Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 101 (2007)

Stackelberg Network Pricing Games

Patrick Briest [*] Martin Hoefer [†] Piotr Krysta [‡]

Abstract

We study a multi-player one-round game termed Stackelberg Network Pricing Game, in which a *leader* can set prices for a subset of m pricable edges in a graph. The other edges have a fixed cost. Based on the leader's decision one or more *followers* optimize a polynomial-time solvable combinatorial minimization problem and choose a minimum cost solution satisfying their requirements based on the fixed costs and the leader's prices. The leader receives as revenue the total amount of prices paid by the followers for pricable edges in their solutions. Our model extends several known pricing problems, including single-minded and unit-demand pricing, as well as Stackelberg pricing for certain follower problems like shortest path or minimum spanning tree. Our first main result is a tight analysis of a single-price algorithm for the single follower game, which provides a $(1+\varepsilon)\log m$-approximation for any $\varepsilon > 0$. This can be extended to provide a $(1+\varepsilon)(\log k + \log m)$-approximation for the general problem and k followers. The latter result is essentially best possible, as the problem is shown to be hard to approximate within $\mathcal{O}(\log^{\varepsilon} k + \log^{\varepsilon} m)$. If followers have demands, the single-price algorithm provides a $(1+\varepsilon)m^2$-approximation, and the problem is hard to approximate within $\mathcal{O}(m^{\varepsilon})$ for some $\varepsilon > 0$. Our second main result is a polynomial time algorithm for revenue maximization in the special case of Stackelberg bipartite vertex cover, which is based on non-trivial max-flow and LP-duality techniques. Our results can be extended to provide constant-factor approximations for any constant number of followers.

[*] Department of Computer Science, The University of Liverpool, United Kingdom. `patrick.briest@liverpool.ac.uk`.
Supported by DFG grant Kr 2332/1-2 within Emmy Noether program.
[†] Department of Computer & Information Science, Kosntanz University, Germany. `hoefer@inf.uni-konstanz.de`.
Supported by DFG Graduiertenkolleg "Explorative Analysis and Visualization of Large Information Spaces".
[‡] Department of Computer Science, The University of Liverpool, United Kingdom. `p.krysta@csc.liv.ac.uk`.

ISSN 1433-8092

1 Introduction

Algorithmic pricing problems model the task of assigning revenue maximizing prices to a retailer's set of products given some estimate of the potential customers' preferences in purely computational [14], as well as strategic [3] settings. Previous work in this area has mostly focused on settings in which these preferences are rather restricted, in the sense that products are either *pure complements* [2, 7, 15, 16] and every customer is interested in exactly one subset of products or *pure substitutes* [1, 8, 10, 14, 15, 16], in which case each customer seeks to buy only a single product out of some set of alternatives. A customer's real preferences, however, are often significantly more complicated than that and therefore pose some additional challenges.

The modelling of consumer preferences has received considerable attention in the context of *algorithmic mechanism design* [18] and *combinatorial auctions* [12]. The established models range from relatively simple bidding languages to bidders that are represented by oracles allowing certain types of queries, e.g., revealing the desired bundle of items given some fixed set of prices. The latter would be a somewhat problematic assumption in the theory of pricing algorithms, where we usually assume to have access to a rather large number of potential customers through some sort of sampling procedure and, thus, are interested in preferences that allow for a compact kind of representation.

In this paper we focus on customers that have non-trivial preferences, yet can be fully described by their *types* and *budgets* and do not require any kind of oracles. Assume that a company owns a subset of the links in a given network. The remaining edges are owned by other companies and have fixed publicly known prices and some customer needs to purchase a path between two terminals in the network. Since she is acting rational, she is going to buy the shortest path connecting her terminals. How should we set the prices on the pricable edges in order to maximize the company's revenue? What if there is another customer, who needs to purchase, e.g., a minimum cost spanning tree?

This type of pricing problem, in which preferences are implicitly defined in terms of some optimization problem, is usually referred to as *Stackelberg pricing* [23]. In the standard 2-player form we are given a *leader* setting the prices on a subset of the network and a *follower* seeking to purchase a min-cost network satisfying her requirements. We proceed by formally defining the model before stating our results.

1.1 Model and Notation

In this paper we consider the following class of multi-player one-round games. Let $G = (V, E)$ be a multigraph. There are two types of players in the game, one *leader* and one or more *followers*. We consider two classes of *edge* and *vertex games*, in which either the edges or the vertices have costs. For most of the paper, we will consider edge games, but the definitions and results for vertex games follow analogously. In an edge game, the edge set E is divided into two sets $E = E_p \cup E_f$ with $E_p \cap E_f = \emptyset$. For the set of *fixed-price* edges E_f there is a fixed cost $c(e) \geq 0$ for each edge $e \in E_f$. For the set of *pricable* edges E_p the leader can specify a price $p(e) \geq 0$ for each edge $e \in E_p$. We denote the number of pricable edges by $m = |E_p|$. Each follower $i = 1, \ldots, k$ has a set $\mathcal{S}_i \subset 2^E$ of *feasible subnetworks*. The *weight* $w(S)$ of a subnetwork $S \in \mathcal{S}_i$ is given by the costs of fixed-price edges and the price of pricable edges,

$$w(S) = \sum_{e \in S \cap E_f} c(e) + \sum_{e \in S \cap E_p} p(e).$$

The *revenue* $r(S)$ of the leader from subnetwork S is given by the prices of the pricable edges that are included in S, i.e.,

$$r(S) = \sum_{e \in S \cap E_p} p(e).$$

Throughout the paper we assume that for any price function p every follower i can in polynomial time find a subnetwork $S_i^*(p)$ of minimum weight. Our interest is to find the pricing function p^* for the leader that generates maximum revenue, i.e.,

$$p^* = \arg\max_p \sum_{i=1}^{k} r(S_i^*(p)).$$

We denote this maximum revenue by r^*. To guarantee that the revenue is bounded and the optimization problem is non-trivial, we assume that there is at least one feasible subnetwork for each follower i that is composed only of fixed-price edges. In order to avoid technicalities, we assume w.l.o.g. that among subnetworks of identical weight the follower always chooses the one with higher revenue for the leader. It is not difficult to see that in the 2-player case we also need followers with a large number of feasible subnetworks in order to make the problem interesting.

