Towards More Expressive Cake Cutting

Ioannis Caragiannis
University of Patras
caragian@ceid.upatras.gr

John K. Lai
Harvard SEAS
jklai@post.harvard.edu

Ariel D. Procaccia
Harvard SEAS
arielpro@seas.harvard.edu

Abstract

Cake cutting is a playful name for the problem of fairly dividing a heterogeneous divisible good among a set of agents. The agent valuations for different pieces of cake are typically assumed to be additive. However, in certain practical settings this assumption is invalid because agents may not have positive value for arbitrarily small "crumbs" of cake. In this paper, we propose a new, more expressive model of agent valuations that captures this feature. We present an approximately proportional algorithm for any number of agents that have such expressive valuations. The algorithm is optimal in the sense that no other algorithm can guarantee a greater worst-case degree of proportionality. We also design an optimal approximately proportional and fully envy-free algorithm for two agents.

1 Introduction

The problem of allocating a heterogeneous, divisible resource is often described, in intuitive terms, as the problem of cutting a cake. The participating agents value pieces of the cake differently, e.g., based on their preference for different toppings. The challenge is to divide the cake *fairly* among the agents.

Specifically, the cake is represented as the interval $[0, 1]$. A piece of cake is a finite collection of subintervals of $[0, 1]$. Each agent holds a *valuation function* that assigns values to pieces of cake. These valuation functions are typically assumed to be *additive*, in the sense that the sum of an agent's values for two disjoint intervals is equal to its value for their union. It is also assumed that valuations are *normalized*, i.e., an agent assigns a value of one to the entire cake. A division of the cake is *proportional* if each of the n agents receives a piece of cake that is worth at least $1/n$ according to the agent's valuation function; and *envy-free (EF)* if each agent assigns a value to its own piece of cake that is at least as high as the value it assigns to the piece allocated to any other agent. Under standard assumptions on the valuation functions, envy-freeness implies proportionality,[1] and EF allocations always exist (see, e.g., [Brams and Taylor, 1996]).

[1]Because of additivity there must be a piece worth $1/n$ to an agent, and then envy-freeness implies that the agent's piece is worth at least $1/n$.

The cake cutting problem was first proposed in the 1940's, and since then has been the focus of work in several different fields, including mathematics, economics, and political science (see, e.g., the books by Robertson and Webb [1998] and Brams and Taylor [1996]). In the last decade, computer scientists have investigated various aspects of the cake cutting setting (see, e.g., [Edmonds and Pruhs, 2006]), and indeed the perspective of computer science is called for because the cake cutting literature is algorithmic in nature. Very recently, this problem has attracted the attention of members of the AI community [Procaccia, 2009; Chen *et al.*, 2010]. Again, the ties with AI are natural when viewed through the lens of multiagent systems, which give rise to fundamental resource allocation problems [Chevaleyre *et al.*, 2006].

Algorithmic model and restricted valuations. Most of the literature on discrete cake cutting algorithms can be mapped into a query model where the algorithm can query the agents for their value for a given piece or ask the agents to cut a piece that is worth a specific value [Robertson and Webb, 1998]. The actual agent valuations are never fully revealed, and indeed, this is important because fully general valuation functions cannot be succinctly represented. An example of a procedure that operates under this query model is the well-known *Cut-and-Choose* mechanism. The first agent is asked to cut a piece that is worth exactly 1/2 (with an end point at 0). The second agent is then asked to evaluate the two pieces and is allocated the piece that it prefers.

In this paper, we adopt a different algorithmic model that is in the spirit of direct revelation mechanisms and follows the approach of Chen et al. [2010]. We assume that agents report their valuation functions and the algorithm operates on these reports. For this to be a reasonable approach, we must be able to tractably specify the valuation functions. This is made possible by considering a restricted family of valuation functions that can be succinctly represented.

The main result of Chen et al. [2010] concerns a setting where agents have *piecewise uniform valuations*. Each agent is associated with a set of desired subintervals of $[0, 1]$, which the agent values uniformly. The value that an agent assigns to a piece of cake is then proportional to the total length of its intersection with the agent's desired intervals. In other words, each agent simply wants to receive as much of its desired intervals as possible. An agent's valuation is fully specified by the intervals of interest and the valuations can be tractably re-

ported to the algorithm. Chen et al. argue that these valuation functions are natural in realistic settings. For example, the cake can correspond to access time to a shared backup server. Each agent is interested in backing up its data when its computer is idle, but it is indifferent between idle time slots, and moreover can resume the backup even during very short idle time slots.

