Approximation of Utilitarian Mechanisms

Stefano Leonardi
Sapienza University of Rome

MITACS Workshop on Internet and Network Economics
SFU Harbour Centre, Vancouver
May 30 - June 1, 2011

Outline

- Utilitarian Mechanism Design
- FPTAS for knapsack problems
- FPTAS based on approximate Pareto-optimal solutions
- PTAS based on Lagrangean relaxation
- Monotone primal-dual algorithms
- Conclusions

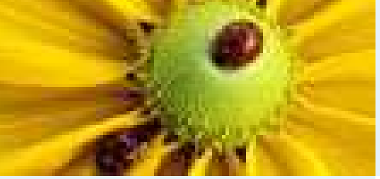
Utilitarian Mechanism Design

Utilitarian Mechanisms

- Suppose n agents/network providers control the edges of a network $G = (V, E)$
- Every edge has cost $c(e)$
- The goal of the algorithm/mechanism is to find a tree that connects all vertices at minimum total cost:

$$\min\{\sum_{e \in E} c(e) x_e \mid x \in \mathcal{X}\}$$

where $\mathcal{X}$ is the set of incidence vectors of the spanning trees.
- Each edge is controlled by a selfish agent with valuation/cost $v(e) = -c(e)$ if selected in the spanning tree.



Utilitarian mechanism design

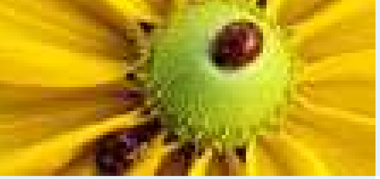
- Agent e can lie by declaring $c'(e) \neq c(e)$.
- A *truthful* mechanism:
  - Compute a Spanning Tree T,
  - Define a payment $p(e)$ for each agent $e \in T$, such that *truth-telling* is a dominant strategy, i.e., it maximizes

$$\texttt{Utility } u(e) = \begin{cases} p(e) - c(e), e \in T \\ 0, e \notin T \end{cases}$$

  whichever strategy is played by the other players.

- The goal of utilitarian mechanism design is to maximize the total utility of the players including the mechanism:

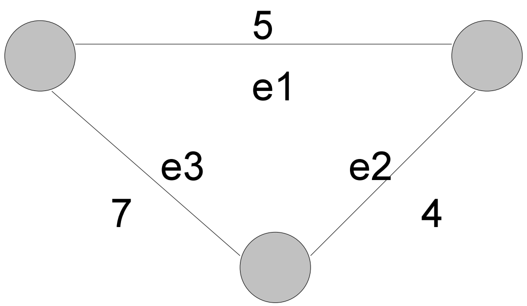
$$\texttt{min} \sum_{e \in E} c(e)$$



VCG mechanism

- Vickrey-Clarke-Groves mechanism is the standard technique for designing truthful utilitarian mechanisms.
- VCG maximizes the social welfare of the players:
  $\texttt{min} \sum_{e \in E} c(e)$
  $\texttt{max} \sum_{e \in E} v(e)$
- VCG requires to solve optimally the underlying optimization problem, e.g., the MST.
- VCG pays agent e the total benefit that the other agents receive from the existence of e.

In the MST instance below: $p(e_1) = c(e_2) = 7$; $p(e_3) = 0$.



Multi-unit Auction

Auction of k identical items $n > k$ bidders.

Player i has valuation v_i for receiving one of the k items

Order bidders by non-increasing valuation $v_1, \ldots, v_n$

- VCG allocation: Allocate k items to the k highest bidders $\{v_1, \ldots, v_k\}$
- Payments: Charge price v_{k+1} to the k highest bidders: Each bidder that receives an item reduces by v_{k+1} the social welfare of the $k+1$ highest bidder.

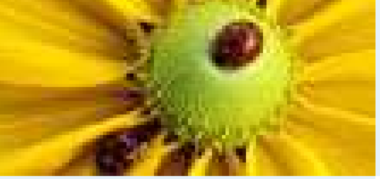
Non-unit demand Bidders

Agent i has valuation v_i for receiving q_i units.

- *Known single minded agents*: only one type of allocation has non-zero valuation (less than q_i has 0 valuation)
- The problem of maximizing the social welfare is the knapsack problem that is NP-hard.
- VCG requires to solve optimally an NP-hard problem.
- VCG pricing does not work with approximation algorithms.

Need an alternative approach:

Monotone Mechanisms [Lehmann, O'Callaghan, Shoham, 2002].



Monotone approximation algorithms

Single-parameter Domain Bidders:

- Le A be the set of all possible allocations.
- $W_i \subseteq A$: winning allocations for bidder i.
- $f(v_i, v_{-i}) \in A$: allocation function of the mechanism.
- Player i decides a scalar value v_i that defines its type.

All incentive compatible mechanisms are defined by the following property of the allocation function:

Definition 1 *An allocation function is monotone if for each v_{-i} and $v_i \leq v_i' \in \Re$, $f(v_i, v_{-i}) \in W_i \longrightarrow f(v_i', v_{-i}) \in W_i$.*

If v_i is a winning declaration then any higher declaration is also a winning declaration.

