradients are piecewise-constant functions, which we then can multiply by the variables y_{uv} by the machinery developed in Section 4.3.4. In the end, we obtain a function which can be computed by a linear pseudo-circuit, and the whole linear program can be computed by a linear-OPT-gate. By using this linear-OPT-gate as the oracle, the game becomes a LBRO game (Definition 4.4.4 and the theorem follows as a corollary of Theorem 4.4.3. □

Once gain, we obtain the PPAD-membership of multi-class network atomic splittable congestion game when the latency functions are piecewise-linear functions given explicitly as part of the input as a direct corollary of Theorem 4.5.3. The result is however more general, and also captures cases where the latency functions and their subgradients are accessed implicitly via their corresponding linear pseudo-circuits. Additionally, similarly to the statement of Theorem 4.5.2, we remark that the no free cycles condition is implicitly made in the case of linear latencies, and hence we obtain a strict generalization of the previous results.

Again, for completeness, we provide a simple example of a game that only has irrational equilibria, if the latency functions go beyond those captured by Theorem 4.5.3.

Example 6 (Only irrational Nash equilibria)**.** The example is the same as the one in Example 5. We consider a network with a single player and two arcs e_1 and e_2 from s to t, with the same latency functions as before. Here, since we have an atomic splittable game, the single player has to minimize the latency $x \cdot l_1(x) + (1 - x) \cdot l_2(1 - x) = x^3 + 2 \cdot (1 - x)^3$ which for $x \in [0, 1]$ is minimized in $x = 2 - \sqrt{2}$.

Again, Example 6 establishes that the class captured by our Theorem 4.5.3 is tight, as if one moves to larger classes of latency functions, Nash equilibria may be irrational.

Again, for similar reasons as before, a FIXP-membership for general concave latency functions follows from the results of [101].

4.6 Competitive Equilibria in Arrow-Debreu Markets

In this section, we show how our linear-OPT-gate can be used to show the PPAD-membership of finding competitive equilibria in Arrow-Debreu markets. We will consider such markets in which the utility and production functions have a certain form which allows equilibrium points to be rational numbers. For a gentle introduction, we first apply the technique to the simpler cases of exchange markets (i.e., no production) with linear utilities (Section 4.6.1) and of markets with linear utilities and productions (Section 4.6.2), before we move to our most advanced applications.

The most general known class of utilities and productions for which rational competitive equilibria are possible are those of *Leontief-free utilities and productions*, introduced by Garg et al. [119], who also proved the PPAD-membership of finding equilibria in those markets. We provide an alternative proof via the employment of the linear-OPT-gate, which is *conceptually* and *technically* much simpler in Section 4.6.3.

The Leontief-free class generalizes the well-known class of *separable piecewise linear concave (SPLC)* utilities, studied by several works, e.g., see [112, 116, 220]. SPLC utilities is in turn a generalization of linear utilities in which every agent has a piecewise linear concave utility for the amount of a good j that she receives, and her total utility for her bundle is additive over goods.

SSPLC, a new class of utilities. In Section 4.6.4, we define a new class of utility functions, coined *succinct separable piecewise linear concave (SSPLC) utilities*, which generalize the SPLC class in a different way than the Leontief-free class. Intuitively, these are functions which can have exponentially many pieces, but those are accessed *implicitly*, via access to boolean circuits computing their slopes. In this sense, the class differs from SPLC utilities not in the definition of the utility function, but in the way that it is inputted in the computational problem. As such, the existence of rational solutions for this class already implicitly follows from previous work [112] for SSPLC utilities. As we explain below however, the "LCP-approach" is inherently limited in its ability to establish computational results for this class, and in particular PPAD-membership. In contrast, by taking advantage of the machinery that we developed in Section 4.3.4, we show that our linear-OPT-gate can be used in very much the same fashion as in all the other proofs of the section, to prove that computing competitive equilibria for SSPLC utilities is in PPAD. Our proof also incorporates (explicit) SPLC production functions; we provide a discussion on the challenges of extending our technique to the case of SSPLC production functions as well in Remark 35.

Features of our Proof vs Previous Approaches

Before we proceed, we first discuss the previous results that appeared in a succession of important works in the area, and highlight the advantages of our approach over those.

Previous work and previous proofs. The origins of the literature that deals with the PPAD-membership of finding *exact* and *rational* competitive equilibria for markets for which this is possible can be traced back to the work of Eaves [80]. Eaves studied the exchange market model with linear utilities (the same one we present first in Section 4.6.1) and proved that its competitive equilibria can be found by Lemke's algorithm [160]. Referencing our discussion in Section 4.1.2, Eaves' approach, similarly to most subsequent approaches, formulates the problem as an LCP. Clearly, the class PPAD had not been defined then, but as we explained in Section 4.1.2, the PPAD-membership of the problem is implied by his proof.

Vazirani and Yannakakis [220] considered the case of SPLC utilities. Their proof does not employ the "LCP approach", but rather the "approximation and rounding approach", see Section 4.1.2. An issue with this method is that very small changes in the prices may result in drastic changes in the optimal bundles of the consumers, and hence Vazirani and Yannakakis [220] devise a set of technical lemmas that allow them to "force" certain allocations over others. Subsequently, Garg et al. [116] proved the PPAD-membership of equilibria for the same class of utility functions, via instead using the "LCP approach". Garg et al. point out that the LCP that they derive does not fall into any of the classes known to avoid secondary rays, and hence they necessarily devise an argument against ray termination (see Section 4.1.2). Besides that, the formulation of the LCP itself *"turns out to be quite complex"* [116]. An integral part of the proof, which is also useful in our regime, is the definition of the "desire" for a good j, to be the total amount of the good represented by its non-zero utility segments in the SPLC function.

For markets with production, the aforementioned work of Vazirani and Yannakakis [220] also provides a PPAD-membership result for markets with SPLC utilities and SPLC productions. The same PPAD-membership result was later recovered by [112] via the "LCP approach"; this is in fact the paper from which the quote of Section 4.1.2 is taken. The quote highlights the increasing challenge of developing these LCPs and establishing their successful termination. Indeed, the LCP of [112] naturally differs from that of [116], in that it needs to account for production. This is done via devising a set of linear programs, and then using complementary slackness and their feasibility conditions to develop the LCP needed for production. The non-homogeneity of the resulting LCP for the equilibrium problem is dealt with in a manner which is different from previous works [80, 116] and, naturally, since the developed LCP is different, Garg and Vazirani again need to argue against ray termination.

The last of the works in this sequence regarding competitive markets for goods is by Garg et al. [119]. Garg et al. define a new class of utilities/productions, coined "Leontief-free utilities/productions", which generalize the SPLC functions mentioned earlier (see Section 4.6.3 for a precise definition). What is interesting about this class is that it contains functions which are not separable, but yet they admit rational solutions, hence dispelling a potential suspicion that non-separability is what essentially leads to irrationality. Garg et al. [119] obtain the PPAD-membership of finding competitive equilibria in those markets via, again, employing the "LCP approach". Their LCP formulation turns out to be even more complex than those of previous works, as it has to differentiate between "normal" and "abnormal" variables, the latter preventing the employment of Lemke's algorithm. To circumvent this, they exploit some additional structure of their *nonstandard* LCP, and then they also *modify* Lemke's algorithm, to account for the possibility of abnormal variables becoming zero. Finally, as they devise a new LCP, they also have to argue once again against ray termination.

Our proofs. We employ the linear-OPT-gate to provide proofs which are conceptually and technically simpler than the ones discussed above. In particular, we formulate the optimal consumption (i.e., the consumers' utility maximization) and the optimal production (i.e, the firms' profit maximization) as linear programs (e.g., see Figure 4.7), which can be effectively substituted by linear-OPT-gates in a linear arithmetic circuit. Intuitively, one could view these as the "parallel" of best responses that we used in previous sections, in the domain of markets. What differentiates the proofs of this section is that these consumptions/productions need to be supported at a set of market-clearing prices.

To make sure that we can work with linear constraints (in the inputs to linear arithmetic circuit), we apply a standard variable change which was first used by Eaves [80], and then subsequently on all of the works that apply the "LCP approach": instead of sets of goods for consumption and production, we use the *expenditures* of the consumers to buy goods in their bundles, and of the firms to produce goods using other goods as raw material. Interestingly, Eaves [80] attributes this idea to Gale, and hence we refer to it as *Gale's substitution*, see Remark 22. For the prices, we develop a feasibility program (e.g., see Figure 4.8), which at a high level establishes that if certain goods (or segments) have smaller expenditures, then they are sufficiently cheaper. This is the main property that we use to argue that the prices computed by the feasibility program at a fixed point are market-clearing.

For our results in Section 4.6.4, we take advantage of the capability of the linear-OPT-gate to compute implicit functions and correspondences, as detailed in Section 4.3.4. In contrast, the "LCP approach" would need to have explicit variables for each segment, resulting in an LCP of exponential size in the size of the input (which is the size of the given input circuits). This is sufficient to prove rationality of solutions, but not to construct the computational reduction needed for a PPAD-membership.

Compared to the applications in other sections of our paper, the proofs that we provide in this section are probably the most technically involved. The main complication lies in arguing the market clearing of the outputted prices, which still requires a relatively short proof. We remark that most of the space in the following sections is used for introducing the corresponding settings and putting our technique in context, rather than the proofs themselves.

Finally, we emphasize that, as we mentioned also in Section 4.1.2, the "LCP approach" satisfies some other desirable properties, e.g., see [119] for a discussion.

As our focus here is the PPAD-membership of the problems, we do not discuss them further.

Roadmap. We provide a brief roadmap of the section. In Section 4.6.1 we introduce our technique in the simple setting of exchange markets with linear utilities. To gently incorporate production, in Section 4.6.2 we consider markets with linear utilities and linear productions. In Section 4.6.3 we consider the case of Leontief-free utilities and productions. Finally in Section 4.6.4 we prove the PPAD-membership of finding competitive equilibria in markets with SSPLC utilities and SPLC productions.

4.6.1 Exchange Markets with Linear Utilities

First, we explain how our linear-OPT-gate can be used to show the PPAD-membership of finding competitive equilibria in a simple but fundamental variant of the Arrow-Debreu market model, that of *exchange markets* with *linear utilities*. An exchange market is one in which there is no production, and each consumer brings an endowment to the market. The PPAD-membership for these markets demonstrates the strength of the technique for a simple model, before moving on to apply it to more general market settings.

Our main theorem in this section is the following:

Theorem 4.6.1. *Computing a competitive equilibrium in an exchange market with linear utilities is in PPAD.*

Exchange Markets. In an exchange market $\mathcal{M}$, we have a set N of consumers and a set G of infinitely divisible goods. Let $n = |N|$ and $m = |G|$. We will typically use index i to refer to consumers and j or g to refer to goods. Each consumer brings an endowment $w_i = (w_{i1}, \dots, w_{im})$ to the market, with $w_{ij} \geq 0$ for all $i \in N$ and $j \in G$. We may assume without loss of generality that for every good j, we have $\sum_{i \in N} w_{ij} = 1$, i.e., that the total endowment of each good is 1. We will use $\mathbf{x}_i = (x_{i1}, \dots, x_{im})$ to denote the vector of quantities of goods allocated to consumer $i \in N$ in $\mathcal{M}$, and we will call it the *bundle* of consumer i. Let $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ be the vector of such bundles. We will use $\mathbf{p} = (p_1, \dots, p_m)$ to denote the vector of *prices* in $\mathcal{M}$, one for each good $j \in G$. Prices are non-negative, so $p_j \geq 0$ for all $j \in G$. Given a vector of prices $\mathbf{p}$ the *budget* of consumer $i \in N$ is defined as $\sum_{j \in G} w_{ij} p_j$; intuitively, this is the amount of money that the consumer acquires by selling her endowment at prices $\mathbf{p}$.

Utility Functions. Every consumer has a utility function $u_i : \mathbb{R}^m_{\geq 0} \to \mathbb{R}_{\geq 0}$ mapping a bundle $\mathbf{x}_i$ to a non-negative real number. In this section, these utilities are *linear*, i.e., every consumer $i \in N$ has a utility u_{ij} for every good $j \in G$, and her utility for the bundle $\mathbf{x}_i$ is $u_i(\mathbf{x}_i) = \sum_{j \in G} u_{ij} x_{ij}$, where u_{ij} is the value of consumer i for good j.

Competitive Equilibrium. Next, we provide the definition of a competitive equilibrium (often referred to as a *market* or a *Walrasian* equilibrium).

Definition 4.6.1 (Competitive equilibrium)**.** A competitive equilibrium of an exchange market $\mathcal{M}$ is a pair $(\mathbf{p}^*, \mathbf{x}^*)$ consisting of a prices $\mathbf{p}^*$ and and bundles $\mathbf{x}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*)$ such that

1. $u_i(x_i^*) = \sum_{j \in G} x_{ij}^* u_{ij}$ is maximized, subject to $\sum_{j \in G} x_{ij}^* p_j^* \leq \sum_{j \in G} w_{ij} p_j^*$ and $x_{ij}^* \geq 0$, for any consumer $i \in N$. ***(bundle optimality)***

2. $\sum_{i \in N} x_{ij}^* = 1$, for any good $j \in G$. ***(market clearing)***

Condition 1 above guarantees that at the equilibrium prices $\mathbf{p}^*$, every consumer receives the best possible bundle that they can buy given their budget and that each good is allocated in a non-negative amount. Condition 2 guarantees that the market clears, i.e., that the total quantity of goods sold is equal to the total endowment $\sum_{i \in N} w_{ij}$, which, recall, we have assumed without loss of generality that is equal to 1.

Sufficiency conditions. A competitive equilibrium as defined above exists for every exchange market $\mathcal{M}$, under some *sufficiency conditions*. Following Vazirani and Yannakakis [220], we will use the sufficiency conditions used by Eaves [80], namely that:

1. For every good $j \in G$, there exists some consumer $i \in N$ that values the good positively, i.e., $u_{ij} > 0$.

2. For every consumer $i \in N$, there exists some good $j \in G$ that the consumer endows in a positive amount, i.e., $w_{ij} > 0$.

3. The market is *not reducible*. A market is reducible where there exists a proper subset $N' \subset N$ of the consumers and a subset $G' \subseteq G$ of the goods such that

   a) all of the goods in G' are entirely endowed by consumers in N', i.e., $\sum_{i \in N'} w_{ij} = 1$ for all $j \in G$,

   b) all of the consumers in N' only value positively goods in G', i.e., for all $i \in N'$ and $g \in G \setminus G'$, it holds that $u_{ig} = 0$.

   Reducibility intuitively captures the fact that if a strict subset of consumers completely endow a subset of the goods and only value those goods positively, then they could form their own market $\mathcal{M}'$ and exchange between them. Looking ahead, the non-reducibility of the market is a special case of a property called *strong connectivity of the economy graph* [167], which we will make use of for the more general market settings that we consider later on.

Optimality and bang-per-buck (BPB). The bundle optimality condition of the competitive equilibrium stipulates that each consumer buys the best possible bundle at the given prices. For linear markets, these optimal bundles have a crisp characterization, in terms of the *bang-per-buck (BPB)*. Given consumer $i \in N$ and prices $\mathbf{p}$, the BPB of a good $j \in G$ is defined as $\mathrm{BPB}_i(j) = \frac{u_{ij}}{p_j}$. An optimal bundle only contains non-zero quantities of goods for which the BPB is maximum, i.e., $j \in \arg\max_{j \in G} \mathrm{BPB}_i(j)$.

Bounds on the prices. We may without loss of generality assume that all the prices are strictly positive, i.e., $p_j > 0$ for all $j \in G$. Indeed, if $p_j = 0$, then there would be infinite demand for good j, contradicting market clearing. As such, in any equilibrium, the prices are strictly positive. Note that by this assumption, the quantity $\mathrm{BPB}_i(j)$ is well-defined for every $j \in G$.

Remark 21 (Normalized Prices)**.** Given that $p_j > 0$ for all $j \in G$, we can normalize the prices to sum to 1 without loss of generality, i.e., we may assume that $\sum_{j \in G} p_j = 1$.

We will use a parameter ε to capture the fact that if the price p_j for a good $j \in G$ is sufficiently smaller than the price $p_{j'}$ for a good $j' \in G$, then $\mathrm{BPB}(j) > \mathrm{BPB}(j')$. This will allow us to "control" which goods certain consumers buy in their optimal bundles. Specifically, we can compute $\varepsilon > 0$ such that

$$\text{If } p_j \leq \varepsilon \cdot p_{j'} \text{ and } u_{ij} > 0 \text{ then } \mathrm{BPB}_i(j) > \mathrm{BPB}_i(j').$$

Additionally, we can pick ε to be sufficiently small such that $\varepsilon < \frac{w_{ij}}{m}$ for all $i \in N$ and all $j \in G$ with $w_{ij} > 0$.

Given ε, we will impose a stricter lower bound on the prices, which will be useful later on: in particular, we will assume that for all $j \in G$, $p_j \geq \frac{\varepsilon^m}{m}$. Our PPAD-membership proof will also establish that a competitive equilibrium still always exists even under this additional restriction.

Preprocessing

The high-level overview of the proof will be the following. We will define a function F from a convex compact domain D to itself, in a way that ensures that a competitive equilibrium of $\mathcal{M}$ can be recovered from a fixed point of F. The input and output variables of the function will be outputs of a linear program $\mathcal{P}$ (see Program (4.2) in Section 4.3) and a feasibility program $\mathcal{Q}$ (see Program (4.3) in Section 4.3). Intuitively, $\mathcal{P}$ will be used to compute optimal bundles $\mathbf{x}_i$ for the consumers, and $\mathcal{Q}$ will be used to compute equilibrium prices $\mathbf{p}$. These programs will be of a particular form amenable to the use of our linear-OPT-gate. As such, we will be be able to encode F via a linear arithmetic circuit containing linear-OPT-gates.

Straightforward choice: x and p. A straightforward choice would be to have the pair $(\mathbf{x}, \mathbf{p})$ as the input/output to the function F. In that case, the linear program P

would be the following:

$$\begin{aligned}
\text{maximize } & \sum_{i\in N}\sum_{j\in M} u_{ij}x_{ij} \\
\text{subject to } & \sum_{j\in G} x_{ij}p_j \leq \sum_{j\in G} w_{ij}p_j,\ \forall i \in N \\
& x_{ij} \geq 0,\ \forall i \in N, j \in G
\end{aligned} \tag{4.23}$$

While the variables of this linear program are the variables x_{ij}, the prices p_j are gate inputs (see Definition 4.3.1), and the budget constraints of the form $\sum_{j\in G} x_{ij}p_j \leq \sum_{j\in G} w_{ij}p_j$ cannot be present in a linear program that we would like to compute by the linear-OPT-gate.

Change of variables: Expenditure. To circumvent this obstacle, we follow an idea used by Eaves [80] that "linearizes" the constraints via an appropriate change of variables. Eaves [80] attributes this idea to Gale, via private communication. Thus, we refer to this change of variables as *Gale's substitution*.

Remark 22 (Gale's substitution)**.** Let $q_{ij} = x_{ij}p_j$ be the *expenditure* of consumer i on good j, i.e., how much money the consumer spends on the good given an allocation x_{ij} and a price p_j. This change of variables is by now very much standard in the literature, used among others by Chaudhury et al. [48], Eaves [80], Garg and Vazirani [112], Garg et al. [119].

With this at hand, the budget constraints now become

$$\sum_{j\in G} q_{ij} \leq \sum_{j\in G} w_{ij}p_j.$$

These are clearly linear in the variables q_{ij} with no gate inputs appearing in the expression. In turn, the objective function of $\mathcal{P}$ becomes

$$\sum_{i\in N}\sum_{j\in M} \frac{u_{ij}}{p_j} q_{ij}.$$

Interestingly, the coefficient of each variable q_{ij} is the bang-per-buck $\text{BPB}_i(j)$ of good j for consumer i. In an optimal solution to $\mathcal{P}$, the consumer will only spend money on goods with maximum BPB. That being said, the format of the objective function (with p_j appearing in the denominator) is not one that can be handled by the linear-OPT-gate, as its subgradient cannot be computed by a linear pseudo-circuit. For this reason, we transform the objective function to

$$\sum_{i\in N}\sum_{j\in M} \frac{p_j}{u_{ij}} q_{ij}.$$

and the maximization in linear program (4.23) to minimization. Clearly, in an optimal solution, the consumer is again only spending on goods with minimum ratio $\frac{p_j}{u_{ij}}$, i.e.,

Linear Program $\mathcal{P}$	Feasibility Program Q
$\text{minimize} \sum_{i\in N}\sum_{j\in G}\frac{p_j}{u_{ij}}q_{ij}$	$e_j < e_{j'} \Rightarrow p_j \leq \varepsilon\cdot p_{j'},\ \forall j,j' \in G$
$\text{subject to} \sum_{j\in G} q_{ij} = \sum_{j\in G} w_{ij}p_j$	$p_j \geq \frac{\varepsilon^m}{m},\ \forall j \in G$
$q_{ij} \geq 0,\ \forall j \in G$	$\sum_{j\in G} p_j = 1$

Figure 4.6: The linear program $\mathcal{P}$ used to recover optimal bundles (left) and the feasibility program Q used to find market-clearing prices (right).

with maximum BPB, therefore purchasing an optimal bundle. This objective function form is of a form amenable to the use of the linear-OPT-gate, as its subgradient is a linear function.

However, there is a potential issue that arises when converting the maximization objective to minimization. If we keep the budget constraint $\sum_{j\in G} q_{ij} \leq \sum_{j\in G} w_{ij}p_j$ intact, then setting $q_{ij} = 0$ for all $i \in N$ and $j \in G$ is an optimal solution to the linear program, while it is clearly not an expenditure consistent with a competitive equilibrium. To handle this, we observe that in an optimal solution to linear program (4.23), where the objective is maximization, we may assume that all the budget constraints have zero slack, i.e., that $\sum_{j\in G} x^*_{ij}p^*_j = \sum_{j\in G} w_{ij}p^*_j$. Indeed, if one could increase some variable x_{ij} without violating the budget constraint for some consumer $i \in N$, then the consumer would certainly not receive a lower utility in the objective function. Correspondingly, the constraints $\sum_{j\in G} q_{ij} \leq \sum_{j\in G} w_{ij}p_j$ can be converted to equality constraints, i.e., $\sum_{j\in G} q_{ij} = \sum_{j\in G} w_{ij}p_j$. In turn, these constrain the expenditure variables to receive positive values in an optimal solution, ensuring that each consumer spends their entire budget.

Summarizing, the linear program $\mathcal{P}$ can be seen on the left-hand side of Figure 4.6.

Excess expenditure and the program Q. While linear program $\mathcal{P}$ will be used to obtain the optimal expenditures q_{ij} (and as consequence the optimal bundles x_{ij}), as we mentioned earlier, the equilibrium prices will be obtained via a feasibility program Q. Before we define the program, we introduce the notion of excess expenditure, which will be useful later on. The *excess expenditure* e_j of a good $j \in G$ is defined as the difference between the total expenditure of all consumers $i \in N$ for that good and the price of the good, i.e.,

$$e_j = \sum_{i\in N} q_{ij} - p_j.$$

Then, at equilibrium prices $\mathbf{p}^*$ we will have market clearing, i.e., $e^*_j = 0$ for all $j \in G$.

We are now ready to define our feasibility program Q; see the right-hand side of Figure 4.6. The equilibrium prices will be obtained as the output $\mathbf{p}^*$ of Q.

Membership in PPAD: The proof of Theorem 4.6.1.

We will develop the proof in three steps, namely (a) construction of the function F and arguing that it can be represented by a linear arithmetic circuit containing linear-OPT-gates, (b) showing that the linear-OPT-gate can compute all the necessary components, and (c) arguing that a competitive equilibrium can be recovered from a fixed point of F.

The function F. Given the above, we will define $F : D \to D$ with domain

$$D = \left\{ p \in \Delta^{m-1} : \forall j \in G, p_j \geq \frac{\varepsilon^m}{m} \right\} \times [0,1]^{nm}.$$

An input to F is a pair $(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ of prices and expenditures, where $\bar{\mathbf{q}} = (\bar{q}_{ij})_{i \in N, j \in G}$ and the output is another such pair $(\mathbf{p}^*, \mathbf{q}^*)$. The domain D is the one above since $\sum_{j \in G} \bar{p}_j = 1$ (see Remark 21), and by the fact that in any feasible set of expenditures, $\sum_{j \in G} \bar{q}_{ij} \leq \sum_{j \in G} w_{ij} \bar{p}_j \leq 1$.

Remark 23. Note that while the linear program $\mathcal{P}$ and the feasibility program Q were written in a general form in Figure 4.6, when using $(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ as the input, we would have to substitute p_j with $\bar{p}_j$ in $\mathcal{P}$ (with the variables being q_{ij}) and q_{ij} and p_{ij} with $\bar{q}_{ij}$ and $\bar{p}_j$ respectively in the expression for e_j in Q (with the variables being p_j). While we do not write this out explicitly in all of the applications in our paper, here we thought it would be important to mention, in order to avoid any confusion with p_j potentially being on the left-hand side of the conditional constraints in Q. It is $\bar{p}_j$ that appears on the left-hand side, which is a gate input rather than a variable, and thus is in accordance with the definition of feasibility programs in (4.3) of Section 4.3.

Computation by the linear-OPT-gate. In the next lemma, we argue formally what we have alluded to earlier, namely that the solutions to the linear program $\mathcal{P}$ and the feasibility program Q of Figure 4.6 can be computed by our linear-OPT-gate.

Lemma 21. *Consider the linear program $\mathcal{P}$ and the feasibility program Q of Figure 4.6. An optimal solution to $\mathcal{P}$ can be computed by the linear-OPT-gate. Q is solvable, and a solution can be computed by the linear-OPT-gate.*

Proof. For the linear program $\mathcal{P}$, we observe that the feasible domain $[0,1]^n$ is nonempty and bounded. The gate inputs p_j appear only on the right-hand side of the constraints, and the subgradient of the objective function is linear, and hence can be given by a linear pseudo-circuit. For the feasibility program Q, there are no variables appearing in the left-hand side of the conditional constraints (see also Remark 23), and no gate inputs appearing on the right-hand side. This means that the linear-OPT-gate correctly computes the outcome of the program, assuming that the program is solvable. To argue solvability, notice that Q is of the form Q_{app} as defined in Section 4.3.2, and hence by Lemma 12 it suffices to argue that its feasibility graph G_Q is acyclic. In our case here, the vertices V_Q of G_Q are the goods, and an edge (j, j') exists if and only if $e_j < e_{j'}$. Clearly, G_Q is acyclic. □

Arguing optimality and market clearing. To conclude the proof, what is left to show is that a fixed point $(\mathbf{p}^*, \mathbf{q}^*)$ of F indeed corresponds to a competitive equilibrium. We argue that in the following lemma.

Lemma 22. *Let $(\mathbf{p}^*, \mathbf{q}^*)$ be a fixed point of F. Then the pair $(\mathbf{p}^*, \mathbf{x}^*)$, where $x^*_{ij} = q^*_{ij}/p^*_j$ is a competitive equilibrium of $\mathcal{M}$.*

Proof. By the form of the objective function of the linear program $\mathcal{P}$ of Figure 4.6, each consumer spends money only on goods with maximum BPB and therefore the allocation that she receives is only one of goods for which she has maximum BPB. Therefore, the consumer receives an optimal bundle which satisfies Condition 1 of the competitive equilibrium definition in Definition 4.6.1. The allocation quantities x^*_{ij} can straightforwardly be recovered from the values of q^*_{ij}.

What remains to show is that $\mathbf{p}^*$ is a vector of market-clearing prices. By the definition of e_j, this is equivalent to arguing that for all $j \in G$, we have that $e^*_j = 0$. We first argue that $\sum_{j\in G} e^*_j = 0$. Indeed:

$$\begin{aligned}\sum_{j\in G} e^*_j &= \sum_{j\in G}\left(\sum_{i\in N} q^*_{ij} - p^*_j\right) = \sum_{j\in G}\sum_{i\in N} q^*_{ij} - \sum_{j\in G} p^*_j = \sum_{j\in G}\sum_{i\in N} w_{ij}p^*_j - 1 \\ &= \sum_{j\in G} p^*_j \cdot \sum_{i\in N} w_{ij} - 1 = \sum_{j\in G} p^*_j - 1 = 0,\end{aligned}$$

where in the calculation above we used:

- in Equations 3 and 6, that $\sum_{j\in G} p^*_j = 1$, which is without loss of generality (see Remark 21),
- in Equation 5, that $\sum_{i\in N} w_{ij} = 1$, which is without loss of generality, and
- in Equation 3, that $\sum_{j\in G}\sum_{i\in N} q^*_{ij} = \sum_{j\in G}\sum_{i\in N} w_{ij}p^*_j$, which follows from the constraints of linear program $\mathcal{P}$ of Figure 4.6.

Define the set J to be the set of goods with minimum excess expenditure, i.e.,

$$J = \{j \in G : e^*_j \leq e^*_{j'}, \text{ for all } j' \in G\}.$$

It now suffices to show that $e^*_j \geq 0$ for all $j \in J$. Assume by contradiction that there exists some $j_1 \in J$ such that $e^*_{j_1} < 0$. By definition of the set J, we have that

$$\text{for all } j \in J, \text{ we have } e^*_j < 0. \tag{4.24}$$

Also, by the fact that $\sum_{j\in G} e^*_j = 0$ which we established above, it must be the case that there also exists $j_2 \in G \setminus J$ such that $e^*_{j_2} > 0$. In particular $G \setminus J \neq \emptyset$. Define

$$N' = \{i \in N : \text{ there exists } j \in G \setminus J \text{ such that } w_{ij} > 0\}.$$

In words, N' is the set of consumers with strictly positive endowment for some good which is not in the set J. We will argue that

there exists $i_0 \in N'$ and $j \in J$ such that $u_{i_0 j} > 0$.

To see this, assume by contradiction that for all $i \in N'$ and $j \in J$, it was the case that $u_{ij} = 0$. Note also that for any good $j' \in G \setminus J$, it holds that $w_{ij'} > 0$ only for consumers $i \in N'$. That would imply that N' and $G \setminus J$ would constitute a reduced market, contradicting Feasibility Condition 3.

Consider any good $j \in J$ and any good $j' \in G \setminus J$ such that $w_{i_0 j' > 0}$. By the definition of J, we have that $e^*_j < e^*_{j'}$. By the fact that $\mathbf{p}^*$ is a solution to the feasibility program Q of Figure 4.6, we have that $p^*_j \leq \varepsilon \cdot p^*_{j'}$. We have the following inequalities:

$$\sum_{g \in G} w_{i_0 g} \cdot p^*_g \geq w_{i_0 j'} \cdot p^*_{j'} \quad \text{and} \quad p^*_{j'} \geq \frac{m \cdot p^*_j}{w_{i_0 j'}}. \tag{4.25}$$

The first inequality above is trivial, since $j' \in G$. The second inequality follows since by the choice of ε,

$$p^*_{j'} \geq \frac{p^*_j}{\varepsilon} \geq \frac{p^*_j}{w_{i_0 j'}/m} = \frac{m \cdot p^*_j}{w_{i_0 j'}}.$$

Since j was chosen to be any good in J, the inequalities in (4.25) hold for j such that p^*_j is the maximum price amongst all goods in J. Therefore in (4.25) we can substitute p^*_j by $\max_{g \in J} p^*_g$ and we have

$$\sum_{g \in G} w_{i_0 g} \cdot p^*_g \geq w_{i_0 j'} \cdot p^*_{j'} \geq m \cdot \max_{g \in J} p^*_g \geq \sum_{g \in J} p^*_g. \tag{4.26}$$

Inequality 4.26 implies that consumer i_0 has enough budget to buy all the goods in J. At the same time, by the choice of ε, consumer i_0 prefers to buy quantities of good j rather than j', i.e., $\text{BPB}_{i_0}(j) > \text{BPB}_{i_0}(j')$. Since consumer i_0 only spends money on goods with maximum BPB, we have that

$$\sum_{g \in G} q^*_{i_0 g} = \sum_{g \in J} q^*_{i_0 g}. \tag{4.27}$$

From Inequality 4.26, Equation 4.27, and since $\mathbf{q}^*$ is a feasible solution to linear program $\mathcal{P}$ of Figure 4.6 (and hence satisfies the constraints $\sum_{g \in G} q^*_{i_0 g} = \sum_{g \in G} w_{i_0 g} \cdot p^*_g$), we have that

$$\sum_{g \in J} q^*_{i_0 g} \geq \sum_{g \in J} p^*_g.$$

This last inequality however implies that there must exist $j_0 \in J$ for which $q^*_{i_0 j_0} \geq p^*_{j_0}$ and therefore $e^*_{j_0} = \sum_{i \in N} q^*_{i j_0} - p^*_{j_0} \geq 0$, contradicting (4.24). This completes the proof. □

4.6.2 Arrow-Debreu Markets with Linear Utilities and Productions

We now move on to the next fundamental variant of the Arrow-Debreu market model, that of markets with *linear utilities* as well as *linear productions*. Our results about exchange markets in Section 4.6.1 were used to demonstrate the application of our technique to one of the simplest variants of the main market model. The results of this section can be seen as extending this exposition to the case where the markets also have linear production functions. This is rather informative for the reader, as it illustrates the approach that will be used for the most general market setting that we prove PPAD-membership for, which we present in Section 4.6.3.

Our main theorem in this section is the following:

Theorem 4.6.2. *Computing a competitive equilibrium in an Arrow-Debreu market with linear utilities and linear productions is in PPAD.*

We provide the main definitions for Arrow-Debreu markets with linear utilities and linear productions below. To ensure that the section is self-contained, we have elected to fully define the setting, rather than to only highlight the changes to the definitions of exchange markets of Section 4.6.1. Still, to avoid unnecessary repetition, in the preprocessing and setup steps for the proof, we do refer to the appropriate arguments presented in Section 4.6.1.

Markets with Production. In an Arrow-Debreu market with production $\mathcal{M}$, we have a set N of consumers, a set G of infinitely divisible goods, and a set F of firms. Let $n = |N|$, $m = |G|$, and $\ell = |F|$. We will typically use index i to refer to consumers, j or g to refer to goods and f to refer to firms. Each consumer brings an endowment $w_i = (w_{i1}, \dots, w_{im})$ to the market, with $w_{ij} \geq 0$ for all $i \in N$ and $j \in G$. We may assume without loss of generality that for every good j, we have $\sum_{i \in N} w_{ij} = 1$, i.e., that the total endowment of each good is 1. We will use $\mathbf{x}_i = (x_{i1}, \dots, x_{im})$ to denote the vector of quantities of goods allocated to consumer $i \in N$ in $\mathcal{M}$, and we will call it the *bundle* of consumer i. Let $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$ be the vector of such bundles. We will use $\mathbf{p} = (p_1, \dots, p_m)$ to denote the vector of *prices* in $\mathcal{M}$, one for each good $j \in G$. Prices are non-negative, so $p_j \geq 0$ for all $j \in G$. Given a vector of prices $\mathbf{p}$, the *budget* of consumer $i \in N$ is defined as $\sum_{j \in G} w_{ij} p_j$; intuitively, this is the amount of money that the consumer acquires by selling her endowment at prices $\mathbf{p}$.

Utility Functions. Every consumer has a utility function $u_i : \mathbb{R}^m_{\geq 0} \to \mathbb{R}_{\geq 0}$ mapping a bundle $\mathbf{x}_i$ to a non-negative real number. In this section, these utilities are *linear*, i.e., every consumer $i \in N$ has a utility u_{ij} for every good $j \in G$, and her utility for the bundle $\mathbf{x}_i$ is $u_i(\mathbf{x}_i) = \sum_{j \in G} u_{ij} x_{ij}$, where u_{ij} is the value of consumer i for good j.

Firm Shares. Each consumer $i \in N$ has a share $\theta_{if} \in [0, 1]$ of the profit of each firm $f \in F$. We assume that the profits are entirely shared among the consumers, i.e., for every firm $f \in F$, we have that $\sum_{i \in N} \theta_{if} = 1$.

Production Functions and Conversion Rates. Each firm $f \in F$ produces a set of goods using a set of goods as raw material. Following Garg and Vazirani [112], for simplicity we will assume without loss of generality that each firm produces a single good. We may also assume without loss of generality that each firm uses a single good as raw material for the production. This because the production functions that we will consider are separable, and hence a firm using/producing multiple goods can be split into multiple firms using/producing single goods, with the shares of the agents' being duplicated (see also [112]). Given the above we will let:

- g_f^{out} be the good produced by firm $f \in F$; we will refer to this as the ***output good***,
- g_f^{in} be the good used as raw material by firm $f \in F$; we will refer to this as the ***input good***.

For every firm $f \in F$ there is a function P_f which determines the firm's ability to produce units of the output good g_f^{out} as a function of quantities of the input good g_f^{in}. In this section, we will assume that these functions are linear, of the form $P_f(y) = c_f \cdot y$, where $c_f \geq 0$ is a fixed *conversion rate* for firm f, specifying that the firm can use y units of good g_f^{in} to produce $c_f \cdot y$ units of good g_f^{out}. Given a conversion rate c_f, and prices $p_{g_f^{\text{out}}}$ and $p_{g_f^{\text{in}}}$ for the goods g_f^{out} and g_f^{in} respectively, the *profit* of f from using y units of g_f^{in} to produce $c_f \cdot y$ units of g_f^{out} is defined as $p_{g_f^{\text{out}}} \cdot c_f \cdot y - p_{g_f^{\text{in}}} \cdot y$.

Competitive Equilibrium. We are now ready to define the notion of a competitive equilibrium in markets with production.

Definition 4.6.2 (Competitive Equilibrium - Markets with Production)**.** A competitive equilibrium of an Arrow-Debreu market with linear utilities and linear production functions is a triple $(\mathbf{p}^*, \mathbf{x}^*, \mathbf{y}^*)$ consisting of non-negative prices $\mathbf{p}^*$, non-negative bundles $\mathbf{x}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*)$ and non-negative amounts of input goods $\mathbf{y}^* = (y_1^*, \ldots, y_\ell^*)$, such that

1. $p^*_{g_f^{\text{out}}} \cdot c_f \cdot y_f^* - p^*_{g_f^{\text{in}}} \cdot y_f^*$ is maximized, for any firm $f \in F$. ***(firm profit maximization)***
2. $u_i(\mathbf{x}_i)$ is maximized for every consumer $i \in N$, subject to $\sum_{j \in G} p_j^* \cdot x_{ij} \leq \sum_{j \in G} p_j^* \cdot w_{ij} + \sum_{f \in F} \theta_{if} \cdot (p^*_{g_f^{\text{out}}} \cdot c_f \cdot y_f^* - p^*_{g_f^{\text{in}}} \cdot y_f^*)$. ***(bundle optimality)***
3. $z_j^* \leq 0$, and $z_j^* p_j^* = 0$, where $z_j^* = \sum_{i \in N} x_{ij}^* + \sum_{f \in F:\, g_f^{\text{in}} = j} y_f^* - \sum_{f \in F:\, g_f^{\text{out}} = j} c_f \cdot y_f^* - 1$, for every good $j \in G$. ***(market clearing)***

Condition 1 requires that at the chosen set of prices $\mathbf{p}^*$, each firm maximizes its profit, given its production functions. Condition 2 requires that at the chosen set of prices $\mathbf{p}^*$, each consumer maximizes her utility subject to their budget constraints, where the budget consists of the amount earned from selling all the consumer's endowments $\sum_{j\in G} p_j^* w_{ij}$ and the profit share $\sum_{f\in F} \theta_{if} \cdot (p^*_{g_f^{\text{out}}} \cdot c_f \cdot y_f^* - p^*_{g_f^{\text{in}}} \cdot y_f^*)$ of the consumer from the production of the firms. Finally, Condition 3 is the market clearing condition, which requires that the total consumption of each good is at most the total production plus the total endowment of the consumers, and supply equals demand for all goods which are not priced at 0. As we detail later on in the section, we may in fact assume without loss of generality that in any competitive equilibrium *all the prices are positive*, and hence Condition 3 reduces to $z_j^* = 0$ for all $j \in G$. Note that in Condition 3 we have used that $\sum_{\in N} w_{ij} = 1$ for each good $j \in G$.

Sufficiency Conditions. A competitive equilibrium as defined above exists for every market $\mathcal{M}$, under some sufficiency conditions. The weakest known such conditions are the ones provided by Maxfield [167] (see also [112]), which generalize the sufficiency conditions of Eaves [80] that we used in Section 4.6.1. We use the following conditions, which are very close to that of Maxfield [167].

1. For every good $j \in G$, there exists some consumer $i \in N$ that values the good positively, i.e., $u_{ij} > 0$.

2. For every consumer $i \in N$, there exists some good $j \in G$ that the consumer endows in a positive amount, i.e., $w_{ij} > 0$.

3. Consider a graph $\mathcal{G}_F(\mathcal{M})$ in which the nodes are the goods, and an edge (j, j') has weight
$$\alpha_{jj'} = \max_{f\in F:\ g_f^{\text{out}}=j', g_f^{\text{in}}=j} c_f,$$
i.e., j' can be produced from j at conversion rate $\alpha_{jj'}$ by some firm $f \in F$. Then, for any cycle $C = (g_0, g_1), (g_1, g_2) \dots, (g_{k-2}, g_{k-1})$ of $G_F(\mathcal{M})$ the product of the weights of the edges is less than 1, i.e., $\prod_{e\in C} \alpha_e < 1$.

   This condition is known as the ***no production out of nothing and no vacuous production*** condition. Indeed, if $\prod_{e\in C} \alpha_e > 1$, then it would be possible to increase the quantity of some good, without decreasing the quantity of any other good. The case of $\prod_{e\in C} \alpha_e = 1$ refers to the case of vacuous production, which is also disallowed in our model.[14]

4. Consider the *economy graph* $\mathcal{G}_E(\mathcal{M})$ of the market $\mathcal{M}$ in which the nodes are the consumers and the firms, and

   - there is an edge (i, i') between consumer/firm-node i and consumer-node i' if i endows/produces a good j for which $u_{i'j} > 0$,

[14]Maxfield [167] technically allows for vacuous production. Vacuous production is disallowed in the conditions imposed in the original setting of Arrow and Debreu [8] and is also disallowed by Garg and Vazirani [112].

- there is an edge (i, f) between consumer/firm-node i and firm-node f, if i endows/produces the raw good g_f^{in} that f uses for production.

Then, $\mathcal{G}_E(\mathcal{M})$ contains a strongly connected component containing all the consumer-nodes. This condition generalizes that of the market non-reducibility condition that we used in Section 4.6.1.

Remark 24 (Bounds on Production)**.** Condition 3 above imposes a bound on the total amount of a production of any firm $f \in F$ in a competitive equilibrium. Note that the production starts from finite endowments $\sum_{i \in N} w_{ij} = 1$ for all $j \in G$. Since no firm operates at a loss at an equilibrium, any cycle of production would violate Condition Item 3. Since there are no such cycles, production can take place along chains. The longest such chain is obviously bounded by m, and the maximum production of any firm can be bounded by some sufficienctly large global constant $L^c =$ (e.g., some constant such that $L^c \geq m^m(\max_f c_f + 1)^n$. See also [112] for a very similar argument. Looking ahead, this will allow us to impose "loose" upper bounds on the production and consumption in the linear programs that we will devise, without compromising the existence of a competitive equilibrium.

Optimality and bang-per-buck (BPB). Similarly to Section 4.6.1, the optimal bundles of Condition 2 in Definition 4.6.2 are characterized by their *bang-per-buck (BPB)*. Given consumer $i \in N$ and prices $\mathbf{p}$, the BPB of a good $j \in G$ is defined as $\text{BPB}_i(j) = \frac{u_{ij}}{p_j}$. An optimal bundle only contains non-zero quantities of goods for which the BPB is maximum, i.e., $j \in \arg\max_{j \in G} \text{BPB}_i(j)$.

Bounds on the prices. Again, similarly to Section 4.6.1, we may assume without loss of generality that all the prices are strictly positive, i.e., $p_j > 0$ for all goods $j \in G$. Indeed, not that for any good j, there is some consumer i with $u_{ij} > 0$. Then its price p_j cannot be 0 in any equilibrium. If j is not being produced, then the demand of consumer i cannot be satisfied. If its being produced, it has to be produced using an input good of 0 price as well, for the production to be profitable. This in fact implies that it has to be produced along a chain of goods with 0 price, contradicting the market clearing condition. Very similar arguments have been made in the related literature, e.g., see [112, 119]. Note that by these assumptions, the quantity $\text{BPB}_i(j)$ is well defined for every $j \in G$.

Remark 25 (Normalized Prices)**.** Given that $p_j > 0$ for all $j \in G$, we can normalize the prices to sum to 1 without loss of generality, i.e., we may assume that for every good $j \in G$, we have that $\sum_{j \in G} p_j = 1$.

Again, in a similar manner to Section 4.6.1, we will use a parameter ε to capture the fact that if the price p_j for a good $j \in G$ is sufficiently smaller than the price $p_{j'}$ for a good $j' \in G$, then $\text{BPB}(j) > \text{BPB}(j')$. Specifically, we can compute $\varepsilon > 0$ such that

$$\text{If } p_j \leq \varepsilon \cdot p_{j'} \text{ and } u_{ij} > 0 \text{ then } \text{BPB}_i(j) > \text{BPB}_i(j').$$

Linear Program $\mathcal{P}_1$

$$\begin{aligned}
&\text{minimize} \sum_{i\in N}\sum_{j\in G}\frac{p_j}{u_{ij}}q_{ij}\\
&\text{subject to} \sum_{j\in G} q_{ij} = \sum_{j\in G} w_{ij}p_j + \sum_{f\in F}\theta_{if}\cdot(r_f - s_f)\\
&\qquad 0 \le q_{ij} \le (C\cdot L+1),\ \forall j\in G
\end{aligned}$$

Linear Program $\mathcal{P}_2$

$$\begin{aligned}
&\text{maximize} \quad (c_f\cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}})\cdot s_f\\
&\text{subject to} \quad 0 \le s_f \le L\cdot p_{g_f^{\text{in}}}
\end{aligned}$$

Figure 4.7: The linear programs $\mathcal{P}_1$ and $\mathcal{P}_2$ used to recover optimal bundles (left), and optimal production (right) respectively.

Additionally, we can pick ε to be sufficiently small such that $\varepsilon < \frac{w_{ij}}{m}$ for all $i \in N$ and all $j \in G$. Given ε, we will impose a stricter lower bound on the prices, which will be useful later on: in particular, we will assume that for all $j \in G$, $p_j \ge \frac{\varepsilon^m}{m}$.

Preprocessing

The high-level approach will again be to define a function F from a convex compact domain D to itself, and show that a competitive equilibrium of $\mathcal{M}$ can be recovered from a fixed point of F. We will need to show that F can be computed by a linear arithmetic circuit and hence, similarly to Section 4.6.1, we cannot work directly with the allocations $\mathbf{x}$ and the quantities of input goods $\mathbf{y}$ as in Definition 4.6.2, as those would need to be multiplied with the the prices $\mathbf{p}$, which will also be inputs to the circuit. To circumvent this, we will again apply a change of variables very similar to *Gale's substitution* (see Remark 22) and work with the expenditure variables as we did in Section 4.6.1. In more detail, we will need such variables for the expenditure of the consumers as before, but we will also need variables for the expenditure and the revenue of the firms in production. The same variable change was used by Garg and Vazirani [112].

We will then make use of our linear-OPT-gate to obtain the equilibrium quantities as outcomes of linear programs and a feasibility program. In particular, linear programs $\mathcal{P}_1$ and $\mathcal{P}_1$ will be used to compute the optimal consumer and firm expenditures respectively, and from those the optimal bundles $\mathbf{x}$ and optimal quantities of input goods $\mathbf{y}$ will be recovered. The prices will be the outcome of a feasibility program Q.

Consumer expenditure. For the consumers, we will use the standard change of variables that we used in Section 4.6.1 for exchange markets. Namely, we will let $q_{ij} = x_{ij}p_j$ be the *expenditure* of consumer i on good j, i.e., how much money the consumer spends on the good given an allocation x_{ij} and a price p_j.

Firm expenditure. For the firms, we will use a similar substitution for the case of production. Namely, we will let

- $s_f = p_{g_f^{\text{in}}} \cdot y_f$ be the *expenditure* of firm f on its input good g_f^{in}, i.e., the amount of money spend on y_f units of the input good.
- $r_f = p_{g_f^{\text{out}}} \cdot c_f \cdot y_f$ be the *revenue* of firm f from its output good g_f^{out}, i.e., the amount of money earned from $c_f \cdot y_f$ units of the output good.[15]

By definition, the profit of firm $f \in F$ is then exactly $r_f - s_f$.

The program $\mathcal{P}_1$ for optimal consumer expenditure. We can now write the linear program $\mathcal{P}_1$, the solution of which will give us the optimal expenditures q_{ij}^* for the consumers; see the left-hand side of Figure 4.7. It now becomes evident why we need to apply the aforementioned substitution: without it the quantity $\sum_{f \in F} \theta_{if} \cdot (p_{g_f^{\text{out}}} \cdot c_f \cdot y_f - p_{g_f^{\text{in}}} \cdot y_f)$ would appear in the constraints of the linear program $\mathcal{P}_1$. The constants $p_{g_f^{\text{in}}}$, $p_{g_f^{\text{out}}}$ and y_f are gate inputs for the linear program, and such a constraint cannot be handled by our linear-OPT-gate. With the gate inputs r_f and s_f instead, the constraint becomes linear and thus respects the conditions of the linear-OPT-gate. Note that the linear program $\mathcal{P}_1$ is parameterized by two constants L and C. Intuitively, $C \cdot L + 1$ is an upper bound on the amount of money that consumer i can spend on good j. Recall that there is a global upper bound L^c on the total possible production, and as a result there are global bounds on the total possible expenditure for both production and consumption. We will set L and C to be sufficiently large to not constrain these global upper bound; the precise bounds are established in the proofs of Claim 7 and Claim 8.

Similarly to Section 4.6.1, in an optimal solution, the consumer is again only spending on goods with minimum ratio $\frac{p_j}{u_{ij}}$, i.e., with maximum BPB, therefore purchasing an optimal bundle. At the same time, the subgradient of this objective function can be computed by a linear pseudo-circuit.

The program $\mathcal{P}_2$ for optimal firm expenditure. We are now ready to define the linear program for each firm $f \in F$, the optimal solution of which will be the optimal expenditure s_f^* of firm f on the amount y_f^* of the input good that the firm uses for production; see the right-hand side of Figure 4.7. The linear program $\mathcal{P}_2$ is also parameterized by the constant L, which we mentioned above; intuitively, L imposes an upper bound on the quantity of the input good that firm f can use, and will be large enough to not constrain the global upper bound L^c on production.

Remark 26 ($\mathcal{P}_2$ ensures optimal production)**.** At first glance, the objective function of linear program $\mathcal{P}_2$ might seem a bit unintuitive: why are we multiplying the expenditure with the profit? How do we guarantee that in the competitive equilibrium, each firm maximizes its profit? To provide some intuition, we remark that linear program $\mathcal{P}_2$ could equivalently be substituted by the following feasibility program $Q(\mathcal{P}_2)$:

[15]We remark that Garg and Vazirani [112] used r_f to refer to the expenditure (spending) and s_f to refer to the revenue. We have switched those so that the variables names are more indicative of the quantities that they represent.

Feasibility Program $Q(\mathcal{P}_2)$

$$c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} > 0 \Rightarrow s_f = L \cdot p_{g_f^{\text{in}}}$$
$$c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} < 0 \Rightarrow s_f = 0$$
$$0 \leq s_f \leq L \cdot p_{g_f^{\text{in}}}$$

Indeed, linear program $\mathcal{P}_2$ has a very simple form, and hence in an optimal solution

- the firm spends as much as possible on the input good (i.e., s_f is maximum), when it is profitable to produce, (i.e., when $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} > 0$),
- the firm spends as little as possible on the input good (i.e., s_f is 0), when it is not profitable to produce, (i.e., when $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} < 0$),
- the firm spends any feasible amount when producing or not producing yield the same profit (i.e., when $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} = 0$).

We could have in fact used $Q(\mathcal{P}_2)$ rather than $\mathcal{P}_2$, as it is a valid feasibility program that can be solved using the linear-OPT-gate. However, we elected to go the the linear program $\mathcal{P}_2$ instead, as it sets up the machinery that we will use in Section 4.6.3, where the corresponding linear program will be more involved and cannot simply be substituted by a feasibility program as above.

We remark that in a competitive equilibrium, we will establish that $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} \leq 0$, and hence the firm will never be required to use exactly L units of g_f^{in} for production.

Remark 27 (Calculating r_f from s_f)**.** Looking ahead, the terms r_f which appear in the constraints of linear program $\mathcal{P}_1$ (and also in the feasibility program Q, see below), will not be inputs to the function F that we will construct to compute a competitive equilibrium. These will need to be recovered from the values of s_f, or more precisely, the terms themselves will need to be substituted with expressions of the term s_f. This can be done as follows:

$$r_f = s_f + \max\{0, c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}}\} \cdot L \tag{4.28}$$

As we will only be concerned with the fixed point behavior of our function F, this equation needs to correctly capture the revenue of firm f at a competitive equilibrium. Indeed, given our discussion above, we have that:

- When $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} < 0$, the firm does not spend any amount on producing and as a result it does not produce anything. In that case $r_f^* = s_f^* = 0$.
- When $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} > 0$, the firm spends an amount of $s_f^* = L \cdot p_{g_f^{\text{in}}}$ for L units of the input good and obtains a revenue of $r_f^* = L \cdot c_f \cdot p_{g_f^{\text{out}}}$.

$$\boxed{\begin{array}{l} \underline{\text{Feasibility Program } Q} \\ \\ e_j < e_{j'} \Rightarrow p_j \le \varepsilon \cdot p_{j'},\ \forall j, j' \in G \\ p_j \ge \dfrac{\varepsilon^m}{m},\ \forall j \in G \\ \displaystyle\sum_{j \in G} p_j = 1 \end{array}}$$

Figure 4.8: The feasibility program Q used to find market-clearing prices.

- When $c_f \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}} = 0$, the firm can spend any amount s_f^* for production, and we have $r_f^* = s_f^*$ by definition.

In each case, Equation (4.28) correctly computes the value of r_f^* from s_f^* at a competitive equilibrium.

Excess expenditure and the program Q. Similarly to Section 4.6.1, the equilibrium prices $\mathbf{p}^*$ will be obtained via a feasibility program Q. Before we define the program, we introduce the notion of excess expenditure; the definition is very similar to the one we used in Section 4.6.1, except that it now takes into account the expenditure/revenue due to production. The *excess expenditure* e_j of a good $j \in G$ is defined as the difference between the total expenditure of all consumers $i \in N$ and firms $f \in F$ for that good and the price of the good plus the total revenue of firms $f \in F$ from that good, i.e.,

$$e_j = \sum_{i \in N} q_{ij} + \sum_{f \in F:\ g_f^{\text{in}} = j} s_f - p_j - \sum_{f \in F:\ g_f^{\text{out}} = j} r_f.$$

Then, at equilibrium prices p_j^* we will have market clearing, i.e., $e_j^* = 0$.

We are now ready to define our feasibility program Q; see Figure 4.8. The equilibrium prices will be obtained as the output $\mathbf{p}^*$ of Q.

Membership in PPAD: The proof of Theorem 4.6.2.

We will again develop the proof in three steps, namely (a) construction of the function F and arguing that it can be represented by a linear arithmetic circuit containing linear-OPT-gates, (b) showing that the linear-OPT-gate can compute all the necessary components, and (c) arguing that a competitive equilibrium can be recovered from a fixed point of F.

The function F. Given the above, we will define $F : D \to D$ with domain

$$D = \left\{ p \in \Delta^{m-1} : \forall j \in G, p_j \ge \frac{\varepsilon^m}{m} \right\} \times [0, C \cdot L + 1]^{nm} \times [0, L]^{\ell}.$$

An input to F is a triple $(\bar{\mathbf{p}},\bar{\mathbf{q}},\bar{\mathbf{s}})$ of prices, consumer expenditures and firm expenditures, where $\bar{\mathbf{q}}=(\bar{q}_{ij})_{i\in N,j\in G}$ and $\bar{\mathbf{s}}=(\bar{s}_1,\dots \bar{s}_\ell)$, and the output is another such triple $(\mathbf{p}^*,\mathbf{q}^*,\mathbf{s}^*)$. The domain D is the one above since $\sum_{j\in G}\bar{p}_j=1$ (see Remark 25), and by the upper bounds of $C\cdot L+1$ and L set on the consumer expenditures in linear program $\mathcal{P}_1$ and the firm expenditures in linear program $\mathcal{P}_2$ respectively.

