Computational Mechanism Analysis:

Towards a "CPLEX for Mechanisms"

Kevin Leyton-Brown

drawing on joint work with Albert Xin Jiang,
David R.M. Thompson and Navin A.R. Bhat

CFEM Inauguration: October 13, 2010

Operations Research Analogy

Contrast mechanism design with another tool used in management science: mathematical programming:

- LP, MIP, QP (...) models of many interesting problems
  - e.g., I can express a vehicle routing problem

Operations Research Analogy

Contrast mechanism design with another tool used in management science: mathematical programming:

- LP, MIP, QP (...) models of many interesting problems
  - e.g., I can express a vehicle routing problem
- Many theoretical tools for analyzing these models
  - the problem is NP-complete
  - identify qualitative characteristics of optimal solutions given information about the instance distribution
  - a given algorithm achieves a constant-factor approximation

Operations Research Analogy

Contrast mechanism design with another tool used in management science: mathematical programming:

- LP, MIP, QP (...) models of many interesting problems
  - e.g., I can express a vehicle routing problem
- Many theoretical tools for analyzing these models
  - the problem is NP-complete
  - identify qualitative characteristics of optimal solutions given information about the instance distribution
  - a given algorithm achieves a constant-factor approximation
- General solvers like CPLEX pick up where theory leaves off
  - identify the solution for a given instance—even if it is "intractable"
  - sample from the instance distribution and give statistics on qualitative properties of the solution (e.g., average distance between stops; balance (across vehicles) of tour length, ...

Mechanism Design

Now consider mechanism design/game theory:

- Expressive models
  - e.g., I can make a game-theoretic model of a sponsored-search auction problem

Mechanism Design

Now consider mechanism design/game theory:

- Expressive models
  - e.g., I can make a game-theoretic model of a sponsored-search auction problem
- Rich theoretical tools can prove lots of useful things
  - a truthful mechanism exists
  - optimal auction for a single slot
  - efficient envy-free equilibria exist in GSP, and weakly revenue-dominate VCG revenue

Mechanism Design

Now consider mechanism design/game theory:

- Expressive models
  - e.g., I can make a game-theoretic model of a sponsored-search auction problem
- Rich theoretical tools can prove lots of useful things
  - a truthful mechanism exists
  - optimal auction for a single slot
  - efficient envy-free equilibria exist in GSP, and weakly revenue-dominate VCG revenue
- Few computational techniques
  - I can simulate agent behavior, but can't compute the same game-theoretic equilibrium concepts that we study theoretically
    - Why? The corresponding games are enormous...

Why Computational Analysis?

- Many mechanisms become important "in the wild" rather than arising through theoretically-driven mechanism design
  - e.g., GFP, GSP for sponsored search
- These mechanisms can have properties (e.g., no dominant strategies, strange tie breaking rules, reserve prices, budgets) that make theoretical analysis difficult or impossible

Goal

Determine economic properties that arise in equilibrium of real-world mechanisms under given valuation distributions

- This also gives us a way of choosing among a set of candidate mechanisms (in this sense, mechanism analysis enables mechanism design)

Advantages and Disadvantages of CMA

Advantages:

- General valuation distribution
  - beyond the simple distributions often needed for analytic methods (e.g., strong monotonicity assumptions about value per click across slots)
- General equilibrium concept
  - beyond e.g., DS; locally envy-free; PSNE
- Can handle reserve prices
- Can answer quantitative questions
  - e.g., what fraction of optimal social welfare?
  - e.g., which auction design achieves higher revenue?

(Potential) drawbacks:

- Results tied to specific valuation distributions
- Discrete (rounding and tie-breaking)

Realizing the Agenda

This may sound like a pretty applied agenda! However, delivering on it requires meeting three (interrelated) theoretical challenges:

- Compact representation languages for describing interesting game-theoretic interactions
- Efficient computational procedures for computing solution concepts of interest given such games
- Encodings of mechanisms of interest into compact representations

By capitalizing on recent theoretical work on equilibrium computation in compactly-represented games and leveraging existing equilibrium-finding algorithms, we can hope to focus on simpler computational problems like expected utility computation.