Payments of Monotone Algorithms

Critical value, i.e., the minimum value of a winning declaration.

Definition 2 *The payment of a winning agent:*

$p_i(v_{-i}) = sup\{v_i : f(v_i, v_{-i}) \notin W_i\}$ *if defined.*

Theorem 3 (Lehmann, O'Callaghan, Shoham 2002) *An approximation algorithm is truthful if it is monotone and uses critical pricing*

Intuition:

- Decreasing v_i has either no effect or may result in i discarded from the solution, if v_i goes below the critical value.
- Increasing v_i has either no effect or it may result in payment p_i higher than true v_i.

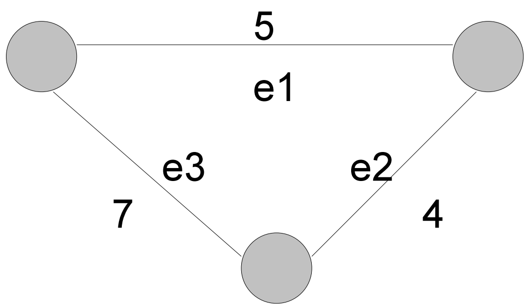
Critical Price: Examples

Multi-unit Auction for Unit-demand Bidders

- The critical price for winning bidders is exactly equal to the $k+1$ highest bid.

Spanning tree auction:

- The VCG allocation and payment scheme is monotone if ties are consistently broken. In the example the critical price is $p(e_1) = p(e_2) = 7$.



Non-unit Demand Bidders

Two monotone algorithms:

- VAL: Sort agents by non increasing valuation
- DENS: Sort agents by non increasing ratio v_i/q_i.

Both algorithms allocate items to players according to a fixed order.

Example: $n = 3$, $m = 4$.

$v_1 = 5$, $q_1 = 2$; $v_2 = 3$, $q_2 = 1$; $v_3 = 4$, $q_3 = 2$.

VAL: players 1and 3 win. Payments: $p_1 = p_3 = 3$, $p_2 = 0$.

DENS: players 1 and 2 win. Payments: $p_1 = 4$, $p_2 = 2$, $p_3 = 0$.

None of the two algorithms is approximated while Max(VAL, DENS) is 2-approximated.

Is the composition of monotone algorithms still monotone? No!

Bitonic algorithms

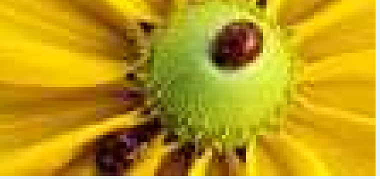
Resort to the notion of bitonic algorithm [Mu'alem and Nisan 2002, Briest, Krysta, Voecking, 05]:

- If agent i wins for v_i then i also wins for any $v_i' \geq v_i$.
- If agent i does not win for v_i then for any $v_i' \geq v_i$ either i wins or *the value of the solution does not improve.*

Theorem 4 *The composition of a set of bitonic algorithms is monotone.*

- Intuition: if a solution from another algorithm is selected when $v_i' \geq v_i$ then this solution should also include player i.

The VAL and DENS algorithms are bitonic.



Monotone FPTAS for Knapsack

There exist an FPTAS for Knapsack: an algorithm that approximates the problem up to an $(1+\epsilon)$ factor in time $\texttt{poly}(1/\epsilon)$.

Round the v_i's to admit only a polynomial number of different valuations and then solve optimally by using a pseudopolynomial time algorithm

- $\alpha = \frac{\epsilon v_{\texttt{max}}}{n}$
- for all i set $v_i' = \alpha \lfloor \frac{v_i}{\alpha} \rfloor$;
- output optimal solution for $v_i', \ldots, v_n'$

Unfortunately the FPTAS for Knapsack is not monotone because the rounding depends from the highest valuation.

It requires to make the the rounding independent from player's declarations.

Monotone FPTAS for Knapsack

- Output the best solution between an infinite set of calls to bitonic algorithms
- The best solution can be computed after $O(\log n)$ calls to the standard FPTAS for knapsack

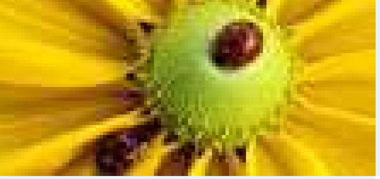
The following algorithm is called for each $k \in \mathcal{Z}$.

Algorithm $\mathtt{A}(\mathtt{k})$:

- $M = 2^k$; $\alpha := \frac{\epsilon M}{n}$;
- for all i set $v_i(k) = \min\{\alpha \lfloor \frac{v_i}{\alpha} \rfloor, M\}$;
- output optimal solution $S(k)$ wrt $v_1(k), \ldots, v_n(k)$ breaking ties in favor of small k.

Output the solution $S(k)$ that maximizes $V(k) = \sum_{i \in S(k)} v_i(k)$

The output specification is monotone because the algorithm $\mathtt{A}(\mathtt{k})$ is bitonic wrt $V(k) = \sum_{i \in S(k)} v_i(k)$.



