PAPER *Special Section on Foundations of Computer Science*

Approximation Preserving Reductions among Item Pricing Problems

Ryoso HAMANE[†a)], ***Nonmember***, **Toshiya ITOH**[††b)], ***Member*, *and*** **Kouhei TOMITA**[†††c)], ***Nonmember***

SUMMARY When a store sells items to customers, the store wishes to determine the prices of the items to maximize its profit. Intuitively, if the store sells the items with low (resp. high) prices, the customers buy more (resp. less) items, which provides less profit to the store. So it would be hard for the store to decide the prices of items. Assume that the store has a set V of n items and there is a set E of m customers who wish to buy those items, and also assume that each item $i \in V$ has the production cost d_i and each customer $e_j \in E$ has the valuation v_j on the bundle $e_j \subseteq V$ of items. When the store sells an item $i \in V$ at the price r_i, the profit for the item i is $p_i = r_i - d_i$. The goal of the store is to decide the price of each item to maximize its total profit. We refer to this maximization problem as the *item pricing* problem. In most of the previous works, the item pricing problem was considered under the assumption that $p_i \geq 0$ for each $i \in V$, however, Balcan, et al. [In Proc. of WINE, LNCS 4858, 2007] introduced the notion of "loss-leader," and showed that the seller can get more total profit in the case that $p_i < 0$ is allowed than in the case that $p_i < 0$ is not allowed. In this paper, we derive approximation preserving reductions among several item pricing problems and show that all of them have algorithms with good approximation ratio.

key words: *item pricing problem, approximation preserving reductions, price models, selfloops*

1. Introduction

1.1 Background

When a store sells items to customers, the store wishes to determine the prices of the items to maximize its profit. Intuitively, if the store sells the items with low (resp. high) prices, then the customers buy more (resp. less) items, which provides less profit to the store. So it would be hard for the store to determine the prices of items. Assume that the store has a set $I = \{1, 2, \ldots, n\}$ of n items and there is a set $C = \{c_1, c_2, \ldots, c_m\}$ of m customers who wish to buy those items. The goal of the store is to decide the price of each item to maximize its profit. We refer to this maximization problem as the *item pricing* problem. We classify the item pricing problem according to how many items the store can sell and how the customers valuate items. If the store can sell every item i with unlimited (resp. limited) amount, we refer to this as the *unlimited* supply model (resp. the *limited* supply model). The item pricing problem is said to be *single-minded* [9] if each customer $c_j \in C$ is interested in only a single bundle $e_j = \{i_j^1, i_j^2, \ldots\} \subseteq I$ of items with valuation $v_j \geq 0$ and has valuation "0" on all other bundles of items. Note that each customer c_j purchases the whole bundle e_j if and only if the sum of prices included in the bundle e_j is not greater than the valuation v_j. We say that the item pricing problem is *unit-demand* [9] if each customer $c_j \in C$ assigns valuation $v_j^i \geq 0$ to each item $i \in I$ and buys one of the most beneficial items for $c_j \in C$.

By regarding the set I of n items as the set V of n vertices and the set C of m customers as the set E of m hyperedges, each of which has weight $v_j \geq 0$, this can be formulated by a *weighted* hypergraph $\tilde{G} = (V, E, \{v_j\})$. Note that the hypergraph $\tilde{G}$ might have selfloops (corresponding to customers that are interested in a single item) and multiedges (corresponding to customers that want to get the same bundle of items). For a weighted hypergraph $\tilde{G} = (V, E, \{v_j\})$, assume that each item $i \in V$ has the *production* cost d_i and each customer $e_j \in E$ has the valuation v_j. For $\tilde{G}$, we define a *reduced* instance $G = (V, E, \{w_j\})$ to be $w_j = v_j - \sum_{i \in e_j} d_i$ for each $e_j \in E$. If an item $i \in V$ is assigned a price p_i in the reduced instance G, then its selling price is given by $r_i = p_i + d_i$. In this paper, we focus on the single-minded and unlimited supply model and consider reduced instances G's of weighted hypergraphs. We say that $G = (V, E, \{w_j\})$ is an instance of the *k-hypergraph vertex pricing* problem if $|e_j| \leq k$ for each $e_j \in E$, an instance of the *graph vertex pricing* problem if $|e_j| \leq 2$ for each $e_j \in E$, and an instance of the *bipartite graph vertex pricing* problem if G is a bipartite graph. As a special case of the hypergraph vertex pricing problem, we also say that $G = (V, E, \{w_j\})$ is an instance of the *highway* problem if each $e_j \in E$ is an interval on V (the definition will be given in Definition 2.7 for the *line* highway problem and in Definition 2.8 for the *cycle* highway problem).

In most of the previous works [1], [4], [5], [9], the item pricing problem is considered under the model that $p_i \geq 0$ for each item $i \in V$ (this is called the *positive price* model). By introducing the notion of *loss-leader* [6], however, Balcan, et al. [3] consider several price models in which $p_i < 0$ for some item $i \in V$ (these are referred to as the *discount* model, the *B-bounded discount* model, the *coupon* model, etc., and are formally defined in §2.1), and showed that the seller could get more profit in the case that $p_i < 0$ is allowed

Manuscript received March 24, 2008.
Manuscript revised June 24, 2008.

[†]The author is with the Department of Information Processing, Tokyo Institute of Technology, Tokyo, 152–8550 Japan.

[††]The author is with the Global Scientific Information and Computing Center (GSIC), Tokyo Institute of Technology, Tokyo, 152–8550 Japan.

[†††]The author is with the Department of Computer Science, Tokyo Institute of Technology, Tokyo, 152–8550 Japan.

a) E-mail: hamane@dac.gsic.titech.ac.jp
b) E-mail: titoh@dac.gsic.titech.ac.jp
c) E-mail: tomita@dac.gsic.titech.ac.jp

DOI: 10.1587/transinf.E92.D.149

Copyright © 2009 The Institute of Electronics, Information and Communication Engineers

than in the case that $p_i < 0$ is not allowed.

