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 priceable 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 priceable 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. This can be extended to provide a $(1+\varepsilon)(\log k + \log m)$-approximation for the general problem and k followers. The problem is also shown to be hard to approximate within $\mathcal{O}(\log^{\varepsilon} k + \log^{\varepsilon} m)$ for some $\varepsilon > 0$. If followers have demands, the single-price algorithm provides an $\mathcal{O}(m^2)$-approximation, and the problem is hard to approximate within $\mathcal{O}(m^{\epsilon})$ 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. This approach can be extended to provide constant-factor approximations for any constant number of followers.

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 [19], 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, 11, 20, 21] and every customer is interested in exactly one subset of products or *pure substitutes* [1, 12, 15, 19, 20, 21], 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* [25] and *combinatorial auctions* [16]. The established models range from relatively

*An extended abstract of this paper has appeared in STACS 2008 [9].

†Department of Computer Science, University of Paderborn, Germany. patrick.briest@upb.de.

‡Department of Computer Science, RWTH Aachen University, Germany. mhoefer@cs.rwth-aachen.de. Supported by DFG Graduiertenkolleg "AlgoSyn".

§Department of Computer Science, The University of Liverpool, United Kingdom. P.J.Krysta@csc.liv.ac.uk. Supported by DFG grant Kr 2332/1-2 within Emmy Noether program.

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 decsribed by their *types* and *budgets* and do not require any kind of oracles. We consider the framework of *Stackelberg pricing* [30], which models scenarios that arise naturally in a variety of combinatorial pricing problems in practice. This approach originates in the operations research literature [24].

For instance, consider a simple pricing problem to optimally set tolls in a traffic network. There is a company that owns a certain subset of road segments (like speedways or private highways) on which the company can set prices. Customers are cars travelling between their source and destination through the network. Let us disregard effects of congestion and assume that every non-priceable edge has a fixed publicly known cost, which could come from a (constant) travel time or a toll that a competitor company places on this edge. Customers are rational, i.e., they choose a shortest path with respect to the sum of edge costs on the path. How can the company price their road segments in order to achieve highest revenue?

As another example consider a telecommunication company that provides certain connections in a network. In the market there are competitors of this company that operate similar or different connections, for which users must pay fixed publicly known prices. In this case possible customers are interested in establishing connections via the network (e.g., setting up telephone or internet connections) for which they must pay the corresponding prices. Again, as customers are acting rationally, they are going to purchase the cheapest set of connections that meets their communication requirement. How can the company set prices for the connections such that it receives the highest revenue from the customers?

Note, that in both cases customer preferences are represented implicitly via some network design problem and the customers' objective of minimizing total cost. The revenue optimization problem is then to manipulate the prices of network edges to obtain the largest share of revenue in the optimum solution of the customer. In general, this approach of defining customer preferences implicitly in terms of some optimization problem is the characteristic of Stackelberg pricing. In the standard 2-player form we are given a *leader* setting the prices on a subset of network elements and a *follower* seeking to purchase a min-cost network satisfying her requirements. In general, this gives rise to a bi-level optimization problem for the revenue of the leader. 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 are completely analogous. 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 *priceable* 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 priceable 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 priceable edges,

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

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

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

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. Hence, we further assume that the sets of $\mathcal{S}_i$ are not part of the input but can be represented in a compact way, e.g., by a logic formula or a set of constraints that is polynomial in the size of G and k. Our interest is to find the pricing function p^* for the leader that generates maximum revenue, i.e.,

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

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. In general we will refer to the revenue optimization problem by STACK.

Problem STACK**:** Given a graph $G = (V, E)$, a subset $E_p \subset E$ of priceable edges, fixed costs $c(e)$ for $e \in E - E_p$, and k followers with follower i specified by a compact representation of her feasible subnetworks $\mathcal{S}_i \subseteq 2^E$, find prices $p(e)$ for all $e \in E_p$ such that the revenue $\sum_{i=1}^{k} r(S_i^*(p))$ is maximized.

It is not difficult to see that for games with $k = 1$ follower, we need a follower with a large number of feasible subnetworks in order to make STACK 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. For* STACK *with $k = 1$ follower, if $|\mathcal{S}| = \mathcal{O}(poly(m))$, then 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 $e \in E_p$:

$$\text{max.} \quad \sum_{e \in S_j \cap E_p} x_e \tag{1}$$

$$\text{s.t.} \quad \sum_{e \in S_j \cap E_p} x_e + \sum_{e \in S_j \cap E_f} c(e) \leq \sum_{e \in S \cap E_p} x_e + \sum_{e \in S \cap E_f} c(e) \quad \forall S \in \mathcal{S}_j \tag{2}$$

$$x_e \geq 0 \tag{3}$$

Constraints 2 require that S_j is the cheapest feasible network for follower j, formally $w(S_j) \leq w(S)$ 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 polynomial time separation oracle.