More expressive valuations. In some settings where agents' valuation functions are plausibly piecewise uniform, agents have no use for "crumbs" of cake, i.e., very small subintervals. For example, thinking again of the cake as time, consider the allocation of advertising time: an agent may be interested in time slots when related shows are aired, but does not derive value from slots that are shorter than, say, thirty seconds. Similarly, one can consider priority access time to an Internet service provider for, e.g., streaming video or gaming. As a different example, consider the allocation of a strip of beach to real estate developers; each developer is interested in areas with specific properties, but has no use for tiny plots.

Therefore, more *expressive* valuations are called for. We augment piecewise uniform valuations with an option to specify the minimum length of a usable interval; we call these augmented valuations *piecewise uniform with minimum length (PUML)*. An agent's value for a piece of cake under PUML valuations is proportional to the total length of its intersection with the agent's desired intervals, excluding subintervals in the intersection that are shorter than the agent's specified minimum length.

PUML valuations depart from those used in the cake cutting literature in that they can be non-additive: it may be the case that two disjoint intervals have zero value but their union forms a contiguous interval that is longer than the minimum. In this respect, PUML valuations blur the line between divisible and indivisible goods: some divisions of a good (i.e., an interval) are possible, whereas other divisions are impossible (as they would induce worthless subintervals). We are aware of a single existing paper that studies cake cutting under non-additive valuations [Berliant *et al.*, 1992, Section 5], but these valuations do not include PUML valuations, and are studied only in the context of the existence of Pareto-efficient allocations. Expressiveness in mechanisms has been studied, e.g., by Benisch et al. [2008], but that work focuses on the tradeoffs between simplicity and expressiveness. We focus on designing algorithms for our more expressive valuations.

Our results. Consider two agents with PUML valuations, where the minimum length for each agent is 1 (that is, any strict subinterval of the entire cake is worthless). Clearly no proportional allocation exists, but an EF allocation does exist: by assigning some nonempty piece of cake to each agent, both agents have value zero for both pieces (this shows that under PUML valuations envy-freeness does not imply proportionality). Worse, one of the agents must have value zero, so even approximate proportionality in a multiplicative sense, as studied in [Edmonds and Pruhs, 2006], is unattainable. We therefore consider approximate proportionality in an additive sense, to be made formal later.

In Section 3 we propose a polynomial-time algorithm for any number of agents with PUML valuations that provides an additive worst-case approximate proportionality guarantee. The algorithm is a generalization of a well-known fully proportional algorithm in the traditional cake cutting setting. We also prove that our algorithm is optimal, as no algorithm can attain a better worst-case proportionality guarantee.

With proportionality understood, in Section 4 we consider envy-freeness in combination with proportionality. For two agents with PUML valuations, we find that (rather surprisingly) we can obtain full envy-freeness while still satisfying the optimal approximate proportionality guarantee, in polynomial time. We do this via an algorithm that is very different from the approximately proportional algorithm for n agents and makes extensive use of discarding intervals in order to attain full envy-freeness.

2 The Model

We seek to divide a heterogeneous cake, represented by $[0, 1]$, among a set of agents $N = \{1, 2, \ldots, n\}$. A *piece of cake* X is a finite union of disjoint subintervals of $[0, 1]$. We will treat X as a set of inclusion maximal contiguous subintervals. For example, the piece $[0, 0.25] \cup [0.25, 0.5] \cup [0.75, 1]$ is treated as $\{[0, 0.5], [0.75, 1]\}$. If I is a contiguous interval, then $|I|$ denotes the absolute length of that interval, treating open and closed intervals equivalently. Similarly, given a piece of cake X, $|X|$ is defined by summing the length of each interval in X.

2.1 The classic model

In the classic model of cake cutting, each agent $i \in N$ has a *value density function* $v_i : [0, 1] \to [0, \infty)$ that is piecewise continuous. The value density function v_i induces the valuation function of agent i, by letting $V_i(X) = \sum_{I \in X} \int_I v_i(x)dx$. $V_i(X)$ is *non-atomic*, that is $V_i([x, x]) = 0$. As a result, V_i takes the same value on open and closed intervals with the same end points.

An *allocation* is an assignment of pieces of cake X_i to each agent i such that the assigned pieces are disjoint (but we allow the pieces to overlap at the boundaries). An allocation is *proportional* with respect to $V_1, \ldots, V_n$ if $V_i(X_i) \geq 1/n$ for all $i \in N$, and *envy-free* (EF) with respect to $V_1, \ldots, V_n$ if $V_i(X_i) \geq V_i(X_j)$ for all $i, j \in N$, i.e., each agent weakly prefers its assigned piece. As noted above, in the classic model envy-freeness implies proportionality, due to additivity of the valuation functions.

