Happy Birthday Joel !!!

Simple randomized algorithms for auction and pricing problems

Avrim Blum

CMU

Joint work with Nina Balcan, Jason Hartline, and Yishay Mansour

[Presented at DIMACS conference in honor of Joel Spencer's 60th birthday]

Plan

A couple problems in intersection of CS and economics with simple randomized algorithms.

Properties:

- About pricing, revenue, etc.
- Inputs to problem given by entities who have their own interest in the outcome of the procedure.

Imagine the following setting…

- Say you are a supermarket trying to decide what price to sell your goods (apples, pop-tarts, detergent, …). Or cell-phone company selling various services.
- Customers have shopping lists. Decide what to buy or whether to shop at all based on prices of items in list.
- Goal: set prices to maximize revenue
  - Simple case: customers make separate decisions on each item based on its own price.
  - Harder case: customers buy everything or nothing based on sum of prices in list.
  - Or could be even more complex.

"Unlimited supply combinatorial auction with additive / single-minded / general bidders"

Three versions (easiest to hardest)

Algorithmic

- Customers' shopping lists / valuations known to the algorithm. (Seller knows market well)

Incentive-compatible auction

- Customers submit lists / valuations to mechanism, which decides who gets what for how much. Must be in customers' interest to report truthfully.

On-line pricing

- Customers arrive one at a time, buy what they want at current prices. Seller modifies prices over time.

Algorithmic problem, single-minded bidders

- You are a supermarket trying to decide what price to sell your goods (apples, pop-tarts, detergent, …). Or cell-phone company selling various services.
- Each customer i has a shopping list L_i and will only shop if the total cost of items in L_i is at most some amount c_i (otherwise he will go elsewhere).

What prices on the items will make you the most money?

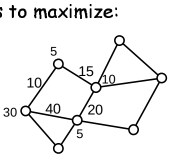
Say all marginal costs to you are 0, and you know all the $\langle L_i, c_i \rangle$ pairs.

- Easy if all L_i are of size 1. (Why?)
- What happens if all L_i are of size 2?

Algorithmic problem, single-minded bidders

- Given a multigraph G with values c_e on the edges e.
- Goal: assign prices $p_v \geq 0$ on vertices to maximize:

$$\sum_{\substack{e=(u,v) \\ p_u + p_v \leq c_e}} p_u + p_v$$

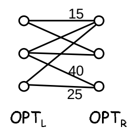


- NP-hard.
- Question 1: can you get a factor 2 approx if G is bipartite?

Algorithmic problem, single-minded bidders

- Given a multigraph G with values c_e on the edges e.
- Goal: assign prices $p_v \geq 0$ on vertices to maximize:

$$\sum_{\substack{e=(u,v)\\ p_u+p_v \leq c_e}} p_u + p_v$$

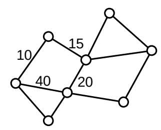


- NP-hard.
- Question 1: can you get a factor 2 approx if G is bipartite? (Set prices on one side to 0, optimize other)
- Question 2: can you get a factor 4 algorithm in general?

Algorithmic problem, single-minded bidders

- Given a multigraph G with values c_e on the edges e.
- Goal: assign prices $p_v \geq 0$ on vertices to maximize:

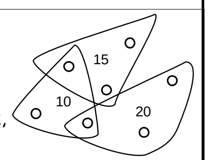
$$\sum_{\substack{e=(u,v)\\ p_u+p_v \leq c_e}} p_u + p_v$$



- NP-hard.
- Question 1: can you get a factor 2 approx if G is bipartite? (Set prices on one side to 0, optimize other)
- Question 2: can you get a factor 4 algorithm in general? (sure, flip a coin for each node to put in L or R)
- Question 3: can you beat this? (We don't know)

Algorithmic problem, single-minded bidders

What about lists of size $\leq k$?