Now assume that $|\mathcal{S}| = \mathcal{O}(poly(m))$ for an instance of STACK with $k = 1$ follower. By enumerating all $S \in \mathcal{S}$ and optimizing revenue for each subnetwork separately, we obtain a polynomial time algorithm. □

Note, that our definition of Stackelberg pricing generalizes most of the previously employed models in algorithmic pricing. In particular, it is equivalent to item 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 and our algorithmic results apply in this setting, as well.

While we will start by addressing the general case of STACK, we will also focus on more restricted variants, in which we can use the additional problem structure to show improved results. As an example we consider the *bipartite Stackelberg vertex cover game*. In this game we are given a bipartite graph $G = (A \cup B, E)$ with a subset of *priceable vertices* $V_p \subset V = A \cup B$ and fixed costs $c_f(v)$ on the remaining vertices $v \in V_f = V - V_p$. Hence, the game is defined as a vertex game rather than an edge game. Each follower seeks to purchase a min-cost vertex cover of some subset of the graph's edges E. In particular, follower i has a set $E_i \subseteq E$ of edges, and her feasible subsets $\mathcal{S}_i \subseteq 2^V$ are the subsets of vertices that form vertex covers of E_i. The *weight* $w(S)$ of a subset $S \in \mathcal{S}_i$ is again given by

$$w(S) = \sum_{v \in S \cap V_f} c(v) + \sum_{v \in S \cap V_p} p(v) \quad .$$

The *revenue* $r(S)$ of the leader is given by the prices of the priceable vertices included in S, i.e.,

$$r(S) = \sum_{v \in S \cap V_p} p(v) \quad .$$

Note that the set $\mathcal{S}_i$ of all vertex covers for a follower i can be described by a number of linear constraints that is polynomial in the size of G. For any price function p follower i can in polynomial time find a minimum-weight vertex cover $S_i^*(p)$ for E_i using a well-known algorithm based on max-flow computations. Our interest is to find the pricing function p^* for the leader that generates maximum revenue.

Problem STACKVC**:** Given a bipartite graph $G = (V, E)$, a subset V_p of priceable vertices, fixed costs $c(v)$ for $v \in V - V_p$, and k followers with follower i specified by edge set $E_i \subseteq E$, find prices $p(v)$ for all $v \in V_p$ such that the revenue $\sum_{i=1}^{k} r(S_i^*(p))$ is maximized.

1.2 Previous Work and New Results

The single-follower shortest-path Stackelberg pricing problem (STACKSP) has first been considered by Labbé et al. [24], who derive a bilevel LP formulation of the problem and prove NP-hardness. Roch et al. [26] present a first polynomial time approximation algorithm with a provable performance guarantee, which yields logarithmic approximation ratios. Bouhtou et al. [6] 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 [29]. A different variant

called the shortest-path-tree game, in which a customer purchases the shortest path tree from a node to every other node in the network, was studied by Biló et al. [5]. They solve the pricing problem optimally in time $O(n^2 \log n)$ for 2 priceable edges. 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 [18, 22, 23, 27, 28, 31].

Recently, Cardinal et al. [13] investigated 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 priceable edges, turns out to be even more widely applicable and yields similar approximation guarantees for any matroid based Stackelberg game. Very recently, in [14] they provided constant-factor approximation algorithms based on dynamic programming for STACKMST in the special case of planar and bounded-treewidth graphs.

The first result of our paper is a generalization of the single-price algorithm 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 [17]. Extending the problem even further, we also look at the case of multiple *weighted* followers, which arises naturaly in network settings where different followers come with different routing demands. While previous upper-bounding techniques could not yield approximation guarantees essentially better than the number of followers in this scenario, we present an alternative analysis of the single-price algorithm that results 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 [6]. 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 single-follower 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 priceable 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.

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 priceable edges. Clearly, we can compute c_0 by

assigning price $+\infty$ to all priceable 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 priceable 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 for Stackelberg network pricing 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, 20]. 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 priceable edges. Consider the follower's cheapest feasible subnetworks $S^*(p)$ for different values of p. If there exists any feasible subnetwork that uses at least j priceable edges, we let

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

be the largest price at which such a subnetwork is chosen. If no feasible subnetwork with at least j priceable 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 priceable edges, formally

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

Note, that clearly $c_0 \geq c_1 \geq \cdots \geq c_m$ by definition. Now let $\Delta_j = c_0 - c_j$ and 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 priceable 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: "⇒" Let θ_j be true threshold value, i.e., at price θ_j the chosen subnetwork contains exactly j priceable edges. We observe that at any price p the cheapest subnetwork containing j priceable 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 $\mathcal{H}$.