A *cake cutting algorithm* is a function from $V_1, \ldots, V_n$ to an allocation. A cake cutting algorithm is proportional (resp. EF) if it always returns an allocation that is proportional (resp. EF) with respect to $V_1, \ldots, V_n$. Note that we reason only about the agents' reports and not the agents' actual valuations since those are unknown to the algorithm. Henceforth when we refer to an agent's value for a given piece of cake X, we mean $V_i(X)$, which may differ from the agent's true value.

2.2 Piecewise Linear Valuations

We examine the special case where valuation functions are *piecewise uniform* [Chen *et al.*, 2010]. This requires the value density function $v_i(x)$ to be either 0 or some constant $r_i > 0$ which is chosen so that $\int_0^1 v_i(x)dx = 1$.

Under piecewise uniform valuations, we can succinctly represent an agent's valuation function by providing the intervals in which it is uniformly interested. Specifically, let $D(i, X)$ give the subintervals of X that are also of interest to agent i. For instance, if agent 1 uniformly desires $[0, 0.25], [0.3, 1]$, then $D(1, [0.2, 0.4]) = \{[0.2, 0.25], [0.3, 0.4]\}$. Let $d(i, X) = |D(i, X)|$, the length of intervals in X desired by agent i. With piecewise uniform valuation functions, V_i now has a simpler form,

$$V_i(X) = \frac{d(i, X)}{d(i, [0, 1])}.$$

In words, it is the ratio of desired lengths received to the total lengths desired by agent i. For example, if $D(i, [0, 1]) = \{[0, 0.2], [0.5, 0.8]\}$ and $X = \{[0.1, 0.3], [0.4, 0.7]\}$, then $D(i, X) = \{[0.1, 0.2], [0.5, 0.7]\}$, $d(i, X) = 0.3$, $d(i, [0, 1]) = 0.5$, and $V_i(X) = 0.6$. Under piecewise uniform valuations it is very simple to construct an EF (and therefore proportional) cake cutting algorithm; we leave this as an easy exercise for the reader. The challenge in [Chen *et al.*, 2010] stemmed not from the fairness criteria, but from their pursuit of truthfulness.

2.3 PUML valuations

As discussed above, we wish to augment piecewise uniform valuations with a minimum length parameter. Given $i \in N$, the minimal length parameter λ_i indicates that agent i has no value for intervals of length less than λ_i.

Definition 1. Under valuations that are *piecewise uniform with minimum length (PUML)*, each agent $i \in N$ uniformly desires a piece of cake $D(i, [0, 1])$, and holds a minimum length parameter λ_i. The valuation function of the agent is defined by

$$V_i(X) = \frac{\sum_{I \in D(i,X):\ |I| \geq \lambda_i} |I|}{d(i, [0, 1])}.$$

Note that the summation only includes intervals with length at least λ_i.

Going back to the previous example, if $D(i, [0, 1]) = \{[0, 0.2], [0.5, 0.8]\}$ and $X = \{[0.1, 0.3], [0.4, 0.7]\}$, and in addition $\lambda_i = 0.2$, then $\{I \in D(i, X) : |I| \geq \lambda_i\} = \{(0.5, 0.7)\}$, and therefore $V_i(X) = 0.4$.

We will assume that every interval I in $D(i, [0, 1])$ satisfies $|I| \geq \lambda_i$, that is, agents do not desire worthless intervals. We also assume *free disposal*, in that we can choose not to allocate part of the cake without penalty. Under PUML valuations, this assumption is without loss of generality, because we can "destroy" intervals by partitioning them into worthless tiny subintervals. Furthermore, under PUML valuations, and generally under the free disposal assumption, proportionality and envy-freeness are incomparable. In particular, envy-freeness does not imply proportionality because an allocation that throws away the entire cake is EF but not proportional.

In order to discuss computational complexity, as we do below, we must understand how the input is represented. PUML valuations can be concisely represented via the boundaries of the desired intervals, and $\lambda_1, \ldots, \lambda_n$. The size of the input is the number of bits used to represent these parameters. As in [Chen *et al.*, 2010] and discussed in Section 1, we assume that agents report their entire valuation function to the algorithm.

3 Proportionality

Under classic assumptions on the valuations, it is well known that proportional allocations always exist for any number of agents. A simple algorithm that achieves this is the Dubins-Spanier procedure (see, e.g., [Brams and Taylor, 1996]). A referee moves a knife from the left end of the cake to the right. An agent shouts "stop" when the piece of cake to the left of the knife is worth $1/n$ according to its valuation; the piece is then cut and given to that agent, and the procedure resumes with the remaining cake and the rest of the agents; the last agent receives what's left of the cake. It is a simple exercise to show that this results in a proportional allocation.

