U

Undirected Feedback Vertex Set

2005; Dehne, Fellows, Langston, Rosamond, Stevens
2005; Guo, Gramm, Hüffner, Niedermeier, Wernicke

Jiong Guo[1]
Department of Mathematics and Computer Science, University of Jena, Jena, Germany

Keywords and Synonyms

Odd cycle transversal

Problem Definition

The Undirected Feedback Vertex Set (UFVS) problem is defined as follows:

> **Input**: An undirected graph $G = (V, E)$ and an integer $k \geq 0$.
> **Task**: Find a *feedback vertex set* $F \subseteq V$ with $|F| \leq k$ such that each cycle in G contains at least one vertex from F. (The removal of all vertices in F from G results in a forest.)

Karp [11] showed that UFVS is NP-complete. Lund and Yannakakis [12] proved that there exists some constant $\epsilon > 0$ such that it is NP-hard to approximate the optimization version of UFVS to within a factor of $1 + \epsilon$. The best-known polynomial-time approximation algorithm for UFVS has a factor of 2 [1,4]. There is a simple and elegant randomized algorithm due to Becker et al. [3] which solves UFVS in $O(c \cdot 4^k \cdot kn)$ time on an n-vertex and m-edge graph by finding a feedback vertex set of size k with probability at least $1 - (1 - 4^{-k})^{c4^k}$ for an arbitrary constant c. An exact algorithm for UFVS with a running time of $O(1.7548^n)$ was recently found by Fomin et al. [9]. In the context of parameterized complexity [8,13], Bodlaender [5] and Downey and Fellows [7] were the first to show that the problem is *fixed-parameter tractable*, i. e., that the combinatorial explosion when solving it can be confined to the parameter k. The currently best fixed-parameter algorithm for UFVS runs in $O(c^k \cdot mn)$ for a constant c [6,10] (see [6] for the so far best running time analysis leading to a constant $c = 10.567$). This algorithm is the subject of this entry.

Key Results

The $O(c^k \cdot mn)$-time algorithm for the Undirected Feedback Vertex Set is based on the so-called "iterative compression" technique, which was introduced by Reed et al. [14]. The central observation of this technique is quite simple but fruitful: To derive a fixed-parameter algorithm for a minimization problem, it suffices to give a fixed-parameter "compression routine" that, given a size-$(k + 1)$ solution, either proves that there is no size-k solution or constructs one. Starting with a trivial instance and iteratively applying this compression routine a linear number of rounds to larger instances, one obtains a fixed-parameter algorithm of the problem. The main challenge of applying this technique to UFVS lies in showing that there is a fixed-parameter compression routine.

The compression routine from [6,10] works as follows:

1. Consider all possible partitions (X, Y) of the size-$(k + 1)$ feedback vertex set F with $|X| \leq k$ under the assumption that set X is entirely contained in the new size-k feedback vertex set F' and $Y \cap F' = \emptyset$
2. For each partition (X, Y), if the vertices in Y induce cycles, then answer "no" for this partition; otherwise, remove the vertices in X. Moreover, apply the following data reduction rules to the remaining graph:
   - Remove degree-1 vertices.
   - If there is a degree-2 vertex v with two neighbors v_1 and v_2, where $v_1 \notin Y$ or $v_2 \notin Y$, then remove v and connect v_1 and v_2. If this creates two parallel edges

[1]Supported by the Deutsche Forschungsgemeinschaft, Emmy Noether research group PIAF (fixed-parameter algorithms), NI 369/4

between v_1 and v_2, then remove the vertex of v_1 and v_2 that is not in Y and add it to any feedback vertex set for the reduced instance.

Finally, exhaustively examine every vertex set S with size at most $k - |X|$ of the reduced graph as to whether S can be added to X to form a feedback vertex set of the input graph. If there is one such vertex set, then output it together with X as the new size-k feedback vertex set.

