Theoretical Computer Science 410 (2009) 1564–1572

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Dynamic mechanism design[☆]

Davide Bilò [a,*], Luciano Gualà [b], Guido Proietti [c,d]

[a] Institut für Theoretische Informatik, ETH, Zürich, Switzerland
[b] Dipartimento di Matematica, Università di Tor Vergata, Roma, Italy
[c] Dipartimento di Informatica, Università di L'Aquila, Italy
[d] Istituto di Analisi dei Sistemi ed Informatica "A. Ruberti", CNR, Roma, Italy

ARTICLE INFO

Keywords:
Algorithmic mechanism design
Truthful mechanisms
Approximate mechanisms
On-line problems
Dynamic algorithms

ABSTRACT

In this paper we address the question of designing truthful mechanisms for solving optimization problems on dynamic graphs with selfish edges. More precisely, we are given a graph G of n nodes, and we assume that each edge of G is owned by a selfish agent. The strategy of an agent consists in revealing to the system – at each time instant – the cost at the actual time for using its edge. Additionally, edges can enter into and exit from G. Among the various possible assumptions which can be made to model how this edge-cost modifications take place, we focus on two settings: (i) the *dynamic*, in which modifications can happen at any time, and for a given optimization problem on G, the mechanism has to maintain efficiently the output specification and the payment scheme for the agents; (ii) the *time-sequenced*, in which modifications happens at fixed time steps, and the mechanism has to minimize an objective function which takes into consideration both the quality and the set-up cost of a new solution. In both settings, we investigate the existence of exact and approximate truthful (w.r.t. to suitable equilibrium concepts) mechanisms. In particular, for the dynamic setting, we analyze the *minimum spanning tree* problem, and we show that if edge costs can only decrease and each agent adopts a *myopic best response* strategy (i.e., its utility is only measured instantaneously), then there exists an efficient dynamic truthful (in myopic best response equilibrium) mechanism for handling a sequence of k declarations of edge-cost reductions having runtime $\mathcal{O}((h+k)\log n)$, where h is the overall number of payment changes.

1. Introduction

Algorithmic mechanism design (AMD) is concerned with the computational complexity of implementing, in a centralized fashion, truthful mechanisms for solving optimization problems in multi-agents systems [15]. AMD is by now one of the hottest topics in theoretical computer science, especially since the game-theoretic nature of the Internet. As a result, many classic network optimization problems have been resettled and solved under this new perspective [3,4,7–9]. Apparently, however, the canonical approach is that of dealing with Internet problems by means of one-shot mechanisms, whose natural computational counterpart are static graph problems. This is in contrast with the intrinsic dynamicity of the Internet infrastructure, where links and node can rapidly appear, disappear, or even change their characteristics. Thus, surprisingly enough, there is a lack of modeling for those situations in which agents need to adapt their strategies over time, according

☆ Work partially supported by the Research Project GRID.IT, funded by the Italian Ministry of Education, University and Research. A preliminary version of this paper was presented at the *2nd International Workshop on Internet and Network Economics (WINE)*, December 15–17, 2006, Patra, Greece.

* Corresponding author.
E-mail addresses: davide.bilo@di.univaq.it (D. Bilò), guala@di.univaq.it (L. Gualà), proietti@di.univaq.it (G. Proietti).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.12.029

to sudden changes in their owned components. To the best of our knowledge, the only effort towards this direction has been done in the framework of the so-called *on-line mechanism design* (OMD) [6,16]. There, the dynamic aspect resides in the fact that agents arrive and depart once over time, and their strategy consists of a *single* announcement of a bidding value for a time interval included between the arrival and the departing time. However, the limitation of OMD is that agents are not allowed to play different strategies over time, thus preventing to model those situations in which bidding values need to be continuously adjusted.

In this paper, we aim to exactly fill this gap, by exploring the difficulties and the potentialities emerging in this new challenging scenario. In doing that, we combine some of the theoretical achievements of the AMD with techniques which are proper of dynamic and on-line algorithms. The result of this activity is what we call as *dynamic mechanism design* (DMD). As a paradigmatic framework, we consider the situation in which each agent owns an edge of a given underlying graph G of n nodes, and its strategy consists in revealing to the system the cost (which can change over time) for using its edge. We focus on two main realistic scenarios:

(1) In the first scenario, we consider the case – sounding very realistic in the Internet setting – in which edge costs are subject to sudden changes, due to boundary condition variations. In the extreme case, an edge might become unavailable to the system, due to a failure for instance. On the opposite side, some new edge might become available. All these variations are presented *on-line* to the system, which is completely unaware of possible future changes. In the rest of the paper, we will refer to this as the *dynamic* scenario.

From an algorithmic point of view, this scenario translates into a continuously evolving input graph, over which a feasible (possibly optimal) solution to a given optimization problem has to be maintained. On the other hand, from a game-theoretic point of view, this scenario models a situation where agents play an *infinitely repeated game*, for which very little is known in terms of mechanism designing, though. Thus, in this paper we make the simplifying yet reasonable assumption that agents are *bounded rational*, and they only measure their utility *instantaneously*. In this way, the standardly adopted notion of dominant strategy equilibrium can be relaxed to that of *myopic best response equilibrium* (see also [10]). With this equilibrium concept in mind, we can easily prove the existence of a corresponding *fully dynamic* truthful mechanism (i.e., a mechanism which updates efficiently both the output specification and the payment scheme for the agents), as soon as the one-shot counterpart of the game admits a truthful (in dominant strategies) mechanism. What is algorithmically interesting in this type of mechanisms is that while classic dynamic graph algorithms can be used for the maintenance of the output specification, as far as the payment scheme updating is concerned, this defines novel dynamic graph problems, which would make no sense in a canonical centralized framework. To make things more concrete, we face a basic graph problem that has served as a case study for several papers on AMD, namely the *minimum spanning tree* (MST) problem. After observing that efficient dynamic MST algorithms [11] can be turned into an efficient dynamic mechanism for handling a sequence of k edge-cost modifications having runtime $\mathcal{O}(k\, n \log^4 n)$, we will show that for the case in which edges can only become less expensive, then the mechanism runtime drops to $\mathcal{O}((h + k) \log n)$, where h is the overall number of payment changes. We emphasize that this edge-cost lowering scenario is interesting because of the competitive nature of Internet.