Computation by the linear-OPT-gate. Next, we argue that the solutions to the linear programs $\mathcal{P}_1$ and $\mathcal{P}_2$ and the feasibility program Q of Figure 4.6 can be computed by our linear-OPT-gate.

Lemma 23. *Consider the linear programs $\mathcal{P}_1$, $\mathcal{P}_2$ of Figure 4.7, and the feasibility program Q of Figure 4.8. An optimal solution to $\mathcal{P}_1$ and $\mathcal{P}_2$ can be computed by the linear-OPT-gate. Q is solvable, and a solution can be computed by the linear-OPT-gate.*

Proof. For the linear program $\mathcal{P}_1$, the feasible domain $[0,1]^n$ is non-empty and bounded. The gate inputs p_j and s_f appear only on the right-hand side of the constraints, and the subgradient of the objective function is linear, and hence can be given by a linear pseudo-circuit. For the linear program $\mathcal{P}_2$, the feasible domain $[0,L]$ is non-empty and bounded, and the gate inputs $p_{g_f^{\text{in}}}$ only appear on the right-hand side of the constraints. The subgradient of the objective function is linear, and hence can be given by a linear pseudo-circuit. For the feasibility program Q, the arguments that it is of the correct form and that it is solvable are identical to those of Lemma 21, noting the updated definition of the expenditure e_j. □

Arguing optimality and market clearing. To conclude the proof, what is left to show is that a fixed point $(\mathbf{p}^*,\mathbf{q}^*,\mathbf{s}^*)$ of F indeed corresponds to a competitive equilibrium. We argue that in the following lemma.

Lemma 24. *Let $(\mathbf{p}^*,\mathbf{q}^*,\mathbf{s}^*)$ be a fixed point of F. Then the triple $(\mathbf{p}^*,\mathbf{x}^*,\mathbf{y}^*)$, where $x^*_{ij}=q^*_{ij}/p^*_j$ and $y^*_f=s^*_f/p^*_{g_f^{in}}$ is a competitive equilibrium of $\mathcal{M}$.*

Proof. By the form of the objective function of the linear program $\mathcal{P}_1$ of Figure 4.7, each consumer spends money only on goods with maximum BPB and therefore the allocation that she receives is only one of goods for which she has maximum BPB. Therefore, the consumer receives an optimal bundle which satisfies Condition 2 of definition in Definition 4.6.2. The allocation quantities x^*_{ij} can straightforwardly be recovered from the values of q^*_{ij}. As we discussed earlier in Remark 26, an optimal solution to linear program $\mathcal{P}_2$ also results in the optimal expenditure for the firms.

What remains to show is that $\mathbf{p}^*$ is a vector of market-clearing prices. By the definition of e_j, this is equivalent to arguing that for all $j\in G$, we have that $e^*_j=0$. We first argue that $\sum_{j\in G} e^*_j = 0$. Indeed:

$$\begin{aligned}
\sum_{j\in G} e^*_j &= \sum_{j\in G}\left(\sum_{i\in N} q^*_{ij} - p^*_j + \sum_{f\in F:\, g^{\text{in}}_f = j} s^*_f - \sum_{f\in F:\, g^{\text{out}}_f = j} r^*_f\right)\\
&= \sum_{i\in N}\sum_{j\in G} q^*_{ij} - \sum_{j\in G} p^*_j + \sum_{f\in F}(s^*_f - r^*_f)\\
&= \sum_{j\in G}\sum_{i\in N} w_{ij}p^*_j - 1 + \sum_{i\in N}\sum_{f\in F}\theta_{if}\cdot(r^*_f - s^*_f) + \sum_{f\in F}(s^*_f - r^*_f)\\
&= \sum_{j\in G} p^*_j \cdot \sum_{i\in N} w_{ij} - 1 + \sum_{f\in F}(r^*_f - s^*_f)\cdot\sum_{i\in N}\theta_{if}\cdot\sum_{f\in F}(s^*_f - r^*_f)\\
&= \sum_{i\in N} w_{ij} - 1 + \sum_{f\in F}(r^*_f - s^*_f) + \sum_{f\in F}(s^*_f - r^*_f)\\
&= 0,
\end{aligned}$$

where in the calculations above we used:

- in Equation 2, that $\sum_{j\in G}\sum_{f\in F:\, g^{\text{in}}_f = j} s^*_f = \sum_{f\in F} s^*_j$ and that $\sum_{j\in G}\sum_{f\in F:\, g^{\text{out}}_f = j} r^*_f = \sum_{f\in F} r^*_f$,
- in Equation 3, that $\sum_{i\in N}\sum_{j\in G} q^*_{ij} = \sum_{i\in N}\sum_{j\in G} w_{ij}p^*_j + \sum_{i\in N}\sum_{f\in F}\theta_{if}(r^*_f - s^*_f)$, which follows from the constraints of linear program $\mathcal{P}_2$ of Figure 4.7,
- in Equations 3 and 5, that $\sum_{j\in G} p^*_j = 1$, which is without loss of generality, see Remark 25,
- in Equation 5, that $\sum_{i\in N}\theta_{if} = 1$ for every $f \in F$, by definition, and
- in Equation 6, that $\sum_{i\in N} w_{ij} = 1$, which is without loss of generality.

From the above, it suffices to prove that $e^*_j \geq 0$, for all $j \in G$. Assume by contradiction that there exists some $j_1 \in G$ such that $e^*_{j_1} < 0$; we will obtain a contradiction to the strong connectivity of the economy graph of the market. We define the following three sets:

- $J = \{j \in G\colon e^*_j \leq e^*_{j'},$ for all $j' \in G\}$. In other words, J is the set of goods with minimum excess expenditure.
- $N_J = \{i \in N\colon$ there exists $j \in J$ such that $u_{ij} > 0\}$. In other words, N_J contains the set of consumers that value at least one item in J positively.
- $F_J = \{f \in F\colon g^{\text{in}}_f \in J\}$. In other words, F_J contains the set of firms for which the input good is in J.

Since $e^*_{j_1} < 0$, by the definition of J we know that

$$\text{for all } j \in J, \text{ we have } e^*_j < 0. \tag{4.29}$$

Also, since $e^*_{j_1} < 0$, and $\sum_{j \in G} e^*_j = 0$, there must exist some other good j_2 such that $e^*_{j_2} > 0$. In particular, that implies that J is a strict subset of G, i.e., $J \subsetneq G$. We state and prove the following claim.

Claim 7. *For all $i \in N_J$ and all $j' \in G \setminus J$, it holds that $w_{ij'} = 0$.*

Proof. Assume by contradiction that there exists some consumer $i_0 \in N_J$ and some good $j' \in G \setminus J$ such that $w_{i_0 j'} > 0$. We will show that this implies that there exists some $j_\ell \in J$ with $e^*_{j_\ell} \geq 0$, contradicting Statement (4.29). Since $\mathbf{p}^*$ is a solution to feasibility program Q of Figure 4.8, we have that $p^*_j \leq \varepsilon \cdot p^*_{j'}$, for every $j \in J$. Hence, by choosing ε to be sufficiently small, we may lower-bound the budget of consumer i by

$$w_{i_0 j'} \cdot p^*_{j'} \geq \frac{w_{i_0 j'}}{\varepsilon \cdot m} \cdot \sum_{j \in J} p^*_j \geq (C \cdot L + 1) \sum_{j \in J} p^*_j$$

Since $i_0 \in N_J$, by the choice of ε, there is at least one good $g \in J$ such that $\mathrm{BPB}_{i_0}(g) > \mathrm{BPB}_{i_0}(g')$, for all $g' \in G \setminus J$, i.e., the consumer prefers to buy quantities of good g rather than any good which is not in the set J. Let $j \in J$ be a good with maximum BPB for consumer i_0. Since $q^*_{i_0 j}$ is an optimal solution to linear program $\mathcal{P}_1$ of Figure 4.7, it should be that $q^*_{i_0 j} = C \cdot L + 1$, which implies that $q^*_{i_0 j} \geq (C \cdot L + 1) p^*_j$. By substituting that into the definition of the excess expenditure e^*_j, we obtain:

$$\begin{aligned} e^*_j &= \sum_{i \in N} q^*_{ij} + \sum_{f \in F:\, g^{\text{in}}_f = j} s^*_f - p^*_j - \sum_{f \in F:\, g^{\text{out}}_f = j} r^*_f \\ &\geq q^*_{i_0 j} + \sum_{f \in F:\, g^{\text{in}}_f = j} s^*_f - p^*_j - \sum_{f \in F:\, g^{\text{out}}_f = j} r^*_f \\ &\geq (C \cdot L) p^*_j - \sum_{f \in F:\, g^{\text{out}}_f = j} r^*_f \end{aligned}$$

Therefore, it suffices to show that $\sum_{f \in F:\, g^{\text{out}}_f = j} r^*_f \leq (C \cdot L) p^*_j$; in that case we will have $e^*_j \geq 0$ and will obtain a contradiction, by setting $j = j_\ell$. Now consider any firm $f \in F$. Referencing also Remark 26, note that

- if $c_f \cdot p^*_{g^{\text{out}}_f} < p^*_{g^{\text{in}}_f}$, then the firm does not produce anything, and we have $r^*_f = 0$,
- if $c_f \cdot p^*_{g^{\text{out}}_f} > p^*_{g^{\text{in}}_f}$, the firm produces L units of g^{out}_f and we have $r^*_f = L \cdot c_f \cdot p_{g^{\text{out}}_f}$,
- if $c_f \cdot p^*_{g^{\text{out}}_f} > p^*_{g^{\text{in}}_f}$, the firm may produce any amount and we have $r^*_f = s^*_f \leq L \cdot c_f \cdot p_{g^{\text{out}}_f}$, since s^*_f is a feasible solution to linear program $\mathcal{P}_2$ of Figure 4.7.

By letting $C \geq |F| \cdot \max_{f \in F} c_f$, it follows that $\sum_{f \in F:\, g_f^{\text{out}} = j} r_f^* \leq (C \cdot L) p_j^*$ and we obtain a contradiction. □

Claim 7 establishes that a consumer in the set N_J cannot endow any good that is not in the set J in a positive quantity. Since $J \subsetneq G$, this implies that $N_J \subsetneq N$, since every good is positively endowed by some consumer $i \in N$. Now consider any consumer $i \in N_J$ and any consumer $i' \in N \setminus N_J$ or any firm $f \in F \setminus F_J$. Claim 7 implies that there cannot be an edge (i, i') or an edge (i, f) in the economy graph of the market. Indeed, for (i, i'), notice that by the claim consumer i only endows positively goods that are in J, but consumer i' has value 0 for them by virtue of being in $N \setminus N_J$. Similarly, for (i, f), f uses good $g_f^{\text{in}} \in G \setminus J$ by definition, but consumer i only positively endows goods in J.

Next, we state and prove the following claim.

Claim 8. *For any $f \in F_J$, it holds that $g_f^{out} \in J$.*

Proof. Assume by contradiction that there exists some firm $f_0 \in F_J$ with $g_{f_0}^{\text{out}} \notin J$. For convenience, let $j = g_{f_0}^{\text{out}}$ be that good, and let $j_0 = g_{f_0}^{\text{in}}$ be the firm's input good. Note that by definition of F_J, we have that $j_0 \in J$. By Statement (4.29), we have that $e_{j_0}^* < 0$. Since $\mathbf{p}^*$ is a solution to feasibility program Q of Figure 4.8, we have that $p_{j_0}^* \leq \varepsilon \cdot p_j^*$. The value of ε can be chosen sufficiently small for this to imply that $c_{f_0} \cdot p_j^* - p_{j_0}^* > 0$. Since this is the multiplier of the variable s_{f_0} in the objective function of linear program $\mathcal{P}_2$, the optimality of $s_{f_0}^*$ as a solution to the linear program implies that $s_{f_0}^* = L \cdot p_{j_0}^*$. Using this we may bound the expenditure $e_{j_0}^*$ as follows:

$$\begin{aligned} e_{j_0}^* &\geq L \cdot p_{j_0}^* - \sum_{f \in F:\, g_f^{\text{out}} = j_0} r_f^* - p_{j_0}^* \\ &= (L-1) p_{j_0}^* - \sum_{f \in F:\, g_f^{\text{out}} = j_0} r_f^* \end{aligned}$$

where in the calculations above we used the obvious fact that $\sum_{f \in F:\, g_f^{\text{in}} = j} s_f^* \geq s_{f_0}$.

Obviously, if good j_0 is not being produced by any firm $f \in F$, we can simply set $L \geq 1$ and obtain that $e_{j_0}^* \geq 0$, a contradiction, since $r_f^* = 0$ for all firms that have j_0 as their output good. Now consider the sequence of goods $j_0, j_1, \ldots, j_k$, where good $j_{\ell-1}$ is being produced (in positive quantity) by some firm f_ℓ using j_ℓ as the input good. Note that by the discussion above, we may assume that $k \geq 1$. Additionally, by the "no production out of nothing and no vacuous production" feasibility condition of the market $\mathcal{M}$, it also holds that $k \leq m$, as otherwise the graph $G_F(\mathcal{M})$ would have a cycle in which the product of the weight of the edges would be at least 1.

Finally, we remark that for each $\ell \in \{0, 1, \ldots, k\}$, we have that $j_\ell \in J$. To see this, observe that from the conditional constraints of feasibility program Q of Figure 4.8, the price for any good in J is at least ε times smaller than the price of any good in $G \setminus J$. By taking ε to be small enough, we can guarantee that if a firm f is producing a good $g_f^{\text{out}} \in J$, then it must use a good $g_f^{\text{in}} \in J$ as input, otherwise s_f^* would not be an optimal expenditure. Since $j_0 \in J$ by the definition of F_J, it must hold that $j_\ell \in J$ for all $\ell \in \{0, 1, \dots, k\}$.

We will argue by induction that for $1 \le \ell' \le k$, we have:

$$r^*_{f_\ell} \ge \left(\frac{L-1}{|F|^\ell \cdot \prod_{i=1}^{\ell-1} c_{f_i}} - \sum_{i=1}^{\ell-1} \frac{1}{|F|^i \cdot \prod_{j=\ell-i+1}^{\ell-1} c_{f_j}} \right) \cdot p^*_{j_{\ell-1}} \tag{4.30}$$

With this at hand, at step k we will have

$$e^*_{j_k} \ge \left(\frac{L-1}{c_{f_k} \cdot |F|^k \cdot \prod_{i=1}^{k-1} c_{f_i}} - \sum_{i=1}^{k-1} \frac{1}{c_{f_k} \cdot |F|^i \cdot \prod_{j=k-i+1}^{k-1} c_{f_j}} - 1 \right) p^*_{j_k} - \sum_{f \in F:\, g^{\text{out}}_f = j_k} r^*_f$$

Since k is the last index in the sequence, i.e, good j_{k-1} is the last to be produced, we know that $r^*_f = 0$ for any firm f that has j_k as its output good. From the above, we have that

$$e^*_{j_k} \ge \left(\frac{L-1}{c_{f_k} \cdot |F|^k \cdot \prod_{i=1}^{k-1} c_{f_i}} - \sum_{i=1}^{k-1} \frac{1}{c_{f_k} \cdot |F|^i \cdot \prod_{j=k-i+1}^{k-1} c_{f_j}} - 1 \right) p^*_{j_k}$$

By picking L to be sufficiently large, e.g., large enough for the following to hold

$$\frac{L}{(|F| \cdot \max_f c_f)^m} - \frac{m}{(\min_f c_f)^m} \ge 0,$$

we obtain a contradiction. We compute the proof below by proving Inequality (4.30).

Base Case: Let $\ell = 1$, and consider firm f_1 that produces good j_0 from good j_1. It follows that in the objective function of linear program $\mathcal{P}_2$ of Figure 4.7 corresponding to firm f_1, the multiplier of the expenditure s_{f_1} is non-negative, i.e., $c_{f_1} \cdot p^*_{j_0} - p_{j^*_1} \ge 0$. There are two cases:

- If $c_{f_1} \cdot p^*_{j_0} - p_{j^*_1} > 0$, then $s^*_{f_1} = L \cdot p^*_{j_1}$.
- If $c_{f_1} \cdot p^*_{j_0} - p_{j^*_1} = 0$, then $s^*_{f_1} = r^*_{f_1} \ge \frac{L-1}{|F|} \cdot p^*_{j_0}$

where the last inequality follows from the fact that $e_{j_0} < 0$, by Statement (4.29). In either case, in the second case using the fact that $p^*_{j_0} \ge p^*_{j_1}/c_{f_1}$, we can bound the expenditure $e^*_{j_1}$ as follows:

$$0 > e^*_{j_1} \ge \left(\frac{L-1}{|F| \cdot c_{f_1}} - 1 \right) \cdot p^*_{j_1} - \sum_{f \in F:\, g^{\text{out}}_f = j_1} r^*_f$$

From this, we get that

$$r^*_{f_2} \ge \left(\frac{L-1}{|F|^2 \cdot c_{f_1}} - \frac{1}{|F|} \right) \cdot p^*_{j_1}$$

Induction Step: Consider some step ℓ and consider firm f_ℓ that produces good $j_{\ell-1}$ from good j_ℓ. It follows that in the objective function of linear program $\mathcal{P}_2$ of Figure 4.7 corresponding to firm f_ℓ, the multiplier of the expenditure s_{f_ℓ} is non-negative, i.e., $c_{f_\ell} \cdot p^*_{j_{\ell-1}} - p_{j^*_\ell} \ge 0$. There are two cases:

- If $c_{f_\ell} \cdot p^*_{j_{\ell-1}} - p_{j^*_\ell} > 0$, then $s^*_{f_\ell} = L \cdot p^*_{j_\ell}$.
- If $c_{f_\ell} \cdot p^*_{j_{\ell-1}} - p_{j^*_\ell} = 0$, then

$$s^*_{f_\ell} = r^*_{f_\ell} \geq \left(\frac{L-1}{|F|^\ell \cdot \prod_{i=1}^{\ell-1} c_{f_i}} - \sum_{i=1}^{\ell-1} \frac{1}{|F|^i \cdot \prod_{j=\ell-i+1}^{\ell-1} c_{f_j}} \right) \cdot p^*_{j_{\ell-1}}$$

where the inequality in the second case follows from the induction hypothesis. In either case, in the second case using the fact that $p^*_{j_{\ell-1}} \geq p^*_{j_\ell}/c_{f_\ell}$, we can bound the expenditure $e^*_{j_\ell}$ as follows:

$$0 > e^*_{j_\ell} \geq \left(\frac{L-1}{|F|^\ell \cdot \prod_{i=1}^{\ell} c_{f_i}} - \sum_{i=1}^{\ell-1} \frac{1}{|F|^i \cdot \prod_{j=\ell-i+1}^{\ell} c_{f_j}} - 1 \right) \cdot p^*_{j_\ell} - \sum_{f \in F:\, g^{\text{out}}_f = j_1} r^*_f$$

From this, we obtain that

$$r^*_{f_{\ell+1}} \geq \left(\frac{L-1}{|F|^{\ell+1} \cdot \prod_{i=1}^{\ell} c_{f_i}} - \sum_{i=1}^{\ell} \frac{1}{|F|^i \cdot \prod_{j=\ell-i+1}^{\ell} c_{f_j}} \right) \cdot p^*_{j_\ell}$$

□

Claim 8 implies that for any firm $f \in F_J$, there cannot be an edge to any consumer $i \in N \setminus N_J$ or any firm $f' \in F \setminus F_J$ in the economy graph. Indeed, for an edge (f, i), the claim asserts that the production good g^{out}_f of f will be in the set J, whereas consumer i only values goods in the set $G \setminus J$ positively, by virtue of being in $N \setminus N_J$. Similarly, for an edge (f, f'), firm f produces the good $g^{\text{out}}_f \in J$, whereas the input good $g^{\text{in}}_{f'}$ of firm f' is in $G \setminus J$, by virtue of f' being in $F \setminus F_J$.

From the two paragraphs succeeding Claim 7 and Claim 8, N_J and $N \setminus N_J$ are two strongly connected components in the economy graph of the market, contradicting the sufficiency condition requiring that in the economy graph there is a strongly connected component containing all the consumer-nodes. This completes the proof. □

4.6.3 Markets with Leontief-free Utilities and Productions

In this section we show how our linear-OPT-gate can be used to obtain the PPAD-membership of finding competitive equilibria in Arrow-Debreu markets with Leontief-free utilities and productions. Recall that, as we mentioned in the beginning of the section, this is the largest class of functions for which membership in PPAD (and hence rationality of solutions) has been proven [119]. In the following section we define another class of functions which generalizes all previous ones, but its generality is incomparable to the Leontief-free case. Our main theorem of the section is the following.

Theorem 4.6.3. *Computing a competitive equilibrium of an Arrow-Debreu market with Leontief-free utilities and productions is in PPAD.*

Our proof in this section provides a significant simplification over that of Garg et al. [119]. To keep the section as self-contained as possible, we define these markets in detail here, rather than explain how they generalize the markets of Section 4.6.2, but we make appropriate references to that section.

Markets with Production. In an Arrow-Debreu market with production $\mathcal{M}$, we have a set N of consumers, a set G of infinitely divisible goods, and a set F of firms. Let $n = |N|$, $m = |G|$, and $\ell = |F|$. We will typically use index i to refer to consumers, j or g to refer to goods and f to refer to firms. Each consumer brings an endowment $w_i = (w_{i1}, \dots, w_{im})$ to the market, with $w_{ij} \geq 0$ for all $i \in N$ and $j \in G$. We may assume without loss of generality that for every good j, we have $\sum_{i \in N} w_{ij} = 1$, i.e., that the total endowment of each good is 1. We will use $\mathbf{x}^i = (x_{i1}, \dots, x_{im})$ to denote the vector of quantities of goods allocated to consumer $i \in N$ in $\mathcal{M}$, and we will call it the *bundle* of consumer i. Let $\mathbf{x} = (\mathbf{x}^1, \dots, \mathbf{x}^n)$ be the vector of such bundles. We will use $\mathbf{p} = (p_1, \dots, p_m)$ to denote the vector of *prices* in $\mathcal{M}$, one for each good $j \in G$. Prices are non-negative, so $p_j \geq 0$ for all $j \in G$. Given a vector of prices $\mathbf{p}$, the *budget* of consumer $i \in N$ is defined as $\sum_{j \in G} w_{ij} p_j$; intuitively, this is the amount of money that the consumer acquires by selling her endowment at prices $\mathbf{p}$.

Utility Functions. Every consumer has a utility function $u_i \colon \mathbb{R}^m_{\geq 0} \to \mathbb{R}_{\geq 0}$ mapping a bundle $\mathbf{x}^i$ to a non-negative real number. In this section, we consider *Leontief-free utilities*. Such utility functions are specified by a finite list K_i of *tranches* $\{s^i_k\}_{k \in K_i}$.[16] Associated with every tranche s^i_k is a number $L^i_k \in \mathbb{R}_{\geq 0} \cup \{\infty\}$, which is an upper bound for the total utility that can be accrued from this tranche, and for every $j \in G$ there is a number $u^i_{jk} \geq 0$, which is the rate at which good j provides utility for consumer i on tranche k. That is, the utility consumer i receives from the bundle $\mathbf{x}^i$ is calculated by solving the following linear program.

$$\begin{aligned} \text{maximize } & \sum_{k \in K_i} \sum_{j \in G} u^i_{jk} x^i_{jk} \\ \text{subject to } & \sum_{j \in G} u^i_{jk} x^i_{jk} \leq L^i_k, \quad \text{for all } k \in K_i \\ & \sum_{k \in K_i} x^i_{jk} \leq x^i_j, \quad \text{for all } j \in \mathcal{G} \end{aligned}$$

Note that the class of linear utilities that we considered in Sections 4.6.1 and 4.6.2 is a special case of this construction where every tranche is unbounded and can accrue utility from only a single good. In the case of SPLC utilities which we mentioned in the beginning of Section 4.6, the restriction that the tranches must be unbounded is dropped.

[16]Garg et al. [119] us the term "segments" to refer to these "portions" of the utility function. We use the term "segments" for the parts of the SSPLC functions in Section 4.6.4 which has a different meaning, so we adopt the term "tranche" here instead.

Firm Shares. Each consumer $i \in N$ has a share $\theta_{if} \in [0, 1]$ of the profit of each firm $f \in F$. We assume that the profits are entirely shared among the consumers, i.e., for every firm $f \in F$, we have that $\sum_{i \in N} \theta_{if} = 1$.

Production functions. Every firm f has a *Leontief-free production function*. Here we will use slightly different terminology from Section 4.6.2, following the one used by Garg et al. [119]. We will say that each firm f has a *partitioning* of the set of goods G into *raw goods* and *produced goods*, i.e., $\mathcal{G} = \mathcal{R}_f \sqcup \mathcal{P}_f$. The firm then first converts the raw goods into *raw units*, and these in turn are converted into produced goods.

More precisely, the production function of firm f is specified by a list of tranches $\{s_k^f\}_{k \in K_f}$. To each of these tranches we have the following associated quantities:

- a number $L_k^f \in \mathbb{R}_{\geq 0} \cup \{\infty\}$, which is a bound on how many raw units can be produced on this tranche, for every raw good $j \in \mathcal{R}_{f,k} \subseteq \mathcal{R}_f$,
- a number $\alpha_{jk}^f > 0$, which denotes the rate at which good j can be converted into raw units on this tranche (i.e., the raw-good-to-raw-unit conversion rate), and,
- for every produced good $j' \in \mathcal{P}_{f,k} \subseteq \mathcal{P}_f$, a number $\beta_{j'k}^f > 0$, which denotes the rate at which raw units can be converted into good j' on this tranche (i.e,, the raw-unit-to-produced-good conversion rate).

Let $\mathbf{y}^f = (y_j)_{j \in G}$ be the *production vector* of firm f, i.e., the vector of amounts of raw goods and produced goods involved in its production function.

Remark 28 (One tranche for each firm)**.** Note that whether a firm produces on a tranche k is independent of whether it produces on another tranche k'. Therefore, to simplify the exposition, we may assume that the production function of any firm has only one tranche. If a firm's production function has multiple tranche, then we may simply "split" the firm into one firm for every tranche of the function, and allocate to the consumers shares in the new firms that are identical to the shares they had in the original firm. This change will not impact the production of the firms or the budget constraints of the consumers (see Definition 4.6.4 below). Given this, we can write α_j^f and $\beta_{j'}^f$ for the conversion rate of firm f from quantities of the raw good j to raw units, and for the conversion rate of the firm from raw units to quantities of the produced good j', respectively. Similarly, we can also write L^f for the bound on the number of raw units that the firm can produce on its tranche.

Next, we define the optimal production for firms and the optimal consumption for consumers. We start from the former.

Optimal Production. Consider firm f, and let p_j and p'_j be the prices of goods j and j' in the market respectively. Using some raw good $j \in \mathcal{R}_f$, the firm can produce 1 raw unit on its tranche at a *cost* of p_j/α_j^f. For any produced good $j' \in \mathcal{P}_f$, the firm can use 1 raw unit to produce $\beta^f_{j'}$ units of good j', resulting in a *revenue* of $p_{j'}\beta^f_{j'k}$. With this in mind, we define the *cost-per-unit (cpu)* and *revenue-per-unit (rpu)* of firm f as follows:

$$\text{cpu}^f(\mathbf{p}) = \min_{j \in \mathcal{R}_f} \frac{p_j}{\alpha^f_j} \quad \text{and} \quad \text{rpu}^f(\mathbf{p}) = \max_{j' \in \mathcal{P}_f} p_{j'}\beta^f_{j'}$$

Intuitively, the cpu of a firm is the minimum cost that it incurs from converting any of its raw goods into raw units, and the rpu is the maximum revenue that it obtains by converting raw units into any of its produced goods.

The *profit-per-unit (ppu)* of firm f is then defined as

$$\text{ppu}^f(\mathbf{p}) = \text{rpu}^f(\mathbf{p}) - \text{cpu}^f(\mathbf{p})$$

We can now define a firm's optimal production.

Definition 4.6.3 (Optimal Production)**.** Given the production vector $\mathbf{y}^f$ of firm f, a production for the firm is *optimal* if the following conditions are satisfied:

1. $\sum_{j' \in \mathcal{P}_f} y^f_{j'}/\beta^f_{j'} \le \sum_{j \in \mathcal{R}_f} \alpha^f_j y^f_j \le L^f$. (***feasibility***)
2. If $\text{ppu}^f(\mathbf{p}) < 0$, then $\mathbf{y}^f = \mathbf{0}$. (***zero unprofitable production***)
3. If $\text{ppu}^f(\mathbf{p}) > 0$, then $\sum_{j \in \mathcal{R}_f} \alpha^f_j y^f_j = L^f$. (***maximum profitable production of raw units***)
4. If $\text{rpu}^f(\mathbf{p}) > 0$, then $\sum_{j' \in \mathcal{P}_f} y^f_{j'}/\beta^f_{j'} = \sum_{j \in \mathcal{R}_f} \alpha^f_j y^f_j$. (***maximum usage of raw units***)
5. For every $j \in \mathcal{R}_f$, if $y^f_j > 0$, then $p_j/\alpha^f_j = \text{cpu}^f(\mathbf{p})$. (***only use cost-optimal raw goods***)
6. For every $j' \in \mathcal{P}_f$, if $y^f_{j'} > 0$, then $p_{j'}\beta^f_{j'} = \text{rpu}^f(\mathbf{p})$. (***only produce revenue-optimal produced goods***)

Before we proceed, with offer the following remark with respect to Condition Item 3 above.

Remark 29 (Maximum Profitable Production)**.** In Definition 4.6.3, when $L^f = \infty$, Condition 3 stipulates that the firm should use an infite amount of raw goods for production. As we explained in the previous section (see Remark 24) and as we we will reiterate in the context of the markets of this section in Remark 30 below, in a competitive equilibrium (see Definition 4.6.4) there is a finite, global upper bound to how much a firm can produce. In turn, this also imposes a global upper bound on how much a consumer can consume. Looking ahead, in the linear programs that we will construct, we will use some upper bounds on production and consumption, which will be sufficiently large to not constrain the aforementioned global upper bounds. For the case of production, that bound will be represented by L, and then, Item 3 should be interpreted as

$$\text{If } \mathrm{ppu}^f(\mathbf{p}) > 0, \text{ then } \sum_{j\in\mathcal{R}_f} \alpha^f_j y^f_j = \min\{L^f, L\},$$

where L will be guaranteed to be large enough such that $\min\{L^f, L\} = L$ only when $L^f = \infty$.

Optimal Consumption and bang-per-buck (BPB). For the consumers, the optimal consumption amounts to them utilizing tranches in order of decreasing *bang-per-buck*. Given consumer $i \in N$ and prices $\mathbf{p}$, the BPB of a pair (j,k) consisting of a good $j \in G$ and a tranche $k \in K_i$ is defined as $\mathrm{BPB}_i(j,k) = u^i_{jk}/p_j$. In particular, $\mathrm{BPB}_i(j,k)$ has a maximum BPB over pairs $(j,k) \in G \times K_i$, then consumer i will buy quantities of good j, and "allocate" the good to tranche k until the upper bound L^i_k on the utility that the consumer gain from this tranche has been met. This process is repeated until the budget of the consumer is exhausted.

Competitive Equilibrium. We are now ready to define the notion of a competitive equilibrium in markets with Leontief-free utilities and productions.

Definition 4.6.4 (Competitive Equilibrium - Markets with Leontief-free Utilites and Productions)**.** A competitive equilibrium of an Arrow-Debreu market with Leontief-free utilities and Leontief-free productions is a tuple $(\bar{\mathbf{p}}, \bar{\mathbf{x}}, \bar{\mathbf{y}})$ consisting of non-negative prices $\bar{\mathbf{p}}$, non-negative bundles $\bar{\mathbf{x}}^i$ for $i \in \mathcal{N}$, and non-negative amounts of goods $\bar{y}^f \in \mathbb{R}^{|\mathcal{R}_f \cup \mathcal{P}_f|}_{\geq 0}$ for $f \in F$ satisfying the following conditions:

1. For every firm $f \in F$, $\bar{y}^f$ is an optimal production vector for f. ***(firm profit maximization)***

2. For every consumer $i \in \mathcal{N}$, $\bar{\mathbf{x}}^i$ is an optimal consumption vector for i under the budget constraint $\sum_{j\in G}\sum_{k\in K_i} \bar{p}_j \cdot \bar{x}^i_{jk} \leq \sum_{j\in G} w^i_j \bar{p}_j + \sum_{f\in F} \theta^i_f \left(\sum_{j\in\mathcal{P}_f} \bar{p}_j \cdot \bar{y}^f_j - \sum_{j\in\mathcal{R}_f} \bar{p}_j \cdot \bar{y}^f_j\right)$, ***(bundle optimality)***

3. $\bar{z}_j \leq 0$, and $\bar{z}_j \cdot \bar{p}_j$, where for every good $j \in \mathcal{G}$,
   $\bar{z}_j = \sum_{i\in N}\sum_{k\in K_i} \bar{x}^i_{jk} + \sum_{f\in F:\, j\in\mathcal{R}_f} \bar{y}^f_j - \sum_{f\in F:\, j\in\mathcal{P}_f} \bar{y}^f_j - 1$ ***(market clearing)***

Condition 1 requires that at the chosen set of prices $\bar{\mathbf{p}}$, each firm maximizes its profit, given its production functions. Condition 2 requires that at the chosen set of prices $\bar{\mathbf{p}}$, each consumer maximizes her utility subject to their budget constraints, where the budget consists of the amount earned from selling all the consumer's endowments and the profit share of the consumer from the production of the firms on the different tranches. Finally, Condition 3 is the market clearing condition, which requires that the total consumption of each good is at most the total production plus the total endowment of the consumers, and supply equals demand for all goods which are not priced at 0. Similarly to Section 4.6.2, and as we explain later, we may in fact assume without loss of generality that in any competitive equilibrium *all the prices are positive*, and hence Condition 3 reduces to $\bar{z}_j = 0$ for all $j \in G$. Note that in Condition 3 we have used that $\sum_{\in N} w_{ij} = 1$ for each good $j \in G$.

Sufficiency Conditions. A competitive equilibrium as defined above exists for every market $\mathcal{M}$, under some sufficiency conditions. Similarly to Section 4.6.2, we will use the set of conditions used by Maxfield [167], also used in the series of papers on market equilibria that we mentioned in the beginning of Section 4.6.

1. For every consumer $i \in N$, there exists some good $j \in G$ that the consumer endows in a positive amount, i.e., $w_{ij} > 0$.

2. We will say that a consumer $i \in N$ is ***nonsatiated*** *for good* $j \in G$ if there exists some tranche k such that $u^i_{jk} > 0$ and $L^i_k = \infty$. Following Garg et al. [119], we will assume that for any good $j \in G$, there exists some consumer $i \in N$ that is nonsatiated with respect to j. Similarly, a firm is ***nonsatiated*** *for good* $j \in G$ if $j \in \mathcal{R}_f$ and $L^f = \infty$.

   We remark that this condition naturally generalizes the condition that "for any good there exists some consumer with positive utility for that good", which we used in Sections 4.6.1 and 4.6.2.

3. Consider a graph $\mathcal{G}_F(\mathcal{M})$ in which the nodes are the goods, and an edge (j, j') has weight
$$w_{jj'} = \max_{f \in F:\ j \in \mathcal{R}_f \wedge j' \in \mathcal{P}_f} \alpha^f_j \cdot b'^f_j,$$
   If the set is empty, the weight is defined to be 0.

   The above weight captures the fact that j' can be produced from j, via the intermediate raw units, at combined conversion rate $w_{jj'}$ by some firm $f \in F$. Then, for any cycle $C = (g_0, g_1), (g_1, g_2) \ldots, (g_{k-2}, g_{k-1})$ of $G_F(\mathcal{M})$ the product of the weights of the edges is less than 1, i.e., $\prod_{e \in C} \alpha_e < 1$.

   This condition is known as the ***no production out of nothing and no vacuous production*** condition. Indeed, if $\prod_{e \in C} \alpha_e > 1$, then it would be possible to increase the quantity of some good, without decreasing the quantity of any other good. The case of $\prod_{e \in C} \alpha_e = 1$ refers to the case of vacuous production, which is also disallowed in our model, similarly to the related works. e.g., see Garg and Vazirani [112].

4. Consider the *economy graph* $\mathcal{G}_E(\mathcal{M})$ of the market $\mathcal{M}$ in which the nodes are the consumers and the firms, and

   - there is an edge (i, i') between consumer-node i and consumer/firm-node i' if i endows a good j for which i' is nonsatiated.
   - there is an edge (f, i) between firm-node f and consumer/firm-node i, if $L^f = \infty$ and there exists some good $j \in \mathcal{P}_f$ for which i is nonsatiated.

Then, $\mathcal{G}_E(\mathcal{M})$ contains a strongly connected component containing all the consumer-nodes. This condition generalizes the corresponding strong connectivity condition that we used in Section 4.6.2.

Remark 30 (Bounds on Production)**.** Very similarly to Section 4.6.2, we remark that there are some inherent bounds on how much a firm can produce on a tranche, even if for that segment we have $L^f = \infty$. The idea here is very similar to that of Remark 24; given that we start from a finite set of endowed goods, the absence of cycles of profitable production in an equilibrium restricts the production to take place in chains of length bounded by m. This implies a global upper bound L^c on how many raw goods can be used or produced. Again, looking ahead, this will allow us to impose "loose" upper bounds on the production and consumption in the linear programs that we will devise, without compromising the existence of a competitive equilibrium.

Bounds on the prices. Again, similarly to Section 4.6.2, we may assume without loss of generality that all the prices are strictly positive, i.e., $p_j > 0$ for all goods $j \in G$. The argument to achieve this is very similar to the one used in Section 4.6.2, and has also been established in [119]. Note that by this assumption, the quantity $\mathrm{BPB}_i(j,k)$ is well-defined for every $j \in G$ and $k \in K_i$.

Remark 31 (Normalized Prices)**.** Given that $p_j > 0$ for all $j \in G$, we can normalize the prices to sum to 1 without loss of generality, i.e., we may assume that for every good $j \in G$, we have that $\sum_{j \in G} p_j = 1$.

Again, in a similar manner to Sections 4.6.1 and 4.6.2, we will use a parameter ε to capture the fact that if the price p_j for a good $j \in G$ is sufficiently smaller than the price $p_{j'}$ for a good $j' \in G$, then $\mathrm{BPB}(j,k) > \mathrm{BPB}(j',k')$, for any $k,k' \in K_i$. Specifically, we can compute $\varepsilon > 0$ such that

$$\text{If } p_j \leq \varepsilon \cdot p_{j'} \text{ and } u^i_{jk} > 0 \text{ then } \mathrm{BPB}_i(j,k) > \mathrm{BPB}_i(j',k').$$

Additionally, we can pick ε to be sufficiently small such that $\varepsilon < \frac{w_{ij}}{m}$ for all $i \in N$ and all $j \in G$. Given ε, we will impose a stricter lower bound on the prices, which will be useful later on: in particular, we will assume that for all $j \in G$, $p_j \geq \frac{\varepsilon^m}{m}$.

Preprocessing

The approach that we will take for the proof is very similar to the one that we used in Section 4.6.2. The main difference comes from the fact that we now need to consider pairs of (goods, tranches) rather than just goods. This mainly complicates notation, but the type of arguments that we make are very much along the same lines as in the previous section. Again, we employ a standard variable change (see *Gale's Substitution* in Remark 22) to work with expenditures for the consumers and the firms, rather than with quantities of goods. This is to avoid multiplications of parameters which will be input to the circuit that we will construct, in particular the prices $\mathbf{p}$ and the quantities x^i_{jk}, and y^i_{jk} in the bundle optimality condition of the competitive equilibrium in Definition 4.6.4.

Linear Program $\mathcal{P}_{\text{con}}$

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{j\in\mathcal{G}}\sum_{k\in K_i}\frac{p_j}{u^i_{jk}}\cdot q^i_{jk} \\ \text{subject to} & \displaystyle\sum_{j\in\mathcal{G}} q^i_{jk} \le \min\{L^i_k, CL+1\}\cdot\min_j \frac{p_j}{u^i_{jk}}, \quad \forall k\in K_i \\ & \displaystyle\sum_{j\in\mathcal{G}}\sum_{k\in K_i} q^i_{jk} = \sum_{j\in\mathcal{G}} w^i_j p_j + \sum_{f\in\mathcal{F}}\theta^i_f\cdot\left(\sum_{j\in\mathcal{P}_f} r^f_j - \sum_{j\in\mathcal{R}_f} s^f_j\right) \\ & q^i_{jk}\ge 0, \ \ \forall j\in\mathcal{G}, \ \ \forall k\in K_i \end{array}$$

Figure 4.9: The linear program $\mathcal{P}_{\text{con}}$ for the optimal expenditures in consumption. Note that $CL+1$ will be chosen to be large enough such that $\min\{L^i_k, CL+1\} = CL+1$ only when $L^i_k = \infty$.

Consumer expenditure. For the consumers, we will use the following standard change of variables. We will let $q^i_{jk} = x^i_{jk}\cdot p_j$ be the *expenditure* of consumer i on good j and tranche k, i.e., how much money the consumer spends on the good at a given tranche given an allocation x^i_{jk} and a price p_j.

Firm expenditure. For the firms, we will use a similar substitution for the case of production. Namely, we will let

- For $j\in\mathcal{R}_f$, let $s^f_j = p_j\cdot y^f_j$ be the *expenditure* of firm f on its raw good j on all tranches, i.e., the amount of money spend on y_f units of the raw good j, at given set of prices $\mathbf{p}$.
  Let $\mathbf{s}^f = \left(s^f_j\right)_{j\in\mathcal{R}_f}$ and let $\mathbf{s} = \left(\mathbf{s}^f\right)_{f\in F}$.
- For $j\in\mathcal{P}_f$, let $r^f_j = p_j\cdot y^f_j$ be the *revenue* of firm f from its produced good j on all tranches, i.e., the amount of money earned from y_f units of the produced good j, at a given set of prices $\mathbf{p}$.
  Let $\mathbf{r}^f = \left(r^f_j\right)_{j\in\mathcal{P}_f}$ and let $\mathbf{r} = \left(\mathbf{r}^f\right)_{f\in F}$.

The program $\mathcal{P}_{\text{con}}$ for optimal consumer expenditure. We are now ready to write the linear program $\mathcal{P}_1$, the solution of which will give us the optimal expenditures for the consumers, see Figure 4.9. In the linear program, notice the quantity $\min\{L^i_k, CL+1\}$. $CL+1$ will be chosen to be large enough, so that $\min\{L^i_k, CL+1\} = CL+1$ only when $L^i_k = \infty$. This is to capture the cases in which the consumer should be allowed to spend an infinite amount of money on goods on a tranche, since there is upper bound on the utility the she can accrue from this tranche. Still, we remark again (see Remark 30) that there are inherent bounds on how much a consumer can spend, which do not come from L^i_k but rather from the inherent global upper bounds on production.

In particular, it holds that $\sum_{j\in G} u^i_{jk}x^i_{jk} \leq L^p$, where L^p is a global upper bound on the consumption resulting from the global upper bound L^c on the production.

In previous sections, it was straightforward to see that the from the optimal solution $\mathbf{q}^i$ of the corresponding linear program, one could recover the optimal bundle $\mathbf{x}^i$ of consumer i. Here, we prove that in a simple lemma below. Note that we only need to guarantee that the linear program computes the optimal expenditures correctly at a competitive equilibrium, and so we can use the fact that $\sum_{j\in G} u^i_{jk}x^i_{jk} \leq L^p$.

Lemma 25. *Suppose that $\mathbf{q}^i$ is an optimal solution to linear program $\mathcal{P}_{con}$ of Figure 4.9, and let Let $x^i_{jk} = q^i_{jk}/p_j$ for all $j \in G$ and $k \in K_i$. Then, the resulting bundle $\mathbf{x}^i$ is an optimal consumption vector for consumer i at a competitive equilibrium of the market.*

Proof. First note that the linear program is feasible, because by the strong connectivity of the economy graph $G_E(\mathcal{M})$, i is non-satiated with respect to some good, which means that she can spend its entire budget on this good. By construction, consumer i will spend money in order of increasing $\mathrm{BPB}_i(j,k)$, establishing the optimality of the resulting bundle $\mathbf{x}^i$. It remains to show that the feasibility constraints are satisfied. Note that if $x^i_{jk} > 0$, then $q^i_{jk} > 0$, establishing that $p_j/u^i_{jk} = \min_{j'} p_{j'}/u^i_{j'k}$. Hence, for any $i \in G$ an $k \in K_i$, we have that

$$\begin{aligned}\sum_{j\in G} u^i_{jk}x^i_{jk} &= \sum_{j\in G}\left(\frac{u^i_{jk}}{p_j}\right)q^i_{jk} = \frac{1}{\min\limits_{j'\in G}\ p_{j'}/u^i_{j'k}}\sum_{j\in G} q^i_{jk}\\ &\leq \frac{1}{\min\limits_{j'\in G}\ p_{j'}/u^i_{j'k}}\cdot\min\{L^i_k, CL+1\}\cdot\min_{j'\in G}\frac{p_{j'}}{u^i_{j'k}} = \min\{L^i_k, CL+1\}\end{aligned}$$

This completes the proof of the lemma, since $CL+1$ will be chosen to be larger than any finite L^i_k and larger than L^p. □

The programs $\mathcal{P}_{\mathbf{prod1}}$ and $\mathcal{P}_{\mathbf{prod2}}$ for optimal firm expenditure. In Section 4.6.2 we devised a single linear program for the optimal production of the firms. Here, since we have variables for both the expenditures on raw goods and the expenditures on produced goods, we will devise two linear programs, one for each case. Those can be seen in Figure 4.10. Again, in Section 4.6.2 the corresponding linear program almost straightforwardly captured the optimal production of the firm which it represented; here we argue that this is the case with another simple lemma.

Lemma 26. *Suppose that $\mathbf{s}^f$ and $\mathbf{r}^f$ are solutions to the linear programs $\mathcal{P}_{prod1}$ and $\mathcal{P}_{prod2}$ of Figure 4.10 respectively. Let $y^f_j = s^f_j/p_j$ for $j \in \mathcal{R}_f$ and $y^f_{j'} = r^f_{j'}/p_{j'}$ for $j' \in \mathcal{P}_f$. Then $\mathbf{y}^f$ is an optimal production vector for firm f at a competitive equilibrium of the market.*

Proof. We will argue that $\mathbf{y}^f$ satisfies all the conditions of Definition 4.6.3. Since $\mathbf{y}^f$ consists of optimal solutions to the two linear programs, Condition 5 and Condition 6

<u>Linear Program $\mathcal{P}_{\text{prod1}}$</u>

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{j\in\mathcal{R}_f}\left(\text{rpu}^f(\mathbf{p})-\frac{p_j}{\alpha^f_j}\right)s^f_j \\ \text{subject to} & \displaystyle\sum_{j\in\mathcal{R}_f} s^f_j \le \min\{L^f,L\}\cdot\text{cpu}^f(\mathbf{p}) \\ & s^f_j \ge 0,\ \forall j\in\mathcal{R}_f \end{array}$$

<u>Linear Program $\mathcal{P}_{\text{prod2}}$</u>

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{j'\in\mathcal{P}_f}\left(p_{j'}\beta^f_{j'}\right)\cdot r^f_{j'} \\ \text{subject to} & \displaystyle\sum_{j'\in\mathcal{P}_f} r^f_{j'} = \sum_{j\in\mathcal{R}_f} s^f_j + B \\ & B = \max\{0,\text{rpu}^f(\mathbf{p})-\text{cpu}^f(\mathbf{p})\}\cdot\min\{L^f,L\} \\ & r^f_{j'} \ge 0,\ \forall j'\in\mathcal{P}_f \end{array}$$

Figure 4.10: The linear programs $\mathcal{P}_{\text{prod1}}$ and $\mathcal{P}_{\text{prod2}}$ for the optimal expenditures in production, for the raw goods (left) and for the produced goods (right). Note that L will be sufficiently large such that $\min\{L^f,L\}=L$ will only hold when $L^f=\infty$, in both programs. Also note that the variable B is only used for notational convenience.

are clearly satisfied. Now, suppose that $\text{ppu}^f(p)<0$. This implies that all the coefficients of linear program $\mathcal{P}_{\text{prod1}}$ are negative, meaning that $s^f_j=0$ for all $j\in\mathcal{R}_f$. By the constraints of linear program $\mathcal{P}_{\text{prod1}}$, it follows that also $r^f_j=0$ for all $j\in\mathcal{P}_f$. Hence $y^f=\mathbf{0}$, and Condition 2 also holds. If $\text{ppu}^f(p)>0$, then at least one coefficient of linear program $\mathcal{P}_{\text{prod1}}$ is positive, implying that $\sum_{j\in\mathcal{R}_f}s^f_j=\min\{L^f,L\}\cdot\text{cpu}^f(\mathbf{p})$. Also, if $s^f_j>0$, then $p_j/\alpha^f_j=\text{cpu}^f(\mathbf{p})$. Hence, we have that

$$\sum_{j\in\mathcal{R}_f}\alpha^f_j y^f_j=\sum_{j\in\mathcal{R}_f}\frac{\alpha^f_j}{p_j}\cdot s^f_j=\frac{1}{\text{cpu}^f(\mathbf{p})}\sum_{j\in\mathcal{R}_f}s^f_j=\frac{1}{\text{cpu}^f(\mathbf{p})}\cdot\min\{L^f,L\}\cdot\text{cpu}^f(\mathbf{p})=\min\{L^f,L\}$$

and hence Condition 3 is satisfied (see also Remark 29). What remains is to argue is that $\mathbf{y}^f$ satisfied Condition Item 1 of Definition 4.6.3, i.e. that it is a feasible production vector. If $\text{ppu}^f(\mathbf{p})<0$, then $y^f=\mathbf{0}$ is feasible. If $\text{ppu}^f(\mathbf{p})>0$, then it can be verified that $\sum_{j\in\mathcal{R}_f}\alpha^f_j y^f_j=\sum_{j\in\mathcal{P}_f}y^f_j/\beta^f_j$. Finally, if $\text{ppu}^f(\mathbf{p})=0$, then

$$\sum_{j'\in\mathcal{P}_f}\frac{y^f_{j'}}{\beta^f_{j'}}=\sum_{j'\in\mathcal{P}_f}\frac{r^f_{j'}}{p_{j'}\cdot\beta^f_{j'}}=\sum_{j'\in\mathcal{P}_f}\frac{r^f_{j'}}{\text{rpu}^f(\mathbf{p})}=\sum_{j\in\mathcal{R}_f}\frac{s^f_j}{\text{cpu}^f(\mathbf{p})}=\sum_{j\in\mathcal{R}_f}\frac{\alpha^f_j s^f_j}{p_j}=\sum_{j\in\mathcal{R}_f}\alpha^f_j y^f_j.$$

This establishes all of the properties of Definition 4.6.3. Finally, let $\hat{G}=\{j\in R_f\colon \text{rpu}(\mathbf{p})-p_j/\alpha^f_j>0\}$. The corresponding constraint of linear program $\mathcal{P}_{\text{prod1}}$ stipulates that $\sum_{j\in\hat{G}}s^f_j=\min\{L^f,L\}\cdot\text{cpu}^f(\mathbf{p})$. Our proof will establish that in a competitive equilibrium, if $L^f>L^c$ (the global upper bound on production), then $\hat{G}$ does not contain any goods. This is to establish that the optimality of s^f_j does not impose any artificial constraints on the expenditure on raw goods due to the constraints of the linear program. $\square$

Feasibility Program Q

$$e_j < e_{j'} \Rightarrow p_j \le \varepsilon \cdot p_{j'},\ \forall j, j' \in G$$
$$p_j \ge \frac{\varepsilon^m}{m},\ \forall j \in G$$
$$\sum_{j \in G} p_j = 1$$

Figure 4.11: The feasibility program Q used to find market-clearing prices.

Excess expenditure and the program Q. Similarly to Sections 4.6.1 and 4.6.2, the equilibrium prices will be obtained via a feasibility program Q. Before we define the program, we introduce the notion of excess expenditure; the definition is very similar to the one we used in previous sections. The *excess expenditure* e_j of a good $j \in G$ is defined as the difference between the total expenditure of all consumers $i \in N$ and firms $f \in F$ for that good and the price of the good plus the total revenue of firms $f \in F$ from that good, i.e.,

$$e_j = \sum_{i \in N} \sum_{k \in K_i} q^i_{jk} + \sum_{f \in F:\ j \in \mathcal{R}_f} s^f_j - p_j - \sum_{f \in F:\ j \in \mathcal{P}_f} r^f_j.$$

Then, at equilibrium prices $\bar{\mathbf{p}}$ we will have market clearing, i.e., $\bar{e}_j = 0$, for all $j \in G$.

We are now ready to define our feasibility program Q; see Figure 4.11. The program looks in fact identical with that which we used in the previous sections; the only difference being the updated definition of the expenditures e_j. The equilibrium prices will be obtained as the output $\bar{\mathbf{p}}$ of Q.

Membership in PPAD: The proof of Theorem 4.6.3

We will again develop the proof in three steps, namely (a) construction of the function F and arguing that it can be represented by a linear arithmetic circuit containing linear-OPT-gates, (b) showing that the linear-OPT-gate can compute all the necessary components, and (c) arguing that a competitive equilibrium can be recovered from a fixed point of F.

The function F. Given the above, we will define $F : D \to D$ with domain

$$D = \left\{ p \in \Delta^{m-1} : \forall j \in G, p_j \ge \frac{\varepsilon^m}{m} \right\} \times_{i \in N} \left([0, C \cdot L + 1]^{m|K_i|} \right) \times [0, L]^{\ell m}.$$

An input to F is a tuple $(\hat{\mathbf{p}}, \hat{\mathbf{q}}, \hat{\mathbf{s}}, \hat{\mathbf{r}})$ of prices, consumer expenditures and firm expenditures for raw goods and produced goods, and the output is another such tuple $(\bar{\mathbf{p}}, \bar{\mathbf{q}}, \bar{\mathbf{s}}, \bar{\mathbf{r}})$. The domain D is the one above since $\sum_{j \in G} \hat{p}_j = 1$ (see Remark 31), and by the upper bounds of $C \cdot L + 1$ and L imposed on the maximum expenditures for consumption and production respectively.

Computation by the linear-OPT-gate. Next, we argue that the solutions to the linear programs $\mathcal{P}_{\text{con}}$ of Figure 4.9, $\mathcal{P}_{\text{prod1}}$ and $\mathcal{P}_{\text{prod2}}$ of Figure 4.10, and the feasibility program $\mathcal{Q}$ of Figure 4.11 can be computed by our linear-OPT-gate.

Lemma 27. *Consider the linear program $\mathcal{P}_{con}$ of Figure 4.9, the linear programs $\mathcal{P}_{prod1}$ and $\mathcal{P}_{prod2}$ of Figure 4.10, and the feasibility program $\mathcal{Q}$ of Figure 4.11. An optimal solution to $\mathcal{P}_{con}$, to $\mathcal{P}_{prod1}$ and to $\mathcal{P}_{prod2}$ can be computed by the linear-OPT-gate. $\mathcal{Q}$ is solvable, and a solution can be computed by the linear-OPT-gate.*

Proof. The proof is very similar to that of Lemma 23. For all of the linear programs, the feasibly domains non-empty and bounded. The gate inputs appear only on the right-hand side of the constraints: these are the prices $\hat{p}_j$, and the production expenditures $\hat{s}^f_j$ and $\hat{r}^f_j$ for linear program $\mathcal{P}_{\text{con}}$, the prices $\hat{p}_j$ (as part of $\text{cpu}^f(\hat{\mathbf{p}})$) for linear program $\mathcal{P}_{\text{prod1}}$, the prices $\hat{p}_j$ (as part of $\text{cpu}^f(\hat{\mathbf{p}})$ and $\text{cpu}^f(\hat{\mathbf{p}})$) and the expenditures s^f_j for linear program $\mathcal{P}_{\text{prod2}}$. The subgradients of the objective functions for all three linear programs are linear, and hence can be computed by a linear pseudo-circuit. For the feasibility program $\mathcal{Q}$, the arguments that it is of the correct form and that it is solvable are identical to those of Lemma 23 (and hence also of Lemma 21), noting the updated definition of the expenditure e_j. □

Arguing optimality and market clearing. To conclude the proof, what is left to show is that a fixed point $(\bar{\mathbf{p}}, \bar{\mathbf{q}}, \bar{\mathbf{s}}, \bar{\mathbf{r}})$ of F indeed corresponds to a competitive equilibrium. We argue that in the following lemma.