The correctness of the compression routine follows from its brute-force nature and the easy to prove correctness of the two data reduction rules. The more involved part is to show that the compression routine runs in $O(c^k \cdot m)$ time: There are 2^k+1 partitions of F into the above sets (X, Y) and one can show that, for each partition, the reduced graph after performing the data reduction rules has at most $d \cdot k$ vertices for a constant d; otherwise, there is no size-k feedback vertex set for this partition. This then gives the $O(c^k \cdot m)$-running time. For more details on the proof of the $d \cdot k$-size bound see [6,10].

Given as input a graph G with vertex set $\{v_1, \ldots, v_n\}$, the fixed-parameter algorithm from [6,10] solves UFVS by iteratively considering the subgraphs $G_i := G[\{v_1, \ldots, v_i\}]$. For $i = 1$, the optimal feedback vertex set is empty. For $i > 1$, assume that an optimal feedback vertex set X_i for G_i is known. Obviously, $X_i \cup \{v_{i+1}\}$ is a solution set for G_{i+1}. Using the compression routine, the algorithm can in $O(c^k \cdot m)$ time either determine that $X_i \cup \{v_{i+1}\}$ is an optimal feedback vertex set for G_{i+1}, or, if not, compute an optimal feedback vertex set for G_{i+1}. For $i = n$, we thus have computed an optimal feedback vertex set for G in $O(c^k \cdot mn)$ time.

Theorem 1 Undirected Feedback Vertex Set *can be solved in $O(c^k \cdot mn)$ time for a constant c.*

Applications

The Undirected Feedback Vertex Set is of fundamental importance in combinatorial optimization. One typical application, for example, appears in the context of combinatorial circuit design [1]. For applications in the areas of constraint satisfaction problems and Bayesian inference, see Bar-Yehuda et al. [2].

Open Problems

It is open to explore the practical performance of the described algorithm. Another research direction is to improve the running time bound given in Theorem 1. Finally, it remains a long-standing open problem whether the Feedback Vertex Set on *directed* graphs is fixed-parameter tractable. The answer to this question would represent a significant breakthrough in the field.

Recommended Reading

1. Bafna, V., Berman, P., Fujito, T.: A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM J. Discret. Math. **3**(2), 289–297 (1999)
2. Bar-Yehuda, R., Geiger, D., Naor, J., Roth, R.M.: Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference. SIAM J. Comput. **27**(4), 942–959 (1998)
3. Becker, A., Bar-Yehuda, R., Geiger, D.: Randomized algorithms for the Loop Cutset problem. J. Artif. Intell. Res. **12**, 219–234 (2000)
4. Becker, A., Geiger, D.: Approximation algorithms for the Loop Cutset problem. In: Proc. 10th Conference on Uncertainty in Artificial Intelligence, pp. 60–68. Morgan Kaufman, San Fransisco (1994)
5. Bodlaender, H.L.: On disjoint cycles. Int. J. Found. Comp. Sci. **5**(1), 59–68 (1994)
6. Dehne, F., Fellows, M.R., Langston, M.A., Rosamond, F., Stevens, K.: An $O(2^{O(k)}n^3)$ FPT algorithm for the undirected feedback vertex set problem. In: Proc. 11th COCOON. LNCS, vol. 3595, pp. 859–869. Springer, Berlin (2005). Long version to appear in: J. Discret. Algorithms
7. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness. Congres. Numerant. **87**, 161–187 (1992)
8. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Heidelberg (1999)
9. Fomin, F.V., Gaspers, S., Pyatkin, A.V.: Finding a minimum feedback vertex set in time $O(1.7548^n)$. In: Proc. 2th IWPEC. LNCS, vol. 4196, pp. 184–191. Springer, Berlin (2006)
10. Guo, J., Gramm, J., Hüffner, F., Niedermeier, R., Wernicke, S.: Compression-based fixed-parameter algorithms for Feedback Vertex Set and Edge Bipartization. J. Comp. Syst. Sci. **72**(8), 1386–1396 (2006)
11. Karp, R.: Reducibility among combinatorial problems. In: Miller, R., Thatcher, J. (eds.) Complexity of Computer Computations, pp. 85–103. Plenum Press, New York (1972)
12. Lund, C., Yannakakis, M.: The approximation of maximum subgraph problems. In: Proc. 20th ICALP. LNCS, vol. 700, pp. 40–51. Springer, Berlin (1993)
13. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press, Oxford (2006)
14. Reed, B., Smith, K., Vetta, A.: Finding odd cycle transversals. Oper. Res. Lett. **32**(4), 299–301 (2004)