Under PUML valuations, proportionality (or even multiplicative approximate proportionality) is not always achievable, as demonstrated by the example given in the introduction where both agents desire the entire cake and $\lambda_1 = \lambda_2 = 1$. Therefore, we seek to achieve an additive approximate proportionality guarantee. This guarantee will depend on λ_i: agents with larger λ_i are guaranteed less, as having a larger λ_i restricts the allocations that can give the agent its proportionality guarantee. Let $\ell_i = \lambda_i / d(i, [0, 1])$ for each $i \in N$.

Definition 2. An allocation $X_1, \ldots, X_n$ is *β-proportional* with respect to valuations $V_1, \ldots, V_n$ if for all $i \in N$, $V_i(X_i) \geq 1/n - \beta \cdot \ell_i$. A cake cutting algorithm is *β-proportional* if when given input $V_1, \ldots, V_n$, it always produces an allocation that is β-proportional with respect to $V_1, \ldots, V_n$.

Equivalently, a β-proportional algorithm guarantees that

$$\sum_{I \in D(i,X_i):\ |I| \geq \lambda_i} |I| \geq \frac{d(i, [0, 1])}{n} - \beta \cdot \lambda_i.$$

3.1 Algorithmic results

To achieve approximately proportional allocations, we present Algorithm 1. This algorithm is inspired by the Dubins-Spanier procedure, but shifts the points where agents metaphorically say "stop" in a way that, as we shall see, provides optimal guarantees. The output of the algorithm is the allocation $X_1, \ldots, X_n$, which is fully assigned before the algorithm returns.

Algorithm 1 $(2(n-1)/n)$-proportional algorithm

Input: $V_1, \ldots, V_n$

1. SUBROUTINE$(N, 0, (V_1, \ldots, V_n))$

SUBROUTINE$(S, u, (V_1, \ldots, V_n))$:

1. If $S = \{i\}$, set $X_i = [u, 1]$ and return.
2. For each $i \in S$:
   $r_i^* = \min\{r : r \in [u, 1], V_i([u, r]) \geq \frac{V_i([u,1])}{|S|} - \frac{2(|S|-1)\ell_i}{|S|}\}$.
3. $r^* = \min_{i \in S} r_i^*, i^* = \arg\min_{i \in S} r_i^*$ (break ties arbitrarily).
4. Set $X_{i^*} = [u, r]$.
5. SUBROUTINE$(S \setminus \{i^*\}, r)$

This is obviously a polynomial-time algorithm. The following theorem quantifies its proportionality guarantees.

Theorem 3. *Under PUML valuations, Algorithm 1 is* $(2(n-1)/n)$*-proportional and polynomial-time.*

In particular, the algorithm is at most 1-proportional for $n = 2$. Before proving the theorem, we establish a simple lemma.

Lemma 4. *Consider a contiguous interval* $[u, v]$*. Let* $w \in [u, v]$*. Then* $V_i([w, v]) \geq V_i([u, v]) - V_i([u, w]) - 2\ell_i$*.*

Proof. $V_i([u, w]) + V_i([w, v])$ can be smaller than $V_i([u, v])$ because the cut point w might break an interval that was previously of length at least λ_i into two intervals that have length less than λ_i. For instance, suppose $w \in (b, c)$ where $(b, c) \in D_i([u, v])$. If $w - b \geq \lambda_i, c - w \geq \lambda_i$, then no value is lost by adding the values for $[u, w], [w, v]$ separately compared to the value for $[u, v]$. However, if $w - b < \lambda_i$ or $c - w < \lambda_i$, then we lose value by adding values over $[u, w], [w, v]$ separately. The most value that can be lost is $2\ell_i$ (ℓ_i on either side of the cut at w). Therefore $V_i([u, w]) + V_i([w, v]) \geq V_i([u, v]) - 2\ell_i$, and rearranging yields the lemma. □

Proof of Theorem 3 (sketch). The proof proceeds by induction on $|S|$ and showing that SUBROUTINE gives each $i \in S$ at least value $V_i([u, 1])/|S| - 2(|S|-1)\ell_i/|S|$, i.e., SUBROUTINE is $2(|S|-1)/|S|$- proportional with respect to $[u, 1]$ and the agents in S. This is straightforward to show for the agent that was allocated a piece. For the remaining agents, we use the fact that the agents were not chosen in combination with Lemma 4 to conclude that they have a large enough value for the unallocated cake such that the inductive hypothesis provides their proportionality guarantee. □

3.2 Impossibility results

Consider once more the case of two agents. In this case Algorithm 1 guarantees that each agent receives value of $1/2 - \ell_i$. Suppose that both agents desire the entire cake and $\lambda_1 = \ell_1 = \lambda_2 = \ell_2 = 1/2 + \epsilon$; then one agent will receive value of zero, that is, it is impossible to guarantee a value of more than $1/2 - \ell_i + \epsilon$ for any $\epsilon > 0$; hence the algorithm is optimal for the case of $n = 2$. More generally, the following theorem establishes that the algorithm is worst-case optimal for any number of agents; the proof is omitted due to space constraints.