Lemma 28. *Let $(\bar{\mathbf{p}}, \bar{\mathbf{q}}, \bar{\mathbf{s}}, \bar{\mathbf{r}})$ be a fixed point of F. Then the triple $(\bar{\mathbf{p}}, \bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{r}})$, where $\bar{x}^i_{jk} = \bar{q}^i_{jk}/\bar{p}_j$, $\bar{y}^f_j = \bar{s}^f_j/\bar{p}_j$ for $j \in \mathcal{R}_f$ and $\bar{y}^f_{j'} = \bar{r}^f_{j'}/\bar{p}_{j'}$ for $j' \in \mathcal{P}_f$ is a competitive equilibrium of $\mathcal{M}$.*

Proof. In Lemma 25 and Lemma 26 we argued that the linear programs $\mathcal{P}_{\text{con}}$, $\mathcal{P}_{\text{prod1}}$ and $\mathcal{P}_{\text{prod2}}$ correctly calculate the optimal consumption vectors of the consumers and the optimal production vectors of the firms. What is left to argue is that $\bar{\mathbf{p}}$ is a vector of market-clearing prices. The proof will follow along very much the same lines as the one of Lemma 24, using updated the updated terminology and notation of this section.

By the definition of e_j, arguing about market-clearing is equivalent to arguing that for all $j \in G$, we have that $\bar{e}_j = 0$. Using a very similar calculation to the one that we used in Lemma 24, this is equivalent to arguing that $\bar{e}_j \geq 0$ for all $j \in G$. We provide the calculation below.

$$\begin{aligned}
\sum_{j\in G}\bar{e}_j &= \sum_{j\in G}\left(\sum_{i\in N}\sum_{k\in K_i}\bar{q}^i_{jk} - \bar{p}_j + \sum_{f\in F\colon\, j\in\mathcal{R}_f}\bar{s}_f - \sum_{f\in F\colon\, j\in\mathcal{P}_f}\bar{r}_f\right)\\
&= \sum_{i\in N}\sum_{j\in G}\sum_{k\in K_i}\bar{q}^i_{jk} - \sum_{j\in G}\bar{p}_j + \sum_{f\in F}(\bar{s}_f - \bar{r}_f)\\
&= \sum_{j\in G}\sum_{i\in N} w_{ij}\bar{p}_j - 1 + \sum_{i\in N}\sum_{f\in F}\theta_{if}\cdot(\bar{r}_f - \bar{s}_f) + \sum_{f\in F}(\bar{s}_f - \bar{r}_f)\\
&= \sum_{j\in G}\bar{p}_j\cdot\sum_{i\in N} w_{ij} - 1 + \sum_{f\in F}(\bar{r}_f - \bar{s}_f)\cdot\sum_{i\in N}\theta_{if}\cdot\sum_{f\in F}(\bar{s}_f - \bar{r}_f)\\
&= \sum_{i\in N} w_{ij} - 1 + \sum_{f\in F}(\bar{r}_f - \bar{s}_f) + \sum_{f\in F}(\bar{s}_f - \bar{r}_f)\\
&= 0,
\end{aligned}$$

where in the calculations above we used:

- in Equation 2, that $\sum_{j\in G}\sum_{f\in F\colon\, j\in\mathcal{R}_f}\bar{s}_f = \sum_{f\in F}\bar{s}_j$ and that $\sum_{j\in G}\sum_{f\in F\colon\, j\in\mathcal{P}_f}\bar{r}_f = \sum_{f\in F}\bar{r}_f$,
- in Equation 3, that $\sum_{i\in N}\sum_{j\in G}\sum_{k\in K_i}\bar{q}_{ij} = \sum_{i\in N}\sum_{j\in G} w_{ij}\bar{p}_j + \sum_{i\in N}\sum_{f\in F}\theta_{if}(\bar{r}_f - \bar{s}_f)$, which follows from the bundle optimality of the market equilibrium in Definition 4.6.4, which at an optimal solution is sastisfied with equality,
- in Equations 3 and 5, that $\sum_{j\in G}\bar{p}_j = 1$, which is without loss of generality, see Remark 31,
- in Equation 5, that $\sum_{i\in N}\theta_{if} = 1$ for every $f\in F$, by definition, and
- in Equation 6, that $\sum_{i\in N} w_{ij} = 1$, which is without loss of generality.

Assume by contradiction that there exists some $j_1\in G$ such that $\bar{e}_{j_1} < 0$; we will obtain a contradiction to the strong connectivity of the economy graph of the market. We define the following three sets:

- $J = \{j\in G\colon\, \bar{e}_j \le \bar{e}_{j'},$ for all $j'\in G\}$. In other words, J is the set of goods with minimum excess expenditure.
- $N_J = \{i\in N\colon$ there exists $j\in J$ such that consumer i is nonsatiated for good $j\}$.
- $F_J = \{f\in F\colon$ there exists $j\in J$ such that firm f is nonsatiated for good $j\}$.

Since $e^*_{j_1} < 0$, by the definition of J we know that

$$\text{for all } j\in J,\ \text{we have } e^*_j < 0. \tag{4.31}$$

Also, since $e^*_{j_1} < 0$, and $\sum_{j\in G} e^*_j = 0$, there must exist some other good j_2 such that $e^*_{j_2} > 0$. In particular, that implies that J is a strict subset of G, i.e., $J \subsetneq G$. We state and prove the following claim.

Claim 9. *For all $i \in N_J$ and all $j' \in G \setminus J$, it holds that $w_{ij'} = 0$.*

Proof. The proof of the claim is very similar to that of Claim 7 which we presented in Section 4.6.2. Assume by contradiction that there exists some consumer $i_0 \in N_J$ and some good $j' \in G \setminus J$ such that $w_{i_0 j'} > 0$. We will show that this implies that there exists some $j_\ell \in J$ with $\bar{e}_{j_\ell} \geq 0$, contradicting Statement (4.31). Since $\bar{\mathbf{p}}$ is a solution to feasibility program Q of Figure 4.8, we have that $\bar{p}_j \leq \varepsilon \cdot \bar{p}_{j'}$, for every $j \in J$. Hence, by choosing ε to be sufficiently small, we may lower-bound the budget of consumer i by

$$w_{i_0 j'} \cdot p^*_{j'} \geq \frac{w_{i_0 j'}}{\varepsilon \cdot m} \cdot \sum_{j \in J} \bar{p}_j \geq (C \cdot L + 1) \sum_{j \in J} p^*_j$$

Since $i_0 \in N_J$, by the choice of ε, there is at least one good $g \in J$ such that $\mathrm{BPB}_{i_0}(g,k) > \mathrm{BPB}_{i_0}(g',k)$, for all $g' \in G \setminus J$, and for any two tranches $k, k \in K_i$, i.e., the consumer prefers to buy quantities of good g rather than any good which is not in the set J, on any tranche. Since consumer i_0 is nonsatiated for some good $j \in J$ (by virtue of being in N_J), by the fact that $\bar{q}_{i_0 j}$ is an optimal solution to linear program $\mathcal{P}_{\text{con}}$ of Figure 4.9, it should be that $\bar{q}_{i_0 j} = C \cdot L + 1$, which implies that $\bar{q}_{i_0 j} \geq (C \cdot L + 1) p^*_j$. By substituting that into the definition of the excess expenditure $\bar{e}_j$, we obtain the following inequality:

$$0 > \bar{e}^*_j \geq (CL+1)\bar{p}_j - \sum_{f \in F:\ j \in \mathcal{P}_f} \bar{r}^f_j - \bar{p}_j = CL \cdot \bar{p}_j - \sum_{f \in F:\ j \in \mathcal{P}_f} \bar{r}^f_j.$$

Therefore, it suffices to show that

$$\sum_{f \in F:\ j \in \mathcal{P}_f} \bar{r}^f_j \leq (C \cdot L) \cdot \bar{p}_j;$$

in that case we will have $\bar{e}_j \geq 0$ and we will obtain a contradiction. Now consider any firm $f \in F$. Note that if $\bar{r}^f \leq 0$, then the inequality above clearly follows. Otherwise, if $\bar{r}^j_f > 0$, then $\mathrm{rpu}(\bar{\mathbf{p}}) = \bar{p}_j \cdot \beta^f_j$. In this case, using the constraints of linear program $\mathcal{P}_{\text{prod2}}$ of Figure 4.10, we may bound the production of good j by firm f as follows:

$$\begin{aligned}
\bar{r}^f_j &\leq \sum_{j \in \mathcal{R}_f} \bar{s}^f_j + \max\{0, \mathrm{rpu}^f(\bar{\mathbf{p}}) - \mathrm{cpu}^f(\bar{\mathbf{p}})\} \cdot \min\{L^f, L\} \\
&\leq \mathrm{rpu}^f(\bar{\mathbf{p}}) \cdot \min\{L^f, L\} \\
&= (\beta^f_j \min\{L^f, L\}) \cdot \bar{p}_j \\
&\leq (\beta^f_j L)\bar{p}_j.
\end{aligned}$$

This clearly bounds $\bar{r}^f_j$ as long as the $L^f \neq \infty$, as in that case $L^f \leq L$. $L^f = \infty$, then $\bar{r} j^f$ is upper bounded by the global upper bound L^c on the production. Again, $L^c \leq L$, and the above inequality follows. By letting $C \geq |\mathcal{F}| \cdot \max\{\beta^f_j : f \in \mathcal{F} \wedge j \in \mathcal{P}_f\}$, we obtain the contradiction $0 > \bar{e}_j \geq 0$. This concludes the proof of the claim. □

Claim 9 establishes that a consumer in the set N_J cannot endow any good that is not in the set J in a positive quantity. Since $J \subsetneq G$, this implies that $N_J \subsetneq N$, since every good is positively endowed by some consumer $i \in N$. Now consider any consumer $i \in N_J$ and any consumer $i' \in N \setminus N_J$ or any firm $f \in F \setminus F_J$. Claim 9 implies that there cannot be an edge (i, i') or an edge (i, f) in the economy graph of the market. Indeed, for (i, i'), notice that by the claim consumer i only endows positively goods that are in J, but consumer i' has value 0 for them by virtue of being in $N \setminus N_J$, and hence she is not nonsatiated for any good endowed positively by consumer i. Similarly, for (i, f), we have that $\mathcal{R}_f \subseteq G \setminus J$ by definition, but consumer i only positively endows goods in J, so clearly f cannot be nonsatiated for any of the goods endowed by consumer i.

Next, we state and prove the following claim.

Claim 10. *Consider and $f \in F_J$, with $L^f = \infty$. For any good $j' \in \mathcal{P}_f$, it holds that $j' \in J$.*

Proof. The claim also follows a very similar idea to that of Claim 10 which we presented in Section 4.6.2. Let $\alpha = \max_{f',j} \alpha_j^{f'}$ and $\beta = \max_{f',j} \beta_j^{f'}$. Assume by contradiction that for firm f, there is some good $j' \in \mathcal{P}_f$ such that $j' \in G \setminus J$. Note that f is nonsatiated with respect to some good in J (by virtue of being in F_J and $L^f = \infty$). Additionally, since $\bar{\mathbf{p}}$ is an optimal solution to feasibility program Q of Figure 4.11, we have that $\bar{p}_j \cdot \varepsilon \bar{p}_{j'}$ for all goods $j \in J$. By the fact that $\bar{\mathbf{s}}^{\mathbf{f}}$ is an optimal solution to linear program $\mathcal{P}_{\text{prod1}}$, it follows that firm f will utilize only goods from J as raw goods for its production. Hence, there exists some good $j_1 \in J$ such that

$$\bar{s}^f_{j_1} \geq \frac{L \cdot \text{cpu}^{f_1}(\mathbf{p})}{m} \geq \frac{L}{m\alpha} \cdot \bar{p}_{j_1}$$

Using this, we conclude that

$$0 > \bar{e}_{j_1} \geq \Big(\frac{L}{m\alpha} - 1\Big)\bar{p}_{j_1} - \sum_{f' :\, j_1 \in \mathcal{P}_{f_1}} \bar{r}^{f'}_{j_1} \tag{4.32}$$

If j_1 were not being produced, we would reach the contradiction that $0 > \bar{e}_{j_1} \geq 0$ by picking L sufficiently large. If j_1 *is* being produced, then it must be produced using some good j_2 from J. This is because by the choice of ε, it is only profitable for any firm to produce units of goods $j \in J$ (that have very low prices) only by using units of goods $j \in J$ as raw goods. Again, if j_2 is not being produced, we reach a contradiction by choosing L sufficiently large, and otherwise we may repeat the argument. Since there cannot be a cycle of profitable production in the graph $G_{\mathcal{F}}(\mathcal{M})$, by the no production out of nothing and no vacuous production sufficiency condition, the argument will need to be repeated at most m times, one for each good in the possibly longest production chain of the graph. In the end, we can pick L to be large enough to ensure that we get a contradiction.

We remark that we made the very same argument in the proof of Claim 8 via establishing appropriate bounds inductively for every step of the chain. In the case of markets with Leontief-free productions however, calculating the resulting quantities becomes very tedious, so we elected to present the proof a bit differently.

As it stands, Inequality 4.32 implies that there exists a firm f_1 such that $r_{j_1}^{f_1} \geq \left(\frac{L}{\ell m\alpha} - \frac{1}{\ell}\right)\bar{p}_{j_1}$. As f_1 produces a good in J, it holds that

1. $\text{rpu}^{f_1}(\mathbf{p}) \geq \text{cpu}^{f_1}(\mathbf{p})$ and
2. f_1 produces only using goods from J.

We consider two cases, depending on whether the inequality in Inequality 1 is strict or not.

Case 1: $\mathbf{rpu}^{f_1}(\bar{\mathbf{p}}) = \mathbf{cpu}^{f_1}(\bar{\mathbf{p}})$. In this case, we find that

$$\left(\frac{L}{\ell m\alpha} - \frac{1}{\ell}\right)\bar{p}_{j_1} \leq \bar{r}_{j_1}^{f_1} \leq \sum_{j\in\mathcal{P}_{f_1}} \bar{r}_j^{f_1} = \sum_{j\in\mathcal{R}_{f_1}} \bar{s}_j^{f_1}$$

From this it follows that there exist a good j_2 such that $\bar{s}_{j_2}^{f_1} \geq (\frac{L}{\ell m^2\alpha} - \frac{1}{\ell m})\bar{p}_{j_1}$. Using that $\bar{p}_{j_2}/\alpha_{j_2}^{f_1} = \text{cpu}^{f_1}(\bar{\mathbf{p}}) = \text{rpu}^{f_1}(\bar{\mathbf{p}}) = \beta_{j_1}^{f_1}\bar{p}_{j_1}$, this implies that $\bar{s}_{j_2}^{f_1} \geq (\frac{L}{\ell m^2\alpha^2\beta} - \frac{1}{\ell m\alpha\beta})\bar{p}_{j_2}$. Now we may bound $\bar{e}_{j_2}$ as before and repeat the argument.

Case 2: $\mathbf{rpu}^{f_1}(\bar{\mathbf{p}}) > \mathbf{cpu}^{f_1}(\bar{\mathbf{p}})$. First, we show that $L^{f_1} = \infty$. Assume that it is not. Then we have:

$$\left(\frac{L}{\ell m\alpha} - \frac{1}{\ell}\right)\bar{p}_{j_1} \leq \bar{r}_{j_1}^{f_1} \leq \sum_{j\in\mathcal{P}_{f_1}} \bar{r}_j^{f_1} \leq \min\{L^{f_1}, L\}\cdot \text{rpu}^{f_1}(\bar{\mathbf{p}}) \leq (\min\{L^{f_1}, L\}\beta)\bar{p}_{j_1} \leq (L^{f_1}\beta)\bar{p}_{j_1}, \tag{4.33}$$

where the last inequality follows because L is chosen sufficiently large for $L_1^f \leq L$ to hold, since $L_1^f \neq \infty$ by assumption. For L sufficiently large though, this leads to a contradiction. Hence we may assume henceforth that $L^{f_1} = \infty$.

Given the above, as $\text{rpu}^{f_1}(\bar{\mathbf{p}}) > \text{cpu}^{f_1}(\bar{\mathbf{p}})$, we have that $\sum_{j\in\mathcal{R}_{f_1}} \bar{s}_j^{f_1} = L\cdot\text{cpu}^{f_1}(\bar{\mathbf{p}})$. As f_1 produces a good from J, f_1 can only produce using goods from J as raw goods. This is because, as we discussed earlier, the prices of goods in J are sufficiently smaller that those for goods in $G \setminus J$, and hence any profitable production that uses goods in J as raw goods should also involve only goods in J as produced goods. Therefore, we establish that there exists a $j_2 \in J$ such that $\bar{s}_{j_2}^{f_1} \geq \frac{L}{m}\text{cpu}^{f_1}(\bar{\mathbf{p}}) \geq \frac{L}{m\alpha}\bar{p}_{j_2}$. Using this, we may again bound $\bar{e}_{j_2}$ and repeat the arguments above.

Finding $j_k \in J$ with $0 > \bar{e}_{j_k} \geq 0$. As we mentioned earlier, by the *no production out of nothing and no vacuous production* sufficiency condition, the above argument has to be repeated at most m times, as that is the maximum possible length of any profitable production chain. In both of the two cases consider above, L becomes smaller by a certain factor as a function of the input parameters. This means that L can be picked large enough to guarantee that at the end of the chain, when the last good is not being produced, we obtain that $0 > \bar{e}_{j_k} \geq 0$, a contradiction.

This concludes the proof of the claim. □

Claim 10 implies that for any firm $f \in F_J$, with $L^f = \infty$, there cannot be an edge to any consumer $i \in N \setminus N_J$ or any firm $f' \in F \setminus F_J$ in the economy graph. Indeed, for an edge (f, i), the claim asserts that the any produced good j' of f will be in the set J, whereas consumer i only values goods in the set $G \setminus J$ positively, by virtue of being in $N \setminus N_J$. Similarly, for an edge (f, f'), firm f produces only goods in J, whereas the for firm f' we have $\mathcal{R}_{f'} \subseteq G \setminus J$, by virtue of f' being in $F \setminus F_J$.

From the two paragraphs succeeding Claim 9 and Claim 10, N_J and $N \setminus N_J$ are two strongly connected components in the economy graph of the market, contradicting the sufficiency condition requiring that in the economy graph there is a strongly connected component containing all the consumer-nodes. This completes the proof. □

4.6.4 Arrow-Debreu Markets with Succinct SPLC Utilities and Production

In this section we provide our PPAD-membership result for *succinct* separable piecewise-linear utilities (SSPLC). These are SPLC utilities, i.e., each consumer has a piecewise-lineaer utility function over different pieces (or segments) of each good, and her total utility for her bundle is additive over those utility functions. The "succinct" part comes from the fact that, in contrast with the SPLC utilities that have been studied before in the literature (e.g., see [112, 116, 220]), these functions can be accessed implicitly, given access to a boolean circuit that evaluates the function at different points. Effectively, these allows us to model functions with exponentially many pieces, while still developing a reduction which is polynomial in the size of the input, i.e., the size of those circuits. Our main theorem of the section is the following.

Theorem 4.6.4. *Computing a competitive equilibrium in an Arrow-Debreu market with SSPLC utilities and SPLC production functions is in PPAD.*

Note that we are also incorporating (explicit) SPLC productions in our result. Whether we can generalize those to SSPLC production functions is an interesting technical open problem; we discuss the challenges and why our technique falls short in this regard in Remark 35.

Markets with Production. In an Arrow-Debreu market with production $\mathcal{M}$, we have a set N of consumers, a set G of infinitely divisible goods, and a set F of firms. Let $n = |N|$, $m = |G|$, and $\ell = |F|$. We will typically use index i to refer to consumers, j or g to refer to goods and f to refer to firms. Each consumer brings an endowment $w_i = (w_{i1}, \ldots, w_{im})$ to the market, with $w_{ij} \geq 0$ for all $i \in N$ and $j \in G$. We may assume without loss of generality that for every good j, we have $\sum_{i \in N} w_{ij} = 1$, i.e., that the total endowment of each good is 1. We will use $\mathbf{x}_i = (x_{i1}, \dots, x_{im})$ to denote the vector of quantities of goods allocated to consumer $i \in N$ in $\mathcal{M}$, and we will call it the *bundle* of consumer i. Let $\mathbf{x} = (x_1, \dots, x_n)$ be the vector of such bundles. We will use $\mathbf{p} = (p_1, \dots, p_m)$ to denote the vector of *prices* in $\mathcal{M}$, one for each good $j \in G$. Prices are non-negative, so $p_j \geq 0$ for all $j \in G$. Given a vector of prices $\mathbf{p}$, the *budget* of consumer $i \in N$ is defined as $\sum_{j \in G} w_{ij} p_j$; intuitively, this is the amount of money that the consumer acquires by selling her endowment at prices $\mathbf{p}$.

Utility functions. Every consumer has a utility function $u_i \colon \mathbb{R}^m_{\geq 0} \to \mathbb{R}_{\geq 0}$ mapping a bundle $\mathbf{x}_i$ to a non-negative real number. In this section, these utilities are *succinct separable piecewise-linear concave (SSLPC)*. A utility u_i being SPLC means that it is separable over the goods and that the utility accrued from each good is given by a piecewise-linear concave function u_{ij}, that is, $u_i(\mathbf{x}_i) = \sum_{j=1}^{m} u_{ij}(x_{ij})$. The word *succint* refers to the piecewise-linear utilities concave u_{ij} being given by boolean circuits computing their slopes, thus allowing for an exponential number of changes to the value of the slope. Concretely, we assume that the utility functions are given by the following data:

1. A natural number $A \in \mathbb{N}$ such that the piecewise linear functions u_{ij} have constant slope in the interval (A, ∞).

2. A natural number $B \in \mathbb{N}$ such that the slopes of the utility functions can change only in the points $\frac{k}{B} \cdot A$ for $k \in \{0, 1, \dots, B\}$.

3. For every $i \in N$ and $j \in G$, a boolean circuit C_{ij} with $\lfloor \log_2(B) \rfloor + 1$ input bits and output bits which on input $k \in \{0, 1, \dots, B\}$ computes $1/u_{ijk}$, where u_{ijk} is the slope of u_{ij} in the interval $((k/B)A, ((k+1)/B)A)$. Note that this makes sense only if the slope is strictly positive. Also, the utility functions are concave, meaning that $u_{ij0} \geq u_{ij1} \geq \cdots \geq u_{ijB}$.

4. For every $i \in N$ and $j \in J$, if the slope of u_{ij} eventually becomes zero, then we are given a natural number D_{ij} such that the slope becomes zero in the point $(D_{ij}/B)A$.

Remark 32. The reason that we assume that the boolean circuits C_{ij} compute the inverse of the slopes is that this allows us to compute bang-per-buck ratios which is required in our construction.

Firm Shares. Each consumer $i \in N$ has a share $\theta_{if} \in [0, 1]$ of the profit of each firm $f \in F$. We assume that the profits are entirely shared among the consumers, i.e., for every firm $f \in F$, we have that $\sum_{i \in N} \theta_{if} = 1$.

Production functions. In this section we assume that the firms have *separable piecewise-linear concave* production functions given explicitly. Every firm f has a single *output good* g_f^{out} which it can produce, and for every $j \in G$, it has a piecewise-linear concave function $P_j^f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ defining the ability of f to produce g_f^{out} as a function of the good j. The overall production of firm f from bundle $\mathbf{y} = (y_1, \ldots, y_m)$ is then given by $P^f(\mathbf{y}) = \sum_{j \in G} P_j^f(y_j)$. Because of separability, we may assume that every firm has an *input good* g_f^{in}, which is the only good with which it can produce a strictly positive amount of g_f^{out}. As such the production function P^f is simply a piecewise-linear concave function. We assume that this function is given to us explicitly as follows:

1. The nonnegative real line $\mathbb{R}_{\geq 0}$ is divided into a finite number K_f of subsequent segments of length L_{fk} for $1 \leq k \leq K_f$. The length of the final segment is infinite.

2. To every segment is associated a *conversion rate* $c_{fk} \geq 0$ which is the slope of P_f in this segment, that is, c_{fk} is the rate at which f can convert good g_f^{in} to good g_f^{out} on this segment. The production function P^f being concave means that $c_{f1} \geq c_{f2} \geq \cdots \geq c_{fK_f}$.

One may wonder why we do not assume that the slopes of the production functions are given succinctly by a boolean circuit as we did for the utility functions. We discuss the problems arising from this representation later in the section.

Competitive Equilibrium. We are now ready to define the notion of a competitive equilibrium in markets with production.

Definition 4.6.5 (Competitive Equilibrium - Markets with SSPLC Utilites and SPLC Production)**.** A competitive equilibrium of an Arrow-Debreu market with SSPLC utilities and PLC production functions is a triple $(\mathbf{p}^*, \mathbf{x}^*, \mathbf{y}^*)$ consisting of non-negative prices $\mathbf{p}^*$, non-negative bundles $\mathbf{x}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*)$ and non-negative amounts of input goods $\mathbf{y}^* = (y_1^*, \ldots, y_\ell^*)$, such that

1. $p^*_{g_f^{\text{out}}} \cdot P_f(y_f^*) - p^*_{g_f^{\text{in}}} \cdot y_f^*$ is maximized, for any firm $f \in F$. ***(firm profit maximization)***

2. $u_i(\mathbf{x}_i)$ is maximized for every consumer $i \in N$, subject to $\sum_{j \in G} p_j^* \cdot x_{ij} \leq \sum_{j \in G} p_j^* \cdot w_{ij} + \sum_{f \in F} \theta_{if} \cdot (p^*_{g_f^{\text{out}}} \cdot P_f(y_f^*) - p^*_{g_f^{\text{in}}} \cdot y_f^*)$. ***(bundle optimality)***

3. $z_j^* \leq 0$, and $z_j^* p_j^* = 0$, where $z_j^* = \sum_{i \in N} x_{ij}^* + \sum_{f \in F:\, g_f^{\text{in}} = j} y_f^* - \sum_{f \in F:\, g_f^{\text{out}} = j} P_f(y_f^*) - 1$,
for every good $j \in G$. ***(market clearing)***

Condition 1 requires that at the chosen set of prices $\mathbf{p}^*$, each firm maximizes its profit, given its production functions. Condition 2 requires that at the chosen set of prices

$\mathbf{p}^*$, each consumer maximizes her utility subject to their budget constraints, where the budget consists of the amount earned from selling all the consumer's endowments $\sum_{j\in G} p_j^* w_{ij}$ and the profit share $\sum_{f\in F} \theta_{if} \cdot (p^*_{g_f^{\text{out}}} \cdot P_f(y_f^*) - p^*_{g_f^{\text{in}}} \cdot y_f^*)$ of the consumer from the production of the firms. Finally, Condition 3 is the market clearing condition, which requires that the total consumption of each good is at most the total production plus the total endowment of the consumers, and supply equals demand for all goods which are not priced at 0. As we detail later on in the section, we may in fact assume without loss of generality that in any competitive equilibrium *all the prices are positive*, and hence Condition 3 reduces to $z_j^* = 0$ for all $j \in G$. Note that in Condition 3 we have used that $\sum_{\in N} w_{ij} = 1$ for each good $j \in G$.

Sufficiency Conditions. A competitive equilibrium as defined above exists for every market $\mathcal{M}$, under some sufficiency conditions. Similarly to Section 4.6.2, we will use the set of conditions used by Maxfield [167], also used in the series of papers on market equilibria that we mentioned in the beginning of Section 4.6.

1. Consider a graph $\mathcal{G}_F(\mathcal{M})$ in which the nodes are the goods, and an edge (j, j') has weight
$$w_{jj'} = \max_{f\in F:\ j=g_f^{\text{in}} \wedge j'=g_f^{\text{out}}} c_{f1}$$
If the set is empty, the weight is defined to be 0.

   The above weight captures the fact that j' can be produced from j at conversion rate $w_{jj'}$ by some firm $f \in F$. Then, for any cycle $C = (g_0, g_1), (g_1, g_2) \dots, (g_{k-2}, g_{k-1})$ of $G_F(\mathcal{M})$ the product of the weights of the edges is less than 1, i.e., $\prod_{e\in C} \alpha_e < 1$.

   This condition is known as the ***no production out of nothing and no vacuous production*** condition. Indeed, if $\prod_{e\in C} \alpha_e > 1$, then it would be possible to increase the quantity of some good, without decreasing the quantity of any other good. The case of $\prod_{e\in C} \alpha_e = 1$ refers to the case of vacuous production, which is also disallowed in our model, similarly to the related works, e.g., see [112].

2. We will say that a consumer $i \in N$ is ***nonsatiated*** *for good* $j \in G$ if $C_{ij}(B) > 0$. We will assume that for any good $j \in G$, there exists some consumer $i \in N$ that is nonsatiated with respect to j. Similarly, we say that a firm f ***nonsatiated*** for good $j \in G$ if $j = g_f^{\text{in}}$ and $c_{fK_f} > 0$.

3. Consider the *economy graph* $\mathcal{G}_E(\mathcal{M})$ of the market $\mathcal{M}$ in which the nodes are the consumers and the firms, and there is an edge from node a to node b in node a owns/produces a good for which node b is nonsatiated. Then, $\mathcal{G}_E(\mathcal{M})$ contains a strongly connected component containing all the consumer-nodes. This condition generalizes the corresponding strong connectivity condition that we used in Section 4.6.2.

Remark 33 (Bounds on Production)**.** Condition 1 above imposes a bound on the total amount of a production of any firm $f \in F$ in a competitive equilibrium. Note that the production starts from finite endowments $\sum_{i \in N} w_{ij} = 1$ for all $j \in G$. Since no firm operates at a loss at an equilibrium, any cycle of production would violate Condition Item 1. Since there are no such cycles, production can take place along chains. The longest such chain is obviously bounded by m, and the maximum production of any firm can be bounded by some sufficiently large global constant L^c (e.g., some constant such that $L^c \geq m^m(\max_f C_f(0) + 1)^n$). See also [112] for a very similar argument. Looking ahead, this will allow us to impose "loose" upper bounds on the production and consumption in the linear programs that we will devise, without compromising the existence of a competitive equilibrium.

Optimality and bang-per-buck (BPB). Similarly to Section 4.6.1, the optimal bundles of Condition 2 in Definition 4.6.2 are characterized by their *bang-per-buck (BPB)*. Given consumer $i \in N$ and prices $\mathbf{p}$, the bang-per-buck of good good j on segment k is defined as $\text{BPB}_i(j,k) = \frac{u_{ijk}}{p_j}$.

Bounds on the prices. Recall that we have assumed that for every good j, there is some agent i that is nonsatiated with respect to j. Hence, using very similar arguments to Section 4.6.2 and Section 4.6.3, we may assume that all the prices are strictly positive.

Remark 34 (Normalized Prices)**.** Given that $p_j > 0$ for all $j \in G$, we can normalize the prices to sum to 1 without loss of generality, i.e., we may assume that for every good $j \in G$, we have that $\sum_{j \in G} p_j = 1$.

Again, in a similar manner to Section 4.6.1, we will use a parameter ε to capture the fact that if the price p_j for a good $j \in G$ is sufficiently smaller than the price $p_{j'}$ for a good $j' \in G$, then $\text{BPB}(j) > \text{BPB}(j')$. Specifically, we can compute $\varepsilon > 0$ such that

$$\text{If } p_j \leq \varepsilon \cdot p_{j'} \text{ and } u_{ijk} > 0 \text{ then } \text{BPB}_i(j,k) > \text{BPB}_i(j',k') \text{ for all } k'.$$

Because the inverses of the utilities are in the set $\{1, 2, \ldots, B\}$ by assumption, the actual utilities are in the set $\{1/B, 1/(B-1), \ldots, 1/2, 1\}$. For the above to hold, it is thus sufficient that if $p_j \leq \varepsilon p_{j'}$, then $(1/B)/p_j \geq 1/p_{j'}$. For this implication to hold, it suffices to choose $\varepsilon \leq 1/B$. Additionally, we can pick ε to be sufficiently small such that $\varepsilon < \frac{w_{ij}}{m}$ for all $i \in N$ and all $j \in G$ such that $w_{ij} > 0$. Given ε, we will impose a stricter lower bound on the prices, which will be useful later on: in particular, we will assume that for all $j \in G$, $p_j \geq \frac{\varepsilon^m}{m}$.

Expenditures. As in all of the previous sections, we do not work directly with the quantities $\mathbf{x}$, $\mathbf{y}$. Instead we work with the consumer and firm *expenditures* respectively, which are defined similarly to those of Section 4.6.2. Specifically, for every for agent i we will have a variable q_{ij} for every $j \in G$ which denotes the spending of agent i on good j, and for every f and every $1 \leq k \leq K_f$, we will have a variable s_{fk} which denotes the spending of firm f on its input good g_f^{in} on the kth segment.

Membership in PPAD: The proof of Theorem 4.6.4

We will again develop the proof in three steps, namely (a) construction of the function F and arguing that it can be represented by a linear arithmetic circuit containing linear-OPT-gates, (b) showing that the linear-OPT-gate can compute all the necessary components, and (c) arguing that a competitive equilibrium can be recovered from a fixed point of F.

The function F. We define a function $F\colon D \to D$ where

$$D = \{p \in \Delta^{m-1} : \forall j \in G, p_j \geq \frac{\varepsilon^m}{m}\} \times [0, CL+1]^{nm} \times_f [0, L]^{K_f} \tag{4.34}$$

An input to F is a tuple $(\mathbf{p}, \mathbf{q}, \mathbf{s})$ of prices, consumer expenditures and firm expenditures for raw goods, and the output is another such tuple $(\bar{\mathbf{p}}, \bar{\mathbf{q}}, \bar{\mathbf{s}})$.

For any firm f, the output spending variable $\bar{\mathbf{s}}_f = (s_{fk})_{1\leq k\leq K_f}$ is set to be the output of the linear OPT-gate for the following linear program.

Linear Program $\mathcal{P}_{\text{prod}}$

$$\begin{aligned} \text{maximize} \quad & \sum_{k=1}^{K_f} (c_{fk} p_{g_f^{\text{in}}} - p_{g_f^{\text{out}}}) \cdot v_{fk} \\ \text{subject to} \quad & 0 \leq v_{fk} \leq \min\{L_{fk}, L\} \cdot p_{g_f^{\text{in}}} \end{aligned} \tag{4.35}$$

As in the case considered in Section 4.6.2, we define

$$r_{fk} = s_{fk} + \max\{0, c_{fk} \cdot p_{g_f^{\text{out}}} - p_{g_f^{\text{in}}}\} \cdot \min\{L_{fk}, L\}.$$

Furthermore, we let $s_f = \sum_{k=1}^{K_f} s_{fk}$ and $r_f = \sum_{k=1}^{K_f} r_{fk}$. Then, for every agent i, we let the output expenditure $\mathbf{q}_i$ be the output of the linear opt-gate of the following linear program.

Linear Program $\mathcal{P}_{\text{con}}$

$$\begin{aligned} \text{minimize} & \sum_{j\in G} \left(p_j \cdot C_{ij}\left(\frac{q_{ij}}{p_j}\right) + (B+1)\cdot H\left(q_{ij} - \frac{D_{ij}}{B} A p_j\right)\right) \cdot v_{ij} \\ \text{subject to} & \sum_{j\in G} v_{ij} = \sum_{j\in G} w_{ij} p_j + \sum_{f\in F} \theta_{if} \cdot (r_f - s_f) \\ & 0 \leq v_{ij} \leq (C\cdot L+1),\ \forall j \in G \end{aligned} \tag{4.36}$$

The first term $p_j \cdot C_{ij}(q_{ij}/p_j)$ of the coefficient in front of v_{ij} is the linear pseudo-circuits from Proposition 5 computing the product of p_j and the piecewise constant correspondence represented by (C_{ij}, B). This term should represent the inverse bang-per-buck ratio of agent i for good j when her spending on the good is q_{ij}. The second term in the coefficient in front of v_{ij} in the objective function only appears if the slope of the PLC function that agent i has for good j eventually becomes zero. This term is meant to ensure that in an optimal solution, an agent does not spend money on a segment where the slope is zero.

Excess Expenditure and the Feasibility Program for the prices. Finally, for every $j \in G$, similarly to Section 4.6.2 and Section 4.6.3, we define the *excess expenditure* of good j as

$$e_j = \sum_i q_{ij} + \sum_{f:\, g_f^{\text{in}}=j} s_f - \sum_{f:\, g_f^{\text{out}}=j} r_f - p_j.$$

As in previous sections, we let the output price vector $\bar{p}$ be the output of the following feasibility program.

Feasibility Program Q

$$\begin{aligned} & e_j < e_{j'} \Rightarrow p_j \leq \varepsilon \cdot p_{j'},\ \forall j, j' \in G \\ & p_j \geq \frac{\varepsilon^m}{m},\ \forall j \in G \\ & \sum_{j \in G} p_j = 1 \end{aligned} \tag{4.37}$$

Computation by the linear-OPT-gate. Next, we argue that the solutions to the linear programs and the feasibility program can be computed by our linear-OPT-gate. The subgradients of the LPs for production and consumption can be computed by linear circuits. The main part of the construction is the linear pseudo-circuit from Proposition 5, computing the correspondence $p_j \cdot C_{ij}(q_{ij}/p_j)$. For the feasibility program, the arguments that it is of the correct form and that it is solvable are identical to those of Lemma 23 (and hence also of Lemma 21), noting the updated definition of the expenditure e_j.

Fixed points. Suppose that $(\mathbf{p}, \mathbf{q}, \mathbf{s})$ is a fixed point of F. Let $f \in F$ be some firm. As $\mathbf{s}_f$ is a solution to LP (4.35), we have that $s_{fk} > 0$ only if $c_{fk} \cdot p_{g_f^{\text{out}}} \geq p_{g_f^{\text{in}}}$. Furthermore, if the inequality is strict, then the firm produces using all of that segment. We conclude that $\mathbf{s}$ is an optimal production vector at prices $\mathbf{p}$. Next, we prove the following:

Claim 11. *Let $i \in N$ and let $x_{ij} = q_{ij}/p_j$ for every $j \in G$. Then $\mathbf{x}_i$ is an optimal bundle for agent i at prices* $\mathbf{p}$.

Proof. First we show that if the slope of u_{ij} becomes zero at some point, then $q_{ij} \leq \frac{D_{ij}}{B} \cdot A \cdot p_j$. If this were not the case, then the coefficient in front of v_{ij} would be at least $B+1$. However, by assumption there is a good j' for which agent i is nonsatiated, and the coefficient in front of $v_{ij'}$ is at most

$$p_{j'} \cdot C_{ij'}\left(\frac{q_{ij'}}{p_{j'}}\right) \leq p_{j'} \cdot B \leq B < B+1.$$

Because $\mathbf{q}_i$ is an optimal solution to LP (4.35), this would then imply that $q_{ij} = 0 \leq \frac{D_{ij}}{B} \cdot A \cdot p_j$. This means that any agent i can only spend money on a good j up until the point where the inverse of the bang-per-buck ratio is no longer well-defined. Now the result follows, because all the goods $j \in G$ such that $q_{ij} > 0$ must have the minimal coefficient. From this it follows that agent i has bought the goods in a greedy manner according to the bang-per-buck ratio, meaning that $\mathbf{q}_i$ represents an optimal spending for agent i. □

What remains to show is that the market clears, that is, $e_j = 0$ for all $j \in G$. As in previous sections, we can use a standard calculation to show that $\sum_{j \in G} e_j = 0$. Hence, it suffices to argue that $e_j \geq 0$ for all $j \in G$. We define:

- $J = \{j \in G \colon e_j \leq e_{j'}$ for all $j' \in G\}$
- $N_J = \{i \in N \colon$ there exists $j \in J$ such that i is nonsatiated for good $j\}$
- $F_J = \{f \in F \colon$ there exists $j \in J$ such that f is nonsatiated for good $j\}$

Assume by contradiction that $e_j < 0$ for $j \in J$. As $\sum_{j \in G} e_j = 0$, this implies that there must exists a good j' such that $e_{j'} > 0$, meaning that $J \subsetneq G$. The contradiction will be obtain via the following two claims, very similar to Claim 7 and Claim 8. Using an identical reasoning to that of Lemma 24, the two claims together will establish that the economy graph $\mathcal{G}_E(\mathcal{M})$ is not strongly connected, violating the corresponding sufficiency condition.

Claim 12. *If $i \in N_J$ and $j' \in G \setminus J$, it holds that $w_{ij'} = 0$.*

Proof. Assume for contradiction that $i \in N_J$ and j' $G \setminus J$ and that $w_{ij'} > 0$. As in previous sections, this implies that we can lower bound the budget of agent i by $(CL+1)\sum_{j \in J} p_j$ by picking $\varepsilon \leq \frac{w_{min}}{m \cdot (CL+1)}$. By assumption, i is nonsatiated with respect to some $j \in G$. As $p_j \leq \varepsilon p_{j''}$ for all $j'' \in G \setminus J$, we get by choice of ε that i will spend her entire budget on goods in J. Hence, there is some $j \in J$ such that $q_{ij} \geq (CL+1)p_j$. This implies that

$$e_j \geq (CL+1)p_j - \sum_{f \colon g_f^{\text{out}} = j} r_f - p_j \tag{4.38}$$

Now for any f with $g_f^{\text{in}} = j$, we have that $r_f = \sum_{k=1}^{K_f} r_{fk} \leq \sum_{k=1}^{K_f} Lc_{fk}p_j \leq (K_f \cdot \max_{f \in F} c_{f1})Lp_j$. By picking $C \geq |F| \cdot \max_f K_f \cdot \max_f c_{f1}$, we obtain the contradiction that $e_j \geq 0$. We conclude that if $i \in N_J$ and $j' \in G \setminus J$, then $w_{ij'} = 0$. □

Claim 13. *If $f \in F_J$, then $g_f^{out} \in J$.*

Proof. Let $f \in F_J$ and let $j = g_f^{\text{in}}$. Assume for contradiction that $g_f^{\text{out}} \notin J$. Then $p_j \leq \varepsilon p_{g_f^{\text{out}}}$, meaning that f will produce fully if $\varepsilon > 0$ is chosen sufficiently small. Hence $s_f \geq s_{fK_f} = Lp_j$, and so

$$0 > e_j \geq (L-1) \cdot p_j - \sum_{f' :\, g_{f'}^{\text{out}} = j} r_{f'} \tag{4.39}$$

If no firm produces j, then we reach a contradiction by choosing $L \geq 1$. The above inequality implies that there must exist a firm f_1 such that $g_{f_1}^{\text{out}} = j$ and $r_{f_1} \geq \frac{L-1}{\ell} p_j$. Let $j_1 = g_{f_1}^{\text{in}}$ and note that $j_1 \in J$. As in previous sections, we may thus bound $\frac{L-1}{\ell} p_j \leq r_{f_1} \leq c_{f_1 1} s_{f_1}$. If any production occurs, then $c_{f_1 1} p_j \geq p_{j_1}$. These two inequalities combine to show that $s_{f_1} \geq \frac{L-1}{\ell \cdot c_{f_1 1}^2} p_{j_1}$. Using this, we may again bound

$$0 > e_{j_1} \geq \Big(\frac{L-1}{\ell \cdot c_{f_1 1}^2} - 1 \Big) \cdot p_{j_1} - \sum_{f' :\, g_{f'}^{\text{out}} = j_1} r_{f'} \tag{4.40}$$

Again, if j_1 is not produced, then we may pick L large enough to reach a contradiction. Otherwise we repeat the previous argument. This can happen at most m times, because of the *no assumption out of nothing* assumption. □

Beyond SSPLC Utilities?

We conclude the section with two remarks. The first one briefly discusses the challenges of using the machinery of Section 4.3.4 for SSPLC productions also.

Remark 35. The reason for not allowing a succinct representation of the production functions is that this seemingly necessitates multiplying a piecewise-linear function by an input variable of the circuit. A natural idea would be to have the cumulative sum of conversion rates given succinctly by a boolean circuit C'. However, in order to compute the revenue of firm f given spending s_f, one would then have to compute the piecewise-linear function $p_{g_f^{\text{out}}} \cdot C'(s_f / p_{g_f^{\text{in}}})$ which is something that is not captured by the results in Section 4.3.4.

The second remark regards the possibility of defining a class of "succinct Leontief-free functions", similarly to the case of SSPLC.

Remark 36. One might hope to define a class which captures the case of Leontief-free utility and production functions and the case of SSPLC utility functions and SPLC production functions simultaneously. For the utility functions, one approach would be to consider utility functions u^i given by a finite list K_i of tranches $\{s_k^i\}_{k \in K_i}$. Associated with a tranche s_k^i would again be a number $L_k^i \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ which is an upper bound for the total amount of utility that can be accrued from this tranche. Also, for every good j, instead of having a single number u_{jk}^i which is the rate at which good j

provides utility for agent i on tranche k, we would have that j would provide utility for agent i on segment k according to a piecewise-linear concave function U^i_{jk} given to us succinctly in the form of a boolean circuit. Such a class of utility functions would generalize both Leontief-free utility functions and SSPLC utility functions. However, because the rates at which goods provide utility is non-constant, it is unclear how to obtain constraints that mimic the constraint $\sum_{j \in G} U^i_{jk}(x^i_{jk}) \leq L^i_k$ after transforming the variables via Gale's substitution (Remark 22).

4.7 Pacing Equilibria in Auto-Bidding Auctions

In this section, we consider environments which we will broadly refer to as "auto-bidding auctions". In these, a set of items are sold separately to a set of buyers via parallel single-item auctions (e.g., first-price auctions or second-price auctions). To participate in these auctions, each buyer *i scales* her valuations for the items by the same multiplier α_i, which is known as a *pacing multiplier*, under some constraints on her expenditure, typically provided by budgets, or return-on-investment (ROI) thresholds. The objective is to find a *pacing equilibrium* (see Definitions 4.7.1 and 4.7.3), i.e., a vector of pacing multipliers and allocations which ensure that the auction is run correctly and that it satisfies the spending constraints of all the buyers simultaneously.

In Section 4.7.1 we consider the former case, i.e., that of single-item auctions with budget constraints. For the case of first-price auctions, [33] showed that a pacing equilibrium always exists and can be computed in polynomial time via a convex program. Later on Conitzer et al. [61] showed that in this case the pacing multipliers are unique in every pacing equilibrium, and presented an interesting connection between pacing equilibria and solutions of the well-known Eisenberg-Gale convex program for quasi-linear utilities [60]. For second-price auctions, the existence of a pacing equilibrium was established in [62]. Chen et al. [57] later on proved that the problem of computing a pacing equilibrium in these auctions is in PPAD, which also implied that a solution described by rational numbers also exists. We provide an alternative PPAD-membership result for the same problem. Compared to the proof of Chen et al. [57], our proof makes uses of our linear-OPT-gate and is *significantly* simpler. We highlight the main ideas of our proof and how they differ from those of Conitzer et al. [62] and Chen et al. [57] in Section 4.7.1 below. We remark that Chen et al. [57] also showed the problem to be PPAD-hard, thus establishing its PPAD-completeness.

In Section 4.7.2, we also consider the case of (on average) ROI-constrained buyers, in the model introduced by Li and Tang [162]. The existence of pacing equilibria for this case was proven by Li and Tang [162], following a very similar approach to that of Conitzer et al. [62]. Our proof follows along very much the same lines as the one we present for the case of budget-constrained buyers in Section 4.7.1, and is again significantly simpler. Interestingly, it turns out that our linear-OPT-gate cannot be used to obtain PPAD-membership of the problem in this case. Still, using the OPT-gate for FIXP developed by [101], we prove membership of the problem in the class FIXP. This is the first membership result for this variant of the problem; we provide more details in Section 4.7.2. In Section 4.7.2 we show that membership in FIXP (rather than PPAD) is necessary, because pacing equilibria in this case can be irrational. Whether the problem of computing pacing equilibria in second-price auctions for ROI-constrained buyers is actually FIXP-complete is an intriguing open problem. We note that for a notion of approximate pacing equilibria, a PPAD-hardness result is known from [162].

4.7.1 Pacing Equilibria in Auctions with Budgets

Following Chen et al. [57], we will refer to the auction market environment as a *second-price pacing game (SPPG)* $\mathcal{G}$, consisting of

- a set N of buyers, with $n = |N|$,
- a set G of items, with $m = |G|$,
- a set of vectors of *valuations* $\mathbf{v}_i = (v_{i1}, \dots v_{im})$ for every buyer $i \in N$, where v_{ij} denotes the value of buyer $i \in N$ for item $j \in G$; without loss of generality we assume that for all $j \in G$, there exists some $i \in N$ such that $v_{ij} > 0$,
- a vector of *budgets* $\mathbf{B} = (B_1, \dots, B_n)$, such that $B_i > 0$ for every buyer $i \in N$.

For each buyer $i \in N$, there is a *pacing multiplier* $\alpha_i \in [0, 1]$, and let $\alpha = (\alpha_1, \dots, \alpha_n)$ be the vector of those multipliers. Given α, the *bid* of buyer i on item $j \in G$ is given by $\alpha_i \cdot v_{ij}$, i.e., every buyer *scales* their value v_{ij} by the multiplier α_i. The items are sold in parallel second-price auctions, which implies that

1. the item is allocated to the buyer with the highest bid, breaking ties using some tie-breaking rule in case there are multiple such buyers. Let $h_j(\alpha) = \max_{i \in N} \alpha_i v_{ij}$ be the highest bid on item $j \in G$.
2. the price of item $j \in G$ is the second-highest bid. Let $p_j(\alpha) = \min_{i \in N} \max_{i' \in N, i \neq i'} \alpha_{i'} v_{i'j}$ be that price.

Another way to interpret Condition 1 above is that each one of the (potential) winners achieves a fraction of the item, which can be interpreted as the probability that that buyer is the final winner of the auction for that item. Formally, let $W_j(\alpha) = \{i \in N : \alpha_i v_{ij} = h_j(\alpha)\}$ be the set of possible winners (those with the highest bid) for item $j \in G$. Each buyer $i \in N$ receives an allocation x_{ij} of item $j \in N$, with the restriction that $x_{ij} = 0$ for all $i \in N \setminus W_j(\alpha)$.

Pacing Equilibria. We are now ready to define the notion of a pacing equilibrium. Informally, a pacing equilibrium is a set of pacing multipliers and allocations that satisfy some constraints that guarantee the validity of the auction and that the buyers are in a sense meeting their maximum buying capabilities. Following Conitzer et al. [61, 62], we formally refer to those equilibria as *second-price pacing equilibria.*

Definition 4.7.1 (Second-price Pacing Equilibrium (SPPE))**.** A pair $(\alpha, \mathbf{x})$, with $\alpha \in [0,1]^n$ and $\mathbf{x} \in [0,1]^{nm}$, is a *second-price pacing equilibrium (SPPE)* of a SPPG $\mathcal{G}$, if the following conditions hold:

I.a. $\sum_{i \in N} x_{ij} \leq 1$, for all $j \in G$, ***(feasible allocation)***

I.b. $x_{ij} > 0 \Rightarrow \alpha_i v_{ij} = h_j(\alpha)$, for all $i \in N$ and $j \in G$, ***(only highest bids win)***

I.c. $h_j(\alpha) > 0 \Rightarrow \sum_{i \in N} x_{ij} = 1$, for all $j \in G$, ***(full allocation of items with positive bids)***

I.d. $\sum_{j \in G} x_{ij} p_j(\alpha) \leq B_i$, for all $i \in N$, ***(budget constraints)***

I.e. $\sum_{j \in G} x_{ij} p_j(\alpha) < B_i \Rightarrow \alpha_i = 1$, for all $i \in N$, ***(maximum pacing)***

We remark that Definition 4.7.1 requires the flexibility to choose the way the items are allocated to the possible winners; this is in fact necessary to guarantee the existence of a SPPE for every SPPG. As we mentioned earlier, a SPPE is always guaranteed to exist [57, 62]. Our PPAD-membership proof also provides an alternative, easier proof of existence. We provide a high-level overview of the proof, as well as a comparison to the existing proofs in the next section below.

Features of our proof

Previous proofs of existence and PPAD-membership. The existence of SPPE in SPPGs was first established by Conitzer et al. [62]. Their proof proceeds by defining an (ε, H)-*smoothed pacing game*, for $\varepsilon > 0$ and $H > 0$, which is derived from the SPPG $\mathcal{G}$ as follows:

- There is an artificial *reserve bid* of 2ε on all items.
- The set of possible winners $W_j(\alpha)$ (i.e., those buyers $i \in N$ with $x_{ij} > 0$ for item $j \in G$) contains all buyers with *almost* the highest bid (i.e., a bid of at least $h_j(\alpha) - \varepsilon$). Each buyer in W_j receives a specific fraction of item j (i.e., x_{ij} is fixed for all $i \in N$ and $j \in N$, as a function of the α_i's.) and pays a specific price s_{ij} (see [62], Definition 3 for the exact allocations and prices).
- There is an artificial item j_a of unlimited supply. Buyer $i \in N$ receives a quantity α_i of that good and values that good at $v_{ij_a} = 2\varepsilon$.
- Every buyer $i \in N$ has a utility which is a step function depending on the budget B_i, the price s_{ij} she pays for quantities of item j, α_i, ε and the parameter H in the definition of the (ε, H)-smoothed pacing game (see [62], Definition 3 for the exact utility function).

The reason for introducing all these seemingly convoluted conditions is that they ensure that the (ε, H)-smoothed pacing game satisfies a set of desired properties, namely (a) compactness and convexity of the strategy space, (b) continuity of the utility function in all strategies and (c) quasi-concavity of the utility function in a buyer's own strategy. With these at hand, one can then invoke the Debreu-Fan-Glicksberg theorem for continuous games [1952], which we presented in Section 4.4.3, to establish the existence of a Nash equilibrium.

The proof of Conitzer et al. [62] proceeds by proving Properties (a-c) above, and then obtains the existence of a Nash equilibrium for the (ε, H)-smoothed pacing game. To obtain the existence of a SPPE, they consider a sequence of Nash equilibria for (ε, H)-smoothed pacing games, and show that as ε converges to 0 and H converge to infinity, a SPPE is obtained as a limit point.

Besides the overhead of defining (ε, H)-smoothed pacing games and establishing their properties, the proof uses off-the-shelf existence theorems and limit arguments. Therefore, it is not surprising that it cannot be used to obtain PPAD-membership, or even to establish that rational solutions are always possible. Both of these properties were established via an alternative proof provided by Chen et al. [57], which we describe briefly next.

Similarly in spirit to [62], Chen et al. [57] define an *approximate* version of a SPPE, called a (δ, γ)-SPPE, with $\delta, \gamma > 0$, in which

- Condition I.b. in Definition 4.7.1 has been substituted by $x_{ij} > 0 \Rightarrow a_i v_{ij} = (1-\delta)h_j(\alpha)$, for all $i \in N$ and all $j \in G$.

- Condition I.e. in Definition 4.7.1 has been substituted by $\sum_{j\in G} x_{ij} p_j(\alpha) < (1-\gamma)B_i \Rightarrow \alpha_i \geq (1-\gamma)$, for all $i \in N$.