Proposition 1 *Given follower j and a fixed subnetwork $S_j \in \mathcal{S}_j$, we can compute prices p with $w(S_j) = \min_{S\in\mathcal{S}_j} w(S)$ maximizing $r(S_j)$ or decide that such prices do not exist in polynomial time. In the 2-player game, if $|\mathcal{S}| = \mathcal{O}(poly(m))$, revenue maximization can be done in polynomial time.*

Proof: Fix follower j and subnetwork $S_j \in \mathcal{S}_j$. We formulate the problem of extracting maximum revenue from S_j as the following LP, where variable x_e defines the price of edge $e \in E_p$:

$$\begin{aligned} \text{max.} \quad & \sum_{e\in S_j\cap E_p} x_e && (1)\\ \text{s.t.} \quad & \sum_{e\in S_j\cap E_p} x_e + \sum_{e\in S_j\cap E_f} c(e) \le \sum_{e\in S\cap E_p} x_e + \sum_{e\in S\cap E_f} c(e) \quad \forall S \in \mathcal{S}_j && (2)\\ & x_e \ge 0 && (3) \end{aligned}$$

Constraints 2 require that S_j is the cheapest feasible network for follower j, formally $w(S_j) \le w(S)$ for all feasible networks $S \in \mathcal{S}_j$. Clearly the number of these constraints might be exponential in m. However, by our assumption we can compute the min-cost subnetwork for any given set of prices and, thus, have a poly-time separation oracle.

Now assume that $|\mathcal{S}| = \mathcal{O}(poly(m))$ in the 2-player case. By enumerating all $s \in \mathcal{S}$ and optimizing revenue for each subnetwork separately, we obtain a poly-time algorithm. □

In general we will refer to the revenue optimization problem by STACK. Note, that our model extends the previously considered pricing models and is essentially equivalent to pricing with general valuation functions, a problem that has independently been considered in [4]. Every general valuation function can be expressed in terms of Stackelberg network pricing on graphs (of potentially exponential size) and our algorithmic results apply in this setting, as well.

1.2 Previous Work and New Results

The single-follower shortest path Stackelberg pricing problem (STACKSP) has first been considered by Labbé et al. [17], who derive a bilevel LP formulation of the problem and prove NP-hardness. Roch et al. [19] present a first polynomial time approximation algorithm with a provable performance guarantee, which yields logarithmic approximation ratios. Bouhtou et al. [5] extend the problem to multiple (weighted) followers and present algorithms for a restricted shortest path problem on parallel links. For an overview of most of the initial work on Stackelberg network pricing the reader is referred to [22]. A different line of research has been investigating the application of Stackelberg pricing to network congestion games in order to obtain low congestion Nash equilibria for sets of selfish followers [11, 20, 21].

More recently, Cardinal et al. [9] initiated the investigation of the corresponding minimum spanning tree (StackMST) game, again obtaining a logarithmic approximation guarantee and proving APX-hardness. Their *single-price algorithm*, which assigns the same price to all pricable edges, turns out to be even more widely applicable and yields similar approximation guarantees for any matroid based Stackelberg game.

The first result of our paper is a generalization of this result to general Stackelberg games. The previous limitation to matroids stems from the difficulty to determine the necessarily polynomial number of candidate prices that can be tested by the algorithm. We develop a novel characterization of the small set of *threshold prices* that need to be tested and obtain a polynomial time $(1+\varepsilon)H_m$-approximation (where H_m denotes the m'th harmonic number) for arbitrary $\varepsilon > 0$, which turns out to be perfectly tight for shortest path as well as minimum spanning tree games. This result is found in Section 2.

We then extend the analysis to multiple followers, in which case the approximation ratio becomes $(1 + \varepsilon)(H_k + H_m)$. This can be shown to be essentially best possible by an approximation preserving reduction from single-minded combinatorial pricing [13]. Extending the problem even further, we also look at the case of multiple *weighted* followers, which arises naturally in network settings where different followers come with different routing demands. It has been conjectured before that no approximation essentially better than the number of followers is possible in this scenario. We disprove this conjecture by presenting an alternative analysis of the single-price algorithm resulting in an approximation ratio of $(1 + \varepsilon)m^2$. Additionally, we derive a lower bound of $\mathcal{O}(m^\varepsilon)$ for the weighted player case. This resolves a previously open problem from [5]. The results on multiple followers are found in Section 3.

The generic reduction from single-minded to Stackelberg pricing yields a class of networks in which we can price the vertices on one side of a bipartite graph and players aim to purchase minimum cost vertex covers for their sets of edges. This motivates us to return to the classical Stackelberg setting and consider the 2-player bipartite vertex cover game (StackVC). As it turns out, this variation of the game allows polynomial-time algorithms for exact revenue maximization using non-trivial algorithmic techniques. We first present an upper bound on the possible revenue in terms of the min-cost vertex cover not using any pricable vertices and the minimum portion of fixed cost in any possible cover. Using iterated max-flow computations, we then determine a pricing with total revenue that eventually coincides with our upper bound. These results are found in Section 4

The rest of the paper is organized as follows. Sections 2 through 4 contain our results on the single-price algorithm and the bipartite vertex cover game. Some of the proofs have been moved to the appendix due to space limitations. Section 5 concludes and presents several intriguing open problems for further research.

2 A Single-Price Algorithm for a Single Follower

Let us assume that we are faced with a single follower and let c_0 denote the cost of a cheapest feasible subnetwork for the follower not containing any of the pricable edges. Clearly, we can compute c_0 by assigning price $+\infty$ to all pricable edges and simulating the follower on the resulting network. The *single-price algorithm* proceeds as follows. For $j = 0, \ldots, \lceil \log c_0 \rceil$ it assigns price $p_j = (1 + \varepsilon)^j$ to all pricable edges and determines the resulting revenue $r(p_j)$. It then simply returns the pricing that results in maximum revenue. We present a logarithmic bound on the approximation guarantee of the single-price algorithm.