(2) In the second scenario, we consider the case in which the graph evolves in a sequence of time steps, and every agent has a specific cost for using its edge in each of these steps. Here, the time-depending modifications of the graph suggest that the mechanism's goal should now be the composition of two objectives: maintaining a *good* (not necessarily optimal) solution at a *low* (not necessarily minimal) cost of setting it up. Thus, on a sequence of graph changes, the objective function is now given by the overall cost of the sequence of solutions, plus the overall set-up cost. This approach is inspired to that proposed in the past in [12] to model the fact that on an on-line sequence of changes, it is important to take care of the modifications on the structure of the solution, since radical alterations might be too onerous in terms of set-up costs. In the rest of the paper, we will refer to this as the *time-sequenced* scenario. Here, on a positive side, we will show that: (i) if each set-up cost is upper bounded by the initial one and changes are presented *on-line* to the system, then a ρ-approximate monotone algorithm for a given optimization problem Π on G, translates into an approximate *equilibria-truthful* mechanism (see [14]) for Π which on a sequence of graph changes of size k has an approximation ratio of $\max\{k, \rho\}$ (see Section 2 for the formal definition of *equilibria-truthfulness*); (ii) if the underlying graph optimization problem is utilitarian and polynomial-time solvable, and changes are presented *off-line* to the system, then there exists a VCG-like truthful mechanism for solving optimally the sequence, which can be computed in polynomial time by means of a dynamic programming technique. On the other hand, on a negative side, we will show that even if graph changes are presented off-line to the system and set-up costs are uniform, then any truthful mechanism which solves the problem by means of a *divide et impera* paradigm (as explained in more detail in Section 6) cannot achieve a better than k approximation ratio.

The paper is organized as follows: in Section 2 we give preliminary definitions; after, in Section 3 we present the dynamic mechanism for the MST problem, while in Section 4 we define formally the time-sequenced model; finally, in the last two sections we give, respectively, positive and negative results on the existence of time-sequenced truthful mechanisms.

2. Preliminaries

In this section we recall basic definitions and results of mechanism design theory for one-shot games, which will be then used and extended to our dynamic scenarios in the following sections.

Let a communication network be modeled by a graph $G = (V, E)$ with n nodes and m edges. We will deal with the case in which each edge $e \in E$ is controlled by a selfish agent a_e holding a private information t_e, namely the *true type* of a_e. Only agent a_e knows t_e. Each agent has to declare a public *bid* b_e to the mechanism. We will denote by t the vector of types, and by b the vector of bids. For a given optimization problem Π defined on G, let $\textsc{Sol}(\Pi)$ denote the corresponding set of feasible solutions. We will assume that $\textsc{Sol}(\Pi)$ does not depend on the agents' types. For each $x \in \textsc{Sol}(\Pi)$, an objective function is defined, which depends on the agents' types. A *mechanism* for Π is a pair $\mathcal{M} = \langle g(b), p(b) \rangle$, where $g(b)$ is an algorithm that, given the agents' bids, computes a solution for Π, and $p(b)$ is a scheme which describes the payments provided to the agents. For each solution x, a_e incurs a cost $\nu_e(t_e, x)$ for participating to x (also called *valuation of* a_e *w.r.t.* x). Each agent tries to maximize its *utility*, which is defined as the difference between the payment provided by the mechanism and the cost incurred by the agent w.r.t. the computed solution. On the other hand, the mechanism aims to compute a solution which minimizes the objective function of Π w.r.t. to the agents' types, but of course it does not know t directly. In a *truthful* (in *dominant strategy equilibrium*) mechanism this tension between the agents and the system is resolved, since each agent maximizes its utility when it declares its type, regardless of what the other agents do. A weaker notion of truthfulness is that of *equilibria truthfulness* in which truth-telling is a *Nash equilibrium* (i.e., each agent maximizes its utility if it bids truthfully whenever the other agents bid truthfully).

Given a positive real function $\varepsilon(n)$ of the input size n, an $\varepsilon(n)$-*approximate* mechanism returns a solution whose measure comes within a factor $\varepsilon(n)$ from the optimum. A mechanism has a runtime of $\mathcal{O}(f(n))$ if $g(\cdot)$ and $p(\cdot)$ are computable in $\mathcal{O}(f(n))$ time. Moreover, a mechanism design problem is called *utilitarian* if the objective function of Π is equal to $\sum_{e \in E} \nu(t_e, x)$. For utilitarian problems, there exists a well-known class of truthful mechanisms, i.e., the *Vickrey–Clarke–Groves (VCG) mechanisms*.

In [2], Archer and Tardos have shown how to design truthful mechanisms for another well-known class of mechanism design problems called *one-parameter*. A problem is said one-parameter if (i) the true type of each agent a_e can be expressed as a single parameter $t_e \in \mathbb{R}$, and (ii) each agent's valuation has the form $\nu_e(t_e, x) = t_e\, \omega_e(b)$, where $\omega_e(b)$ is called the *work curve* for agent a_e, i.e., the amount of work for a_e depending on the output specified by the mechanism algorithm, which in its turn is a function of the bid vector b. When, for each agent a_e, $\omega_e(b)$ can be either 0 or 1, then the problem is also called *binary demand* [13]. In [2] it is shown that for one-parameter problems, a sufficient condition for truthfulness is given by a monotonicity property of the mechanism algorithm. In particular, for a binary demand problem, such property reduces to the following. Let b be the vector of bids of the agents, and let b_{-e} denote the vector of all bids besides b_e; the pair (b_{-e}, b_e) will denote the vector b. If we fix b_{-e}, a *monotone* algorithm $\mathcal{A}$ defines a threshold value $\theta_e(b_{-e}, \mathcal{A})$ such that if a_e bids no more than $\theta_e(b_{-e}, \mathcal{A})$, then e will be selected, while if a_e bids above $\theta_e(b_{-e}, \mathcal{A})$, e will not be selected.[1] Sometimes, we will write $\theta_e(b_{-e})$ when the algorithm $\mathcal{A}$ is clear from the context. The results in [2] imply that the only truthful mechanism for a binary demand problem using an algorithm $\mathcal{A}$ is the one-parameter mechanism $\mathcal{M} = \langle \mathcal{A}, p^{\mathcal{A}}(\cdot) \rangle$, where $\mathcal{A}$ is required to be monotone, and the payment $p_e^{\mathcal{A}}(b)$ for each agent a_e is defined as its threshold value if it owns a selected edge, and 0 otherwise.

3. An efficient dynamic mechanism for the MST problem

We start by addressing the problem of designing an efficient mechanism for the fully dynamic MST problem. Since we assume that agents' types change over time, we allow the agents to do a new bid to the mechanism at any time. Recall that edge-cost changes are presented on-line to the system, which is unaware of possible future changes.