In other words, they consider an approximate SPPE in which the set of winners $W_j(\alpha)$ contains buyers that have approximately the highest bid, and in which a buyer that does not spend almost all of her budget uses almost a maximum pacing multiplier. On top of that, they define a variant of (δ, γ)-SPPE, which they refer to as *smooth* (δ, γ)-SPPE in which the allocation rule is specified, and is in fact similar to that used by Conitzer et al. [62] in their definition of a (ε, H)-smoothed pacing game defined above. By definition, a smooth-(δ, γ)-SPPE is (δ, γ)-SPPE. Their PPAD-membership proof then proceeds in three steps:

- They show the PPAD-membership of smooth (δ, γ)-SPPE via a reduction to the computational version of Sperner's lemma [204]. This implies the existence of a (δ, γ)-SPPE.

- They devise an intricate rounding procedure, which converts (δ, γ)-SPPE into γ-SPPE, i.e., (δ, γ)-SPPE for which $\delta = 0$, and establish the correctness of this procedure via a series of arguments.

- They employ a general technique (introduced by [86]) of converting approximate solutions of inversely exponential precision to exact solutions via a carefully constructed linear program. They use this to convert a γ-SPPE into a SPPE.

Our proof. Our proof is markedly different from the aforementioned ones, and conceptually much simpler, while at the same time obtaining both existence and PPAD-membership, and as a result establishing the existence of SPPE described by rational numbers. Contrary to [62] and [57], we do not need to define relaxations of the SPPE. Instead, we apply Gale's substitution (see Remark 22), i.e., the standard change of variables from allocations x_{ij} to expenditures $q_{ij} = x_{ij}p_j(\alpha)$, see Section 4.7.1 below.[17] This is in fact the very same change of variables that we used in Section 4.6 for competitive markets, and serves as a means to escape having to deal with quantities of the form $x_{ij}p_j(\alpha)$, as such multiplications cannot be handled by linear arithmetic circuits.

Membership in PPAD then follows from constructing a simple function F whose fixed points will be the SPPE. For each buyer $i \in N$, the expenditures q_{ij} will be obtained as the solutions to a linear program which can be handled by our linear-OPT-gate, and the pacing multipliers will be the solutions to a single equation involving the inputs and the outputs of the function. Arguing that a fixed point of F indeed corresponds to a SPPE reduces to establishing that Conditions I.a. to I.e. of Definition 4.7.1 are satisfied, which turns out to be rather simple.

Preprocessing

Before we present our proof, we will first modify and simplify the constraints of Definition 4.7.1, to make them amenable to the use of our linear-OPT-gate. En route to that, we will prove that without loss of generality, we may assume that in any SPPE $(\alpha, \mathbf{x})$, both the pacing multipliers α_i and the prices $p_j(\alpha)$ can be assumed to be bounded away from 0, which we will use later on.

Lower bounds on the pacing multipliers and prices. Below we prove simple lower bounds on the value of the pacing multipliers α_i^* and the prices $p_j(\alpha^*)$ in any SPPE $(\alpha^*, \mathbf{x}^*)$.

Lemma 29. *Let $(\alpha^*, \mathbf{x}^*)$ be any SPPE. Then, for any $i \in N$, we have that $\alpha_i^* \in [\ell_i, 1]$, where $\ell_i = \min\{1, B_i / \sum_{j\in G} v_{ij}\}$.*

Proof. If $\alpha^* = 1$ then the lemma holds trivially. Otherwise, by Condition I.e., we have that

$$B_i = \sum_{j\in G} x_{ij}^* p_j(\alpha^*) \le \sum_{j\in G : x_{ij}^* > 0} h_j(\alpha^*) = \sum_{j\in G : x_{ij}^* > 0} \alpha_i^* v_{ij} \le \alpha_i^* \sum_{j\in G} v_{ij},$$

where the first inequality above holds by the fact that $x_{ij}^* p_j(\alpha^*) \le x_{ij}^* h_j(\alpha^*) \le h_j(\alpha^*)$, since $h_j(\alpha^*) \ge p_j(\alpha^*)$ by the definition of the auction, and the second equation holds by Condition I.b.. □

Lemma 30. *Let $(\alpha^*, \mathbf{x}^*)$ be any SPPE. Then, for any $j \in G$, we may assume without loss of generality that $p_j(\alpha^*) > 0$.*

[17]Interestingly, Conitzer et al. [61] also apply this change of variables in their Mixed-Integer Linear Programming (MILP) formulation (see Section 6.3 in their paper), but not in the existence proof.

Proof. Assume that there is some item j' for which $p'_j(\alpha^*) = 0$. For any choice of pacing multipliers α^*, i.e., in any PE, we may choose an arbitrary allocation of the good j' for which $x_{ij'} > 0$ only for buyers $i \in W_j(\alpha)$. This way, Conditions I.b. and I.c. of Definition 4.7.1 are satisfied. Since $p'_j(\alpha^*) = 0$, the term for j' contributes 0 to the sum $\sum_{j \in G} x^*_{ij} p_j(\alpha^*)$ for every buyer $i \in N$, and hence Conditions I.d. and I.e. are unaffected. Condition I.a. is trivially satisfied as well. Therefore, given any SPPE $(\alpha^*, \mathbf{x}^*_{-j})$ for items $j \in G \setminus \{j'\}$, we get the same SPPE, together with item j' allocated as described above. □

Change of variables. Looking at the conditions of Definition 4.7.1, we observe that in Condition I.d., the allocation x_{ij} is multiplied by the price $p_j(\alpha)$. Looking ahead, this condition will appear as a constraint in a linear program $\mathcal{P}$ that we will aim to solve via our linear-OPT-gate, as part of a linear arithmetic circuit. This linear program would have x_{ij} as the variables, but $p_j(\alpha)$ would be gate inputs in the context of the linear-OPT-gate. Hence, such a constraint could not be handled by the linear-OPT-gate. To remedy this, we work in a very similar fashion as we did in Section 4.6, via applying Gale's substitution (Remark 22). Namely, we will let $q_{ij} = x_{ij} p_j(\alpha)$ be the *expenditure* of buyer $i \in N$ on item $j \in G$. Additionally, Lemma 29 implies that $h_j(\alpha^*) > 0$ for all items $j \in G$. In turn, this implies that Conditions I.a. and I.c. can be combined into a single condition, namely

$$\sum_{i \in N} x_{ij} = 1, \text{ for all } j \in G.$$

Putting everything together, we have the following equivalent definition of a *second-price pacing expenditure equilibrium (SPPEE)*, from which a SPPE can be straightforwardly recovered.

Definition 4.7.2 (Second-price Pacing Expenditure Equilibrium (SPPEE))**.** A pair $(\alpha, \mathbf{q})$, with $\alpha \in [0,1]^n$ and $\mathbf{q} \in [0,1]^{nm}$, is a *second-price pacing expenditure equilibrium (SPPEE)* of an SPPG $\mathcal{G}$, if the following conditions hold:

II.a. $\sum_{i \in N} q_{ij} = p_j(\alpha)$, for all $j \in G$, ***(feasible expenditure)***

II.b. $q_{ij} > 0 \Rightarrow \alpha_i v_{ij} = h_j(\alpha)$, for all $i \in N$ and $j \in G$, ***(only highest bids spend)***

II.c. $\sum_{j \in G} q_{ij}(\alpha) \leq B_i$, for all $i \in N$, ***(budget constraints)***

II.d. $\sum_{j \in G} q_{ij} < B_i \Rightarrow \alpha_i = 1$, for all $i \in N$. ***(maximum pacing)***

Membership in PPAD

We are now ready to prove the main result of the section.

Theorem 4.7.1. *Computing a SPPE of any SPPG $\mathcal{G}$ is in PPAD.*

Following our standard methodology, we will develop the proof in three steps, namely (a) construction of the function F and arguing that it can be computed by a linear arithmetic circuit containing linear-OPT-gates, (b) showing that the linear-OPT-gate can compute all the necessary components, and (c) arguing that a SPPE can be recovered from a fixed point of F.

The function F. We define the function $F : D \to D$ with domain $D = \prod_{i\in N}[l_i, 1] \times \prod_{i\in N}\prod_{j\in G}[0, v_{ij}]$. Let $(\alpha, \mathbf{q})$ and $(\alpha^*, \mathbf{q}^*)$ denote inputs and outputs to F respectively. For any item $j \in G$, the vector of expenditures $\mathbf{q}_j^* = (q_{1j}^*, \ldots, q_{nj}^*)$ is obtained as the output of the following linear program.

Linear Program $\mathcal{P}$

$$\begin{aligned} \text{maximize } & \sum_{i\in N}(\alpha_i v_{ij})q_{ij} \\ \text{subject to } & \sum_{i\in N} q_{ij} \le p_j(\alpha) \\ & q_{ij} \ge 0,\ \forall i \in N \end{aligned}$$

The pacing multipliers will be obtained via the following equation

$$\alpha_i^* = \max\left\{\frac{B_i}{\sum_{j\in G} v_{ij}}, \min\left\{1, \alpha_i + B_i - \sum_{j\in G} q_{ij}\right\}\right\} \tag{4.41}$$

Equation (4.41) uses max, min and addition operations on the variables, and hence can be computed by a linear arithmetic circuit. Similarly, recall that $p_j(\alpha) = \min_{i\in N}\max_{i'\in N, i\neq i'} \alpha_{i'} v_{i'j}$, and hence the linear program $\mathcal{P}$ can also be computed by a linear arithmetic circuit.

Computation by the linear-OPT-gate. In the next lemma, we argue that the output $\mathbf{q}_j^*$ of the linear program $\mathcal{P}$ can be computed by the linear-OPT-gate.

Lemma 31. *Consider the linear program $\mathcal{P}$. An optimal solution to $\mathcal{P}$ can be computed by a linear-OPT-gate.*

Proof. We will argue that $\mathcal{P}$ satisfies all the properties required for the linear-OPT-gate, as highlighted in Section 4.3.2. The feasible domain $[0, 1]^n$ of $\mathcal{P}$ is non-empty and bounded, and the gate inputs $p_j(\alpha)$ appear only on the right-hand side of the constraints. The subgradient of the objective function is a linear function, and hence can be computed by a linear pseudo-circuit. □

Fixed points lead to SPPE. We will show that a fixed point $(\alpha^*, \mathbf{q}^*)$ of F is a SPPEE (Definition 4.7.2); a SPPE can then be straightforwardly recovered by setting $x_{ij} = q_{ij}/p_j(\alpha^*)$ for all $i \in N$ and $j \in G$, since all the prices can be assumed to be non-zero by Lemma 30. In particular, we show that $(\alpha, \hat{\mathbf{q}}) = (\alpha^*, \mathbf{q}^*)$ satisfies Conditions II.a. to II.d. of Definition 4.7.2.

1. Let $j \in G$ be any item. Since $\mathbf{q}_j$ is an optimal solution to linear program $\mathcal{P}$, and since $\alpha_i v_{ij}$ is positive for all $i \in N$, the "payment" constraint of $\mathcal{P}$ must be tight, i.e., $\sum_{i \in N} q^*_{ij} = p_j(\alpha) = p_j(\alpha^*)$. Hence Condition II.a. is satisfied.

2. Consider any buyer $i \in N$ and item $j \in N$ such that $q^*_{ij} > 0$. By the optimality of $\mathbf{q}^*_j$, $\alpha_i v_{ij}$ must be maximum, i.e., $\alpha_i v_{ij} = \alpha^*_i \cdot v_{ij} = h_j(\alpha^*)$. Hence Condition II.b. is satisfied.

3. Let $i \in N$ be any buyer. Assume by contradiction that Condition II.c. is violated, i.e., that $B_i < \sum_{i \in N} q^*_{ij}$. Since $\alpha_i = \alpha^*_i$, Equation (4.41) implies that $\alpha^*_i = B_i / \sum_{j \in G} v_{ij}$. However, since $\mathbf{q}^*_j$ is a feasible solution to the linear program $\mathcal{P}$, we have

$$\begin{aligned}\sum_{j \in G} q^*_{ij} &= \sum_{j \in G : q^*_{ij} > 0} q^*_{ij} = \sum_{j \in G : \alpha^*_i v_{ij} = h_j(\alpha^*)} q^*_{ij} \le \sum_{j \in G : \alpha^*_i v_{ij} = h_j(\alpha^*)} p_j(\alpha^*) \\ &\le \sum_{j \in G : \alpha^*_i v_{ij} = h_j(\alpha^*)} h_j(\alpha^*) = \sum_{j \in G : \alpha^*_i v_{ij} = h_j(\alpha^*)} \alpha^*_i v_{ij} \le \sum_{j \in G} \alpha^*_i v_{ij} \\ &= \frac{B_i}{\sum_{j \in G} v_{ij}} \sum_{j \in G} v_{ij} = B_i,\end{aligned}$$

   where in the calculations above:

   - the second equation follows from Condition II.b., which we we established in Item 2 above,
   - the first inequality follows from the "payment" constraints of the linear program $\mathcal{P}$, and
   - the third inequality follows from the fact that $h_j(\alpha^*) \ge p_j(\alpha^*)$ by definition.

   We have reached a contradiction.

4. Consider some buyer $i \in N$ such that $\sum_{j \in G} q^*_{ij} < B_i$. Since $\alpha_i = \alpha^*_i$, Equation (4.41) in that case implies that $\alpha^*_i = 1$, and hence Condition II.d. is satisfied.

This completes the proof.

4.7.2 Pacing Equilibria in ROI-constrained Auctions

We now turn our attention to market environments where the total expenditure of each bidder is not constrained by a fixed budget, but by a constraint on (average) return-on-investement (ROI), in the setting proposed by Li and Tang [162]. For consistency with Section 4.7.1, we will refer to these environments as *second-price pacing ROI games (SPPRG)*. A SPPRG $\mathcal{G}$ consists of

- a set N of buyers, with $n = |N|$,
- a set G of items, with $m = |G|$,
- a set of vectors of *valuations* $\mathbf{v}_i = (v_{i1}, \dots v_{im})$ for every buyer $i \in N$, where v_{ij} denotes the value of buyer $i \in N$ for item $j \in G$; without loss of generality we assume that for all $j \in G$, there exists some $i \in N$ such that $v_{ij} > 0$.

Similarly to Section 4.7.1, for each buyer $i \in N$ there is a pacing multiplier α_i and the buyer's bid on item j is $\alpha_i \cdot v_{ij}$. For this setting to be meaningful, following [162], we will not constrain the pacing multipliers to lie in $[0,1]$, but in $[1,A]$ for some constant $A \in \mathbb{R}, A \geq 1$. Again, the items are sold in parallel second-price auctions, which implies that

1. the item is allocated to the buyer with the highest bid, breaking ties using some tie-breaking rule in case there are multiple such buyers. Let $h_j(\alpha) = \max_{i \in N} \alpha_i v_{ij}$ be the highest bid on item $j \in G$.
2. the price of item $j \in G$ is the second-highest bid. Let $p_j(\alpha) = \min_{i \in N} \max_{i' \in N, i \neq i'} \alpha_{i'} v_{i'j}$ be that price.

Again, let $W_j(\alpha) = \{i \in N : \alpha_i v_{ij} = h_j(\alpha)\}$ be the set of possible winners (those with the highest bid) for item $j \in G$. Each buyer $i \in N$ receives an allocation x_{ij} of item $j \in N$, with the restriction that $x_{ij} = 0$ for all $i \in N \setminus W_j(\alpha)$.

ROI Pacing Equilibria. Next, we define the notion of a second-price ROI pacing equilibrium.[18] Note that the only significant difference between Definition 4.7.3 and Definition 4.7.1 of Section 4.7.1 is that the budget constraints have now been replaced by ROI constraints.

Definition 4.7.3 (Second-price ROI Pacing Equilibrium (SPRPE))**.** A pair $(\alpha, \mathbf{x})$, with $\alpha \in [1,A]^n$ and $\mathbf{x} \in [0,1]^{nm}$, is a *second-price ROI pacing equilibrium (SPRPE)* of an SPPRG $\mathcal{G}$, if the following conditions hold:

III.a. $\sum_{i \in N} x_{ij} \leq 1$, for all $j \in G$, ***(feasible allocation)***

III.b. $x_{ij} > 0 \Rightarrow \alpha_i v_{ij} = h_j(\alpha)$, for all $i \in N$ and $j \in G$, ***(only highest bids win)***

III.c. $h_j(\alpha) > 0 \Rightarrow \sum_{i \in N} x_{ij} = 1$, for all $j \in G$, ***(full allocation of items with positive bids)***

III.d. $\sum_{j \in G} x_{ij} p_j(\alpha) \leq \sum_{j \in G} x_{ij} v_{ij}$, for all $i \in N$, ***(ROI constraints)***

III.e. $\sum_{j \in G} x_{ij} p_j(\alpha) < \sum_{j \in G} x_{ij} v_{ij} \Rightarrow \alpha_i = A$, for all $i \in N$. ***(maximum pacing)***

[18]Li and Tang [162] refer to these equilibria as "auto-bidding equilibria"; we choose the term "second-price ROI pacing equilibria" for consistency with the results of Section 4.7.1.

Li and Tang [162] established that a SPRPE always exists. Their proof follows closely that of Conitzer et al. [62] for second-price pacing equilibria in auctions with budgets, which we referred to in Section 4.7.1. In particular, they also define a very similar (ε, H)-smoothed game that satisfies the properties required for the Debreu-Fan-Glicksberg theorem [1952], and recover a SPRPE as a limit point of the sequence of Nash equilibria of this game. The proof that we present in this section follows very much along the same lines of the proof that we presented in Section 4.7.1 and is thus conceptually much simpler than that of Li and Tang [162]. In addition, since it does not exhibit the discontinuous arguments of the proof of Li and Tang [162], it can also be used to show FIXP-membership of the problem, as we highlight in Section 4.7.2.

Since $\alpha_i^* \geq 1$ for any $i \in N$, and since for every $j \in G$, there exists some buyer $i \in N$ with $v_{ij} > 0$, we can again merge Conditions III.a. and III.c. above and obtain the following equivalent definition for a SPRPE.

Definition 4.7.4 (Second-price ROI Pacing Equilibrium (SPRPE)-simplified)**.** A pair $(\alpha, \mathbf{x})$, with $\alpha \in [1, A]^n$ and $\mathbf{x} \in [0,1]^{nm}$, is a *second-price ROI pacing equilibrium (SPRPE)* of an SPPRG $\mathcal{G}$, if the following conditions hold:

IV.a. $\sum_{i \in N} x_{ij} = 1$, for all $j \in G$, ***(feasible allocation)***

IV.b. $x_{ij} > 0 \Rightarrow \alpha_i v_{ij} = h_j(\alpha)$, for all $i \in N$ and $j \in G$, ***(only highest bids win)***

IV.c. $\sum_{j \in G} x_{ij} p_j(\alpha) \leq \sum_{j \in G} x_{ij} v_{ij}$, for all $i \in N$, ***(ROI constraints)***

IV.d. $\sum_{j \in G} x_{ij} p_j(\alpha) < \sum_{j \in G} x_{ij} v_{ij} \Rightarrow \alpha_i = A$, for all $i \in N$. ***(maximum pacing)***

A new proof of existence

We state the main theorem of this section.

Theorem 4.7.2. *A SPRPE exists for any SPPRG $\mathcal{G}$.*

Proof. We define the function $F : D \to D$ with domain $D = \prod_{i \in N}[1, A] \times \prod_{i \in N} \prod_{j \in G}[0,1]$. Let $(\alpha, \mathbf{x})$ and $(\alpha^*, \mathbf{x}^*)$ denote inputs and outputs to F respectively. For any item $j \in G$, the vector of allocations $\mathbf{x}_j^* = (x_{1j}^*, \ldots, x_{nj}^*)$ is obtained as the output of the following linear program.

<u>Linear Program $\mathcal{P}$</u>

$$\text{maximize} \sum_{i \in N} (\alpha_i v_{ij}) x_{ij}$$

$$\text{subject to} \sum_{i \in N} x_{ij} = 1, \ \forall i \in N$$

$$x_{ij} \geq 0, \ \forall i \in N$$

The pacing multipliers will be obtained via the following equation

$$\alpha_i^* = \max\left\{1, \min\left\{A, \alpha_i + \sum_{j\in G} x_{ij}v_{ij} - \sum_{j\in G} x_{ij}p_j(\alpha_i)\right\}\right\} \tag{4.42}$$

We will show that a fixed point $(\alpha^*, \mathbf{x}^*)$ of F is a RPE. In particular, we show that $(\alpha, \mathbf{x}) = (\alpha^*, \mathbf{x}^*)$ satisfies Conditions IV.a. to IV.d. of Definition 4.7.4.

1. Let $j \in G$ be any item. Since $\mathbf{x}_j$ is an optimal solution to linear program $\mathcal{P}$, and since $\alpha_i v_{ij}$ is positive for all $i \in N$, the allocation constraint of $\mathcal{P}$ must be tight, i.e., $\sum_{i\in N} x_{ij}^* = 1$. Hence Condition IV.a. is satisfied.

2. Consider any buyer $i \in N$ and item $j \in N$ such that $x_{ij}^* > 0$. By the optimality of $\mathbf{x}_j^*$, $\alpha_i v_{ij}$ must be maximum, i.e., $\alpha_i v_{ij} = \alpha_i^* \cdot v_{ij} = h_j(\alpha^*)$. Hence Condition IV.b. is satisfied.

3. Let $i \in N$ be any buyer. Assume by contradiction that Condition IV.c. is violated, i.e., that $\sum_{j\in G} x_{ij}^* v_{ij} < \sum_{i\in N} x_{ij}^* p_j(\alpha^*)$. Since $\alpha_i = \alpha_i^*$, Equation (4.42) implies that $\alpha_i^* = 1$. In turn, this implies that for any item $j \in G$, we have $v_{ij} = \alpha_i^* v_{ij}$. Consider any such an item $j' \in G$ for which $x_{ij'}^* > 0$. By Condition IV.b. which was established in Item 2 above, we have that $\alpha_{ij'}^* v_{ij'} = h_j'(\alpha^*)$. Combining the above two expressions, and the fact that $h_j'(\alpha^*) \geq p_j'(\alpha^*)$, we obtain that $v_{ij'} \geq p_j'(\alpha^*)$. By summing over all items $j \in G$ with $x_{ij} > 0$, we obtain a contradiction.

4. Consider some buyer $i \in N$ such that $\sum_{j\in G} x_{ij}^* p_j(\alpha^*) < \sum_{j\in G} x_{ij}^* v_{ij}$. Since $\alpha_i = \alpha_i^*$, Equation (4.42) in that case implies that $\alpha_i^* = A$, and hence Condition III.d. is satisfied.

This completes the proof. □

Membership in the class FIXP

The main technical difference between our proof in Section 4.7.1 and that in Section 4.7.2 is that in the latter case, we did not perform a variable change. Indeed, the nature of the ROI constraints in Condition IV.c. prevents us from doing that, as the variables x_{ij} appear in both sides of the constraints, once multiplied by the constants v_{ij} and once with the gate inputs $p_j(\alpha)$. The problem with having expressions of the form $\sum_{j\in G} x_{ij} p_j(\alpha)$ is that we are not allowed to multiply two parameters which are both inputs to a linear arithmetic circuit. We can do that however with (general) arithmetic circuits, see Definition 4.2.2.

As we discussed in Section 4.2, computation of fixed points via (general) arithmetic circuits corresponds to the class FIXP [86], which allows for computation of solutions that are described by irrational numbers. Here we can make use of the OPT-gate for FIXP that was designed by Filos-Ratsikas et al. [101], in order to show FIXP-membership of the problem. We have the following theorem.

Theorem 4.7.3. *Computing a SPRPE of any SPPRG $\mathcal{G}$ is in FIXP.*

Proof. To turn the existence proof of Theorem 4.7.2 to a FIXP-membership proof, we have to

- argue that the function F that we defined in Section 4.7.2 can be computed by an arithmetic circuit,
- argue that the OPT-gate for FIXP of [101] can solve the linear program $\mathcal{P}$ of Section 4.7.2.

The first part follows by observing that all the parameters of the linear program $\mathcal{P}$ and those of Equation (4.41) can be computed using the standard operations of the circuit. Note that compared to the corresponding argument in Section 4.7.1, here we indeed need the capability to multiply input variables, in order to compute α_i^* in Equation (4.41). Linear program $\mathcal{P}$ is very simple and can be easily seen to satisfy all the properties required for the OPT-gate for FIXP to work, stated in [101]. □

Limits of rationality: second-price ROI pacing equilibria can be irrational

To round off the section, it remains to determine whether our FIXP-membership result for SPRPE, rather than a PPAD-membership result, is due to our proof technique falling short, or whether there is some inherent reason for this. As we show below, there are simple examples of SPRPGs $\mathcal{G}$ for which all SPRPE are irrational and hence a PPAD-membership result is not possible.

Example 7 (Example of a SPRPG $\mathcal{G}$ that only has irrational SPRPE)**.** Consider a SPRPG $\mathcal{G}$ with 2 buyers and 3 items, with $\mathbf{v}_1 = (1,0,1)$ and $\mathbf{v}_2 = (0,1,2)$. Let A be sufficiently large, e.g., $A > 4$; the proof can easily be adapted for other values of A by appropriate rescaling of the parameters. Via a case analysis, we show that in any SPRPE, the pacing multipliers α_i for $i \in \{1,2\}$ are irrational.

Let $(\alpha, \mathbf{x})$ be a SPRPE. Clearly,

- buyer 1 will be allocated the entire quantity of item 1, i.e., $x_{11} = 1$, at price $p_1(\alpha) = 0$,
- buyer 2 will be allocated the entire quantity of item 2, i.e., $x_{22} = 1$, at price $p_2(\alpha) = 0$.

Case 1: $\alpha_1 < 2\alpha_2$**.** In this case, buyer 2 wins the entirety of item 3 at price α_1, thus we have $p_3(\alpha) = \alpha_1$, $x_{13} = 0$ and $x_{23} = 1$. By the ROI constraint (Condition III.d. of Definition 4.7.3) for buyer 2 it holds that

$$\alpha_1 = p_3(\alpha) \cdot x_{23} \leq 1 + 2 \cdot x_{23} = 3$$

However, we have that

$$x_{11} \cdot p_1(\alpha) = 0 < 1 = v_{11} \cdot x_{11} \tag{4.43}$$

By the maximum pacing constraint (Condition III.e. of Definition 4.7.3) for buyer 1, it follows that $\alpha_1 = A$. Hence $A = \alpha_1 \le 3$, which is not possible by the choice of A. This implies that $(\alpha, \mathbf{x})$ is not an SPRPE, a contradiction.

Case 2: $\alpha_1 > 2\alpha_2$. In this case, buyer 1 wins the entirety of item 3 at price $2\alpha_2$, thus we have $p_3(\alpha) = 2\alpha_2$, $x_{13} = 1$ and $x_{23} = 0$. By the ROI constraint (Condition III.d. of Definition 4.7.3) for buyer 1 it holds that

$$2\alpha_2 \cdot x_{13} = p_3(\alpha) \cdot x_{13} \le 1 + 1 \cdot x_{13} = 2 \tag{4.44}$$

and hence $\alpha_2 \le 1$, which implies that $\alpha_2 = 1$, since $\alpha_2 \in [1, A]$. Therefore, we have that $\alpha_2 \cdot v_{21} = 1$, whereas $\sum_{j \in \{1,2,3\}} x_{2j} \cdot p_j(\alpha) = 0$. By the maximum pacing constraint (Condition III.e. of Definition 4.7.3) for buyer 2, it follows that $\alpha_2 = A \le 1$, which is not possible since $A > 1$. This implies that $(\alpha, \mathbf{x})$ is not an SPRPE, a contradiction.

Case 3: $\alpha_1 = 2\alpha_2$. In this case, both buyers 1 and 2 submit the same bid, and hence they are both eligible to receive positive quantities of item 3. Since the first- and second-highest bids coincide, the price of the item is $p_3(\alpha) = \alpha_1 = 2\alpha_2 \ge 2$, since $\alpha_i \in [1, A]$ for $i \in \{1, 2\}$. By the ROI constraint (Condition III.d. of Definition 4.7.3) for both buyers, we have

$$p_3(\alpha) \cdot x_{13} \le 1 + x_{13} \tag{4.45}$$

$$p_3(\alpha) \cdot x_{23} \le 1 + 2x_{23} \tag{4.46}$$

Next, we consider whether these inequalities can be strict or not. First, if the second inequality is strict, then by the maximum pacing constraint (Condition III.e. of Definition 4.7.3) for buyer 2, we have that $\alpha_2 = A$. However, then $\alpha_1 = 2\alpha_2 = 2A$, which contradicts the fact that $\alpha_1 \in [1, A]$. This implies that the second inequality is in fact an equality, as otherwise $(\alpha, \mathbf{x})$ is not an SPRPE.

Now suppose that the first inequality is strict. By the maximum pacing constraint (Condition III.e. of Definition 4.7.3) for buyer 1, we have that $\alpha_1 = A$ and hence $p_3(\alpha) = A$. Now adding the two inequalities together we have

$$A = \alpha_1 = p_3(\alpha) \cdot (x_{13} + x_{23}) < 2 + x_{13} + 2x_{23} = 3 + x_{23} \le 4 \tag{4.47}$$

which is a contradiction since $A > 4$. This implies that the first inequality is also in fact an equality, as otherwise $(\alpha, \mathbf{x})$ is not an SPRPE.

We are now left with two equations, namely that

$$x_{12} = \frac{1}{p_3(\alpha) - 1} \text{ and } x_{23} = \frac{1}{p_3(\alpha) - 2}$$

Adding these together, we have that

$$1 = x_{13} + x_{23} = \frac{1}{p_3(\alpha) - 1} + \frac{1}{p_3(\alpha) - 2}$$

Multiplying through by the denominators, we obtain the quadratic equation $p_3(\alpha)^2 - 5p_3(\alpha) + 5 = 0$ which has only irrational solutions $(5 \pm \sqrt{5})/2$. Since $p_3(\alpha) \geq 2$, this means that $p_3(\alpha) = (5 + \sqrt{5})/2$ is the unique solution corresponding to a SPRPE. We conclude that $\alpha_1 = p_3(\alpha)$ and $\alpha_2 = p_3(\alpha)/2$ are irrational.

4.8 Fair Division

In this section we consider applications of our linear-OPT-gate to fair division. In particular, we will mainly consider two rather fundamental settings, namely *envy-free cake cutting* and *rental harmony*. The former is one of the prototypical problems in fair division [108], the origins of which date back to the late 1940s and the work of Steinhaus [208], and concerns the fair division of a single infinitely divisible resource among a set of agents with heterogeneous preferences over parts of the resource. The latter was studied famously by Su [212], who credits its origins to the chore division problem of Gardner [109], and is concerned with the fair allocation or rooms to tenants, taking into account their preferences over rooms and the rent for each room.

The PPAD-membership for *approximate* envy-free cake cutting has been known since the work of Deng et al. [76], and has been hinted at by the existence proof of Simmons [212] that in fact goes via an approximate version and applies Sperner's Lemma [204]. Filos-Ratsikas et al. [101] recently showed that finding an exact envy-free division is in FIXP in general. Still, there are cases for which exact solutions are rational (e.g., when the preferences are captured by piecewise-constant densities), and for those FIXP is not the appropriate class. Our main theorem in Section 4.8.1 below extends the ideas of Filos-Ratsikas et al. to show the desired PPAD-membership result for these cases. We remark that a FIXP-hardness result is also known from [101], as well as a PPAD-hardness result from [76] for approximate divisions, but those only concern very general versions of the problem and leave much room for improvement.

For rental harmony, there were no known complexity results before our work, to the best of our knowledge. It turns out that our linear-OPT-gate allows us to obtain the PPAD-membership of finding fair partitions using very much the same approach as for the case of envy-free cake cutting, again for these cases for which partitions in rational numbers exist. For more general settings (for which all partitions might be irrational), we complement our results with a FIXP-membership proof, based on the OPT-gate for FIXP of [101].

We remark that very recently Caragiannis et al. [44] already used the linear-OPT-gate (which we made them aware of via personal communications) to establish that computing envy-free and Pareto-optimal allocations of multiple divisible goods is in PPAD.

4.8.1 Envy-free cake cutting

We start from arguably the most fundamental fair division problem, that of *envy-free cake cutting*.

Definition 4.8.1 ((Contiguous) envy-free cake cutting)**.** Let the interval $[0,1]$ be called a *cake*, which is to be divided into n subintervals (pieces) using $n-1$ cuts. Let $\mathbf{x} \in \Delta^{n-1}$ denote a division of the cake, i.e., each division is a point in the simplex, and let x_j be the j'th coordinate (or "the j'th piece"). There is a set N of n agents, and for each piece j, each agent $i \in N$ has a *valuation function* $u_{ij} : \Delta^{n-1} \to \mathbb{R}_{\geq 0}$, assigning a real number to a division of the cake. Given a division $\mathbf{x}$, we will say that agent $i \in N$ *prefers* the j-th piece if $u_{ij}(\mathbf{x}) \geq u_{ij'}(\mathbf{x})$ for any piece j'. A division $\mathbf{x}$ is envy-free if there exists a permutation π of $\{1,\dots,n\}$ such that for every $i \in N$, agent i prefers piece $\pi(i)$.

We offer the following remarks regarding Definition 4.8.1:

- The definition considers the *contiguous* version of the problem, where the pieces are single intervals. In the more general version of the problem, each piece can be a collection of possibly disconnected intervals. Since a contiguous division is clearly a division, the PPAD-membership of computing contiguous divisions is a stronger result.

- The definition above is very general, in the sense that the valuations of the agents for the pieces do not only depend on the pieces themselves, but they could depend on the whole division $\mathbf{x}$, e.g., on how the remaining pieces have been allocated. This captures for example valuations that exhibit externalities. In contrast, several textbooks on the problem (e.g., see [38, 188, 190]) often consider the problem where the valuation of an agent only depends on her own piece, and in fact these valuations are captured by additive measures over the cake (i.e., the value of an agent for the union of two intervals is the sum of her value for each interval). Again, with regard to PPAD-membership, the more general setting that we consider here makes the result stronger.

Sufficiency condition: Hungriness. We will say that an agent $i \in N$ is *hungry*, if she prefers a non-empty piece of cake to an empty piece. An instance of the envy-free cake cutting problem satisfies the hungriness condition if all of the agents are hungry.

Known results for existence and complexity. The existence of envy-free cake cutting divisions (in the sense of Definition 4.8.1) was proven independently by Woodall [229] and Simmons (credited in [212]), under the hungriness condition. Both of these proofs go via proving existence for an approximate version of the problem (via Sperner's Lemma [204] or the related K-K-M Lemma [156]) and then take limits to obtain the existence of exact divisions. An alternative proof by Woodall [229] uses Brouwer's fixed point theorem [39]. The first continuous proof of existence was given by Filos-Ratsikas et al. [101] in the context of proving that computing an envy-free division is in the class FIXP, via the employment of their OPT-gate for FIXP:

Theorem 4.8.1 ([101])**.** *Computing an envy-free division of the cake is in FIXP.*

In general, FIXP is indeed the right class for the computation of exact envy-free divisions, as there are simple examples showing that even when the valuations are given by linear density functions, all envy-free divisions might be irrational (e.g., see [24]). We provide a simple one below for completeness:

Example 8 (Example where all envy-free cake cutting divisions are irrational)**.** Consider an instance with two identical agents and let u be their common valuation function. For u, we will specify the densities over the cake $[0,1]$, which are both given by $f(t) = 2t$. Let z be the point where the cake is cut at an envy-free division. It must hold that $u([0,z]) = u([z,1])$, i.e.,

$$\int_0^z 2t\,dt = \int_z^1 2t\,dt$$

The first integral evaluates to z^2 and the second integral evaluates to $1 - z^2$. It follows that the only envy-free division is obtained by cutting the cake at $z = 1/\sqrt{2}$, an irrational number.

Still, there are interesting valuation functions for which rational envy-free divisions exist, and for those cases a FIXP-membership result is unsatisfactory. For example, it is known that when agents have piecewise-constant density functions (i.e., step functions), an envy-free division in rational numbers exists (e.g., see [126]). For these cases for which rationality is possible, we would like to obtain a PPAD-membership result instead. This is achievable via the "approximation and rounding" approach which was discussed in Section 4.1.2, i.e., to start from an ε-approximately envy-free division of the cake, known to be computable in PPAD from [76], and then round it to an exact solution, as long as ε is small enough [126]. We provide an alternative, conceptually much simpler proof, via the employment of our linear-OPT-gate, which does not require approximations or rounding. We will obtain the PPAD-membership of the problem for any valuation function that is given by a linear pseudo-circuit. We state the main theorem of the section.

Theorem 4.8.2. *Computing an envy-free division of the cake when the agents' valuation functions are given by* linear *pseudo-circuits is in PPAD.*

The valuation functions are the integrals of the density functions, so Theorem 4.8.2 immediately implies a PPAD-membership result for envy-free divisions with piecewise-constant densities, which we mentioned above. Using the machinery developed in Section 4.3.4, we can also capture simple interesting cases where the inputs are given as the integrals of the density functions rather than the utilities themselves. We develop the proof in Section 4.8.1 below.

The proof of Theorem 4.8.2

The proof follows very much along the same lines of that of Theorem 4.8.1 presented in [101], with a crucial modification to make it amenable to the use of linear arithmetic circuits. For that reason, it is instructive to first present the proof of [101], and then explain how to obtain the proof of Theorem 4.8.2 from there.

Envy-free divisions as matchings in bipartite graphs. Let $\mathbf{x} \in \Delta^{n-1}$ be a division of the cake. Let $G = (N, P, E)$ be the bipartite graph in which N is the set of agents, P is the set of pieces and an edge (i, j), $i \in N$, $j \in P$ is in E if and only if agent i prefers piece j. Given this interpretation, $\mathbf{x}$ is envy-free if and only if G admits a perfect matching. We will find that perfect matching via a maximum flow argument, therefore we define capacities c_e on the edges of the corresponding flow network.

Envy-free divisions as fixed points. Filos-Ratsikas et al. [101] construct a function $F : D \rightarrow D$ and recover envy-free divisions from its fixed points. In particular the domain of F will be $D = \Delta^{n-1} \times \left(\Delta^{n-1}\right)^n \times ([0,1]^n)^n$. In D:

- a point $\mathbf{x} \in \Delta^{n-1}$ will be the division of the cake,
- a point $\mathbf{c}_i \in \Delta^{n-1}$ represents the capacities of the edges from agent $i \in N$,
- a point $\mathbf{y}_i$ in $[0,1]^n$ represents the flow along the edges from agent $i \in N$.

In other words, the input of F will be a vector $(\mathbf{x}, \mathbf{c}_1, \ldots, \mathbf{c}_n, \mathbf{y}_1, \ldots, \mathbf{y}_n)$ and let $(\mathbf{x}^*, \mathbf{c}_1^*, \ldots, \mathbf{c}_n^*, \mathbf{y}_1^*, \ldots, \mathbf{y}_n^*)$ denote the output. Let $r_k = \max\{0, 1 - \sum_{i=1}^n y_{ik}\}$ denote the *flow excess* incoming into piece k. The division $\mathbf{x}^*$ will be computed via the following equations:

$$x_j^* = \frac{x_j + r_j}{1 + \sum_{k=1}^n r_k}, \text{ for all pieces } j \tag{4.48}$$

The capacities $\mathbf{c}_i^*$ and the flows $\mathbf{y}_i^*$ will be obtained as the outcomes of the linear programs of Figure 4.12. $\mathcal{P}_1$ is a set of linear programs (one for each agent $i \in N$), the optimal solutions of which will give the capacities c_{ij}^*, and $\mathcal{P}_2$ is a linear program that gives the values y_{ij}^* of the flow.

Remark 37. The $1/n^3$ term in the constraints of $\mathcal{P}_2$ ensures that there exists a *strictly feasible* point (i.e., a point $\hat{\mathbf{z}}$ such that the general constraint $\mathbf{C} \cdot \mathbf{z} \leq \mathbf{d}$ is satisfied with strict inequality, i.e., $\mathbf{C} \cdot \hat{\mathbf{z}} < \mathbf{d}$, and hence satisfies the Slater condition [202]). This is required for the OPT-gate for FIXP of Filos-Ratsikas et al. [101] but is not required for our linear-OPT-gate. In particular, we could have the constraints be $0 \leq y_{ij} \leq c_{ij}$ instead, and in fact that would even slightly simplify the proofs. However, we elected to keep linear program $\mathcal{P}_2$ with the term $1/n^3$ in the constraints, for two reasons. First, it allows us to directly compare with the proof of [101]. Secondly, it allows us to refer to linear program $\mathcal{P}_2$ when we prove the FIXP-membership for the rental harmony problem in Section 4.8.2.

Linear Program $\mathcal{P}_1$

$$\begin{aligned}
\text{maximize} \quad & \sum_{j=1}^{n} c_{ij} \cdot u_{ij}(\mathbf{x}) \\
\text{subject to} \quad & \sum_{j=1}^{n} c_{ij} = 1 \\
& c_{ij} \geq 0, \quad \text{for any piece } j
\end{aligned}$$

Linear Program $\mathcal{P}_2$

$$\begin{aligned}
\text{maximize} \quad & \sum_{1 \leq i,j \leq n} y_{ij} \\
\text{subject to} \quad & 0 \leq y_{ij} \leq c_{ij} + \frac{1}{n^3}, \text{ for any agent } i \text{ and piece } j \\
& \sum_{i=1}^{n} y_{ij} \leq 1 \text{ for any piece } j \\
& \sum_{i=1}^{n} y_{ij} \geq 1 \quad \text{for all agents } i \in N
\end{aligned}$$

Figure 4.12: The linear program $\mathcal{P}_1$ used for capacities $\mathbf{c}_i$ (left) and the linear program $\mathcal{P}_2$ used for the flow $\mathbf{y}$.

Filos-Ratsikas et al. [101] state and prove the following simple lemma.

Lemma 32 ([101])**.** *If the total flow $\sum_{i \in N} \sum_j y^*_{ij} = n$, then $\mathbf{x}^*$ is an envy-free division.*

Proof Sketch. We sketch the proof here, and refer the reader to [101] for the (only slightly longer and more detailed) complete proof. By Hall's Theorem [134], it follows that unless $\mathbf{x}^*$ is an envy-free division indeed, there will be some set of agents $A \subseteq N$ such that $|N(A)| < |A|$, where $N(A)$ is the set of pieces preferred by agents in A. Linear program $\mathcal{P}_1$ dictates that in an optimal solution, $c^*_{ij} > 0$ only for pieces that agent i prefers, i.e., $c^*_{ij} = 0$ for any $i \in A$ and $j \neq N(A)$. Using this in linear program $\mathcal{P}_2$, we get that $y^*_{ij} \in [0, 1/n^3]$, which allows the total flow out of A to be bounded by a quantity strictly smaller than $|A|$ and consequently, the total flow in the network by a quantity strictly smaller than n, obtaining a contradiction. □

Lemma 32 reduces the problem of finding an envy-free division to that of finding a flow of total value n. In turn, by the definition of r_j, this is equivalent to stating that $r_j = 0$ for all pieces j. Filos-Ratsikas et al. [101] use the definition of x^*_j to establish that in a fixed point of F, this reduces to proving the statement of Lemma 33 below. Indeed, the definition of x^*_j establishes that in a fixed point (where $x^*_j = x_j$), it must be the case that

$$x_j \cdot \sum_{k=1}^{n} r_k = r_j, \text{ for all pieces } j,$$

which implies that

$$x_j > 0 \text{ and } r_j = 0 \Rightarrow \sum_{k=1}^{n} r_k = 0,$$

as desired.

Lemma 33 ([101])**.** *There exists some piece j with $r^*_j = 0$ and $x^*_j > 0$.*

Proof. Since $\mathbf{x}$ is a valid division, there exists some ℓ such that $x^*_\ell > 0$. If $r^*_\ell = 0$ we are done, so assume that $r^*_\ell > 0$. By the definition of r^*_ℓ, that means that piece ℓ has positive residual capacity, which implies that the total flow in the network is less than n. Hence, there exists some agent $i \in N$, such that the total flow out of i is less than 1, i.e., $\sum_{k=1}^n y^*_{ik} < 1$. The next step is to establish that there exists some piece $\ell' \in N(i)$, where $N(i)$ is the set of preferred pieces for agent i, with $r^*_{\ell'} = 0$, which will conclude the proof, since $x_{\ell'} > 0$ follows from the hungriness condition. Assume by contradiction that for all $k \in N(i)$, it holds that $r^*_k > 0$. Note that $\sum_{k=1}^N y^*_{ik} < 0$ as established before, and that $\sum_{k \in N(i)} c^*_{ik} = 1$, which means that it is possible to send more flow from i to $N(i)$. Since each piece in $N(i)$ is preferred by agent i, this contradicts the optimality of $\mathbf{y}^*$ as a solution to linear program $\mathcal{P}_2$. □

We now move on to the proof of Theorem 4.8.2. The proof mimics that presented above, except for one crucial part: we can no longer use Equation (4.48) to retrieve the envy-free division as a fixed point of the function F. This is because Filos-Ratsikas et al. [101] only need to argue that F can be represented by an arithmetic circuit that uses OPT-gates for FIXP, whereas we need to argue that it is possible to represent F by a *linear* arithmetic circuit using linear-OPT-gates. In particular, we cannot divide by $1 + \sum_{k=1}^n r_k$ in a linear arithmetic circuit.

It turns out that there is an easy "fix" for this. Consider the following linear program $\mathcal{P}_0$.

Linear Program $\mathcal{P}_0$

$$\begin{aligned} \text{maximize} \quad & \sum_{k=1}^n r_k \cdot x_k \\ \text{subject to} \quad & \sum_{k=1}^n x_k = 1 \\ & x_k \geq 0 \text{ for any piece } k \end{aligned}$$

The division $\mathbf{x}^*$ will be obtained as the optimal solution to a set of linear programs of the form $\mathcal{P}_0$, one for each piece. We argue that a fixed point of the function F is an envy-free division below. The PPAD-membership of the problem then follows by restricting the functions $u_{ij}(\mathbf{x})$ to assume an appropriate form that allows F to be computable by a linear arithmetic circuit.

Proof of Theorem 4.8.2. Consider a fixed point of the function F and let $\mathbf{x}^*$ be the corresponding division, which is an optimal solution of linear program $\mathcal{P}_0$ by design. Note that in such an optimal solution, positive weight is only placed on variables x_ℓ for which r_ℓ is maximum, i.e., $x^*_\ell > 0 \Rightarrow r^*_\ell = \max_k r^*_k$. Hence, if there exists some piece j such that $x^*_j > 0$ and $r^*_j = 0$, that means that $\max_k r^*_k = 0$. Since $r^*_k \geq 0$ for all k, this is equivalent to $r^*_k = 0$ for all k. Thus, we have established exactly what Equation (4.48) does without dividing in the circuit.

r_k^* can obviously be computed by a linear arithmetic circuit. Linear programs $\mathcal{P}_0$, $\mathcal{P}_1$ and $\mathcal{P}_2$ use only linear constraints with the gate inputs appearing only on the right-hand side of the constraints. The gate inputs do not appear in the objective function of $\mathcal{P}_2$, whereas the objective function of $\mathcal{P}_0$ has a linear subgradient. For $\mathcal{P}_1$, the subgradient of the objective function is a linear function in $u_{ij}(\mathbf{x})$, and since $u_{ij}(\mathbf{x})$ is given by a linear pseudo-circuit, so is the subgradient of the objective function. This establishes that F can be represented by a linear arithmetic circuit, which concludes the proof. □

4.8.2 Rental Harmony

We define the *rental harmony* problem, notably studied by Su [212].

Definition 4.8.2 (Rental harmony)**.** In the rental harmony problem, the total rent of a house has to be divided among the n rooms of the house, in a way that makes it possible to assign the rooms to n tenants in an envy-free manner, i.e., no tenant would prefer to have someone else's room (with the respective rent) to their own. Formally, we denote the rent by the unit interval $[0,1]$. Each tenant has a valuation function $u_{ij} : \Delta^{n-1} \to \mathbb{R}_{\geq 0}$, for each room j, assigning a real number to each division of the rent. Given a division $\mathbf{x}$, we will say that tenant i *prefers* the j-th room if $u_{ij}(\mathbf{x}) \geq u_{ij'}(\mathbf{x})$ for any room j'. A division $\mathbf{x}$ is envy-free if there exists a permutation π of $\{1,\ldots,n\}$ such that for every $i \in \{1,\ldots,n\}$, tenant i prefers room $\pi(i)$.

Known results for existence. The existence of a solution to the rental harmony problem was proven by Su [212], via an interesting adaptation of the idea of Simmons for proving the existence of envy-free cake cutting solutions, one which employs a "dual Sperner labelling". Similarly to the proofs for cake-cutting, the proof also appeals to limits of approximate solutions. To the best of our knowledge, computational complexity results about the general version of the problem did not exist before our work.

One can see the similarities between Definition 4.8.2 above and Definition 4.8.1 of Section 4.8.1. In fact, those are even more clear if we consider the following problem, which we refer to as *envy-free chore division*.

Definition 4.8.3 ((Contiguous) envy-free chore division)**.** Let the interval $[0,1]$ be called a *chore*, which is to be divided into n subintervals (pieces) using $n-1$ cuts. Let $\mathbf{x} \in \Delta^{n-1}$ denote a division of the chore, i.e., each division is a point in the simplex, and let x_j be the j'th coordinate (or "the j'th piece"). There is a set N of n agents, and for each piece j, each agent $i \in N$ has a *valuation function* $u_{ij} : \Delta^{n-1} \to \mathbb{R}_{\geq 0}$, assigning a real number to a division of the cake. Given a division $\mathbf{x}$, we will say that agent $i \in N$ *prefers* the j-th piece if $u_{ij}(\mathbf{x}) \geq u_{ij'}(\mathbf{x})$ for any piece j'. A division $\mathbf{x}$ is envy-free if there exists a permutation π of $\{1,\ldots,n\}$ such that for every $i \in N$, agent i prefers piece $\pi(i)$.

Looking at Definition 4.8.3 above, one can observe two things:

(a) It defines the same problem as the one in Definition 4.8.2, by simply substituting the term "room" with "piece" (and the term "tenant" with "agent").

(b) It defines the same problem as the one in Definition 4.8.1 in Section 4.8.1, by simply substituting the term "chore" with "cake".

Observation (a) is because we are using a very general definition of envy-free chore-division (as we used a very general definition of envy-free cake-cutting in Section 4.8.1), one in which there is a different valuation function for each piece. In that generality, there is no difference between a piece and a room. For restricted valuation functions, the two problems could be different. We remark again that for proving membership results, considering these very general versions makes the results stronger.

Now looking at Observation (b), it seems as if the envy-free chore division problem and the envy-free cake cutting problem are the same. Is that really the case? The difference lies in the *sufficiency condition*. Recall that for cake-cutting, the sufficiency condition was hungriness, i.e., that agents always prefer a non-empty piece to any empty piece. For chore division/rental harmony, we will have the exact opposite.

Sufficiency condition: Miserly agents. We will say that an agent i is a *miser*, if she prefers an empty piece of the chore (respectively a room with zero rent) to a non-empty piece (respectively a room with non-zero rent). An instance of the chore division/rental harmony problem satisfies the miserly agents condition if all of the agents are misers.

Given the above, in our setting envy-free chore division (Definition 4.8.3) and rental harmony (Definition 4.8.2) are equivalent, so we will use the terminology of envy-free chore division in the proof that we will develop, mainly for consistency with Section 4.8.1. Our main theorem of the section is the following, which is reminiscent of Theorem 4.8.2 for envy-free cake cutting.

Theorem 4.8.3. *Computing an envy-free division of the chore (i.e., a rental harmony partition) when the agents' valuation functions are given by* linear *arithmetic circuits in PPAD.*

As in the case of Section 4.8.1, using the machinery developed in Section 4.3.4, we can also capture simple interesting cases where the inputs are given as the integrals of the density functions rather than the utilities themselves.

Before we proceed with the proof, we make the following remark.

Remark 38. One may be inclined to believe that there would be an easy reduction from envy-free chore division to envy-free cake cutting, e.g., by setting $u_{ij}^{\text{cake}}(\mathbf{x}) = -u_{ij}^{\text{chore}}(\mathbf{x})$ for all agents i and pieces j. Indeed, this would turn a hungry agent into a miser, thus establishing the correct sufficiency condition. However, letting $\mathbf{x}^{\text{cake}} = \mathbf{x}^{\text{chore}}$ would not result in an envy-free chore division, as shown by the simple following example with 2 agents. Let $x_1^{\text{cake}} = [0, 1/2]$ and $x_2^{\text{cake}} = (1/2, 1]$ and let $u_{11}^{\text{cake}}(x_1^{\text{cake}}, x_2^{\text{cake}}) = 1 - \epsilon$ and $u_{12}^{\text{cake}}(x_1^{\text{cake}}, x_2^{\text{cake}}) = \epsilon$ for some $\epsilon < 1$. Let agent 2 be indifferent between any allocation of the cake. We have that $u_{11}^{\text{chore}}(x_1^{\text{chore}}, x_2^{\text{chore}}) = -u_{11}^{\text{cake}}(x_1^{\text{cake}}, x_2^{\text{cake}}) = -1 + \epsilon$ and $u_{12}^{\text{chore}}(x_1^{\text{chore}}, x_2^{\text{chore}}) = -u_{12}^{\text{cake}}(x_1^{\text{cake}}, x_2^{\text{cake}}) = -\epsilon$. Agent 1 obviously prefers piece 2 rather than piece 1, and this is not an envy-free chore division.

In this example it is easy to remedy that, by labelling the pieces oppositely, i.e., by having $x_1^{\text{cake}} = x_2^{\text{chore}}$ and vice-versa. What happens however in more complex scenarios, with more agents and different valuations? Finding the appropriate division would essentially end up constructing the dual Sperner labelling used by Su [212]. Our results in this section provide an alternative way of finding an envy-free chore division, which is very much in line with the approach that we used in Section 4.8.1. In other words, our technique based on the linear-OPT-gate provides a *unified proof of existence* of solutions for both envy-free cake cutting and rental harmony.

4.8.3 The proof of Theorem 4.8.3

Exactly in the same manner as in Section 4.8.1, we will consider a bipartite graph with agents i on one side and pieces j on the other side, with edges (i, j) indicating that agent i prefers pieces j. An envy-free chore division will correspond to a perfect matching in the graph, which we will again compute via a maximum flow argument in a fixed point.

Again, we construct a function $F : D \to D$ and recover envy-free chore divisions from its fixed points. In particular the domain of F will be $D = \Delta^{n-1} \times \left(\Delta^{n-1}\right)^n \times ([0, 1]^n)^n$. In D:

- a point $\mathbf{x} \in \Delta^{n-1}$ will be the division of the chore,
- a point $\mathbf{c}_i \in \Delta^{n-1}$ represents the capacities of the edges from agent $i \in N$,
- a point $\mathbf{y}_i$ in $[0, 1]^n$ represents the flow along the edges from agent $i \in N$.

In other words, the input of F will be a vector $(\mathbf{x}, \mathbf{c}_1, \ldots, \mathbf{c}_n, \mathbf{y}_1, \ldots, \mathbf{y}_n)$ and let $(\mathbf{x}^*, \mathbf{c}_1^*, \ldots, \mathbf{c}_n^*, \mathbf{y}_1^*, \ldots, \mathbf{y}_n^*)$ denote the output. As in Section 4.8.1, the capacities and the flow values will be obtained as optimal solutions to the linear programs $\mathcal{P}_1$ and $\mathcal{P}_2$ of Figure 4.12 respectively.

Again, we will use $r_k = \max\{0, 1 - \sum_{i=1}^n y_{ik}\}$ to denote the flow excess incoming into piece k. Lemma 32 holds verbatim for our setting here as well, which establishes that it suffices to argue that in our fixed point, $r_j^* = 0$ for all pieces j. Now, the chore division $\mathbf{x}^*$ will be an optimal solution to the following linear program, which is a straightforward adaptation of linear program $\mathcal{P}_0$ presented in Section 4.8.1.

Linear Program $\mathcal{P}_0^c$

$$\begin{aligned}\text{minimize}\quad & \sum_{k=1}^{n} r_k \cdot x_k\\ \text{subject to}\quad & \sum_{k=1}^{n} x_k = 1\\ & x_k \geq 0 \text{ for any piece } k\end{aligned}$$

The following lemma is analogous to Lemma 33, although the proof is somewhat different.