1.2 Related Works

Positive Price Models: For the hypergraph vertex pricing problem, Guruswami, et al. [9, Theorem 5.2] show an $O(\log m + \log n)$-approximation algorithm. On the other hand, Demaine, et al. [7, Theorem 3.2] present that it is hard to approximate the hypergraph vertex pricing problem within a factor of $\log^{\delta} n$ for some $\delta > 0$ under the assumption that NP $\not\subseteq$ BPTIME($2^{n^{\epsilon}}$) for some $\epsilon > 0$. For the k-hypergraph vertex pricing problem, Briest and Krysta [4, Theorem 5.1] show an $O(k^2)$-approximation algorithm, which is improved to an $O(k)$-approximation algorithm [1, Theorem 2]. For the graph vertex pricing problem, Balcan and Blum derive a 4-approximation algorithm [1, Theorem 1], while by the reduction from the vertex cover, Guruswami, et al. [9, Theorem 3.1] show that the graph vertex pricing problem is APX-hard even when all valuations are identical (if selfloops are allowed) or all valuations are either 1 or 2 (if selfloops are not allowed). For the highway problem, Balcan and Blum [1, Theorem 3] show an $O(\log n)$-approximation algorithm and for the highway problem that forms a hierarchy, Balcan and Blum [1, Theorem 4] show a fully polynomial time approximation scheme (FPTAS). For the nonapproximability for the highway problem, see [4], [8].

Other Models Based on Loss-Leader: For the highway problem, we know the $\Omega(\log n)$ gap between the positive price model and the (B-bounded) discount model [2, Theorem 1], and the $\Omega(\log n)$ gap between the coupon model and the (B-bounded) discount model [2, Theorem 2]. For the graph vertex pricing problem, the $\Omega(\log n)$ gap between the positive price model and the B-bounded discount model [2, Theorem 3] is known. For the highway problem, Balcan, et al. [3, Theorem 3] show a 2.33-approximation algorithm under the coupon model if all valuations are identical and for the highway problem on tree, Balcan, et al. [2, Theorem 15] show a 4-approximation algorithm under the coupon model if all valuations are identical. For the k-hypergraph vertex pricing problem, Balcan, et al. [3, Theorems 4 and 7] show a fully polynomial time approximation scheme (FPTAS) under the coupon and (B-bounded) discount models if the bundles of different customers form a hierarchy. For the graph vertex pricing problem, Balcan, et al. [3, Theorem 5] show a fully polynomial time approximation scheme (FPTAS) under the coupon model if the graph is minor-free.

1.3 Main Results

In this paper, we consider several kinds of the item pricing problem such as

- GRAPH_SL — the graph vertex pricing problem with selfloops;
- GRAPH_NSL — the graph vertex pricing problem with no selfloops;
- BPT_NSL — the bipartite graph vertex pricing problem with no selfloops;
- BPT_OWHW — the bipartite oneway highway problem;
- LINE_HW — the line highway problem;
- CYC_HW — the cycle highway problem,

and show that any two of them are reducible to each other with preserving approximation ratio within a constant factor (see Lemmas 3.1–3.9).

To derive approximation algorithms for the item pricing problems shown above, we consider the case with $[s, \ell]$-valuations, i.e., the valuations w_j's range over nonnegative integer interval $[s, \ell]$, and we show that

(1) Under the coupon model, there exists a $(1 + \ln \frac{\ell}{s})$-approximation algorithm $\text{ALG}^{\text{nsl}}_{\text{graph}}$ for GRAPH_NSL with $[s, \ell]$-valuations (see Theorem 4.1 in §4.1).

Note that s is the smallest valuation and ℓ is the largest valuation. From the result (1) and Lemmas 3.1–3.9, we can derive algorithms with good approximation ratio for the other problems, i.e.,

(2) Under the coupon model, there exists a $(2 + \ln \frac{\ell}{s})$-approximation algorithm $\text{ALG}^{\text{sl}}_{\text{graph}}$ for GRAPH_SL with $[s, \ell]$-valuations (see Corollary 4.1 in §4.2).

(3) Under the coupon model, there exists a $(1 + \ln \frac{\ell}{s})$-approximation algorithm $\text{ALG}^{\text{nsl}}_{\text{bpt}}$ for BPT_NSL with $[s, \ell]$-valuations (see Corollary 4.2 in §4.2).

(4) Under the coupon model, there exists a $4(1 + \ln \frac{\ell}{s})$-approximation algorithm $\text{ALG}^{\text{hw}}_{\text{line}}$ for LINE_HW with $[s, \ell]$-valuations (see Corollary 4.3 in §4.2).

(5) Under the coupon model, there exists a $4(1 + \ln \frac{\ell}{s})$-approximation algorithm $\text{ALG}^{\text{hw}}_{\text{cycle}}$ for CYC_HW with $[s, \ell]$-valuations (see Corollary 4.4 in §4.2).

The results (1) and (2) hold for graphs that are not minor-free [12], and are general results for the graph vertex pricing problem. The result (4) holds for the line highway problem with multiple valuations, but the result by Balcan, et al. [3, Theorems 3] holds only for the one with identical valuations. The result (5) holds for the cycle highway problem with multiple valuations, which is introduced in this paper, but the result by Balcan, et al. [3, Theorems 3] holds for the line highway problem with identical valuations and the results by Balcan, et al. [2, Theorems 15] holds for the highway problem on tree with identical valuations.

The result (2) is derived by the result (1) and the reduction from GRAPH_SL to GRAPH_NSL (Lemma 3.2). By a different (direct) approach, we derive an algorithm with better approximation ratio for GRAPH_SL, which improves the result (2).

(6) Under the coupon model, there exists a $(\frac{3}{2} + \ln \frac{\ell}{s})$-approximation algorithm $\text{ALG}^{\text{sl}}_{\text{graph}}$ for GRAPH_SL with $[s, \ell]$-valuations (see Theorem 4.2 in §4.2).

2. Preliminaries

2.1 Price Models

Let $G = (V, E, \{w_j\})$ be a reduced instance of the item pricing problem. For a hyperedge $e_j \in E$ and a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ over the n vertices, let $p(e_j) = \sum_{i \in e_j} p_i$ be the sum of the profit on $e_j \in E$, i.e., the profit that is returned from the customer $e_j \in E$ for the price vector $\boldsymbol{p}$. In most of the previous works [1], [4], [5], [9], the item pricing problem is considered under the model that $p_i \geq 0$ for each item $i \in V$. By introducing the notion of *loss-leader*, however, Balcan, et al. [3] considered several price models in which $p_i < 0$ for some item $i \in V$, and showed that the seller could get more profit in the case that $p_i < 0$ is allowed than in the case that $p_i < 0$ is not allowed.