Monotone FPTAS for Knapsack

Proof of Approximation and number of calls needed.

- Let $k^* = \lceil \log(v_{\mathtt{max}}) \rceil$ and $M^* = 2^{k^*}$ so that $M^*/2 \leq v_{\mathtt{max}} \leq M^*$
- If $k \geq k^*$ then $V(k)$ does not win against $V(k^*)$ since it only ignores some of the less significant bits
- If $k \leq k^* - \log n - 2$ then all values are less or equal than $M^*/4n$ and $V(k) \leq M^*/4$.
- The solution for k^* is at least an $1 - 2\epsilon$ approximation of opt.

Multi Objective Optimization

Motivating Example

- Suppose n agents/network providers control the edges of a network $G = (V, E)$.
- Every edge has cost $c(e)$ and a length/delay $l(e)$.
- The goal of the algorithm/mechanism is to connect all nodes at minimum total cost with bounded total delay:

$$\min\{\sum_{e \in E} c(e)x_e \mid x \in \mathcal{X}, \sum_{e \in E} \ell(e)x_e \leq L\}.$$

  where $\mathcal{X}$ is the set of incidence vectors of the spanning trees.
- Problem 1: Budgeted Minimum spanning tree (BMST) problem is NP-hard. PTASs are known.
- Problem 2: Each edge is controlled by a selfish agent that can declare a higher/lower cost $c(e)$ and/or promise a higher/lower delay $l(e)$.

Multi-parameter domain Agents

Agent e can lie by declaring $(c'(e), \ell'(e)) \neq (c(e), \ell(e))$.

Monotone algorithms:

- If $e \in T$ for $(c(e), l(e))$ then $e \in T'$ and $c(T') \leq c(T)$ for $c'(e) \leq c(e)$ and $l'(e) \leq l(e)$.
- The payment of the algorithm is the critical value $\bar{c}(e)$ for which the agent is selected.

The agents can lie only in one direction on $l(e)$, i.e., they cannot promise less than the minimum delay.

Bitonic algorithms:

- If agent $e \in T$ for $(c(e), l(e))$ then $e \in T'$ and $c(T') \leq c(T)$ whenever $c'(e) \leq c(e)$ and $l'(e) \leq l(e)$.
- If agent $e \notin T$ for $(c(e), l(e))$ then either $e \in T'$ or $c(T') \geq c(T)$ whenever $c'(e) \leq c(e)$ and $l'(e) \leq l(e)$.

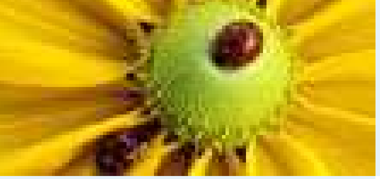
Multi-objective optimization

Optimize on one criteria (in our case the total valuation of the agents) and impose budgets on all other criteria.

$$\texttt{best}\ \{\sum_{e \in \mathcal{U}} \ell_0(e)\, x_e \mid x \in \mathcal{X} \text{ and } \sum_{e \in \mathcal{U}} \ell_i(e)\, x_e \succeq_i B_i \text{ for } i = 1, \ldots, k\}.$$

$\texttt{best}\ \in \{\max, \min\}$, $\succeq_i \in \{\geq, \leq\}$ and k is constant.

- Pareto-optimal solutions are feasible solutions that cannot improved on all objectives. There can be an exponential number of Pareto-optimal solutions.
- Papadimitriou and Yannakakis [2001] show how to compute in poly$(1/\epsilon, n)$ a concise representation of Pareto-nealy optimal curves, i.e., that approximate on all objectives for at most an $(1+\epsilon)$ factor. The condition is that the exact underlying combinatorial problem can be solved in psudopolynomial time.



Monotone FPTAS for Multi-Objective

- Turn the approximate Pareto-optimal construction for the multi-budgeted version of a problem $\mathcal{P}$:

$$\texttt{best}\ \{\sum_{e \in \mathcal{U}} \ell_0(e)\, x_e \mid x \in \mathcal{X} \text{ and } \sum_{e \in \mathcal{U}} \ell_i(e)\, x_e \succeq_i B_i \text{ for } i = 1, \ldots, k\}.$$

  into a monotone multi-criteria FPTAS that violates budget constraints for at most an $1 + \epsilon$ factor if the *exact version* of $\mathcal{P}$ admits a pseudo-polynomial time algorithm.
- Include *multi-budgeted shortest paths* and *spanning trees*, possibly with budget lower bounds.
- Probabilistically truthful for perfect matching since the only pseudopolynomial time algorithm is Monte-Carlo.

Monotone FPTAS for BMST

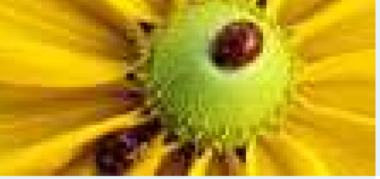
FPTAS for BMST

The BMST problem:

$$\min\{\sum_{e \in E} c(e)x_e \mid x \in \mathcal{X},\ \sum_{e \in E} \ell(e)x_e \leq L\}.$$

- There exists a FPTAS for this problem if there exists a poly-time algorithm for the *gap* problem [Papadimitriou, Yannakakis, 2001]:

  Given a pair (C, L) either returns a solution x with $c(x) \leq C$ and $l(x) \leq L$ or answer that there is no solution x' with $c(x') \leq \frac{C}{1+\epsilon}$ and $l(x') \leq \frac{L}{1+\epsilon}$.
- The gap problem can be solved for all those problems that admit a pseudopolynomial time algorithm that decides whether there exist a solution $x \in \mathcal{X}$ of cost exactly equal to some value M.