Theorem 1 *Given any $\varepsilon > 0$, the single-price algorithm computes an $(1+\varepsilon)H_m$-approximation with respect to r^*, the revenue of an optimal pricing.*

2.1 Analysis

The single-price algorithm has previously been applied to a number of different combinatorial pricing problems [1, 15]. The main issue in analyzing its performance guarantee for Stackelberg pricing is to determine the right set of candidate prices. We first derive a precise characterization of these candidates and then argue that the geometric sequence of prices tested by the algorithm is a good enough approximation. Slightly abusing notation, we let p refer to both price p and the assignment of this price to all pricable edges. If there exists a feasible subnetwork for the follower that uses at least j pricable edges, we let

$$\theta_j = \max\Big\{p \,\Big|\, |S^\star(p) \cap E_p| \geq j\Big\}$$

be the largest price at which such a subnetwork is chosen. If no feasible subnetwork with at least j pricable edges exists, we set $\theta_j = 0$. As we shall see, these thresholds are the key to prove Theorem 1.

We want to derive an alternative characterization of the values of θ_j. For each $1 \leq j \leq m$ we let c_j refer to the minimum sum of prices of fixed-price edges in any feasible subnetwork containing at most j pricable edges, formally

$$c_j = \min\Big\{\sum_{e \in S \cap E_f} f_e \,\Big|\, S \in \mathcal{S} \,:\, |S \cap E_p| \leq j\Big\},$$

and $\Delta_j = c_0 - c_j$. For ease of notation let $\Delta_0 = 0$. Consider the point set $(0, \Delta_0), (1, \Delta_1), \ldots, (m, \Delta_m)$ on the plane. By $\mathcal{H}$ we refer to a minimum selection of points spanning the upper convex hull of the point set. It is a straightforward geometric observation that we can define $\mathcal{H}$ as follows:

Fact 1 *Point (j, Δ_j) belongs to $\mathcal{H}$ if and only if $\min_{i<j} \frac{\Delta_j - \Delta_i}{j-i} > \max_{j<k} \frac{\Delta_k - \Delta_j}{k-j}$.*

We now return to the candidate prices. By definition we have that $\theta_1 \geq \theta_2 \geq \cdots \geq \theta_m$. We say that θ_j is *true threshold value* if $\theta_j > \theta_{j+1}$, i.e., if at price θ_j the subnetwork chosen by the follower contains exactly j pricable edges. Let $i_1 < i_2 < \cdots < i_\ell$ denote the indices, such that θ_{i_k} are true threshold values and for ease of notation define $i_0 = 0$.

Lemma 1 *θ_j is true threshold value if and only if (j, Δ_j) belongs to $\mathcal{H}$.*

Proof: "$\Rightarrow$" Let θ_j be true threshold value, i.e., at price θ_j the chosen subnetwork contains exactly j pricable edges. We observe that at any price p the cheapest subnetwork containing j pricable edges has cost $c_j + j \cdot p = c_0 - \Delta_j + j \cdot p$. Thus, at price θ_j it must be the case that $\Delta_j - j \cdot \theta_j \geq \Delta_i - i \cdot \theta_j$ for all $i < j$ and $\Delta_j - j \cdot \theta_j > \Delta_k - k \cdot \theta_j$ for all $j < k$. It follows that

$$\min_{i<j} \frac{\Delta_j - \Delta_i}{j - i} \geq \theta_j > \max_{j<k} \frac{\Delta_k - \Delta_j}{k - j},$$

and, thus, we have that (j, Δ_j) belongs to $\mathcal{H}$.

"$\Leftarrow$" Assume now that (j, Δ_j) belongs to $\mathcal{H}$ and let

$$p = \min_{i<j} \frac{\Delta_j - \Delta_i}{j - i}.$$

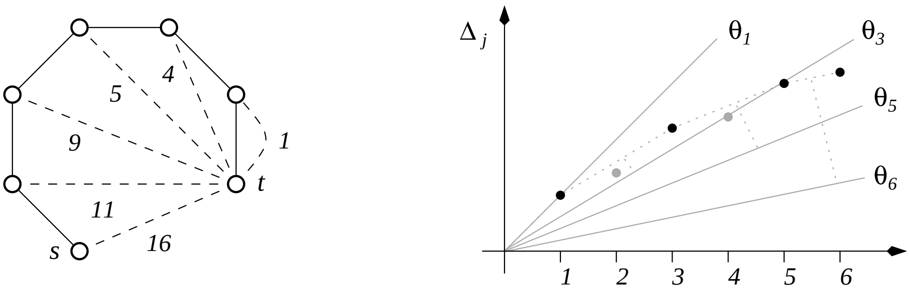


Figure 1: A geometric interpretation of (true) threshold values θ_j. The follower seeks to purchase a shortest path from s to t, dashed edges are fixed-cost.

Consider any $k < j$. It follows that $\Delta_k - k \cdot p = \Delta_j - j \cdot p - (\Delta_j - \Delta_k) + (j - k)p \leq \Delta_j - j \cdot p$, since $p \leq (\Delta_j - \Delta_k)/(j - k)$ and, thus, the network chosen at price p cannot contain less than j pricable edges. Analogously, let $k > j$. Using $p > (\Delta_k - \Delta_j)/(k - j)$ we obtain $\Delta_k - k \cdot p = \Delta_j - j \cdot p + (\Delta_k - \Delta_j) - (k - j)p < \Delta_j - j \cdot p$, and, thus, the subnetwork chosen at price p contains exactly j pricable edges. We conclude that θ_j is a true threshold. □

It is not difficult to see that the price p defined in the second part of the proof of Lemma 1 is precisely the threshold value θ_j. Let θ_{i_k} be any true threshold. Since points $(i_0, \Delta_{i_0}), \dots, (i_\ell, \Delta_{i_\ell})$ define the convex hull we can write that $\min_{i<i_k}(\Delta_{i_k} - \Delta_i)/(i_k - i) = (\Delta_{i_k} - \Delta_{i_{k-1}})/(i_k - i_{k-1})$. We state this important fact again in the following lemma.