Unified Energy-Efficient Unicast and Broadcast Topology Control

► Degree-Bounded Planar Spanner with Low Weight

University Admissions Problem

► Hospitals/Residents Problem

Using Visualization in the Empirical Assessment of Algorithms

► Visualization Techniques for Algorithm Engineering

Utilitarian Mechanism Design for Single-Minded Agents

2005; Briest, Krysta, Vöcking

Piotr Krysta[1], Berthold Vöcking[2]
[1] Department of Computer Science, University of Liverpool, Liverpool, UK
[2] Department of Computer Science, RWTH Aachen University, Aachen, Germany

Keywords and Synonyms

Forward (combinatorial, multi-unit) auction

Problem Definition

This problem deals with the design of efficiently computable incentive compatible, or truthful, mechanisms for combinatorial optimization problems with selfish one-parameter agents and a single seller. The focus is on approximation algorithms for NP-hard mechanism design problems. These algorithms need to satisfy certain monotonicity properties to ensure truthfulness.

A *one parameter agent* is an agent who as her private data has some *resource* as well as a *valuation*, i. e., the maximum amount of money she is willing to pay for this resource. Sometimes, however, the resource is assumed to be known to the mechanism. The scenario where a single seller offers these resources to the agents is primarily considered. Typically, the seller aims at maximizing the social welfare or her revenue. The work by Briest, Krysta and Vöcking [6] will mostly be considered, but also other existing models and results will be surveyed.

Utilitarian Mechanism Design

A famous example of mechanism design problems is given by combinatorial auctions (CAs), in which a single seller, auctioneer, wants to sell a collection of goods to potential buyers. A wider class of problems is encompassed by a *utilitarian mechanism design (maximization) problem* Π defined by a finite set of *objects* $\mathcal{A}$, a set of feasible outputs $O_\Pi \subseteq \mathcal{A}^n$ and a set of n agents. Each agent declares a set of objects $S_i \subseteq \mathcal{A}$ and a valuation function $v_i : \mathcal{P}(\mathcal{A}) \times \mathcal{A}^n \to \mathbb{R}$ by which she values all possible outputs. Given a vector $S = (S_1, \ldots, S_n)$ of declarations one is interested in output $o^* \in O_\Pi$ maximizing the *social welfare*, i. e., $o^* \in \operatorname{argmax}_{o \in O_\Pi} \sum_{i=1}^n v_i(S_i, o)$. In CAs, an object a corresponds to a subset of goods. Each agent declares all the subsets she is interested in and the prices she would be willing to pay. An output specifies the sets to be allocated to the agents.

Here, a limited type of agents called *single-minded* is considered, introduced by Lehmann et al. [10]. Let $R_\preceq \subseteq \mathcal{A}^2$ be a reflexive and transitive relation on $\mathcal{A}$, such that there exists a special object $\varnothing \in \mathcal{A}$ with $\varnothing \preceq a$ for any $a \in \mathcal{A}$ to model the situation in which some agent does not contribute to the solution at all. For $a, b \in \mathcal{A}$ $(a, b) \in R_\preceq$ will be denoted by $a \preceq b$. The single-minded agent i declares a *single* object a_i and is fully defined by her *type* (a_i, v_i), with $a_i \in \mathcal{A}$ and $v_i > 0$. The valuation function introduced earlier reduces to

$$v_i(a_i, o) = \begin{cases} v_i\,, & \text{if } a_i \preceq o_i \\ 0\,, & \text{else}\,. \end{cases}$$

Agent i is called *known* if object a_i is known to the mechanism [11]. Here, mostly unknown agents will be considered. Intuitively, each a_i corresponds to an object agent i offers to contribute to the solution, v_i describes her valuation of any output o that indeed selects a_i. In CAs, relation $R_\preceq$ is set inclusion: an agent interested in set S will is also satisfied by S' with $S \subseteq S'$. For ease of notation let $(a, v) = ((a_1, v_1), \ldots, (a_n, v_n))$, $(a_{-i}, v_{-i}) = ((a_1, v_1), \ldots, (a_{i-1}, v_{i-1}), (a_{i+1}, v_{i+1}), \ldots, (a_n, v_n))$ and $((a_i, v_i), (a_{-i}, v_{-i})) = (a, v)$.