In the following, we formally present the definitions of positive price model, discount model, B-bounded discount model, and coupon model [3] with respect to the reduced instance.

Definition 2.1 (Positive Price Model)**:** *Under the condition that $p_i \geq 0$ for each $i \in V$, find a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ that maximizes*

$$\textsc{Profit}_{\text{pos}}(\boldsymbol{p}) = \sum_{e_j \in E : w_j \geq p(e_j)} p(e_j).$$

Definition 2.2 (Discount Model)**:** *Find a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ that maximizes*

$$\textsc{Profit}_{\text{disc}}(\boldsymbol{p}) = \sum_{e_j \in E : w_j \geq p(e_j)} p(e_j).$$

Note that $p_i < 0$ is allowed in the discount model. For a fixed $B > 0$, if we assume that $p_i \geq -B$ in the discount model, then it is called the B-bounded discount model.

Definition 2.3 (Coupon Model)**:** *Find a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ that maximizes*

$$\textsc{Profit}_{\text{coup}}(\boldsymbol{p}) = \sum_{e_j \in E : w_j \geq p(e_j)} \max\{p(e_j), 0\}.$$

Under the coupon model, if $w_j \leq 0$, then e_j never contributes to the profit for any price vector $\boldsymbol{p}$. So without loss of generality, we assume that $w_j > 0$ for each $e_j \in E$ under the coupon model.

2.2 Item Pricing Problems

Graph Vertex Pricing Problem: As mentioned in §1.1, the graph vertex pricing problem may have selfloops. We classify the graph vertex pricing problem according to whether it is allowed to have selfloops or not.

Definition 2.4: *We say that $G = (V, E, \{w_j\})$ is a reduced instance of the* graph vertex pricing problem with selfloops (graph_sl) *if G is allowed to have selfloops.*

Definition 2.5: *We say that $G = (V, E, \{w_j\})$ is a reduced instance of the* graph vertex pricing problem with no selfloops (graph_nsl) *if G does not have selfloops.*

Definition 2.6: *We say that $G = (V, E, \{w_j\})$ is a reduced instance of the* bipartite graph pricing problem with no selfloops (bpt_nsl) *if G is a bipartite graph and does not have selfloops.*

For any pair of integers $a \leq b$, let $[a, b] = \{a, a + 1, \ldots, b\}$. We say that $G = (V, E, \{w_j\})$ is a reduced instance of graph_sl, graph_nsl, or bpt_nsl with $[s, \ell]$-*valuations* if the valuations w_j's range over integer interval $[s, \ell]$, where s is the smallest integer valuation in G and ℓ is the largest integer valuation in G. In particular, we say that $G = (V, E, \{w_j\})$ is a reduced instance of graph_sl, graph_nsl, or bpt_nsl with a *single* valuation if $w_j = w > 0$ for each $e_j \in E$.

Highway Problem: Informally, we say that $G = (V, E, \{w_j\})$ is an instance of the *line* highway problem [9] if each $e_j \in E$ is an interval in the line on V. We introduce the *cycle* highway problem as a generalization of the line highway problem, and we say that $G = (V, E, \{w_j\})$ is an instance of the cycle highway problem if each $e_j \in E$ is an interval in the cycle on V.

Definition 2.7: *We say that $G = (V, E, \{w_j\})$ is a reduced instance of the* line highway problem (line_hw) *if $e_j = [s_j, t_j] \subseteq V$ for each $e_j \in E$, where $V = [1, n]$ and $1 \leq s_j \leq t_j \leq n$.*

Definition 2.8: *We say that $G = (V, E, \{w_j\})$ is a reduced instance of the* cycle highway problem (cyc_hw) *if $e_j = [s_j, t_j] \subseteq V$ or $e_j = [t_j, n] \cup [1, s_j] \subseteq V$ for each $e_j \in E$, where $V = [1, n]$ and $1 \leq s_j \leq t_j \leq n$.*

We can similarly define a reduced instance of line_hw or cyc_hw with $[s, \ell]$-*valuations* and a *single* valuation.

2.2.1 DAG Representation of line_hw

In this subsection, we present the DAG representation of the line highway problem due to Balcan, et al. [3, §3]. For a reduced instance $G = (V, E, \{w_j\})$ of the line highway problem, define the DAG representation $H = (U, F, \{w_j\})$ of G as follows: For $V = \{1, 2, \ldots, n\}$, let $U = \{u_0, u_1, \ldots, u_n\}$ be the set of $n + 1$ vertices, and for each $e_j = [s_j, t_j] \in E$, let $f_j = (u_{s_j - 1}, u_{t_j}) \in F$ be the arc $u_{s_j - 1} \to u_{t_j}$ with weight w_j.

Let $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ be a price vector for $G = (V, E, \{w_j\})$. For the DAG representation $H = (U, F, \{w_j\})$ of G, define the partial sum for $u_i \in U$ by $s_i = \sum_{h=1}^{i} p_h$, where $s_0 = 0$. On the other hand, let $\boldsymbol{s} = (s_0, s_1, \ldots, s_n)$ be the partial sum vector for the DAG representation H of G. For the partial sum vector $\boldsymbol{s}$, we can recover the price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ for G by $p_i = s_i - s_{i-1}$ for each $i \in [1, n]$.

Definition 2.9: *We say that $G = (V, E, \{w_j\})$ is a reduced instance of the* bipartite oneway highway problem (bpt_owhw) *if the* DAG *representation $H = (L \cup R, F, \{w_j\})$ of G is a directed bipartite graph such that $i \in L$ and $j \in R$ for each arc $f = (i, j) \in F$.*

We can define a reduced instance of BPT_OWHW with $[s, \ell]$-*valuations* and a *single* valuation analogously.

3. Approximation Preserving Reductions

We say that a maximization problem A is *approximation preserving reducible* to a maximization problem B if for an α-approximation algorithm for B and some constant $c > 0$, it is possible to construct a $c\alpha$-approximation algorithm for A that runs efficiently (it is similarly defined for the case of the minimization problem).

