Approximation Techniques for Utilitarian Mechanism Design

Berthold Vöcking

Department of Computer Science
RWTH Aachen
Germany

joint work with Patrick Briest and Piotr Krysta

05/16/2006

Motivating example: path auctions

Suppose n bidders compete for connections in a network $G = (V, E)$ with capacities $c : E \to \mathbb{R}_+$.

Bidder i ...

- wants to allocate a path between a known source s_i and a known destination t_i
- is single-minded: if she gets a path with a bandwith share matching her demand d_i then her valuation is v_i, else 0.

Objective: Find suitable paths for a subset of bidders $S \subseteq \{1, \ldots, n\}$ such that the capacities are respected and the *social welfare* $\sum_{i \in S} v_i$ is maximized.

Bidders are selfish! How do we convince them to reveal their demands and valuations?

An icentive compatible mechanism ...

- computes suitable paths for a subset of the bidders, and
- charges prices to these bidders in such a way that
- truth-telling is a dominant strategy for all players.

Example: The VCG mechanism ...

- computes the optimal allocation for the specified demands and valuations and
- charges each player the difference between the social welfare of the others and what it would have been without him.

The considered problem is NP-hard.
VCG pricing does not work together with approximation algorithms.
Thus, we need a different concept.

Monotonicity

For an algorithm A, let $S(A(d,v))$ denote the set of served bidders.
We say that A is *monotone* if

$$i \in S(A((d_i, v_i), (d_{-i}, v_{-i}))) \quad \Rightarrow \quad i \in S(A((d_i', v_i'), (d_{-i}, v_{-i})))$$

for any $d_i' \leq d_i$ and $v_i' \geq v_i$.

Critical value pricing: Charge threshold valuations v_i^*.

Lemma: (Lehmann, O'Callaghan, Shoham)

A mechanism is incentive compatible if it is monotone (and exact) and uses critical value pricing.

Intuition:

- Lying about the valuation will not help: If I win, then I anyway pay the smallest value that will win. If the truth will make me loose, then I really don't want to win, since my payment will be higher than my real value.
- Lying about the demand will not help: Increasing the demand can only lower my chances to be served. Decreasing the demand means that I'm not satisfied when I'm served.

Our starting point ...

Unfortunately, standard techniques from the design of approximation algorithms like

- relaxing integrality and randomized rounding of the LP-solution
- distinguishing between entities with large and small parameters
- transforming pseudopolynomial algorithms into approximation schemes

do not guarantee monotonicity and do not give truthful mechanisms.

Def: single-minded multi-unit auctions

- Auctioneer wants to sell $b \in \mathbb{N}$ units of a good.
- There are $n \geq 1$ single-minded bidders.
- Bidder i is interested in buying $a_i \in \mathbb{N}$ units of the good.
- Her valuation for getting this amount is $v_i \in \mathbb{R}_+$, that is, for an assignment $x \in \mathbb{R}_+^n$, her valuation is

$$v_i(a_i, x) = \begin{cases} v_i & \text{if } a_i \leq x_i \\ 0 & \text{otherwise} \end{cases}$$

- **Objective:** find an assignment x that maximizes the social welfare $\sum_i v_i(a_i, x)$.

Underlying optimization problem

The knapsack problem:

Given

- a knapsack of capacity b and
- n objects with weights $a_1, \ldots, a_n$ and values $v_1, \ldots, v_n$
- w.l.o.g., $a_i \leq b$

Compute set $S \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in S} v_i$ is maximized under $\sum_{i \in S} a_i \leq b$.

Complexity of the knapsack problem

KNAPSACK is NP-hard but admits a pseudopolynomial algorithm, which can be used to derive a fully polynomial time approximation scheme (FPTAS).

FPTAS for KNAPSACK

- $\alpha := \dfrac{\varepsilon v_{\max}}{n}$;
- for all i set $v'_i = \alpha \left\lfloor \frac{v_i}{\alpha} \right\rfloor$;
- output optimal solution wrt $v'_1, \ldots, v'_n$.

Unfortunately, this FPTAS is not monotone.

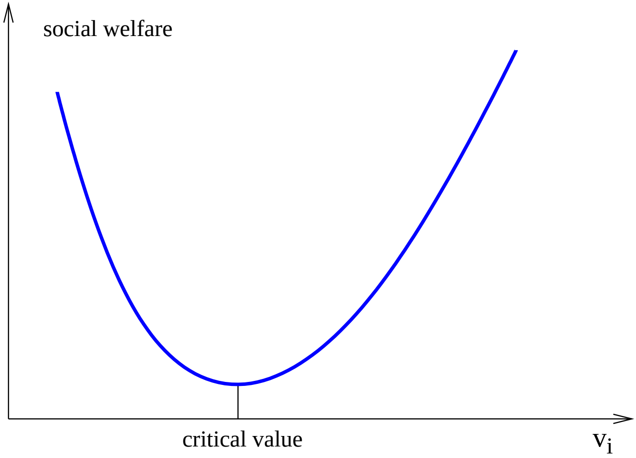
Monotone 2-Approximation by Mu'alem and Nisan

- Let A_1 denote the greedy algorithm for KNAPSACK.
- Let A_2 denote the algorithm returning the single object with highest value.
- Call A_1 and A_2 and output the better solution.