Mechanism

A *mechanism* $M = (A, p)$ consists of an algorithm A computing a solution $A(a, v) \in O_\Pi$ and an n-tuple $p(a, v) = (p_1(a, v), \ldots, p_n(a, v)) \in \mathbb{R}_+^n$ of payments collected from the agents. If $a_i \preceq A(a, v)_i$, agent i is *selected*, and let $S(A(a, v)) = \{i | a_i \preceq A(a, v)_i\}$ be the set of selected agents. Agent i's type is her private knowledge. Thus, the types declared by agents may not match their true types. To reflect this, let (a_i^*, v_i^*) refer to agent i's true type and (a_i, v_i) be the declared type. Given an output $o \in O_\Pi$, the *utility* of agent i is $u_i(a, v) = v_i(a_i^*, o) - p_i(a, v)$. Each agent's goal is to maximize her utility. To achieve this, she will try to manipulate the mechanism by declaring a false type if this could result in higher utility. A mechanism is called *truthful*, or *incentive compatible*, if no agent i can gain by lying about her type, i. e., given declarations (a_{-i}, v_{-i}), $u_i((a_i^*, v_i^*), (a_{-i}, v_{-i})) \geq u_i((a_i, v_i), (a_{-i}, v_{-i}))$ for any $(a_i, v_i) \neq (a_i^*, v_i^*)$.

Algorithm A^k_{Π}:
1 $\alpha_k := \frac{n}{\varepsilon \cdot 2^k}$;
2 **for** $i = 1, \ldots, n$ **do**
3 $\quad v'_i := \min\{v_i, 2^{k+1}\}$;
4 $\quad v''_i := \lfloor \alpha_k \cdot v'_i \rfloor$;
5 return $A_{\Pi}(a, v'')$;

Algorithm A^{FPTAS}_{Π}
1 $V := \max_i v_i$, $Best := (\emptyset, \ldots, \emptyset)$, $best := 0$;
2 **for** $j = 0, \ldots, \lceil \log(1-\varepsilon)^{-1} n \rceil + 1$ **do**
3 $\quad k := \lfloor \log(V) \rfloor - j$;
4 $\quad$ **if** $w_k(A^k_{\Pi}(a, v)) > best$ **then**
5 $\quad\quad Best := A^k_{\Pi}(a, v)$; $best := w_k(A^k_{\Pi}(a, v))$;
6 return $Best$;

Utilitarian Mechanism Design for Single-Minded Agents, Figure 1
A monotone FPTAS for utilitarian problem Π and single-minded agents

Monotonicity

A sufficient condition for truthfulness of approximate mechanisms for single-minded CAs was first given by Lehmann et al. [10]. Their results can be adopted for the considered scenario. An algorithm A is *monotone* with respect to $R_{\preceq}$ if

$$\begin{aligned} & i \in S(A((a_i, v_i), (a_{-i}, v_{-i}))) \\ \Rightarrow\ & i \in S(A((a'_i, v'_i), (a_{-i}, v_{-i}))) \end{aligned}$$

for any $a'_i \preceq a_i$ and $v'_i \geq v_i$. Intuitively, one requires that a winning declaration (a_i, v_i) remains winning if an object a'_i, smaller according to $R_{\preceq}$, and a higher valuation v'_i are declared. If declarations (a_{-i}, v_{-i}) are fixed and object a_i declared by i, algorithm A defines a *critical value* θ^A_i, i. e., the minimum valuation v_i that makes (a_i, v_i) winning, i. e., $i \in S(A((a_i, v_i), (a_{-i}, v_{-i})))$ for any $v_i > \theta^A_i$ and $i \notin S(A((a_i, v_i), (a_{-i}, v_{-i})))$ for any $v_i < \theta^A_i$. The *critical value payment scheme* p^A associated with A is defined by $p^A_i(a, v) = \theta^A_i$, if $i \in S(A(a, v))$, and $p^A_i(a, v) = 0$, otherwise. The critical value for any fixed agent i can be computed, e. g., by performing binary search on interval $[0, v_i]$ and repeatedly running algorithm A to check if i is selected. Also, mechanism $M_A = (A, p^A)$ is *normalized*, i. e., agents that are not selected pay 0. Algorithm A is *exact*, if for declarations (a, v), $A(a, v)_i = a_i$ or $A(a, v)_i = \varnothing$ for all i. In analogy to [10] one obtains the following.