In this scenario, all the ingredients of the game (i.e., types, payments, utilities, etc.) must be made dependant on the time. Consequently, to provide positive results in terms of mechanism design, we need to adopt a suitable equilibrium concept in order to avoid falling in the intractability of infinitely repeated games. To this aim, we assume that each agent is now *bounded rational*, and that at any time it only tries to maximize its *instantaneous utility*, which is defined as the difference between the current payment and its cost in the current solution. In other words, we assume that each agent follows a *myopic best response* strategy, namely it only chooses the best response to the current situation without considering the effect that such a strategy will have on the future of the game. In this setting, instead of the standard dominant strategy equilibrium, it makes sense to consider the myopic best response equilibrium concept [10], for which a dynamic truthful mechanism is immediately obtained by repeatedly applying a truthful mechanism in dominant strategies for the one-shot counterpart of the game, if any. Since this is the case for the dynamic MST problem, this implies the existence of a dynamic mechanism such that at any time τ each agent a_e maximizes its instantaneous utility when it bids $b_e(\tau) = t_e(\tau)$, regardless of what the other agents do, i.e., a myopic best response truthful mechanism.

In the rest of the section, we will therefore focus on a snapshot of the dynamic mechanism, and thus we can avoid specifying the time variable. The dynamic mechanism then works as follows: At any time, whenever it receives a new bid

[1] As usual, we will assume that there always exists a feasible solution not containing e, which implies $\theta_e(b_{-e}) < +\infty$.

from an agent, it computes a new MST w.r.t. the new bid profile, and it updates the payments exactly as the one-parameter mechanism for the MST problem. Therefore, concerning the time complexity, the mechanism has to maintain: (i) an MST of G, and (ii) the corresponding payments. Moreover, it has to support a payment query in $\mathcal{O}(1)$ time.

To dynamically maintain an MST, one can use the algorithm proposed in [11], which takes $\mathcal{O}(k \log^4 n)$ time for processing k edge-cost updates (deletions of edges are simulated by setting to $+\infty$ the cost of an edge). Thus, it remains to manage the payment scheme. We remind that for binary demand problems, the payment provided to a_e is equal to $\theta_e(b_{-e})$ if e is selected, and zero otherwise. This means it suffices to maintain the threshold value $\theta_e(b_{-e})$ for each e belonging to the current solution. We emphasize that the algorithm in [11] can be straightforwardly used to accomplish such a task, and from this it follows that there exists a truthful mechanism for the fully dynamic MST which runs in $\mathcal{O}(k\, n \log^4 n)$ time for processing k updates. Improving this latter result is a challenging open problem. In the following, we show that for the *edge-cost decreasing* case, in which edge costs are only allowed to decrease, a significant improvement is possible. We argue this is not a very special case, as it includes the well-known *partially dynamic* scenario, where only edge insertions are allowed.

How to maintain the payments. Let G be a graph, and let T be an MST of G. For each *non-tree edge* $f = (u, v) \in E \setminus E(T)$, $T(f)$ will denote the set of tree edges belonging to the (unique) path in T between u and v. For each $e \in E(T)$, let $C_T(e) = \{f \in E \setminus E(T) \mid e \in T(f)\}$. We denote by *swap*$(e)$ the cheapest non-tree edge in $C_T(e)$.[2] Note that $\theta_e(b_{-e}) = b_{swap(e)}$.

Clearly, if a tree edge decreases its cost, no payment changes. Consider now the situation in which a non-tree edge f decreases its cost from b_f to b'_f. Denote by T' the new MST, i.e., the MST computed w.r.t. the cost profile $b' = (b_{-f}, b'_f)$. We have two cases:

Case 1: $T' = T$; Clearly, only the threshold of edges in $T(f)$ may change, since for each $e' \notin T(f)$, no edge in $C_T(e')$ has changed its cost. Moreover

Fact 1. *Let $e \in T(f)$. Then, the threshold of e changes iff $\theta_e(b_{-e}) > b'_f$. In this case the new threshold value becomes $\theta_e(b'_{-e}) = b'_f$.*

Case 2: $T' \neq T$. Clearly $T' = T \setminus \{e\} \cup \{f\}$. Moreover, the payment for a_e becomes 0, while that for a_f becomes $\theta_f(b'_{-f}) = b_e$, since $C_{T'}(f) \subseteq C_T(e) \cup \{e\}$.

Lemma 3.1. *For every $e' \in E(T') \setminus T'(e)$, $\theta_{e'}(b'_{-e'}) = \theta_{e'}(b_{-e'})$.*

Proof. The lemma trivially follows from the fact that for each $e' \in E(T') \setminus T'(e)$, $C_{T'}(e') = C_T(e')$ and $f \notin C_T(e')$. $\square$

Lemma 3.2. *The threshold of an edge $e' \in T'(e)$ changes iff $\theta_{e'}(b_{-e'}) > b_e$. In this case, $\theta_{e'}(b'_{-e'}) = b_e$.*

Proof. Let $e' \in T'(e)$ be such that $\theta_{e'}(b_{-e'}) > b_e$. Since $e \in C_{T'}(e')$, then $\theta_{e'}(b'_{-e'}) \leq b_e$. We have to show that $\nexists f' \in C_{T'}(e')$ with $b_{f'} < b_e$. For the sake of contradiction, suppose that $\exists f' \in C_{T'}(e')$ such that $b_{f'} < b_e$. Then, we show T was not an MST by proving that $f' \in C_T(e)$. Suppose that $f' \notin C_T(e)$; then $T(f') = T'(f')$, which implies $\theta_{e'}(b_{-e'}) < b_e$.