Lemma: (Mu'alem and Nisan)

Taking the maximum over two or more "bitonic" algorithms gives a "bitonic" and, hence, monotone algorithm.

Bitonic Functions



Monotone FPTAS for KNAPSACK

Analytic trick: specify the same algorithm in two different ways.

Output specification

Output is best solution over an infinite number of calls to bitonic algorithms.

Implementation

The solution specified by the output specification can be computed with only $O(\log n)$ calls of the standard FPTAS.

Output specification ensures monotonicity, implementation efficiency.

Output specification

The following algorithm is „called“ for every $k \in \mathbb{Z}$.

Algorithm A(k)

- $M := 2^k$; $\alpha := \dfrac{\varepsilon M}{n}$;
- for all i set $v_i(k) = \min\left\{\alpha \left\lfloor \dfrac{v_i}{\alpha} \right\rfloor, M\right\}$;
- output optimal solution $S(k)$ wrt $v_1(k), \ldots, v_n(k)$.

Output the solution $S(k)$ maximizing $V(k) = \sum_{i \in S(k)} v_i(k)$, breaking ties in favour of smaller indices.

The output specifation is monotone because $A(k)$, $k \in \mathbb{Z}$, is bitonic wrt to $\sum_{i \in S(k)} v_i(k)$.

Efficient implementation

Let $k^* = \lceil \log_2(v_{\max}) \rceil$ and $M^* = 2^{k^*}$ so that $\frac{M^*}{2} < v_{\max} \leq M^*$.

Observations:

- $V(k^*)$ is at least $(1-2\varepsilon) \cdot opt$. Consequently, the output specification guarantees at least a $(1-2\varepsilon)$-approximation.
- If $k > k^*$ then $V(k)$ cannot win against $V(k^*)$ as $V(k)$ only ignores some of the less significant bits that $V(k^*)$ might take into account.
- If $k \leq k^* - \log_2 n - 2$ then $V(k)$ cannot win against $V(k^*)$ as all values are truncated at $M^*/(4n)$ (or smaller) so that $V(k) \leq M^*/4 < V(k^*)$.

$\Rightarrow$ It suffices to call $A(k)$ only for $k \in \{k^* - \log_2 n - 1, \ldots, k^*\}$. $\square$

Further applications

Monotone FPTAS

The same enumeration technique works, e.g., for

- backward auctions
- constrained shortest paths
- scheduling to minimize the weighted number of tardy jobs

Monotone PTAS

A different enumeration technique yields polynomial time approximation schemes for

- multiple knapsack problem with a fixed number of knapsack
- general assignment problem with a fixed number of resources

Combinatorial Auctions (CA)

Combinatorial auction:

- a seller wants to sell a set of items U to n bidders
- the objective is to maximize the *social welfare*, i.e., the sum of the values that the bidders receive

Multi-Unit Combinatorial Auction:

- there are $b_e \geq B$ copies of each item $e \in U$

Assumptions:

- bidders are *single-minded*, i.e., bidder i is interested in a single set $S_i \subseteq U$ which she valuates with v_i
- S_i and v_i are private values

Underlying optimization problem

Set packing

Given $S = (S_1, \ldots, S_n)$ and $v = (v_1, \ldots, v_n)$, find a set $A \subseteq \{1, \ldots, n\}$ that satisfies $|\{i \in A : e \in S_i\}| \leq b_e$, for all $e \in U$, and maximizes $\sum_{i \in A} v_i$.

(Simplifying assumption $B = b_e$, for all $e \in U$.)

The Algorithm

CA-Greedy

Input: Declarations $S_1, \ldots, S_n$, and $v_1, \ldots, v_n$.

$\mathcal{T} = \emptyset$, $\mathcal{S} = \{1, \ldots, n\}$;

for all $e \in U$ **do** $y_e = 1$;

repeat

    $i = \operatorname{argmax} \left\{ \frac{v_i}{\sum_{e \in S_i} y_e} \;\middle|\; i \in \mathcal{S} \setminus \mathcal{T} \right\}$;

    $\mathcal{T} = \mathcal{T} \cup \{i\}$;

    for all $e \in S_i$ **do** $y_e = y_e \cdot em^{1/(B-1)}$;

until $\sum_{e \in U} y_e \geq e^{B-1} m$;

Output: Set of winning bidders $\mathcal{T}$.

