

Tutorial: How To Think About Algorithmic Mechanism Design

Tim Roughgarden (Stanford)

Motivation

Optimal auction design: what's the point?

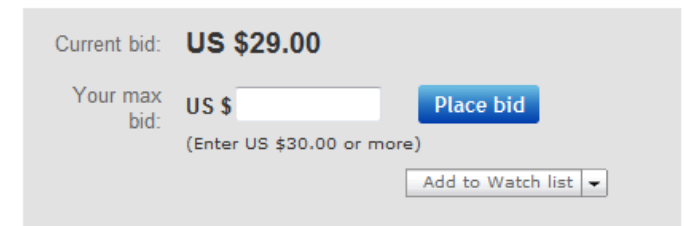
One primary reason: suggests auction formats likely be useful in practice.

Exhibit A: single-item Vickrey auction.

- ❑ maximizes welfare (ex post) [Vickrey 61]
- ❑ with suitable reserve price, maximizes expected revenue with i.i.d. bidder valuations [Meyerson 81]

Review: Vickrey Is Truthful

Utility Model: bidder i has private *valuation* v_i , submits *bid* b_i to max *utility* $= v_i - \text{price}$



Current bid: **US \$29.00**

Your max bid: **Place bid**

(Enter US \$30.00 or more)

Claim: a Vickrey auction is *truthful*

- bidding $b_i = v_i$ always maximizes utility

Proof idea : bidder i effectively faces a "take-it-or-leave it" offer at a fixed price $p = \max\{\text{reserve, highest other bid}\}$.

The Dark Side

Issue: in more complex settings, optimal auctions can say little about how to really solve problem.

Example #1: single-item auction, independent but *non-identical* bidders. to maximize revenue:

- ❑ winner = use highest "virtual bid"
- ❑ charge winner its "threshold bid"

Example #2: combinatorial auctions (max welfare)

- ❑ absurd to implement VCG mechanism, even for modest number of goods

Alternative Approach

Standard Approach: solve for optimal auction over huge set, hope optimal solution is reasonable

Alternative: optimize only over "plausibly implementable" auctions.

Sanity Check: want performance of optimal restricted auction close to that of optimal (unrestricted) auction.

- if so, have theoretically justified and potentially practically useful solution

Algorithmic Mechanism Design

"**Plausibly Implementable**": for this talk, define as *always running in polynomial time*.

Goal #1: Positive Results

- ideal: get starting point for real-world solution
- if not: still get a "possibility proof"
 - and often some useful design techniques

Goal #2: Negative Results

- impossibility results (perhaps assuming $P \neq NP$)

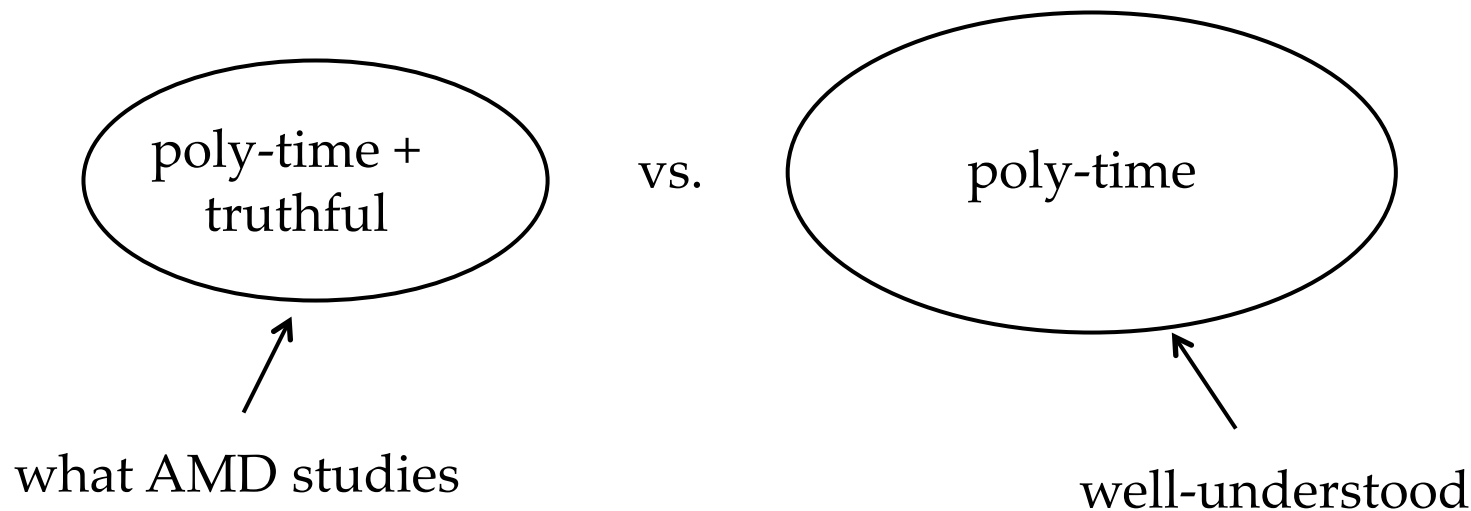
Our Mechanism Design Goals

- runs in polynomial time
 - in input size, or in # of bidders/goods
- (approximately) optimizes a natural objective
 - this talk: focus on welfare-maximization
- restrict to dominant-strategy implementations
 - assumes weakest-possible behavioral model
 - interesting to relax this assumption (eg, Bayes-Nash)

In a nutshell: for what problems can we replicate all of the Vickrey auction's laudable properties?