The algorithm for the gap problem

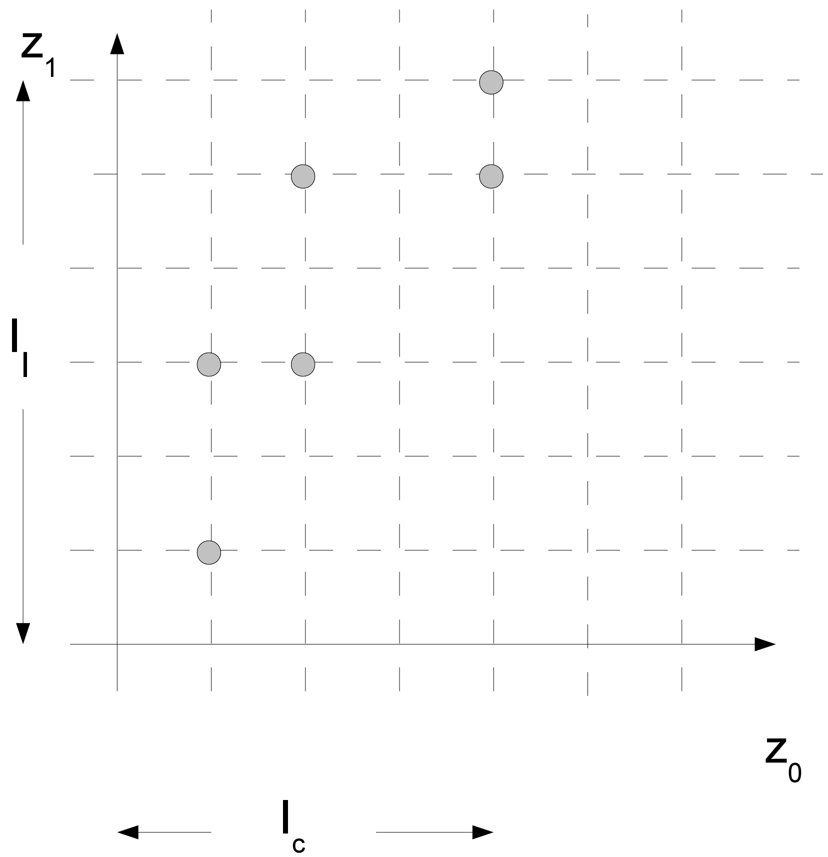
Discretize on both objectives:

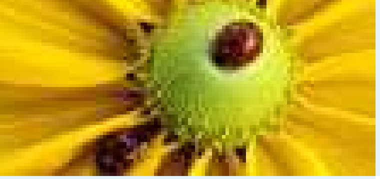
`feasible`$(\mathcal{P}, \epsilon, C)$

- Discard all $e : c(e) > C$ and all $e : l(e) > L$
- Discretize coefficients $c'(e) = \lceil \frac{C'}{C} c(e) \rceil$ and $l'(e) = \lfloor \frac{L'}{L} l(e) \rfloor$.
- $C' = \lceil \frac{m(1+\epsilon)}{\epsilon} \rceil$ and $I_c = \{0, 1, \ldots, C'\}$
- Let $L' = \lceil \frac{m}{\epsilon} \rceil$ and $I_l = \{0, 1, \ldots, L'\}$
- Let $M = m \times \mathtt{max}\{C, L\} + 1$.
- Return $\left(x, \frac{C}{C'} c'(x)\right)$ with best lexicographic z such that $c'(x) + Ml'(x) = z_0 + M \times z_1$, for all $z = (z_0, z_1) \in I_c \times I_l$.

The returned solution is an $(1 + \epsilon)$ approximation of the optimum cost that can be achieved with edges of cost at most L.

The algorithm for the gap problem





Monotone FPTAS for BMST

A family of bitonic algorithms:

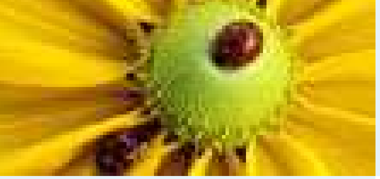
`multi`$(\mathcal{P}, \epsilon)$

- Let $C_j = (1+\epsilon)^j$, $j = 1, \ldots, q$, between $c_{\mathtt{min}} \frac{1}{m(1+\epsilon)}$ and $m\, c_{\mathtt{max}} \lceil \frac{m(1+\epsilon)}{\epsilon} \rceil$.
- For $j = 1, \ldots, q$, let $(S_j, c_j(\cdot)) = \mathtt{feasible}(\mathcal{P}_j, \epsilon, C_j)$.
- Return the solution $S^* = S_h$ optimizing $c_h(S_h)$, the *best* solution with largest index h in case of ties.