Lemma 34. *In a fixed point of F, $r_j^* = 0$ for all pieces j.*

Proof. Consider a fixed point $(\mathbf{x}^*, \mathbf{c}_1^*, \ldots, \mathbf{c}_n^*, \mathbf{y}_1^*, \ldots, \mathbf{y}_n^*)$ of F. For any piece j, $x_j^* > 0 \Rightarrow r_j^* = \min_k r_k^*$, which follows from the fact that x_j^* is an optimal solution to linear program $\mathcal{P}_0^c$, and hence only places positive weight to pieces with minimum r_k^*.

We will first argue that $x_j^* > 0$ for all pieces j. Assume first that $x_{j_0}^* = 0$ for some piece j_0 and consider any piece j such that $x_j^* > 0$; note that at least one such j must exist by the fact that $\mathbf{x}^*$ is a valid division of the chore. By the miserly agents sufficiency condition, we have that any agent $i \in N$ prefers j_0 to j, i.e., $u_{ij_0}(\mathbf{x}^*) > u_{ij}(\mathbf{x}^*)$. Since c_{ij}^* is an optimal solution to linear program $\mathcal{P}_1$, it follows that $c_{ij}^* = 0$. In turn, by the corresponding constraint of linear program $\mathcal{P}_2$, it follows that $y_{ij}^* \leq 1/n^3$, and that $\sum_{i=1}^n y_{ij}^* \leq 1/n^2$. From this, we conclude that $r_j^* \geq 1 - 1/n^2$.

Since $x_j^* > 0$, by the discussion above this implies that $\min_k r_k^* \geq 1 - 1/n^2$, which of course implies that $r_\ell^* \geq 1 - 1/n^2$ for all pieces ℓ. It follows that $\sum_{i=1}^n y_{i\ell}^* \leq 1/n^2$ for all pieces ℓ, and by summing over ℓ we obtain that $\sum_{\ell=1}^n \sum_{i=1}^n y_{ij}^* < 1/n$. Since $\mathbf{c}_i$ is a feasible solution to linear program $\mathcal{P}_1$, there exists a set of pieces A_ℓ with $\sum_{i \in A_\ell} c_{ij}^* = 1$ for at least one agent $i \in N$ (and those are precisely the pieces with $x_\ell^* = 0$). That implies that there is a solution to linear program $\mathcal{P}_2$ for which $\sum_{\ell=1}^n \sum_{i=1}^n y_{ij} \geq 1$. This contradicts the fact that $\mathbf{y}^*$ is an optimal solution to linear program $\mathcal{P}_2$.

Now since $x_j^* > 0$ for all pieces j, by the discussion in the first paragraph of the proof it follows that $r_j^* = r_{j'}^*$ for all pieces j and j'. Let r^* denote the value of r_j for any j. We will argue that $r^* = 0$, which will conclude the proof. Assume by contradiction that $r^* > 0$, or, equivalently, that $\sum_{i=1}^n y_{ij}^* < 1$ for piece j. This implies that in linear program $\mathcal{P}_2$ the corresponding constraint for piece j is not be tight. In turn, that means that there is a feasible flow $\mathbf{y}$ of value larger than that of $\mathbf{y}^*$, contradicting the fact that $\mathbf{y}^*$ is an optimal solution to linear program $\mathcal{P}_2$.

□

We can now conclude the proof of Theorem 4.8.3.

Proof of Theorem 4.8.3. Lemma 34 establishes that a fixed point of F corresponds to an envy-free chore division. It remains to show that F can be computed by a linear arithmetic circuit containing linear-OPT-gates. This is virtually identical to the corresponding argument in the proof of Theorem 4.8.2. □

FIXP-membership for (unrestricted) rental harmony

In the envy-free cake cutting application, Theorem 4.8.2 provides a crisper complexity result than that of Theorem 4.8.1, for the case when the valuation functions are given by linear pseudo-circuits, which admits envy-free divisions in rational numbers. For rental harmony/chore division, the situation is similar, except, as we mentioned earlier, a FIXP-membership result (or any other kind of complexity result for that matter) was not known before our work. It is almost straightforward to generalize our proof developed in Section 4.8.3 to obtain a theorem analogous to Theorem 4.8.1, for the case of fairly dividing a chore as well, when the valuation functions are generally represented by arithmetic circuits (not necessarily linear).

Theorem 4.8.4. *Computing an envy-free division of the chore is in FIXP.*

Proof. Construct a function F exactly as in Section 4.8.3 and consider a fixed point $(\mathbf{x}^*, \mathbf{c}_1^*, \dots, \mathbf{c}_n^*, \mathbf{y}_1^*, \dots, \mathbf{y}_n^*)$ of F. The arguments on why $\mathbf{x}^*$ constitutes an envy-free division are identical to those developed in the previous section. What we need to establish is that F can be computed by an arithmetic circuit that contains instances of the OPT-gate for FIXP developed by Filos-Ratsikas et al. [101]. In reality, this has practically already been established in [101]: linear programs $\mathcal{P}_1$ and $\mathcal{P}_2$ are identical to those used in [101], and linear program $\mathcal{P}_0$ is also amenable to the use of their OPT-gate for FIXP. We note that it is possible to multiply two inputs in arithmetic circuits, and hence the valuations $u_{ij}(\mathbf{x})$ in the objective function of linear program $\mathcal{P}_1$ are not restricted as in Theorem 4.8.3. □

We conclude the section with an example showing that if one goes beyond linear valuations, there are cases where all envy-free divisions are irrational, and hence the FIXP-membership result of Theorem 4.8.4 (rather than a PPAD-membership result) is justified.

Example 9 (Example where all envy-free chore divisions are irrational)**.** Consider an instance of the rental harmony/envy-free chore division problem with two agents. A division of the chore is represented by a point $(x, 1-x) \in \Delta^1$. The common utility functions of the two agents are given by $u_1(x, 1-x) = 2(1-x)^2$ and $u_2(x, 1-x) = x^2$. Note that if $x = 0$ the agents will prefer piece 1, and if $x = 1$ the agents will prefer piece 2. Therefore, the agents are indeed misers. For $(x, 1-x)$ to be an envy-free chore division, then it must hold that $2(1-x)^2 = x^2$. As $x \in [0,1]$, this implies that $x = 2 - \sqrt{2}$, so the only solution is irrational.

4.9 Conclusion and Future Work

In this work, we developed the linear-OPT-gate, a powerful general-purpose tool for showing the PPAD-membership of problems that have *exact rational solutions.* We demonstrated its strength by applying it to a plethora of domains related to game theory, competitive markets, auto-bidding auctions and fair division. For those applications, we obtain new results and generalizations of the state-of-the-art complexity results, as well as *significant* simplifications in terms of the proof techniques.

There are some interesting open directions related to our work, mainly in the domain of competitive markets. First, it will be very interesting to see whether one could extend our machinery in Section 4.3.4 to also capture markets with SSPLC production sets; we discussed the challenges of this task in Remark 35. Similarly, it would be interesting to try to design a class of succinct utility and production functions that subsumes all the known classes for which rational solutions are known to exist, i.e., one that would generalize the Leontief-free class of functions. Finally, one application that we did not study in our work is that of *competitive markets for mixed manna*, where there are goods but also bads to be allocated to the consumers. Chaudhury et al. [48] studied these markets for SPLC utility functions and provided a PPAD-membership result. There does not seem to be any technical obstacle to applying our technique on those markets as well (and also possibly incorporating production functions as well); the details are still to be worked out.

Looking at the big picture, our linear-OPT-gate complements and refines the OPT-gate for FIXP of Filos-Ratsikas et al. [101] as a tool to proving computational membership of *exact* problems in the appropriate complexity classes. One interesting question is whether one could hope to develop a similar gate for *approximate* problems, i.e., an optimization gate that could be used in a very similar manner to those other two gates to establish PPAD-membership of more general problems (with irrational solutions), for their approximate versions. This certainly introduces new challenges and intriguing questions. One would have to work with approximate rather than exact fixed points. How should the gate be constructed to be useful in this regard? Should the gate work approximately as well? Applications domains like competitive markets, where the approximation in the competitive equilibrium notion comes from relaxing the clearing condition rather than the bundle optimality of the consumers, seem to suggest otherwise.

Chapter 5

Strong Approximate Consensus Halving and the Borsuk-Ulam Theorem

Abstract

In the consensus halving problem we are given n agents with valuations over the interval $[0,1]$. The goal is to divide the interval into at most $n+1$ pieces (by placing at most n cuts), which may be combined to give a partition of $[0,1]$ into two sets valued equally by all agents. The existence of a solution may be established by the Borsuk-Ulam theorem. We consider the task of computing an approximation of an exact solution of the consensus halving problem, where the valuations are given by distribution functions computed by algebraic circuits. Here approximation refers to computing a point that ε-close to an exact solution, also called *strong* approximation. We show that this task is polynomial time equivalent to computing an approximation to an exact solution of the Borsuk-Ulam search problem defined by a continuous function that is computed by an algebraic circuit.

The Borsuk-Ulam search problem is the defining problem of the complexity class BU. We introduce a new complexity class BBU to also capture an alternative formulation of the Borsuk-Ulam theorem from a computational point of view. We investigate their relationship and prove several structural results for these classes as well as for the complexity class FIXP.

5.1 Introduction

Many computational problems, e.g. linear and semidefinite programming, are most naturally expressed using real numbers. When the model of computation is discrete, these problems must be recast as discrete problems. In the case of linear programming this causes no problems. Namely, when the input is given as rational numbers and an optimal solution exists, a rational valued optimal solution exists and may be computed in polynomial time. For semidefinite programming however, it may be the case that all optimal solutions are irrational. For dealing with such cases we may instead consider the *weak optimization* problem as defined by Grötschel, Lovász and Schrijver [132]: Given $\varepsilon > 0$, the task is to compute a rational-valued vector x that is ε-close to the set of feasible solutions and has objective value ε-close to optimal. Assuming we are also given, as an additional input, a strictly feasible solution and a bound on the magnitude of the coordinates of an optimal solution, the weak optimization problem may be solved in polynomial time using the ellipsoid algorithm [132]. Let us note however that without additional assumptions, even the complexity of the basic *existence problem* of semidefinite feasibility is unknown. In fact, the problem is likely to be computationally very hard [215]. More precisely, it is hard for the problem PosSLP, which is the fundamental problem of deciding whether an integer given by a division free arithmetic circuit is positive [5].

In this paper we consider real valued search problems, where existence of a solution is guaranteed by topological existence theorems such as the Brouwer fixed point theorem and the Borsuk-Ulam theorem. This means that the search problems are *total*, thereby fundamentally differentiating them from general search problems where, as described above, even the existence problem may be computational hard. We are mainly interested in the *approximation* problem: given $\varepsilon > 0$, the task is to compute a rational-valued vector x that is ε-close to the set of solutions.

Recall that the Brouwer fixed point theorem states every continuous function $f : B^n \to B^n$, where B^n is the unit n-ball, has a fixed point, i.e. there is $x \in B^n$ such that $f(x) = x$ [39]. The Borsuk-Ulam theorem states that every continuous function $f : S^n \to \mathbb{R}^n$, where S^n is the unit n-sphere in $\mathbb{R}^{n+1}$, maps a pair of antipodal points of S^n to the same point in $\mathbb{R}^n$, i.e. there is $x \in S^n$ such that $f(x) = f(-x)$ [35]. The Brouwer fixed point theorem is of course not restricted to apply to the domain B^n, but applies to any domain that is homeomorphic to B^n. Similarly the Borsuk-Ulam theorem applies to any domain homeomorphic to S^n by an antipode-preserving homeomorphism. It is well-known that the Borsuk-Ulam theorem generalizes the Brouwer fixed point theorem, in the sense that the Brouwer fixed point theorem is easy to prove using the Borsuk-Ulam theorem [211, 222].

The Brouwer fixed point theorem and the Borsuk-Ulam theorem naturally define corresponding real valued search problems, and thereby also corresponding approximation problems. In addition, the statements of the theorems naturally leads to another notion of approximation. For the case of the Brouwer fixed point theorem we may look for an *almost* fixed point, i.e. $x \in B^n$ such that $f(x)$ is ε-close to x, and for the case of the Borsuk-Ulam theorem we look for a pair of antipodal points that *almost* map to the same point, i.e. $x \in S^n$ such that $f(x)$ and $f(-x)$ are ε-close. Following [85], we shall refer to this notion of approximation as *weak* approximation and to make the distinction clear we refer to the former (and general) notion of approximation as *strong* approximation. In the setting of weak approximation in relation to the Borsuk-Ulam theorem we assume that f has co-domain B^n.

In their seminal work, Etessami and Yannakakis [85] introduced the complexity class FIXP to capture the computational complexity of the real-valued search problems associated with the Brouwer fixed point theorem, and proved that the problem of finding a Nash equilibrium in a given 3-player game in strategic form is FIXP-complete. In order to have a notion of completeness, the class FIXP is defined to be closed under reductions. The type of reductions chosen by Etessami and Yannakakis, SL-reductions, consists of mapping between sets of solutions by a composition of a *projection* reduction followed by individual affine transformation applied to each coordinate.

Etessami and Yannakakis consider different ways to cast real valued search problems as discrete search problems. In addition to the approximation problem, these are the *partial computation* problem where the task is to compute a solution to a given number of bits of precision and *decision* problems, where the task is to evaluate a sign condition of the set of solutions given the promise that either all solutions satisfy the condition or none of them do. Of these we shall only consider the approximation problem. The class FIXP$_a$ denotes the class of discrete search problems corresponding to strong approximation of Brouwer fixed points and is defined to be closed under polynomial time reductions. Etessami and Yannakakis also prove that the problem PosSLP reduce to the problem of approximating a Nash equilibrium, thereby showing that FIXP$_a$ likely contains search problems that are computationally very hard.

While the notion of SL-reductions is very restricted, it is sufficient for proving completeness of the problem of finding Nash equilibrium. Likewise, SL-reductions are sufficient for showing that FIXP is robust with respect to the choice of domain for the Brouwer function.

Another important reason for using SL-reductions is that they immediately imply polynomial time reductions between the corresponding decision and approximation problems (the partial computation problem is more fragile and requires additional assumptions, cf. [85]). As we are mainly interested in the approximation problem more expressive notions of reducibility can be considered, while maintaining the property that reducibility implies polynomial time reducibility between the corresponding approximation problems. A sufficient condition for this is that the mapping of solutions is *polynomially continuous* and polynomial time computable.

5.1.1 The Borsuk-Ulam Theorem

Deligkas, Fearnley, Melissourgos, and Spirakis [72] recently introduced a complexity class BU to capture, in an analogy to FIXP, the computational complexity of the real-valued search problems associated with the Borsuk-Ulam theorem.

The Borsuk-Ulam theorem has a number of equivalent statements that are also easy to derive from each other. A function f defined on the unit sphere S^n is *odd* if $f(x) = -f(-x)$ for all $x \in S^n$. Note that the boundary ∂B^n of the unit n-ball B^n is identical to S^{n-1}. We thus say that a function f defined on B^n is odd on ∂B^n if f is odd when restricted to S^{n-1}. We present the simple proof of the known fact that the different formulations can be derived from each other, for the purpose of discussing equivalence from a computational point of view.

Theorem 5.1.1 (Borsuk-Ulam)**.** *The following statements hold:*

1. *If $f\colon S^n \to \mathbb{R}^n$ is continuous there exists $x \in S^n$ such that $f(x) = f(-x)$.*
2. *If $g\colon S^n \to \mathbb{R}^n$ is continuous and odd there exists $x \in S^n$ such that $g(x) = 0$.*
3. *If $h\colon B^n \to \mathbb{R}^n$ is continuous and odd on ∂B^n there exists $x \in B^n$ such that $h(x) = 0$.*

Proof of equivalence. Given f we may define $g(x) = f(x) - f(-x)$. Clearly g is odd and we have $g(x) = 0$ if and only if $f(x) = f(-x)$, which shows that (2) implies (1). Conversely, given g we simply let $f = g$. If $f(x) = f(-x)$, then since g is odd we have $f(x) = g(x) = -g(-x) = -f(-x) = -f(x)$ and hence $g(x) = f(x) = 0$, which therefore shows (1) implies (2).

We may view S^n as two hemispheres, each homeomorphic to B^n, which are glued together along their equators. Let $\pi\colon S^n \to B^n$ be the orthogonal projection defined by $\pi(x_1,\dots,x_{n+1}) = (x_1,\dots,x_n)$. Then given h we may define

$$g(x) = \begin{cases} h(\pi(x)) & \text{if } x_{n+1} \geq 0 \\ -h(-\pi(x)) & \text{if } x_{n+1} \leq 0 \end{cases} .$$

The assumption that h is odd on ∂B^n makes g a well-defined continuous odd function. We have $g(x) = 0$ if and only if $h(x) = 0$, which shows that (2) implies (3). Conversely, given g we define h by $h(x) = g\left(x, (1 - \|x\|_2^2)^{\frac{1}{2}}\right)$. Then h is continuous and odd on ∂B^n, since $x \in \partial B^n$ if and only if $\|x\|_2^2 = 1$. Clearly if $h(x) = 0$ we may let $y = (x, (1 - \|x\|_2^2)^{\frac{1}{2}})$ and have $g(y) = 0$. On the other hand, when $g(y) = 0$ we may define $x = (y_1,\dots,y_n)$ if $y_{n+1} \geq 0$ and $x = (-y_1,\dots,-y_n)$ if $y_{n+1} < 0$, and we have $h(x) = 0$. Together this shows that (3) implies (2). □

The class BU defined in [72] corresponds to first formulation of the above theorem. We may clearly consider the second formulation equivalent to the first also from a computational point of view. In particular, when translating between formulations, the set of solutions is unchanged. Note that this set of solutions has the property that all solutions come in pairs: when x is a solution then $-x$ is a solution as well. For the third formulation of the theorem this property only holds for solutions on the boundary ∂B^n.

In contrast, while the mapping of solutions of the third formulation to the second (and first) formulation given above is continuous this is not the case in the other direction. More precisely, consider $y \in S^n$ such that $g(y) = 0$. For a solution strictly contained in the upper hemisphere, the orthogonal projection to the first n coordinates produces $x \in B^n$ such that $h(x) = 0$. For a solution y strictly contained in the lower hemisphere, the projection is instead applied to the antipodal solution $-y$.

To clarify this issue from a computational point of view we introduce a new class BBU of real valued search problems corresponding to the third formulation of Theorem 5.1.1, and it will follow from definitions that BU $\subseteq$ BBU. In the context of strong approximation however, the corresponding classes of discrete search problems

BU_a and BBU_a will be shown to coincide. The idea is that given an approximation to $y \in S^n$, where $g(y) = 0$, that is sufficiently close to the equator of S^n, there is no harm in *incorrectly* deciding to which hemisphere y belongs, since solutions $x \in \partial B^n$ for which $h(x) = 0$ also come in pairs.

For the class BU, the notion of SL-reductions is clearly too restrictive to allow a reasonable comparison to FIXP. Closing the class BU by SL-reductions, the solutions would still come in pairs, thereby imposing strong conditions on the set of solutions. On the other hand the reductions should also not be *too* strong. In particular it would be desirable that FIXP would be still be closed under the chosen notion of reductions. This issue is not discussed in [72]. We shall therefore propose a suitable notion of reductions for both BU and BBU.

5.1.2 Consensus Halving

The Consensus halving problem is a classical problem of *fair division* [201]. We are given a set of n bounded and continuous measures $\mu_1, \dots, \mu_n$ defined on the interval $A = [0,1]$. The goal is to partition the interval A into at most $n+1$ intervals, i.e. by placing at most n *cuts*, such that unions of these intervals form another partition $A = A^+ \cup A^-$ of A satisfying $\mu_i(A^+) = \mu_i(A^-)$ for every i. We may think of the intervals being assigned a *label* from the set $\{+,-\}$, and A^+ is precisely the union of the intervals labeled by $+$. Such a partition is also known as a consensus halving. Using the Borsuk-Ulam theorem, Simmons and Su [201] proved that a consensus halving using at most n cuts always exists. Simmons and Su represent a division of A as a point x on the unit n-sphere S^n_1 with respect to the ℓ_1-norm. The point x is viewed as representing a division into *precisely* $n+1$ intervals, where some intervals are possibly empty. More precisely, the i-th interval has length $|x_i|$, and intervals of length 0 may simply be discarded. The intervals of positive length are then labeled according to $\mathrm{sgn}(x_i)$. Note that for any x, the antipode $-x$ represent the division where the sets A^+ and A^- are exchanged. This naturally leads to a formulation using the Borsuk-Ulam theorem [201]. Namely we may consider the function $F \colon S^n_1 \to \mathbb{R}^n$ given by $F(x)_i = \mu_i(A^+)$, and note that any $x \in S^n_1$ for which $F(x) = F(-x)$ represent a consensus halving.

We are interesting in the simple setting of additive measures, where we have corresponding density functions $f_1, \dots, f_n$ such that $\mu_i(B) = \int_B f_i(x)\,dx$. To cast the consensus halving problem as a real valued search problem we follow [72] and assume that the measures $\mu_1, \dots, \mu_n$ are given by the distribution functions $F_1, \dots, F_n$ defined by $\int_0^x f_i(x)\,dx$. An instance of the consensus halving problem is then given as a list of algebraic circuits computing these distribution functions.

Corresponding to the different formulations of the Borsuk-Ulam theorem as a real valued search problem with domain S^n or B^n we get two different formulations of the consensus halving problem. We denote these by CH and BCH respectively. Deligkas et al. proved membership of CH in BU following the proof of Simmons and Su, and proved hardness of CH for FIXP. Combining these, it follows that $\mathrm{FIXP} \subseteq \mathrm{BU}$.

5.1.3 Strong versus Weak Approximation

The difference between weak and strong approximation was studied in detail in the general context of the Brouwer fixed point theorem by Etessami and Yannakakis. A central example is the problem of finding a Nash equilibrium (NE). An important notion of approximation of a NE is the notion of an ε-NE. Computing an ε-NE of a given strategic form game Γ is polynomial time equivalent to computing a weak ε'-approximation to a fixed point the Nash's Brouwer function F_Γ associated to Γ [85, Proposition 2.3]. In turn, computing a weak ε'-approximation to a fixed point of F_Γ polynomial time reduces to computing a strong ε''-approximation to a fixed point of F_Γ [85, Proposition 2.2], since the function F_Γ is polynomially continuous and polynomial time computable. In general however an ε-NE might be far from any actual NE, unless ε is inverse doubly exponentially small as a function of the size of the game [85, Corollary 3.8].

For the problem of consensus halving we can illustrate the difference between weak and strong approximation by a simple example. We shall refer to a weak ε-approximation of a consensus halving as simply an ε-consensus halving. Consider a single agent whose measure μ is on the interval $[0,1]$ is given by the following density

$$f(x) = \begin{cases} (1+\varepsilon)/\varepsilon & \text{if } 0 \le x < \varepsilon/2 \\ 0 & \text{if } \varepsilon/2 \le x < 1-\varepsilon/2 \\ (1-\varepsilon)/\varepsilon & \text{if } 1-\varepsilon/2 \le x \le 1 \end{cases}$$

We have $\mu([0,1]) = 1$ and since μ is a step function, the corresponding distribution function F is piecewise linear. The unique consensus halving is obtained by placing a cut at the point $\varepsilon/2 - \varepsilon^2/(2+2\varepsilon)$. Placing a cut at any point $t \in [\varepsilon/2 - \varepsilon^2/(1+\varepsilon), 1-\varepsilon/2]$ results in an ε-consensus halving, i.e. such that $|\mu([0,t]) - \mu([t,1])| \le \varepsilon$. Thus an ε-consensus halving might be very far from an actual consensus halving. Note also that placing a cut at any point $t \in [0, 3\varepsilon/2 - \varepsilon^2/(2+2\varepsilon)]$ is a strong ε-approximation, which illustrates that a strong approximation is not necessarily a weak approximation. On the other hand, a strong $(\varepsilon^2/2)$-approximation is also an ε-consensus halving.

The Brouwer fixed point theorem and the Borsuk-Ulam theorem can both be proved starting from combinatorial analogoues of the two theorems, namely from Sperner's lemma and Tucker's lemma, respectively. The proofs of these two lemmas are constructive, but using them to derive the Brouwer fixed point theorem and the Borsuk-Ulam theorem involve a nonconstructive limit argument. Let us in passing note that while Sperner's lemma, like the Borsuk-Ulam theorem, has several different formulations, it is usually formulated as the combinatorial analogue of the third formulation of Theorem 5.1.1.

Sperner's and Tucker's lemma give rise to total NP search problems. These turn out to be complete for the complexity classes PPAD and PPA introduced in seminal work by Papadimitriou [182]. Papadimitriou proved PPAD-completeness of the problem given by Sperner's lemma as well as membership in PPA of the problem given by Tucker's lemma, while PPA-completeness of the latter problem was proved recently by Aisenberg, Bonet, and Buss [4]. These results also imply that the classes PPAD and PPA corresponds to the problems of computing weak approximations to Brouwer fixed points and to Borsuk-Ulam points.

The computational complexity of the problems of computing an ε-NE and of computing an ε-consensus halving was settled in breakthroughs of two lines of research. Computing an ε-NE was shown to be PPAD-complete by Daskalakis, Goldberg and Papadimitriou [67] and Cheng and Deng [51]. Computing an ε-consensus halving was shown to be PPA-complete by Filos-Ratsikas and Goldberg [94, 95].

5.1.4 Our Results

Our main result is that the problem of strong approximation of consensus halving is equivalent to strong approximation of the Borsuk-Ulam theorem.

Theorem 5.1.2. *The strong approximation problem for* CH *is* BU_a*-complete.*

As described we view the consensus halving problem as the real valued search problem with its domain being either the unit sphere or the unit ball with respect to the ℓ_1-norm. The theorem is proved by reduction from the real valued search problem associated with the Borsuk-Ulam theorem on the domain being the unit ball with respect to the ℓ_∞-norm, i.e. from a defining problem of the class BBU.

It is of general interest to study the relationship between search problems given by the Borsuk-Ulam theorem on different domains from a computational point of view. The reduction establishing the proof of Theorem 5.1.2 gives additional motivation for this. The domains we consider are unit spheres S^n_p and unit balls B^n_p with respect to the ℓ_p-norm for $p \geq 1$ or $p = \infty$. It is of course straightforward to construct homeomorphisms between unit spheres or unit balls with respect to different norms, and these could be used to define reductions between the different problems. We would however like that the mapping of solutions is simple, and in particular we would like to avoid divisions and root operations. We prove that one may in fact reduce between domains using SL-reductions.

Deligkas et al. gave a reduction from the FIXP-complete problem of finding a Nash equilibrium to CH. Combined with membership of CH in BU, this gives the inclusion FIXP $\subseteq$ BU. We observe that a proof due to Volovikov [222] of the Brouwer fixed point theorem from the Borsuk-Ulam theorem may be adapted to give a simple proof of the inclusion FIXP $\subseteq$ BU.

For the class FIXP we prove two interesting structural properties that do not appear to have been observed earlier. While FIXP is defined using SL-reductions, we show that FIXP is closed under polynomial time reductions where the mapping of solutions is expressed by *general* algebraic circuits. This in particular supports that one may reasonably define the classes BU and BBU using less restrictive notions of reductions than SL-reductions. We propose to have the mapping of solutions be computed by algebraic circuits involving the operations of addition, multiplication by scalars, as well as maximization. This means that the mapping of solutions is a piecewise linear function, and we refer to these as PL-reductions. The second structural result for

FIXP is a characterization of the class by very simple Brouwer functions. These are defined on the unit-hypercube domain $[0,1]^n$ and each coordinate function is simply one of the operations $\{+,-,*,\max,\min\}$, modified to be have the output truncated to the interval $[0,1]$.

For the classes BU and BBU we prove that they are also closed under reductions where the mapping of solutions is computed by general algebraic circuits, but with the additional requirement that this function must be odd.

For the class FIXP, an interesting consequence of the proof that finding a Nash equilibrium is complete, is that the class may be characterized by Brouwer functions computed by algebraic circuits without the division operation. The proof also shows that the class FIXP is unchanged even when allowing root operations as basic operations. We prove by a simple transformation that the classes BU and BBU may be characterized using algebraic circuits without the division operation. Furthermore, as a consequence of Theorem 5.1.2 the class of strong approximation problems $\mathrm{BU}_a = \mathrm{BBU}_a$ is unchanged even when allowing root operations as basic operations.

5.1.5 Comparison to previous work

As a precursor to the proof of PPA-completeness of computing an ε-consensus halving, Filos-Ratsikas, Frederiksen, Goldberg and Zhang [98] proved the problem to be PPAD-hard. Deligkas et al. [72] uses ideas from this proof together with additional new ideas to obtain their proof of FIXP-hardness for computing an exact consensus halving.

While PPAD $\subseteq$ PPA, the PPAD-hardness result of [98] is not implied by the recent proofs of PPA-completeness. In particular, the work [98] proves PPAD-hardness even for *constant* ε, while the work of [95] only proves PPA-hardness for ε being inverse polynomially small. In the same way, while FIXP $\subseteq$ BU, FIXP-hardness of computing an exact consensus halving is not implied by our reduction, since Theorem 5.1.2 establishes BU_a-hardness rather than BU-hardness. Recently a considerably simpler proof of PPA-hardness for computing an ε-consensus halving was given by Filos-Ratsikas, Hollender, Sotiraki and Zampetakis [99], and our reduction is inspired by this work.

All reductions described above are similar in the sense that one or more evaluations of a circuit are expressed in the consensus halving instance. The full interval A is partitioned into subintervals, cuts within these subintervals encode values in various ways, and agents implement the gates of the circuit by placing cuts. A main difference between the reductions establishing PPAD-hardness and FIXP-hardness to those establishing PPA-hardness is that in the former reductions, all cuts are constrained to be placed in distinct subintervals. This reason this is possible is that the objective is to find a fixed point of the circuit, which means that inputs and outputs may be identified.

In the setting of PPA and BBU the objective is to find a "zero" of the circuit. More precisely, for the setting of PPA the objective is to find two adjacent points of a given Tucker labeling that receive complementary labels, i.e. labels of different sign but same absolute value. For the setting of BBU the objective is to find an actual zero point of the circuit. All of the reductions establishing PPA-hardness of computing an ε-consensus halving have the property that cuts encoding the input of the circuit are *free* cuts, meaning that they can in principle be placed anywhere, and as a result also interfere with the evaluations of the circuit. This is also the case for our reduction, and this invariably limits its applicability to the approximation problem.

In the reduction of [99], the interval A is structured into different regions, a coordinate-encoding region, a constant-creation region, several circuit-simulation regions, and finally a feedback region. Our reduction also has a coordinate-encoding region and several circuit simulation regions, but the functions performed by the constant-creation region and feedback regions perform in [99] is our reduction integrated in the individual circuit simulation regions and done differently.

A novelty of the reduction of [99] compared to previous reductions is in how values are encoded by cuts in subintervals. In previous reductions, values are encoded by what we will call *position encoding*. Here it is required that there is exactly one cut in the subinterval, and the value encoded is determined by the distance between the cut position and the left endpoint of the interval. In [99] values are encoded by what we will call *label encoding*. Here there is no requirement on the number of cuts in the subinterval, and the value encoded is simply the difference between the Lebesgue measures of the subsets of the interval receiving label + and label −. We shall employ a hybrid approach where the coordinate-encoding region uses label encoding while the circuit-simulation regions uses position encoding. The first step performed in a circuit-simulation region is thus to copy the input from the coordinate-encoding region. Switching to position encoding allows us in particular to implement a multiplication gate, similarly to [72]. Here the multiplication xy is computed via the identity $xy = ((x+y)^2 - x^2 - y^2)/2$. In [72] where values range over $[0,1]$, the squaring operation may be implemented directly by agents. In our case values range over the interval $[-1,1]$, and the squaring operation is decomposed further, having agents compute it separately over the intervals $[-1,0]$ and $[0,1]$.

In analogy to [99] we have feedback agents that ensures that the circuit evaluates to 0 on the encoded input. The criteria that the agents check is however different, and for our purposes it is crucial that we have the same sign pattern in the position encoding of the output of the circuit as the copy of the input made by the circuit-simulation region. The actual detection of an output of 0 is performed by using approximations of the Dirac delta function. For computing the distribution functions of the feedback agents, we make use of the fact that these are computed by algebraic circuits, which enable us to make a strong approximation of the Dirac delta function via repeated squaring.

5.1.6 Organization of Paper

In Section 5.2 we introduce necessary terminology and we give a detailed account of real valued search problems and reducibility between these. Our structural results for FIXP are given in Section 5.3 and our structural results for BU and BBU are given in

Section 5.4. Section 5.4 also includes the simple proof of the inclusion FIXP $\subseteq$ BU. We present our main result, Theorem 5.1.2, in Section 5.6.

5.2 Preliminaries

5.2.1 Algebraic Circuits

Let B be a finite set of real valued functions, for example $B = \{+, -, *, \div, \max, \min\}$. An *algebraic circuit* C with n inputs and m outputs over the *basis* B is given by an acyclic graph $G = (V, A)$ as follows. The *size* of C is equal to the number of nodes of G, which are also referred to as *gates*. The *depth* of C is equal to the length of the longest path of G. Every node of indegree 0 is either an *input gate* labeled by a variable from the set $\{x_1, \dots, x_n\}$ or a *constant gate* labeled by a real valued constant. Every other node is labeled by an element of B called the *gate function*. If a node v is labeled by a gate function $g\colon A \to \mathbb{R}$ with $A \subset \mathbb{R}^k$ we require that g has exactly k ingoing arcs with a linear order specifying the order of arguments to g. The output of C is specified by an ordered list of m (not necessarily distinct) nodes of G. The computation of C on a given input $x \in \mathbb{R}^n$ is defined in the natural way. Computation may fail in case a gate of C labeled by a function $g\colon A \to \mathbb{R}$ receives an input outside A, and in this case the output of C is undefined. Otherwise we say that the output is well defined and denote its value by $C(x)$. If $D \subseteq \mathbb{R}^n$ we say that C computes a function $f\colon D \to \mathbb{R}^m$ if $C(x)$ is well defined for all $x \in D$.

We shall in this paper just consider algebraic circuits where the basis consists only of continuous functions. This means in particular that any algebraic circuits computes a continous function as well. We shall also only consider consider constant gates labeled with rational numbers. In this case we are also interested in the *bitsize* of the encoding of the constants, which is the maximum bitsize of the numerator or denominator. An important special class of algebraic circuits are those over the basis $\{+, -, *, \div\}$ and using just the constant 1. We refer to these as *arithmetic circuits*. An arithmetic circuit with no division gates is called division-free. Note that any integer of bitsize τ may be computed by a division-free arithmetic circuit of size $O(\tau)$.

By using multiplication with the constant -1, the functions $-$ and min may be simulated using $+$ and max, respectively. In this way we may convert a circuit over the full basis $\{+, -, *, \div, \max, \min\}$ into an equivalent $\{+, *, \div, \max\}$-circuit. We shall also consider circuits where use of the multiplication operator $*$ is *restricted* to having one of the arguments being a constant gate. We denote this by the symbol $*\zeta$ and use it in particular for defining $\{+, *\zeta, \max\}$-circuits.

At times it will convinient to consider gate functions with their output range truncated to stay within a given interval. If $g\colon A \to \mathbb{R}$ is a gate function and $a \leq b$ defines a real interval $[a, b]$ we denote by $g_{T[a,b]}$ the gate function defined by $g_{T[a,b]}(x) = a$ if $g(x) < a$, $g_{T[a,b]}(x) = b$ if $g(x) > b$, and $g_{T[a,b]}(x) = g(x)$ otherwise. Note that $g_{T[a,b]}$ is continuous whenever g is continuous.

While we shall not consider circuits with the discontinous sign function sgn, in the context of approximating functions, it is sometimes sufficient to use an approximation of sgn instead. A typical use of $\operatorname{sgn}(z)$ is to perform a selection between two values x and y. We define the δ-approximate selection function to be the function that based on $\operatorname{sgn}(z)$ outputs values x or y except in the interval of length δ centered around 0 where it instead linearly interpolates between x and y.

Definition 5.2.1. For given $\delta > 0$, the (two-sided) δ-approximate selection function Sel is defined by

$$\operatorname{Sel}_\delta(x,y,z) = \begin{cases} x & \text{if } z \le -\delta/2 \\ (y-x)z/\delta + (y+x)/2 & \text{if } -\delta/2 \le z \le \delta/2 \\ y & \text{if } \delta/2 \le z \end{cases}$$

We note that $\operatorname{Sel}_\delta$ may be computed as $\operatorname{Sel}_\delta(x,y,z) = (1-t)/2 \cdot x + (1+t)/2 \cdot y$, where t defined by $t = \max(\min(z,\delta/2), -\delta/2)/(\delta/2)$ is the δ-approximation of $\operatorname{sgn}(z)$. In particular is $\operatorname{Sel}_\delta(x,y,z)$ computed by a $\{+,*,\max\}$-circuit (or a $\{+,*,\div,\max\}$-circuit if we also view δ as a variable).

5.2.2 Search problems

A general search problem Π is defined by specifying to each input instance I a *search space* (or *domain*) D_I and a set $\operatorname{Sol}(I) \subseteq D_I$ of *solutions*. We distinguish between *discrete* and *real-valued* search problems. For discrete search problems we assume that $D_I \subseteq \{0,1\}^{d_I}$ for an integer d_I depending on I. Analogously, for real-valued search problems we assume that $D_I \subseteq \mathbb{R}^{d_I}$ for an integer depending on I. One could likewise distinguish between search problems with discrete input and real-valued input. We are however mostly interested in problems where the input is discrete, that is we assume that instances I are encoded as strings over a given finite alphabet Σ (e.g. $\Sigma = \{0,1\}$).

A very important class of discrete search problems arise from decision problems given as languages in NP, thereby forming the class of NP search problems. More precisely, these are the discrete search problems where we assume there are polynomial time algorithms that (i) given I compute d_I whose magnitude is polynomial in $|I|$, (ii) given I and $x \in \{0,1\}^{d_I}$ checks whether $x \in D_I$, and lastly, (iii) given I and $x \in D_I$ checks whether $x \in \operatorname{Sol}(I)$. The corresponding language in NP is then $L = \{I \mid \operatorname{Sol}(I) \neq \emptyset\}$. The class of all NP search problems is denoted by FNP. The subclass TFNP of FNP consists of the NP search problems for which $\operatorname{Sol}(I) \neq \emptyset$ for every input I. An NP search problem Π is said to be solvable in polynomial time if there is a Turing machine running in polynomial time that on input I gives as output *some* member y of $\operatorname{Sol}(I)$ in case $\operatorname{Sol}(I) \neq \emptyset$ and rejects otherwise. The subclass of FNP consisting of the search problems solvable in polynomial time is denoted by FP, and it holds that FP = FNP if and only if P = NP.

Many natural search problems are however defined with a continous search space. Not all of these may adequately be recast as discrete search problems, but are more naturally viewed as real-valued search problems. One approach for studying such problems would be to switch to the Blum-Shub-Smale model of computation [30]. A

BSS machine resembles a Turing machine, but operates with real numbers instead of symbols from a finite alphabet. In particular is the input real-valued, and input instances are therefore encoded as real-valued vectors. All basic arithmetic operations and comparisons are unit-cost operations. One may then define real-valued analogues of Turing machine based classes. In particular, Blum, Shub and Smale defined and studied the real-valued analogues $\mathrm{P}_{\mathbb{R}}$ and $\mathrm{NP}_{\mathbb{R}}$ of P and NP. A BSS machine may in general make use of real-valued *machine constants*. If a BSS machine only uses rational valued machine constants we shall call it *constant-free*. Real-valued analogoues of the classes FP, FNP, and TFNP for the BSS machine model do not appear to be defined in the literature, but can be defined in a straight-forward way. Let us just note that the proof that P = NP implies FP = FNP does not generalize to the setting of BSS machines, since it crucially depends on the search space being discrete.

For the classes $\mathrm{P}_{\mathbb{R}}$ and $\mathrm{NP}_{\mathbb{R}}$, if we simply restrict the input to be discrete and consider only constant-free BSS machines this results in complexity classes, denoted by $\mathrm{BP}(\mathrm{P}^0_{\mathbb{R}})$ and $\mathrm{BP}(\mathrm{NP}^0_{\mathbb{R}})$, that may directly be compared to Turing machine based complexity classes. Indeed, it was proved by Allender, Bürgisser, Kjeldgaard-Pedersen and Miltersen [5, Proposition 1.1] that $\mathrm{BP}(\mathrm{P}^0_{\mathbb{R}}) = \mathrm{P}^{\mathrm{PosSLP}}$, where PosSLP is the problem of deciding whether an integer given by a division free arithmetic circuit is positive. While the precise complexity of PosSLP is not known, Allender et al. proved that it is contained in the *counting hierarchy* CH (not to be confused with the consensus halving problem whose abbreviation coincides).

The class $\mathrm{BP}(\mathrm{NP}^0_{\mathbb{R}})$ is equal to the class $\exists\mathbb{R}$ that was defined by Schaefer and Štefankovič [197] to capture the complexity of the existential theory of the reals ETR. It is known that $\mathrm{NP} \subseteq \exists\mathbb{R} \subseteq \mathrm{PSPACE}$, where the latter inclusion follows from the decision procedure for ETR due to Canny [41]. Schaefer and Štefankovič also prove $\exists\mathbb{R}$-completeness for deciding existence of a probability-constrained Nash equilibrium in a given 3-player game in strategic form; later works have extended this to $\exists\mathbb{R}$-completeness for many other decision problems about existence of Nash equilibria satisfying different properties in 3-player games in strategic form [27–29, 120]. The proofs of $\exists\mathbb{R}$-hardness makes critical use of the fact that the input is discrete and it is not known if these problems are also complete for $\mathrm{NP}_{\mathbb{R}}$.

We define the class of $\exists\mathbb{R}$ search problems as the following subclass of all real valued search problems. Instaces I are encoded as string over a given finite alphabet Σ and we assume there is a polynomial time algorithm that given I computes d_I, where $D_I \subseteq \mathbb{R}^{d_I}$. We next assume that there are polynomial time constant free *BSS machines* that given I and $x \in \mathbb{R}^{d_I}$ checks whether $x \in D_I$, and given I and $x \in D_I$ checks whether $x \in \mathrm{Sol}(I)$. The corresponding language in $\exists\mathbb{R}$ is then $L = \{I \mid \mathrm{Sol}(I) \neq \emptyset\}$.

5.2.3 Solving real-valued search problems

Let Π be a $\exists\mathbb{R}$ search problem. In analogy with the case of NP search problems, one could consider the task of solving Π to be that of giving as output some member y of $\mathrm{Sol}(I)$ in case $\mathrm{Sol}(I) \neq \emptyset$. In general each member of $\mathrm{Sol}(I)$ may be irrational valued which precludes a Turing machine to compute a solution explicitly. This is in general also the case for a BSS machine, even when allowing machine constants. Regardless, we shall restrict our attention to Turing machines below.

On the other hand, when $\mathrm{Sol}(I) \neq \emptyset$ a solution is guaranteed to exist with coordinates being algebraic numbers, since a member of $\mathrm{Sol}(I)$ may be defined by an existential first-order formula over the reals with only rational-valued coefficients. This means that one could instead compute an indirect description of the coordinates of a solution, for instance by describing isolated roots of univariate polynomials. If such a description could be computed in polynomial time in $|I|$ we could consider that to be a polynomial time solution of Π.

Etessami and Yannakakis [85] suggest several other computational problems one may alternatively consider in place of solving a search problems Π explicitly or exactly. Our main interest is in the problem of *approximation*. We shall assume for simplicity that $D_I \subseteq [-1,1]^{d_I}$. Together with an instance I of Π we are now given as an auxiliary input a rational number $\varepsilon > 0$, and the task is to compute $x \in \mathbb{Q}^{d_I}$ such that there exist $x^* \in \mathrm{Sol}(I)$ with $\|x^* - x\|_\infty \leq \varepsilon$. We shall turn this into a *discrete* search problem by encoding the coordinates of x as binary strings. More precisely, to Π we shall associate a discrete search problem Π_a where instances are of the form (I,k), where I is an instance of Π and k is a positive integer. We define $\varepsilon = 2^{-k}$ and let the domain of (I,k) be $D_{I,k} = \{0,1\}^{d_I(k+3)}$, thereby allowing the specification of a point $x \in D_I$ with coordinates of the form $x_i = a_i 2^{-k+1}$, where $a_i \in \{-2^{k+1},\ldots,2^{k+1}\}$. The solution set $\mathrm{Sol}(I,k)$ is defined from $\mathrm{Sol}(I)$ by approximating each coordinate. That is, we define $\mathrm{Sol}(I,k) = \{x \in D_{I,k} \mid \exists x^* \in \mathrm{Sol}(I) : \|x^* - x\|_\infty \leq \varepsilon\}$. Note that if we had defined $\mathrm{Sol}(I,k)$ by instead truncating the coordinates of solutions $x^* \in \mathrm{Sol}(I)$ to k bits of precision, we would have obtained the possibly harder problem of *partial computation* which was also considered by Etessami and Yannakakis [85].

We say that Π can be approximated in polynomial time if the approximation problem Π_a can be solved in time polynomial in $|I|$ and k.

5.2.4 Reductions between search problems

Let Π and Γ be search problems. A *many-one reduction* from Π to Γ consists of a pair of functions (f,g). The function f is called the instance mapping and the function g the solution mapping. The instance mapping f maps any instance I of Π to an instance $f(I)$ of Γ and for any solution $y \in \mathrm{Sol}(f(I))$ of Γ the solution mapping g maps the pair (I,y) to a solution $x = g(I,y) \in \mathrm{Sol}(I)$ of Π. It is required that $\mathrm{Sol}(f(I)) \neq \emptyset$ whenever $\mathrm{Sol}(I) \neq \emptyset$. We will only consider many-one reductions, and will refer to these simply as *reductions*.

If Π_1 and Π_2 are discrete search problems a reduction (f,g) between Π_1 and Π_2 is a *polynomial time reduction* if both functions f and g are computable in polynomial time. If Π_1 and Π_2 are real-valued search problems it is less obvious which notion of reduction is most appropriate and we shall consider several different types of reductions. For all these we assume that f is computable in polynomial time. The reduction (f,g) is a *real polynomial time reduction* if g is computable in polynomial time by a constant free BSS machine. We shall generally consider this notion of reduction too powerful. In particular the definitioon does not guaranteed that the function g is a continuous function in its second argument y. For this reason we instead consider reductions defined by algebraic circuits over a given basis B of real-valued basis functions.

We say that the reduction (f,g) is a *polynomial time B-circuit reduction* if there is a function computable in polynomial time thats maps an instance I to a B-circuit C_I in such a way that C_I computes a function $C_I\colon D_{f(I)} \to D_I$ where $g(I,y) = C_I(y)$ for all $y \in \mathrm{Sol}(f(I))$. Note in particular that the size of C_I and the bitsize of all constant gates are bounded by a polynomial in $|I|$. If in addition there exists a constant h such that the depth of C_I is bounded by h for all I we say that the reduction (f,g) is a *polynomial time constant depth B-circuit reduction*. Etessami and Yannakakis [85] defined the even weaker notion where the function f is a *separable linear* transformation. The reduction (f,g) is an SL-reduction if there is a function $\pi\colon \{1,\dots,d_I\} \to \{1,\dots,d_{f(I)}\}$ and rational constants a_i, b_i, for $i = 1,\dots,d_I$, all computable in polynomial time from I, such that for all $y \in \mathrm{Sol}(f(I))$ it holds that $x_i = a_i y_{\pi(i)} + b_i$, where $x = g(I,y)$. Thus an SL-reduction is simply a *projection reduction* together with an individual affine transformation of each coordinate of the solution.

Functions computed by algebraic circuits over the basis $\{+, *\zeta, \max\}$ are piecewise linear. We shall thus call polynomial time $\{+, *\zeta, \max\}$-circuit reductions for polynomial time piecewise linear reductions, or simply PL-reductions.

It is easy to see that all notions of reductions defined above are transitive, i.e. if Π reduces to Γ and Γ reduces to Λ, then Π reduces to Λ as well.

A desirable property of PL-reductions is that the solution mapping g is *polynomially continuous*. By this we mean that for all rational $\varepsilon > 0$ there is a rational $\delta > 0$ such that for all points x and y of the domain, $\|x - y\|_\infty \le \delta$ implies $\|g(x) - g(y)\|_\infty \le \varepsilon$, and the bitsize of δ is bounded by a polynomial in the bitsize of ε and of $|I|$. An example of a notion of reductions not guaranteed to be polynomially continuous would be $\{+, *, \max\}$-circuit reductions, since a circuit might perform repeated squaring. However, constant depth $\{+, *, \max\}$-circuit reductions would still be polynomially continuous.

5.2.5 Total real-valued search problems

Like in the case of TFNP where interesting classes of total NP search problems may be defined in terms of existence theorems for finite structures [129, 182], we may define classes of total real valued $\exists\mathbb{R}$ search problems based on existence theorems concerning domains $D_I \subseteq \mathbb{R}^n$. Typical examples of such domains D_I are spheres and balls. Suppose p is either a real number $p \ge 1$ or $p = \infty$. By S^n_p and B^n_p we denote the unit n-sphere and unit n-ball with respect to the ℓ_p-norm defined as $S^n_p = \{x \in \mathbb{R}^{n+1} \mid \|x\|_p = 1\}$ and $B^n_p = \{x \in \mathbb{R}^n \mid \|x\|_p \le 1\}$, respectively. If p is not specified, we simply assume $p = 2$.

The Brouwer fixed point theorem and FIXP

We recall here the definition of the class FIXP by Etessami and Yannakis [85]. The class FIXP is defined by starting with $\exists\mathbb{R}$ search problems given by the Brouwer fixed point theorem, and afterwards closing the class with respect to SL-reductions. We shall refer to these defining problems as *basic* FIXP problems.

Definition 5.2.2. An $\exists\mathbb{R}$ search problem Π is a *basic* FIXP problem if every instance I describes a nonempty compact convex domain D_I and a continuous function $F_I\colon D_I \to D_I$, computed by an algebraic circuit C_I, and these descriptions must be computable in polynomial time. The solution set is $\mathrm{Sol}(I) = \{x \in D_I \mid F_I(x) = x\}$.

The Brouwer fixed point theorem guarantees that every basic FIXP problem is a total $\exists\mathbb{R}$ search problem. To define the class FIXP, Etessami and Yannakis restrict attention to a concrete class of basic FIXP problems.

Definition 5.2.3. The class FIXP consists of all *total* $\exists\mathbb{R}$ search problems that are SL-reducible to a basic FIXP problem for which each domain D_I is a convex polytope described by a set of linear inequalities with rational coefficients and the function F_I is defined by a $\{+, -, *, \div, \max, \min\}$-circuit C_I.

The class FIXP$_a$ is the class of strong approximation problems corresponding to FIXP. More precisely, FIXP$_a$ consist of all discrete search problems polynomial time reducible to the problem Π_a for $\Pi \in$ FIXP.

The definition of FIXP is quite robust with respect to the choice of domain and set of basis functions allowed by circuits in the basic FIXP problems. Etessami and Yannakis proved that basic FIXP problems defined by $\{+, -, *, \div, \max, \min, \sqrt[k]{}\}$-circuits are still in the class FIXP. Likewise, basic FIXP-problems where D_I is a ball with rational-valued center and diameter, or more generally an ellipsoid given by a rational center-point and a positive-definite matrix with rational entries, are still in the class FIXP [85, Lemma 4.1]. The same argument allows for using as domain the ball B_p^d with respect to the ℓ_p norm for any rational $p \geq 1$ or $p = \infty$, with the coordinates possibly transformed by individual affine functions.

On the other hand, Etessami and Yannakakis also proved that one may greatly restrict the class of basic FIXP problems used to define FIXP without changing the class. The domains may be restricted to be unit hypercubes $[0, 1]^{d_I}$ and the circuits may be restricted to $\{+, *, \max\}$-circuits. Both restrictions may in fact be imposed at the same time. The restriction to $\{+, *, \max\}$-circuits is a consequence of first proving that the problem of finding a Nash equilibrium in a given finite game in strategic form is hard for FIXP with respect to SL-reductions and then proving FIXP-membership of this problem using $\{+, *, \max\}$-circuits.

Another way to restrict circuits is by limiting their depth. The function of Nash for expressing Nash equilibrium as Brouwer fixed points involve divisions but as noted by Etessami and Yannakakis it may be viewed as a constant depth circuit, if one allows for addition gates of arbitrary fanin. Thus in the definition of FIXP one may restrict circuits to be constant depth $\{+,*,\max\}$-circuits, where the addition gates are allowed to have unbounded fanin.

We show in Proposition 12 of Section 5.3 that one may in fact take this much further and *completely* flatten the circuits of defining problems for FIXP to be depth 1 circuits of fanin at most 2, additionally also without requiring division. In other words, each coordinate function becomes just a simple function of at most 2 coordinates of the input. We also show in Proposition 10 that FIXP is closed under *much* more powerful reductions than just the basic SL-reductions used to define the class FIXP.

The Borsuk-Ulam theorem and BU

A new class BU of total $\exists\mathbb{R}$ search problems based on the Borsuk Ulam theorem was recently introduced by Deligkas et al. [72]. The definition of BU is meant to capture the Borsuk-Ulam theorem as stated in formulation (1) of Theorem 5.1.1. Following the definition of FIXP by Etessami and Yannakakis, Deligkas et al. first consider a set of basic search problems and then close the class under reductions.

Definition 5.2.4. An $\exists\mathbb{R}$ search problem Π is a basic BU problem if every instance I describes a domain $D_I \subseteq \mathbb{R}^{d_I}$ which is homeomorphic to S^{d_I-1} by an an antipode preserving homeomorphism and a continuous function $F_I\colon D_I \to \mathbb{R}^{d_I-1}$, computed by an algebraic circuit C_I, and these descriptions must be computable in polynomial time. The solution set is $\mathrm{Sol}(I) = \{x \in D_I \mid F_I(x) = F_I(-x)\}$.

In defining the class Deligkas et al. restrict their attention to spheres with respect to the ℓ_1-norm as domains and functions computed by $\{+,-,*,\max,\min\}$-circuits. Compared to the definition of FIXP, division gates are thus excluded. However we show later in Section 5.4 that division gates can always be eliminated. Having thus fixed the set of basic BU search problems what remains in order to define BU is to settle on a notion of reductions. In their journal paper, Deligkas et al. [72] suggest using reductions computable by general algebraic circuits including non-continuous comparison gates, whereas in the preceeding conference paper [71] they did not precisely define a choice of reductions. We shall revisit the question of choice of reduction in Section 5.4 before proposing our definition of BU.

5.2.6 Consensus Halving

We give here a formal definition of consensus halving with additive measures as real valued search problems.

Definition 5.2.5. The problem CH is defined as follows. An instance I consists of a list of $\{+,-,*,\div,\max,\min\}$-circuits $C_1,\dots,C_n$ computing distribution functions $F_1,\dots,F_n$ defined on the interval $A = [0,1]$. The domain is $D_I = S_1^n$ and $\mathrm{Sol}(I)$ constists of all x for which

$$\sum_{j:x_j>0} F_i(t_j) - F_i(t_{j-1}) = \sum_{j:x_j<0} F_i(t_j) - F(t_{j-1}) \ , \tag{5.1}$$

where $t_0 = 0$ and $t_j = \sum_{k\leq j}|x_k|$, for $j = 1,\dots,n+1$.

Given $\{+,-,*,\div,\max,\min\}$-circuits computing the distribution functions F_i, the function F computing the left-hand-side of equation (5.1) may clearly be computed by $\{+,-,*,\div,\max,\min\}$-circuits as well. The result of Deligkas et al. that CH is contained in BU follows.

The existence proof of a consensus halving by Simmons and Su as well the formulation of a $\exists\mathbb{R}$ search problem by Deligkas et al. match the Borsuk-Ulam theorem as stated in formulation (1) of Theorem 5.1.1. We shall also define a variation BCH of CH to match formulation (3) of Theorem 5.1.1. A point $y \in B^n_1$ may be lifted to the point $x = (1-\|y\|_1, y) \in S^n_1$. This means that we may view $y \in B^n_1$ as describing a partition of A by the partition described by x. Compared to the representation of partitions of A into $n+1$ intervals given by points of S^n_1 we thus restrict the label of the first interval to be $+$, in case it has positive length.