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

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

Consider any $k < j$. It follows that

$$\Delta_k - k \cdot p = \Delta_j - j \cdot p - (\Delta_j - \Delta_k) + (j - k)p \leq \Delta_j - j \cdot p \quad ,$$

since $p \leq (\Delta_j - \Delta_k)/(j - k)$ and, thus, the network chosen at price p cannot contain less than j priceable edges. Analogously, let $k > j$. Using $p > (\Delta_k - \Delta_j)/(k - j)$ we obtain

$$\Delta_k - k \cdot p = \Delta_j - j \cdot p + (\Delta_k - \Delta_j) - (k - j)p < \Delta_j - j \cdot p \quad ,$$

and, thus, the subnetwork chosen at price p contains exactly j priceable edges. We conclude that θ_j is a true threshold. □

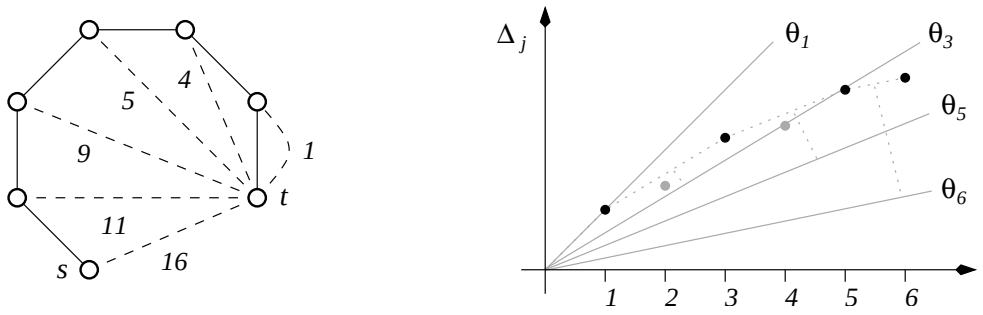


Figure 1: A geometric interpretation of (true) threshold values. Price θ_{i_k} corresponds to the slope of the segment of the upper convex hull of point set $(0, \Delta_0), (1, \Delta_1), \ldots, (m, \Delta_m)$ between $(i_{k-1}, \Delta_{i_{k-1}})$ and (i_k, Δ_{i_k}).

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 $(i_0, \Delta_{i_0}), \ldots, (i_\ell, \Delta_{i_\ell})$ 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, which is also illustrated in Fig. 1, again in the following lemma.

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

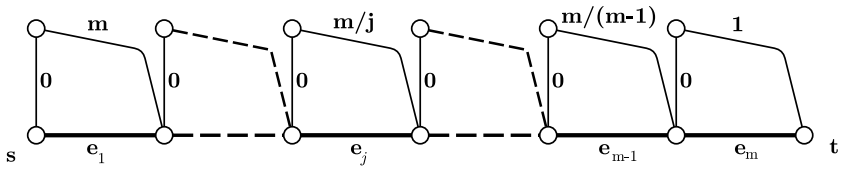


Figure 2: An instance of Stackelberg Shortest Path, on which the analysis of the approximation guarantee of the single-price algorithm is tight. Bold edges are priceable, vertex labels of regular edges indicate cost. The instance yields tightness of the analysis also for Stackelberg Minimum Spanning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

From the fact that points $(i_0, \Delta_{i_0}), \ldots, (i_\ell, \Delta_{i_\ell})$ 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 priceable 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 priceable 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 priceable 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 priceable edges is $r(\theta_{i_k}) = i_k \cdot \theta_{i_k}$. Let r 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 Fig. 2 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 priceable 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 priceable edges $e_{j+1}, \ldots, e_m$ and, thus, it contains at most j priceable 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 single-follower Stackelberg pricing is found in [10], where STACKSP is shown to be non-approximable within $2 - o(1)$. For the spanning tree case the best lower bound is APX-hardness shown in [13]. To the authors' best knowledge, up to now no non-constant inapproximability results could be 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 aproximation 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 Stackelberg network pricing 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

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

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 priceable 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 priceable edges in $G_1 \cup \ldots \cup G_k$ is $k \cdot m$, we obtain an approximation ratio of $(1+\varepsilon)H_{km}$ as desired. □

This reduction from the multiple to single follower case 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 [6], arises naturally in the context of network pricing games with different demands for each player. Formally, for each follower 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

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

It has been conjectured before that in the weighted case no approximation guarantee essentially beyond $\mathcal{O}(k \cdot \log m)$ is possible. 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 Stackelberg network pricing 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 priceable 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 priceable 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 priceable 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

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

Let c_0^j denote the cost of a cheapest feasible subnetwork for follower j consisting only of fixed price edges. It follows that $w_e(S_j) \leq w^*(S_j) \leq c_0^j$ and, thus, follower j is going to purchase a subnetwork containing at least one priceable 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

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

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. □

3.3 Lower Bounds

Hardness of approximation of Stackelberg pricing with multiple followers follows quite easily from known results about other combinatorial pricing models, which have received considerable attention lately. More formally, we will show lower bounds on the approximability of both weighted and unweighted multi-follower Stackelberg pricing games expressed in the number of priceable edges m and the number of followers k based on hardness of the unit-demand and single-minded envy-free pricing problems, respectively. Theorem 4 is based on a reduction from the unit-demand version of envy-free pricing. 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. This resolves an open problem from [6].