It remains to show that if $\theta_{e'}(b_{-e'}) \leq b_e$, then $\theta_{e'}(b'_{-e'}) = \theta_{e'}(b_{-e'})$. Notice that if *swap*$(e') \in C_T(e)$, then $\theta_{e'}(b_{-e'}) \geq b_e$ from the minimality of T, which implies $\theta_{e'}(b_{-e'}) = b_e$. Otherwise, *swap*$(e') \in C_{T'}(e')$. In both cases $\theta_{e'}(b'_{-e'}) \leq \theta_{e'}(b_{-e'})$. Moreover, since $C_{T'}(e') \subseteq C_T(e') \cup C_T(e) \cup \{e\}$, then

$$\begin{aligned}\theta_{e'}(b'_{-e'}) &= \min_{f \in C_{T'}(e')} \{b_f\} \geq \min_{f \in C_T(e') \cup C_T(e) \cup \{e\}} \{b_f\} \\ &= \min\{b_{swap(e')}, b_e\} = \theta_{e'}(b_{-e'}). \quad \square\end{aligned}$$

Implementation. To update the payments, we use a *top tree*, a data structure introduced by Alstrup *et al.* [1] to maintain information about paths in trees. More precisely, a top tree represents an edge-weighted forest $\mathcal{F}$ with weight function $c(\cdot)$. Some operations defined for top trees are:

- `link`$((u, v), x)$, where u and v are in different trees. It links these trees by adding the edge (u, v) of weight $c(u, v) = x$ to $\mathcal{F}$.
- `cut`(e). It removes the edge e from $\mathcal{F}$.
- `update`(e, x), where e belongs to $\mathcal{F}$. It sets the weight of e to x.
- `max`(u, v), where u and v are connected in $\mathcal{F}$. It returns the edge with maximum weight among the edges on the path between u and v in $\mathcal{F}$.

[2] If there are more than one such cheapest edges, we pick one of them arbitrarily.

In [1,5,17], it is shown how to implement a top tree (by using $\mathcal{O}(n)$ space) for supporting each of the above operations in $\mathcal{O}(\log n)$ time.

To our scopes, we use two top trees, say $\mathcal{T}_1$ and $\mathcal{T}_2$, as follows. Both $\mathcal{T}_1$ and $\mathcal{T}_2$ maintain the current MST where the cost of each edge $e \in E(T)$ is b_e in $\mathcal{T}_1$ and $\theta_e(b_{-e})$ in $\mathcal{T}_2$. Concerning Case 1, we only need to update the threshold of some edges in $T(f)$. So, let $f = (x, y)$ be the edge which has decreased its cost. We update $\mathcal{T}_2$ as follows. While $c(e') > b'_f$, where $e' = \mathtt{max}(x, y)$, then we (i) update the payment for $a_{e'}$ to b'_f, and (ii) perform $\mathtt{update}(e', b'_f)$. For what concerns Case 2, we first locate the edge $e = (x', y')$ in T not in T' by performing $\mathtt{max}(x, y)$ in $\mathcal{T}_1$, where $f = (x, y)$ is the edge which has decreased its cost. Then, we update the MST by performing $\mathtt{cut}(e)$ in both $\mathcal{T}_1$ and $\mathcal{T}_2$, and $\mathtt{link}(f, b'_f)$ in $\mathcal{T}_1$ and $\mathtt{link}(f, b_e)$ in $\mathcal{T}_2$. Next, we update the payment for a_e (resp., a_f) to 0 (resp., b_e). Finally, we update $\mathcal{T}_2$ as follows. While $c(e') > b_e$, where $e' = \mathtt{max}(x', y')$, then (i) we update the payment for $a_{e'}$ to b_e, and (ii) we perform $\mathtt{update}(e', b_e)$.

The above discussion yields the following:

Theorem 3.1. *There exists a dynamic truthful (in myopic best response equilibrium) mechanism for the MST problem supporting a sequence of k edge-cost decreasing operations in $\mathcal{O}((h + k) \log n)$ time, where h is the overall number of payment changes.* □

4. Time-sequenced scenario: Problem statement

Let $G = (V, E)$ be a graph with a positive real weight $w(e)$ associated with each edge $e \in E$. Henceforth, unless stated otherwise, by Π we will denote a *communication network* problem on (G, w), which asks for computing a subgraph $H \in \textsc{Sol}(\Pi)$ of G by minimizing an objective function $\phi(H, w)$ of the form

$$\phi(H, w) = \sum_{e \in E(H)} w(e) \cdot \mu_H(e),$$

where $\mu_H(\cdot)$ is any positive real function depending only on H. Notice that this definition embraces the quasi-totality of communication network problems, like the MST problem, the *shortest-paths tree* problem, and so on.

Let k be a positive integer. We assume that the type of each agent a_e is $t_e = \langle t_e^1, \ldots, t_e^k \rangle$, while its bid is $b_e = \langle b_e^1, \ldots, b_e^k \rangle$. Intuitively, t_e^i represents the true cost incurred by a_e for using its link e at time i. We will denote by $t^i \in \mathbb{R}^m$ the vector of agents' types at time i, and by t the vector $\langle t^1, \ldots, t^k \rangle$.

Given a communication network problem Π, we want to design a truthful mechanism for the optimization problem that we will denote by $\textsc{Seq}(\Pi)$. This latter problem asks for computing a sequence $\mathcal{H} = \langle H_1, \ldots, H_k \rangle$, where $H_i \in \textsc{Sol}(\Pi)$, $i = 1, \ldots, k$, by minimizing the following objective

$$\Psi(\mathcal{H}, t) = \Phi(\mathcal{H}, t) + \Gamma(\mathcal{H}),$$

where $\Phi(\mathcal{H}, t)$ is a function measuring the quality of the solution $\mathcal{H}$, and $\Gamma(\mathcal{H})$ is a function measuring the overall set-up cost. For a given sequence $\mathcal{H}$, we will naturally assume that the valuation of a_e w.r.t. $\mathcal{H}$ is:

$$\nu_e(\mathcal{H}, t_e) = \sum_{i=1}^{k} \nu_e^i(H_i, t_e^i), \quad \text{where } \nu_e^i(H_i, t_e^i) = \begin{cases} t_e^i & \text{if } e \in E(H_i); \\ 0 & \text{otherwise.} \end{cases}$$

Depending on the cost model to be adopted, the functions $\Phi(\cdot)$ and $\Gamma(\cdot)$ can be defined accordingly. In this paper, we will consider the prominent *additive* cost model, in which

$$\Phi(\mathcal{H}, t) = \sum_{i=1}^{k} \phi(H_i, t^i), \qquad \Gamma(\mathcal{H}) = \sum_{i=1}^{k} \gamma(i, \mathcal{H}),$$

where

$$\gamma(i, \mathcal{H}) = \begin{cases} \gamma_1 \in \mathbb{R}^+ & \text{if } i = 1; \\ \gamma_i \in \mathbb{R}^+ & \text{if } H_i \neq H_{i-1},\ i = 1, \ldots, k; \\ 0 & \text{otherwise.} \end{cases}$$