Definition 5.2.6. The problem BCH is defined as follows. An instance I consists of a list of $\{+,-,*,\max,\min\}$-circuits $C_1,\dots,C_n$ computing distribution functions $F_1,\dots,F_n$ defined on the interval $A=[0,1]$. The domain is $D_I = B^n_1$ and $\mathrm{Sol}(I)$ constists of all y for which

$$F_i(t_1) + \sum_{j:y_j>0} F_i(t_{j+1}) - F_i(t_j) = \sum_{j:y_j<0} F_i(t_{j+1}) - F(t_j) \ , \tag{5.2}$$

where $t_0 = 0$, and $t_j = 1 - \sum_{k\geq j}|y_j|$, for $j = 1,\dots,n+1$.

5.2.7 Tools from Real Algebraic Geometry

For obtaining our results concerning strong approximation we need concrete bounds on $\delta > 0$ as a function of $\varepsilon > 0$ witnessing the truth of "epsilon-delta" statements. When such a statement is expressible in the first-order theory of the reals, such bounds can be obtained in a generic way using the general machinery of real algebraic geometry [21]. This approach has been used several times previously for establishing FIXP_a membership of the problem of strong approximation of Nash equilibrium refinements [83, 88, 136].

Concretely, suppose that $\Phi(\varepsilon,\delta)$ is a formula with free variables ε and δ of the form

$$\Phi(\varepsilon,\delta) = (Q_1 x_1 \in \mathbb{R}^{n_1})\cdots(Q_\omega x_\omega \in \mathbb{R}^{k_\omega})F(x_1,\dots,x_\omega,\varepsilon,\delta) \ ,$$

where $Q_i \in \{\forall,\exists\}$, and F is a Boolean formula whose atoms are polynomial equalities and inequalities involving polynomials of degree at most d and having integer coefficients of bitsize at most τ.

We now assume that the statement $(\forall\varepsilon > 0)(\exists\delta > 0)\Phi(\varepsilon,\delta)$ is true, and fix $\varepsilon = 2^{-k}$, for a positive integer k, resulting in the formula $(\delta > 0) \wedge \Phi(\varepsilon,\delta)$, with δ as the only variable. We may now perform *quantifier elimination* [21, Algorithm 14.21] on this to obtain an equivalent quantifier free formula $\Psi(\delta)$. The formula $\Psi(\delta)$ is simply a Boolean formula whose atoms involve univariate polynomial equalities and inequalities. The bounds given by Basu, Pollack and Roy for the result of quantifier elimination imply that the degree of the univariate polynomials are bounded by

$d^{O(k_1)\dots O(k_\omega)}$ with coefficients of bitsize at most $\max(k,\tau)d^{O(k_1)\dots O(k_\omega)}$. We may now appeal to Theorem 13.17 of [21] to conclude that $\Psi(\delta)$, and hence also $\Phi(\varepsilon,\delta)$ is true, for some $\delta \geq 2^{-\max(k,\tau)d^{O(k_1)\dots O(k_\omega)}}$.

In our applications, the formula Φ is defined from a given instance I. Both τ and d will be bounded by fixed polynomials of $|I|$. The number of blocks ω of quantified variables will be a fixed constant, and k_i for $1 \leq i \leq \omega$ are bounded by fixed polynomials of $|I|$ as well. In other words there will be a fixed polynomial q such that the formula $\Phi(\varepsilon,\delta)$ is true for some $\delta \geq (1/\varepsilon)^{2^{g(|I|)}}$.

The first-order formulas we consider are expressed using also the evaluation of functions computable by algebraic circuits as a primitive. We may in a generic way transform such formulas to having only polynomial inequalities and equalities and required above. Namely, we may perform a Tseitin-style transformation by introducing existentially quantified variables for each gate of the circuit and express using polynomial inequalities and equalities that each gate is computed correctly, and the variables corresponding to the output gates may then be used instead in place of the function. As long as the number of evaluations of functions is constant, this leaves the number of blocks of quantified variables constant.

5.3 Structural Properties of FIXP

Recall that FIXP is defined to be the closure of all basic FIXP problems with respect to the very simple notion of SL-reductions. We first show that FIXP is in fact closed under *general* circuit reductions.

Proposition 10. *Suppose that* Π *is a* $\exists\mathbb{R}$ *search problem defined with unit hypercube domains and reduces to* $\Gamma \in$ FIXP *by a polynomial time* $\{+,-,*,\div,\max,\min,\sqrt[k]{\ }\}$-*circuit reduction. Then* Π *belongs to* FIXP *as well.*

Proof. We may without loss of generality assume the domain of Γ is also the unit hypercube. Let (f,g) be the assumed reduction from Π to Γ. Let I be an instance of Π. By assumption $D_I = [0,1]^m$ and $D_{f(I)} = [0,1]^n$, where $m = d_I$ and $n = d_{f(I)}$. From the definition of (f,g) we may given I in polynomial time compute $f(I)$ as well as the circuit C_I that defines a function $G\colon [0,1]^n \to [0,1]^m$ such that $g(I,x) = G(x)$ for all $x \in \mathrm{Sol}(f(I))$. By assumption on Γ we may in polynomial time compute another circuit $C_{f(I)}$ that defines a function $F\colon [0,1]^n \to [0,1]^n$ such that $\mathrm{Sol}(f(I))$ are the fixed points of F.

We now define the function $H\colon [0,1]^{n+m} \to [0,1]^{n+m}$ by $H(x,y) = (F(x),G(x))$. Clearly the set of fixed points of H is equal to $\{(x,G(x)) \mid x \in \mathrm{Sol}(f(I))\}$, and since H is computable by a $\{+,-,*,\div,\max,\min,\sqrt[k]{\ }\}$-circuit this defines a $\exists\mathbb{R}$ search problem Λ in FIXP with the same set of instances as Π. We note that the projection of a fixed point of H to the last m coordinates gives a solution to Π from which it follows that Π in particular SL-reduces to Λ. Therefore Π belongs to FIXP as well. $\square$

Our next basic result is based on properties of the basic FIXP problem used by Etessami and Yannakakis to show that the division operation is not necessary to express all of FIXP. We give a brief review of their construction. An instance I describes a d-player game in strategic form. Player i has a set S_i of $n_i = |S_i|$ pure strategies and a utility function $u_i \colon S_1 \times \cdots \times S_d \to \mathbb{R}$. Let $n = n_1 + \cdots + n_d$ be the total number of strategies. The domain is given as $D_I = \Delta_{n_1-1} \times \cdots \times \Delta_{n_d-1}$, where the $(n_i - 1)$-dimensional unit simplex Δ_{n_i-1} is identified with the set of probability distributions on S_i, for $i = 1, \dots, d$. The domain D_I may be viewed as a subset of $\mathbb{R}^n$ in the natural way. The utility functions define the function $v \colon D_I \to \mathbb{R}^n$ given by

$$v(x)_{ia_i} = \sum_{a_{-i} \in S_{-i}} u_i(a_1, \dots, a_d) \prod_{j \neq i} x_{ja_j} \ ,$$

where $S_{-i} = S_1 \times \cdots \times S_{i-1} \times S_{i+1} \times \cdots \times S_d$ and $a_{-i} = (a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_d) \in S_{-i}$. Define further the function $h \colon D_I \to \mathbb{R}^n$ by $h(x) = x + v(x)$ and finally let $G_I \colon D_I \to D_I$ be defined by letting $G_I(x)$ be the projection of $h(x)$ onto D_I. For all $i = 1, \dots, d$, it holds that $G_I(x)_{i,a_i} = \max(h_{i,a_i} - t_i, 0)$, where t_i is the unique value satisfying $\sum_{a_i \in S_i} \max(h_{i,a_i} - t_i, 0) = 1$. The fixed points of G_I are exactly the Nash equilibria of the game described by I [85, Lemma 4.5], and the search problem is therefore FIXP-complete [85, Theorem 4.3].

The definitions of the functions v, h, and G_I allows us to extend their domain from D_I to the n-dimensional unit cube $[0,1]^n$. By definition of G_I this does not change the set of fixed points of G_I. Likewise, applying the same affine transformation to $u_i(x)$, for $i = 1, \dots, d$, does not change the set of fixed points of G_I. We may thus assume that u_i has codomain $[0,1]$. Making use of a sorting network, Etessami and Yannakakis show that G_I may be computed by a polynomial size $\{+, -, *, \max, \min\}$-circuit C_I [85, Lemma 4.6]. It is furthermore straightforward to ensure that all constants used in C_I as well as values computed by gate functions of C_I belong to the interval $[0,1]$ for any input $x \in [0,1]^n$ (cf. [72]). We summarize these observations below.

Proposition 11. *There is a basic* FIXP *problem* Π_{NE}*, complete for* FIXP *under SL-reductions, such that for any instance I it holds that* $D_I = [0,1]^{d_I}$ *and such that* C_I *is a* $\{+, -, *, \max, \min\}$*-circuit that satisfies that all gate functions of* C_I *compute values in* $[0,1]$ *given input* $x \in D_I$.

From here we may derive a characterization of FIXP in terms of depth 1 circuits, where the addition and subtraction operators (necessarily) are truncated to the interval $[0,1]$. This is simply done by a Tseitin-style transformation. One may note that a Tseitin-style transformation is already used in the proof that Π_{NE} is FIXP-hard. This means such a transformation is applied twice at different points of the proof to yield the statement below.

Proposition 12. *There is a basic* FIXP *problem* Π*, complete for* FIXP *under SL-reductions, such that for any instance I it holds that* $D_I = [0,1]^{d_I}$ *and such that* C_I *is a depth 1* $\{+_{T[0,1]}, -_{T[0,1]}, *, \max, \min\}$*-circuit, using only constants from the interval* $[0,1]$.

Proof. We reduce from the problem $\Pi_{\mathrm{N}E}$ of Proposition 11. The instances of Π are the same instances of Π_{NE}. Let I be an instance of Π_{NE} and let $D = [0,1]^{d_I}$ and C_I be the corresponding domain and $\{+,-,*,\max,\min\}$-circuit as given by Proposition 11. Suppose that C_I has m_I gates $g_1,\dots,g_{m_I}$. We define the new domain D'_I for Π simply by $D'_I = [0,1]^{d'_I}$, where $d'_I = d_I + m_I$. We next define the gates of C'_I which all are output gates of C'_I. We may consider the input as pairs $(x,y) \in [0,1]^{d_I} \times [0,1]^{m_I}$ and we may think of the output gates as variables, similarly grouped as (z,w) and ranging over $[0,1]^{d_I} \times [0,1]^{m_I}$. If g_j is an input gate labeled by x_i, we let $w_j = x_i$, and if g_j is a constant gate labeled by $c \in [0,1]$ we let $w_j = c$. If g_j is an addition gate taking as input gates g_k and g_ℓ we let $w_j = (y_k + y_\ell)_{T[0,1]}$, i.e. the addition of g_k and g_ℓ is simulated by a truncated addition of y_k and y_ℓ. The case of subtraction is analogous. If g_j is a multiplication gate taking as input g_k and g_ℓ we let $w_j = y_k \cdot y_\ell$. The case of maximum and minimum gates are analogous. Finally if g_j is the ith output gate of C_I we let $z_i = y_j$. By construction C'_I computes a function $F'_I\colon D'_I \to D'_I$ and $F'_I(x,y) = (x,y)$ if and only if g_j computes the value y_j on input x for all j and $C_I(x) = x$. We thus obtain x such that $C_I(x) = x$ as the projection of (x,y) to the first d_I coordinates. □

In case we prefer to construct a normal $\{+,-,*,\max,\min\}$-circuit without truncated operations we can clearly simulate the truncated addition and subtraction operations by depth 3 circuits. We can also easily convert the circuits to constant depth $\{+,*,\max\}$ circuits by considering the the domain $B_\infty^{d'_I} = [-1,1]^{d'_I}$ instead of $[0,1]^{d'_I}$.

5.4 Definition and Structural Properties of BU and BBU

In this section we define two classes of $\exists\mathbb{R}$ search problems BU and BBU based on the Borsuk-Ulam theorem corresponding to formulations (1) and (3) of Theorem 5.1.1. We start by defining basic BU and basic BBU problems. We shall restrict our attention to the unit n-sphere and unit n-ball, but with regards to any ℓ_p norm for $p \geq 1$ or $p = \infty$. For the case of BU this amounts to specializing Definition 5.2.4.

Definition 5.4.1. A basic BU problems is a basic ℓ_p-BU problem if for every instace I we have $D_I = S_p^{d_I}$.

Similarly we define the set of basic BBU problems with respect to the ℓ_p-norm.

Definition 5.4.2. An $\exists\mathbb{R}$ search problem Π is a basic ℓ_p-BBU problem if for every instance I we have $D_I = B_p^{d_I}$ and I describes a continuous function $F_I\colon D_I \to \mathbb{R}^{d_I}$, which is odd on the boundary $\partial B_p^{d_I}$. The function F_I must be computed by an algebraic circuit C_I whose description is computable in polynomial time. The solution set is $\mathrm{Sol}(I) = \{x \in D_I \mid F_I(x) = 0\}$.

The condition that the function F_I is odd on $\partial B_p^{d_I}$ is a semantic condition. However, typically the function F_I would be defined from a basic ℓ_p-BU problem by a transformation done in a similar way as in the proof of Theorem 5.1.1, and thereby F_I would satisfy the condition automatically.

To define the classes BU and BBU, we restrict our attention to domains with respect to the ℓ_∞-norm.

Definition 5.4.3. The class BU (respectively, BBU) consists of all *total* $\exists\mathbb{R}$ search problems that are PL-reducible to a basic ℓ_∞-BU problem (respectively, basic ℓ_∞-BBU problem) for which the function F_I is defined by a $\{+,-,*,\div,\max,\min\}$-circuit C_I.

While the definition of BU in [72] was using as domain the unit sphere with respect to the ℓ_1-norm and not allowing for division gates, we show in this section these changes do not change the class. We propose choosing PL-reductions for closing the class under reductions. PL-reductions are sufficient for obtaining all of our results and they are polynomially continuous. Another reason for this choice is that if we restrict the circuits defining the classes FIXP and BU to also be piecewise linear, i.e. be $\{+,*\zeta,\max\}$-circuits, we obtain the classes LinearFIXP and LinearBU, that when closed under polynomial-time reductions are equal to PPAD and PPA, respectively [72, 85].

5.4.1 Elimination of Division Gates

In this section, we show how to eliminate division gates from circuits defining an instance of the BU or BBU problems. Let therefore C denote an algebraic circuit defined over the basis $\{+,-,*,\div,\max,\min,\sqrt[k]{\ }\}$.

Moving Divisions to the Top. In the paper [85], it is shown how to move all division gates to the top of the circuit by keeping track of the numerator and denominator of every gate. For sake of completeness we describe this transformation. Every gate g_i is replaced by two gates g'_i and g''_i keeping track of the numerator and denominator, that is the value of g_i in the original circuit will be equal to the value of g'_i/g''_i in the transformed circuit. Firstly, if g_i is an input gate or a constant-c gate we put $g'_i = x_j$ for appropriate j (respectively $g'_i = c$) and $g''_i = 1$. In order to maintain the equality $g_i = g'_i/g''_i$, we proceed as follows: if $g_i = g_j \pm g_k$ is an addition/subtraction gate in the original circuit, then we update the numerator and denominator to $g'_i = g'_j \cdot g''_k \pm g'_k \cdot g''_j$ and $g''_i = g''_j \cdot g''_k$; if $g_i = g_j \cdot g_k$, then $g'_i = g'_j \cdot g'_k$ and $g''_i = g''_j \cdot g''_k$; if $g_i = g_j \div g_k$, then $g'_i = g'_j \cdot g''_k$ and $g''_i = g''_j \cdot g'_k$. For root gates, we note that if $g_j = g'_j/g''_j$ is input to a $\sqrt[k]{\ }$–gate g_i for k even, then $g_j \geq 0$, from which it follows that $\operatorname{sgn}(g'_j) = \operatorname{sgn}(g''_j)$. With this in mind, we see that we may maintain the numerator and denominator of g_i by putting $g'_i = \sqrt[k]{g'_j g''_j}$ and $g''_i = \sqrt[k]{g''_j \cdot g''_j}$. Finally, for the max-gate we note that $\max(ca,cb) = c\max(a,b)$ for $c \geq 0$. Using this we see that if $g_i = \max(g_j,g_k)$, then we may maintain the numerator and denominator via the formulas $g'_i = \max(g'_j \cdot g''_j \cdot (g''_k)^2, g'_k \cdot g''_k \cdot (g''_j)^2)$ and $g''_i = (g''_j)^2 \cdot (g''_k)^2$. We note that all this can be done only blowing up the size of the circuit by a constant factor. In the aforementioned paper, the authors then have division gates at the top outputting $out_i = out'_i/out''_i$. However, for our application this is unnecessary and we may completely remove division gates.

Removing Division Gates for BBU. Suppose that Π is a BBU problem. Let I be an instance of Π and denote by C_I an algebraic circuit computing a continuous function

$F_I\colon B^{d_I} \to \mathbb{R}^{d_I}$ that is odd on S^{d_I-1} such that $\mathrm{Sol}(I) = \{x \in B^{d_I} \mid F_I(x) = 0\}$. As described above, we may transform the circuit C_I to a circuit C_I^+ that maintains the numerator and denominator of every gate. In the same way we define a circuit C_I^- that is exactly like C_I^+, except it multiplies the input by -1 at the very beginning. Let out_i^{n+}, out_i^{d+} and out_i^{n-}, out_i^{d-} denote the gates in $C_I^\pm$ representing the numerators and denominators of the output gates of C_I. We now define a circuit C_I^* that on input x feeds this into C_I^+ and C_I^- and then outputs the values $out_i^{n+} \cdot out_i^{d-}$ for $i = 1, \ldots, d_I$. If we denote by $F_i = F_i'/F_i''$ the coordinate functions of F_I, then C_I^* is a circuit computing the function F_I^* with coordinate functions $F_i'(x)F_i''(-x)$. Now, if $x \in S^{d_I-1}$ then $F_i'(x)/F_i''(x) = -F_i'(-x)/F_i''(-x)$, so $F_i'(x)F_i''(-x) = -F_i'(-x)F_i''(-(-x))$, meaning that F_I^* is odd on the boundary. In this way we have defined a BBU problem Γ with the same instances as Π. Furthermore, given an instance I of Π one may in polynomial time compute an instance $f(I)$ of Γ by computing C_I^*. We note that for any $x \in B^{d_I}$ it holds that $F_I(x) = 0$ if and only if $F_I^*(x) = 0$. We conclude that Π SL-reduces to the division-free BBU−problem Γ.

Removing Division Gates for BU. Now let I be an instance of a BU−problem Π and denote by C_I an algebraic circuit computing a continuous function $F_I\colon S^{d_I} \to \mathbb{R}^{d_I}$ such that $\mathrm{Sol}(I) = \{x \in S^{d_I} \mid F_I(x) = F_I(-x)\}$. We make the same reduction as for BBU defining a circuit C_I^* that computes a function $F_I^*\colon S^{d_I} \to \mathbb{R}^{d_I}$ whose coordinate functions are given by $F_i'(x)F_i''(-x)$ where $F_i'(x)/F_i''(x)$ is the ith coordinate function of F_I. By definition, x is a BU-point of F_I if and only if $F_i'(x)/F_i''(x) = F_i'(-x)/F_i''(-x)$ for all i. This happens if and only if $F_i'(x)F_i''(-x) = F_i'(-x)F_i''(-(-x))$ for all i, meaning that x is a BU-point of F_I^*. Again, we conclude that Π SL-reduces to a division-free BU−problem.

In the previous two paragraphs, we have shown the following result.

Proposition 13. *The classes* BU *and* BBU *remain the same even if the circuits are restricted to not using division gates.*

5.4.2 Relationship with FIXP

As a consequence of their results about consensus halving, Deligkas et al. proved that FIXP ⊆ BU. We observe here that the direct proof that the Bosuk-Ulam theorem implies the Brouwer fixed point theorem due to Volovikov [222] gives a much simpler way to derive this relationship. For completeness we present the construction and proof of Volovikov.

Proposition 14 (Volovikov). *Let $f\colon B_\infty^d \to B_\infty^d$ be a continuous function. Define the continous function $g\colon S_\infty^d \to \mathbb{R}^d$ by $g(x,t) = (1+t)(tf(x) - x)$. If $g(x,t) = g(-x,-t)$ then $|t| = 1$ and $f(tx) = tx$.*

Proof. Note first that

$$g(x,t) - g(-x,-t) = t\left[(1+t)f(x) + (1-t)f(-x)\right] - 2x \ .$$

It follows that $g(x,t) = g(-x,-t)$ if and only if $k(x,t) = x$, where

$$k(x,t) = \frac{t}{2}\left[(1+t)f(x) + (1-t)f(-x)\right] \ .$$

If $(x,t) \in S^d_\infty$ and $|t| < 1$ it holds that $\|x\|_\infty = 1$. Then since

$$\begin{aligned}\|k(x,t)\|_\infty \leq & \frac{|t|}{2}\left[(1+t)\|f(x)\|_\infty + (1-t)\|f(-x)\|_\infty\right] \\ \leq & \frac{|t|}{2}\left[(1+t) + (1-t)\right] = |t| < 1 \ ,\end{aligned}$$

we have $k(x,t) \neq x$. Thus $g(x,t) = g(-x,-t)$ implies that $|t| = 1$. When $|t| = 1$ we clearly have $k(x,t) = tf(tx)$. In conclusion, $g(x,t) = g(-x,-t)$ implies $tf(tx) = x$, or equivalently that $f(tx) = tx$. □

The above construction immediately give a simple reduction from any basic FIXP problem with domains $B^{d_I}_\infty$ to a basic ℓ_∞-BU problem. The solution mapping of the reduction must map solutions (x,t) to tx. This may be done by simply using multiplication gates. But since any solution (x,t) has $|t| = 1$ the multiplication tx_i may also be expressed as $\mathrm{Sel}_2(-x_i, x_i, t)$, which means the solution mapping can also be computed by constant depth $\{+,*,\max\}$-circuits.

Proposition 15. *Any* $\Pi \in \mathrm{FIXP}$ *reduces to a basic* ℓ_∞-BU *problem with* $\{+,-,*,\max,\min\}$-*circuit by polynomial time constant depth B-circuit reductions, for both* $B = \{+,-,*\}$ *and* $B = \{+,-,*\zeta,\max,\min\}$.

Proof. Any $\Pi \in \mathrm{FIXP}$ SL-reduces to a basic FIXP problem Γ with domains $D_I = B^{d_I}_\infty$ and $\{+,*,\max\}$-circuits C_I. From that, the instance mapping as described by Proposition 14 produces a $\{+,*,\max\}$-circuit and domain $S^{d_I}_\infty$. The composition of the SL-reduction and the reduction described above then yields the claimed types of reductions. □

5.4.3 Change of Domains for BU and BBU

In this section we show reduce between different domains for the BBU and BU problems.

Proposition 16. *Let B be a set of gates that contains* $\{+,-,*,\div,\max,\min\}$. *Suppose that* Π *is an* $\exists\mathbb{R}$ *search problem whose domains are contained in hypercubes that reduces to a basic* $\ell_p - \mathrm{BBU}$ *problem* Γ *by a polynomial time B-circuit reduction* (f,g). *Furthermore, suppose that for any instance I of* Π *the function* $g(I,\cdot)$ *mapping solutions of* $f(I)$ *to solutions of I is odd and assume that* $C_{f(I)}$ *is also a B−circuit. (i) If* $p = \infty$ *then* Π *SL-reduces to a basic* $\ell_\infty - \mathrm{BBU}$ *problem using gates in B. (ii) If* $1 \leq p < \infty$ *then* Π *SL-reduces to a basic* $\ell_p - \mathrm{BBU}$ *problem using* $B \cup \{\sqrt[p]{\cdot}\}$*−circuits.*

Proof. (i) First assume that the domains of Γ are unit hypercubes. Let I denote an instance of Π. By assumption $D_I \subseteq [-1,1]^m$ and $D_{f(I)} = [-1,1]^n$ where $m = d_I$ and $n = d_{f(I)}$. From the definition of (f,g) we may given I in polynomial time compute $f(I)$ and a circuit C_I computing a function $G\colon [-1,1]^n \to [-1,1]^m$ such that $G(x) = g(I,x) \in \mathrm{Sol}(I)$ for every $x \in \mathrm{Sol}(f(I))$. By assumption of Γ we may in polynomial time compute another circuit $C_{f(I)}$ that defines a function $F\colon [-1,1]^n \to \mathbb{R}^n$ that is odd on the boundary such that $\mathrm{Sol}(f(I))$ are the zeroes of F.

Define $H\colon [-1,1]^{n+m} \to \mathbb{R}^{n+m}$ by $H(x,y) = ((1-\|y\|_\infty)F(x), y - \frac{1}{2}G(x)))$. As G is odd and F is odd on the boundary, one may verify that H is odd on the boundary of $[-1,1]^{m+n}$. As H is polynomial-time computable by a B−circuit, it defines an $\ell_\infty - \mathrm{BBU}$ problem Λ with the same instances as Π. Furthermore, if (x,y) is a zero of H, then $y = G(x)/2$, so $\|y\|_\infty < 1$. The equality $(1-\|y\|_\infty)F(x) = 0$ from the first component then implies $F(x) = 0$. Therefore, the zeroes of H are contained in $\{(x, G(x)/2) \mid x \in \mathrm{Sol}(f(I))\}$. Given a zero of H one may recover a solution to Π by projecting onto the last m coordinates and multiplying by 2. In particular, Π SL-reduces to Λ.

(ii) Now, suppose that the domains of Γ are p-balls, where $1 \le p < \infty$. Again by assumption we have that $D_I \subseteq [-1,1]^m$ and $D_{f(I)} = B_p^n$ where $m = d_I$ and $n = d_{f(I)}$, and we may given an instance I of Π in polynomial time compute a circuit C_I defining a function $G\colon B_p^n \to [-1,1]^m$ such that $G(x) = g(I,x) \in \mathrm{Sol}(I)$ for every $x \in \mathrm{Sol}(f(I))$. Furthermore, we may in polynomial time compute a circuit $C_{f(I)}$ computing a function $F\colon B_p^n \to \mathbb{R}^n$ that is odd on S_p^{n-1} such that $\mathrm{Sol}(f(I))$ is the zeroes of F.

Now define an odd function $h\colon B_p^n \to B_p^n$ by $h(x) = x/\max(1/2, \|x\|_p)$, which may be computed by a circuit using also $\sqrt[p]{\cdot}$ gates, and define $H\colon B_p^{n+m} \to \mathbb{R}^{n+m}$ by

$$H(x,y) = (\max(0, \tfrac{1}{2} - \|y\|_p^p)F(h(x)), y - \tfrac{1}{3n}G(h(x)))$$

First we remark that H is odd on the boundary of B_p^{n+m}. Clearly, the second coordinate is always odd, and the first coordinate evaluates to 0 if $\|y\|_p^p > 1/2$. If $(x,y) \in S_p^{n+m-1}$ and $\|y\|_p^p < 1/2$, then $\|x\|_p^p > 1/2$ which implies that $\|x\|_p > 1/2$. This then implies that $h(x) = x/\|x\|_p$ and so $F(h(x)) = -F(-h(x)) = -F(h(-x))$, because h is odd and F is odd on S_p^{n-1}.

Now, if (x,y) is a zero of H, then $y = \frac{1}{3n}G(h(x))$ and so $\|y\|_p^p \le (n\|y\|_\infty)^p \le \frac{1}{3^p} < \frac{1}{2}$. From the first first coordinate equality $\max(0, 1/2 - \|y\|_p^p)F(h(x)) = 0$ one then obtains that $F(h(x)) = 0$ so $h(x) \in \mathrm{Sol}(f(I))$. Thus, the set of zeroes of H are contained in $\{(x, \frac{1}{3n}G(h(x)) \mid h(x) \in \mathrm{Sol}(f(I))\}$. Furthermore, H can be computed by circuit over $B \cup \{\sqrt[p]{\cdot}\}$, so this defines a basic $\ell_p - \mathrm{BBU}$ problem Λ with $(B \cup \{\sqrt[p]{\cdot}\})$-circuits and the same instances as Π. From a zero of H we may again recover a solution to Π by projecting onto the last m coordinates and multiplying the result by $3n$. We conclude that Π SL-reduces to Λ. □

Proposition 17. *Any basic* $\ell_\infty - \mathrm{BBU}$ *problem SL-reduces to a basic* $\ell_p - \mathrm{BBU}$ *problem using gates in* $\{+,-,*,\div,\max,\min,\sqrt[p]{\cdot}\}$.

Proof. Let Π be a basic $\ell_\infty - \mathrm{BBU}$ problem. By the previous proposition it suffices to argue that Π polynomial time $\{+,-,*,\div,\max,\min\}$−reduces to a a basic $\ell_p - \mathrm{BBU}$

problem. Given an instance I of Π, compute in polynomial time a circuit C_I defining a function $F\colon B_\infty^n \to \mathbb{R}^n$ that is odd on S_p^{n-1} such that $\mathrm{Sol}(I)$ are the zeroes of F. Also, define the map $\pi\colon B_\infty^n \to B_\infty^n$ by $\pi(x) = x/\max(\frac{1}{2n}, \|x\|_\infty)$. Now we may in polynomial time compute $\{+,-,*,\div,\max,\min\}$–circuit computing the function $G\colon B_p^n \to \mathbb{R}^n$ given by $G(x) = F(\pi(x))$. If $\|x\|_p = 1$, then $\|x\|_\infty \geq 1/(2n)$, so $\|\pi(x)\|_\infty = 1$, implying that $G(x) = G(-x)$. Thus we have defined a map f taking instances I of Π to instances $f(I)$ of a basic ℓ_p – BBU problem Γ. We note that f is computable in polynomial time. Furthermore, from $x \in \mathrm{Sol}(f(I))$ one may recover a solution by $g(I,x) = \pi(x)$ to I. As the function $g(I,\cdot)$ is odd and computable by a $\{+,-,*,\div,\max,\min\}$–circuit we conclude that (f,g) satisfies the requirements of the previous proposition. We conclude that Π SL-reduces to a ℓ_p – BBU using gates from $\{+,-,*,\div,\max,\min,\sqrt[k]{\cdot}\}$. □

Proposition 18. *Any basic ℓ_p –* BBU *problem Π SL-reduces to a basic ℓ_∞ –* BBU *problem where the circuits are allowed to use $\sqrt[k]{\cdot}$ gates.*

Proof. Let Π be a basic ℓ_p – BBU problem and let I denote an instance of Π. We may compute a circuit C_I defining a function $F\colon B_p^n \to \mathbb{R}^n$ that is odd on the boundary of B_p^n such that $\mathrm{Sol}(I)$ is the set of zeroes of F. Now, define a function $h\colon \mathbb{R}^n \to \mathbb{R}^n$ by $h(x) = x/\max(1/2, \|x\|_p)$. We may now in polynomial time compute a $\{+,-,*,\div,\max,\max,\sqrt[k]{\cdot}\}$-circuit computing the function $H\colon B_\infty^n \to \mathbb{R}^n$ given by $H(x) = F(h(x))$.

If $x \in \partial B_\infty^n$, then $\|x\|_p \geq \|x\|_\infty = 1$ which shows that $h(x) = x/\|x\|_p$ so $\|h(x)\|_p = 1$. As h is odd and F is odd on the boundary, it follows that $H(x) = F(h(x)) = F(-h(-x)) = -F(h(-x)) = -H(-x)$ showing that H is odd on the boundary of B_∞^n, so it defines an instance of an ℓ_∞ – BBU problem Γ. Mapping back solutions amounts to computing $h(x)$ which can be done by a $\{+,-,*,\div,\max,\max,\sqrt[k]{\cdot}\}$-circuit. The result now follows from part (i) of Proposition 16. □

Now we proceed with showing the reductions between basic ℓ_p – BU problems.

Proposition 19. *Suppose that Π is an $\exists\mathbb{R}$ search problem whose domains are contained in hypercubes that reduces to a basic ℓ_p –* BU *problem by a B–circuit reduction (f,g), where $\{+,-,*,\div,\max,\min\} \subseteq B$. Assume also that for every instance I of Π, and that $g(I,\cdot)$ is an odd map $\mathbb{R}^{d_{f(I)}} \to [-1,1]^{d_I}$. (i) If $p = \infty$ then Π SL-reduces to a basic ℓ_∞ –* BU *problem with circuits over B. (ii) If $1 \leq p < \infty$ then Π SL-reduces to a basic ℓ_p –* BU *problem with circuits over $B \cup \{\sqrt[k]{\cdot}\}$.*

Proof. (i) Let I denote an instance of Π and let $m = d_I$. By assumption of (f,g) we may in polynomial time compute a circuit defining a function $F\colon S_\infty^n \to \mathbb{R}^n$ such that $\mathrm{Sol}(f(I))$ consists of the $x \in S_\infty^n$ such that $F(x) = F(-x)$. By the result in Section 5.4.1 we may assume that the circuit computing F is division-free, and so we may extend the domain of F to be $\mathbb{R}^{n+1}$. Also, we may in polynomial time compute a circuit defining a function $G\colon \mathbb{R}^{n+1} \to [-1,1]^m$ mapping $\mathrm{Sol}(f(I))$ to $\mathrm{Sol}(I)$. Define $H\colon S_\infty^{m+n} \to \mathbb{R}^{m+n}$ by $H(x,y) = (F(x), y - \frac{1}{2}G(x))$. We note that H may be computed by a circuit over B, so it defines a basic ℓ_∞ – BU problem Λ with B–circuits and the same instances as

Π. If $(x,y) \in S_\infty^{n+m}$ has $H(x,y) = H(-x,-y)$ then $y - \frac{1}{2}G(x) = -y + \frac{1}{2}G(x)$, as G is odd. Therefore $\|y\|_\infty = \|G(x)/2\|_\infty < 1$ and so $\|x\|_\infty = 1$. Also, the first coordinate shows that $F(x) = F(-x)$. This says that $x \in \mathrm{Sol}(f(I))$, and so $G(x) \in \mathrm{Sol}(I)$. As $2y = G(x)$ we see that Π SL-reduces to Λ.

(ii) Again let I denote an instance of Π with $m = d_I$. From $f(I)$ we may in polynomial time compute a circuit computing a map $F\colon S_p^n \to \mathbb{R}^n$ such that $\mathrm{Sol}(f(I)) = \{x \in S_p^n \mid F(x) = F(-x)\}$ and a $B-$circuit computing a map $G\colon \mathbb{R}^{n+1} \to [-1,1]^m$ sending $\mathrm{Sol}(f(I))$ to $\mathrm{Sol}(I)$. Again, we may extend the domain of F. Define a map $h\colon \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ by $h(x) = x/\max(1/2, \|x\|_p)$ and $H\colon S_p^{m+n} \to \mathbb{R}^{m+n}$ by

$$H(x,y) = (F(h(x)), y - \tfrac{1}{2}G(h(x)))$$

In this way, we have defined a basic $\ell_p -$BU problem Λ with $B \cup \{\sqrt[p]{\cdot}\}-$circuits and the same instances as Π. If $(x,y) \in S_p^{n+m}$ has $H(x,y) = H(-x,-y)$ we find that $y = \frac{1}{2}G(h(x))$ so $\|y\|_p \le 1/2$. This implies that $\|x\|_p \ge 1/2$, and so $h(x) = x/\|x\|_p \in S_p^n$. Also, the first component shows that $F(h(x)) = F(h(-x)) = F(-h(x))$ where we use that h is odd. Therefore, $h(x) \in \mathrm{Sol}(f(I))$, and so $G(h(x)) \in \mathrm{Sol}(I)$. As $2y = G(h(x))$, we conclude that Π SL-reduces to Λ. □

Proposition 20. *Let $B = \{+,-,*,\div,\max,\min\}$. (i) A basic $\ell_p -$BU problem Π with B-circuits SL-reduces to a basic $\ell_\infty -$BU problem with $B \cup \{\sqrt[p]{\cdot}\}$-circuits. (ii) A basic $\ell_\infty -$BU problem Π with B-circuits SL-reduces to a basic $\ell_p -$BU problem using B-circuits.*

Proof. (i) Let Π denote a basic $\ell_p -$BU problem. Suppose an instance I is defined by some continuous function $F\colon S_p^n \to \mathbb{R}^n$. By Section 5.4.1 we may assume that the circuit computing F is division-free and so extend F to be defined in all of $\mathbb{R}^{n+1}$. Define a function $g\colon \mathbb{R}^{n+1} \to [-1,1]^{n+1}$ by $g(x) = x/\max(1/2, \|x\|_p)$ and $H\colon S_\infty^n \to \mathbb{R}^n$ by $H(x) = \overline{F}(g(x))$. Let f denote the map sending the instance I to the instance $f(I)$ given by H of a basic $\ell_\infty -$BU problem Γ. One may verify that (f,g) is a reduction satisfying the properties of Proposition 19, so by part (i) of Proposition 19 we have that Π SL-reduces to the basic $\ell_\infty -$BU.

(ii) Let Π denote a basic $\ell_p -$BU problem. Suppose an instance I is defined by some continuous function $F\colon S_\infty^n \to \mathbb{R}^n$. Again, we may extend F. Similarly to the case above, we define a function $g\colon \mathbb{R}^{n+1} \to [-1,1]^{n+1}$ by $g(x) = x/\max(1/(n+1), \|x\|_p)$ and $H\colon S_p^n \to \mathbb{R}^n$ by $H(x) = \overline{F}(g(x))$. Let f denote the map sending the instance I to the instance $f(I)$ given by H of a basic $\ell_\infty -$BU problem Γ.

First, the map g satisfies the condition of Proposition 19. If $x \mathrm{Sol}(f(I))$ then it holds that $x \in S_p^n$, and so $1 = \|x\|_p \le (n+1)\|x\|_\infty$, implying that $\|x\|_\infty \ge 1/(n+1)$. From this it follows that $g(x) = x/\|x\|_\infty$ by definition. Furthermore, using that g is odd we find that $F(g(x)) = H(x) = H(-x) = F(g(-x)) = F(-g(x))$. We conclude that $g(x) \in \mathrm{Sol}(I)$. In conclusion, (f,g) is a reduction from Π to Γ satisfying the properties of Proposition 19. By part (ii) of Proposition 19 we conclude that Π SL-reduces to a basic $\ell_\infty -$BU problem. □

5.5 Relation between $\ell_p - \mathrm{BU}$ and $\ell_p - \mathrm{BBU}$

Let B be some finite set of gates containing $\{+,-,*,\div,\max,\min\}$. In this section we study reductions between $\ell_p - \mathrm{BU}$ problems and $\ell_p - \mathrm{BBU}$ problems. Suppose we are given a basic $\ell_p - \mathrm{BU}$ problem Π with circuits defined over B. In order to show that Π reduces to a basic $\ell_p - \mathrm{BBU}$ problem we follow the proof of Theorem 5.1.1. Given an instance of Π we may in polynomial time compute the dimension $n = d_I$ and a circuit over B defining a map $F_I\colon S_p^n \to \mathbb{R}^n$ such that $\mathrm{Sol}(I) = \{x \in S_p^n \mid F_I(x) = F_I(-x)\}$. Define also the map $\pi\colon B_p^n \to S_p^n$ by

$$\pi(x) = \begin{cases} (x, (1-\|x\|_p^p)^{1/p}) & \text{if } 1 \le p < \infty \\ (x/t, 2\cdot(1-t)) & \text{if } p = \infty \end{cases}$$

where $t = \max(1/2, \|x\|_\infty)$. Define a map $H\colon B_p^n \to \mathbb{R}^n$ by $H(x) = F_I(\pi(x)) - F(-\pi(x))$. If $x \in S_p^{n-1}$ then the last coordinate of $\pi(x)$ vanishes and so $\pi(x) = -\pi(-x)$ implying that $H(x) = -H(-x)$, so H is odd on the boundary. As H is computable by a $(B \cup \{\sqrt[p]{\cdot}\})$–circuit if $p < \infty$ (and B–circuit if $p = \infty$) this defines an $\ell_p - \mathrm{BBU}$ problem Γ with the same instances as Π. Furthermore, the set of BU-points of H is exactly $\{x \in B_p^n \mid F_I(\pi(x)) = F_I(-\pi(x))\}$, so mapping solutions x of Γ to solutions of Π amounts to computing $\pi(x)$ which can be done by a circuit over $B \cup \{\sqrt[p]{\cdot}\}$ if $p < \infty$ (and over B if $p = \infty$).

However, when $p \neq 1$ these reductions make use of $\sqrt[p]{}$-gates for $p < \infty$ or divison gates for $p = \infty$. We can remedy this by applying Propositions 17, 18, and 20 which give that we may go back and forth between different domains for BBU and BU by SL-reductions. Specifically, for any $\ell_p - \mathrm{BU}$ problem we may SL-reduce to a $\ell_1 - \mathrm{BU}$ problem (that also uses $\sqrt[p]{}$ gates if $p < \infty$). Then we may apply the above $\{+,*\zeta\}$-reduction from $\ell_1 - \mathrm{BU}$ to $\ell_1 - \mathrm{BBU}$. And from there we may again SL-reduce to an $\ell_p - \mathrm{BBU}$ problem. In conclusion we obtain the following result.

Proposition 21. *Any basic $\ell_p - \mathrm{BU}$ problem $\{+,*\zeta\}$-reduces to a basic $\ell_p - \mathrm{BBU}$ problem.*

Note that the reductions of this proposition are a special case of PL-reductions. Because these are polynomially continuous, we automatically also get the following result.

Proposition 22. *Any basic $\ell_p - \mathrm{BU}_a$ problem polynomial time reduces to a basic $\ell_p - \mathrm{BBU}_a$ problem.*

For reductions in the other direction, consider an instance $H\colon B_\infty^n \to \mathbb{R}^n$ of a basic $\ell_\infty - \mathrm{BBU}$ problem. Given this instance we define an instance of a basic $\ell_\infty - \mathrm{BU}$ problem given by $F\colon S_\infty^n \to \mathbb{R}^n$ where

$$F(x) = \mathrm{Sel}_2(-H(-\pi(x)), H(\pi(x)), x_{n+1})$$

and $\pi\colon \mathbb{R}^{n+1} \to \mathbb{R}^n$ is the projection $\pi(x_1, \ldots, x_{n+1}) = (x_1, \ldots, x_n)$. Now suppose that $x \in S^n_\infty$ satisfies $F(x) = F(-x)$. If $x_{n+1} = 1$ this implies that

$$H(\pi(x)) = F(x) = F(-x) = -H(-\pi(-x)) = -H(\pi(x))$$

showing that $H(\pi(x)) = 0$, so $\pi(x)$ is a solution to the original problem. Similarly, if $x_{n+1} = -1$, then $H(-\pi(x)) = 0$, so $-\pi(x)$ is a solution to the original problem. In the case where $|x_{n+1}| < 1$ we have that $\|\pi(x)\|_\infty = 1$ and so $H(\pi(x)) = -H(-\pi(x))$, because H is odd on the boundary. By definition of the selection-function Sel this implies that

$$F(x) = \mathrm{Sel}_2(-H(-\pi(x)), H(\pi(x), x_{n+1}) = \mathrm{Sel}_2(H(\pi(x)), H(\pi(x)), x_{n+1}) = H(\pi(x))$$

and similarly $F(-x) = -H(\pi(x))$. The equality $F(x) = F(-x)$ then implies that $H(\pi(x)) = H(-\pi(x)) = 0$, so both $\pi(x)$ and $-\pi(x)$ is a solution to the original instance in this case[1]. In conclusion, if we could recover the sign of x_{n+1} then we could define a solution map sending x to $\mathrm{sgn}(x_{n+1})\pi(x)$, but we do not allow this. However, in the approximate version, we may do this.

Proposition 23. *Any basic $\ell_p - \mathrm{BBU}_a$ problem polynomial time reduces to a basic basic $\ell_p - \mathrm{BU}_a$ problem.*

Proof. After changing domain we may assume that $p = \infty$. Given an instance (H, ϵ) of a basic $\ell_\infty - \mathrm{BBU}_a$ problem we apply the above construction and the map f outputs the instance (F, ϵ') of a basic $\ell_\infty - \mathrm{BU}_a$ problem where $\epsilon' = \min(\epsilon, 1/2)$. Now suppose that x is a solution to the problem (F, ϵ') . This means there exists some x^* with $\|x - x^*\|_\infty \le \epsilon'$ and $F(x^*) = F(-x^*)$. We now claim that we may map back the solution x of (F, ϵ') to a solution of (H, ϵ) by the map $g(x) = \mathrm{sgn}(x_{n+1})\pi(x)$.

If $|x_{n+1}| \ge 1/2$ then we have that $\mathrm{sgn}(x_{n+1}) = \mathrm{sgn}(x^*_{n+1})$ as $\epsilon' \le 1/2$. Therefore

$$\|\mathrm{sgn}(x)\pi(x) - \mathrm{sgn}(x^*)\pi(x^*)\|_\infty = \|\pi(x) - \pi(x^*)\|_\infty \le \|x - x^*\| \le \epsilon' \le \epsilon$$

Also $\mathrm{sgn}(x^*)\pi(x^*)$ is a zero of H by the discussion above the proposition. In the case where $|x_{n+1}| < 1/2$ we have that $|x^*_{n+1}| < 1$ and so both of $\pm\pi(x^*)$ is a zero of H. As $\mathrm{sgn}(x)\pi(x)$ is ϵ-close to $-\pi(x^*)$ or $\pi(x^*)$, we conclude that $g(x)$ is a solution to the problem (H, ϵ). □

Combining Proposition 22 and Proposition 23 we obtain the following result.

Theorem 5.5.1. $\mathrm{BU}_a = \mathrm{BBU}_a$

[1]We are grateful to Alexandros Hollender for noting that the reduction is possible without introducing approximation error.

5.6 Consensus Halving

In this section we present the proof of our main result Theorem 5.1.2. This result enables an additional structural result, given in Section 5.6.5 about the class of strong approximation problems $\mathrm{BU}_a = \mathrm{BBU}_a$, showing that the class is unchanged even when allowing root operations as basic operations.

Suppose we are given a basic $\ell_\infty - \mathrm{BBU}_a$ problem Π_a with circuits over the basis $\{+, -, *, \max, \min\}$. Let (I, k) denote an instance of Π_a and put $\varepsilon = 2^{-k}$. We may in polynomial time compute a circuit C defining a function $F\colon B^n_\infty \to \mathbb{R}^n$ that is odd on the boundary S^{n-1}_∞ such that $\mathrm{Sol}(I) = \{x \in B^n_\infty \mid F(x) = 0\}$. We now provide a reduction from Π_a to a CH_a-problem. In the reduction we will make use of the "almost implies near" paradigm.

Lemma 35. *Let $F\colon B^n_\infty \to \mathbb{R}^n$ be a continuous map. For any $\varepsilon > 0$ there is a $\delta > 0$ such that if $\|F(x)\|_\infty \leq \delta$ then there is an $x^* \in B^n_\infty$ such that $\|x - x^*\|_\infty \leq \varepsilon$ and $F(x^*) = 0$.*

Proof. Let F and $\varepsilon > 0$ be given. Suppose the claim is false. Then for any $n \in \mathbb{N}$ there is an x_n such that $\|F(x)\|_n \leq 1/n$ and if $x^* \in B^n_\infty$ has $\|x_n - x^*\|_\infty \leq \varepsilon$ then $F(x^*) \neq 0$. By compactness the Bolzano-Weierstrass theorem implies the existence of a subsequence $\{x_{n_i}\}$ converging to some $x^* \in B^n_\infty$. By continuity of F and $\|\cdot\|_\infty$ we get that $\|F(x^*)\|_\infty = \lim_{i\to\infty} \|F(x_{n_i})\|_\infty = 0$, showing that $F(x^*) = 0$. However, for sufficiently large $i \in \mathbb{N}$ it holds that $\|x_{n_i} - x^*\| \leq \varepsilon$ contradicting the choice of the x_n. □

This lemma says that for any $\varepsilon > 0$, if $\|F(x)\|_\infty$ is sufficiently close to being zero, then x is ε-close to a real zero of F. When F is computed by an algebraic circuit of polynomial size, it follows by the results in Section 5.2.7 that there exists some fixed polynomial q with integer coefficients such that the above lemma holds true for some $\delta \geq (\varepsilon)^{2^{q(|I|)}}$. The lemma then holds true for $\delta = (\varepsilon)^{2^{q(|I|)}}$, and we may construct this number using a circuit of polynomial size by repeatedly squaring the number ε exactly $q(|I|)$ times. This number will be used by the feedback agents in our CH_a instance in order to ensure that any solution gives a solution to the $\ell_\infty - \mathrm{BBU}_a$ instance.

5.6.1 Overview of the Reduction

Overview. As in previous works, we describe a consensus halving instance on an interval $A = [0, M]$, where M is bounded by a polynomial in $|I|$, rather than the interval $[0, 1]$. This instance may then be translated to an instance on the interval $[0, 1]$ by simple scaling. Like [99], in the leftmost end of the instance we place the *Coordinate-Encoding* region consisting of n intervals. In a solution S, these intervals will encode a value $x \in [-1, 1]^n$. A *circuit simulator C* will simulate the circuit of F on this value x. The circuit simulators will consist of a number of agents each implementing one gate of the circuit. However, such a circuit simulator may fail in simulating F properly, so we will use a polynomial number of circuit simulators $C_1, \ldots, C_{p(n)}$. Each of these circuit simulators will output n values $[C_j(x)]_1, \ldots, [C_j(x)]_n$ into intervals $I_{1j}, \ldots, I_{nj}$ immediately after the simulation. Finally, we introduce the so-called *feedback agents*

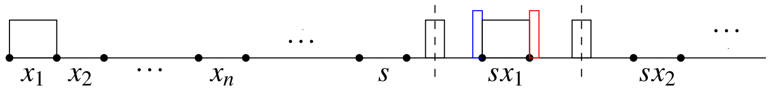
$f_1, \ldots, f_n$. The agent f_i will have some very thin *Dirac blocks* centered in each of the intervals I_{ij} where $j \in [p(n)]$. These agents will ensure that if z is an exact solution to the CH instance, then the encoded value x satisfies that $\|F(x)\|_\infty$ is sufficiently small that we may conclude that x is ε-close to a zero x^* of F.

Label Encoding. For a unit interval I we let $I^\pm$ denote the subsets of I assigned the corresponding label. We define the *label encoding* of I to be a value in $[-1, 1]$ given by the formula $v_l(I) := \lambda(I^+) - \lambda(I^-)$, where λ denotes the Lebesgue measure on the real line $\mathbb{R}$. This makes sense as $I^\pm$ is measurable, because they are the union of a finite number of intervals.

Coordinate-Encoding Region. The interval $[0, n]$ is called the *Coordinate-Encoding* region. For every $i \in [n]$, the subinterval $[i-1, i]$ of the Coordinate-Encoding region encodes a value $x_i := v_l([i-1, i])$ via the label encoding.

Position Encoding. For an an interval I which contains only a single cut, thus dividing I into two subintervals $I = I_a \cup I_b$, we define the *position encoding* of I to be the value $v_p(I) := \lambda(I_1) - \lambda(I_2)$. We note that $v_p(I) = v_l(I)$ if the labeling sequence is $-/+$, and $v_p(I) = -v_l(I)$ in the case the labeling sequence is $+/-$.

From Label to Position. Before a circuit simulator there is a *sign detection* interval I_s which detects the labeling sequence. Unless it contains a stray cut, this interval will encode a sign $s = \pm 1$ (to be precise 1 if the label is + and -1 is the label is $-$). By placing agents that flip the label as indicated below, we may now obtain position encodings of the values $sx_1, \ldots, sx_n$. These values will be read-in as inputs to the subsequent circuit simulator.



Circuit Simulators. As mentioned above, the circuit simulator C_j will read-in the values $s_jx_1, \ldots, s_jx_n$ and simulate the circuit computing F on this input. They then output their values into n intervals immediately after the simulation.

Feedback Agents. By the discussion after the proof of Lemma 35 we may by repeated squaring construct a circuit of polynomial size in $|I|$ computing a tiny number $\delta > 0$ such that if $\|F(x)\|_\infty \leq \delta$ then x is $(\varepsilon/2)$-close to a zero of F. Now fix $i \in [n]$ and let c_{ij} denote the centre of the feedback interval I_{ij} outputs the value $[C_j(s_j \cdot x)]_i$. We then define the ith feedback agent to have constant density $1/\delta$ in the intervals $[c_{ij} - \delta/2, c_{ij} + \delta/2]$.

The reason for having the feedback agents have these very narrow Dirac blocks is that if $F_i(x) > \delta$ for some i, then in any of the "uncorrupted" circuits (i.e. circuits outputting the correct values) all the density of the *i*th agent will contribute to the same label. Moreover, we will show using the boundary condition of F that the contribution is to the same label in all the uncorrupted circuit simulators. This will contradict that the feedback agents should value I^+ and I^- equally. That is the feedback agents ensure that $||F(x)||_\infty \leq \delta$ if x is the value encoded by an exact solution to the consensus halving instance we construct.

Stray Cuts. Any of the agents implementing one of the gates in a circuit simulator will force a cut to be placed in an interval in that same circuit simulator. The only agents whose cuts we have no control over are the n feedback agents.The expectation is that these agents should make cuts in the Coordinate-Encoding region that flip the label. If they do not do this we will call it a *stray cut*. If a circuit simulator contains a stray cut, we will say nothing about its value.

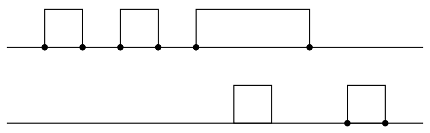
Observation 2. *If it is not the case that every unit interval encoding a coordinate x_i in the Coordinate-Encoding region contains a cut that flips the label, then the encoded point $x \in B^n_\infty$ will lie on the boundary S^n_∞. With this in mind we may ensure that $x \in S^n_\infty$ or $s_1 = s_2 = \cdots = s_{p(n)} = \pm 1$ where the sign is the same as the label of the first interval. This can be done by, if necessary, placing one single-block agent after the Coordinate-Encoding region and each of the circuit simulators (if placing such an agent is necessary depends on, respectively, the number of variables n and the size of the circuits).*

5.6.2 Construction of Gates

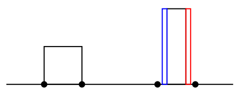
In this section we describe how to construct Consensus-Halving agents implementing the required gates $\{+, -, *, \max, \min\}$. First, we show that we may transform the circuit such that all gates only take values in the interval $[-1, 1]$ on input from B^n_∞.

Transforming the Circuit. By propagating every gate to the top of the circuit we may assume that the circuit is layered. Let C' denote the resulting circuit. By repeated squaring we may maintain a gate with value $1/2^{2^d}$ in the dth layer. Suppose $g = \alpha(g_1, g_2)$ is a gate with inputs g_1, g_2 in layer d. We modify the gates as follows: if $\alpha \in \{+, -, \max, \min\}$ then we multiply g_i by $1/2^{2^d}$ before applying α; if $\alpha = *$, then we multiply the input by 1 before applying α. Finally, we transform C' into the circuit C'' as follows: on input x, the circuit C'' multiplies the input by $1/2$ and then evaluates C' on input $x/2$. Inductively, one may show that if g is a gate in layer d in the circuit C', then the corresponding gate in in the circuit C'' has value $g/2^{2^d}$. As all the gates are among $\{+, -, *, \max, \min\}$, this ensures that all the gates in C'' take values in $[-1, 1]$.

Addition Gate [G_+]. We may construct an addition gate using two agents. The first agent has two unit input intervals that we assume contain one cut each. This then forces a cut in the long output interval that has length 3. The second agent then truncates this value.



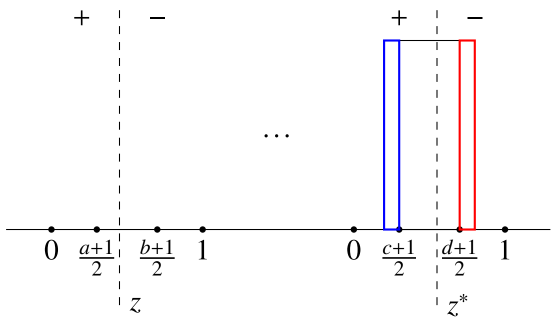
Constant Gate [G_ζ]. Let $\zeta \in [-1,1] \cap \mathbb{Q}$ be a rational constant. The agent will have a block of unit height in the sign interval and a block of width $\zeta/2$ and height $2/\zeta$ centered in another interval.