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($2^{\mathcal{O}(n^{\delta})}$). *The same holds for the river tarification problem.*

Proof: The distribution-based unit-demand envy-free pricing problem with uniform budgets (UDP-MIN) is defined as follows. We are given a universe $\mathcal{U}$, $|\mathcal{U}| = m$, of products and a set of consumers $\mathcal{C}$, where each $c \in \mathcal{C}$ is defined by her budget $b_c \in \mathbb{R}_0^+$ and the set $S_c \subseteq \mathcal{U}$ of products she is interested in. Additionally, probability distribution $\mathcal{D}$ assigns to each consumer c a probability $\Pr_{\mathcal{D}}(c)$. We want to find product prices $p : \mathcal{U} \to \mathbb{R}_0^+$ maximizing the expected revenue

$$r(p) = \sum_{c \in \mathcal{C}} \Pr_{\mathcal{D}}(c) \cdot \min_{u \in A_c(p)} p(u)$$

from a sale to a consumer sampled according to $\mathcal{D}$, where $A_c(p) = \{u \in \mathcal{U} \,|\, u \in S_c \wedge p(u) \leq b_c\}$ denotes the set of products c can afford under price assignment p and we define $\min_{u \in A_c(p)} p(u) = 0$ whenever $A_c(p) = \emptyset$. UDP-MIN is hard to approximate within $\mathcal{O}(m^{\varepsilon})$ for some $\varepsilon > 0$, unless NP $\subseteq \bigcap_{\delta>0}$ BPTIME($2^{\mathcal{O}(n^{\delta})}$) [7].

We can encode UDP-MIN in terms of a weighted multi-follower shortest-path Stackelberg pricing game in a directed graph in a straightforward way. For every product $u \in \mathcal{U}$ we define two vertices v_u, w_u and the directed priceable edge (v_u, w_u). For each consumer $c \in \mathcal{C}$ we add vertices s_c, t_c, directed fixed-price edge (s_c, t_c) of cost b_c and directed fixed-price edges (s_c, v_u), (w_u, t_c) of cost 0 for every product $u \in S_c$. We then define a follower seeking to route a total demand of $d_c = \Pr_{\mathcal{D}}(c)$ along a shortest path from s_c to t_c.

Obviously the leader's revenue from assigning any prices to the priceable edges equals the expected revenue from assigning the same prices to the products of the UDP-MIN instance. Furthermore, the simple structure of the constructed graph satisfies the requirements of the river tarification problem from [6]. The construction is depicted in Figure 3(a). □

In the unweighted case, a reduction from the single-minded version of the envy-free pricing problem yields lower bounds on the approximability of multi-follower Stackelberg pricing games. Theorem 5 shows that the single-price algorithm is essentially best possible in this situation. The resulting pricing game is an instance of bipartite Stackelberg Vertex Cover Pricing and, thus, yields the same result for this special case.

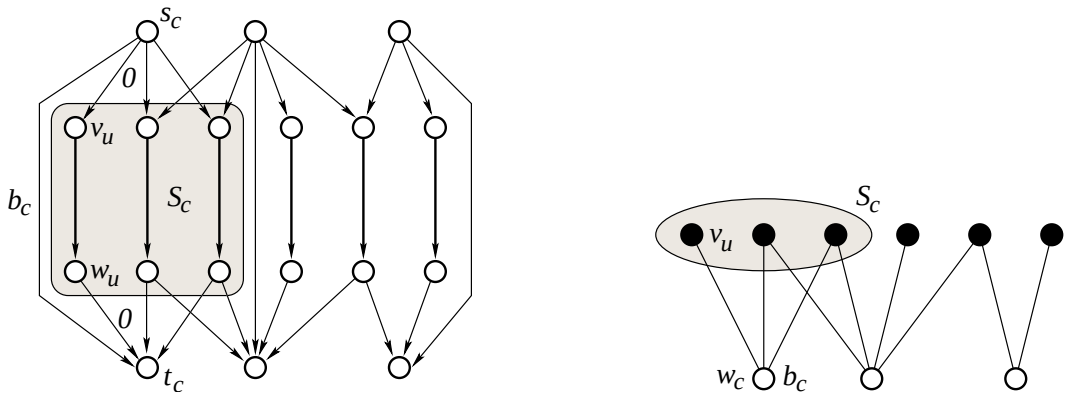


Figure 3: Reductions from pricing problems to Stackelberg pricing. (a) Unit-demand reduces to directed STACKSP. Bold edges are priceable, edge labels indicate cost. Regular edges without labels have cost 0. Vertex labels indicate source-sink pairs for the followers. (b) Single-minded pricing reduces to bipartite STACKVC. Solid vertices are priceable, vertex labels indicate cost. For each customer there is one fixed-cost vertex and a corresponding follower, who needs to cover all the 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($2^{\mathcal{O}(n^{\delta})}$). *In particular, this holds for bipartite Stackelberg Vertex Cover Pricing.*

Proof: We prove the theorem by a reduction from the single-minded envy-free pricing problem (SMP). In this problem, given a universe of products $\mathcal{U}$, $|\mathcal{U}| = m$, and a set of consumer samples $\mathcal{C}$, $|\mathcal{C}| = k$, consisting of budgets $b_c \in \mathbb{R}_0^+$ and product sets $S_c \subseteq \mathcal{U}$, we need to find prices $p : \mathcal{U} \to \mathbb{R}_0^+$ maximizing the revenue

$$r(p) = \sum_{c \in \mathcal{C}: p(S_c) \leq b_c} p(S_c)$$

from sales to consumers $\mathcal{C}$, where $p(S_c) = \sum_{u \in S_c} p(u)$ is shorthand notation for the sum of prices of products in S_c. Intuitively, each consumer in SMP buys the whole set of products she is interested in, if the sum of prices does not exceed her budget. SMP is hard to approximate within $\mathcal{O}(\log^{\varepsilon} k + \log^{\varepsilon} m)$ for some $\varepsilon > 0$, unless NP $\subseteq \bigcap_{\delta>0}$ BPTIME($2^{\mathcal{O}(n^{\delta})}$) [17].