The solution is an $(1+\epsilon)^2$ approximation that violates each constraint by at most an $(1+\epsilon)$ factor.

[Grandoni, Krysta, L., Ventre, 2010]

Also apply to Combinatorial Auctions and Multi-Knapsack problems.



Bitonicity

- We need to prove that each single algorithm `feasible`$(\mathcal{P}_j, \epsilon, C_j)$ is bitonic for a value $C_j = 2^j$ with respect to $c_j(\cdot)$. Let $\mathcal{F}$ be the set of feasible solutions for `feasible`$(\mathcal{P}_j, \epsilon, C_j)$.
- Assume $e \in S_j$. Whenever $\bar{c}(e) \leq c(e)$ we get for `feasible`$(\mathcal{P}_j, \epsilon, C_j)$ a set of feasible solution $\mathcal{F} \subseteq \bar{\mathcal{F}}'$. Every solution to $\bar{\mathcal{F}}/\mathcal{F}$ must contain e. We show that $\bar{S}_j \succ S_j$ in the lexicographic order and therefore the solution that is returned on $\bar{c}(\ldots)$ contains e.
- Assume $e \notin S_j$. Each $\bar{\mathcal{F}}/\mathcal{F}$ contains e. If a solution in $\mathcal{F}$ it is reported then $c_j(S) = \bar{c}_j(S)$ and the cost of the solution does not decrease.

Monotone PTAS for BMST

Monotone PTAS for Lagrangean Relaxation

PTAS based on Lagrangean relaxation [Ravi, Goemans, 1996].

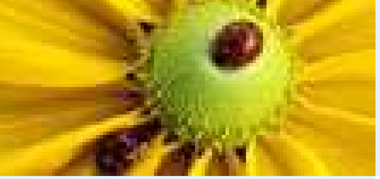
- BMST:

$$\min\{\sum_{e\in E} c(e)x_e \mid x \in \mathcal{X},\ \sum_{e\in E} \ell(e)x_e \leq L\}.$$

- Lagrangean relaxation:

$$LAG(\lambda) = \min\{\sum_{e\in E} c(e)x_e + \lambda \cdot (\sum_{e\in E} \ell(e)x_e - L) \mid x \in \mathcal{X}\}.$$

- It is a MST problem with respect to lagrangean costs $c(e) + \lambda l(e)$.
- Turn the PTAS into a probability distribution over monotone algorithms [Grandoni, Krysta, L., Ventre, 2010]



Lagrangean function

For each $\lambda \geq 0$, $LAG(\lambda) \leq OPT$.

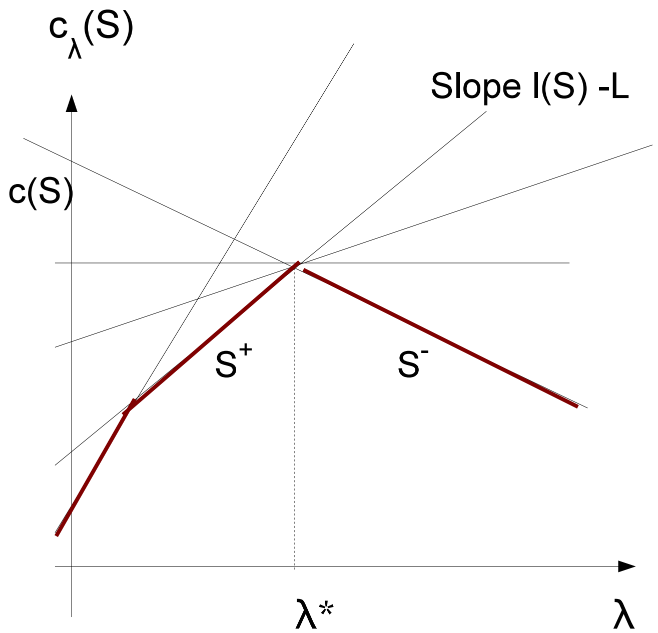
- Let λ^* be a value of $\lambda \geq 0$ which maximizes $LAG(\lambda)$: best lower bound on OPT.
- λ^* can be computed in strongly polynomial time using Megiddo's parametric search.
- Lagrangean cost of solution S is a linear function of λ:

$$\begin{aligned} c_\lambda(S) &:= \sum_{e \in E} c(e)x_e(S) + \lambda(\sum_{e \in E} \ell(e)x_e(S) - L) \\ &= c(S) + \lambda(l(S) - L) \end{aligned}$$

- Slope of $c_\lambda(S)$ is positive if S is infeasible, and non-positive otherwise.
- $c(S) = LAG(\lambda^*) - \lambda^*(l(S) - L)) \leq c(OPT)$ if $l(S) \geq L$.

The lower envelope

The solutions intersecting the lower envelop $LAG(\lambda)$ with decreasing length have increasing cost.