For any $1 \le i \le j \le k$, by $[i, j]$ we will denote the *interval* $\{i, \ldots, j\}$. We will write $[i, j)$ instead of $[i, j-1]$. Given two intervals $[i, j]$, $[i', j']$, we write $[i, j] \prec [i', j']$ if $j < i'$. An *interval vector* $s = \langle I_1, \ldots, I_h \rangle$ is a vector of pairwise disjoint intervals whose union is $\{1, \ldots, k\}$, and such that $I_1 \prec \cdots \prec I_h$. Given an interval I, let b^I be the vector defined as $b_e^I = \sum_{i \in I} b_e^i$, for each edge $e \in E$. Moreover, we will denote by H_I^* an optimum solution for Π when the input is (G, b^I). Finally, given two sequences $\mathcal{H} = \langle H_1, \ldots, H_i \rangle$, $\mathcal{H}' = \langle H'_1, \ldots, H'_j \rangle$, by $\mathcal{H} \odot \mathcal{H}'$ we denote the sequence $\langle H_1, \ldots, H_i, H'_1, \ldots, H'_j \rangle$.

5. Time-sequenced mechanisms: Positive results

In this section we first define the class of *time-sequenced single-parameter (TSSP) mechanisms*, and we prove that any mechanism in this class is truthful for $\textsc{Seq}(\Pi)$ when graph changes are presented off-line to the system. Moreover, for the case in which each set-up cost is upper bounded by γ_1, we show that there exists a $\max\{k, \rho\}$-approximate TSSP mechanism, where ρ is the approximation ratio of a monotone algorithm for Π. We point out that such mechanism can also be used for the on-line version of $\textsc{Seq}(\Pi)$ for which we prove that it enjoys the property of being equilibria-truthful. Then, we turn our attention to the special case in which Π is utilitarian and polynomial-time solvable, and we show that if the graph changes are presented off-line to the system, then there exists a VCG-like truthful mechanism for solving optimally $\textsc{Seq}(\Pi)$, which can be computed in polynomial time by means of a dynamic programming technique.

5.1. On-line sequences with bounded set-up costs

From now on, by $\tilde{s}$ we will denote the interval vector $\langle [1, 1], \ldots, [k, k] \rangle$.

Definition 5.1. Given a communication network problem Π, and a monotone algorithm $\mathcal{A}$ for Π, a *TSSP mechanism* $\mathcal{M}(s) = \langle g_s(b), p(b) \rangle$ *with interval vector* $s = \langle I_1, \ldots, I_h \rangle$ for $\textsc{Seq}(\Pi)$ is defined as follows:

(1) $g_s(\cdot)$ returns a sequence $\mathcal{H} = \mathcal{H}_1 \odot \cdots \odot \mathcal{H}_h$, in which

$$\forall j = 1, \ldots, h, \quad \mathcal{H}_j = \langle \hat{H}_j, \ldots, \hat{H}_j \rangle \text{ has size } |I_j|,$$

where $\hat{H}_j$ is the solution returned by $\mathcal{A}$ with input (G, b^{I_j});

(2) For each agent a_e

$$p_e(b) = \sum_{j=1}^{h} p_e^{\mathcal{A}}\big(b^{I_j}\big),$$

where $p_e^{\mathcal{A}}(b^{I_j})$ is the payment provided to a_e by the one-parameter mechanism $\langle \mathcal{A}, p^{\mathcal{A}}(\cdot) \rangle$ for the problem Π when the input is (G, b^{I_j}).

We can state the following:

Proposition 5.1. *$\mathcal{M}(s)$ is a truthful mechanism for* $\textsc{Seq}(\Pi)$ *when graph changes are presented off-line to the system.*

Proof. The mechanism breaks the problem in h instances $(G, b^{I_1}), \ldots, (G, b^{I_h})$ which are independent each other. Then it uses the one-parameter mechanism $\langle \mathcal{A}, p^{\mathcal{A}}(\cdot) \rangle$ for each of them in order to locally guarantee the truthfulness. □

Notice that, by definition, $\mathcal{M}(\tilde{s})$ is the only TSSP mechanism that can be used for the on-line version of $\textsc{Seq}(\Pi)$. In such an on-line setting, we allow each agent to choose its bid at a given step based on what happened in the previous ones. Now we analyze $\mathcal{M}(\tilde{s})$ and we show that (i) unfortunately for the on-line scenario $\mathcal{M}(\tilde{s})$ turns to be not truthful even when the agents cannot observe the other agents' strategies, (ii) $\mathcal{M}(\tilde{s})$ however enjoys the property of being equilibria-truthful, and (iii) $\mathcal{M}(\tilde{s})$ had an approximation guarantee of $\max\{k, \rho\}$ when all the setup costs are upper bounded by the initial one γ_1.

Let us consider the mechanism $\mathcal{M}(\tilde{s})$ when $k = 3$ and Π is the MST problem. Let G be a graph consisting of two parallel edges e_1 and e_2, and assume that the types of the agents controlling e_1 and e_2 are $t_{e_1} = \langle 1, 1, 1 \rangle$ and $t_{e_2} = \langle 3, 3, 3 \rangle$, respectively. We remind that in this case $\mathcal{M}(\tilde{s})$ reduces to the following mechanism. At each step, the mechanism takes the two bids of the agents as input, selects the cheapest edge and pays 0 the non-selected agent, while provides a payment to the selected agent that is equal to the bid of the non-selected one. Now, consider the following on-line strategy for a_2: a_2 bids truthfully at step 1, and then it bids $b_{e_2}^2 = b_{e_2}^3 = 3$ if it belongs to the solution computed at step 1, $b_{e_2}^2 = b_{e_2}^3 = 0$ otherwise. It is easy to see that in this case, a_{e_1} has convenience to bid $b_{e_1} = \langle 4, 1, 1 \rangle$ (by obtaining an overall utility of 4) instead of bidding truthfully (from which it would obtain an overall utility of 2).

However, for the on-line version of $\textsc{Seq}(\Pi)$, we have the following:

Proposition 5.2. *$\mathcal{M}(\tilde{s})$ is an equilibria-truthful mechanism for* $\textsc{Seq}(\Pi)$ *when graph changes are presented on-line to the system.*

Proof. Let a_e be an agent, and let us assume that all the other agents bid truthfully at all time steps. It is clear that at any time step i, the strategy of the other agents does not depend on what a_e does at steps $j \leq i$. Thus, in order to maximize the overall utility, a_e has to maximize its utility at each step. Then the claim follows form the fact that $\mathcal{M}(\tilde{s})$ uses a (truthful) one-parameter mechanism for Π, which implies that a_e maximizes its utility at each step when it bids truthfully. □

We are now ready to prove the following:

Theorem 5.1. *Given a ρ-approximate monotone algorithm $\mathcal{A}$ for Π, the on-line mechanism $\mathcal{M}(\tilde{s})$ applied to* $\textsc{Seq}(\Pi)$ *with the assumption that each set-up cost is upper bounded by γ_1, has a performance guarantee of* $\max\{k, \rho\}$.