Theorem 1 *Let A be a monotone and exact algorithm for some utilitarian problem Π and single-minded agents. Then mechanism $M_A = (A, p^A)$ is truthful.*

Additional definitions

In the *unsplittable flow problem* (UFP), an undirected graph $G = (V, E)$, $|E| = m$, $|V| = n$, with edge capacities b_e, $e \in E$, and a set K of $k \geq 1$ commodities described by terminal pairs $(s_i, t_i) \in V \times V$ and a demand d_i and a value c_i are given. One assumes that $\max_i d_i \leq \min_e b_e$, $d_i \in [0, 1]$ for each $i \in K = \{1, \ldots, k\}$, and $b_e \geq 1$ for all $e \in E$. Let $B = \min_e\{b_e\}$. A feasible solution is a subset $K' \subseteq K$ and a single flow s_i-t_i-path for each $i \in K'$, such that the demands of K' can simultaneously and unsplittably be routed along the paths and the capacities are not exceeded. The goal in *UFP*, called *B-bounded UFP*, is to maximize the total value of the commodities in K'. A generalization is allocating bandwidth for multicast communication, where commodity is a set of terminals that should be connected by a multicast tree.

Key Results

Monotone approximation schemes

Let Π be a given utilitarian (maximization) problem. Given declarations (a, v), let $Opt(a, v)$ denote an optimal solution to Π on this instance and $w(Opt(a, v))$ the corresponding social welfare. Assuming that A_{Π} is a pseudopolynomial exact algorithm for Π an algorithm A^k_{Π} and monotone FPTAS for Π is defined in Fig. 1.

Theorem 2 *Let Π be a utilitarian mechanism design problem among single-minded agents, A_{Π} monotone pseudopolynomial algorithm for Π with running time $poly(n, V)$, where $V = \max_i v_i$, and assume that $V \leq w(Opt(a, v))$ for declaration (a, v). Then A^{FPTAS}_{Π} is a monotone FPTAS for Π.*

Theorem 2 can also be applied to minimization problems. Section "Applications" describes how these approximation schemes can be used for forward multi-unit auctions and job scheduling with deadlines.

Truthful primal-dual mechanisms

For an instance $G = (V, E)$ of UFP defined above, let S_i be the set of all s_i-t_i-paths in G, and $S = \bigcup_{i=1}^k S_i$. Given $S \in S_i$, let $q_S(e) = d_i$ if $e \in S$, and $q_S(e) = 0$ otherwise.

Algorithm Greedy-1:

1 $\mathcal{T} := \emptyset$; $K := \{1, \ldots, k\}$;
2 **forall** $e \in E$ **do** $y_e := 1/b_e$;
3 **repeat**
4 **forall** $i \in K$ **do** $S_i := \text{argmin}\left\{\sum_{e \in S} y_e \;\middle|\; S \in S_i\right\}$;
5 $j := \text{argmax}\left\{\dfrac{c_i}{d_i \sum_{e \in S_i} y_e} \;\middle|\; i \in K\right\}$;
6 $\mathcal{T} := \mathcal{T} \cup \{S_j\}$; $K := K \setminus \{j\}$;
7 **forall** $e \in S_j$ **do** $y_e := y_e \cdot \left(e^{B-1} m\right)^{q_{S_j}(e)/(b_e - 1)}$;
8 **until** $\sum_{e \in E} b_e y_e \geq e^{B-1} m$ **or** $K = \emptyset$;
9 return $\mathcal{T}$.