Before proceeding with the remaining gates, we construct a general function gate, an agent that implements any decreasing function.

Function Gate [G_h]. Let $-1 \le a < b \le 1$ and $-1 \le c < d \le 1$ be rational numbers and consider a continuously differentiable map $h\colon [a,b] \to [c,d]$ satisfying $h(a) = d$ and $h(c) = c$. Let $\overline{h}$ denote the extension of h that is constant on $[-1,a]$ and $[b,1]$. We now construct an agent with input interval I and output interval O computing this map, that is the agent should force a cut in the output interval such that $\overline{h}(v_p(I)) = v_p(O)$.

The agent that we construct has a block of height $2/(d-c)$ in the sub-interval $[(c+1)/2,(d+1)/2]$ of the output interval and density $f(z) := -2h'(2z-1)/(d-c)$ in the sub-interval $((a+1)/2,(b+1)/2)$ of the input interval. We note that f is positive in this interval as h is assumed to be a decreasing map, so it makes sense for the agent to have density f. One may verify that the agent values the input interval and output interval equally. We further add two rectangles to the output interval colored blue and red in the sketch below. These will ensure that if the cut in the input interval is placed at $z \le (a+1)/2$ such that $v_p(I) \le a$, then the cut in the output interval must be placed at $z^* = (d+1)/2$, meaning that $v_p(O) = d$. Similarly, if $v_p(I) \ge b$ then $v_p(O) = c$.



Suppose cuts are placed in z in the input interval and in z^* in the output interval. As the agent must value the parts with positive and negative label equally, we get the equality

$$1 = \int_{(a+1)/2}^{z} \tfrac{-2h'(2t-1)}{d-c}\, dt + (z^* - \tfrac{c+1}{2})\tfrac{2}{d-c}$$

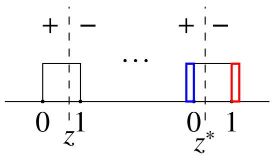
From this we obtain that

$$\begin{aligned} d - c &= -\int_{a}^{2z-1} h'(u)\, du + 2z^* - c - 1 \\ &= -h(2z-1) + d + 2z^* - c - 1 \end{aligned}$$

where we use that $h(a) = d$ by assumption. We conclude that $h(2z-1) = 2z^* - 1$, that is we obtain the equality $h(v_p(I)) = v_p(O)$.

Using this general function gate, we may now build up the remaining gates required by the circuit.

Multiplication By -1 Gate $[G_{-(\cdot)}]$**.** In order to realise this gate, we consider the function $h\colon [-1,1] \to [-1,1]$ given by $x \mapsto -x$. The agent's density function in the input interval is then given by $f(z) = 1$.



Subtraction Gate $[G_-]$**.** We may build this using the gates $G_{-(\cdot)}$ and G_+.

Multiplication by $\zeta \in [-1,1]$ $[G_{\cdot\zeta}]$**.** If $\zeta < 0$ we mahy construct $G_{\cdot\zeta}$ as a function gate using the function $h\colon [-1,1] \to [\zeta, -\zeta]$. If $\zeta > 0$ we construct using $-\zeta$ and a minus gate, i.e. $G_{\cdot\zeta} = -G_{\cdot(-\zeta)}$.

Maximum Gate $[G_{\max}]$**.** First we show how to construct a gate computing the absolute value of the input. We may construct gates G_1, G_2 such that $G_1(x) = -\max(x,0)$ and $G_2(x) = \max(-x,0)$ as function gates by using the functions $h_1\colon [0,1] \to [-1,0]$ given by $x \mapsto -x$ and $h_2\colon [-1,0] \to [0,1]$ given by $x \mapsto -x$. Now, we may constrcuct the absolute value gate as $G_{|\cdot|} = -G_1 + G_2$. We may now construct $G_{\max}$ by using the formula $\max(x,y) = (x + y + |x - y|)/2$.

Minimum Gate $[G_{\min}]$**.** We may build this using $\min(x,y) = x + y - \max(x,y)$.

Multiplication Gate $[G_*]$**.** We start off by constructing a gate squaring the input. First we construct G_1 and G_2 as function gates with respect to $h_1\colon [-1,0] \to [0,1]$ given by $x \mapsto x^2$ and $h_2\colon [0,1] \to [-1,0]$ given by $x \mapsto -x^2$. Then we may construct the squaring gate as $G_{(\cdot)^2} = G_1 - G_2$. Now we may use the previously constructed gates to make a multiplication gate via the identity $xy = ((x+y)^2 - x^2 - y^2)/2$.

5.6.3 Describing valuation functions as circuits.

In the description above, we described the valuations of the agents by providing formulas for their densities. However, an instance of CH actually consists of a list of algebraic circuits computing the distribution functions of the agents. In order to construct gates, it is sufficient for agents to have densities that are piece-wise polynomial. Therefore, consider an agent with polynomial densities f_i in the intervals $[a_i, b_i)$ for $i = 1, \ldots, s$, and let F_i denote the indefinite integral of f_i. We note that F_i is a polynomial so it may be computed by an algebraic circuit. Now we claim that the distribution function of this agent may be computed by an algebraic circuit via the formula

$$F(x) = \sum_{i=1}^{s} [F_i(\max(a_i, \min(x, b_i))) - F_i(a_i)] \tag{5.3}$$

This is the case, because the summands will be equal to $F_i(a_i) - F_i(a_i) = 0$ if $x < a_i$, to $F_i(x) - F_i(a)$ if $a_i \le x \le b_i$ and to $F_i(b) - F_i(a)$ if $x > b_i$, meaning that this formula does indeed calculate the valuation of the agent in the interval $[0, x]$.

5.6.4 Reduction and Correctness

Recall that we are given an instance (F, ε) of the BBU$_a$ problem and that we have to construct an instance of the CH$_a$ problem. The reduction now outputs an instance of the CH$_a$ problem where the consensus halving instance is constructed as above with $p(n) = 2n + 1$ circuit simulators and the approximation parameter is given by $\varepsilon' = \varepsilon/(4n)$. Let z denote a solution to this CH$_a$ instance. By definition, there exists an exact solution z^* to the consensus-halving problem such that $||z - z^*||_\infty \le \varepsilon'$.

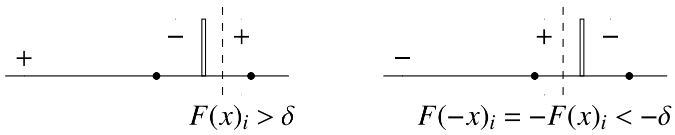
Let x and x^* denote the values encoded by respectively z and z^* in the Coordinate-Encoding region. Suppose, generally, we are given an interval I with a number of cut points $t_1, \ldots, t_s$. Moving a cut point by a distance $\le \varepsilon'$ we create a new interval I'. This changes the label encoding by at most $2\varepsilon'$, that is $|v_l(I) - v_l(I')| \le 2\varepsilon'$. Succesively, if we move all the cuts by a distance $\le \varepsilon'$, then we get an interval I^* such that $|v_l(I) - v_l(I^*)| \le 2s\varepsilon'$. As $||z - z^*||_\infty \le \varepsilon'$ and any of the subintervals in the Coordinate-encoding region can contain at most n cuts, we conclude that $||x - x^*||_\infty \le 2n\varepsilon' = 2n(\varepsilon/(4n)) = \varepsilon/2$. In order to show that x is ε-close to a zero of F, it now suffices by the triangle inequality to show that x^* is $(\varepsilon/2)$-close to a zero of F. This will follow from the two following lemmas.

Lemma 36. *If there are no stray cuts in the exact solution z^*, then the associated value x^* encoded in the Coordinate-encoding region satisfies $F(x^*) = 0$.*

Proof. We recall that if the solution z^* contain no stray cuts, then the signs of all the circuit simulators are equal $s_1 = \cdots = s_{2n+1} = s$ where $s = \pm 1$. Furthermore, all the circuit simulators will output the same values $F_1(sx^*), \ldots, F_n(sx^*)$ into the feedback intervals. Thus, there can be no cancellation, so in order for the feedback agents to value the positive and negative part equally it must be the case that $F(sx^*) = 0$. □

Lemma 37. *If there is a stray cut in the exact solution z^*, then the associated value x^* encoded in the Encoding-region satisfies the inequality $\|F(x^*)\|_\infty \le \delta$.*

Proof. Suppose toward contradiction that $|F(x)_i| > \delta$ for some i. Without loss of generality we assume that $F(x)_i > \delta$. As there is a stray cut, the Coordinate-Encoding region can contain at most $n-1$ cuts. Thus, at least one of the coordinates x_i^* must be ± 1 showing that $x^* \in S^{n-1}$. From this and the boundary condition we conclude that $F(x^*) = -F(-x^*)$. Furthermore, there is at most n stray cuts, so at most n circuit simulators can become corrupted. This means that $n+1$ circuit simulators work correctly. Now suppose that the circuit simulator C_j is uncorrupted. If the label is $s_j = +1$, then C_j will output $F(x)$ into the feedback region and the labeling sequence will be $+/-$; if the label is $s_j = -1$ then C_j will output $F(-x) = -F(x)$ into the feedback region and the labeling sequence will be $-/+$. This is indicated below:



From this we conclude that the $n+1$ uncorrupted circuit simulators altogether contribute $(n+1)\delta$ to the part with negative label. However, the n corrupted circuit simulators can contribute at most $n\delta$ to the part with positive label. This implies that f_i cannot value the negative and positive part equally. This contradicts the assumption that z^* is an exact consensus-halving. We conclude that $\|F(x^*)\|_\infty \le \delta$. □

By the two lemmas above, it follows that the value x^* encoded by the exact consensus-halving z^* satisfies the inequality $\|F(x^*)\|_\infty \le \delta$. By choice of δ, this implies that there exists some x^{**} such that $\|x^* - x^{**}\|_\infty \le \varepsilon/2$ and $F(x^{**}) = 0$. From the discussion before the two lemmas, it follows that x is ε-close to a zero of F and is thus a solution to the BBU_a instance (F, ε).

Mapping back a Solution. What remains is to show that we may recover a solution x to the BBU_a instance from the solution z to the CH_a instance. Recall that in a solution $z = (z_1, \ldots, z_N)$ to the consensus-halving problem $|z_i|$ and $\mathrm{sgn}(z_i)$ represents the length and label of the ith interval. For $i \le n$ and $j \le n+1$ we introduce

$$\begin{aligned} t_j &= \sum_{k=1}^{j-1} |z_k| \\ x_{ij}^+ &= \max(0, \min(t_{j-1} + z_j, i) - \max(t_{j-1}, i-1)) \\ x_{ij}^- &= \max(0, \min(t_{j-1} - z_j, i) - \max(t_{j-1}, i-1)) \end{aligned}$$

These numbers may be computed efficiently by a circuit over $\{+, -, \max, \min\}$. We notice that if $z_j > 0$ then $x_{ij}^- = 0$ (and if $z_j < 0$ then $x_{ij}^+ = 0$). Furthermore, by checking a couple of cases, one finds that if $z_j > 0$ (respectively $z_j < 0$) then x_{ij}^+ (respectively x_{ij}^-) is the length of the jth interval that is contained in $[i-1, i]$. As the coordinate-encoding region can contain at most n cuts (corresponding to at most $n+1$ intervals), we deduce from the above that the values encoded can be computed as

$$x_i = \sum_{j=1}^{n+1} x_{ij}^{+} - x_{ij}^{-}$$

for every $i \leq n$. If there is a stray cut then both x and $-x$ are valid solutions by the boundary condition of F. If there is no stray cut, then $s_1 = s_2 = \cdots = s_{p(n)} = s = \mathrm{sgn}(z_1)$ by Observation 2 and in this case we may recover a solution as sx.

5.6.5 Removing Root Gates.

In this subsection, we argue by going through CH_a that the strong approximation problems $\mathrm{BU}_a = \mathrm{BBU}_a$ do not change even if we allow the circuits to use root-operations as basic operations.

Proposition 24. *The class $\ell_\infty - \mathrm{BBU}_a$ remains unchanged even if we allow the circuits to use root-gates.*

Proof. Let Π_a be a basic $\ell_\infty - \mathrm{BBU}_a$ problem where the circuits are allowed to use gates from the basis $\{+,-,*,\max,\min,\sqrt[k]{}\}$. In the previous section, we constructed a polynomial time reduction from Π_a to a CH_a problem Γ_a in such a way that the circuits computing the distribution functions of the agents are defined over $\{+,-,*,\max,\min\}$. Namely, the root gates can be implemented by first noting that the power-gate $(\cdot)^k$ can be implemented by an agent with polynomial densities by using the general function gate construction. Then, in order to construct an agent implementing the root gate we simply interchange the input interval and output interval of the power-gate. By the proof of the result of Deligkas et al. that CH is contained in BU, the problem Γ_a polynomial time reduces to a $\ell_1 - \mathrm{BU}_a$ problem Λ that only uses gates from $\{+,-,*,\max,\min\}$. By Proposition 22, Λ reduces to a basic $\ell_1 - \mathrm{BBU}_a$ problem Ξ which again uses only gates from $\{+,-,*,\max,\min\}$. Finally, by Proposition 18, Ξ reduces to a basic $\ell_\infty - \mathrm{BBU}_a$ problem, again using only gates from $\{+,-,*,\max,\min\}$. Altogether, we see that Π_a polynomial time reduces to a $\ell_\infty - \mathrm{BBU}_a$ without root-gates. □

Chapter 6

The Frontier of Intractability for EFX with Two Agents

Abstract

We consider the problem of sharing a set of indivisible goods among agents in a fair manner, namely such that the allocation is envy-free up to any good (EFX). We focus on the problem of computing an EFX allocation in the two-agent case and characterize the computational complexity of the problem for most well-known valuation classes. We present a simple greedy algorithm that solves the problem when the agent valuations are weakly well-layered, a class which contains gross substitutes and budget-additive valuations. For the next largest valuation class we prove a negative result: the problem is PLS-complete for submodular valuations. All of our results also hold for the setting where there are many agents with identical valuations.

6.1 Introduction

The field of fair division studies the following fundamental question: given a set of resources, how should we divide them among a set of agents (who have subjective preferences over those resources) in a fair way? This question arises naturally in many settings, such as divorce settlement, division of inheritance, or dissolution of a business partnership, to name just a few. Although the motivation for studying this question is perhaps almost as old as humanity itself, the first mathematical investigation of the question dates back to the work of Banach, Knaster and Steinhaus [205, 207].

Of course, in order to study fair division problems, one has to define what exactly is meant by a *fair* division. Different fairness notions have been proposed to formalize this. Banach, Knaster and Steinhaus considered a notion which is known today as *proportionality*: every agent believes that it obtained at least a fraction $1/n$ of the total value available, where n is the number of agents. A generally[1] stronger notion, and one which seems more adapted to the motivating examples we mentioned above,

[1]As long as agents' valuations are subadditive, every envy-free division also satisfies proportionality.

is that of *envy-freeness* [103, 108, 218]. A division of the resources is said to be envy-free, if no agent is envious, i.e., no agent values the bundle of resources obtained by some other agent strictly more than what it obtained itself.

As our motivating examples already suggest, the case with few agents – in fact, even just with two agents – is very relevant in practice. When the resources are divisible, such as for example money, water, oil, or time, the fair division problem with two agents admits a very simple and elegant solution: the cut-and-choose algorithm, which already appears in the Book of Genesis. As its name suggests, in the cut-and-choose algorithm one agent cuts the resources in half (according to its own valuation), and the other agent chooses its preferred piece, leaving the other piece to the first agent. It is easy to check that this guarantees envy-freeness, among other things. The case of divisible resources, which is usually called *cake cutting*, has been extensively studied for more than two agents. One of the main objectives in that line of research can be summarized as follows: come up with approaches that achieve similar guarantees to cut-and-choose, but for more than two agents. This has been partially successful, and notable results include the proof of the existence of an envy-free allocation for any number of agents [210, 213, 228], as well as a finite, albeit very inefficient, protocol for computing one [10].

In many cases, however, assuming that the resources are divisible might be too strong an assumption. Indeed, some resources are inherently *indivisible*, such as a house, a car, or a company. Sometimes these resources can be made divisible by sharing them over time, for example, one agent can use the car over the week-end and the other agent on weekdays. But, in general, and in particular when agents are not on friendly terms with each other, as one would expect to often be the case for divorce settlements, this is not really an option.

Indivisible resources make the problem of finding a fair division more challenging. First of all, in contrast to the divisible setting, envy-free allocations are no longer guaranteed to exist. Indeed, this is easy to see even with just two agents and a single (indivisible) good that both agents would like to have. No matter who is given the good, the other agent will envy them. In order to address this issue of non-existence of a solution, various relaxations of envy-freeness have been proposed and studied in the literature. The strongest such relaxation, namely the one which seems closest to perfect envy-freeness, is called *envy-freeness up to any good* and is denoted by EFX [43, 131]. An allocation is EFX if for all agents i and j, agent i does not envy agent j, after removal of *any* single good from agent j's bundle. In other words, an allocation is *not* EFX, if and only if there exist agents i and j, and a good in j's bundle, so that i envies j's bundle even after removal of that good.

For this relaxed notion of envy-freeness, it is possible to recover existence, at least in some cases. An EFX allocation is guaranteed to exist for two agents with any monotone valuations [185], and for three agents if we restrict the valuations to be additive [45]. It is currently unknown whether it always exists for four or more agents, even just for additive valuations.

Surprisingly, proving the existence of EFX allocations for two agents is non-trivial. In order to use the cut-and-choose approach, we need to be able to “cut in half”. In the divisible setting, this is straightforward. But, in the indivisible setting, we need to "cut in half in the EFX sense," i.e., divide the goods into two bundles such that the first agent is EFX with either bundle. In other words, we first need to show the existence of EFX allocations for two identical agents, namely two agents who share the same valuation function, which is not a trivial task.

Plaut and Roughgarden [185] provided a solution to this problem by introducing the *leximin++ solution*. Given a monotone valuation function, they defined a total ordering over all allocations called the leximin++ ordering. They proved that for two identical agents, the leximin++ solution, namely the global maximum with respect to the leximin++ ordering, must be an EFX allocation. As mentioned above, using the cut-and-choose algorithm, this shows the existence of EFX allocations for two, possibly different, agents. Unfortunately, computing the leximin++ solution is computationally intractable[2] and so, while this argument proves the existence of EFX allocations, it does not yield an efficient algorithm.

Nevertheless, for two agents with *additive* valuations, Plaut and Roughgarden [185] provided a polynomial-time algorithm based on a modification of the Envy-Cycle elimination algorithm of Lipton et al. [163]. They also provided a lower bound for the problem in the more general class of submodular valuations, but not in terms of computational complexity (i.e., not in the standard Turing machine model). Namely, they proved that for two identical agents with submodular valuations computing an EFX allocation requires an exponential number of queries in the query complexity model.

Their work naturally raises the following two questions about the problem of computing an EFX allocation for two agents:

1. What is the *computational* complexity of the problem for submodular valuations?

2. What is the computational complexity of the problem for well-known valuation classes lying between additive and submodular,[3] such as gross substitutes, OXS, and budget-additive?

Note that it does not make sense to study the query complexity for additive valuations, since a polynomial number of queries is sufficient to reconstruct the whole valuation functions (and the amount of computation then needed to determine a solution is not measured in the query complexity). However, it does make sense to study the computational complexity of the problem for submodular valuations, as well as other classes beyond additive. The query lower bound by Plaut and Roughgarden essentially says that many queries are needed in order to gather enough information

[2]Computing the leximin++ solution is NP-hard, even for two identical agents with additive valuations. This can be shown by a reduction from the PARTITION problem (see [185, Footnote 7] and note that the argument also applies to leximin++).

[3]In particular, Plaut and Roughgarden [185, Section 7] propose studying the complexity of fair division problems with respect to the hierarchy of complement-free valuations (*additive* $\subseteq$ *OXS* $\subseteq$ *gross substitutes* $\subseteq$ *submodular* $\subseteq$ *XOS* $\subseteq$ *subadditive*) introduced by Lehmann et al. [159].

about the submodular valuation function to be able to construct an EFX allocation. But it does not say anything about the *computational* hardness of finding an EFX allocation. Their lower bound does not exclude the possibility of a polynomial-time algorithm for submodular valuations in the standard Turing machine model. Studying the problem in the computational complexity model allows us to investigate how hard it is to solve when the valuation functions are given in some succinct representation, e.g., as a few lines of code, or any other form that allows for efficient evaluation.

Our contribution. We answer both of the aforementioned questions:

1. For submodular valuations, we prove that the problem is PLS-complete in the standard Turing machine model, even with two identical agents.
2. We present a simple greedy algorithm that finds an EFX allocation in polynomial time for two agents with *weakly well-layered* valuations, a class of valuation functions that we define in this paper and which contains all well-known strict subclasses of submodular, such as gross substitutes, OXS, and budget-additive.

Together, these two results completely resolve the computational complexity of the problem for all valuation classes in the standard complement-free hierarchy (*additive* ⊆ *OXS* ⊆ *gross substitutes* ⊆ *submodular* ⊆ *XOS* ⊆ *subadditive*) introduced by Lehmann et al. [159]. Furthermore, just like in the work of Plaut and Roughgarden [185], our negative and positive results also hold for any number of *identical* agents.

Regarding the PLS-completeness result, the membership in PLS is easy to show using the leximin++ ordering of Plaut and Roughgarden [185]. The PLS-hardness is more challenging. The first step of our hardness reduction is essentially identical to the first step in the corresponding query lower bound of Plaut and Roughgarden [185]: a reduction from a local optimization problem on the Kneser graph to the problem of finding an EFX allocation. The second step of the reduction is our main technical contribution: we prove that finding a local optimum on a Kneser graph is PLS-hard[4], which might be of independent interest.

Further related work. The existence and computation of EFX allocations has been studied in various different settings, such as for restricted versions of valuation classes [6, 14], when some items can be discarded [26, 42, 46, 47], or when valuations are drawn randomly from a distribution [165].

A weaker relaxation of envy-freeness is *envy-freeness up to one good* (EF1) [40, 163]. It can be computed efficiently for any number of agents with monotone valuations using the Envy-Cycle elimination algorithm [163]. If one is also interested in economic efficiency, then it is possible to obtain an allocation that is both EF1

[4]We note that proving a tight computational complexity lower bound is more challenging than proving a query lower bound, because we have to reduce from problems with more structure. Indeed, the exponential query lower bound for the Kneser problem (and thus also for the EFX problem) can easily be obtained as a byproduct of our reduction.

and Pareto-optimal in pseudopolynomial time for additive valuations [20]. For more details about fair division of indivisible items, we refer to the two recent surveys by Amanatidis et al. [7] and Aziz et al. [12].

Outline. We begin with 6.2 where we formally define the problem and solution concept, as well as some standard valuation classes of interest. In 6.3 we introduce *weakly well-layered* valuation functions, and present our simple greedy algorithm for computing EFX allocations. Finally, in 6.4 we prove our main technical result, the PLS-completeness for submodular valuations.

6.2 Preliminaries

We consider the problem of discrete fair division where an instance consists of a set of agents N, a set of goods M, and for every agent $i \in N$ a valuation function $v_i \colon 2^M \to \mathbb{R}_{\geq 0}$ assigning values to bundles of goods. All valuation functions will be assumed to be *monotone*, meaning that for any subsets $S \subseteq T \subseteq M$ it holds that $v(S) \leq v(T)$, and *normalized*, i.e., $v(\emptyset) = 0$.

We now introduce the different types of valuation functions that are of interest to us. A valuation $v \colon 2^M \to \mathbb{R}_{\geq 0}$ is *additive* if $v(S) = \sum_{g \in S} v(\{g\})$ for every $S \subseteq M$. The hardness result we present in 6.4 holds for *submodular* valuations. These are valuations that satisfy the following diminishing returns condition that whenever $S \subseteq T$ and $x \notin T$ it holds that $v(S \cup \{x\}) - v(S) \geq v(T \cup \{x\}) - v(T)$.

Next, for our results in the positive direction, we introduce the classes of *gross substitutes* and *budget-additive* valuations, both contained in the class of submodular valuations. Before defining gross substitutes valuations, we have to introduce some notation. For a price vector $p \in \mathbb{R}^m$ on the set of goods, where $m = |M|$, the function v_p is defined by $v_p(S) = v(S) - \sum_{g \in S} p_g$ for any subset $S \subseteq M$, and the demand set is $D(v, p) = \arg\max_{S \subseteq M} v_p(S)$. A valuation v is *gross substitutes* if for any price vectors $p, p' \in \mathbb{R}^m$ with $p \leq p'$ (meaning that $p_g \leq p'_g$ for all $g \in M$), it holds that if $S \in D(v, p)$, then there exists a demanded set $S' \in D(v, p')$ such that $\{g \in S : p_g = p'_g\} \subseteq S'$. That is to say, if some good g is demanded at prices p and the prices of some *other* goods increase, then g will still be demanded. These valuations have various nice properties, for instance guaranteeing existence of Walrasian equilibria [133]. Lastly, a valuation v is *budget-additive* if it is of the form $v(S) = \min\{B, \sum_{g \in S} w_g\}$ for reals $B, w_1, \ldots, w_m \geq 0$. [159] show that a budget-additive valuation need not satisfy the gross substitutes condition. See 6.1 for the relationship between the valuation classes.

Envy-freeness up to any good (EFX). The goal of fair division is to find an allocation of the goods to the agents (i.e., a partitioning $M = X_1 \sqcup \cdots \sqcup X_n$) satisfying some notion of fairness. One might hope for an *envy-free* division in which every agent prefers his own bundle over the bundle of any other agent, that is, $v_i(X_i) \geq v_i(X_j)$ for all $i, j \in N$. Such a division need not exist, however, as can be seen in the case where one has to divide one good among two agents, as already mentioned in the

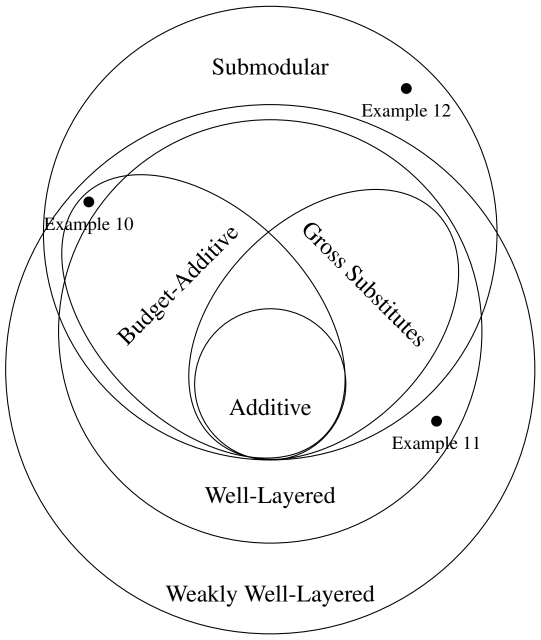


Figure 6.1: Inclusions of valuation classes

introduction. Therefore various weaker notions of fairness have been studied. In this paper we consider the notion of *envy-freeness up to any good* (EFX) introduced by Caragiannis et al. [43], and before that by Gourvès et al. [131] under a different name. An allocation $(X_1,\ldots,X_n)$ is said to be EFX if for any $i,j \in N$ and any $g \in X_j$ it holds that $v_i(X_i) \geq v_i(X_j \setminus \{g\})$.

6.3 Polynomial-time Algorithm for Weakly Well-Layered Valuations

In this section we present our positive result, namely the polynomial-time algorithm for computing an EFX allocation for two agents with $weaklywell-layered$ valuations. To be more precise, our algorithm works for any number of agents that all share the same $weaklywell-layered$ valuation function. As a result, using cut-and-choose it can then be used to solve the problem with two possibly *different* agents. We begin with the definition of this new class of valuations, and then present the algorithm and prove its correctness.

6.3.1 Weakly Well-Layered Valuations

We introduce a property of valuation functions and situate this with respect to well-known classes of valuation functions.

Definition 6.3.1. A valuation function $v\colon 2^M \to \mathbb{R}_{\geq 0}$ is said to be *weakly well-layered* if for any $M' \subseteq M$ the sets $S_0, S_1, S_2, \dots$ obtained by the greedy algorithm (that is, $S_0 = \emptyset$ and $S_i = S_{i-1} \cup \{x_i\}$ where $x_i \in \arg\max_{x \in M' \setminus S_{i-1}} v(S_{i-1} \cup \{x\})$ for $1 \leq i \leq |M'|$) are optimal in the sense that $v(S_i) = \max_{S \subseteq M' \colon |S| = i} v(S)$ for all i.

We can reformulate this definition as follows: a valuation function v is weakly well-layered if and only if, for all $M' \subseteq M$ and all i, the optimization problem

$$\begin{aligned} \max \quad & v(S) \\ \text{s.t.} \quad & S \subseteq M' \\ & |S| \leq i \end{aligned} \tag{6.1}$$

can be solved by using the natural greedy algorithm. Note that since we only consider monotone valuations, we can also use the condition $|S| = i$ instead of $|S| \leq i$.

The reformulation of the definition in terms of the optimization problem (6.1) is reminiscent of one of the alternative definitions of a matroid. Consider the optimization problem

$$\begin{aligned} \max \quad & v(S) \\ \text{s.t.} \quad & S \in \mathcal{F} \end{aligned} \tag{6.2}$$

where $v \colon 2^M \to \mathbb{R}_{\geq 0}$ is a valuation function and $\mathcal{F}$ is an independence system on M. Then, it is well-known that $\mathcal{F}$ is a matroid, if and only if, for all additive valuations v, the optimization problem (6.2) can be solved by the natural greedy algorithm [82, 107, 189]. In other words, the class of set systems (namely, matroids) is defined by fixing a class of valuations (namely, additive). The alternative definition of weakly well-layered valuations given in (6.1) can be viewed as doing the opposite: the class of valuations (namely, weakly well-layered) is defined by fixing a class of set systems (namely, all uniform matroids on subsets $M' \subseteq M$, or, more formally, $\mathcal{F} = \{S \subseteq M' : |S| \leq i\}$ for all $M' \subseteq M$ and all i).

Relationship to other valuation classes. We begin by showing that any gross substitutes valuation is weakly well-layered. In particular, this also implies that OXS valuations, which are a special case of gross substitutes, are also weakly well-layered. Paes Leme [180] proved that gross substitutes valuation functions satisfy the stronger condition of being *well-layered*, that is, for any $p \in \mathbb{R}^m$ it holds that if $S_0, S_1, S_2, \dots$ is constructed greedily with respect to the valuation v_p, where $v_p(S) := v(S) - \sum_{g \in S} p_g$, then S_i satisfies that $S_i \in \arg\max_{S \subseteq M \colon |S| = i} v_p(S)$.

Lemma 38. *If $v \colon 2^M \to \mathbb{R}_{\geq 0}$ is well-layered, then it is also weakly well-layered. In particular, gross substitutes valuations are weakly well-layered.*

Proof. Assume that $v \colon 2^M \to \mathbb{R}_{\geq 0}$ is well-layered and let $M' \subseteq M$. Assume that the sequence $S_0, S_1, S_2, \dots$ is constructed via the greedy algorithm: that is $S_0 = \emptyset$ and $S_i = S_{i-1} \cup \{x_i\}$ where $x_i \in \arg\max_{x \in M' \setminus S_{i-1}} v(S_{i-1} \cup \{x\})$ for $1 \leq i \leq |M'|$. We have to show that $v(S_i) = \max_{S \subseteq M' \colon |S| = i} v(S)$.

In order to exploit the assumption that v is well-layered, we introduce a price vector $p \in \mathbb{R}^m$ given by

$$p_g = \begin{cases} 0 & g \in M' \\ v(M) + 1 & g \notin M' \end{cases}$$

One sees that the sequence $S_0, S_1, S_2, \ldots$ can occur via the greedy algorithm for the valuation v_p, because goods not in M' cannot be chosen as their prices are too high. As v is well-layered, we have that $v_p(S_i) = \max_{S \subseteq M \colon |S|=i} v_p(S)$. As $p_g = 0$ for all $g \in M'$, this implies that $v(S_i) = \max_{S \subseteq M' \colon |S|=i} v(S)$. We conclude that v is weakly well-layered. □

Closure property. We note that the class of weakly well-layered valuations is closed under composition with a strictly increasing function.

Lemma 39. *Let $v \colon 2^M \to \mathbb{R}_{\geq 0}$ be weakly well-layered and let $f \colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ strictly increasing. Then the composition $f \circ v \colon 2^M \to \mathbb{R}_{\geq 0}$ is weakly well-layered.*

Proof. Let $M' \subseteq M$ and assume that $S_0, S_1, S_2, \ldots$ are constructed greedily, that is $S_0 = \emptyset$ and $S_i = S_{i-1} \cup \{x_i\}$ where $x_i \in \arg\max_{x \in M' \setminus S_{i-1}} f(v(S_{i-1} \cup \{x\}))$ for $1 \leq i \leq |M'|$. As f is strictly increasing, we see that $x_i \in \arg\max_{x \in M'} f(v(S_{i-1} \cup \{x\}))$ if and only if $x_i \in \arg\max_{x \in M'} v(S_{i-1} \cup \{x\})$. Therefore $S_0, S_1, S_2, \ldots$ could also arise via the greedy construction based on the valuation v. As v is weakly well-layered, this implies that $v(S_i) = \max_{S \subseteq M' \colon |S|=i} v(S)$ for all i. As f is increasing, this shows that $f(v(S_i)) = \max_{S \subseteq M' \colon |S|=i} f(v(S))$ for all i. We conclude that $f \circ v$ is weakly well-layered. □

Lemma 40. *Let $v \colon 2^M \to \mathbb{R}_{\geq 0}$ be weakly well-layered and $B \geq 0$. Then the valuation $u \colon 2^M \to \mathbb{R}_{\geq 0}$ given by $u(S) = \min(v(S), B)$ is weakly well-layered.*

Proof. Let $S_0, S_1, S_2, \ldots$ be constructed greedily from the valuation u. Suppose that $S_0, S_1, \ldots, S_k$ have utility $< B$ and that $S_{k+1}, S_{k+2}, \ldots$ have utility B. As $x \mapsto \min(x, B)$ is strictly increasing on $[0, B)$, the sets $S_0, S_1, \ldots, S_k$ could have been constructed greedily from v. As v is weakly well-layered, they are therefore optimal of their given size for v and therefore also for u. The sets $S_{k+1}, \ldots$ all have maximal utility B and are therefore optimal of their given sizes. □

As a corollary it follows that the class of budget-additive valuations satisfies the weakly well-layered property.

Corollary 8. *Any budget-additive valuation is weakly well-layered.*

In contrast, we note that a budget-additive valuation is not necessarily well-layered, as the following example shows.

Algorithm 1 Greedy EFX

```
Input: $N, M, v$
Output: EFX allocation
  Let $A_i = \emptyset$ for $i \in N$.
  Let $R = M$.
  while $R \neq \emptyset$ do
    $i = \arg\min_{j \in N} v(A_j)$
    $g = \arg\max_{x \in R} v(A_i \cup \{x\})$
    $A_i = A_i \cup \{g\}$
    $R = R \setminus \{g\}$
  end while
  return $(A_1, \ldots, A_n)$
```

Example 10. Consider the budget-additive valuation on three goods a, b, c with values $v_a = v_b = 2$, $v_c = 4$ and a budget of $B = 4$. Let $p = (1, 1, 2)$ be a price vector. Under these prices, the greedy algorithm would pick good c as its first item. However, $\{a, b\}$ is the unique optimal bundle of size 2, and so the greedy algorithm would fail in this case. As a result, the valuation is not well-layered.

The results of this subsection are summarised in 6.1. Note also that the classes of submodular valuations and weakly well-layered valuations are incomparable. For an example of a valuation function that is submodular but not weakly well-layered, see 12 in the next section. For the other direction, see the following example of a valuation that is well-layered (and thus weakly well-layered), but not submodular.

Example 11. Consider the valuation function v on two goods a, b given by $v(\{a, b\}) = 1$ and $v(\emptyset) = v(\{a\}) = v(\{b\}) = 0$. This valuation function is seen to be well-layered (and thus weakly well-layered), because subsets of equal size have the same valuation. However, it is not submodular, because $v(\{a\} \cup \{b\}) - v(\{a\}) = 1 > 0 = v(\emptyset \cup \{b\}) - v(\emptyset)$.

6.3.2 The Greedy EFX Algorithm

We now present a simple algorithm that computes an EFX allocation for many agents that all share the same weakly well-layered valuation function v.

Theorem 6.3.1. *If the valuation function v is weakly well-layered, then the output of Algorithm 1 is EFX. In particular, by using the cut-and-choose protocol one may compute an EFX allocation for two agents with different valuations as long as one of these valuations is weakly well-layered.*

Proof. We show that the algorithm maintains a partial EFX allocation throughout. Initially the partial allocation is empty and so clearly EFX. Suppose that at the beginning of some round the current partial allocation $(X_1, \ldots, X_n)$ is EFX and that some agent $i \in N$ receives a good g in this round. We have to show that the new (partial) allocation $(X'_1, \ldots, X'_n)$ is EFX, where $X'_i = X_i \cup \{g\}$ and $X'_j = X_j$ for $j \neq i$.

Clearly, the only thing we have to argue is that $v(X'_i \setminus \{g'\}) \leq v(X'_j)$ for all $j \in N$ and all $g' \in X'_i$. As i received a good in the current round we have that $v(X_i) \leq v(X_j) = v(X'_j)$. Therefore, it suffices to argue that $v(X'_i \setminus \{g'\}) \leq v(X_i)$ for all $g' \in X'_i$. This last inequality follows from v being weakly well-layered by taking $M' = X'_i$. With this M', the set X_i could namely be produced by running the greedy algorithm. Therefore, X_i is an optimal subset of $M' = X'_i$ of size $|X_i| = |X'_i| - 1$, meaning that $v(X'_i \setminus \{g'\}) \leq v(X_i)$ for all $g \in X'_i$. □

The algorithm can fail to provide an EFX allocation for submodular valuations that are not weakly well-layered, as the following example shows.

Example 12. Consider an instance with two agents and four goods denoted a, b, c, d, where the valuation function v is given by: $v(\{a\}) = 11, v(\{b\}) = v(\{c\}) = 10, v(\{d\}) = 16, v(\{a,b\}) = 15, v(\{a,c\}) = 15, v(\{b,c\}) = 17, v(\{a,b,c\}) = 18$, and $v(S) = 18$ for all sets S that satisfy $d \in S$ and $|S| \geq 2$. It can be checked by direct computation that v is indeed submodular. The greedy EFX algorithm yields: agent 1 gets good d, and then agent 2 gets goods a, b, c. This allocation is not EFX, because $v(\{d\}) < v(\{b,c\})$.

6.4 PLS-completeness for Submodular Valuations

Total NP**search problems** (TFNP). A total search problem is given by a relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ that satisfies: for all $x \in \{0,1\}^*$, there exists $y \in \{0,1\}^*$ such that $(x,y) \in R$. The relation R is interpreted as the following computational problem: given $x \in \{0,1\}^*$, find some $y \in \{0,1\}^*$ such that $(x,y) \in R$. The class TFNP[169] is defined as the set of all total search problems R such that the relation R is polynomial-time decidable (i.e., given some x, y we can check in polynomial time whether $(x,y) \in R$) and polynomially balanced (i.e., there exists some polynomial p such that $|y| \leq p(|x|)$ whenever $(x,y) \in R$).

Let R and S be two problems in TFNP. We say that R reduces to S if there exist polynomial-time functions $f : \{0,1\}^* \to \{0,1\}^*$ and $g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that for all $x, y \in \{0,1\}^*$: if $(f(x), y) \in S$, then $(x, g(y,x)) \in R$. In other words, f maps an instance of R to an instance of S, and g maps back any solution of the S-instance to a solution of the R-instance.

Polynomial Local Search (PLS). Johnson et al. [147] introduced the class PLS, a subclass of TFNP, to capture the complexity of computing locally optimal solutions in settings where local improvements can be computed in polynomial time. In order to define the class PLS, we proceed as follows: first, we define a set of basic PLS problems, and then define the class PLS as the set of all TFNPproblems that reduce to a basic PLS problem.

A *local search problem* Π is defined as follows. For every instance[5] $I \in \{0,1\}^*$, there is a finite set $F_I \subseteq \{0,1\}^*$ of *feasible solutions*, an objective function $c_I : F_I \to \mathbb{N}$,

[5]A more general definition would also include a polynomial-time recognizable set $D_\Pi \subseteq \{0,1\}^*$ of valid instances. The assumption that $D_\Pi = \{0,1\}^*$ is essentially without loss of generality. Indeed, for

and for every feasible solution $s \in F_I$ there is a neighborhood $N_I(s) \subseteq F_I$. Given an instance I, one seeks a *local optimum* $s^* \in F_I$ with respect to c_I and N_I, meaning, in case of a maximization problem, that $c_I(s^*) \geq c_I(s)$ for all neighbors $s \in N_I(s^*)$.

Definition 6.4.1. A local search problem Π is a basic PLS problem if there exists some polynomial p such that $F_I \subseteq \{0,1\}^{p(|I|)}$ for all instances I, and if there exist polynomial-time algorithms A, B and C such that:

1. Given an instance I, algorithm A produces an initial feasible solution $s_0 \in F_I$.

2. Given an instance I and a string $s \in \{0,1\}^{p(|I|)}$, algorithm B determines whether s is a feasible solution and, if so, computes the objective value $c_I(s)$.

3. Given an instance I and any feasible solution $s \in F_I$, the algorithm C checks if s is locally optimal and, if not, produces a feasible solution $s' \in N_I(s)$ that improves the objective value.

Note that any basic PLS problem lies in TFNP.

Definition 6.4.2. The class PLS is defined as the set of all TFNPproblems that reduce to a basic PLS problem.

A problem is PLS-complete if it lies in PLS and if every problem in PLS reduces to it. Johnson et al. [147] showed that the so-called Flip problem is PLS-complete. We will define this problem later when we make use of it to prove our PLS-hardness result.

6.4.1 PLS-membership

Plaut and Roughgarden [185] prove the existence of an EFX allocation when all agents share the same monotone valuation, by introducing the leximin++ solution. In this section, we show how their existence proof can be translated into a proof of PLS-membership for the following problem.

Definition 6.4.3 (Identical-EFX)**.** An instance $I = (N, M, C)$ of the Identical-EFX search problem consists of a set of agents $N = [n]$, a set of goods $M = [m]$, and a boolean circuit C with m input gates. The circuit C defines a valuation function $v\colon 2^M \to \mathbb{N}$ which is the common valuation of all the agents. A solution is one of the following:

1. An allocation $(X_1, \ldots, X_n)$ that is EFX.

2. A pair of bundles $S \subseteq T$ that violate monotonicity, that is, $v(S) > v(T)$.

$I \notin D_\Pi$ we can define $F_I = \{0\}$, $c_I(0) = 1$ and $N_I(0) = \{0\}$. Note that this does not change the complexity of the problem.

The reason for allowing the violation-of-monotonicity solutions is that the circuit C is not guaranteed to define a monotone valuation, and in this case an EFX allocation is not guaranteed to exist. Importantly, we note that our PLS-hardness result (presented in the next section) does not rely on violation solutions. In other words, even the version of the problem where we are promised that the valuation function is monotone remains PLS-hard.

Theorem 6.4.1. *The* IDENTICAL-EFX *problem lies in* PLS.

The problem of computing an EFX allocation for two non-identical agents with valuations v_1 and v_2 is reducible to the problem of computing an EFX allocation for two identical agents via the cut-and-choose protocol. As a result, we immediately also obtain the following:

Corollary 9. *Computing an EFX allocation for two not necessarily identical agents is in* PLS.

Proof. To show that the IDENTICAL-EFX problem is in PLS, we reduce it to a basic PLS problem. An instance of this basic PLS problem is just an instance of the IDENTICAL-EFX problem, i.e, a tuple $I = (N, M, C)$. The set of feasible solutions F_I is the set of all possible allocations of the goods in M to the agents in N. As an initial feasible solution, we simply take the allocation where one agent receives every good. It remains to specify the objective function c_I and the neighborhood structure N_I, and then to argue that a local optimum corresponds to an EFX allocation.

Plaut and Roughgarden [185, Section 4] introduce the *leximin*++ ordering on the set of allocations, and show that the maximum element with respect to that ordering must be an EFX allocation. In fact, a closer inspection of their proof reveals that even a *local* maximum with respect to the leximin++ ordering must be an EFX allocation. As a result, we construct an objective function that implements the leximin++ ordering and then use the same arguments as Plaut and Roughgarden [185, Theorem 4.2].

For an allocation $(X_1, \dots, X_n)$, we let $O^X = (i_1, \dots, i_n)$ be an ordering of the agents according to increasing values of $v(X_i)$ (if multiple agents have bundles of equal utility, we break ties by ordering tied agents in terms of their agent number, i.e., if agents i and j are tied, and $i < j$, then agent i will appear before agent j in the ordering). The objective value is then defined as

$$\begin{aligned} c_I(X) = &|X_{i_n}| + v(X_{i_n})K \\ &+ |X_{i_{n-1}}|K^2 + v(X_{i_{n-1}})K^3 \\ &+ \dots \\ &+ |X_{i_1}|K^{2n-2} + v(X_{i_1})K^{2n-1} \end{aligned}$$

where K is an upper bound on the size or utility of any bundle. We claim that if an allocation X is not EFX, then one may construct an allocation X' from X by moving a single good from one bundle to another such that the objective strictly increases, $c_I(X') > c_I(X)$. Therefore, we will consider local maximization of this objective and we define the neighborhood of X to be $N_I(X) = \{X' \in F_I \colon \exists i, j \in N, \exists g \in X_j \text{ s.t. } X'_i = X_i \cup \{g\}, X'_j = X_j \setminus \{g\}, X'_k = X_k \text{ for } k \neq i, j\}$. We note that the cardinality of $N_I(X)$ is polynomial in n and m, so the algorithm for finding an improving neighbor if one exists may simply compute the objective value for every allocation in the neighborhood. Thus, this local maximization problem is indeed a basic PLS problem.

Finally, we have to show that any local maximum $X \in F_I$ yields a solution to the Identical-EFX problem, i.e., X is an EFX allocation or X yields a violation of monotonicity. We say that an allocation X yields a violation of monotonicity, if there exist $i \in N$ and $g \in M$ such that $v(X_i \setminus \{g\}) > v(X_i)$ or $v(X_i \cup \{g\}) < v(X_i)$. We note that if X yields a violation of monotonicity, then the violation can be found in polynomial time.

Consider an allocation $X \in F_I$ that is not EFX and that does not yield a violation of monotonicity. We will show that X cannot be a local maximum, which then implies the desired statement by contrapositive. Since X is not EFX, we may find $i, j \in N$ and $g \in X_j$ such that $v(X_i) < v(X_j \setminus \{g\})$. Without loss of generality, we may assume that $i = \arg\min_{k \in N} v(X_k)$, and if more than one agent attains this minimum, then we take the i that appears last among those tied agents in O^X according to the tie-breaking. Now define an allocation X' by

$$\begin{aligned} X'_i &= X_i \cup \{g\} \\ X'_j &= X_j \setminus \{g\} \\ X'_k &= X_k \text{ for } k \neq i, j \end{aligned}$$

and note that $X' \in N_I(X)$. We claim that $c_I(X') > c_I(X)$, meaning that X is not a local maximum.

In order to prove this, we first show that the orderings O^X and $O^{X'}$ agree in their first ℓ positions, where $\ell \in \{0, 1, \ldots, n-1\}$ is the index such that $O^X_{\ell+1} = i$, i.e., agent i appears in position $\ell + 1$ in O^X. Let S denote the set of agents that appear in O^X before agent i, i.e., the first ℓ agents appearing in O^X. Note that S consists of all the agents that have utility $v(X_i)$ in allocation X, excluding i. First, observe that $j \notin S$, because $v(X_j) \geq v(X_j \setminus \{g\}) > v(X_i)$ as X does not yield a violation of monotonicity. Therefore, we find that the bundles of the agents in S are not changed from allocation X to X', and, in particular, these agents still have utility $v(X_i)$ in allocation X'. Furthermore, in allocation X', all other agents (except possibly i) have strictly larger utility than S-agents, namely $v(X'_j) = v(X_j \setminus \{g\}) > v(X_i)$, and $v(X'_k) = v(X_k) > v(X_i)$ for $k \notin S \cup \{i, j\}$. Finally, $v(X'_i) = v(X_i \cup \{g\}) \geq v(X_i)$ as X does not yield a violation of monotonicity, and thus, in allocation X', agent i is either also tied with the agents in S, or it has strictly larger utility. In any case, by the tie-breaking, the first ℓ positions of O^X and $O^{X'}$ are the same.

We now argue that $c_I(X') > c_I(X)$. Since O^X and $O^{X'}$ agree in their first ℓ positions, and the bundles of those first ℓ agents have not changed, the 2ℓ highest-order terms in $c_I(X)$ and $c_I(X')$ have identical coefficients. By definition, $O^X_{\ell+1} = i$. If $O^{X'}_{\ell+1} = i$, then we have that $c_I(X') > c_I(X)$, because $v(X'_i) = v(X_i \cup \{g\}) \geq v(X_i)$ and $|X'_i| = |X_i \cup \{g\}| > |X_i|$, meaning that the coefficient in front of $K^{2n-(2\ell+1)}$ is at least as large in $c_I(X')$ as in $c_I(X)$ and the coefficient in front of $K^{2n-(2\ell+2)}$ is strictly larger. If $O^{X'}_{\ell+1} = k \neq i$, then we have that $c_I(X') > c_I(X)$, because $v(X'_k) > v(X_i)$, implying that the coefficient in front of $K^{2n-(2\ell+1)}$ is strictly larger in $c_I(X')$ than in $c_I(X)$. We conclude that X is not a local maximum. Therefore, by contraposition, a local maximum is an EFX allocation or it yields a violation of monotonicity. □

6.4.2 PLS-hardness

In this section we prove the following theorem.

Theorem 6.4.2. *The problem of computing an EFX allocation for two identical agents with a submodular valuation function is* PLS*-hard.*

The reduction consists of two steps. First, following Plaut and Roughgarden [185], we reduce the problem of local optimization on an odd Kneser graph to the problem of computing an EFX allocation for two agents sharing the same submodular valuation function. Then, in the second step, which is also our main technical contribution, we show that the PLS-complete problem Flip reduces to local optimization on an odd Kneser graph.

Kneser ≤ Identical-EFX

For $k \in \mathbb{N}$, the odd Kneser graph $K(2k+1,k)$ is defined as follows: the vertex set consists of all subsets of $[2k+1]$ of size k, and there is an edge between two vertices if the corresponding sets are disjoint. We identify the vertex set of $K(2k+1,k)$ with the set $\{x \in \{0,1\}^{2k+1} : \|x\|_1 = k\}$, where $\|x\|_1 = \sum_{i=1}^{2k+1} x_i$ denotes the 1-norm. Note that there is an edge between x and x' if and only if $\langle x, x' \rangle = 0$, where $\langle \cdot, \cdot \rangle$ denotes the inner product.

Definition 6.4.4 (Kneser)**.** The Kneser problem of local optimization on an odd Kneser graph is defined as the following basic PLS problem. An instance of the Kneser problem consists of a boolean circuit C with $2k+1$ input nodes for some $k \in \mathbb{N}$. The set of feasible solutions is $F_C = \{x \in \{0,1\}^{2k+1} : \|x\|_1 = k\}$, and the neighborhood of some $x \in F_C$ is given by $N_C(x) = \{x' \in F_C : \langle x, x' \rangle = 0\}$. The goal is to find a solution that is a local maximum with respect to the objective function $C(x) = \sum_{i=0}^{m-1} y_i \cdot 2^i$, where $y_0, \ldots, y_{m-1}$ denote the output nodes of the circuit C.

Lemma 41. Kneser *reduces to* Identical-EFX *with two identical submodular agents.*

Proof. Our proof of this lemma closely follows the corresponding proof of Plaut and Roughgarden [185, Theorem 3.1], with some minor modifications due to the different computational model. First, we describe the map f taking instances C of Kneser to instances of Identical-EFX. We consider a valuation on subsets of $[2k+1]$ given by

$$v(X) = \begin{cases} 2|X| & \text{if } |X| < k \\ 2k - 2^{-C(X)} & \text{if } |X| = k \\ 2k & \text{if } |X| > k \end{cases}$$

Using the description of the circuit C, we may in polynomial time construct a boolean circuit computing v. This valuation may take non-integer values, but this can be fixed by scaling by a larger power of 2. Scaling will not change anything in the arguments below. We now define $f(C) = ([2], [2k+1], v)$. That is, the KNESER instance C is mapped to an IDENTICAL-EFX instance with $2k+1$ goods and with two agents sharing the same valuation v.

We note that $2^{-C(X)} \in (0,1]$, because C takes values in the natural numbers. This ensures that the valuation v is monotone, because $v(S)$ is seen to be non-decreasing in $|S|$. Therefore, the only optimal solutions of $f(C)$ are EFX allocations (X_1, X_2). Note by inspection of v that if (X_1, X_2) is EFX, then $|X_1| = k$ and $|X_2| = k+1$ (or $|X_1| = k+1$ and $|X_2| = k$). If we are in the first case then X_1 corresponds to a feasible solution of the KNESER instance C. Also any neighbor of X_1 in the Kneser graph is of the form $X_2 \setminus \{g\}$ for some $g \in X_2$. As (X_1, X_2) is EFX we have that

$$\begin{aligned} 2k - 2^{-C(X_1)} &= v(X_1) \\ &\geq v(X_2 \setminus \{g\}) = 2k - 2^{-C(X_2 \setminus \{g\})} \end{aligned}$$

implying that $C(X_1) \geq C(X_2 \setminus \{g\})$ for all $g \in X_2$. We conclude that X_1 is a local maximum for the instance of KNESER given by the circuit C. Similarly, when $|X_2| = k$, X_2 will be a local maximum. As a result, we can define the polynomial-time map g that maps solutions of the IDENTICAL-EFX instance to solutions of the KNESER-instance by

$$g((X_1, X_2), C) = \begin{cases} X_1 & \text{if } |X_1| = k \\ X_2 & \text{otherwise} \end{cases}$$

By the discussion above it follows that if (X_1, X_2) is a solution to the IDENTICAL-EFX instance, then $g((X_1, X_2), C)$ is an optimal solution to the KNESER-instance. Therefore, the pair (f, g) constitutes a reduction from KNESER to IDENTICAL-EFX.

Finally, we show that v is submodular. For any $X \subseteq [2k+1]$ and $x \notin X$ we have that

$$v(X \cup \{x\}) - v(X) = \begin{cases} 2 & \text{if } |X| < k-1 \\ 2 - 2^{-C(X \cup \{x\})} & \text{if } |X| = k-1 \\ 2^{-C(X)} & \text{if } |X| = k \\ 0 & \text{if } |X| > k \end{cases}$$

Using that $2^{-C(X)} \in (0,1]$, this shows that $v(X \cup \{x\}) - v(X)$ is non-increasing in $|X|$. Thus, if $Y \subseteq X$ and $x \notin X$, we have that $v(X \cup \{x\}) - v(X) \leq v(Y \cup \{x\}) - v(Y)$, meaning that v is submodular. □

FLIP ≤ KNESER

Johnson et al. [147] introduced the computational problem FLIP and proved that it is PLS-complete. We will now reduce from FLIP to KNESER to show that KNESER, and thus Identical-EFX, are PLS-hard. In particular, this also establishes the PLS-completeness of Kneser, which might be of independent interest.