Our Mechanism Design Goals

- runs in polynomial time
- (approximately) optimizes a natural objective
- restrict to dominant-strategy implementations



How To Think About Algorithmic Mechanism Design

Philosophy: designing truthful mechanisms boils down to designing algorithms in a certain "restricted computational model".

Next: focus on simple class of problems where this point is particularly clear and well understood.

Single-Parameter Problems

Outcome space: a set of vectors of the form
 (x_1, x_2, \dots, x_n) [amount of "stuff" per player]

Utility Model: bidder i has private valuation v_i
(per unit of "stuff")

- utility = $v_i x_i$ - payment
- submits bid b_i to maximize its utility

Examples: k -unit auction, "unit-demand" bidders;
job scheduling on related machines

Mechanism Design Space

The essence of any truthful mechanism
(formalized via the "Revelation Principle"):

- collect bid b_i from each player i
- invoke (randomized) *allocation rule*: b_i 's \longrightarrow x_i 's
 - who gets how much (expected) stuff
- invoke (randomized) *payment rule*: b_i 's \longrightarrow p_i 's
 - and who pays what
- truthfulness: for every i , v_i , other bids, setting $v_i = b_i$ maximizes expected utility $v_i x_i(\mathbf{b}) - p_i(\mathbf{b})$

Two Definitions

Implementable Allocation Rule: is a function x (from bids to expected allocations) that admits a payment rule p such that (x,p) is truthful.

- i.e., truthful bidding $[b_i := v_i]$ always maximizes a bidder's (expected) utility

Two Definitions

Implementable Allocation Rule: is a function x (from bids to expected allocations) that admits a payment rule p such that (x,p) is truthful.

- i.e., truthful bidding $[b_i := v_i]$ always maximizes a bidder's (expected) utility

Monotone Allocation Rule: for every fixed bidder i , fixed other bids b_{-i} , expected allocation only increases in the bid b_i .

- example: highest bidder wins; also sponsored search
- non-example: 2nd-highest bidder wins

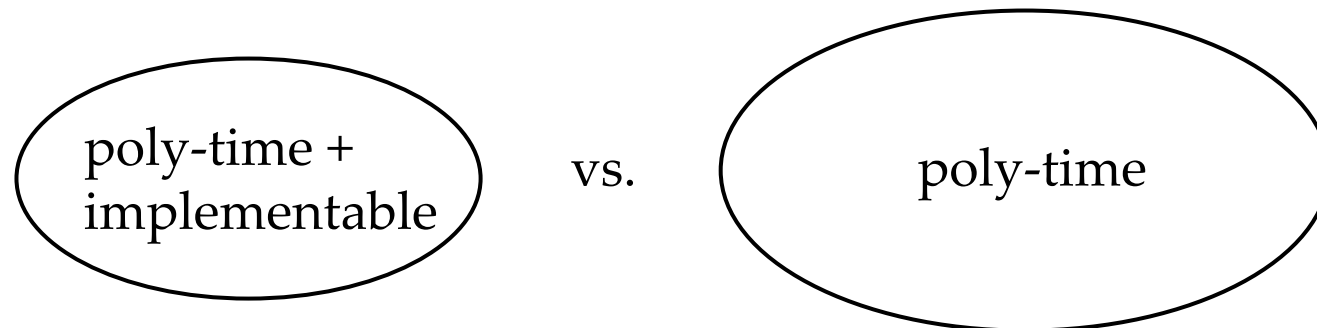
Myerson's Lemma

Myerson's Lemma: [1981; also Archer-Tardos
FOCS 01] *an allocation rule x is implementable if
and only if it is monotone.*

Myerson's Lemma

Myerson's Lemma: [1981; also Archer-Tardos FOCS 01] *an allocation rule x is implementable if and only if it is monotone.*