- Get a k-hypergraph problem
- Generalization of previous alg:
  - Put each node in L with prob $1/k$, in R with prob $1 - 1/k$.
  - Let GOOD = set of edges with exactly one endpt in L. Set prices in R to 0, optimize L wrt GOOD.
- Let $OPT_{j,e}$ be revenue OPT makes selling item j to customer e. Let $X_{j,e}$ be indicator RV for $j \in L \wedge e \in GOOD$.
- Our expected profit at least:

$$\mathbf{E}\left[\sum X_{j,e}\text{OPT}_{j,e}\right] = \sum \mathbf{E}\left[X_{j,e}\right]\text{OPT}_{j,e} = O(1/k)\text{OPT}$$

Algorithmic problem, single-minded bidders

Summary:

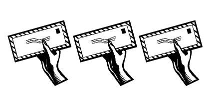
- 4 approx for graph case.
- $O(k)$ approx for k-hypergraph case.
- General $O(\log mn)$ approx by picking the best single price [GHKKKM05].
- $\Omega(\log^{\varepsilon} n)$ hardness for general case [DFHS06].

Incentive-compatible auction problem

Same setup, but we don't know lists or valuations.

Goal: incentive compatible auction

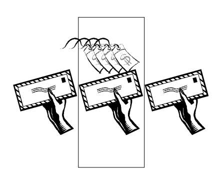
- Customers submit valuation information.
- Auction mechanism determines who buys what for how much.
- Must be in customers' self-interest to submit their true valuations.



Incentive-compatible auction problem

Generic approach to incentive-compatibility

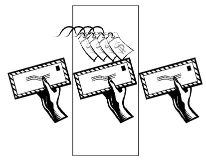
- In the mechanism, each bidder is offered a set of prices that does not depend on what they submitted.
- Mechanism then has them purchase whatever subset has the greatest (valuation – cost).



Incentive-compatible auction problem

Generic approach to incentive-compatibility

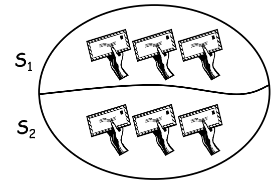
- A lot like a machine-learning problem:
  - Bidders are like examples
  - Preferences/valuations are like labels
  - Goal is to use labels of <u>other</u> examples to "predict" label of current one.



Incentive-compatible auction problem

Simple randomized reduction to alg problem

- Take set S of bids and split randomly into two groups S_1, S_2.
- Run (approx) alg on S_1 to get good item prices for S_1, and use them as offers to bidders in S_2.
- Vice-versa on S_2 to S_1.



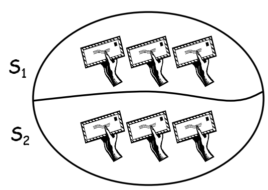
Incentive-compatible auction problem

Guarantee:

- If all valuations are between 1 and h, then $\tilde{O}(hn/\varepsilon^2)$ bidders are sufficient so that whp this loses only factor of $(1+\varepsilon)$ in revenue.
- Analysis idea: not <u>too</u> many sets of prices. Bound each one using McDiarmid tail inequality.

Extensions:

- Pricing functions
- Bound # bidders needed as fn of complexity of class of pricing functions considered.



On-line pricing

Customers arrive one at a time, buy or don't buy at current prices.

- In auction model, we know valuation info for customers 1,…,i-1 when customer i arrives.
- In posted-price model, only know who bought what for how much.
- Goal is to do well compared to best fixed set of item prices.

Fits nicely with setting of online learning in "experts" or "bandit" model.

On-line pricing

Can use approach of [Kalai-Vempala] algorithm, based on [Hannan57]. 

- Hallucinate fake bidders according to appropriate probability distribution.
- Choose optimal prices for combined total (real + imagined) of bidders seen so far.
- Approach works for problems fitting a certain form. In our case, (e.g., for approx. algorithms given in 1st half of talk) can run separate online auctions over items in L, people in GOOD.
- Guarantee: perform comparably to best fixed set of item prices (for pts in L, people in GOOD).

Conclusions & Open problems

- Simple randomized algs achieving factor 4 for graph-vertex pricing problem. Factor O(k) for k-hypergraph vertex pricing.
- Can derandomize (but what's the fun in that!)
- Can then use generic technique to apply in auction setting. Use online learning methods to apply in online setting.

Open Problems:

- $4 - \varepsilon$, $o(k)$.
- How well can you do if negative pricing is allowed (pricing items below cost)?