In this section, we show approximation preserving reductions among the problems GRAPH_SL, GRAPH_NSL, BPT_NSL, BPT_OWHW, LINE_HW, and CYC_HW.

3.1 GRAPH_SL VS. GRAPH_NSL

We show that there exist approximation preserving reductions between GRAPH_SL and GRAPH_NSL.

Lemma 3.1: *Under the coupon model, if there exists a β-approximation algorithm $\textsc{Alg}^{\mathrm{sl}}_{\mathrm{graph}}$ for* GRAPH_SL, *then we can construct a β-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ for* GRAPH_NSL.

Proof: This follows from the fact that GRAPH_NSL is a subproblem of GRAPH_SL. ■

Lemma 3.2: *Under the coupon model, if there exists a β-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ for* GRAPH_NSL, *then we can construct a $(\beta+1)$-approximation algorithm $\textsc{Alg}^{\mathrm{sl}}_{\mathrm{graph}}$ for* GRAPH_SL.

Proof: For an instance $G = (V, E, \{w_j\})$ of GRAPH_SL, let E_{sl} be the set of selfloops in E and $E_{\mathrm{c}} = E - E_{\mathrm{sl}}$, i.e., the set of edges in E that are not selfloops. For each vertex $i \in V$, let E^i_{sl} be the set of selfloops incident to $i \in V$, and for a price p, let $\textsc{Profit}^i_{\mathrm{sl}}(p)$ be the profit from the selfloops in E^i_{sl}.

For a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$, we use $\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{p})$ to denote the total profit returned from the selfloops in E_{sl}, i.e., $\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{p}) = \textsc{Profit}^1_{\mathrm{sl}}(p_1) + \textsc{Profit}^2_{\mathrm{sl}}(p_2) + \cdots + \textsc{Profit}^n_{\mathrm{sl}}(p_n)$, and $\textsc{Profit}_{\mathrm{c}}(\boldsymbol{p})$ to denote the total profit returned from the edges in E_{c}. Let $\boldsymbol{p}^*$ be the optimal price vector for G. Let $\textsc{Opt}(G) = \textsc{Profit}(\boldsymbol{p}^*)$ be the maximum total profit for G, $\textsc{Opt}_{\mathrm{sl}}(G)$ be the total profit returned from the selfloops in E_{sl}, and $\textsc{Opt}_{\mathrm{c}}(G)$ be the total profit returned from the edges in E_{c}. By definitions, it is obvious that $\textsc{Opt}(G) = \textsc{Opt}_{\mathrm{sl}}(G) + \textsc{Opt}_{\mathrm{c}}(G)$. The reduction from GRAPH_SL to GRAPH_NSL is given in Fig. 1.

```
Input: $G = (V, E, \{w_j\})$ of GRAPH_SL.
1. For each vertex $i \in V$, compute a price $\sigma_i$ that satisfies
       $\textsc{Profit}^i_{\mathrm{sl}}(\sigma_i) = \max_{e_j \in E^i_{\mathrm{sl}}} \textsc{Profit}^i_{\mathrm{sl}}(w_j)$,
   and let $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ be a price vector.
2. Define an instance $\tilde{G} = (V, \tilde{E}, \{w_j\})$ of GRAPH_NSL by
   removing all selfloops in $E_{\mathrm{sl}}$.
3. Compute a price vector $\boldsymbol{\tau}$ by running the algorithm
   $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ on input $\tilde{G}$.
4. Output the price vector $\boldsymbol{p}$ given by
       $\boldsymbol{p} = \begin{cases} \boldsymbol{\sigma} & \textsc{Profit}(\boldsymbol{\sigma}) \geq \textsc{Profit}(\boldsymbol{\tau}); \\ \boldsymbol{\tau} & \textsc{Profit}(\boldsymbol{\sigma}) < \textsc{Profit}(\boldsymbol{\tau}). \end{cases}$
```

Fig. 1 The reduction from GRAPH_SL to GRAPH_NSL.

From the definition of the price vector $\boldsymbol{\sigma}$ defined in Step 1, we have that $\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\sigma}) \geq \textsc{Opt}_{\mathrm{sl}}(G)$. From the assumption that $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ is a β-approximation algorithm for GRAPH_NSL, we have that $\textsc{Profit}(\boldsymbol{\tau}) \geq \textsc{Opt}(\tilde{G})/\beta \geq \textsc{Opt}_{\mathrm{c}}(G)/\beta$ for the price vector $\boldsymbol{\tau}$ defined in Step 3. Thus it follows that

$$\begin{aligned}
\textsc{Profit}(\boldsymbol{p}) &= \max\{\textsc{Profit}(\boldsymbol{\sigma}), \textsc{Profit}(\boldsymbol{\tau})\} \\
&\geq \max\left\{\textsc{Opt}_{\mathrm{sl}}(G), \frac{\textsc{Opt}_{\mathrm{c}}(G)}{\beta}\right\} \\
&\geq \frac{1}{\beta+1} \cdot \textsc{Opt}_{\mathrm{sl}}(G) + \frac{\beta}{\beta+1} \cdot \frac{\textsc{Opt}_{\mathrm{c}}(G)}{\beta} \\
&= \frac{1}{\beta+1} \cdot \{\textsc{Opt}_{\mathrm{sl}}(G) + \textsc{Opt}_{\mathrm{c}}(G)\} = \frac{\textsc{Opt}(G)}{\beta+1},
\end{aligned}$$

for the price vector $\boldsymbol{p}$ defined in Step 4. ■

3.2 GRAPH_NSL VS. BPT_NSL

We show that there exist approximation preserving reductions between GRAPH_NSL and BPT_NSL.

Lemma 3.3: *Under the coupon model, if there exists a β-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ for* GRAPH_NSL, *then we can construct a β-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{bpt}}$ for* BPT_NSL.

Proof: This is due to the fact that BPT_NSL is a subproblem of GRAPH_NSL. ■

Lemma 3.4: *Under the coupon model, if there exists a β-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{bpt}}$ for* BPT_NSL, *then we can construct a 2β-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ for* GRAPH_NSL.