Theorem 5. *For every* n *there exist PUML valuations such that no cake cutting algorithm is* β*-proportional for* $\beta < 2(n-1)/n$*.*

Theorem 5 does not exclude the possibility that, for a given instance, there is an allocation with a better degree of proportionality than the one computed by Algorithm 1. Indeed, there could be an algorithm that matches Algorithm 1 in the worst case but returns allocations that have better proportionality guarantees in other cases. However, the combinatorial richness of PUML valuations imposes limits on what can be achieved via *polynomial-time* algorithms due to the following "inapproximability result", whose omitted proof gives a reduction from the NP-hard 3-dimensional matching problem.

Theorem 6. *For any constant* $\epsilon \in (0, 1/2)$*, given* n *agents with PUML valuations such that* $\ell_i < 1/n$*, it is NP-hard to distinguish between the following two statements: (a) there is a* $(1/2 + \epsilon)$*-proportional allocation and (b) no* $(3/2 - \epsilon)$*-proportional allocation exists.*

In particular, for a given set of valuations $V_1, \ldots, V_n$, let $\gamma(V_1, \ldots, V_n)$ be the smallest value γ such that a γ-proportional allocation exists. Theorem 6 says that there is no polynomial time algorithm that always returns a $(\gamma(V_1, \ldots, V_n) + 1 - 2\epsilon)$-proportional allocation for every set of valuations (if there were then we could solve the NP-hard problem in the statement of Theorem 6). In other words, there is no polynomial-time algorithm that approximates the best proportional allocation within an additive factor of 1. Algorithm 1 is close to optimal when viewed under this measure since it provides an additive $2(n-1)/n$ approximation guarantee.

4 Proportionality and Envy-Freeness

While Theorem 3 provides an optimal worst-case proportionality guarantee, it does not address envy-freeness. In fact, it is possible for an agent to be the first allocated yet have greater value for pieces later allocated to other agents.

A natural question to ask is whether we can attain envy-freeness while satisfying the proportionality guarantee of Algorithm 1. In other words, is there an algorithm that is $2(n-1)/n$-proportional and fully EF? For the case of two agents, we show that indeed there is an algorithm (that is 1-proportional and EF), and we leave open the challenging question of a general number of agents.

Under general classic valuations, finding an EF (and therefore also proportional) allocation is trivial when $n = 2$: agent 1 cuts the cake into two pieces that it values equally, and agent 2 chooses the piece that it prefers. This simple algorithm is known as Cut-and-Choose. However, immediate variations of this algorithm do not yield envy-freeness under PUML valuations. In fact, we will see that achieving envy-freeness and 1-proportionality under PUML valuations is surprisingly difficult. Our solution makes extensive use of the free disposal assumption, which was not required above, in order to attain envy-freeness.

We introduce some new notation that is specific to this section. Lengths of intervals will be denoted by Greek letters. It will be convenient to refer to disjoint subintervals of a given interval. We define a *filtering* F to be a function that takes an interval and returns a set of disjoint subintervals of the given interval. For instance, for the interval $[0, 0.25]$, we might choose to only allocate $\{[0, 0.1], [0.2, 0.25]\}$, throwing away $[0.1, 0.2]$. In this case, $F([0, 0.25]) = \{[0, 0.1], [0.2, 0.25]\}$.

4.1 An algorithmic skeleton

We first observe that if a filtering with specific properties exists, then a 1-proportional and EF allocation exists for two agents.

Definition 7. A filtering-point pair (F_i, x_i) is *fair* if

$$V_i(F_i([0, x_i])) = V_i(F_i([x_i, 1])) \quad (1)$$

$$V_i(F_i([0, x_i])) \geq 1/2 - \ell_i \quad (2)$$

Assuming that we can find fair filtering-point pair, Algorithm 2 is 1-proportional and EF. The high level idea is to start with a feasible allocation is that 1-proportional based on the fair filtering-point pair. If some agent is envious, then let that agent choose between the pieces generated by the other agent's fair filtering-point pair.