What we've done so far

- A language for describing complete-information games, suitable for some simple market problems, along with computational procedures for computing solution concepts (mixed/pure/correlated equilibrium) of these games [UAI'04; AAAI'06; AAAI'07; GEB'10]
- Application of these methods to analyzing complete-information ad auctions [EC'09]
- An extension of the language and algorithms to cover temporal settings [UAI'09]
- New extension to Bayesian games [NIPS'10]

I'll give overviews of some of this work, but only at a very high level. I'm happy to go into more detail offline.

Outline

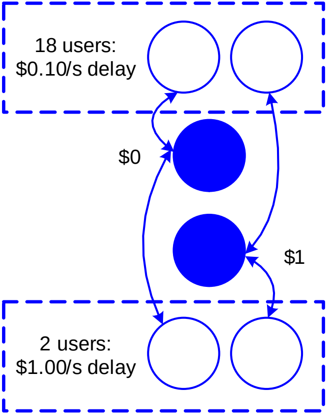
1. Action-Graph Games
2. Position Auctions
3. BAGGs

Action Graph Games [Bhat, L-B, 2004; Jiang, L-B, 2006, 2010]

- A **compact representation** for perfect-information, simultaneous-move games
  - Like Bayes nets or graphical games
    - big table $\rightarrow$ directed graph and small tables
  - Nodes correspond to actions. Table gives utility for playing a given action based on number of agents playing each neighboring action.
- **Representational** savings:
  - Exponentially smaller than NF when in-degree is constant
  - Function nodes (e.g. sum, max) can further reduce size
- **Computational** savings:
  - Expected utility computable in time polynomial in representation size
  - Implies exponential speedup over NF in
    - simplicial subdivision [Scarf, 1967]
    - global Newton method [Govindan, Wilson, 2005]
    - both are implemented in Gambit [McKevley et al, 2006]

An example: "Paris metro pricing" in a network

- Network with one source, one sink, two identical arcs.
  - can be generalized to more interesting graph structures!
- Latency on each edge is a given function of # users
- One arc costs $1, one is free
- Two classes of network users, with different values for latency
- $u(\text{path}) = -\text{toll} - v_i(\text{latency}(\text{path}))$
- Not CG: agents pay differently to consume the same resources
- Not GG: all agents are able to affect each other



Outline

1 Action-Graph Games

2 Position Auctions

3 BAGGs

Types of position auctions

- **GFP**: Yahoo! and Overture 1997–2002
- **uGSP**: Yahoo! 2002–2007
- **wGSP**: Google, Microsoft, Yahoo! 2007–present

Question

Is wGSP better than GFP and uGSP?

- Better by what metric:
  - revenue?
  - efficiency?

Analyzing Position Auctions as Games

- Most existing literature analyzes position auctions as unrepeated, perfect-information interactions
  - unrepeated: probability one user will click on an ad is independent of the probability for the next user
  - perfect info: bidders can probe each others' values
- The literature also proposes bidder valuation models
  - We considered four previously-studied valuation distributions

- Given valuations and a fixed number of bid increments, we have a big normal-form game.
- Problem: it's a really big normal-form game:
  - e.g., 10 bidders, 8 slots, bids in $\{0, 1, \ldots, 40\}$: ~700,000TB

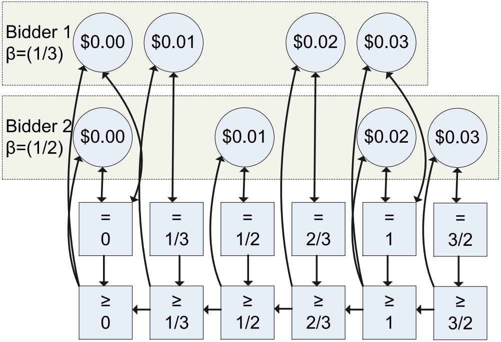
Representing Position Auctions as AGGs

- n bidders, m bid increments
  - nm actions
- Position depends on number of higher/equal bids
  - add 2 sum nodes per action
- GSP price depends on next highest bid
  - add 1 max node per action
- utility tables for each action:
  - GFP: $O(n^2)$ ($\#$ possible tuples from sum nodes)
  - wGSP: $O(n^3m)$ (also includes values of max node, which depends on both per-bidder weight and amount)
- Overall: AGGs are $O(n^4m^2)$, vs NF $O(nm^n)$