Proof: Let $G = (V, E, \{w_j\})$ be an instance of GRAPH_NSL. Let $\boldsymbol{p}^*$ be the optimal price vector for G and $\textsc{Opt}(G) = \textsc{Profit}(\boldsymbol{p}^*)$ be the maximum total profit for G. In Fig. 2, we describe the reduction from GRAPH_NSL to BPT_NSL.

Let $\textsc{Profit}_{G'}(\boldsymbol{p})$ be the total profit returned from the edges in E' for the price vector $\boldsymbol{p}$ of $G' = (V, E', \{w_j\})$. Note that each $e \in E$ is included in E' with probability $1/2$. From the linearity of expectation, it follows that $\mathbf{E}[\textsc{Profit}_{G'}(\boldsymbol{p}^*)] = \textsc{Opt}(G)/2$. Let $\boldsymbol{q}^*$ be the optimal price vector and $\textsc{Opt}(G') = \textsc{Profit}_{G'}(\boldsymbol{q}^*)$ be the maximum total profit for $G' = (V, E', \{w_j\})$. Note that the price vector $\boldsymbol{p}$ is computed for an instance $G' = (V, E', \{w_j\})$ of BPT_NSL and

Input: $G = (V, E, \{w_j\})$ of GRAPH_NSL.

1. Mark each vertex $i \in V$ independently with probability $1/2$. Let L be the set of *marked* vertices and $R = V - L$ be the set of *unmarked* vertices.
2. For the partition of $(L : R)$ of V, define $E' = \{e = (u, v) \in E : u \in L, v \in R\} \subseteq E$ and regard $G' = (V, E', \{w_j\})$ as an instance of BPT_NSL.
3. Compute a price vector $\boldsymbol{p}$ by running the algorithm $\text{ALG}_{\text{bpt}}^{\text{nsl}}$ on input G' and output $\boldsymbol{p}$.

Fig. 2 The reduction from GRAPH_NSL to BPT_NSL.

G' is defined according to the random choice of R. Then we have that

$$
\begin{aligned}
&\mathbf{E}\left[\text{Profit}_G(\boldsymbol{p})\right] \geq \mathbf{E}\left[\text{Profit}_{G'}(\boldsymbol{p})\right] \\
&\geq \mathbf{E}\left[\frac{\text{Opt}(G')}{\beta}\right] = \mathbf{E}\left[\frac{\text{Profit}_{G'}(\boldsymbol{q}^*)}{\beta}\right] \\
&\geq \mathbf{E}\left[\frac{\text{Profit}_{G'}(\boldsymbol{p}^*)}{\beta}\right] = \frac{1}{\beta} \cdot \mathbf{E}\left[\text{Profit}_{G'}(\boldsymbol{p}^*)\right] \\
&= \frac{1}{2\beta} \cdot \text{Opt}(G),
\end{aligned}
$$

where the 2nd inequality is due to the assumption that $\text{ALG}_{\text{bpt}}^{\text{nsl}}$ is a β-approximation algorithm for BPT_NSL and the 3rd inequality is due to the definition of $\boldsymbol{q}^*$. ■

Remark 3.1: *To guarantee that each edge $e \in E$ is included in E' with probability $1/2$, it suffices to mark the vertices in pairwise independent manner. Thus the reduction in* Fig. 2 *can be derandomized as follows: Let $n = |V|$ and let $X_1, X_2, \ldots, X_n : \Omega \to \{0, 1\}$ be pairwise independent random variables such that* $\Pr[X_i = 0] = \Pr[X_i = 1] = 1/2$ *for each $1 \leq i \leq n$ and $|\Omega| = n^{O(1)}$ (for the concrete construction for such pairwise random variables, see* [10], [11]). *For each $\omega \in \Omega$, (1) define $R(\omega)$ in such a way that $i \in V$ is included in $R(\omega)$ if and only if $X_i(\omega) = 1$ in* Step 1 *of* Fig. 2; (2) *define $G'(\omega) = (V, E'(\omega), \{w_j\})$ in the same way as* Step 2 *of* Fig. 2;, (3) *compute $\boldsymbol{p}(\omega)$ in the same way as* Step 3 *of* Fig. 2. *Finally, output $\boldsymbol{p} \in \{\boldsymbol{p}(\omega)\}_{\omega\in\Omega}$ such that* $\text{Profit}_G(\boldsymbol{p}) = \max_{\omega\in\Omega} \text{Profit}_G(\boldsymbol{p}(\omega)).$

3.3 BPT_NSL vs. BPT_OWHW

In this subsection, we show that the problems BPT_NSL and BPT_OWHW are efficiently reducible to each other with preserving profits.

Lemma 3.5: *Under the coupon model, the following holds between* BPT_NSL *and* BPT_OWHW.

(1) *For any instance G of* BPT_NSL *and any price vector $\boldsymbol{p}$ for G, it is possible to efficiently construct a price vector $\boldsymbol{q}$ and an instance G' of* BPT_OWHW *such that* $\text{Profit}_{G'}(\boldsymbol{q}) = \text{Profit}_G(\boldsymbol{p})$;

(2) *For any instance G of* BPT_OWHW *and any price vector $\boldsymbol{q}$ for G, it is possible to efficiently construct a price vector $\boldsymbol{p}$ and an instance G' of* BPT_NSL *such that* $\text{Profit}_{G'}(\boldsymbol{p}) = \text{Profit}_G(\boldsymbol{q})$.