Proof. Equilibria-truthfulness follows from Proposition 5.2. Concerning the approximation ratio of the mechanism, let $\mathcal{H}^* = \langle H'_1, \dots, H'_k \rangle$ be the optimal sequence. By H^*_i we denote an optimal solution for Π with input (G, t^i). It is easy to see that

$$\Psi(\mathcal{H}^*, t) \geq \gamma_1 + \sum_{i=1}^{k} \phi(H'_i, t^i) \geq \gamma_1 + \sum_{i=1}^{k} \phi(H^*_i, t^i).$$

If by $\mathcal{H} = \langle H_1, \dots, H_k \rangle$ we denote the sequence computed by the algorithm of the mechanism $\mathcal{M}(\tilde{s})$, then we have that

$$\frac{\Psi(\mathcal{H}, t)}{\Psi(\mathcal{H}^*, t)} \leq \frac{k\gamma_1 + \sum_{i=1}^{k} \phi(H_i, t^i)}{\Psi(\mathcal{H}^*, t)} \leq \frac{k\gamma_1 + \rho \sum_{i=1}^{k} \phi(H^*_i, t^i)}{\Psi(\mathcal{H}^*, t)} \leq \frac{k + \rho\, x}{1 + x},$$

where $x = \frac{1}{\gamma_1} \sum_{i=1}^{k} \phi(H^*_i, t^i)$. Since the function $\sigma(x) = \frac{k+\rho x}{1+x}$ is non-increasing when $k \geq \rho$, and it is increasing when $\rho > k$, then the approximation ratio achieved by the mechanism is $\max\{k, \rho\}$. □

5.2. Off-line utilitarian problems

In this section we show how to design an exact off-line mechanism when Π is utilitarian. Before defining our mechanism, we show how to compute an optimal sequence by using dynamic programming.

Let $\mathcal{H}^*$ denote an optimal solution for Seq(Π), and let $\mathcal{H}^*_{[1,i]}$ be an optimal solution for Seq(Π) when the input is restricted to the interval $[1, i]$, i.e. we have i time steps and the bid vector is $\langle b^1, \dots, b^i \rangle$. In order to lighten the notation, we will write $\Psi(\mathcal{H}_{[1,i]}, b)$ instead of $\Psi(\mathcal{H}_{[1,i]}, \langle b^1, \dots, b^i \rangle)$, where $\mathcal{H}_{[1,i]}$ is a solution for Seq(Π) restricted to the interval $[1, i]$. Notice that $\mathcal{H}^*_{[1,1]} = \langle H^*_{[1,1]} \rangle$, and $\Psi(\mathcal{H}^*_{[1,1]}, b) = \phi(H^*_{[1,1]}, b^1) + \gamma_1$. Moreover, $\mathcal{H}^*_{[1,k]} = \mathcal{H}^*$.

The dynamic programming algorithm computes $H^*_{[i,j]}$, for every $1 \leq i \leq j \leq k$. Next, starting from $i = 1$ to k, it computes $\mathcal{H}_{[1,i]} = \mathcal{H}_{[1,h_i)} \odot \langle H^*_{[h_i,i]} \rangle$, with

$$h_i = \arg\min_{h=1,\dots,i} \left\{ \Psi'(b, h, i) := \Psi(\mathcal{H}_{[1,h)}, b) + \phi\big(H^*_{[h,i]}, b^{[h,i]}\big) + \gamma_h \right\},$$

where $\mathcal{H}_{[1,1)}$ is the empty sequence, and $\Psi(\mathcal{H}_{[1,1)}, b)$ is assumed to be 0.

Lemma 5.1. *For any $i = 1, \dots, k$, the dynamic programming algorithm computes a solution $\mathcal{H}_{[1,i]}$ such that $\Psi(\mathcal{H}_{[1,i]}, b) = \Psi(\mathcal{H}^*_{[1,i]}, b)$.*

Proof. The proof is by induction on i. The basic case $i = 1$ is trivial.

Let $i > 1$, and assume that for every $j < i$, we have $\Psi(\mathcal{H}_{[1,j]}, b) = \Psi(\mathcal{H}^*_{[1,j]}, b)$. Now we prove the claim for i. Let $\mathcal{H}^*_{[1,i]} = \langle H_1, \dots, H_i \rangle$, and let j^* be the largest index such that $H_{j^*-1} \neq H_{j^*}$. If j^* is undefined, we assume $j^* = 1$. Clearly $\langle H_1, \dots, H_{j^*-1} \rangle = \mathcal{H}^*_{[1,j^*)}$. Since $j^* - 1 < i$, by the inductive hypothesis we have that $\Psi(\mathcal{H}_{[1,j^*)}, b) = \Psi(\mathcal{H}^*_{[1,j^*)}, b)$. Hence, we have that

$$\begin{aligned}\Psi(\mathcal{H}^*_{[1,i]}, b) &= \Psi(\mathcal{H}^*_{[1,j^*)}, b) + \phi\big(H^*_{[j^*,i]}, b^{[j^*,i]}\big) + \gamma_{j^*} \\ &= \Psi(\mathcal{H}_{[1,j^*)}, b) + \phi\big(H^*_{[j^*,i]}, b^{[j^*,i]}\big) + \gamma_{j^*} = \Psi'(b, j^*, i).\end{aligned}$$

To conclude the proof, notice that $\Psi'(b, j, i) \geq \Psi(\mathcal{H}^*_{[1,i]}, b), \forall j \leq i$, as $\gamma_j \geq 0$. □

We are now ready to define our VCG-like mechanism. Let $\mathcal{M}_{\text{VCG}}$ be a mechanism defined as follows:

(1) The algorithmic output specification selects an optimal sequence (w.r.t. the bids b) $\mathcal{H}^*_G$;
(2) Let $G - e = (V, E \setminus \{e\})$, and let $\mathcal{H}^*_{G-e}$ be an optimal solution (w.r.t. the bids b) in $G - e$. Then, the payment function for a_e is defined as

$$p_e(b) = \Psi(\mathcal{H}^*_{G-e}, b) - \Psi(\mathcal{H}^*_G, b) + \nu_e(\mathcal{H}^*_G, b_e).$$

Theorem 5.2. *Let Π be utilitarian and solvable in polynomial time. Then, $\mathcal{M}_{\text{VCG}}$ is an exact off-line truthful mechanism for Seq(Π) which can be computed in polynomial time.* □

6. Time-sequenced mechanisms: Inapproximability results

In this section we consider a natural extension of TSSP mechanisms named *adaptive TSSP mechanisms*, and we prove a lower bound of k to the approximation ratio that can be achieved by any truthful mechanism in this class.