Disclaimer: The prices used by this algorithm are not charged to the bidders. For charging the bidders, we use critical value pricing.

Analysis of CA-Greedy

Monotonicity

The selection rule

$$i = \operatorname{argmax}\left\{ \frac{v_i}{\sum_{e \in S_i} y_e} \,\middle|\, i \in \mathcal{S} \setminus \mathcal{T} \right\}$$

ensures monotonicity both wrt the valuations and the sets

Approximation factor

The algorithm guarantees a solution that is within a factor of

$$O\left(m^{1/(B-1)}\right)$$

of the optimal allocation.

Primal-dual interpretation of the greedy algorithm

LP-relaxation of CAs:

$$\begin{array}{rll} \text{max.} & \sum_{i=1}^{n} v_i x_i & \\ \text{s.t.} & \sum_{i:e\in S_i} x_i \le B & \forall e \in U \\ & x_i \le 1 & \forall i \end{array}$$

Dual LP:

$$\begin{array}{rll} \text{min.} & \sum_{e\in U} B y_e + \sum_{i=1}^{n} z_i & \\ \text{s.t.} & z_i + \sum_{e\in S_i} y_e \ge v_i & \forall i \end{array}$$

CA-Greedy: $z_i = v_i$ when agent i is selected.

Primal-dual analysis

Sketch of the analysis:

- In each step, the CA-Greedy picks the set that violates its dual constraint by the largest factor α.
- Scaling all prices by α yields a feasible solution for the dual.
- After termination, the value of the dual wrt α-scaled prices divided by $2m^{1/(B-1)}$ lower-bounds the values collected in $\mathcal{T}$.
- Consequently,

$$v(\mathcal{T}) \geq \frac{\alpha \cdot \sum_{e \in U} B y_e + \sum_{i=1}^{n} z_i}{2m^{1/(B-1)}} \geq \frac{\text{dual-opt}}{2m^{1/(B-1)}} \geq \frac{\text{opt}}{2m^{1/(B-1)}} .$$

$\square$

Path auctions revisited

Suppose n bidders compete for connections in a network $G = (V, E)$ with capacities $c : E \to \mathbb{R}_+$.

Bidder i ...

- wants to allocate a path between a known source s_i and a known destination t_i
- is single-minded: if she gets a path with a bandwith share matching her demand d_i then her valuation is v_i, else 0.

Objective: Find suitable paths for a subset of bidders $S \subseteq \{1, \dots, n\}$ such that the capacities are respected and the *social welfare* $\sum_{i \in S} v_i$ is maximized.

- The optimization problem underlying path auctions is the *unsplittable flow problem (UFP)*.
- Let B denote the ratio between the smallest bandwidth and the smallest demand, m the number of edges.

Known results for UFP

- Randomized rounding achieves an approximation factor of $O(m^{1/\lfloor B+1 \rfloor})$ for UFP (Kolliopoulos and Stein, 1998).
- The best known combinatorial algorithm achieves an approximation factor of $O(B \cdot m^{1/B})$ (Azar and Regev, 2001).
- A monotone variant of Azar and Regev's algorithm achieves an approximation factor of $O(B \cdot m^{1/(B-1)})$

Azar and Regev's algorithm

Presorted-Greedy-UFP

- each edge is assigned a suitable initial price;
- sort the bidders according to a non-increasing order of v_i/d_i;
- in this order, for each bidder,
  - chooses cheapest path wrt current prices;
  - increase the prices for the edges on this path;
- STOP as soon as one of the capacities is exceeded.

Approximation factor: $O(B \cdot m^{1/(B-1)})$

Our Algorithm

Primal-Dual-Greedy-UFP

- each edge is assigned a suitable initial price;
- WHILE capacities are not exceeded DO
    - compute the most efficient path over all bidders, i.e., the path with the best ratio valuation / price;
    - assign this path to the corresponding bidder;
    - increase the prices for the edges on this path.

Approximation factor: $O(m^{1/(B-1)})$

Summary

	previous	greedy primal-dual
CAs single-minded	$O(B \cdot m^{1/(B-2)})$ *Bartal et al.*	$O(m^{1/(B-1)})$
CAs winner determination	$O(m^{1/(B+1)})$ for $(0,1)$-PIPs *Raghavan*	$O(m^{1/B})$
UFP path auctions	$O(B \cdot m^{1/(B-1)})$ *Azar et al.*	$O(m^{1/(B-1)})$