We encode SMP in terms of a Stackelberg vertex pricing game as follows. For every product $u \in \mathcal{U}$ we define a priceable vertex v_u. For each consumer $c \in \mathcal{C}$ we add a fixed-price vertex w_c of cost b_c and edges $\{w_c, v_u\}$ for every product $u \in S_c$. We then define a follower seeking to purchase a min-cost vertex-cover for the edges connected to w_c.

It is straightforward to check that the follower corresponding to consumer c in the SMP instance purchases the priceable vertices corresponding to the products in her set, if and only if their assigned prices sum to at most b_c. We observe that the constructed graph is clearly bipartite and, furthermore, all priceable vertices are located on one side of the bipartition. The construction is illustrated in Figure 3(b). □

We proceed by taking a closer look at STACKVC and especially focus on the interesting case of a single follower.

4 Stackelberg Vertex Cover

Stackelberg Vertex Cover Pricing is a vertex game. Nevertheless, the approximation results for the single-price algorithm, which are completely independent of the underlying network structure, continue to hold. In general the vertex-cover problem is hard and, consequently, we focus on settings in which the problem can be solved in polynomial time in order to stay within our definition of general Stackelberg network pricing games laid out in Section 1.1. In bipartite graphs the problem can be solved optimally by using a classic and fundamental max-flow/min-cut argumentation. If all priceable vertices are in one side of the partition, we have shown evidence that for multiple followers the single-price algorithm is essentially best possible. Our main result in this section is the fact that the problem can be solved exactly if there is only a single follower. This is despite the fact that the follower's set of feasible vertex covers might of course still be of exponential size and so the enumeration approach sketched in Section Proposition 1 is infeasible in this setting. A simple extension of our algorithm shows that general bipartite STACKVCwith a single follower can be approximated within a factor of 2.

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

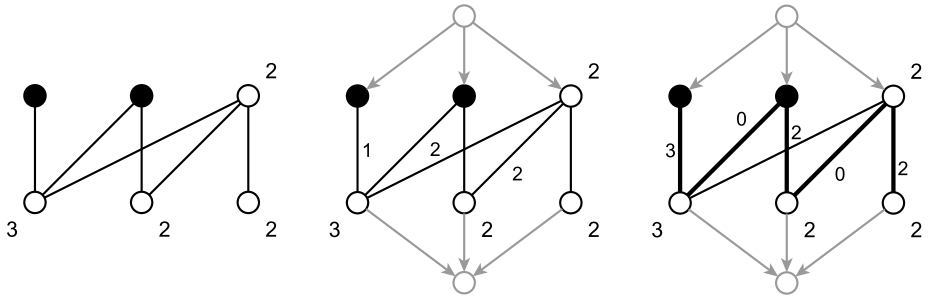
Assume that the prices $p(v)$ of all priceable vertices $v \in V_p$ are fixed. Lemma 3, which is essentially folklore by now, briefly describes how the follower can find a min-cost vertex-cover in this setting using a max-flow approach. Given the bipartite graph $G = (A \cup B, E)$, we define the corresponding *flow network* G_f 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.

Given an s-t-flow ϕ on G_f, we define the *residual network* G_r in the standard fashion. If edge (v, w) in G_f has a remaining free capacity c, G_r contains an edge (v, w) of capacity c. Additionally, if (v, w) carries flow $\phi(v, w) > 0$ in G_f, then G_r contains edge (w, v) of capacity $\phi(v, w)$. Finally, by an augmenting path we refer to an s-t-path in G_r.

Lemma 3 *Given a maximum s-t-flow f on G_f, we obtain a min-cost vertex-cover of G by selecting all vertices in A that are disconnected from s in G_r and all vertices from B which can be reached from s in G_r.*

Proof: Let V_A and V_B the selections of vertices described above, i.e., V_A contains all vertices $v \in A$ for which no directed s-v-path exists in G_r, V_B contains all vertices $v \in B$ for which such a path exists. We first argue that $V_A \cup V_B$ is indeed a feasible vertex-cover. Towards a contradiction, assume that there is an uncovered edge $\{v, w\}$ in G, thus, $v \in A \backslash V_A$ and $w \in B \backslash V_B$. Then there is an edge (v, w) of infinite capacity in G_r. Since $v \in A \backslash V_A$, there is an s-v-path in G_r and by adding edge (v, w) we obtain an s-w-path, as well. It follows that $w \in V_B$, a contradiction.