Definition 6.1. Let δ be a function mapping bid vectors to interval vectors. An *adaptive time-sequenced single-parameter (ATSSP) mechanism* $\mathcal{M}_\delta$ for Π is the mechanism which, for a given vector bid b, is defined exactly as $\mathcal{M}(\delta(b))$.

Lemma 6.1. *Let t^i be a type profile for Π, and let $\mathcal{A}$ be an optimal algorithm for Π. Then, $\forall \eta \in \mathbb{R}^+$, $\theta^i_e(\eta \cdot t^i_{-e}) = \eta \cdot \theta^i_e(t^i_{-e})$.*

Proof. Observe that $\forall H \in \textsc{Sol}(\Pi)$

$$\phi(H, \eta \cdot t^i) = \sum_{e \in E(H)} \eta \cdot t^i_e \, \mu_H(e) = \eta \sum_{e \in E(H)} t^i_e \, \mu_H(e) = \eta \cdot \phi(H, t^i). \quad \square$$

Theorem 6.1. *For any mapping function δ, for any optimal algorithm $\mathcal{A}$ for Π, and for any $c < k$, there exists no c-approximate truthful ATSSP mechanism using $\mathcal{A}$ for $\textsc{Seq}(\Pi)$, even when set-up costs are uniform.*

Proof. The proof is by contradiction. Let $M = \gamma_1 = \cdots = \gamma_k$. Let $\mathcal{M}_\delta$ be a c-approximate truthful ATSSP mechanism for $\textsc{Seq}(\Pi)$. For the sake of clarity, we denote by $H(w)$ an optimum solution for Π with input (G, w). Let $t^1 = (t^1_{-e}, t^1_e)$, with $t^1_{-e} = \langle 0, \ldots, 0 \rangle$, and $t^2 = (t^2_{-e}, 0)$ be two type vectors for Π such that the following three conditions hold:

(i) $2t^1_e < \theta^2_e, t^1_e > 0$, where $\theta^2_e = \theta_e(t^2_{-e})$;
(ii) $\phi(H(t^2_{-e}, +\infty), (t^2_{-e}, +\infty)) \geq (k^2 - 1)M$;
(iii) $\phi(H(t^2_{-e}, x), (t^2_{-e}, x))$ does not depend on M, for any $x < \theta^2_e$ not depending on M.

Lemma 6.2. *There always exist t^1_e and t^2_{-e} satisfying the above conditions.*

Proof. Let $H \in \textsc{Sol}(\Pi)$ be such that $E(H') \not\subset E(H), \forall H' \in \textsc{Sol}(\Pi)$. Let e be an edge of H. Now for each $e' \in E(H) \setminus \{e\}$, let $t^2_{e'} = \frac{1}{\mu_H(e')}$. Moreover, for each $e' \in E \setminus E(H)$, let $t^2_{e'}$ be defined as follows

$$t^2_{e'} = \max_{H' \in \textsc{Sol}(\Pi)} \frac{(k^2 - 1)M}{\mu_{H'}(e')}.$$

By construction, condition (ii) holds. For M large enough, it is easy to see that θ^2_e is at least $(k^2 - 1)M - |E(H)| > 0$, from which (i) follows as well. Finally, condition (iii) follows by observing that $\mu_H(e)$ does not depend on M. $\square$

Let t be the type profile defined as follows:

$$\forall i = 1, \ldots, k, \quad t^i = \begin{cases} t^1 & \text{if } i \text{ is odd;} \\ t^2 & \text{otherwise.} \end{cases}$$

Lemma 6.3. *For M large enough, $\delta(t) \neq \tilde{s}$.*

Proof. The proof is by contradiction. Let $\mathcal{H}$ be the solution computed by the mechanism corresponding to the interval vector $\tilde{s}$. Notice that $\Psi(\mathcal{H}, t) \geq kM$, since $H(t^1) \neq H(t^2)$. Consider now the solution $\mathcal{H}'$ corresponding to the interval vector $\langle [1, k] \rangle$. It is easy to see that for t^1_e small enough, $\Psi(\mathcal{H}', t) = M + \phi\big(H\big(t^{[1,k]}\big), t^{[1,k]}\big) \leq M + k\,\phi(H(t^2_{-e}, t^1_e), (t^2_{-e}, t^1_e))$. It follows that the approximation ratio achieved by the mechanism is at least

$$\frac{\Psi(\mathcal{H}, t)}{\Psi(\mathcal{H}', t)} \geq \frac{kM}{M + k\,\phi(H(t^2_{-e}, t^1_e), (t^2_{-e}, t^1_e))},$$

which, from (iii), goes to k when M goes to $+\infty$. This contradicts the fact that $\mathcal{M}_\delta$ is c-approximate. $\square$

Lemma 6.4. *For M large enough, the utility of a_e in the solution $g_{\delta(t)}(t)$ computed by the mechanism $\mathcal{M}_\delta$ is less than $\left\lfloor \frac{k}{2} \right\rfloor \theta^2_e$.*

Proof. Let $\delta(t) = \langle I_1, \ldots, I_h \rangle$ be the interval vector computed by δ, and let $\mathcal{H}$ be the corresponding solution. For each $j = 1, \ldots, h$, let $I_j = [x_j, y_j]$ be the j-th interval, and let η_j be the number of occurrences of t^2 in $\langle t^{x_j}, \ldots, t^{y_j} \rangle$. Notice that $t^{I_j} = (\eta_j\, t^2_{-e}, (|I_j| - \eta_j)\, t^1_e)$. It is easy too see that $(|I_j| - \eta_j) \leq \eta_j + 1$. Moreover, notice that e belongs to $H\big(t^{I_j}\big)$ iff $\eta_j > 0$. Indeed, whenever $\eta_j > 0$, $(\eta_j + 1)\, t^1_e < \eta_j\, \theta^2_e$ holds from (i), and from Lemma 6.1 this implies that e belongs to $H\big(t^{I_j}\big)$. Finally, notice that whenever $|I_j| > 1$, a_e incurs a cost of at least t^1_e.