Lemma 2 *For all $1 \leq k \leq \ell$ it holds that $\theta_{i_k} = \frac{\Delta_{i_k} - \Delta_{i_{k-1}}}{i_k - i_{k-1}}$.*

From the fact that points $(i_0, \Delta_{i_0}), \dots, (i_\ell, \Delta_{i_\ell})$ define the convex hull we know that $\Delta_{i_\ell} = \Delta_m$, i.e., Δ_{i_ℓ} is the largest of all Δ-values. On the other hand, each Δ_j describes the maximum revenue that can be made from a subnetwork with at most j pricable edges and, thus, Δ_m is clearly an upper bound on the revenue made by an optimal price assignment.

Fact 2 *It holds that $r^* \leq \Delta_{i_\ell}$.*

By definition of the θ_j's it is clear that at any price below θ_{i_k} the subnetwork chosen by the follower contains no less than i_k pricable edges. Furthermore, for each θ_{i_k} the single-price algorithm tests a candidate price that is at most a factor $(1 + \varepsilon)$ smaller than θ_{i_k}. Let $r(p_{i_k})$, $r(\theta_{i_k})$ denote the revenue that results from assigning price p_{i_k} or θ_{i_k} to all pricable edges, respectively.

Fact 3 *For each θ_{i_k} there exists a price p_{i_k} with $(1 + \varepsilon)^{-1}\theta_{i_k} \leq p_{i_k} \leq \theta_{i_k}$ that is tested by the single-price algorithm. Especially, it holds that $r(p_{i_k}) \geq (1 + \varepsilon)^{-1} r(\theta_{i_k})$*

Finally, we know that the revenue made by assigning price θ_{i_k} to all pricable edges is $r(\theta_{i_k}) = i_k \cdot \theta_{i_k}$. Let r

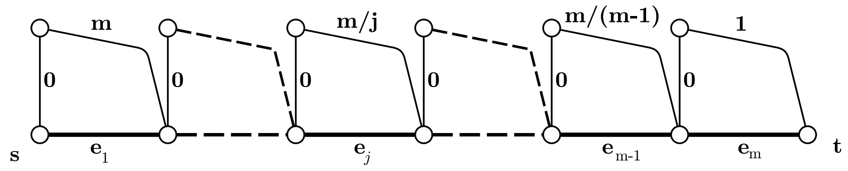


Figure 2: An instance of Stackelberg Shortest Path, on which the analysis of the approximation guarantee of the single-price algorithm is tight. Bold edges are pricable, vertex labels of regular edges indicate cost. The instance yields tightness of the analysis also for Stackelberg Minimum Spanning Tree.

denote the revenue of the single-price solution returned by the algorithm. We have:

$$\begin{aligned}(1+\varepsilon)\cdot H_m \cdot r &= (1+\varepsilon)\sum_{j=1}^{m}\frac{r}{j} \geq (1+\varepsilon)\sum_{k=1}^{\ell}\sum_{j=i_{k-1}+1}^{i_k}\frac{r}{j} \geq (1+\varepsilon)\sum_{k=1}^{\ell}\sum_{j=i_{k-1}+1}^{i_k}\frac{r(p_{i_k})}{j}\\ &\geq \sum_{k=1}^{\ell}\sum_{j=i_{k-1}+1}^{i_k}\frac{r(\theta_{i_k})}{j} \geq \sum_{k=1}^{\ell}\sum_{j=i_{k-1}+1}^{i_k}\frac{i_k\cdot\theta_{i_k}}{j}\\ &\geq \sum_{k=1}^{\ell}(i_k - i_{k-1})\frac{i_k\cdot\theta_{i_k}}{i_k} = \sum_{k=1}^{\ell}(\Delta_{i_k}-\Delta_{i_{k-1}}) \text{ , by Lemma 2}\\ &= \Delta_{i_\ell}-\Delta_0 = \Delta_{i_\ell} \geq r^*.\end{aligned}$$

This concludes the proof of Theorem 1.

2.2 Tightness

The example in Figure 2.1 shows that our analysis of the single-price algorithm's approximation guarantee is tight. The follower wants to buy a path connecting vertices s and t. In an optimal solution we set the price of edge e_j to m/j. Then edges $e_1,\ldots,e_m$ form a shortest path of cost mH_m. On the other hand, assume that all edges $e_1,\ldots,e_m$ are assigned the same price p. If $p \leq 1$ the leader's revenue is clearly bounded by m, if $p > m$ the shortest path does not contain any pricable edge at all. Let then $m/(j+1) < p \leq m/j$ for some $1 \leq j \leq m-1$. It is straightforward to argue that at this price a shortest path from s to t does not contain any of the pricable edges $e_{j+1},\ldots,e_m$ and, thus, it contains at most j pricable edges. It follows that the leader's revenue is at most $j\cdot p \leq m$. Similar argumentation clearly holds if the follower seeks to purchase a minimum spanning tree instead of a shortest path.

The best known lower bound for 2-player Stackelberg pricing is found in [9], where APX-hardness is shown for the minimum spanning tree case. To the authors' best knowledge, up to now no non-constant inapproximability results have been proven. We proceed by extending our results to multiple followers, in which case previous results on other combinatorial pricing problems yield strong lower bounds.

3 Extension to Multiple Followers

In this section we extend our results on general Stackelberg network pricing to scenarios with multiple followers. Recall that each follower j is characterized by her own collection $\mathcal{S}_j$ of feasible subnetworks and k

denotes the number of followers. Section 3.1 extends the analysis from the single follower case to prove a tight bound of $(1+\varepsilon)(H_k + H_m)$ on the approximation guarantee of the single-price algorithm. Section 3.2 presents an alternative analysis that applies even in the case of weighted followers and yields approximation guarantees that do not depend on the number of followers. Section 3.3 derives (near) tight inapproximability results based on known hardness results for combinatorial pricing.