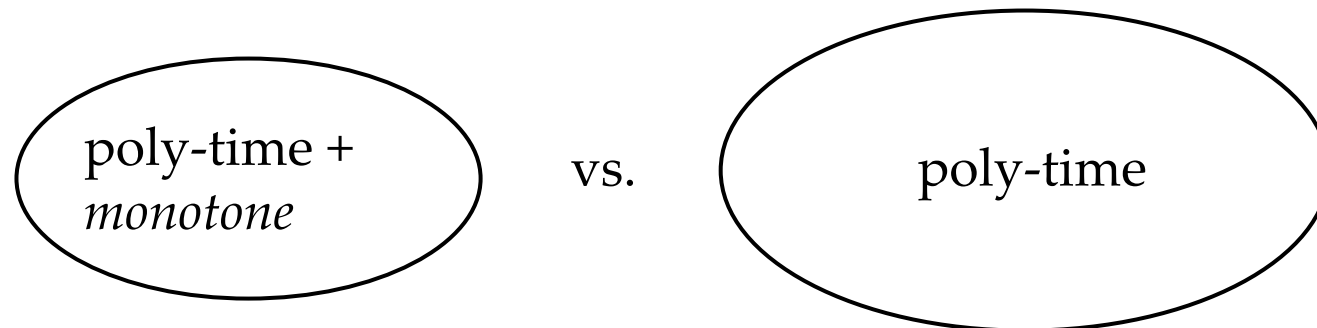
Moreover: for every monotone allocation rule x , gives explicit formula for the unique payment rule p s.t. (x,p) is truthful and losers pay 0.



Myerson's Lemma

Myerson's Lemma: [1981; also Archer-Tardos FOCS 01] *an allocation rule x is implementable if and only if it is monotone.*

Moreover: for every monotone allocation rule x , gives explicit formula for the unique payment rule p s.t. (x,p) is truthful and losers pay 0.



Myerson's Lemma (Proof Idea)

Proof idea: let x be an allocation rule, fix i and b_{-i} .
Write $x(z)$, $p(z)$ for $x_i(z, b_{-i})$, $p_i(z, b_{-i})$.

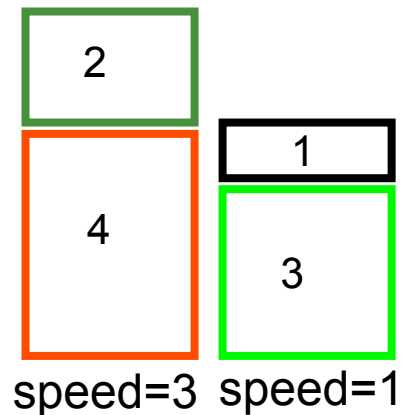
- apply purported truthfulness of (x, p) to two scenarios: true value = z , false bid = $z + \varepsilon$ and true value = $z + \varepsilon$, false bid = z
- take ε to zero get
 - $p'(z) = z \circ x'(z)$ [if x differentiable at z] or
 - jump in p at $z = z \circ [\text{jump in } x \text{ at } z]$

Integrating from 0 to b_i , get sole candidate:

$$p_i(b) := \sum_j y_j \bullet [\text{jump in } x_i \text{ at } y_j]$$

Min Makespan

- m selfish machines with private speeds
 - each wants to min [work - transfer from mechanism]
- n jobs with known sizes
- makespan of a schedule = time last job completes



amount of work = 6 and 4 units

finishing times = 2 and 4

makespan = 4

Min Makespan

Check: an algorithm for min makespan is monotone iff *speeding up a machine can only increase the work assigned to it* (with other speeds fixed).

Example: optimal (exponential-time) algorithm is monotone (with consistent tie-breaking).

Min Makespan

Monotone: *speeding up a machine can only increase the work assigned to it (w/other speeds fixed).*

Known: problem is strongly NP-hard, but PTAS exists [Hochbaum/Shmoys 88], [Epstein/Sgall 04]

But: these PTASes are not monotone.

- do not yield truthful mechanisms!

Theorem: [Archer/Tardos 01; Archer 04] there is a (randomized) monotone 2-approximation.

- key observation: LP relaxation is monotone

A Truthful PTAS

Thm: [Dhangwotnotai/Dobzinski/Dughmi/Roughgarden FOCS 08] There is a (randomized) monotone PTAS for this min makespan problem.

Main Techniques: (to obtain monotonicity)

- multi-step randomized preprocessing (“smoothes” the instance)
- novel compact representation to enable exact poly-time optimization over rich set of near-optimal solution (extends [Epstein/Sgall 04])

Randomized Shuffling

Trick: round jobs to reduce number of distinct sizes.

Problem: two schedules with equal “rounded” load can have (slightly) different “real” load.

- generally leads to monotonicity violations

Fix: for each “bucket” of (near-identical) jobs, make all jobs sizes equal to the average size in bucket.

- at end of algorithm, randomly instantiate each “rounded” jobs by one of original ones
- recovers monotonicity (in expectation)

Combinatorial Auctions (CA)

Setting: n bidders, m goods. Player i has private valuation $v_i(S)$ for each subset S of goods.

Assume: $v_i(\emptyset) = 0$ and v_i is

- *monotone*: $S \text{ subset of } T \Rightarrow v_i(S) \leq v_i(T)$
- *subadditive*: $v_i(S \cup T) \leq v_i(S) + v_i(T)$
- ignore representation issues
[want running time polynomial in n and m]

Facts: there is a poly-time 2-approximation for welfare $\sum_i v_i(S_i)$ [Feige STOC 06]. No good truthful approximation known.