The Lagragean algorithm

Lemma 5 (Ravi, Goemans, 96) *There exists two adjacent solutions in the spanning tree polytope, one is feasible while the other is infeasible.*

- The two solutions differ for one edge.
- We can find these two solutions in polynomial time. We therefore have a solution with optimal cost that has length at most $OPT + c_{max}$.
- We output a solution of non-positive slope that intersects at λ^* a solution of positive slope that is only one edge away.

PTAS for BMST

Enumerate on all subsets of edges of cost larger than ϵC_{OPT}

- At most $\frac{1}{\epsilon}$ such edges: $m^{1/\epsilon}$ different susets.
- For each subset X run the lagrangean algorithm with $L - l(X)$.
- Find the optimal lagrangean solution.
- Two solutions intersecting $LAG(\lambda^*)$ at λ^* are adjacent in the tree polytope, i.e., they differ only by one edge.
- One solution is infeasible with cost smaller than C_{OPT}, the other is feasible.
- Obtain a feasible solution with cost at most $(1 + \epsilon)C_{OPT}$.

Monotonicity

Problems:

- Decreasing the cost of an edge below ϵC_{OPT} may push the edge outside of the final solution.
- There could many, even a non-polynomial number, of adjacent solutions (S^+, S^-).

Solutions:

- Guess for each edge an approximate cost that is used to prune the solution.
- Filtering becomes independent from the real cost: at least one guessing of the cost is close to the actual cost.
- Bitonicity can be ensured by breaking ties in favor of candidate pairs that maximize $c(S^-)$.
- Reduce the number of candidate pairs by perturbing the input instance: w.h.p. no more than two lines intersect at any given point

Monotone PTAS - Subproblem generation

`bmst`$(\mathcal{P}, \epsilon)$

- For all $e \in E$, use cost $c(e)(1 + \epsilon \frac{t_e}{2^m})$ for a random t_e.
- Let $\{c_1, \ldots, c_q\} = \{(1+\epsilon)^i\} : (1+\epsilon)^i \in [c_{min}/(1+\epsilon), c_{max}(1+\epsilon)]$.
- Let $1, \ldots, h$ denote all the pairs $(F, g(\cdot))$ with $F \subseteq E$, $|F| = \frac{1}{\epsilon}$, and $g : F \rightarrow \{c_1, \ldots, c_q\}$.
- Define subproblem $\mathcal{P}_j$ for a given pair $(F_j, g_j(\cdot))$ with budget $L - \ell(F_j)$.
- Remove from G edges of F_j and all the edges of value larger than $\min_{e \in F_j}\{g_j(e)\}$.
- Compute $S_j =$ `lagrangian`$(\mathcal{P}_j)$.
- Return solution $F_j \cup S_j$ minimizing $c(F_j) + c(S_j)$, and maximizing j in case of ties.

Monotone PTAS - Lagrangean problem

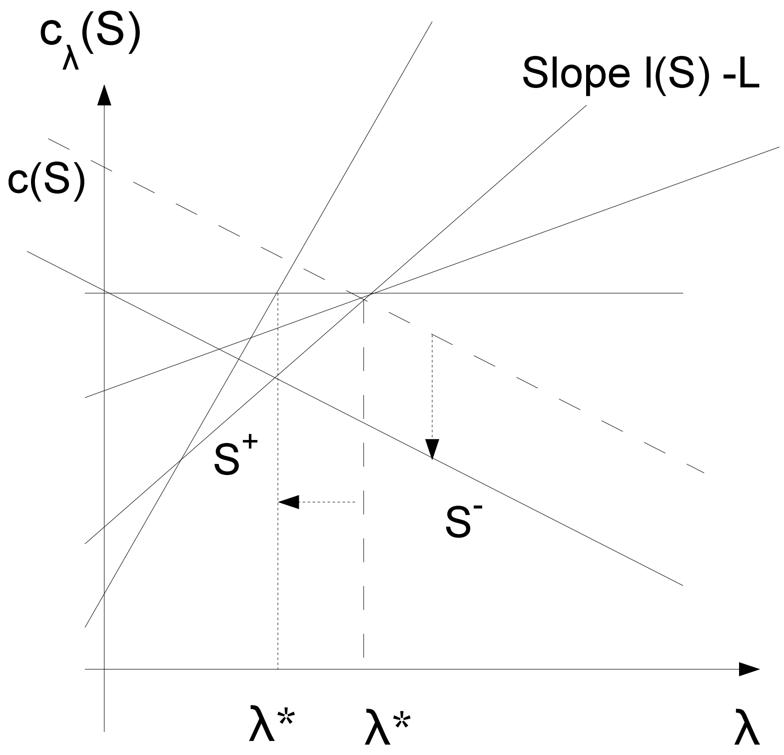
`lagrangian(`$\mathcal{P}_j$`)`

- Compute the optimal Lagrangian multiplier λ^*.
- If $\lambda^* = 0$, return the S^- of minimum-slope (All solutions feasible).
- If $\lambda^* = +\infty$, return $\mathcal{N}$ (No solution feasible).
- Compute a pair of adjacent solutions S^- and S^+.
- Break ties in favor of large $c(S^-)$ and of minimum incidence vector S^-.
- Return S^-.