3.1 An $(1+\varepsilon)(H_k + H_m)$-Approximation for Multiple Followers

Let an instance of Stackelberg network pricing with some number $k \geq 1$ of followers be given. We extend the analysis from Section 2 to obtain a similar bound on the single-price algorithm's approximation guarantee.

Theorem 2 *The single-price algorithm computes an $(1+\varepsilon)(H_k + H_m)$-approximation with respect to r^*, the revenue of an optimal pricing, for* STACK *with multiple followers.*

Proof: Consider graph $G = (V, E)$, $E = E_f \cup E_p$ with $|E_p| = m$, and k followers defined by collections $\mathcal{S}_1, \ldots, \mathcal{S}_k$ of feasible subnetworks. We transform this instance into a single follower pricing game as follows. Let $G_1, \ldots, G_k$ be identical copies of G and define $G^* = G_1 \cup \ldots \cup G_k$. Furthermore, define a single follower by

$$\mathcal{S}^* = \{S_1 \cup \ldots \cup S_k \mid S_1 \in \mathcal{S}_1 \cap G_1, \ldots, S_k \in \mathcal{S}_k \cap G_k\},$$

i.e., for every follower j in the original instance our new follower seeks to purchase a subnetwork from $\mathcal{S}_j$ in copy G_j of the original graph. Clearly, the maximum possible revenue in the new instance is an upper bound on the maximum revenue in the multiple follower case, since we can always assign the same price to every copy of a pricable edge in $G_1, \ldots, G_k$. Furthermore, every pricing returned by the single-price algorithm on $G_1 \cup \ldots \cup G_k$ translates naturally into a corresponding pricing of identical revenue in G, since again all copies of an edge from G are assigned identical prices. Finally, since the number of pricable edges in $G_1 \cup \ldots \cup G_k$ is $k \cdot m$, we obtain an approximation ratio of $(1+\varepsilon)H_{km}$ by Theorem 1 as desired. $\square$

The reduction from the multiple to single follower case in the proof of Theorem 2 relies essentially on the fact that we are considering the single-price algorithm. Thus, the above does not imply anything about the relation of these two cases in general.

3.2 A $(1+\varepsilon)m^2$-Approximation for Weighted Followers

We now turn to an even more general variation of Stackelberg pricing, in which we allow multiple *weighted* followers. This model, which has been previously considered in [5], arises naturally in the context of network pricing games with different demands for each player. Formally, for each follower j we are given her *demand* $d_j \in \mathbb{R}_0^+$. Given followers buying subnetworks $S_1, \ldots, S_k$, the leader's revenue is defined as

$$\sum_{j=1}^{k} d_j \sum_{e \in S_j \cap E_p} p(e).$$

It has been conjectured before that in the weighted case no approximation guarantee essentially beyond $\mathcal{O}(k \cdot \log m)$ is possible [19]. We show that an alternative analysis of the single-price algorithm yields ratios that do not depend on the number of followers.

Theorem 3 *The single-price algorithm computes an* $(1+\varepsilon)m^2$*-approximation with respect to* r^**, the revenue of an optimal pricing, for* STACK *with multiple weighted followers.*

Proof: Let again graph $G = (V, E)$, $E = E_f \cup E_p$ with $|E_p| = m$, and k followers defined by $\mathcal{S}_1, \ldots, \mathcal{S}_k$ and demands $d_1, \ldots, d_k$ be given and consider the optimal pricing p^*. For each pricable edge, let $F(e)$ refer to the set of followers purchasing e under price assignment p^* and denote by $r^*(e) = \sum_{j \in F(e)} d_j p^*(e)$ the corresponding revenue. Clearly, $\sum_{e \in E_p} r^*(e) = r^*$.

Fix some pricable edge e and define a corresponding price $p_e = p^*(e)/m$. By $r(p_e)$ we denote the revenue from assigning price p_e to all pricable edges. Let $j \in F(e)$ and assume that follower j buys subnetwork S_j under price assignment p^*. By $w^*(S_j)$, $w_e(S_j)$ and $c(S_j)$ we refer to the total weight of S_j under price assignments p^* and p_e and the weight due to fixed price edges only, respectively. It holds that

$$w_e(S_j) \leq c(S_j) + m\frac{p^*(e)}{m} = c(S_j) + p^*(e) \leq w^*(S_j).$$

Let c_j^0 denote the cost of a cheapest feasible subnetwork for follower j consisting only of fixed price edges. It follows that $w_e(S_j) \leq w^*(S_j) \leq c_j^0$ and, thus, follower j is going to purchase a subnetwork containing at least one pricable edge under price assignment p_e, resulting in revenue at least $d_j p_e = d_j p^*(e)/m$ from this follower. We conclude that $r(p_e) \geq r^*(e)/m$ and, thus

$$m^2 \max_{e \in E_p} r(p_e) \geq m \sum_{e \in E_p} r(p_e) \geq \sum_{e \in E_p} r^*(e) = r^*.$$

Finally, observe that for each price p_e the single-price algorithm checks some candidate price that is smaller by at most a factor of $(1+\varepsilon)$, which finishes the proof. $\square$

3.3 Lower Bounds

Hardness of approximation of Stackelberg pricing with multiple followers follows immediately from known results about other combinatorial pricing models, which have received considerable attention lately. Theorem 4 is based on a reduction from the (weighted) unit-demand envy-free pricing problem with uniform budgets, which is known to be inapproximable within $\mathcal{O}(m^\varepsilon)$ (m denotes the number of products) [6]. In this problem we are given a universe of products and a collection of (weighted) customers, each of which buys the cheapest product out of some set of alternatives with a price not exceeding her budget. The resulting Stackelberg pricing game is an instance of the so-called *river tarification problem*, in which each player needs to route her demand along one out of a number of parallel links connecting her respective source and sink pair. One direct fixed price connection determines her maximum budget for purchasing a pricable link. Theorem 4 resolves an open problem from [5]. The construction is depicted in Figure 3(a), a formal proof is omitted due to space limitations.