Proof: We first consider the statement (1). Let $G = (L \cup R, E, \{w_j\})$ be an instance of BPT_NSL. Let $F = \{f_j \in L \times R : e_j \in E\}$ by regarding each edge $e_j = (u_j, v_j) \in E$ as an arc $f_j = (u_j, v_j) \in L \times R$. Let $H = (L \cup R, F, \{w_j\})$ be the DAG representation of an instance $G' = (U, E', \{w_j\})$ for BPT_OWHW. For a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ for $G = (L \cup R, E, \{w_j\})$, the total profit returned from the customers in E is given by

$$
\begin{aligned}
&\text{Profit}_G(\boldsymbol{p}) \\
&= \sum_{e_j=(v_{j_\ell}, v_{j_r})\in E: w_j \geq p(e_j)} \max\left\{p_{j_\ell} + p_{j_r}, 0\right\} \\
&= \sum_{e_j=(v_{j_\ell}, v_{j_r})\in E: w_j \geq p(e_j)} \max\left\{p_{j_r} - \left(-p_{j_\ell}\right), 0\right\}.
\end{aligned}
$$

From the price vector $\boldsymbol{p}$, define a partial sum vector $\boldsymbol{s} = (s_0, s_1, s_2, \ldots, s_n)$ of H as follows: Let $s_0 = 0$ and for each $i \in [1, n]$, let $s_i = -p_i$ if $i \in L$ and let $s_i = p_i$ if $i \in R$. From the partial sum vector $\boldsymbol{s}$, we can derive a price vector $\boldsymbol{q}$ for $G' = (U, E', \{w_j\})$ of BPT_OWHW that satisfies $\text{Profit}_{G'}(\boldsymbol{q}) = \text{Profit}_G(\boldsymbol{p})$.

Let us consider the statement (2). Let $H = (L \cup R, F, \{w_j\})$ be the DAG representation of an instance $G = (V, E, \{w_j\})$ for BPT_OWHW. Let $E' = \{e_j : f_j \in L \times R\}$ by regarding each arc $f_j \in L \times R$ as a undirected edge e_j and let $G' = (L \cup R, E', \{w_j\})$ be an instance of BPT_NSL. For a price vector $\boldsymbol{q}$ for $G = (V, E, \{w_j\})$, let $\boldsymbol{\tau} = (\tau_0, \tau_1, \tau_2, \ldots, \tau_n)$ be the partial sum vector for the DAG representation H of G, where $\tau_0 = 0$. For the partial sum vector $\boldsymbol{\tau}$, the total profit returned from the customers in E is given by

$$
\begin{aligned}
&\text{Profit}_G(\boldsymbol{q}) \\
&= \sum_{e_j=(v_{j_\ell}, v_{j_r})\in E: w_j \geq p(e_j)} \max\left\{\sum_{h=j_\ell}^{j_r} q_h, 0\right\} \\
&= \sum_{f_j=(v_{j_s}, v_{j_t})\in L\times R: w_j \geq p(e_j)} \max\left\{\tau_{j_t} - \tau_{j_s}, 0\right\}.
\end{aligned}
$$

From the partial sum vector $\boldsymbol{\tau}$, define a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$ for G' of BPT_NSL as follows: For each $i \in [1, n]$, let $p_i = -\tau_i$ if $i \in L$ and let $p_i = \tau_i$ if $i \in R$. Then the price vector $\boldsymbol{p}$ for the instance G' of BPT_NSL satisfies that $\text{Profit}_{G'}(\boldsymbol{p}) = \text{Profit}_G(\boldsymbol{q})$. ■

3.4 LINE_HW vs. BPT_OWHW

We show that there exist approximation preserving reductions between LINE_HW and BPT_OWHW.

Lemma 3.6: *Under the coupon model, if there exists a β-approximation algorithm $\text{ALG}_{\text{line}}^{\text{hw}}$ for* LINE_HW, *then we can construct a β-approximation algorithm $\text{ALG}_{\text{bpt}}^{\text{owhw}}$ for* BPT_OWHW.

```
Input: $G = (V, E, \{w_j\})$ of LINE_HW.
1. Construct the DAG representation $H = (U, F, \{w_j\})$ of $G$.
2. Mark each vertex $i \in U$ independently with probability $1/2$. Let $L$ be the set of marked vertices and $R = U - L$ be the set of unmarked vertices.
3. For the partition $(L : R)$ of $U$, define $F' = \{(u, v) \in F : u \in L, v \in R\} \subseteq F$ and regard $H' = (L \cup R, F', \{w_j\})$ as the DAG representation of an instance $G' = (V, E', \{w_j\})$ of BPT_OWHW.
4. Compute a price vector $\boldsymbol{p}$ by running the algorithm $\mathrm{ALG}_{\mathrm{bpt}}^{\mathrm{owhw}}$ on input $G'$ and output $\boldsymbol{p}$.
```

Fig. 3 The reduction from LINE_HW to BPT_OWHW.

Proof: This follows from the fact that BPT_OWHW is a special case of LINE_HW. ■

Lemma 3.7: *Under the coupon model, if there exists a β-approximation algorithm $\mathrm{ALG}_{\mathrm{bpt}}^{\mathrm{owhw}}$ for* BPT_OWHW, *then we can construct a 4β-approximation algorithm $\mathrm{ALG}_{\mathrm{line}}^{\mathrm{hw}}$ for* LINE_HW.

Proof: Let $G = (V, E, \{w_j\})$ be an instance of LINE_HW. Let $\boldsymbol{p}^*$ be the optimal price vector for G and $\mathrm{OPT}(G) = \mathrm{PROFIT}(\boldsymbol{p}^*)$ be the maximum total profit for G. In Fig. 3, we describe the reduction from LINE_HW to BPT_OWHW.

For the price vector $\boldsymbol{p}$ of $G' = (V, E', \{w_j\})$, let $\mathrm{PROFIT}_{G'}(\boldsymbol{p})$ be the total profit from the customers in E'. Note that each $e \in E$ is included in E' with probability $1/4$. From the linearity of expectation, it follows that $\mathbf{E}[\mathrm{PROFIT}_{G'}(\boldsymbol{p}^*)] = \mathrm{OPT}(G)/4$. For the optimal price vector $\boldsymbol{q}^*$, let $\mathrm{OPT}(G') = \mathrm{PROFIT}_{G'}(\boldsymbol{q}^*)$ be the maximum total profit for $G' = (V, E', \{w_j\})$. Note that the price vector $\boldsymbol{p}$ is computed for an instance $G' = (V, E', \{w_j\})$ of BPT_NSL and G' is defined according to the random choice of R. Then we have that

$$\begin{aligned}
&\mathbf{E}\left[\mathrm{PROFIT}_G(\boldsymbol{p})\right] \geq \mathbf{E}\left[\mathrm{PROFIT}_{G'}(\boldsymbol{p})\right] \\
&\geq \mathbf{E}\left[\frac{\mathrm{OPT}(G')}{\beta}\right] = \mathbf{E}\left[\frac{\mathrm{PROFIT}_{G'}(\boldsymbol{q}^*)}{\beta}\right] \\
&\geq \mathbf{E}\left[\frac{\mathrm{PROFIT}_{G'}(\boldsymbol{p}^*)}{\beta}\right] = \frac{1}{\beta} \cdot \mathbf{E}\left[\mathrm{PROFIT}_{G'}(\boldsymbol{p}^*)\right] \\
&= \frac{1}{4\beta} \cdot \mathrm{OPT}(G),
\end{aligned}$$

where the 2nd inequality is due to the assumption that $\mathrm{ALG}_{\mathrm{bpt}}^{\mathrm{owhw}}$ is a β-approximation algorithm for BPT_NSL and the 3rd inequality is due to the definition of $\boldsymbol{q}^*$. ■