Utilitarian Mechanism Design for Single-Minded Agents, Figure 2
Truthful mechanism for network (multicast) routing. e ≈ 2.718 is Euler number.

UFP is the following integer linear program (ILP)

$$\max \quad \sum_{i=1}^{k} c_i \cdot \left(\sum_{S \in S_i} x_S\right) \tag{1}$$

$$\text{s.t.} \quad \sum_{S: S \in \mathcal{S}, e \in S} q_S(e) x_S \leq b_e \quad \forall e \in E \tag{2}$$

$$\sum_{S \in S_i} x_S \leq 1 \quad \forall i \in \{1, \ldots, k\} \tag{3}$$

$$x_S \in \{0, 1\} \quad \forall S \in \mathcal{S}\,. \tag{4}$$

The linear programming (LP) relaxation is the same linear program with constraints (4) replaced with $x_S \geq 0$ for all $S \in \mathcal{S}$. The corresponding dual linear program is

$$\min \quad \sum_{e \in E} b_e y_e + \sum_{i=1}^{k} z_i \tag{5}$$

$$\text{s.t.} \quad z_i + \sum_{e \in S} q_S(e) y_e \geq c_i \quad \forall i \in \{1, \ldots, k\}\ \forall S \in S_i \tag{6}$$

$$z_i, y_e \geq 0 \quad \forall i \in \{1, \ldots, k\}\ \forall e \in E. \tag{7}$$

Based on these LPs, Fig. 2 specifies a primal-dual mechanism for routing, called Greedy-1. Greedy-1 ensures feasibility by using y_e's: if an added set exceeded the capacity b_e of some $e \in E$, then this would imply the stopping condition already in the previous iteration. Using the weak duality of LPs the following result can be shown.

Theorem 3 *Greedy-1 outputs a feasible solution, and it is a* $\left(\frac{e\gamma B}{B-1}(m)^{1/(B-1)}\right)$*-approximation algorithm if there is a polynomial time algorithm that finds a* γ*-approximate set* S_i *in line 4.*

In case of UFP $\gamma = 1$, as the shortest s_i-t_i-path computation finds set S_i in line 4 of Greedy-1. For multicast routing, this problem corresponds to the NP-hard Steiner tree problem, for which one can take $\gamma = 1.55$. Greedy-1 can easily be shown to be monotone in demands and valuations as required in Theorem 1. Thus it implies a truthful mechanism for allocating network resources. The commodities correspond to bidders, the terminal nodes of bidders are known, but the bidders might lie about their demands and valuations. In the multicast routing the set of terminals for each bidder is known but the demands and valuations are unknown.

Corollary 1 *Given any* $\epsilon > 0$, $B \geq 1 + \epsilon$, *Greedy-1 is a truthful* $O(m^{1/(B-1)})$*-approximation mechanism for UFP (unicast routing) as well as for the multicast routing problem, where the demands and valuations of the bidders are unknown.*

When B is large, $\Omega(\log m)$, then the approximation factor in Corollary 1 becomes constant. Azar et al. [4] presented further results in case of large B. Awerbuch et al. [3] gave randomized online truthful mechanisms for uni- and multicast routing, obtaining an expected $O(\log(\mu m))$-approximation if $B = \Omega(\log m)$, where μ is the ratio of the largest to smallest valuation. Their approximation holds in fact with respect to the revenue of the auctioneer, but they assume that the demands are known to the mechanism. Bartal et al. [5] give a truthful $O(B \cdot (m/\theta)^{1/(B-2)})$-approximation mechanism for UFP with unknown valuations and demands, where $\theta = \min_i\{d_i\}$.

Greedy-1 can be modified to give truthful mechanisms for multi-unit CAs among unknown single-mined

Algorithm Greedy-2:

$\mathcal{T} := \emptyset$;
forall $e \in U$ **do** $y_e := 1/b_e$;
repeat
  $S := \text{argmax}\left\{ \frac{c_S}{\sum_{e \in U} q_S(e) y_e} \;\middle|\; S \in S \setminus \mathcal{T} \right\}$;
  $\mathcal{T} := \mathcal{T} \cup \{S\}$;
  forall $e \in S$ **do** $y_e := y_e \cdot (e^B m)^{q_S(e)/b_e}$;
until $\sum_{e \in U} b_e y_e \geq e^B m$;
return $\mathcal{T}$.