Theorem 4 *The Stackelberg network pricing problem with multiple weighted followers is hard to approximate within* $\mathcal{O}(m^\varepsilon)$ *for some* $\varepsilon > 0$*, unless NP* $\subseteq \bigcap_{\delta>0}$ *BPTIME*(2^{n^δ})*. The same holds for the river tarification problem.*

Theorem 5 is based on a reduction from the single-minded combinatorial pricing problem, in which each customer is interested in a subset of products and purchases the whole set if the sum of prices does not exceed her budget. Single-minded pricing is hard to approximate within $\mathcal{O}(\log^\varepsilon k + \log^\varepsilon m)$ [13], where k and m denote the numbers of customers and products, respectively. Theorem 5 shows that the single-price algorithm is essentially best possible for multiple unweighted followers.

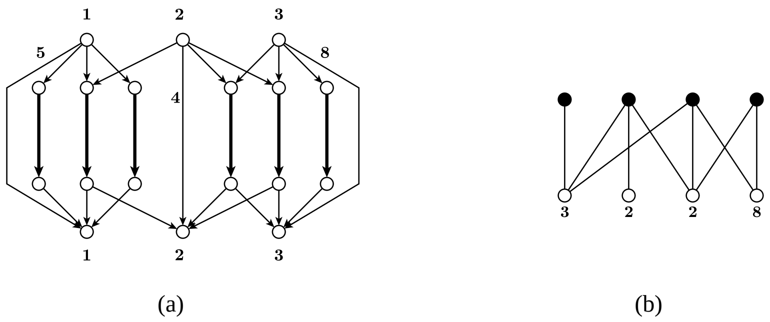


Figure 3: Reductions from pricing problems to Stackelberg pricing. (a) Unit-demand reduces to directed StackSP. Bold edges are pricable, edge labels indicate cost. Regular edges without labels have cost 0. Vertex labels indicate source-sink pairs for the followers. (b) Single-minded pricing reduces to bipartite StackVC. Filled vertices are pricable, vertex labels indicate cost. For each customer there is one follower, who strives to cover all incident edges.

Theorem 5 *The Stackelberg network pricing problem with multiple unweighted followers is hard to approximate within $\mathcal{O}(\log^\varepsilon k + \log^\varepsilon m)$ for some $\varepsilon > 0$, unless NP $\subseteq \bigcap_{\delta>0}$ BPTIME(2^{n^δ}). The same holds for bipartite Stackelberg Vertex Cover Pricing (*StackVC*).*

The idea for the proof of Theorem 5 is illustrated in Figure 3(b). We define an instance of StackVC in bipartite graphs. Vertices on one side of the bipartition are pricable and represent the universe of products, vertices on the other side encode customers and have fixed prices corresponding to the respective budgets. For each customer we define a follower in the Stackelberg game with edges connecting the customer vertex and all product vertices the customer wishes to purchase. Now every follower seeks to buy a min-cost vertex cover for her set of edges.

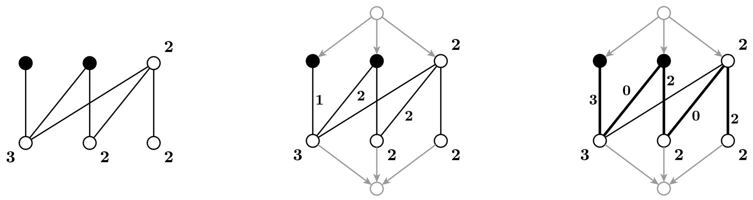
We proceed by taking a closer look at this special type of Stackelberg pricing game and especially focus on the interesting case of a single follower.

4 Stackelberg Vertex Cover

Stackelberg Vertex Cover Pricing is a vertex game, however, the approximation results for the single-price algorithm continue to hold. Note that in general the vertex cover problem is hard, hence we focus on settings, in which the problem can be solved in polynomial time. In bipartite graphs the problem can be solved optimally by using a classic and fundamental max-flow/min-cut argumentation. If all pricable vertices are in one side of the partition, then for multiple followers there is evidence that the single-price algorithm is essentially best possible. Our main theorem in this section states that the setting with a single follower can be solved exactly. As a consequence, general bipartite StackVC can be approximated by a factor of 2.

Theorem 6 *If for a bipartite graph $G = (A \cup B, E)$ we have $V_p \subseteq A$, then there is a polynomial time algorithm computing an optimal price function p^* for* StackVC.

We denote $n = |V_p|$ and again use the values c_j for $1 \le j \le n$ to denote the minimum sum of prices of fixed-price vertices in any feasible subnetwork containing at most j pricable vertices. Then, $\Delta_j = c_0 - c_j$ are again upper bounds on the revenue that can be extracted from a network that includes at most j pricable



(a) (b) (c)

Figure 4: Construction to solve bipartite STACKVC with pricable vertices in one partition and a single follower. Filled vertices are pricable, vertex labels indicate cost. (a) A graph G; (b) The flow network obtained from G. Grey parts are source and sink added by the transformation. Edge labels indicate a suboptimal s-t-flow; (c) An increasing path P indicated by bold edges and the resulting flow. Every path P starts with a pricable vertex, and all pricable vertices remain in the optimum cover at all times.

vertices. We thus have $r^* \leq \Delta_n$. Now consider STACKVC under the condition that the graph G is bipartite, i.e. that the vertex set can be partitioned into $V = A \cup B$ with $A \cap B = \emptyset$ and there are no edges within A and B. Thus, for each $e \in E$, $e = (u, v)$ there is $u \in A$ and $v \in B$.