Optimality of the constructed vertex-cover can be seen as follows. First, we observe that the cost of the vertex-cover equals the total value of flow f. This is immediate by considering the cut of G_f defined by $(\{s\} \cup A \backslash V_A \cup V_B, \{t\} \cup V_A \cup B \backslash V_B)$ and applying the max-flow/min-cut theorem. One can then argue that the max-flow problem on G_f corresponds exactly to the dual of the LP-relaxation of the min-cost vertex-cover problem on G, and the claim follows. We omit the details of this part of the proof. □



(a) (b) (c)

Figure 4: Construction to solve bipartite STACKVC with priceable vertices in one partition and a single follower. Solid vertices are priceable, 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 augmenting path P indicated by bold edges and the resulting flow. Every path P starts with a priceable vertex, and all priceable vertices remain in the optimum cover at all times.

Algorithm 1 is based on the idea of computing a sequence of flows on network G_f. Throughout the iterations, we interpret the flow on edge (s, v) for a priceable vertex $v \in V_p$ as the current price of v. As we will see, Algorithm 1 terminates with the optimum set of vertex prices. The key to the analysis lies in showing that at any time all priceable vertices are part of the min-cost vertex-cover. As we have argued before, this is equivalent to saying that all priceable vertices are in V_A (disconnected from s in G_r) at all times.

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

```
1 Construct the flow network G_f by adding nodes s and t.
2 Set p(v) = 0 for all v ∈ V_p.
3 Compute a maximum s-t-flow φ in G_f.
4 while there is v ∈ V_p s.t. increasing p(v) yields an augmenting path P do
5     Increase p(v) and φ along P as much as possible.
```

Lemma 4 *For every priceable vertex $v \in V_p$ and all prices $p(v)$ and flows ϕ computed by Algorithm 1, it holds that $v \in V_A$ in the corresponding residual network G_r.*

Proof: The claim clearly holds in the initial round with $p(v) = 0$ for all $v \in V_p$, since in this situation edge (s, v) has capacity 0 and all edges leaving v carry zero flow. We then only need to argue that the property is preserved through a single iteration of the algorithm.

We first observe that all computed flows are maximal with respect to the current capacities. To see this, note, that in each iteration the capacity of a single edge and the overall flow are increased by the same margin. Maximality is then immediate by the max-flow/min-cut theorem.

It follows that before and after each iteration, there is no augmenting path in G_r and, thus, G_r is disconnected. Now assume that in an iteration the algorithm increases the capacity of edge (s, v) for a given $v \in V_p$ and this yields an augmenting path $P = (s, v, w_1, \ldots, w_k, t)$. Since there was no s-t-path before increasing

the capacity on (s, v), there is no other s-w_1-path except the one through v. This implies that assigning flow to edge (v, w_1) does not create an s-v-path in G_r. The exact same argument can be applied to any other priceable vertex on the augmenting path, and the claim follows. □

We denote $n = |V_p|$ and again use the values c_j for $1 \leq j \leq n$ to denote the minimum sum of prices of fixed-price vertices in any feasible vertex-cover containing at most j priceable 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 priceable vertices. For the optimal achievable revenue r^* we have $r^* \leq \Delta_n$.

When computing the maximum flow on G_f holding all $p(v) = 0$, we get an initial flow of c_n. In order to prove optimality of Algorithm 1, we now only need to calculate the value of the flow computed in the final iteration. By $\mathcal{C}_{ALG}$ we refer to the min-cost vertex-cover with respect to the prices computed by the algorithm. $\mathcal{C}_0$ and $\mathcal{C}_n$ denote the min-cost vertex-covers with the prices of all priceable vertices set to $+\infty$ or 0, respectively.

Proof:[of Theorem 6] Suppose that after executing Algorithm 1 we increase $p(v)$ above $\phi(s, v)$ for every priceable vertex v. As we are at the end of the algorithm, this does not create any augmenting path and, thus, does not allow us to increase the flow any further. Consequently, the adjustment creates slack capacity on all the edges (s, v) with $v \in V_p$ and causes all priceable vertices to leave $\mathcal{C}_{ALG}$. The new cover must be the cheapest cover that excludes all priceable vertices, i.e., it must be $\mathcal{C}_0$ and have cost c_0. As we have not increased the flow, this implies that the cost of $\mathcal{C}_{ALG}$ is also c_0.

As we have argued before, the vertex-cover corresponding to the initial flow with $p(v) = 0$ for all $v \in V_p$ was $\mathcal{C}_n$ of cost c_n. As all flow increase in the while-loop was made over priceable vertices and all the priceable 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 prices found by the algorithm are optimal.

Notice that adjusting the price of the priceable vertices in each iteration is very convenient for the analysis, but not necessary for the algorithm to work. 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_f. This proves Theorem 6. □

In the next theorem we note that for the general bipartite case we can get a 2-approximation for the optimum 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 priceable 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 priceable vertex, then there are two fixed-price vertices v_2 and v_3 of cost 1 and 0, respectively. v_4 is priceable 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. □

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

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

We conclude the section with a lower bound and show that bipartite STACKVC with a small number of followers is NP-hard.