Utilitarian Mechanism Design for Single-Minded Agents, Figure 3
Truthful mechanism for multi-unit CAs among unknown single-minded bidders. For CAs without multisets: $q_S(e) \in \{0, 1\}$ for each $e \in U, S \in S$.

bidders.[1] Archer et al. [2] used randomized rounding to obtain a truthful mechanism for multi-unit CAs, but only in a probabilistic sense and only for known bidders. Multi-unit CA among single-minded bidders is a special case of ILP (1)–(4), where $|S_i| = 1$ for each $i \in K$, and $q_S(e) \in \{0, 1\}$ for each $e \in U, S \in S$ (E is U in CAs). A bid of bidder $i \in K$ is $(a_i, v_i) = (S, c_S)$, $S \in S_i$, and $c_S = c_i$ is the valuation. The relation $R_{\preceq}$ is $\subseteq$. Algorithm Greedy-2 in Fig. 3 is exact and monotone for CAs with unknown single-minded bidders, as needed in Theorem 1.

Theorem 4 *Algorithm Greedy-2 is a truthful $O(m^{\frac{1}{B}})$-approximation mechanism for multi-unit CAs among unknown single-minded bidders.*

Bartal et al. [5] presented a truthful mechanism for this problem among unknown single-minded bidders which is $O(B \cdot m^{1/(B-2)})$-approximate. (It works in fact for more general bidders.)

Applications

Applications of the techniques described above are presented and a short survey of other results.

Applications of monotone approximation schemes

In a *forward multi-unit auction* a single auctioneer wants to sell m identical items to n possible buyers (bidders). Each single-minded bidder specifies the number of items she is interested in and a price she is willing to pay. Elements in the introduced notation correspond to the requested and allocated numbers of items. Relation $R_{\preceq}$ describes that bidder i requesting q_i items will be satisfied also by any larger number of items. Mu'alem and Nisan [11] give a 2-approximate monotone algorithm for this problem. Theorem 2 gives a monotone FPTAS for multi-unit auctions among unknown single-minded bidders. This FPTAS is truthful with respect to agents where both the number of items and price are private.

In *job scheduling with deadlines (JSD)*, each agent i has a job with running time t_i, deadline d_i and a price v_i she is willing to pay if her job is processed by deadline d_i. Element a_i is defined as $a_i = (t_i, d_i)$. Output for agent i is a time slot for processing i's job. For two elements $a_i = (t_i, d_i)$ and $a'_i = (t'_i, d'_i)$ one has $a_i \preceq a'_i$ if $t_i \leq t'_i$ and $d_i \geq d'_i$. Theorem 2 leads to a monotone FPTAS, which, however, is not exact (see Theorem 1) with respect to deadlines, and so it is a truthful mechanism only if the deadlines are known. The techniques of Theorem 2 apply also to minimization mechanism design problems with a single buyer, such as reverse multi-unit auctions, scheduling to minimize tardiness, constrained shortest path and minimum spanning tree problems [6].

Applications of the primal dual algorithms

The applications of the primal dual algorithms are combinatorial auctions and auctions for unicast and multicast routing. As these applications are tied very much to the algorithms, they have already been presented in Sect. "Key Results".

Survey of other results

First truthful mechanisms for single-minded CAs were designed by Lehmann et al. [10], where they introduced the concept of single-minded agents, identified the role of monotonicity, and used greedy algorithms to design

[1] In the case of *unknown* single-minded bidders, the bidders have as private data not only their valuations (as in the case of *known* single-minded bidders) but also the sets they demand.

truthful mechanisms. Better approximation ratios of these greedy mechanisms were proved by Krysta [9] with the help of LP duality. A tool-box of techniques for designing truthful mechanisms for CAs was given by Mu'alem and Nisan [11].

The previous section presented a monotone FPTAS for job scheduling with deadlines where jobs are selfish agents and the seller offers the agents the facilities to process their jobs. Such scenarios when jobs are selfish agents to be scheduled on (possibly selfish) machines have been investigated further by Andelman and Mansour [1], see also references therein.