Algorithm 1: Solving STACKVC in bipartite graphs with $V_p \subseteq A$

Construct the flow network G_f by adding nodes s and t
Set $p(v) = 0$ for all $v \in V_p$
Compute a maximum s-t-flow ϕ in G_d
while *there is* $v \in V_p$ *s.t. increasing* $p(v)$ *yields an augmenting path* P **do**
    Increase $p(v)$ and ϕ along P as much as possible

Suppose all pricable vertices are located in one partition $V_p \subseteq A$ and consider Algorithm 1. Recall that for a bipartite graph G the LP-dual can be captured by a maximum flow problem on an adjusted flow network G_d constructed as follows. We add a source s and a sink t to G and connect s to all vertices $v \in A$ with directed edges (s, v), and t to all vertices $v \in B$ with directed edges (v, t). Each such edge gets as capacity the price of the involved original vertex - i.e. $p(v)$ for $v \in V_p$ or $c(v)$ if $v \in V_f$. Furthermore, we direct all original edges of the graph from A to B and set their capacity to infinity. It is well-known that the maximum s-t-flow in this network equals the cost of a minimum cost vertex cover of the graph G. For an example see Figure 4. An augmenting path in G_d is a path traversing only forward edges with slack capacity and backward edges with non-zero flow. The optimum vertex cover includes a vertex $v \in A$ if the maximum flow allows no augmenting path from s to v. We denote by $\mathcal{C}_{ALG}$ the cover calculated by Algorithm 1.

Now consider a run of the algorithm. When computing the maximum flow on G_d holding all $p(v) = 0$, we get a flow of c_n. We first note that in the following while-loop we will never face a situation, in which there is an augmenting s-t-path starting with a fixed-price vertex. We call such a path a *fixed* path, while an augmenting s-t-path starting with a pricable vertex is called a *price* path.

Lemma 3 *Every augmenting path considered in the while-loop of Algorithm 1 is a price path.*

Proof: We prove the lemma by induction on the while-loop and by contradiction. Suppose that in the beginning of the current iteration there is no fixed path. In particular, this is true for the first iteration of the while-loop. Then, suppose that after we have increased the flow over a price path P_p, a fixed path P_f is created. P_f must include some of the edges of P_p. Consider the vertex w at which P_f hits P_p. By following

P_f from s to w and P_p from w to t there is a fixed path, which must have been present before flow was increased on P_p. This is a contradiction and proves the lemma. □

Note that we may include a vertex $v \in A$ into the cover $\mathcal{C}$ if there is no augmenting path from s to v. In particular, this means that for a vertex $v \in A \cap \mathcal{C}$ the following two properties are fulfilled:

1. the flow over edge (s, v) equals the capacity and
2. there is no augmenting path from s over a different vertex $v' \in A$ that reaches v by decreasing flow over one of the original edges (v, w) for $w \in B$.

As the algorithm always adjusts the price of a vertex v to equal the current flow on (s, v), we can assume that there is never any slack capacity on edges (s, v) for any $v \in V_p$. Thus, only the violation of property 2 can force a vertex $v \in V_p$ to leave the cover. In particular, such an augmenting path must start with a fixed-price vertex. We call such a path a fixed v-path.

Lemma 4 *Algorithm 1 creates no fixed v-path for any pricable vertex $v \in V_p$.*

Proof: The proof is similar to the proof of the previous lemma. Suppose in the beginning of an iteration there is no fixed path, and additionally for a vertex $v \in V_p$ there is no fixed v-path. Then suppose such a path P_f^v is created by increasing flow over a price path P_p. Note that P_f^v cannot include any edge from P_p, because this would create a fixed path P_f as noted in the previous lemma. Furthermore, v must be included in P_p, because otherwise P_f^v would have existed initially. Now we can again use the same argument as before. Create a fixed path by following P_f^v from s to v and then P_p from v to t. This yields fixed path must have existed initially, which is a contradiction to the assumption. □

As there is no augmenting path from s to any pricable vertex at any time of the algorithm, the following lemma is now obvious.

Lemma 5 *$\mathcal{C}_{ALG}$ includes all pricable vertices.*

Proof of Theorem 6. Finally, we can proceed to argue that the computed pricing is optimal. Suppose that after executing Algorithm 1 we increase $p(v)$ over $\phi(s, v)$ for any pricable vertex v. As we are at the end of the algorithm, it does not allow us to increase the flow in the same way. Thus, the adjustment creates slack capacity on all the edges (s, v) for any $v \in V_p$ and causes every pricable vertex to leave $\mathcal{C}_{ALG}$. The new cover must be the cheapest cover that excludes every pricable vertex, i.e. it must be $\mathcal{C}_0$ and have cost c_0. As we have not increased the flow, we know that the cost of $\mathcal{C}_{ALG}$ is also c_0. Note that before starting the while-loop the cover was $\mathcal{C}_n$ of cost c_n. As all flow increase in the while-loop was made over price paths and all the pricable vertices stay in the cover, the revenue of $\mathcal{C}_{ALG}$ must be $c_0 - c_n = \Delta_n$. This is an upper bound on the optimum revenue, and hence the price function p_{ALG} derived with the algorithm is optimal. Finally, notice that adjusting the price of the pricable vertices in each iteration is not necessary. We can start with computing $\mathcal{C}_n$ and for the remaining while-loop set all prices to $+\infty$. This will result in the desired flow, which directly generates the final price for every vertex v as flow on (s, v). Hence, we can get optimal prices with an adjusted run of the standard polynomial time algorithm for maximum flow in G_d. This proves Theorem 6. □

Algorithm 2 is a very natural extension of Algorithm 1 to the case of pricable vertices being located on both sides of the bipartition. Theorem 7 states that the algorithm achieves a 2-approximation in this situation.

Algorithm 2: A 2-approximation algorithm for STACKVC in bipartite graphs

Fix $p_A(v) = \infty$ for all $v \in V_p \cap B$
Fix $p_B(v) = \infty$ for all $v \in V_p \cap A$
Run Algorithm 1 to determine $p_A(v)$ for $v \in V_p \cap A$
Run Algorithm 1 to determine $p_B(v)$ for $v \in V_p \cap B$
Return p_A or p_B, depending on which one yields more revenue

Theorem 7 *Algorithm 2 is a 2-approximation algorithm for bipartite* STACKVC*, and the analysis of the ratio is tight.*

Proof: Note that by setting $p_A(v) = \infty$ for all pricable vertices of B, we increase their price over the prices in the optimum solution. This obviously allows us to extract more revenue from the vertices in A than p^*. The same argument applies for the vertices in B and p_B. Hence, the sum of both revenues is an upper bound on r^*, and our algorithm delivers a 2-approximation by preserving the greater of the two.