Multi-Parameter Problems

Outcome space: an abstract set Ω

Utility Model: bidder i has private *valuation* $v_i(\omega)$
for each outcome ω

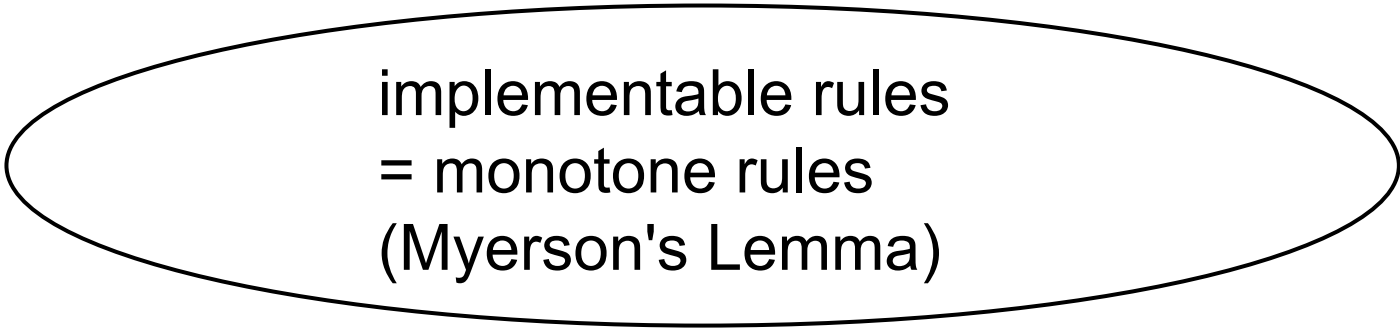
- *utility* = $v_i(\omega)$ - payment

Example: in a combinatorial auction, Ω = all
possible allocations of goods to players

How To Think About Algorithmic Mechanism Design

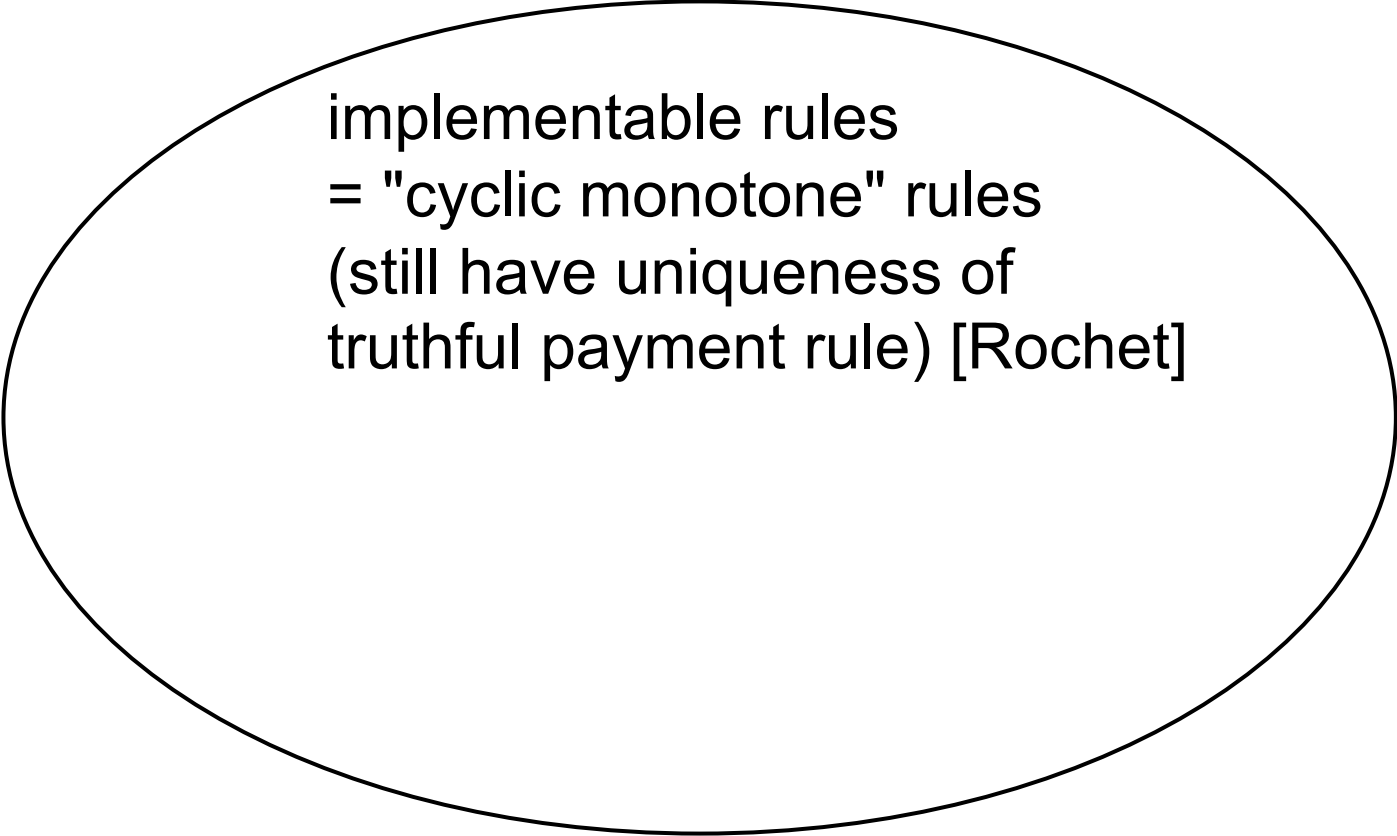
Philosophy: designing truthful mechanisms boils down to designing algorithms in a certain "restricted computational model".