Definition 6.4.5 (Flip)**.** The Flip problem is the following basic PLS problem. The instances of Flip are boolean circuits. For an instance C with n input nodes $x_0, \dots, x_{n-1}$ and m output nodes $y_0, \dots, y_{m-1}$, the set of feasible solutions is all the possible inputs to the circuit: $F_C = \{0,1\}^n$. For any $x \in \{0,1\}^n$, the neighborhood is all the inputs that can be obtained from x by flipping one bit: $N_C(x) = \{x' \in \{0,1\}^n : \Delta(x,x') = 1\}$ where $\Delta(\cdot,\cdot)$ denotes the Hamming distance. The goal is to find a solution that is locally minimal with respect to the objective function defined by $C(x) = \sum_{i=0}^{m-1} y_i \cdot 2^i$.

Lemma 42. Flip *reduces to* Kneser.

Proof. We construct a reduction from Flip to the minimization version of Kneser. The minimization version of Kneser is seen to be equivalent to its maximization version by negating the output nodes of the original circuit. Let C_F be an instance of Flip. Denote by $p = \text{poly}(|C_F|)$ the length of the feasible solutions of C_F. The map of instances f now takes C_F to an instance C_K of the Kneser-problem whose feasible solutions are $F_K = \{x \in \{0,1\}^{2p+1} : \|x\|_1 = p\}$. A typical feasible solution will be written as $s = uvb$ where $u, v \in \{0,1\}^p$ and $b \in \{0,1\}$. We will use the notation $\overline{u}$ to denote the bitwise negation of $u \in \{0,1\}^p$. The circuit C_K is defined as follows:

1. $C_K(u\overline{u}0) = 2 \cdot C_F(u)$,
2. $C_K(uv1) = 2 \cdot \min(C_F(\overline{u}), C_F(v)) + 1$ if $\Delta(\overline{u}, v) = 1$,
3. $C_K(uvb) = M + \Delta(\overline{u}, v)$ otherwise.

Here M denotes a number chosen to be sufficiently large so that it dominates any cost $2 \cdot C_F(w)$. Note that the circuit C_K is well-defined and that it can be constructed in polynomial time given the circuit C_F. At a high level, the definition of the cost of a vertex of the third type is meant to ensure that for any such vertex uvb, there is a sequence of neighbors with decreasing costs that ends in a vertex of the form $u\overline{u}0$. The costs of the first and second vertex types are then meant to ensure that for a vertex $u\overline{u}0$, there is a sequence of neighbors with decreasing costs that ends in a vertex $w\overline{w}0$ where w is an improving neighbor of u in the original Flip-instance.

Below we show that the only local minima of C_K are of the form $u\overline{u}0$ where u is a local minimum for C_F. Therefore, upon defining the solution-mapping by $g(uvb) = u$ we have that (f, g) is a reduction from Flip to Kneser.

No optimal solutions of type (3). If a feasible solution $s = uvb$ is of type (3), then we claim that it must have a neighbor of lower cost. First of all, note that since s is not of type (1) or (2), and since $\|s\|_1 = p$, it follows that $\Delta(\overline{u}, v) \geq 2$. Now, because $\Delta(\overline{u}, v) \geq 2 > 0$ and $\|uv\|_1 \leq p$, there must exist an i such that $u_i = v_i = 0$. Otherwise one would find that $\|uv\|_1 > p$, which contradicts s being a feasible solution. Now, let $s' = u'v'b'$, where $u' = \overline{u}$, $b' = \overline{b}$, and $v'_j = \overline{v}_j$ for all $j \neq i$, but $v'_i = v_i = 0$. We note that $\|s'\|_1 = \|\overline{s}\|_1 - 1 = (p+1) - 1 = p$, so s' is a valid vertex in the Kneser graph. Further, we see that s' is a neighbor of s, because $s'_j s_j = 0$ for all j. If s' is not of type (3), then it has lower cost than s by construction of C_K and choice of M. Finally, if s' is of type (3), then the observation that $\Delta(\overline{u'}, v') < \Delta(\overline{u}, v)$ again yields that s' has lower cost than s.

No optimal solutions of type (2). Suppose $s = uv1$ is of type (2). As $\|s\|_1 = p$ and $\Delta(\overline{u}, v) = 1$, there is some i with $v_i = 0$ and $\overline{u}_i = 1$, and $v_j = \overline{u}_j$ for $j \neq i$. This implies that $\sum_i u_i v_i = 0$, and so both $s' = \overline{u}u0$ and $s'' = v\overline{v}0$ are neighbors of s. Furthermore, by construction of C_K, the cost of s' or of s'' is strictly less than the cost of s.

Optimal solutions. Consider a feasible solution of the form $u\overline{u}0$. If u is not a local minimum for C_F, then let w be an improving neighbor of u. As $\Delta(u, w) = 1$, there are now two cases to consider. If $u_i = 0$ and $w_i = 1$ for some i, then $s' = \overline{w}u1$ is a type (2) neighbor of lower cost. If $u_i = 1$ and $w_i = 0$ for some i, then $s' = \overline{u}w1$ is a type (2) neighbor of lower cost. Therefore, if $u\overline{u}0$ is a local minimum for C_K, then u is a local minimum for C_F. □

Corollary 10. *Let $n \geq 2$ be an integer. Computing an EFX allocation for n identical agents with a submodular valuation function is* PLS*-hard.*

Proof. By 6.4.2 it suffices to produce a reduction from the problem of computing an EFX allocation for two identical agents to the problem of computing an EFX allocation for n identical agents. We sketch this reduction. Let $u\colon 2^M \to \mathbb{R}$ denote the common submodular valuation function of the two agents. Construct an EFX-instance with n agents by adding $n-2$ agents and $n-2$ goods, $M' = M \cup \{g_1, \ldots, g_{n-2}\}$. Define the valuation function of the n agents to be $u' = \overline{u} + v$ where $\overline{u}\colon 2^{M'} \to \mathbb{R}$ is the extension of u given by $\overline{u}(S) = u(S \cap M)$ and where $v\colon 2^{M'} \to \mathbb{R}$ is additive given by $v(\{g_i\}) = u(M) + 1$ for $i = 1, \ldots, n-2$ and $v(\{g\}) = 0$ for $g \in M$. One may verify that $\overline{u}$ is submodular, and so that u' is the sum of two submodular valuations and therefore itself submodular.

Let $(X_1, \ldots, X_n)$ denote an EFX allocation of this instance. We claim that after permuting the bundles, we may assume that $X_{i+2} = \{g_i\}$ for $i = 1, \ldots, n-2$ and $X_1 \cup X_2 = M$. At least one bundle, say X_1, receives no good from $\{g_1, \ldots, g_{n-2}\}$, and so $u'(X_1) = u(X_1) \leq u(M)$. Now suppose some other bundle X_i contains some good g_j. If X_i contained another good g, then

$$u'(X_i \setminus \{g\}) \geq u'(\{g_j\}) = u(M) + 1 > u'(X_1),$$

contradicting $(X_1, \ldots, X_n)$ being EFX. Hence, $X_i = \{g_j\}$, and the claim follows. Now, one sees that (X_1, X_2) is an EFX allocation of the original two-agent instance. □

Bibliography

[1] Atila Abdulkadiroğlu and Tayfun Sönmez. Random serial dictatorship and the core from random endowments in house allocation problems. *Econometrica*, 66(3):689–701, 1998. doi: 10.2307/2998580. 84

[2] Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. On the impact of combinatorial structure on congestion games. *Journal of the ACM (JACM)*, 55 (6):1–22, 2008. 136

[3] Ron Aharoni, Eli Berger, Joseph Briggs, Erel Segal-Halevi, and Shira Zerbib. Fractionally balanced hypergraphs and rainbow KKM theorems. *arXiv preprint arXiv:2011.01053*, 2020. 34, 68

[4] James Aisenberg, Maria Luisa Bonet, and Sam Buss. 2-d tucker is PPA complete. *J. Comput. Syst. Sci.*, 108:92–103, 2020. doi: 10.1016/j.jcss.2019.09.002. 243

[5] Eric Allender, Peter Bürgisser, Johan Kjeldgaard-Pedersen, and Peter Bro Miltersen. On the complexity of numerical analysis. *SIAM J. Comput.*, 38 (5):1987–2006, 2009. doi: 10.1137/070697926. URL `https://doi.org/10.1137/070697926`. 238, 248

[6] Georgios Amanatidis, Georgios Birmpas, Aris Filos-Ratsikas, Alexandros Hollender, and Alexandros A. Voudouris. Maximum Nash welfare and other stories about EFX. *Theoretical Computer Science*, 863:69–85, 2021. doi: 10.1016/j.tcs.2021.02.020. 25, 276

[7] Georgios Amanatidis, Georgios Birmpas, Aris Filos-Ratsikas, and Alexandros A. Voudouris. Fair division of indivisible goods: A survey. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5385–5393, 2022. doi: 10.24963/ijcai.2022/756. 277

[8] Kenneth J. Arrow and Gerard Debreu. Existence of an equilibrium for a competitive economy. *Econometrica*, 22(3):265–290, 1954. doi: 10.2307/1907353. 3, 29, 34, 35, 37, 76, 77, 79, 80, 88, 91, 103, 111, 176

[9] Robert J Aumann. Subjectivity and correlation in randomized strategies. *Journal of mathematical Economics*, 1(1):67–96, 1974. 100, 142

[10] Haris Aziz and Simon Mackenzie. A discrete and bounded envy-free cake cutting protocol for any number of agents. In *Proceedings of the 57th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 416–427, 2016. doi: 10.1109/focs.2016.52. 274

[11] Haris Aziz and Simon Mackenzie. A discrete and bounded envy-free cake cutting protocol for any number of agents. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 416–427, 2016. doi: 10.1109/FOCS.2016.52. 36, 37

[12] Haris Aziz, Bo Li, Herve Moulin, and Xiaowei Wu. Algorithmic fair allocation of indivisible items: A survey and new questions. *arXiv:2202.08713*, 2022. 277

[13] Moshe Babaioff, Robert Kleinberg, and Christos H Papadimitriou. Congestion games with malicious players. *Games and Economic Behavior*, 1(67):22–35, 2009. 103, 149, 150, 155, 156

[14] Moshe Babaioff, Tomer Ezra, and Uriel Feige. Fair and truthful mechanisms for dichotomous valuations. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*, pages 5119–5126, 2021. URL `https://ojs.aaai.org/index.php/AAAI/article/view/16647`. 276

[15] Santiago Balseiro, Yuan Deng, Jieming Mao, Vahab Mirrokni, and Song Zuo. Robust auction design in the auto-bidding world. *Advances in Neural Information Processing Systems*, 34:17777–17788, 2021. 106

[16] Santiago Balseiro, Anthony Kim, Mohammad Mahdian, and Vahab Mirrokni. Budget-management strategies in repeated auctions. *Operations research*, 69 (3):859–876, 2021. doi: 10.1287/opre.2020.2073.

[17] Santiago R Balseiro and Yonatan Gur. Learning in repeated auctions with budgets: Regret minimization and equilibrium. *Management Science*, 65(9): 3952–3968, 2019.

[18] Santiago R Balseiro, Yuan Deng, Jieming Mao, Vahab S Mirrokni, and Song Zuo. The landscape of auto-bidding auctions: Value versus utility maximization. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 132–133, 2021. 106

[19] R. B. Bapat. A constructive proof of a permutation-based generalization of Sperner's lemma. *Mathematical Programming*, 44(1):113–120, 1989. doi: 10.1007/BF01587081. 34, 68, 70, 75

[20] Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding fair and efficient allocations. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)*, pages 557–574, 2018. doi: 10.1145/3219166.3219176. 277

[21] S. Basu, R. Pollack, and M. Roy. *Algorithms in Real Algebraic Geometry*. Springer, updated online version of second edition (2008) edition, 2016. https://perso.univ-rennes1.fr/marie-francoise.roy/bpr-ed2-posted3.html. 253, 254

[22] Eleni Batziou, Kristoffer Arnsfelt Hansen, and Kasper Høgh. Strong Approximate Consensus Halving and the Borsuk-Ulam Theorem. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-195-5. doi: 10.4230/LIPIcs.ICALP.2021.24. URL `https://drops.dagstuhl.de/opus/volltexte/2021/14093`. 4, 15

[23] Eleni Batziou, Kristoffer Arnsfelt Hansen, and Kasper Høgh. Strong approximate Consensus Halving and the Borsuk-Ulam theorem. In *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 24:1–24:20, 2021. doi: 10.4230/LIPIcs.ICALP.2021.24. 20, 38

[24] Xiaohui Bei, Ning Chen, Xia Hua, Biaoshuai Tao, and Endong Yang. Optimal proportional cake cutting with connected pieces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26(1), pages 1263–1269, 2012. 107, 227

[25] Claude Berge. *Topological Spaces: including a treatment of multi-valued functions, vector spaces, and convexity*. Courier Corporation, 1997. 39

[26] Ben Berger, Avi Cohen, Michal Feldman, and Amos Fiat. Almost full EFX exists for four agents. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, pages 4826–4833, 2022. doi: 10.1609/aaai.v36i5.20410. 276

[27] Marie Louisa Tølbøll Berthelsen and Kristoffer Arnsfelt Hansen. On the computational complexity of decision problems about multi-player Nash equilibria. In Dimitris Fotakis and Evangelos Markakis, editors, *SAGT 2019*, volume 11801 of *Lecture Notes in Computer Science*, pages 153–167. Springer, 2019. doi: 10.1007/978-3-030-30473-7_11. 248

[28] Vittorio Bilò and Marios Mavronicolas. A catalog of $\exists\mathbb{R}$-complete decision problems about Nash equilibria in multi-player games. In Nicolas Ollinger and Heribert Vollmer, editors, *STACS 2016*, volume 47 of *LIPIcs*, pages 17:1–17:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi: 10.4230/LIPIcs.STACS.2016.17.

[29] Vittorio Biló and Marios Mavronicolas. $\exists\mathbb{R}$-complete decision problems about symmetric Nash equilibria in symmetric multi-player games. In Heribert Vollmer and Brigitte Vallé, editors, *STACS 2017*, volume 66 of *LIPIcs*, pages

13:1–13:14. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2017. ISBN 978-3-95977-028-6. doi: 10.4230/LIPIcs.STACS.2017.13. 248

[30] Lenore Blum, M. Schub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Amer. Math. Soc.*, 21:1–46, 07 1989. doi: 10.1090/S0273-0979-1989-15750-9. 247

[31] Jacek Bochnak, Michel Coste, and Marie-Françoise Roy. *Real Algebraic Geometry*. Springer, 1998. doi: 10.1007/978-3-662-03718-8. 73

[32] Anna Bogomolnaia and Hervé Moulin. A new solution to the random assignment problem. *Journal of Economic Theory*, 100(2):295–328, 2001. doi: 10.1006/jeth.2000.2710. 35, 84

[33] Christian Borgs, Jennifer Chayes, Nicole Immorlica, Kamal Jain, Omid Etesami, and Mohammad Mahdian. Dynamics of bid optimization in online advertisement auctions. In *Proceedings of the 16th international conference on World Wide Web*, pages 531–540, 2007. 106, 212

[34] Karol Borsuk. Drei Sätze über die n-dimensionale euklidische Sphäre. *Fundamenta Mathematicae*, 20:177–190, 1933. doi: 10.4064/fm-20-1-177-190. 3, 15, 38

[35] Karol Borsuk. Drei sätze über die n-dimensionale euklidische sphäre. *Fundamenta Mathematicae*, 20(1):177–190, 1933. doi: 10.4064/fm-20-1-177-190. 238

[36] Sylvain Bouveret and Jérôme Lang. Efficiency and envy-freeness in fair division of indivisible goods: Logical representation and complexity. 32(1):525–564, jun 2008. ISSN 1076-9757. 21

[37] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2014. 127, 150

[38] Steven J. Brams and Alan D. Taylor. *Fair Division: From cake-cutting to dispute resolution*. Cambridge University Press, 1996. 30, 33, 36, 66, 67, 107, 226

[39] L. E. J. Brouwer. Über abbildung von mannigfaltigkeiten. *Mathematische Annalen*, 71:97–115, 1911. doi: 10.1007/BF01456931. 3, 10, 30, 38, 93, 144, 226, 238

[40] Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011. doi: 10.1086/664613. 21, 276

[41] John Canny. Some algebraic and geometric computations in PSPACE. In *STOC*, pages 460–467. ACM, 01 1988. doi: 10.1145/62212.62257. 248

[42] Ioannis Caragiannis, Nick Gravin, and Xin Huang. Envy-freeness up to any item with high Nash welfare: The virtue of donating items. In *Proceedings of the 20th ACM Conference on Economics and Computation (EC)*, pages 527–545, 2019. doi: 10.1145/3328526.3329574. 276

[43] Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D. Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum Nash welfare. *ACM Transactions on Economics and Computation*, 7(3):12:1–12:32, 2019. doi: 10.1145/3355902. 22, 274, 278

[44] Ioannis Caragiannis, Kristoffer Arnsfelt Hansen, and Nidhi Rathi. On the Complexity of Pareto-Optimal and Envy-Free Allocation Lotteries, 2023. Manuscript in preparation. 109, 225

[45] Bhaskar Ray Chaudhury, Jugal Garg, and Kurt Mehlhorn. EFX exists for three agents. In *Proceedings of the 21st ACM Conference on Economics and Computation (EC)*, pages 1–19, 2020. doi: 10.1145/3391403.3399511. 25, 274

[46] Bhaskar Ray Chaudhury, Jugal Garg, Kurt Mehlhorn, Ruta Mehta, and Pranabendu Misra. Improving EFX guarantees through rainbow cycle number. In *Proceedings of the 22nd ACM Conference on Economics and Computation (EC)*, pages 310–311, 2021. doi: 10.1145/3465456.3467605. 276

[47] Bhaskar Ray Chaudhury, Telikepalli Kavitha, Kurt Mehlhorn, and Alkmini Sgouritsa. A little charity guarantees almost envy-freeness. *SIAM Journal on Computing*, 50(4):1336–1358, 2021. doi: 10.1137/20m1359134. 276

[48] Bhaskar Ray Chaudhury, Jugal Garg, Peter McGlaughlin, and Ruta Mehta. A complementary pivot algorithm for competitive allocation of a mixed manna. *Mathematics of Operations Research*, 2022. 169, 236

[49] Thomas Chen, Xi Chen, Binghui Peng, and Mihalis Yannakakis. Computational hardness of the Hylland-Zeckhauser scheme. In *Proceedings of the 33rd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2253–2268, 2022. doi: 10.1137/1.9781611977073.90. 88

[50] Thomas Chen, Xi Chen, Binghui Peng, and Mihalis Yannakakis. Computational hardness of the Hylland-Zeckhauser scheme. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2253–2268. SIAM, 2022. 137

[51] Xi Chen and Xiaotie Deng. Settling the complexity of two-player Nash equilibrium. In *FOCS 2006*, pages 261–272. IEEE Computer Society Press, 2006. 243

[52] Xi Chen and Xiaotie Deng. On the complexity of 2d discrete fixed point problem. *Theoretical Computer Science*, 410(44):4448–4456, 2009. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2009.07.052. URL `https://www.sciencedirect.com/science/article/pii/S030439750900499X`. Automata, Languages and Programming (ICALP 2006). 10

[53] Xi Chen, Decheng Dai, Ye Du, and Shang-Hua Teng. Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *Proceedings of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 273–282, 2009. doi: 10.1109/FOCS.2009.29. 10, 37

[54] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM*, 56(3):14:1–14:57, 2009. doi: 10.1145/1516512.1516516. 30, 36

[55] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM (JACM)*, 56(3): 1–57, 2009. 10, 92, 106, 135

[56] Xi Chen, Dimitris Paparas, and Mihalis Yannakakis. The complexity of non-monotone markets. *Journal of the ACM*, 64(3):20:1–20:56, 2017. doi: 10.1145/3064810. 10, 30, 35, 37, 79, 80, 92

[57] Xi Chen, Christian Kroer, and Rachitesh Kumar. The complexity of pacing for second-price auctions. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 318–318, 2021. 92, 97, 98, 106, 212, 213, 214, 215, 216

[58] Xi Chen, Christian Kroer, and Rachitesh Kumar. Throttling equilibria in auction markets. In *WINE*, 2021. URL `https://arxiv.org/abs/2107.10923`. 137

[59] Bruno Codenotti, Amin Saberi, Kasturi Varadarajan, and Yinyu Ye. The complexity of equilibria: Hardness results for economies via a correspondence with games. *Theoretical Computer Science*, 408(2–3):188–198, 2008. doi: 10.1016/j.tcs.2008.08.007. 10, 37

[60] Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V Vazirani, and Sadra Yazdanbod. Convex program duality, Fisher markets, and Nash social welfare. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 459–460, 2017. 212

[61] Vincent Conitzer, Christian Kroer, Debmalya Panigrahi, Okke Schrijvers, Nicolas E Stier-Moses, Eric Sodomka, and Christopher A Wilkens. Pacing equilibrium in first price auction markets. *Management Science*, 68(12):8515–8535, 2022. 106, 212, 213, 216

[62] Vincent Conitzer, Christian Kroer, Eric Sodomka, and Nicolas E Stier-Moses. Multiplicative pacing equilibria in auction markets. *Operations Research*, 70 (2):963–989, 2022. 106, 107, 212, 213, 214, 215, 216, 221

[63] Richard W Cottle, Jong-Shi Pang, and Richard E Stone. *The linear complementarity problem.* SIAM, 2009. 96, 135

[64] RW Cottle and GB Dantzig. Complementarity pivot theory of mathematical programming, linear algeb. In *Appl*, volume 1, pages 163–185, 1968. 96, 98, 99, 135

[65] Partha Sarathi Dasgupta and Eric S Maskin. Debreu's social equilibrium existence theorem. *Proceedings of the National Academy of Sciences*, 112(52): 15769–15770, 2015. 101, 140

[66] Constantinos Daskalakis, Alex Fabrikant, and Christos H Papadimitriou. The game world is flat: The complexity of nash equilibria in succinct games. In *ICALP (1)*, pages 513–524, 2006. 100, 142

[67] Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009. 10, 15, 30, 36, 92, 243

[68] Gerard Debreu. A social equilibrium existence theorem. *Proceedings of the National Academy of Sciences*, 38(10):886–893, 1952. doi: 10.1073/pnas.38.10.886. 3, 37, 79, 100, 101, 102, 103, 107, 139, 140, 144, 145, 146, 150, 151, 154, 156, 215, 221

[69] Argyrios Deligkas, John Fearnley, and Rahul Savani. Inapproximability results for approximate Nash equilibria. In *Proceedings of the 12th International Conference on Web and Internet Economics (WINE)*, pages 29–43, 2016. doi: 10.1007/978-3-662-54110-4_3. 36

[70] Argyrios Deligkas, John Fearnley, Rahul Savani, and Paul Spirakis. Computing approximate Nash equilibria in polymatrix games. *Algorithmica*, 77:487–514, 2017. doi: 10.1007/s00453-015-0078-7. 36

[71] Argyrios Deligkas, John Fearnley, Themistoklis Melissourgos, and Paul G. Spirakis. Computing exact solutions of consensus halving and the Borsuk-Ulam theorem. In *ICALP*, volume 132 of *LIPIcs*, pages 138:1–138:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi: 10.4230/LIPIcs.ICALP.2019.138. 16, 17, 252

[72] Argyrios Deligkas, John Fearnley, Themistoklis Melissourgos, and Paul G. Spirakis. Computing exact solutions of consensus halving and the Borsuk-Ulam theorem. *Journal of Computer and System Sciences*, 117:75–98, 2021. doi: https://doi.org/10.1016/j.jcss.2020.10.006. 20, 239, 240, 241, 244, 245, 252, 255, 257

[73] Argyrios Deligkas, John Fearnley, Themistoklis Melissourgos, and Paul G. Spirakis. Computing exact solutions of consensus halving and the Borsuk-Ulam theorem. *Journal of Computer and System Sciences*, 117:75–98, 2021. doi: 10.1016/j.jcss.2020.10.006. 112

[74] Argyrios Deligkas, John Fearnley, Themistoklis Melissourgos, and Paul G. Spirakis. Computing exact solutions of consensus halving and the Borsuk-Ulam theorem. *J. Comput. Syst. Sci.*, 117:75–98, 2021. doi: 10.1016/j.jcss.2020.10.006. 38

[75] Argyrios Deligkas, John Fearnley, and Themistoklis Melissourgos. Pizza sharing is ppa-hard. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 4957–4965. AAAI Press, 2022. URL `https://ojs.aaai.org/index.php/AAAI/article/view/20426`. 20

[76] Xiaotie Deng, Qi Qi, and Amin Saberi. Algorithmic solutions for envy-free cake cutting. *Operations Research*, 60(6):1461–1476, 2012. doi: 10.1287/opre.1120.1116. 10, 30, 33, 36, 67, 108, 225, 227

[77] Nikhil R. Devanur, Christos H. Papadimitriou, Amin Saberi, and Vijay V. Vazirani. Market equilibrium via a primal-dual algorithm for a convex program. *Journal of the ACM*, 55(5):22:1–22:18, 2008. doi: 10.1145/1411509.1411512. 37

[78] Ran Duan and Kurt Mehlhorn. A combinatorial polynomial algorithm for the linear Arrow-Debreu market. *Information and Computation*, 243:112–132, 2015. doi: 10.1016/j.ic.2014.12.009.

[79] Ran Duan, Jugal Garg, and Kurt Mehlhorn. An improved combinatorial polynomial algorithm for the linear Arrow-Debreu market. In *Proceedings of the 27th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 90–106, 2016. doi: 10.1137/1.9781611974331.ch7. 37

[80] B Curtis Eaves. A finite algorithm for the linear exchange model. *Journal of Mathematical Economics*, 3(2):197–203, 1976. 92, 96, 98, 103, 104, 105, 164, 165, 167, 169, 176

[81] Federico Echenique, Antonio Miralles, and Jun Zhang. Constrained pseudo-market equilibrium. *American Economic Review*, 111(11):3699–3732, 2021. doi: 10.1257/aer.20201769. 88

[82] Jack Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971. doi: 10.1007/bf01584082. 279

[83] Kousha Etessami. The complexity of computing a (quasi-)perfect equilibrium for an n-player extensive form game. *Games and Economic Behavior*, 2020. doi: 10.1016/j.geb.2019.03.006. In Press, Journal Pre-proof. 253

[84] Kousha Etessami. The complexity of computing a (quasi-)perfect equilibrium for an n-player extensive form game. *Games and Economic Behavior*, 125: 107–140, 2021. doi: 10.1016/j.geb.2019.03.006. 32, 37, 60

[85] Kousha Etessami and Mihalis Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM J. Comput.*, 39(6):2531–2597, 2010. 5, 12, 30, 32, 33, 34, 36, 37, 40, 41, 58, 59, 60, 64, 71, 79, 238, 239, 242, 249, 250, 251, 255, 257

[86] Kousha Etessami and Mihalis Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM Journal on Computing*, 39(6):2531–2597, 2010. 11, 92, 93, 97, 106, 109, 112, 113, 114, 135, 215, 222

[87] Kousha Etessami, Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, and Troels Bjerre Sørensen. The complexity of approximating a trembling hand perfect equilibrium of a multi-player game in strategic form. In *Proceedings of the 7th International Symposium on Algorithmic Game Theory (SAGT)*, pages 231–243, 2014. doi: 10.1007/978-3-662-44803-8_20. 37, 60

[88] Kousha Etessami, Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, and Troels Bjerre Sørensen. The complexity of approximating a trembling hand perfect equilibrium of a multi-player game in strategic form. In Ron Lavi, editor, *SAGT 2014*, volume 8768 of *LNCS*, pages 231–243. Springer, 2014. doi: 10.1007/978-3-662-44803-8_20. 253

[89] Kousha Etessami, Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, and Troels Bjerre Sørensen. The complexity of approximating a trembling hand perfect equilibrium of a multi-player game in strategic form. In *Algorithmic Game Theory: 7th International Symposium, SAGT 2014, Haifa, Israel, September 30–October 2, 2014. Proceedings 7*, pages 231–243. Springer, 2014. 149

[90] Ky Fan. Fixed-point and minimax theorems in locally convex topological linear spaces. *Proceedings of the National Academy of Sciences*, 38(2):121–126, 1952. doi: 10.1073/pnas.38.2.121. 65

[91] Ky Fan. Fixed-point and minimax theorems in locally convex topological linear spaces. *Proceedings of the National Academy of Sciences*, 38(2):121–126, 1952. 100, 102, 107, 140, 144, 145, 146, 150, 151, 154, 156, 215, 221

[92] John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: CLS = PPAD $\cap$ PLS. *Journal of the ACM*, 70 (1):1–74, 2022. doi: 10.1145/3568163. 112

[93] John Fearnley, Paul W. Goldberg, Alexandros Hollender, and Rahul Savani. The Complexity of Computing KKT Solutions of Quadratic Programs, 2023. Personal communication. 127

[94] Aris Filos-Ratsikas and Paul W. Goldberg. Consensus halving is PPA-complete. In *STOC*, pages 51–64. ACM, 2018. doi: 10.1145/3188745.3188880. 16, 243

[95] Aris Filos-Ratsikas and Paul W. Goldberg. The complexity of splitting necklaces and bisecting ham sandwiches. In *STOC*, pages 638–649. ACM, 2019. doi: 10.1145/3313276.3316334. 16, 20, 243, 244

[96] Aris Filos-Ratsikas, Kristoffer A. Hansen, Kasper Høgh, and Alexandros Hollender. Ppad-membership for problems with exact rational solutions: A general approach via convex optimization. *Manuscript*. 10

[97] Aris Filos-Ratsikas, Kristoffer A. Hansen, Kasper Høgh, and Alexandros Hollender. Fixp-membership via convex optimization: Games, cakes, and markets. *SIAM Journal on Computing*, 0(0):FOCS21–30–FOCS21–84, 0. doi: 10.1137/22M1472656. URL `https://doi.org/10.1137/22M1472656`. 4, 10

[98] Aris Filos-Ratsikas, Søren Kristoffer Stiil Frederiksen, Paul W. Goldberg, and Jie Zhang. Hardness results for consensus-halving. In *MFCS*, volume 117 of *LIPIcs*, pages 24:1–24:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi: 10.4230/LIPIcs.MFCS.2018.24. 16, 244

[99] Aris Filos-Ratsikas, Alexandros Hollender, Katerina Sotiraki, and Manolis Zampetakis. Consensus-halving: Does it ever get easier? In *EC*, pages 381–399. ACM, 2020. doi: 10.1145/3391403.3399527. 18, 244, 245, 265

[100] Aris Filos-Ratsikas, Yiannis Giannakopoulos, Alexandros Hollender, Philip Lazos, and Diogo Poças. On the complexity of equilibrium computation in first-price auctions. In *Proceedings of the 22nd ACM Conference on Economics and Computation (EC)*, pages 454–476, 2021. doi: 10.1145/3465456.3467627. 37

[101] Aris Filos-Ratsikas, Kristoffer Arnsfelt Hansen, Kasper Høgh, and Alexandros Hollender. FIXP-membership via convex optimization: Games, cakes, and markets. *SIAM J. Comput.*, Special Section FOCS 2021, 2023. doi: 10.1137/22M1472656. 93, 101, 107, 108, 109, 110, 112, 113, 114, 140, 149, 157, 161, 163, 212, 222, 223, 225, 227, 228, 229, 230, 235, 236

[102] A. M. Fink. Equilibrium in a stochastic n-person game. *J. Sci. Hiroshima Univ. Ser. A-I Math.*, 28(1):89–93, 1964. doi: 10.32917/hmj/1206139508. 32, 64, 65

[103] Duncan K. Foley. *Resource Allocation and the Public Sector*. PhD thesis, Yale University, 1966. 274

[104] Duncan Karl Foley. *Resource allocation and the public sector.* Yale University, 1966. 85

[105] D. Gale. Equilibrium in a discrete exchange economy with money. *International Journal of Game Theory*, 13:61–64, 1984. doi: 10.1007/BF01769865. 34, 68, 74, 75

[106] David Gale. The law of supply and demand. *Mathematica Scandinavica*, 3: 155–169, 1955. doi: 10.7146/math.scand.a-10436. 72, 74

[107] David Gale. Optimal assignments in an ordered set: An application of matroid theory. *Journal of Combinatorial Theory*, 4(2):176–180, 1968. doi: 10.1016/s0021-9800(68)80039-0. 279

[108] George Gamow and Marvin Stern. *Puzzle-math.* Viking Press, 1958. 33, 36, 66, 107, 225, 274

[109] Martin Gardner. *aha! Insight.* Scientific American, Inc./W. H. Freeman and Company, 1978. 225

[110] Jugal Garg. Market equilibrium under piecewise Leontief concave utilities. *Theoretical Computer Science*, 703:55–65, 2017. 92

[111] Jugal Garg and Aniket Murhekar. Computing fair and efficient allocations with few utility values. In Ioannis Caragiannis and Kristoffer Arnsfelt Hansen, editors, *Algorithmic Game Theory*, pages 345–359, Cham, 2021. Springer International Publishing. ISBN 978-3-030-85947-3. 25

[112] Jugal Garg and Vijay V. Vazirani. On computability of equilibria in markets with production. In *SODA*, pages 1329–1340. SIAM, 2014. 92, 96, 98, 104, 163, 164, 169, 175, 176, 177, 178, 179, 192, 203, 206, 207

[113] Jugal Garg and Vijay V. Vazirani. On computability of equilibria in markets with production. *Proceedings of the 25th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1329–1340, 2014. doi: 10.1137/1.9781611973402.98. 37

[114] Jugal Garg and László A. Végh. A strongly polynomial algorithm for linear exchange markets. In *Proceedings of the 51st ACM Symposium on Theory of Computing (STOC)*, pages 54–65, 2019. doi: 10.1145/3313276.3316340. 37

[115] Jugal Garg, Albert Xin Jiang, and Ruta Mehta. Bilinear games: Polynomial time algorithms for rank based subclasses. In *Internet and Network Economics: 7th International Workshop, WINE 2011, Singapore, December 11-14, 2011. Proceedings 7*, pages 399–407. Springer, 2011. 100, 144

[116] Jugal Garg, Ruta Mehta, Milind Sohoni, and Vijay V. Vazirani. A complementary pivot algorithm for market equilibrium under separable, piecewise-linear concave utilities. *SIAM Journal on Computing*, 44(6):1820–1847, 2015. 37, 98, 104, 163, 164, 203

[117] Jugal Garg, Ruta Mehta, and Vijay V. Vazirani. Dichotomies in equilibrium computation and membership of PLC markets in FIXP. *Theory of Computing*, 12(20):1–25, 2016. doi: 10.4086/toc.2016.v012a020. 34, 35, 37, 75, 79, 80, 81

[118] Jugal Garg, Ruta Mehta, Vijay V. Vazirani, and Sadra Yazdanbod. Settling the complexity of Leontief and PLC exchange markets under exact and approximate equilibria. In *Proceedings of the 49th ACM Symposium on Theory of Computing (STOC)*, pages 890–901, 2017. doi: 10.1145/3055399.3055474. 10, 30, 37, 88

[119] Jugal Garg, Ruta Mehta, and Vijay V Vazirani. Substitution with satiation: A new class of utility functions and a complementary pivot algorithm. *Mathematics of Operations Research*, 43(3):996–1024, 2018. 10, 92, 96, 98, 104, 163, 164, 165, 169, 177, 187, 188, 189, 192, 193

[120] Jugal Garg, Ruta Mehta, Vijay V. Vazirani, and Sadra Yazdanbod. $\exists\mathbb{R}$-completeness for decision versions of multi-player (symmetric) Nash equilibria. *ACM Trans. Econ. Comput.*, 6(1):1:1–1:23, 2018. ISSN 2167-8375. doi: 10.1145/3175494. 248

[121] Rahul Garg, Sanjiv Kapoor, and Vijay Vazirani. An auction-based market equilibrium algorithm for the separable gross substitutability case. In *Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX) and 8th International Workshop on Randomization and Computation (RANDOM)*, pages 128–138, 2004. doi: 10.1007/978-3-540-27821-4_12. 37

[122] John Geanakoplos. Nash and Walras equilibrium via Brouwer. *Economic Theory*, 21:585–603, 2003. doi: 10.1007/s001990000076. 37, 79

[123] I. L. Glicksberg. A further generalization of the Kakutani fixed point theorem, with application to Nash equilibrium points. *Proceedings of the American Mathematical Society*, 3(1):170–174, 1952. doi: 10.2307/2032478. 65

[124] Irving L Glicksberg. A further generalization of the Kakutani fixed theorem, with application to nash equilibrium points. *Proceedings of the American Mathematical Society*, 3(1):170–174, 1952. 100, 102, 107, 140, 144, 145, 146, 150, 151, 154, 156, 215, 221

[125] Michel X Goemans. Smallest compact formulation for the permutahedron. *Mathematical Programming*, 153(1):5–11, 2015. 102, 147

[126] Paul Goldberg, Alexandros Hollender, and Warut Suksompong. Contiguous cake cutting: Hardness results and approximation algorithms. *Journal of Artificial Intelligence Research*, 69:109–141, 2020. 98, 107, 108, 227

[127] Paul W. Goldberg. A survey of PPAD-completeness for computing Nash equilibria. In Robin Chapman, editor, *Surveys in Combinatorics 2011*, London Mathematical Society Lecture Note Series, pages 51–82. Cambridge University Press, 2011. doi: 10.1017/CBO9781139004114.003. 30

[128] Paul W. Goldberg and Alexandros Hollender. The Hairy Ball problem is PPAD-complete. *Journal of Computer and System Sciences*, 122:34–62, 2021. doi: 10.1016/j.jcss.2021.05.004. 38

[129] Paul W. Goldberg and Christos H. Papadimitriou. Towards a unified complexity theory of total functions. *Journal of Computer and System Sciences*, 94:167 – 192, 2018. doi: 10.1016/j.jcss.2017.12.003. 250

[130] Paul W. Goldberg, Kasper Høgh, and Alexandros Hollender. The frontier of intractability for efx with two agents, 2023. 4, 20

[131] Laurent Gourvès, Jérôme Monnot, and Lydia Tlilane. Near fairness in matroids. In *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI)*, pages 393–398, 2014. doi: 10.3233/978-1-61499-419-0-393. 22, 274, 278

[132] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988. 238

[133] Faruk Gul and Ennio Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87(1):95–124, 1999. URL `https://EconPapers.repec.org/RePEc:eee:jetheo:v:87:y:1999:i:1:p:95-124`. 21, 277

[134] P. Hall. On representatives of subsets. *Journal of the London Mathematical Society*, s1-10(1):26–30, 1935. doi: 10.1112/jlms/s1-10.37.26. 15, 69, 229

[135] Kristoffer Arnsfelt Hansen and Troels Bjerre Lund. Computational complexity of proper equilibrium. In *Proceedings of the 2018 ACM Conference on Economics and Computation (EC)*, pages 113–130, 2018. doi: 10.1145/3219166.3219199. 32, 36, 60, 61

[136] Kristoffer Arnsfelt Hansen and Troels Bjerre Lund. Computational complexity of proper equilibrium. In *EC 2018*, pages 113–130, New York, NY, USA, 2018. ACM. doi: 10.1145/3219166.3219199. 253

[137] Kristoffer Arnsfelt Hansen and Troels Bjerre Lund. Computational complexity of proper equilibrium. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 113–130, 2018. 92, 96, 98, 102, 148, 149

[138] Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, and Troels Bjerre Sørensen. The computational complexity of trembling hand perfection and other equilibrium refinements. In *Proceedings of the 3rd International Symposium on Algorithmic Game Theory (SAGT)*, pages 198–209, 2010. doi: 10.1007/978-3-642-16170-4_18. 36

[139] Kristoffer Arnsfelt Hansen, Michal Koucky, Niels Lauritzen, Peter Bro Miltersen, and Elias P. Tsigaridas. Exact algorithms for solving stochastic games. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 205–214, 2011. doi: 10.1145/1993636.1993665. 36

[140] Yinghua He, Antonio Miralles, Marek Pycia, and Jianye Yan. A pseudo-market approach to allocation with priorities. *American Economic Journal: Microeconomics*, 10(3):272–314, 2018. doi: 10.1257/mic.20150259. 88

[141] Joseph T Howson Jr. Equilibria of polymatrix games. *Management Science*, 18(5-part-1):312–318, 1972. 92, 96, 98, 100, 141

[142] Aanund Hylland and Richard Zeckhauser. The efficient allocation of individuals to positions. *Journal of Political Economy*, 87(2):293–314, 1979. doi: 10.1086/260757. 34, 35, 75, 84, 85, 88

[143] Kamal Jain. A polynomial time algorithm for computing an arrow–debreu market equilibrium for linear utilities. *SIAM Journal on Computing*, 37(1):303–318, 2007. 37

[144] Kamal Jain, Mohammad Mahdian, and Amin Saberi. Approximating market equilibria. In *Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX) and 7th International Workshop on Randomization and Computation (RANDOM)*, pages 98–108, 2003. doi: 10.1007/978-3-540-45198-3_9. 37

[145] Elena Janovskaja. Equilibrium points in polymatrix games. *Lithuanian Mathematical Journal*, 8(2):381–384, 1968. 100, 141

[146] Fritz John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays, Courant Anniversary Volume*, pages 187–204, 1948. 54

[147] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988. doi: 10.1016/0022-0000(88)90046-3. 10, 282, 283, 287

[148] David S Johnson, Christos H Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of computer and system sciences*, 37(1):79–100, 1988. 102, 150

[149] Shizuo Kakutani. A generalization of Brouwer's fixed point theorem. *Duke Mathematical Journal*, 8(3):457–459, 1941. doi: 10.1215/S0012-7094-41-00838-4. 3, 11, 32, 38, 39, 93, 99, 135

[150] George Karakostas and Anastasios Viglas. Equilibria for networks with malicious users. *Mathematical Programming*, 110(3):591–613, 2007. 150

[151] William Karush. Minima of functions of several variables with inequalities as side conditions. Master's thesis, University of Chicago, Department of Mathematics, 1939. 54

[152] Alexander S. Kelso and Vincent P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):1483–1504, 1982. ISSN 00129682, 14680262. URL `http://www.jstor.org/stable/1913392`. 21

[153] Shiva Kintali, Laura J. Poplawski, Rajmohan Rajaraman, Ravi Sundaram, and Shang-Hua Teng. Reducibility among fractional stability problems. *SIAM J. Comput.*, 42(6):2063–2113, 2013. 92, 98, 101, 134, 145, 146

[154] Max Klimm and Maximilian J Stahlberg. Complexity of equilibria in binary public goods games on undirected graphs. *arXiv preprint arXiv:2301.11849*, 2023. 136

[155] Max Klimm and Philipp Warode. Complexity and parametric computation of equilibria in atomic splittable congestion games via weighted block Laplacians. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2728–2747. SIAM, 2020. 92, 96, 98, 102, 103, 150, 151, 162

[156] Bronisław Knaster, Casimir Kuratowski, and Stefan Mazurkiewicz. Ein Beweis des Fixpunktsatzes für *n*-dimensionale Simplexe. *Fundamenta Mathematicae*, 14:132–137, 1929. doi: 10.4064/fm-14-1-132-137. 33, 34, 66, 68, 72, 74, 75, 226

[157] Daphne Koller, Nimrod Megiddo, and Bernhard Von Stengel. Efficient computation of equilibria for extensive two-person games. *Games and economic behavior*, 14(2):247–259, 1996. 98, 144

[158] Harold W. Kuhn and Albert W. Tucker. Nonlinear programming. In *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, 1951. 54

[159] Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55(2): 270–296, 2006. doi: 10.1016/j.geb.2005.02.006. 22, 275, 276, 277

[160] Carlton E Lemke. Bimatrix equilibrium points and mathematical programming. *Management science*, 11(7):681–689, 1965. 92, 96, 135, 147, 164

[161] Carlton E Lemke and Joseph T Howson, Jr. Equilibrium points of bimatrix games. *Journal of the Society for industrial and Applied Mathematics*, 12(2): 413–423, 1964. 92, 96, 99, 135

[162] Juncheng Li and Pingzhong Tang. Auto-bidding equilibrium in ROI-constrained online advertising markets. *arXiv preprint arXiv:2210.06107*, 2023. 106, 107, 212, 213, 219, 220, 221

[163] Richard J. Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC)*, pages 125–131, 2004. doi: 10.1145/988772.988792. 22, 275, 276

[164] Olvi L. Mangasarian and Stan Fromovitz. The Fritz John necessary optimality conditions in the presence of equality and inequality constraints. *Journal of Mathematical Analysis and Applications*, 17(1):37–47, 1967. doi: 10.1016/0022-247X(67)90163-1. 54

[165] Pasin Manurangsi and Warut Suksompong. Closing gaps in asymptotic fair division. *SIAM Journal on Discrete Mathematics*, 35(2):668–706, 2021. doi: 10.1137/20m1353381. 276

[166] Andreu Mas-Colell, Michael Dennis Whinston, and Jerry R. Green. *Microeconomic theory*. Oxford University Press, 1995. 78

[167] Robert R Maxfield. General equilibrium and the theory of directed graphs. *Journal of Mathematical Economics*, 27(1):23–51, 1997. 167, 176, 192, 206

[168] Lionel McKenzie. On equilibrium in Graham's model of world trade and other competitive systems. *Econometrica*, 22(2):147–161, 1954. doi: 10.2307/1907539. 37

[169] Nimrod Megiddo and Christos H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2): 317–324, 1991. doi: 10.1016/0304-3975(91)90200-L. 282

[170] Nimrod Megiddo and Christos H Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2): 317–324, 1991. 8, 92, 111

[171] Jean-François Mertens and Abraham Neyman. Stochastic games. *International Journal of Game Theory*, 10(2):53–66, 1981. doi: 10.1007/BF01769259. 36

[172] Frédéric Meunier and Thomas Pradeau. Computing solutions of the multiclass network equilibrium problem with affine cost functions. *Annals of Operations Research*, 274:447–469, 2019. 92, 96, 98, 102, 103, 150, 151, 161

[173] Igal Milchtaich. Generic uniqueness of equilibrium in large crowding games. *Mathematics of Operations Research*, 25(3):349–364, 2000. 102, 151

[174] Roger B. Myerson. Refinements of the Nash equilibrium concept. *International Journal of Game Theory*, 7(2):73–80, 1978. doi: 10.1007/BF01753236. 32, 36, 58, 60, 61, 101, 134, 146, 148

[175] John Nash. Non-cooperative games. *Annals of Mathematics*, 2(54):286–295, 1951. doi: 10.2307/1969529. 3, 10, 15

[176] John Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295, 1951. doi: 10.2307/1969529. 35

[177] John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950. doi: 10.1073/pnas.36.1.48. 11, 15, 29, 35, 41, 91, 99, 111, 135, 140

[178] Abraham Neyman and Sylvain Sorin, editors. *Stochastic games and applications*, volume 570 of *NATO Science Series C: Mathematical and Physical Sciences*. Springer, 2003. doi: 10.1007/978-94-010-0189-2. 36

[179] Miquel Oliu-Barton. New algorithms for solving zero-sum stochastic games. *Mathematics of Operations Research*, 46(1):255–267, 2021. doi: 10.1287/moor.2020.1055. 36

[180] Renato Paes Leme. Gross substitutability: An algorithmic survey. *Games and Economic Behavior*, 106:294–316, 2017. doi: 10.1016/j.geb.2017.10.016. 21, 279

[181] Christos Papadimitriou and Binghui Peng. Public goods games in directed networks. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 745–762, 2021. 100, 136, 137

[182] Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci*, 48(3):498–532, 1994. 8, 10, 30, 35, 67, 92, 98, 99, 111, 135, 242, 250

[183] Christos H Papadimitriou and Tim Roughgarden. Computing correlated equilibria in multi-player games. *Journal of the ACM (JACM)*, 55(3):1–29, 2008. 100, 142

[184] Christos H. Papadimitriou, Emmanouil-Vasileios Vlatakis-Gkaragkounis, and Manolis Zampetakis. The computational complexity of multi-player concave games and Kakutani fixed points. *arXiv preprint arXiv:2207.07557*, 2022. 101

[185] Benjamin Plaut and Tim Roughgarden. Almost envy-freeness with general valuations. *SIAM Journal on Discrete Mathematics*, 34(2):1039–1068, 2020. doi: 10.1137/19m124397x. 22, 23, 24, 274, 275, 276, 283, 284, 286

[186] Henri Poincaré. Sur les courbes définies par les équations différentielles (III). *Journal de Mathématiques Pures et Appliquées*, 4(1):167–244, 1885. 38

[187] Ariel D. Procaccia. Thou shalt covet thy neighbor's cake. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 239–244, 2009. 37

[188] Ariel D. Procaccia. Cake cutting: Not just child's play. *Communications of the ACM*, 56(7):78–87, 2013. doi: 10.1145/2483852.2483870. 33, 66, 107, 226

[189] Richard Rado. Note on independence functions. *Proceedings of the London Mathematical Society*, s3-7(1):300–320, 1957. doi: 10.1112/plms/s3-7.1.300. 279

[190] Jack Robertson and William Webb. *Cake-cutting algorithms: Be fair if you can*. CRC Press, 1998. 30, 33, 36, 66, 67, 107, 226

[191] Ralph Tyrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970. 51, 89

[192] J. B. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, 33(3):520–534, 1965. doi: 10.2307/1911749. 32, 58, 59, 60, 100, 139, 140, 144

[193] Robert W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2(1):65–67, 1973. doi: 10.1007/BF01737559. 149

[194] Aviad Rubinstein. Settling the complexity of computing approximate two-player Nash equilibria. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 258–265, 2016. doi: 10.1109/FOCS.2016.35. 36

[195] Rahul Savani. *Finding Nash equilibria of bimatrix games*. PhD thesis, London School of Economics and Political Science, 2006. 96

[196] Herbert Scarf. The approximation of fixed points of a continuous mapping. *SIAM Journal on Applied Mathematics*, 15(5):1328–1343, 1967. doi: 10.1137/0115116. 37

[197] Marcus Schaefer and Daniel Štefankovič. Fixed points, Nash equilibria, and the existential theory of the reals. *Theory Comput Syst*, 60:172–193, 11 2017. doi: 10.1007/s00224-015-9662-0. 248

[198] David Schmeidler. Equilibrium points of nonatomic games. *Journal of statistical Physics*, 7:295–300, 1973. 102, 151

[199] Reinhard Selten. Reexamination of the perfectness concept for equilibrium points in extensive games. *International Journal of Game Theory*, 4:25–55, 1975. 60, 101, 146, 148

[200] Lloyd S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10):1095–1100, 1953. doi: 10.1073/pnas.39.10.1095. 32, 36, 59, 63, 64

[201] Forest W. Simmons and Francis Edward Su. Consensus-halving via theorems of Borsuk-Ulam and Tucker. *Math. Soc. Sci.*, 45(1):15–25, 2003. doi: 10.1016/S0165-4896(02)00087-2. 15, 17, 241

[202] Morton Slater. Lagrange multipliers revisited. Cowles Commission Discussion Paper: Mathematics 403, Cowles Foundation for Research in Economics, Yale University, 1950. URL `https://elischolar.library.yale.edu/cowles-discussion-paper-series/304`. 31, 49, 228

[203] Troels Bjerre Sørensen. Computing a proper equilibrium of a bimatrix game. In Boi Faltings, Kevin Leyton-Brown, and Panos Ipeirotis, editors, *ACM Conference on Electronic Commerce, EC '12*, pages 916–928. ACM, 2012. 92, 96, 98, 102, 146, 147, 148, 149

[204] Emanuel Sperner. Neuer Beweis für die Invarianz der Dimensionszahl und des Gebietes. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 6:265–272, 1928. doi: 10.1007/BF02940617. 8, 33, 66, 97, 106, 108, 215, 225, 226

[205] Hugo Steinhaus. The problem of fair division. *Econometrica*, 16(1):101–104, 1948. URL `https://www.jstor.org/stable/1914289`. 273

[206] Hugo Steinhaus. Sur la division pragmatique. *Econometrica*, 17:315–319, 1949. doi: 10.2307/1907319. 36

[207] Hugo Steinhaus. Sur la division pragmatique. *Econometrica*, 17(Suppl.): 315–319, 1949. doi: 10.2307/1907319. 273

[208] Hugo Steinhaus. Sur la division pragmatique. *Econometrica: Journal of the Econometric Society*, pages 315–319, 1949. 107, 225

[209] Walter Stromquist. How to cut a cake fairly. *The American Mathematical Monthly*, 87(8):640–644, 1980. doi: 10.1080/00029890.1980.11995109. 33, 36, 66, 91, 107

[210] Walter Stromquist. How to cut a cake fairly. *The American Mathematical Monthly*, 87(8):640–644, 1980. doi: 10.1080/00029890.1980.11995109. 274

[211] Francis Edward Su. Borsuk-Ulam implies Brouwer: A direct construction. *The American Mathematical Monthly*, 104(9):855–859, 1997. doi: 10.2307/2975293. 238

[212] Francis Edward Su. Rental harmony: Sperner's lemma in fair division. *The American Mathematical Monthly*, 106(10):930–942, 1999. doi: 10.1080/00029890.1999.12005142. 30, 33, 36, 66, 107, 108, 225, 226, 231, 233

[213] Francis Edward Su. Rental harmony: Sperner's lemma in fair division. *The American Mathematical Monthly*, 106(10):930–942, 1999. doi: 10.1080/00029890.1999.12005142. 274

[214] Masayuki Takahashi. Equilibrium points of stochastic non-cooperative n-person games. *Journal of Science of the Hiroshima University Series A-I (Mathematics)*, 28(1):95–99, 1964. doi: 10.32917/hmj/1206139509. 32, 64, 65

[215] Sergey P. Tarasov and Mikhail N. Vyalyi. Semidefinite programming and arithmetic circuit evaluation. *Discrete Applied Mathematics*, 156(11):2070 – 2078, 2008. ISSN 0166-218X. doi: 10.1016/j.dam.2007.04.023. 238

[216] Michael J Todd. Orientation in complementary pivot algorithms. *Mathematics of Operations Research*, 1(1):54–66, 1976. 96

[217] Hirofumi Uzawa. Walras' existence theorem and brouwer's fixed-point theorem. *The Economic studies quarterly*, 13:59–62, 1962.

[218] Hal R. Varian. Equity, envy, and efficiency. *Journal of Economic Theory*, 9(1): 63–91, 1974. doi: 10.1016/0022-0531(74)90075-1. 274

[219] Hal R. Varian. Equity, envy, and efficiency. *Journal of Economic Theory*, 9(1): 63–91, 1974. doi: 10.1016/0022-0531(74)90075-1. 85

[220] Vijay V Vazirani and Mihalis Yannakakis. Market equilibrium under separable, piecewise-linear, concave utilities. *Journal of the ACM (JACM)*, 58(3):1–25, 2011. 10, 37, 92, 93, 97, 98, 104, 163, 164, 167, 203

[221] Vijay V. Vazirani and Mihalis Yannakakis. Computational complexity of the Hylland-Zeckhauser scheme for one-sided matching markets. In *Proceedings of 12th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 59:1–59:19, 2021. doi: 10.4230/LIPIcs.ITCS.2021.59. 35, 75, 85

[222] Alexey Yu. Volovikov. Borsuk-Ulam implies Brouwer: A direct construction revisited. *Am. Math. Mon.*, 115(6):553–556, 2008. 238, 243, 258

[223] John von Neumann. Zur Theorie der Gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928. doi: 10.1007/BF01448847. 35

[224] John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944. 35

[225] Léon Walras. *Éléments d'économie politique pure*. L. Corbaz et Cie, 1874. 21, 37, 76

[226] J G Wardrop. Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers*, 1(3):325–362, 1952. doi: 10.1680/ipeds.1952.11259. 149, 153

[227] Gerhard J. Woeginger and Jiří Sgall. On the complexity of cake cutting. *Discrete Optimization*, 4(2):213–220, 2007. doi: 10.1016/j.disopt.2006.07.003. 36

[228] Douglas R. Woodall. Dividing a cake fairly. *Journal of Mathematical Analysis and Applications*, 78(1):233–247, 1980. doi: 10.1016/0022-247x(80)90225-5. 15, 274

[229] Douglas R. Woodall. Dividing a cake fairly. *Journal of Mathematical Analysis and Applications*, 78(1):233–247, 1980. doi: 10.1016/0022-247X(80)90225-5. 34, 67, 107, 226

[230] Mihalis Yannakakis. Equilibria, fixed points, and complexity classes. *Computer Science Review*, 3(2):71–85, 2009. doi: 10.1016/j.cosrev.2009.03.004. 29