For a tight example consider a path $(v_1, v_2, v_3, v_4, v_5)$. The first vertex v_1 is a pricable vertex, then there are two fixed-price vertices v_2 and v_3 of cost 1 and 0, respectively. v_4 is pricable vertex, and v_5 has fixed cost 1. The optimum prices are $p(v_1) = p(v_3) = 1$. This yields the cover $\mathcal{C}^* = \{v_1, v_3, v_4\}$ and generates a revenue of 2. A solution returned by the algorithm, however, is e.g. $p(v_1) = 1$ and $p(v_2) = \infty$ (or vice versa), and hence generates only a revenue of 1. □

Note that Algorithm 2 can be used to obtain a $2k$-approximation for any number of k followers on general bipartite STACKVC. In contrast, the analysis of the single-price algorithm is tight even for one follower in the case, in which all pricable vertices are in one partition. Note further that a simple reduction from the highway pricing problem [7] can be used to show that bipartite STACKVC for at least two followers is NP-hard.

5 Open problems

In the model of Stackelberg games there are a number of important open problems that arise from our work. First, and foremost, we believe that the single-price algorithm is essentially best possible even for the single follower case and general Stackelberg pricing games. However, there is no matching logarithmic lower bound known for this case. The best lower bound remains APX-hardness from [9]. In addition, we believe that for the most general case of weighted followers a better bound than m^2 is possible. It remains an open problem how to tighten the gap between this bound and the $\Omega(m^\varepsilon)$ lower bound we observed.

We have experimented with problems that allow to be solved by dynamic programming, like certain classes of minimum knapsack or vertex cover on trees. It turns out that these algorithms can be modified to optimally solve Stackelberg revenue optimization. It would be interesting to see, whether a dynamic programming approach can be used for more general classes of problems.

More generally, extending other fundamental algorithm design techniques to cope with pricing problems is a major open problem. We have presented how ideas related to LP-duality can be used in the case of bipartite vertex cover. It remains to be shown if these ideas can be adjusted to cope with minimum cut or more general graph partitioning problems.

References

[1] G. Aggarwal, T. Feder, R. Motwani, and A. Zhu. Algorithms for Multi-Product Pricing. In *Proc. of 31st ICALP*, 2004.

[2] N. Balcan and A. Blum. Approximation Algorithms and Online Mechanisms for Item Pricing. In *Proc. of 7th EC*, 2006.

[3] N. Balcan, A. Blum, J. Hartline, and Y. Mansour. Mechanism Design via Machine Learning. In *Proc. of 46th FOCS*, 2005.

[4] M. Balcan, A. Blum, and Y. Mansour. Single Price Mechanisms for Revenue Maximization in Unlimited Supply Combinatorial Auctions. Technical Report *CMU-CS-07-111*, Carnegie Mellon University, 2007.

[5] M. Bouhtou, A. Grigoriev, S. van Hoesel, A. van der Kraaij, and M. Uetz. Pricing Network Edges to Cross a River. In *Proc. of 2nd WAOA*, 2004.

[6] P. Briest. Towards Hardness of Envy-Free Pricing. *ECCC Technical Report TR06-150*, 2006.

[7] P. Briest and P. Krysta. Single-Minded Unlimited-Supply Pricing on Sparse Instances. In *Proc. of 17th SODA*, 2006.

[8] P. Briest and P. Krysta. Buying Cheap is Expensive: Hardness of Non-Parametric Multi-Product Pricing. In *Proc. of 18th SODA*, 2007.

[9] J. Cardinal, E. Demaine, S. Fiorini, G. Joret, S. Langerman, I. Newman, and O. Weimann. The Stackelberg Minimum Spanning Tree Game. In *Proc. of 10th WADS*, 2007. To appear.

[10] S. Chawla, J. Hartline, and R. Kleinberg. Algorithmic Pricing via Virtual Valuations. In *Proc. of 8th EC*, 2007.

[11] R. Cole, Y. Dodis, and T. Roughgarden. Pricing Network Edges for Heterogeneous Selfish Users. In *Proc. of 35th STOC*, 2003.

[12] P. Cramton, Y. Shoham, and R. Steinberg (Editors). *Combinatorial Auctions.* MIT Press, 2006.

[13] E.D. Demaine, U. Feige, M.T. Hajiaghayi, and M.R. Salavatipour. Combination Can Be Hard: Approximability of the Unique Coverage Problem. In *Proc. of 17th SODA*, 2006.

[14] P. Glynn, P. Rusmevichientong, and B. Van Roy. A Non-Parametric Approach to Multi-Product Pricing. *Operations Research*, **54**(1):82–98, 2006.

[15] V. Guruswami, J.D. Hartline, A.R. Karlin, D. Kempe, C. Kenyon, and F. McSherry. On Profit-Maximizing Envy-Free Pricing. In *Proc. of 16th SODA*, 2005.

[16] J. Hartline and V. Koltun. Near-Optimal Pricing in Near-Linear Time. In *Proc. of 8th WADS*, 2005.

[17] M. Labbé, P. Marcotte, and G. Savard. A Bilevel Model of Taxation and its Application to Optimal Highway Pricing. *Management Science*, 44(12): 1608–1622, 1998.

[18] N. Nisan and A. Ronen. Algorithmic Mechanism Design. In *Proc. of 31st STOC*, 1999.

[19] S. Roch, G. Savard, and P. Marcotte. An Approximation Algorithm for Stackelberg Network Pricing. *Networks*, 46(1): 57–67, 2005.

[20] T. Roughgarden. Stackelberg Scheduling Strategies. *SIAM Journal on Computing*, 33(2): 332–350, 2004.

[21] C. Swamy. The Effectiveness of Stackelberg Strategies and Tolls for Network Congestion Games. In *Proc. of 18th SODA*, 2007.

[22] S. van Hoesel. An Overview of Stackelberg Pricing in Networks. Research Memoranda 042, METEOR, Maastricht, 2006.

[23] H. von Stackelberg. *Marktform und Gleichgewicht (Market and Equilibrium)*. Verlag von Julius Springer, Vienna, 1934.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/