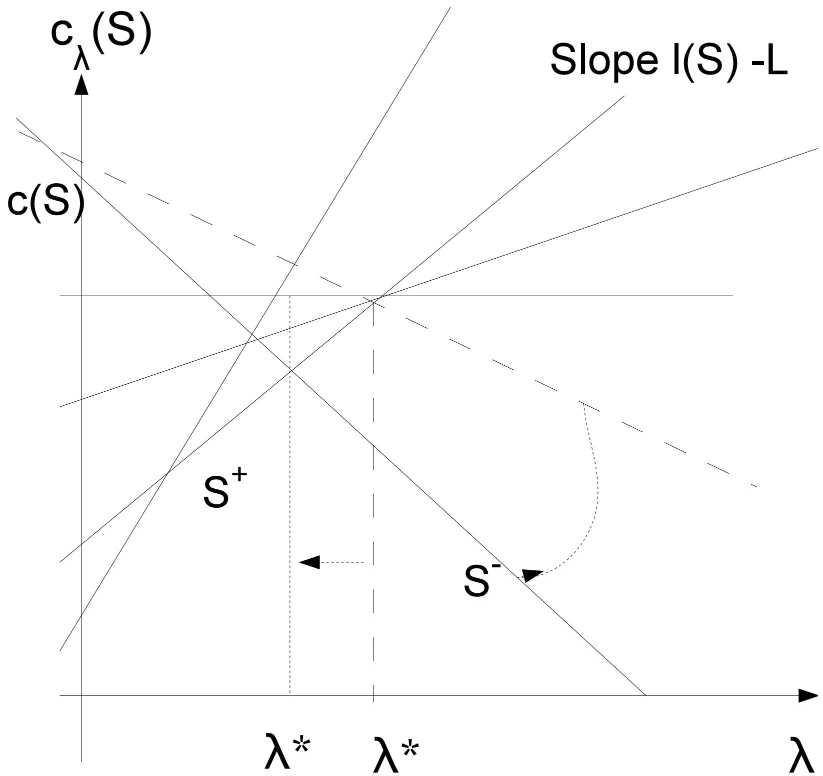
Monotonicity

- Assume agent $f \in S^-$ declares a cost $\bar{c}(f) < c(f)$ or a length $\bar{l}(f) < l(f)$.
- Let $(\lambda, LAG(\lambda))$ the optimal Lagrangean point and S^- the returned solution.
- Reducing $c(e)$ will translate S^- down. Reducing $l(e)$ will rotate S^- to the left.
- It follows $\bar{\lambda}^* \leq \lambda^*$ and all negative slope lines intersecting at $\bar{\lambda}^*$ will also contain f.