Theorem 8 *It is weakly* NP*-hard to compute revenue maximizing prices for bipartite* STACKVC *with*

- *priceable vertices in one partition and at least three followers.*
- *on a tree with priceable vertices in both partitions and at least two followers*

Proof: We reduce from PARTITION, and the reduction is similar to the one used in [11] to show hardness of the highway pricing problem. For an instance of PARTITION given by integers $A = \{a_1, \ldots, a_n\}$ we introduce n *element gadgets*. An element gadget consists of a path of four edges, in which two vertices are priceable (see Figure 5a). All non-priceable vertices have fixed cost a_i. The two outer edges belong to one follower, the two interior edges to a second follower. The vertices of different element gadgets will not be directly connected in the final construction, hence we can merge these followers into a total of two for all gadgets. By repeating arguments of [11] we observe that in the element gadget of a_i we can at most extract a revenue of $2a_i$. In particular, follower 1 will purchase exactly one vertex for each of her edges. Here we can obtain a revenue of $2a_i$ by setting each price to a_i. Follower 2 will either purchase one fixed cost vertex, or both priceable vertices. In this case we can obtain a total revenue from both followers by setting two prices that sum up to a_i. Hence, the crucial decision is whether the prices in the gadget shall sum to $2a_i$ or to a_i. In order to coordinate these decisions, we introduce a third *coordination follower*, who owns a star with a root vertex of fixed cost $\frac{3}{2}\sum_i a_i$. The leaves of the star are all priceable vertices from all element gadgets (see Figure 5b). In total, we can now gain at most $r_{max} = \frac{7}{2}\sum_i a_i$ in revenue, $2\sum_i a_i$ from the followers in the element gadgets and the rest from the coordination follower. Suppose the instance A of PARTITION has a solution $S \subset A$ such that $\sum_{a_i \in S} a_i = \sum_{a_i \notin S} a_i$. Then it is possible to obtain r_{max} as follows. We decide to set the prices to equal a_i for all vertices in the element gadgets of $a_i \in S$. For the remaining gadgets we set the prices to $a_i/2$. This extracts a revenue of $2a_i$ from each gadget and the full $\frac{3}{2}\sum_i a_i$ from the coordination follower. On the other hand, it is easy to verify that whenever we obtain r_{max}, we must decide for each gadget, in which way we intend to obtain the revenue of $2a_i$. In order to extract all possible revenue from the coordination follower, the set of gadgets with prices set to a_i will compose a solution to the PARTITION instance. This completes the first reduction.

For the second part we replace the element gadgets by a path of length five owned by a single follower. The priceable vertices are located in both of the bipartitions (see Figure 5c). All fixed price vertices have cost a_i. Observe that the follower has only 5 reasonable covers to choose from. If both prices are less than a_i, the follower will pick one of two covers including both priceable vertices. If one or both priceable vertices have cost more than a_i, the follower will include only the cheaper vertex in his cover. Finally, if both vertices have