So far social welfare was mostly assumed as the objective, but for a seller probably more important is to maximize her revenue. This objective turns out to be much harder to enforce in mechanism design. Such truthful (in probabilistic sense) mechanisms were obtained for auctioning unlimited supply goods among one-parameter agents [7,8]. Another approach to maximizing seller's revenue is known as optimal auction design [12]. A seller wants to auction a single good among agents and each agent has a private value for winning the good. One assumes that the seller knows a joint distribution of those values and wants to maximize her expected revenue [13,14].

Cross References

Mechanisms that approximately maximize revenue for unlimited-supply goods as of Goldberg, Hartline and Wright 8 are presented in entry ► Competitive Auction.

Recommended Reading

1. Andelman, N., Mansour, Y.: A sufficient condition for truthfulness with single parameter agents. In: Proc. 8th ACM Conference on Electronic Commerce (EC),Ann, Arbor, Michigan, June (2006)
2. Archer, A., Papadimitriou, C.H., Talwar, K., Tardos, E.: An approximate truthful mechanism for combinatorial auctions with single parameter agents. In: Proc. 14th Ann. ACM–SIAM Symp. on Discrete Algorithms (SODA), pp. 205–214. Baltimore, Maryland (2003)
3. Awerbuch, B., Azar, Y., Meyerson, A.: Reducing truth-telling online mechanisms to online optimization. In: Proc. 35th Ann. ACM. Symp. on Theory of Comput. (STOC), San Diego, California (2003)
4. Azar, Y., Gamzu, I., Gutner, S.: Truthful unsplittable flow for large capacity networks. In: Proc. 19th Ann. ACM Symp. on Parallelism in Algorithms and Architectures (SPAA), pp. 320–329 (2007)
5. Bartal, Y., Gonen, R., Nisan, N.: Incentive compatible multi unit combinatorial auctions. In: Proceedings of the 9th conference on Theoretical aspects of rationality and knowledge (TARK), pp. 72–87. ACM Press (2003). http://doi.acm.org/10.1145/846241.846250
6. Briest, P., Krysta, P., Vöcking, B.: Approximation techniques for utilitarian mechanism design. In: Proc. 37th Ann. ACM. Symp. on Theory of Comput. (STOC), pp. 39–48 (2005)
7. Fiat, A., Goldberg, A.V., Hartline, J.D., Karlin, A.R.: Competitive generalized auctions. In: Proc. 34th Ann. ACM. Symp. on Theory of Comput. (STOC), pp. 72–81 (2002)
8. Goldberg, A.V., Hartline, J.D., Wright, A.: Competitive auctions and digital goods. In: Proc. 12th Ann. ACM–SIAM Symp. on Discrete Algorithms (SODA), pp. 735–744 (2001)
9. Krysta, P.: Greedy approximation via duality for packing, combinatorial auctions and routing. In: Proc. 30th Int. Conference on Mathematical Foundations of Comput. Sci. (MFCS). Lecture Notes in Computer Science, vol. 3618, pp. 615–627 (2005)
10. Lehmann, D.J., O'Callaghan, L.I., Shoham, Y.: Truth revelation in approximately efficient combinatorial auctions. In: Proc. 1st ACM Conference on Electronic Commerce (EC), pp. 96–102 (1999)
11. Mu'alem, A., Nisan, N.: Truthful approximation mechanisms for restricted combinatorial auctions. In: Proc. 18th Nat. Conf. Artificial Intelligence, pp. 379–384. AAAI (2002)
12. Myerson, R.B.: Optimal auction design. Math. Oper. Res. **6**, 58–73 (1981)
13. Ronen, A.: On approximating optimal auctions (extended abstract). In: Proc. 3rd ACM Conference on Electronic Commerce (EC), pp. 11–17 (2001)
14. Ronen, A., Saberi, A.: On the hardness of optimal auctions. In: Proc. 43rd Ann. IEEE Symp. on Foundations of Comput. Sci. (FOCS), pp. 396–405 (2002)