Monotonicity

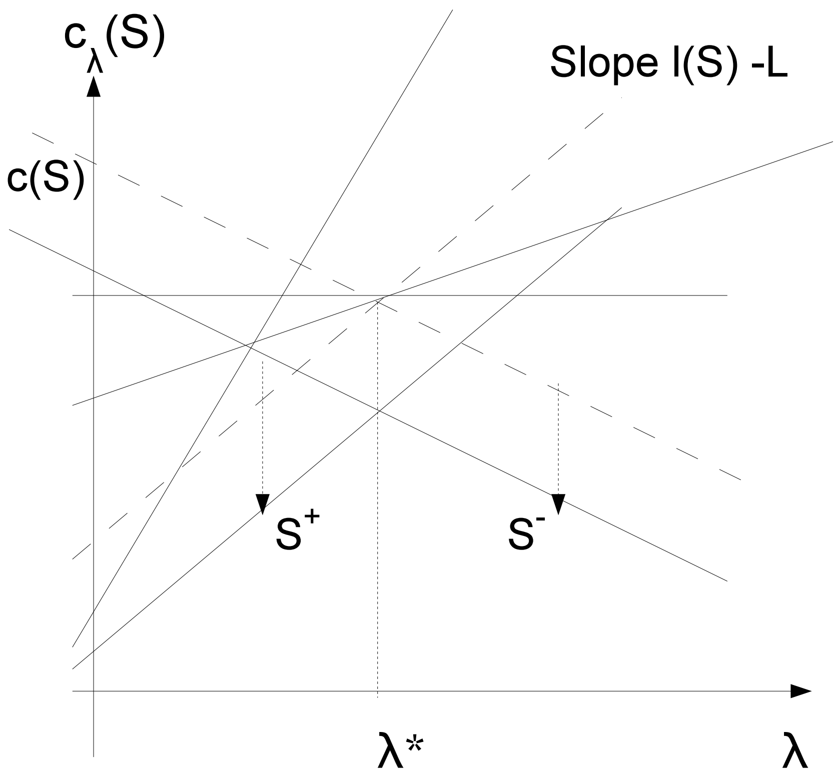


Monotonicity



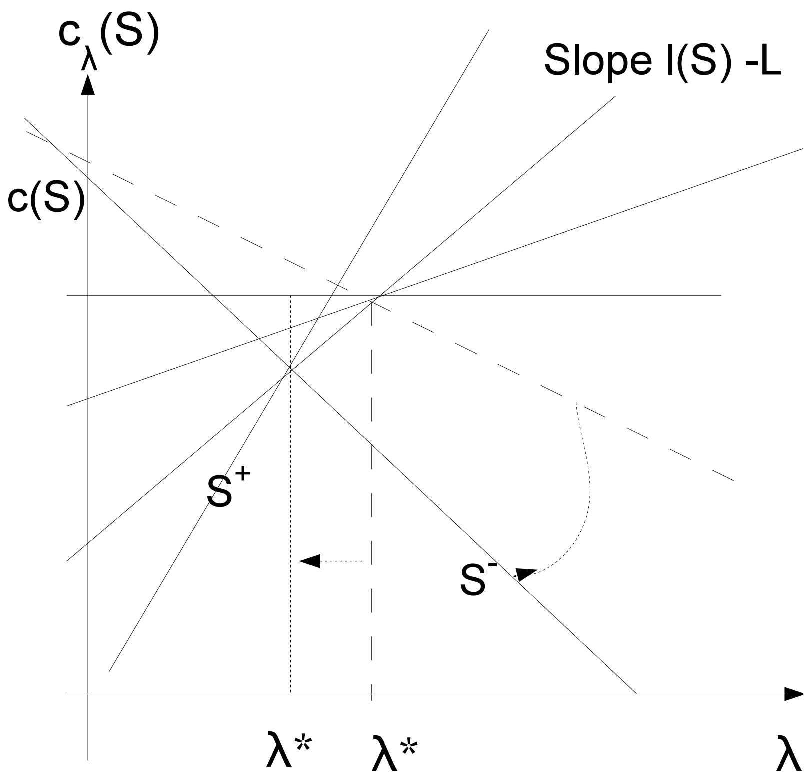
Bitonicity

If $f \in S_j$ and $\bar{\lambda}^* = \lambda$ then all solutions intersecting $(\bar{\lambda}^*, \bar{L}AG(\lambda^*))$ contain f. Moreover $\bar{c}(\bar{S}_j) \leq c(S_j)$.



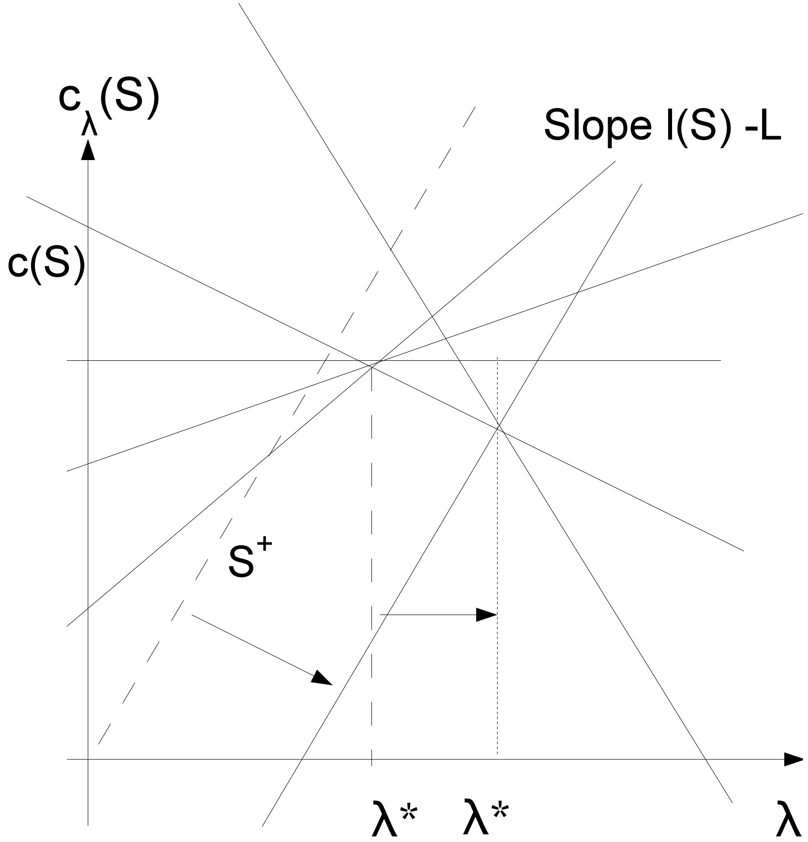
Bitonicity

If $f \in S_j$ and $\bar{\lambda}^* < \lambda$ then there could be some new solution $\bar{S}^+$. But all negative slope solutions that intersect at λ^* contain f and have cost no larger than $c(S_j)$.



Bitonicity

If $f \notin S_j$ then the returned solution $\bar{S}_j$ either contains f or has no lower cost since $\bar{\lambda}^* \geq \lambda^*$.



Conclusions

Conclusions

We show how to adapt basic techniques for designing approximation algorithms to truthful utilitarian mechanisms:

- Combination of algorithms
- FPTAS for Knapsack problems
- FPTAS based on enumeration of approximate Pareto-optimal solutions
- PTAS based on enumeration and Lagrangean relaxation

Many interesting applications and more to come for several interesting and practical problems