Single-Parameter Special Case:



implementable rules
= monotone rules
(Myerson's Lemma)

The Multi-Parameter World



implementable rules
= "cyclic monotone" rules
(still have uniqueness of
truthful payment rule) [Rochet]

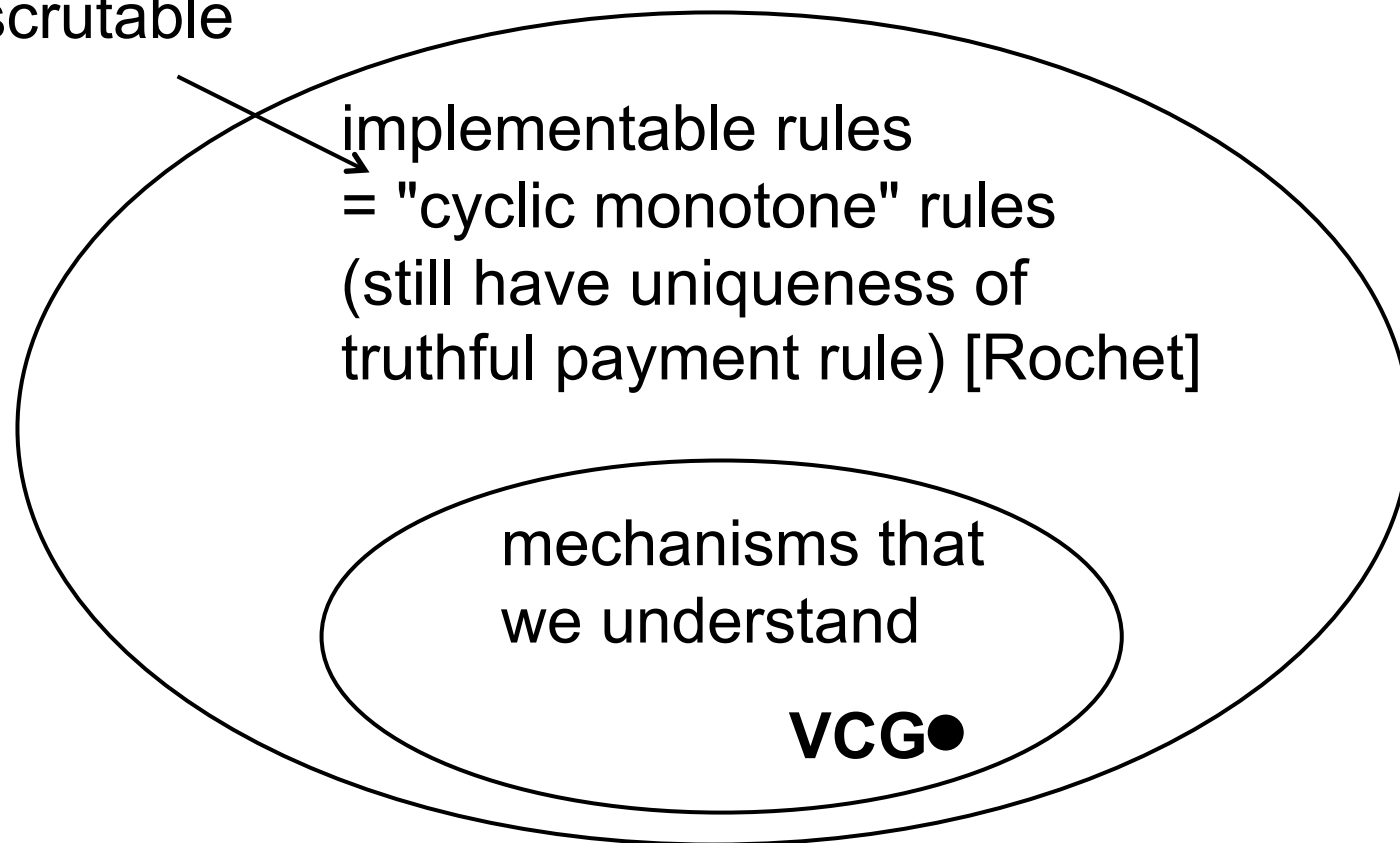
The Multi-Parameter World

inscrutable

implementable rules
= "cyclic monotone" rules
(still have uniqueness of
truthful payment rule) [Rochet]