Algorithm 2 1-proportional and EF algorithm for $n = 2$

1. Compute fair filtering-point pairs (F_1, x_1), (F_2, x_2).
2. Assume $x_1 \leq x_2$. Otherwise, the roles can be reversed.
3. Let $X_1 = F_1([0, x_1])$, $X_2 = F_2([x_2, 1])$. If this is EF, return.
4. If both agents are envious, then swap the allocations and return.
5. If agent 1 is envious, let agent 1 choose between $F_2([0, x_2])$ and $F_2([x_2, 1])$, giving agent 2 the piece that was not chosen.
6. If agent 2 is envious, let agent 2 choose between $F_1([0, x_1])$ and $F_1([x_1, 1])$, giving agent 1 the piece that was not chosen.

Lemma 8. *Assume that there exist fair filtering-point pairs (F_1, x_1), (F_2, x_2). Then Algorithm 2 is 1-proportional and EF.*

Proof. If Algorithm 2 terminates at Step 3, then both agents receive their proportionality guarantee by (2) and are not envious. If Algorithm 2 terminates at Step 4, there is no envy since both agents preferred the other's initial allocation. The proportionality guarantee is satisfied since each agent prefers its final allocation to its initial, but the initial allocation satisfies (2). If Algorithm 2 terminates at Step 5, agent 2 receives its proportionality guarantee, is indifferent, and so is not envious. Agent 1 receives its proportionality guarantee since it at worst receives $F_2([x_2, 1])$ which it preferred to $F_1([0, x_1])$. Since agent 1 gets to choose between $F_2([0, x_2])$ and $F_2([x_2, 1])$, agent 1 cannot be envious. A similar argument applies to termination at Step 6. □

If we can carry out Step 1 of Algorithm 2, that is, compute fair filtering-point pairs (F_1, x_1), (F_2, x_2), then Lemma 8 allows us to find a 1-proportional and EF allocation. We show that such filtering-point pairs always exist (and can be computed efficiently) in the next section.

4.2 Finding fair filtering-point pairs

Algorithm 2 implies that we can treat the agents independently, as long as for any v_i and λ_i we can find a filtering-point pair (F_i, x_i). Therefore, we drop the agent subscripts and pretend we are dealing with a single agent.

Before proceeding to the main constructive proof, we establish a result about what lengths in desired intervals are attainable by throwing away intervals in the allocation.

Lemma 9. *Suppose $d(X) = k\lambda + \epsilon$ for some positive integer k and $0 \leq \epsilon < \lambda$, i.e., the agent's desired intervals on X total $k\lambda + \epsilon$. It is possible to throw away intervals in X so that (a) the agent receives desired lengths worth exactly $k\lambda + \epsilon_1$ for any $0 \leq \epsilon_1 \leq \epsilon$ and (b) the agent receives exactly $(k-1)\lambda$ in desired lengths.*

Proof. (a) Let $0 \leq \epsilon_1 \leq \epsilon$. If there is some desired interval that has length greater than 2λ, then we can remove $\epsilon - \epsilon_1$ in lengths from one side of the interval. If no interval has length greater than 2λ, then each interval has length in $[\lambda, 2\lambda)$. The sum of the excess above λ must be at least ϵ, so we can remove lengths of $\epsilon - \epsilon_1$ without decreasing any interval to less than λ in length. (b) By (a), we can attain desired lengths of exactly $k\lambda$. If any remaining interval has length exactly λ, then we can remove that interval. Otherwise, there is at least λ in excess that can be removed without decreasing any interval to length less than λ. □

We now proceed to the main proof that a fair filtering-point pair (F, x) always exists. Let c be the center of the cake from the point of view of the agent, i.e. $d([0, c]) = d([c, 1]) = d([0, 1])/2$. Note that there may be an infinite number of such points, so we take the right-most one.

Let y and z denote the left and right end points of the desired interval that contains c, as seen in Figure 1.

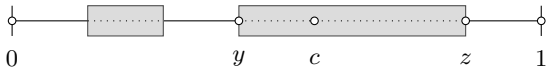

Figure 1: Desired intervals are shaded. The points y and z are the left and right boundaries of the interval containing c.

Although the agent may not receive value $1/2$ from $[0, c]$ and $[c, 1]$ because $c - y$ and $z - c$ may be smaller than λ, $[0, c]$ and $[c, 1]$ always satisfy the proportionality guarantee for both agents. This is formalized in the following Lemma.

Lemma 10. $V([0, c]) \geq 1/2 - \ell, V([c, 1]) \geq 1/2 - \ell$.