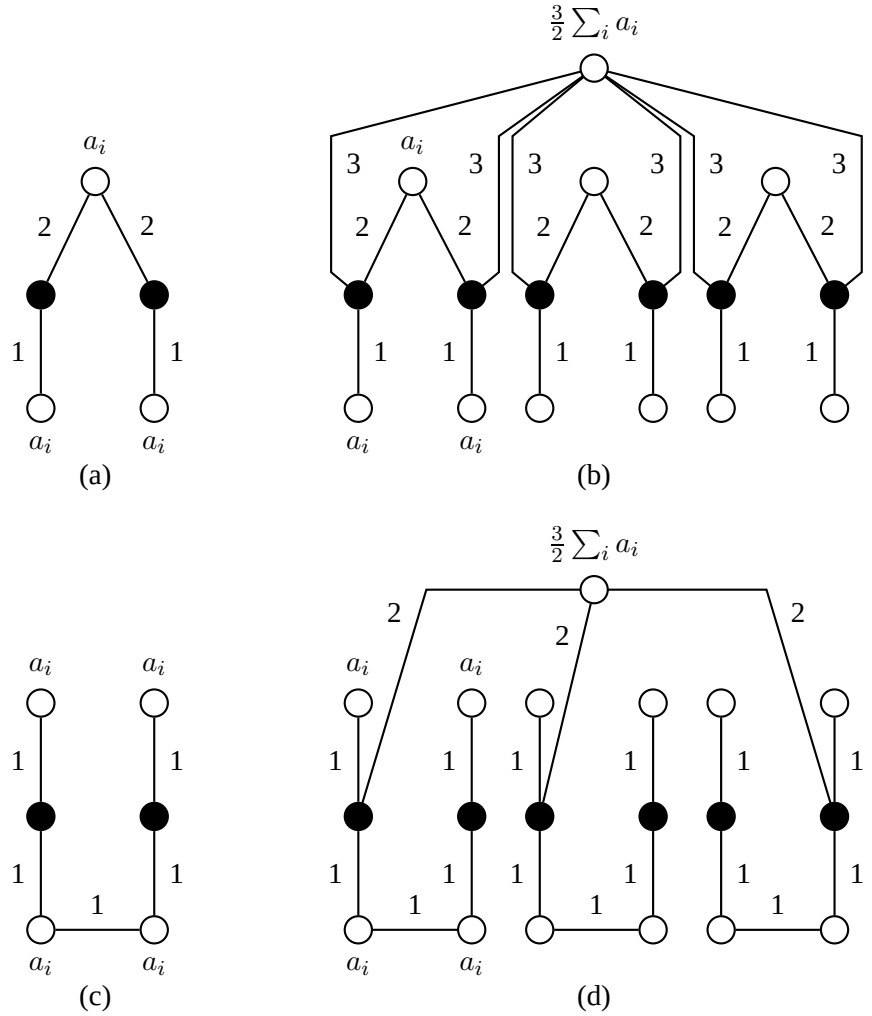


Figure 5: Construction of the reduction to PARTITION. Solid vertices are priceable, vertex labels indicate fixed cost. Edge labels indicate follower ownership, follower 3 is the coordination follower. (a),(c) Element gadgets with two (one) followers for an element a_i; (b),(d) Combination of element gadgets for an instance using three (two) followers.

price more than $2a_i$, none of the priceable vertices will be bought. An optimum revenue of $2a_i$ can again be obtained in two ways: by setting prices of a_i or $2a_i$ to both vertices. Finally, we complete the construction with a coordination player, who owns a star that connects to one priceable vertex of each element gadget (see Figure 5d). The rest of the argument follows with similar observations as before. In particular, in order to obtain the maximum revenue of $\frac{7}{2}\sum_i a_i$, the underlying instance of PARTITION must admit a solution. This completes the proof of the theorem. □

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 for this case. The best known lower bound to date is the constant factor inapproximability presented in [10].

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.

More generally, extending other fundamental algorithm design techniques to cope with pricing problems is a major open problem. We have shown here 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.

Another interesting issue that we explored in [8] is to examine problems, in which customers cannot efficiently optimize over the set of feasible subnetworks. This is obviously the case in many non-trivial practical (network) optimization problems. To obtain a solution followers must resort to approximation algorithms, and pricing for such computationally bounded customers exhibits fundamentally different properties than the ones we observed here. It is an interesting open problem to extend the results in [8] for Min-Knapsack and general vertex cover problems to more general scenarios.

References

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

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

[3] M. 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. Item pricing for revenue maximization. In *Proc. of 9th EC*, 2008.

[5] D. Biló, L. Gualá, G. Proietti, and P. Widmayer. Computational Aspects of a 2-Player Stackelberg Shortest Paths Tree Game. In *Proc. 4th WINE*, 2008.

[6] M. Bouhtou, A. Grigoriev, S. van Hoesel, A. van der Kraaij, and M. Uetz. Pricing Bridges to Cross a River. *Naval Research Logistics*, 54(4): 411–420, 2007.

[7] P. Briest. Uniform Budgets and the Envy-Free Pricing Problem. In *Proc. of 35th ICALP*, 2008.

[8] P. Briest, L. Gualá, M. Hoefer, and C. Ventre. On Stackelberg Pricing with Computationally Bounded Consumers. In *Proc. of 5th WINE*, 2009.

[9] P. Briest, M. Hoefer, and P. Krysta. Stackelberg Network Pricing Games. In *Proc. of 25th STACS*, 2008.

[10] P. Briest, S. Khanna. Improved Hardness of Approximation for Stackelberg Shortest-Path Pricing. *CoRR abs/0910.0110*, 2009.

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

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

[13] 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.

[14] J. Cardinal, E. Demaine, S. Fiorini, G. Joret, S. Langerman, I. Newman, and O. Weimann. The Stackelberg Minimum Spanning Tree Game on Planar and Bounded-Treewidth Graphs. In *Proc. of 5th WINE*, 2009.

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

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

[17] E.D. Demaine, U. Feige, M.T. Hajiaghayi, and M.R. Salavatipour. Combination Can Be Hard: Approximability of the Unique Coverage Problem. *SIAM Journal on Computing*, 38(4): 1464–1483, 2008.

[18] L. Fleischer, K. Jain, and M. Mahdian. Tolls for Heterogeneous Selfish Users in Multicommodity Networks and Generalized Congestion Games In *Proc. of 45th FOCS*, 2004.

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

[20] 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.

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

[22] G. Karakostas and S. Kolliopoulos. Edge Pricing of Multicommodity Networks for Heterogeneous Users. In *Proc. of 45th FOCS*, 2004.

[23] G. Karakostas and S. Kolliopoulos. Stackelberg Strategies for Selfish Routing in General Multicommodity Networks. *Algorithmica*, 53(1): 132–153, 2009.

[24] 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.

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

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

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

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

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

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

[31] H. Yang and H.-J. Huang. The Multi-Class, Multi-Criteria Traffic Network Equilibrium and Systems Optimum Problem. *Transportation Research Part B*, 38, pp. 1-15, 2004.