Then, from Lemma 6.1, the payment provided to a_e is $\sum_{j=1}^{h} \eta_j\, \theta^2_e = \left\lfloor \frac{k}{2} \right\rfloor \theta^2_e$, while concerning the cost incurred by a_e, it is at least $t^1_e > 0$, since from Lemma 6.3 there must exist an index j^* such that $|I_{j^*}| > 1$. $\square$

Consider now the following new type profile $\hat{t}$ which is equal to t except for $\hat{t}^i_e$ that is set to $+\infty$ for every odd i.

Lemma 6.5. *For M large enough, $\delta(\hat{t}\,) = \tilde{s}$.*

Proof. For the sake of contradiction, assume that $\delta(\hat{t}) \neq \tilde{s}$. Then, there must exist an index j for which the solution $\mathcal{H}$ computed by the mechanism does not change at time j. Hence, since either $\hat{t}^j_e = +\infty$ or $\hat{t}^{j-1}_e = +\infty$, from (ii) it must be $\Psi(\mathcal{H}, \hat{t}) \geq k^2 M$. Consider the solution $\mathcal{H}'$ corresponding to the interval vector $\tilde{s}$. Then, the approximation ratio achieved by the mechanism is at least

$$\frac{\Psi(\mathcal{H}, \hat{t})}{\Psi(\mathcal{H}', \hat{t})} \geq \frac{k^2 M}{kM + k\,\phi(H(t^2), t^2)},$$

which, from (iii), goes to k when M goes to $+\infty$. This contradicts the fact that $\mathcal{M}_\delta$ is c-approximate. $\square$

To conclude the proof, observe that when the type profile is t, a_e has convenience to bid b_e defined as

$$\forall i = 1, \ldots, k, \quad b_e^i = \begin{cases} t_e^2 & \text{if } i \text{ is even;} \\ +\infty & \text{otherwise.} \end{cases}$$

Indeed, in this case, from Lemma 6.5, its utility becomes equal to $\lfloor \frac{k}{2} \rfloor \theta_e^2$, which is better than the utility it gets by bidding truthfully (see Lemma 6.4). □

Notice that, since in the uniform set-up cost case each set-up cost is upper bounded by γ_1, and since $\mathcal{M}(\tilde{s})$ belongs to the class ATSSP, then Theorem 6.1 implies that the upper bound in Theorem 5.1 is tight, when $\mathcal{A}$ is optimal (i.e., $\rho = 1$).

References

[1] S. Alstrup, J. Holm, K. de Lichtenberg, M. Thorup, Minimizing diameters of dynamic trees, in: Proc. 24th International Colloquium on Automata Languages and Programming, ICALP'97, in: Lecture Notes in Computer Science, vol. 1256, Springer-Verlag, pp. 270–280.

[2] A. Archer, É Tardos, Truthful mechanisms for one-parameter agents, in: Proc. 42nd IEEE Symp. on Foundations of Computer Science, FOCS'01, 2001, pp. 482–491.

[3] D. Bilò, L. Gualà, G. Proietti, On the existence of truthful mechanisms for the minimum-cost approximate shortest-paths tree problem, in: Proc. 13th Coll. on Structural Information and Communication Complexity, SIROCCO'06, in: Lecture Notes in Computer Science, vol. 4056, Springer-Verlag, 2006, pp. 295–309.

[4] P. Briest, P. Krysta, B. Vöcking, Approximation techniques for utilitarian mechanism design, in: Proc. 37th Ann. ACM Symp. on Theory of Computing, STOC'05, 2005, pp. 39–48.

[5] G.N. Frederickson, Data structures for on-line updating of minimum spanning trees, with applications, SIAM J. Comput. 14 (4) (1985) 781–798.

[6] E.J. Friedman, D.C. Parkes, Pricing WiFi at Starbucks: Issues in online mechanism design, in: Proc. 4th ACM Conf. on Electronic Commerce EC'03, 2003, pp. 240–241.

[7] L. Gualà, G. Proietti, A truthful (2–2/k)-approximation mechanism for the Steiner tree problem with k terminals, in: Proc. 11th Int. Computing and Combinatorics Conference, COCOON'05, in: Lecture Notes in Computer Science, vol. 3595, Springer-Verlag, 2005, pp. 390–400.

[8] L. Gualà, G. Proietti, Efficient truthful mechanisms for the single-source shortest paths tree problem, Concurrency and Computation: Practice and Experience 19 (17) (2007) 2285–2297.

[9] J. Hershberger, S. Suri, Vickrey prices and shortest paths: What is an edge worth? in: Proc. 42nd IEEE Symp. on Foundations of Computer Science, FOCS'01, 2001, pp. 252–259.

[10] B. Heydenreich, R. Müller, M. Uetz, Decentralization and mechanism design for online machine scheduling, in: Proc. 10th Scandinavian Workshop on Algorithm Theory, SWAT'06, in: Lecture Notes in Computer Science, vol. 4059, Springer-Verlag, 2006, pp. 136–147.

[11] J. Holm, K. de Lichtenberg, M. Thorup, Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity, J. ACM 48 (4) (2001) 723–760.

[12] B. Kalyanasundaram, K. Pruhs, On-line network optimization problems, in: Online Algorithms: The State of the Art, in: Lecture Notes in Computer Science, vol. 1442, Springer-Verlag, 1998, pp. 268–280.

[13] M.-Y. Kao, X.-Y. Li, W. Wang, Towards truthful mechanisms for binary demand games: A general framework, in: Proc. 6th ACM Conf. on Electronic Commerce, EC'05, 2005, pp. 213–222.

[14] R. Lavi, C. Swamy, Truthful and near-optimal mechanism design via linear programming, in: Proc. 46th IEEE Symp. on Foundations of Computer Science, FOCS'05, 2005, pp. 595–604.

[15] N. Nisan, A. Ronen, Algorithmic mechanism design, Games Econom. Behav. 35 (2001) 166–196.

[16] D.C. Parkes, S.P. Singh, D. Yanovsky, Approximately efficient online mechanism design, in: Proc. 18th Annual Conf. on Neural Information Processing Syst. 17, NIPS'04, 2004.

[17] D.D. Sleator, R.E. Tarjan, A data structure for dynamic trees, J. Comput. Syst. Sci. 26 (3) (1983) 362–391.