In the case where $V([0, c]) = V([c, 1])$, the identity filtering along with c is a fair filtering-point pair.

We therefore assume $V([0, c]) < V([c, 1])$. To construct a fair filtering, point pair (F, x), we choose an $x \in \{y, c, z\}$ based on certain conditions. We then apply Lemma 9 to throw away intervals in $[0, x]$, $[x, 1]$ until the agent is indifferent between the two intervals. Symmetric arguments can be applied to handle the case $v([0, c]) > v([c, 1])$.

We consider two separate cases, based on whether $z - c < \lambda$ or $z - c \geq \lambda$. Since $V([0, c]) < V([c, 1]) \leq 1/2$, we also have $c - y < \lambda$.

Case I: $z - c < \lambda$.

Let $\delta = c - y, \gamma = z - c$, as depicted in Figure 2.

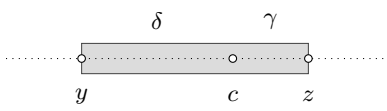

Figure 2: The case where $z - c < \lambda$.

$V([0, c]) \geq V([0, y]) = 1/2 - \delta/d([0, 1])$ and (since $\gamma < \lambda$) $V([c, 1]) = 1/2 - \gamma/d([0, 1])$. Since $V([0, c]) < V([c, 1])$, we obtain that $\gamma < \delta$.

Suppose that the desired lengths in $[z, 1]$ and $[0, z]$ are uniquely expressed as $k\lambda + \rho$ and $k_2\lambda + \rho_2$, respectively, where k, k_2 are positive integers and $0 \leq \rho, \rho_2 < \lambda$. Since $V([0, c]) < V([c, 1])$, $k_2\lambda + \rho_2 < k\lambda + \rho$. There are two cases:

Case I.1: $k_2 = k$, therefore $\rho > \rho_2$. Set $x = c$ and, using Lemma 9a, we throw away intervals in $[z, 1]$ so that the agent receives desired lengths $k_2\lambda + \rho_2$ in both $[0, x]$ and $[x, 1]$.

Case I.2: $k_2 = k-1$. First observe that $\gamma+\delta \geq \lambda$ (otherwise, the agent would not have desired this interval). Set $x = z$ and throw away $[y + \lambda, z]$. The agent receives desired lengths $k\lambda + \rho_2$ and $k\lambda + \rho$ in $[0, x]$ and $[x, 1]$, respectively. If $\rho \geq \rho_2$, by Lemma 9a, we throw away intervals in $[x, 1]$ so that the agent receives desired lengths exactly $k\lambda + \rho_2$ in both $[0, x]$ and $[x, 1]$. If $\rho < \rho_2$, by Lemma 9a, we throw away intervals in $[0, x]$ so that the agent receives desired lengths exactly $k\lambda + \rho$ in both $[0, x]$ and $[x, 1]$.

Case II: $z - c \geq \lambda$**.**

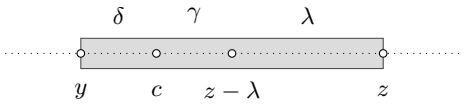

Figure 3: The case where $z - c \geq \lambda$.

$V([c, 1]) = 1/2$ since the agent does not lose desired lengths in $[c, 1]$. Let $\delta = c - y, \gamma = z - \lambda - c$, as seen in Figure 3. It is easy to handle the case where $\gamma \geq \delta$ by throwing away interval $[c, c + \delta]$ and by setting $x = c$. The crucial observation is that since $\gamma \geq \delta$, no desired lengths are lost in $[c + \delta, 1]$ and, hence, the agent has desired lengths of $d([0, 1])/2 - \delta$ in both $[0, x]$ and $[x, 1]$. In the following, assume $\gamma < \delta < \lambda$.

Let $[0, y]$ and $[z - \lambda, 1]$ provide $k_2\lambda + \rho_2$ and $k\lambda + \rho$ in desired lengths for positive integers k, k_2 and $0 \leq \rho, \rho_2 < \lambda$. We distinguish between two cases:

Case II.1: $k_2 = k$. Since $\gamma < \delta$ and $V([0, c]) < V([c, 1])$, $\rho \geq \rho_2$. Set $x = c$, throw away interval $[c, z - \lambda]$ and use Lemma 9a to throw away intervals in $[z - \lambda, 1]$ so that the agent gets exactly $k\lambda + \rho_2$ in both $[0, x]$ and $[x, 1]$.