Remark 3.2: *In a way similar to the argument in* Remark 3.1, *the reduction in* Fig. 3 *can be easily derandomized by applying pairwise independent* 0/1*-random variables with a small sample space* [10], [11] *in* Step 2.

3.5 CYC_HW vs. BPT_OWHW

We show that there exist approximation preserving reductions between CYC_HW and BPT_OWHW.

```
Input: $G = (V, E, \{w_j\})$ of GRAPH_NSL.
1. For each $x \in [s, \ell]$, assign $x/2$ to all vertices $i \in V$ and let $\boldsymbol{p}_x = (x/2, x/2, \ldots, x/2)$.
2. Output the price vector $\boldsymbol{p}$ given by
       $\mathrm{PROFIT}(\boldsymbol{p}) = \max_{x \in [s,\ell]} \mathrm{PROFIT}(\boldsymbol{p}_x)$.
```

Fig. 4 The algorithm $\mathrm{ALG}_{\mathrm{graph}}^{\mathrm{nsl}}$ for GRAPH_NSL.

Lemma 3.8: *Under the coupon model, if there exists a β-approximation algorithm $\mathrm{ALG}_{\mathrm{cycle}}^{\mathrm{hw}}$ for* CYC_HW, *then we can construct a β-approximation algorithm $\mathrm{ALG}_{\mathrm{bpt}}^{\mathrm{owhw}}$ for* BPT_OWHW.

Proof: This follows from the fact that BPT_OWHW is a special case of CYC_HW. ■

Lemma 3.9: *Under the coupon model, if there exists a β-approximation algorithm $\mathrm{ALG}_{\mathrm{bpt}}^{\mathrm{owhw}}$ for* BPT_OWHW, *then we can construct a 4β-approximation algorithm $\mathrm{ALG}_{\mathrm{cycle}}^{\mathrm{hw}}$ for* CYC_HW.

Proof: In a way similar to the proof of Lemma 3.7. ■

4. Item Pricing Problems with $[s, \ell]$-Valuations

4.1 Approximation Algorithm for GRAPH_NSL with $[s, \ell]$-Valuations

In this subsection, we show an approximation algorithm for GRAPH_NSL with $[s, \ell]$-valuations.

Theorem 4.1: *Under the coupon model, there exists a $(1 + \ln \frac{\ell}{s})$-approximation algorithm $\mathrm{ALG}_{\mathrm{graph}}^{\mathrm{nsl}}$ for* GRAPH_NSL *with $[s, \ell]$-valuations.*

Proof: In Fig. 4, we describe the approximation algorithm for GRAPH_NSL with $[s, \ell]$-valuations. As noticed in §1.3, we assume that the valuations w_j's range over nonnegative integer interval $[s, \ell]$.

Let $G = (V, E, \{w_j\})$ be an instance of GRAPH_NSL with $[s, \ell]$-valuations. Let $\boldsymbol{p}^*$ be the optimal price vector for G and $\mathrm{OPT}(G) = \mathrm{PROFIT}(\boldsymbol{p}^*)$ be the maximum total profit for G. For each $x \in [s, \ell]$, let $E_x = \{e_j \in E : w_j = x\}$ be the set of edges with valuation x and $m_x = |E_x|$. Then we have that $m = |E| = m_s + m_{s+1} + \cdots + m_\ell$ and $\mathrm{OPT}(G) \leq sm_s + (s+1)m_{s+1} + \cdots + \ell m_\ell$. For each $x \in [s, \ell]$, the price vector $\boldsymbol{p}_x$ provides $\mathrm{PROFIT}(\boldsymbol{p}_x) = xm_x + xm_{x+1} + \cdots + xm_\ell$. So we have that

$$\begin{aligned}
\mathrm{OPT}(G) &\leq sm_s + (s+1)m_{s+1} + \cdots + \ell m_\ell \\
&= \mathrm{PROFIT}(\boldsymbol{p}_s) + \sum_{x=s+1}^{\ell} \frac{\mathrm{PROFIT}(\boldsymbol{p}_x)}{x} \\
&\leq \mathrm{PROFIT}(\boldsymbol{p}) + \sum_{x=s+1}^{\ell} \frac{\mathrm{PROFIT}(\boldsymbol{p})}{x}
\end{aligned}$$

$$\leq \left(1 + \sum_{k=s+1}^{\ell} \frac{1}{k}\right) \cdot \textsc{Profit}(\boldsymbol{p}).$$

Since $\sum_{k=s+1}^{\ell} 1/k \leq \ln \frac{\ell}{s}$, we have that $\textsc{Opt}(G) \leq (1 + \ln \frac{\ell}{s}) \cdot \textsc{Profit}(\boldsymbol{p})$. ■

4.2 Approximation Algorithm for the Other Problems with $[s, \ell]$-Valuations

From Theorem 4.1 and lemmas given in §3, we can immediately show the following results.