Representing Position Auctions as AGGs

- n bidders, m bid increments
  - nm actions
- Position depends on number of higher/equal bids
  - add 2 sum nodes per action
- GSP price depends on next highest bid
  - add 1 max node per action
- utility tables for each action:
  - GFP: $O(n^2)$ ($\#$ possible tuples from sum nodes)
  - wGSP: $O(n^3 m)$ (also includes values of max node, which depends on both per-bidder weight and amount)
- Overall: AGGs are $O(n^4 m^2)$, vs NF $O(nm^n)$
- 10 bidders, 8 slots, bids in $\{0, 1, \ldots, 40\}$
  - NF: ~700,000TB, vs. AGG: <80MB

Example: (Weighted) Generalized First-Price Auction



Simple example: two bidders, four bid amounts per bidder.
Function nodes count the number of bids equal to each weighted bid amount (for tie breaking) and number of bids greater (for winner determination).

Efficiency: what is known theoretically?

Theorem (Edelman, Ostrovsky & Schwarz, 2007; Varian, 2007)

In EOS and V models, wGSP is efficient in every envy-free Nash equilibrium.[1]

Theorem (Paes Leme & Tardos, 2009)

In EOS and V models, wGSP is 1.62-efficient in every conservative Nash equilibrium.

[1] Caveat: these results apply to continuous case without reserve price.

Efficiency: Experimental Questions

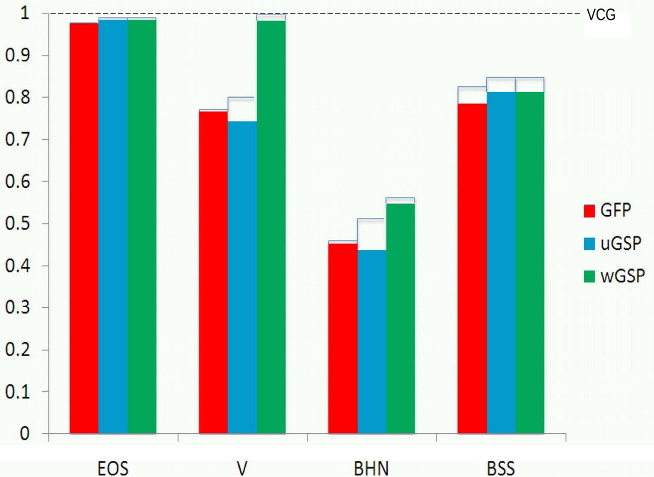
Question

When we go beyond restricted equilibrium families (e.g., envy-free), what happens?

Question

How common are efficiency failures, and how severe are they?

Results: Efficiency



- Broad conclusion: $\{\text{uGSP,GFP}\} \leq^{**} \text{wGSP} \leq^{**} \text{VCG}$

Revenue: Theoretical Predictions and Questions

Theorem (Edelman, Ostrovsky & Schwarz, 2007; Varian, 2007)

In EOS and V models, wGSP generates more revenue than VCG in every "envy-free" Nash equilibrium.

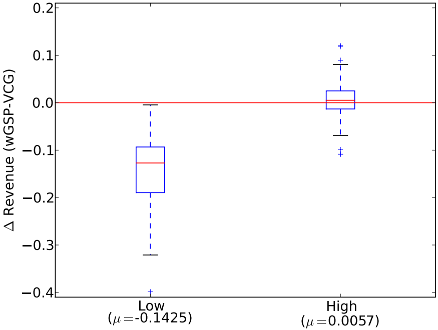
Question

When we go beyond envy-free equilibria, does this result still hold?

Question

How do different auction designs compare in terms of revenue?

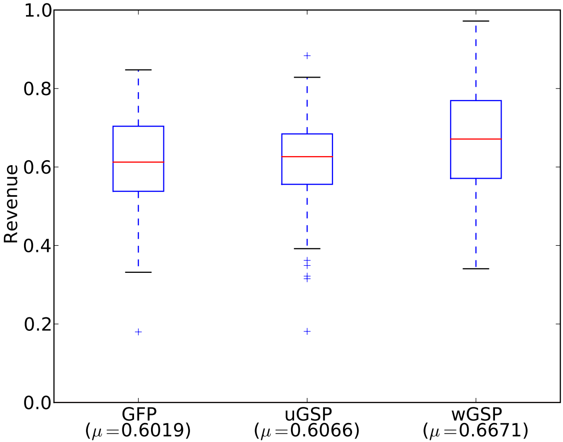
V: revenue range



V: Without envy-free restriction but with restriction to conservative equilibria:

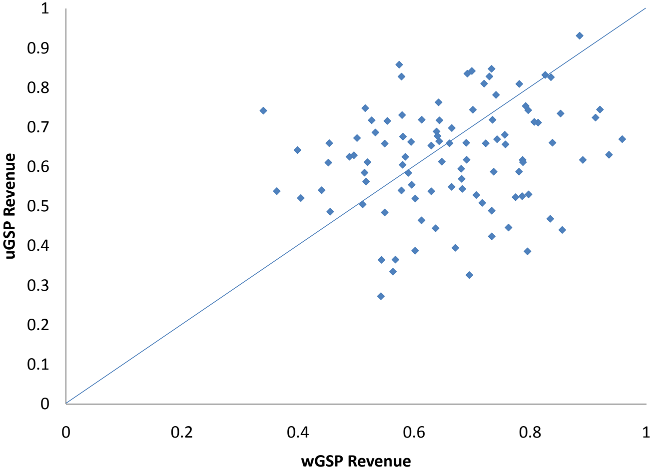
- expected worst wGSP revenue $<^{**}$ expected VCG revenue
- expected best wGSP revenue $>^{**}$ expected VCG revenue

V: best-case revenue



No significant revenue difference between the mechanisms.

V: best-case revenue



No significant revenue difference between the mechanisms.

Outline

1. Action-Graph Games
2. Position Auctions
3. BAGGs

Bayesian Games

- It's desirable to work with Bayesian games as well as with complete-information games
  - so far, a problem that has been largely overlooked/postponed in the computational game theory community
- As far as we know, no general representations or algorithms targeting BNE computation
- This leaves two general approaches, both of which make use of complete-information Nash algorithms:
  - induced normal form
    - one action for each pure strategy (mapping from type to action)
    - set of players unchanged
  - agent form
    - one player for each type of each of the BG's players
    - action space unchanged

Bayesian Action-Graph Games

Idea: construct an AGG-like representation of the Bayesian game's utility functions, which can then compactly encode its agent form.

- Bayesian network for the joint type distribution
- A (potentially separate) action graph for each type of each agent
- A utility function that depends on which types are realized and on the actions taken by the other agents of the appropriate types

Theoretical Results

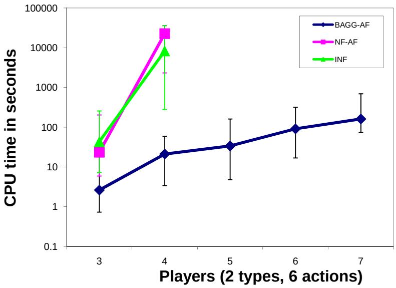
Representational compactness:

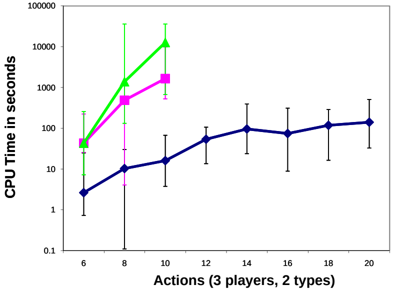
- Representation size grows polynomially in $|\theta|$, $|A|$, n, when action graph has constant-bounded in-degree
  - Exponential savings over an unstructured Bayesian game

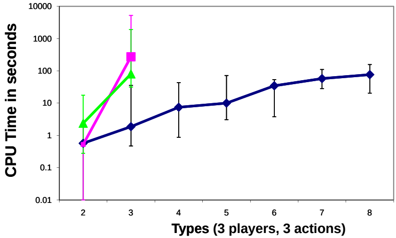
Computational tractability:

- When types are independent, expected utility can be computed in time polynomial in the size of the BAGG.
- When types are not independent, expected utility can still be computed in polynomial time when an induced Bayesian network has bounded treewidth.

Computing BNE with the Govindan–Wilson Algorithm







Conclusions

- Research agenda: computational mechanism analysis
- Requires:
  - General, compact game representations
  - Efficient algorithms for operating on these representations
  - Encodings of interesting problems into these representations
- Many, many open problems along all three dimensions...

- Today I told you about:
  - Representation and algorithms for complete-information games
  - Encoding of ad auctions into this representation
  - Representation and algorithms for Bayesian games