The Multi-Parameter World

inscrutable



The VCG Mechanism

Utility Model: bidder i 's utility: $v_i(\omega)$ - payment

Vickrey-Clarke-Groves: (1961/71/73)

- collect bid $b_i(\omega)$ for all i , all outcomes ω in Ω
- select ω^* in $\operatorname{argmax} \{\sum_i b_i(\omega)\}$
- charge $p_i = [-\sum_{j \neq i} b_j(\omega)] + \text{suitable constant}$
 - align private objectives with global one

Facts: truthful, maximizes welfare $\sum_i v_i(\omega)$ over Ω (assuming truthful bids).

Approximation Mechanisms

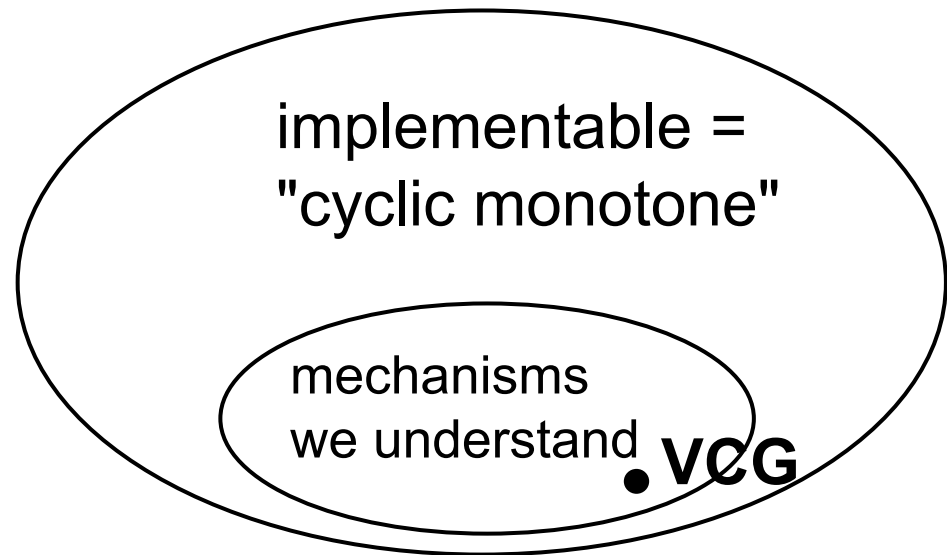
Goals: [Nisan/Ronen 99] (1) truthful; (2) run in time polynomial in natural parameters; and (3) guarantee near-optimal welfare

Best-case scenario: match approximation factor of best polynomial-time approximation algorithm (with valuations given freely as input).

Holy Grail: "black-box reduction" that turns an approximation algorithm into a truthful approximation mechanism.

Approximation Mechanisms

Idea: [Nisan/Ronen 00] use VCG mechanism but substitute approximation algorithm for the previous step "select ω^* in $\operatorname{argmax} \{\sum_i b_i(\omega)\}$ ".

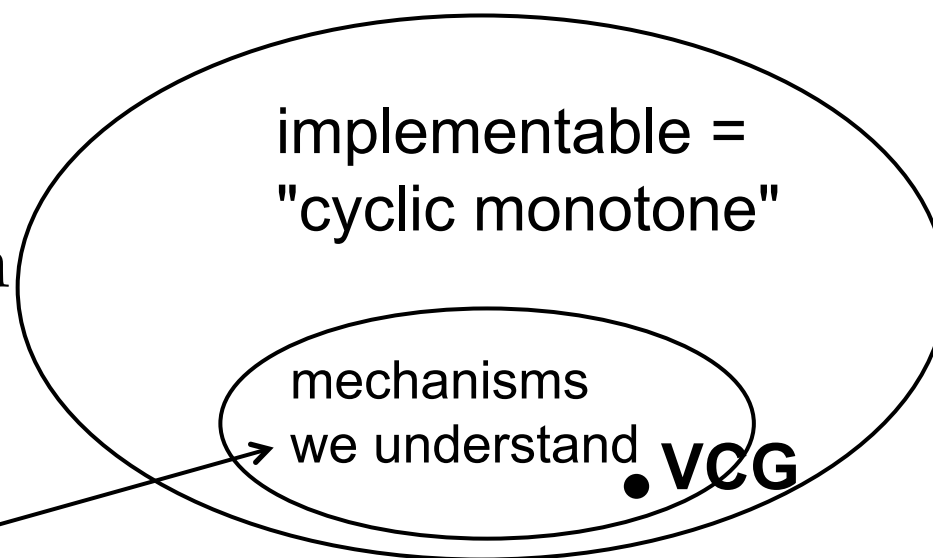


Approximation Mechanisms

Idea: [Nisan/Ronen 00] use VCG mechanism but substitute approximation algorithm for the previous step "select ω^* in $\operatorname{argmax} \{\sum_i b_i(\omega)\}$ ".