Case II.2: $k_2 = k - 1$. Then, the interval $[z, 1]$ gives desired lengths of $(k - 1)\lambda + \rho$. First, use Lemma 9b to throw away lengths in $[z, 1]$ so that the desired lengths in $[z, 1]$ are exactly $(k - 2)\lambda$. Set $x = y$. If $\rho_2 \leq \gamma + \delta$, throw away interval $[y, z-\lambda-\rho_2]$ in order to obtain desired lengths of $(k-1)\lambda+\rho_2$ in both $[0, x]$ and $[x, 1]$. If $\rho_2 > \gamma+\delta$, use Lemma 9a to throw away lengths in $[0, x]$ so that the desired lengths in $[0, x]$ and $[x, 1]$ are exactly $(k - 1)\lambda + \gamma + \delta$. This is the only case in which the desired lengths in $[0, x]$ and $[x, 1]$ are less than $V([0, c])$. So, we still need to prove that $(k - 1)\lambda + \gamma + \delta$ satisfies the proportionality requirement (2). By the definition of point c, we have $(k - 1)\lambda + \rho_2 + \delta = \gamma + k\lambda + \rho = d([0, 1])/2$, i.e., $\rho_2 = \lambda+\rho+\gamma-\delta$ which implies $\delta > \rho+\gamma$ since $\rho_2 < \lambda$. Hence, $(k-1)\lambda+\gamma+\delta > (k-1)\lambda+\rho+2\gamma \geq d([0, 1])/2 - \lambda$.

4.3 Tying things together

Now that we have proven the existence of fair (F, x) for any agent valuations, we can apply Lemma 8. In addition, note that Section 4.2 implicitly provides a computationally efficient implementation of Step 1 of Algorithm 2, so the algorithm is clearly polynomial-time. We therefore have the following result.

Theorem 11. *Assume that $n = 2$ and the agents have PUML valuations. Then Algorithm 2 is 1-proportional, envy-free, and polynomial-time.*

5 Discussion

Game-theoretic considerations. A positive side effect of our algorithmic framework is that it encourages agents not to be "greedy": the smaller an agent's λ_i is, the larger the degree of proportionality the agent is guaranteed. We wish to emphasize though that this is not a formal game-theoretic statement. Indeed, under the algorithms presented in this paper, agents can certainly gain by lying about their valuation function or even about their minimum length. In contrast, Chen et al. [2010] design a (fully) proportional, EF algorithm that is also truthful under piecewise uniform valuations (without minimum length). We reiterate that there is a large gap, both conceptual and technical, between piecewise uniform and PUML valuations. It would be interesting to know whether there is a $(2(n - 1)/n)$-proportional and *truthful* algorithm under PUML valuations.

Extending Algorithm 2 to a general number of agents. Algorithm 1, which works for any number of agents, is inspired by a proportional algorithm that works for all valuations functions under classic assumptions. Similarly, in a sense Algorithm 2 extends the Cut-and-Choose algorithm. Unfortunately, in general envy-freeness is hard to obtain for more than two agents and nearly impossible for more than three agents (see, e.g, the introduction of [Procaccia, 2009]), and in particular the techniques of Algorithm 2 do not appear to generalize to any number of agents. Progress on this front would require fundamentally new techniques.

References

[Benisch *et al.*, 2008] M. Benisch, N. M. Sadeh, and T. Sandholm. A theory of expressiveness in mechanisms. In *Proc. of 23rd AAAI*, pages 17–23, 2008.

[Berliant *et al.*, 1992] M. Berliant, K. Dunz, and W. Thomson. On the fair division of a heterogeneous commodity. *Journal of Mathematical Economics*, 21:201–216, 1992.

[Brams and Taylor, 1996] S. J. Brams and A. D. Taylor. *Fair Division: From Cake-Cutting to Dispute Resolution*. Cambridge University Press, 1996.

[Chen *et al.*, 2010] Y. Chen, J. K. Lai, D. C. Parkes, and A. D. Procaccia. Truth, justice, and cake cutting. In *Proc. of 24th AAAI*, pages 756–761, 2010.

[Chevaleyre *et al.*, 2006] Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. Padget, S. Phelps, J. A. Rodríguez-Aguilar, and P. Sousa. Issues in multiagent resource allocation. *Informatica*, 30:3–31, 2006.

[Edmonds and Pruhs, 2006] J. Edmonds and K. Pruhs. Cake cutting really is not a piece of cake. In *Proc. of 17th SODA*, pages 271–278, 2006.

[Procaccia, 2009] A. D. Procaccia. Thou shalt covet thy neighbor's cake. In *Proc. of 21st IJCAI*, pages 239–244, 2009.

[Robertson and Webb, 1998] J. M. Robertson and W. A. Webb. *Cake Cutting Algorithms: Be Fair If You Can*. A. K. Peters, 1998.