Issue: only truthful for a very special type of approximation algorithm (discussed next).

more on
this next



VCG-Based Mechanisms

Outcome space: an abstract set Ω

Utility Model: bidder i 's utility: $v_i(\omega)$ - payment

Step 1: pre-commit to a subset Ω' of Ω

Step 2: run VCG with respect to Ω'

Facts: truthful, maximizes welfare $\sum_i v_i(\omega)$ over Ω'

Hope: can choose Ω' to recover tractability while controlling approximation factor.

Combinatorial Auctions (CA)

Setting: n bidders, m goods. Player i has private valuation $v_i(S)$ for each subset S of goods.

Assume: $v_i(\emptyset) = 0$ and v_i is

- *monotone*: $S \text{ subset of } T \Rightarrow v_i(S) \leq v_i(T)$
- *subadditive*: $v_i(S \cup T) \leq v_i(S) + v_i(T)$
- ignore representation issues
[want running time polynomial in n and m]

Fact: there is a 2-approximation for welfare $\sum_i v_i(S_i)$
[Feige STOC 06], but this allocation rule is
not implementable.

VCG-Based Solution

- Key Claim:** for every instance, there is a $(1/2\sqrt{m})$ -approximate allocation that either:
- assigns all goods to a single player; OR
 - assigns at most one good to each player

VCG-Based Solution

Key Claim: for every instance, there is a $(1/2\sqrt{m})$ -approximate allocation that either:

- assigns all goods to a single player; OR
- assigns at most one good to each player

Corollary: [Dobzinski/Nisan/Schapira STOC 05] there is a truthful $(1/2\sqrt{m})$ -approximate mechanism for CAs with subadditive bidder valuations.

Proof: define Ω' as above; can optimize in poly-time via max-weight matching + case analysis.

VCG-Based Solution

Proof of Key Claim: Fix v_i 's. Call a player *big* if it gets $> \sqrt{m}$ goods in the optimal allocation. (So there are at most \sqrt{m} of them.)

Case 1: big players account for more than half of optimal welfare, so one big player accounts for a $1/2\sqrt{m}$ fraction. Give all goods to this player.

Case 2: otherwise, small players account for half. Give each its favorite good; by subadditivity, still have a $1/2\sqrt{m}$ fraction of optimal welfare.

Can We Do Better?

[Dobzinski/Nisan STOC 07]: Can't do much better using a deterministic VCG-based mechanism.

- results and techniques launched very active research agenda on lower bounds
 - [Papadimitriou/Schapira/Singer FOCS 08], ..., [Dughmi/Vondrak FOCS 11]

The good news: randomized mechanisms seem to hold much promise, for both problems and for black-box reductions.

- can be strictly more powerful than deterministic

Randomized VCG-Based Mechanisms

Step 1: precommit to subset Δ' of $\Delta(\Omega)$

- "lotteries" over outcomes

Step 2: run VCG with respect to Δ'

Facts: truthful (in expectation), maximizes expected welfare $E[\sum_i v_i(\omega)]$ over Δ'

Hope: can choose Δ' to recover tractability while controlling approximation factor.

- [Lavi/Swamy FOCS 05], [Dobzinski/Dughmi FOCS 09]

A Black-Box Reduction

Theorem: [Dughmi/Roughgarden FOCS 10] If a welfare-maximization problem admits an FPTAS, then it admits a truthful FPTAS.

Proof idea: Choosing Δ suitably and "dualizing", the relevant optimization problem is a slightly perturbed version of the original one. Can use techniques from smoothed analysis [Roglin/Teng FOCS 09] to get expected polynomial running time.

Black-Box Reduction for Bayes-Nash Implementations

Theorem: [Hartline/Lucier STOC 10], [Hartline/Kleinberg/Malekian SODA 11], [Bei/Huang SODA 11] In many Bayesian settings (where valuations are drawn from known distributions), *every* approximation algorithm for welfare maximization can be transmuted into an equally good truthful (in Bayes-Nash equilibrium) approximation mechanism.

Suggestive: Bayes-Nash implementations can elude lower bounds for dominant-strategy truthful mechanisms (when such lower bounds exist).

Welfare Guarantees in Combinatorial Auctions with Item Bidding

(Bhawalkar/Roughgarden SODA 2011)

Combinatorial Auctions

- n bidders, m heterogeneous goods
- bidder i has private valuation $v_i(S)$ for each subset S of goods [$\approx 2^m$ parameters]
 - assume nondecreasing with $v_i(\emptyset) = 0$
- quasi-linear utility: player i wants to maximize $v_i(S_i) - \text{payment}$
- allocation = partition of goods amongst bidders
- welfare of allocation S_1, S_2, \dots, S_n : $\sum_i v_i(S_i)$
 - goal is to allocate goods to maximize this quantity

Item Bidding

Note: if m (i.e., # of goods) is not tiny, then direct revelation is absurd.

Goal: good welfare with much smaller bid space.

- "truthfulness" now obviously impossible

Item Bidding: each bidder submits one bid per good. Each good sold via an independent second-price auction.

- only solicit m parameters per bidder
- this auction is already being used! (via eBay)

Item-Bidding Example

Example: two players (1 & 2), two goods (A & B).

Player #1: $v_1(A) = 1$, $v_1(B) = 2$, $v_1(AB) = 2$.

Player #2: $v_2(A) = 2$, $v_2(B) = 1$, $v_2(AB) = 2$.

- OPT welfare = 4.
- A full-information Nash equilibrium:
 - #1 bids 1 on A, 0 on B
 - #2 bids 0 on A, 1 on B
- which has welfare only 2.

The Price of Anarchy

Definition: *price of anarchy (POA)* of a game
(w.r.t. some objective function):

$$\frac{\text{optimal obj fn value}}{\text{equilibrium objective fn value}}$$

the closer to 1
the better

Well-studied goal: when is the POA small?

- benefit of centralized control --- or, a hypothetical perfect (VCG) implementation --- is small
- note POA depends on choice of equilibrium concept
- only recently studied much in auctions/mechanisms

Complement-Free Bidders

Fact: need to restrict valuations to get interesting worst-case guarantees.

Complement-Free Bidder: $v_i(S \cup T) \leq v_i(S) + v_i(T)$
for all subsets S, T of goods.

Complement-Free Bidders

Fact: need to restrict valuations to get interesting worst-case guarantees.

Complement-Free Bidder: $v_i(S \cup T) \leq v_i(S) + v_i(T)$ for all subsets S, T of goods.

Fact : "bluffing" can yield zero-welfare equilibria.

- consider valuations = 1 and 0, bids = 0 and 1

No Overbidding: for all i and S , $\sum_{j \in S} b_{ij} \leq v_i(S)$.

- our bounds degrade gracefully as this is relaxed

Summary of Results

Theorems [Bhawalkar/Roughgarden SODA 11]:

- worst-case POA of pure Nash equilibria = 2
 - when such equilibria exist
 - extends [Christodoulou/Kovacs/Shapira ICALP 08]
- worse-case POA of Bayes-Nash eq $\leq 2 \ln m$
 - also, strictly bigger than 2
 - also hold for mixed, (coarse) correlated Nash eq
 - assumes independent private valuations
 - with correlated valuations, worst-case POA is polynomial in m .

The Complement-Free Case

Lemma 1: for every complement-free valuation v and subset S , there is a bid vector a_i such that:

- $\sum_{j \in S} a_{ij} \geq v_i(S)/(\ln m)$
- $\sum_{j \in T} a_{ij} \leq v_i(T)$ for every subset T of goods

Proof idea: modify primal-dual set cover algorithm and analysis.

Lemma 2: implies POA bound of $2 \ln m$.

Relation to Smoothness

Key Proof Step: invoke Nash equilibrium condition once per player, with "canonical deviation".

- deviation is *independent of* \mathbf{b}_{-i}

Fact: this conforms to the "smoothness paradigm" of [Roughgarden STOC 09]

Corollary: POA bound of $2(\ln m)$ extends automatically to mixed Nash + correlated equilibria, no-regret learners, and Bayes-Nash equilibria with independent private valuations.

Correlated Valuations

Lower Bound: for Bayes-Nash equilibria with correlated valuations, POA can be $m^{1/4}$.

- even for submodular valuations
- approach #1: direct construction
 - based on random planted matchings
- approach #2: [ongoing with Noam Nisan]
communication complexity
 - good POA bounds imply good poly-size "sketches" for the valuations
 - [Balcan/Harvey STOC 11] such sketches do not exist

Open Problems

- is independent Bayes-Nash POA = $O(1)$?
- what about for independent 1st-price auctions?
 - [Hassidim/Kaplan/Mansour/Nisan EC 11]
- when do smoothness bounds extend to POA of Bayes-Nash equilibria with correlated types?
- optimal trade-offs between auction performance and "auction simplicity"
 - e.g., in terms of size of bid space, number of "tunable parameters", etc.

Recap: Mechanism Design as Constrained Algorithm Design

Philosophy: designing truthful mechanisms boils down to designing algorithms in a certain "restricted computational model".

- single-parameter \Leftrightarrow monotone algorithms
- multi-parameter: includes all the obvious VCG variants, but what else?

Research Challenge: usefully characterize the implementable allocation rules for as many multi-parameter problems as possible.

Recap: Welfare Maximization

- ignoring tractability, VCG works even for arbitrary multi-parameter problems
- truthful approximation mechanisms so far mostly restricted to randomized variants of VCG
- but this already enough for some interesting results

Research Challenges:

- better (randomized) approximation mechanisms for combinatorial auctions; or lower bounds
 - more general black-box reductions
 - understanding "simplicity vs. expressiveness" trade-offs
-