Corollary 4.1: *Under the coupon model, there exists a* $(2 + \ln \frac{\ell}{s})$*-approximation algorithm* $\textsc{Alg}^{\mathrm{sl}}_{\mathrm{graph}}$ *for* GRAPH_SL *with* $[s, \ell]$*-valuations.*

Proof: From Theorem 4.1 and Lemma 3.2. ■

Corollary 4.2: *Under the coupon model, there exists a* $(1 + \ln \frac{\ell}{s})$*-approximation algorithm* $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{bpt}}$ *for* BPT_NSL *with* $[s, \ell]$*-valuations.*

Proof: From Theorem 4.1 and Lemma 3.3. ■

Corollary 4.3: *Under the coupon model, there exists a* $4(1 + \ln \frac{\ell}{s})$*-approximation algorithm* $\textsc{Alg}^{\mathrm{hw}}_{\mathrm{line}}$ *for* LINE_HW *with* $[s, \ell]$*-valuations.*

Proof: This follows from Theorem 4.1 and Lemmas 3.3, 3.5, and 3.7. ■

Corollary 4.4: *Under the coupon model, there exists a* $4(1 + \ln \frac{\ell}{s})$*-approximation algorithm* $\textsc{Alg}^{\mathrm{hw}}_{\mathrm{cycle}}$ *for* CYC_HW *with* $[s, \ell]$*-valuations.*

Proof: This follows from Theorem 4.1 and Lemmas 3.3, 3.5, and 3.9. ■

By the direct approach to GRAPH_SL with $[s, \ell]$-valuations, we can show the following theorem, which improves Corollary 4.1.

Theorem 4.2: *Under the coupon model, there exists a* $(\frac{3}{2} + \ln \frac{\ell}{s})$*-approximation algorithm* $\textsc{Alg}^{\mathrm{sl}}_{\mathrm{graph}}$ *for* GRAPH_SL *with* $[s, \ell]$*-valuations.*

Proof: For an instance $G = (V, E, \{w_j\})$ of GRAPH_SL, let E_{sl} be the set of selfloops in E and $E_{\mathrm{c}} = E - E_{\mathrm{sl}}$, i.e., the set edges in E that are not selfloops. For each vertex $i \in V$, let E^i_{sl} be the set of selfloops incident to $i \in V$, and for a price p, let $\textsc{Profit}^i_{\mathrm{sl}}(p)$ be the profit returned from the selfloops in E^i_{sl}. For a price vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_n)$, we use $\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{p})$ to denote the total profit returned from the selfloops in E_{sl}, i.e.,

$$\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{p}) = \sum_{i=1}^{n} \textsc{Profit}^i_{\mathrm{sl}}(p_i),$$

and use $\textsc{Profit}_{\mathrm{c}}(\boldsymbol{p})$ to denote the total profit returned from the edges in E_{c}. Let $\boldsymbol{p}^*$ be the optimal price vector and $\textsc{Opt}(G) = \textsc{Profit}(\boldsymbol{p}^*)$ be the maximum total profit for G. Let $\textsc{Opt}_{\mathrm{sl}}(G)$ be the total profit returned from the selfloops in E_{sl} and $\textsc{Opt}_{\mathrm{c}}(G)$ be the total profit returned from the edges in E_{c}. By definitions, it is obvious that $\textsc{Opt}(G) = \textsc{Opt}_{\mathrm{sl}}(G) + \textsc{Opt}_{\mathrm{c}}(G)$. The algorithm $\textsc{Alg}^{\mathrm{sl}}_{\mathrm{graph}}$ for GRAPH_SL is given in Fig. 5. As noticed in §1.3, we assume that the valuations w_j's range over nonnegative integer interval $[s, \ell]$.

```
Input: $G = (V, E, \{w_j\})$ of GRAPH_SL.
1. For each vertex $i \in V$, compute a price $\sigma_i$ that satisfies
       $\textsc{Profit}^i_{\mathrm{sl}}(\sigma_i) = \max_{e_j \in E^i_{\mathrm{sl}}} \textsc{Profit}^i_{\mathrm{sl}}(w_j)$,
   and let $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ be a price vector.
2. For each $x \in [s, \ell]$, assign $x/2$ to all vertices $i \in V$ and let
   $\boldsymbol{\tau}_x = (x/2, x/2, \ldots, \ldots, x/2)$.
3. Define $\boldsymbol{\tau}$ to be a price vector that has the maximum profit
   among $\{\boldsymbol{\tau}_x\}_{x \in [s,\ell]}$, i.e.,
       $\textsc{Profit}(\boldsymbol{\tau}) = \max_{x \in [s,\ell]} \textsc{Profit}(\boldsymbol{\tau}_x)$.
4. Output the price vector $\boldsymbol{p}$ given by
       $\boldsymbol{p} = \begin{cases} \boldsymbol{\sigma} & \textsc{Profit}(\boldsymbol{\sigma}) \geq \textsc{Profit}(\boldsymbol{\tau}); \\ \boldsymbol{\tau} & \textsc{Profit}(\boldsymbol{\sigma}) < \textsc{Profit}(\boldsymbol{\tau}). \end{cases}$
```

Fig. 5 The algorithm $\textsc{Alg}^{\mathrm{sl}}_{\mathrm{graph}}$ for GRAPH_SL.

From the definition of the price vector $\boldsymbol{\sigma}$ defined in Step 1, it follows that $\textsc{Profit}(\boldsymbol{\sigma}) \geq \textsc{Opt}_{\mathrm{sl}}(G)$. For each $x \in [s, \ell]$, let m^{sl}_x be the number of selfloops in E_{sl} with valuation x and m^{c}_x be the number of edges in E_{c} with valuation x. From the definition of the price vector $\boldsymbol{\tau}_x$, we have that for each $x \in [s, \ell]$,

$$\begin{aligned} \textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\tau}_x) &\geq \frac{x}{2} m^{\mathrm{sl}}_x + \frac{x}{2} m^{\mathrm{sl}}_{x+1} + \cdots + \frac{x}{2} m^{\mathrm{sl}}_{\ell}; \\ \textsc{Profit}_{\mathrm{c}}(\boldsymbol{\tau}_x) &= x m^{\mathrm{c}}_x + x m^{\mathrm{c}}_{x+1} + \cdots + x m^{\mathrm{c}}_{\ell}. \end{aligned}$$

From the definitions of m^{sl}_x and m^{c}_x, it follows that

$$\begin{aligned} \textsc{Opt}_{\mathrm{sl}}(G) &\leq s m^{\mathrm{sl}}_s + (s+1) m^{\mathrm{sl}}_{s+1} + \cdots + \ell m^{\mathrm{sl}}_{\ell}; \\ \textsc{Opt}_{\mathrm{c}}(G) &\leq s m^{\mathrm{c}}_s + (s+1) m^{\mathrm{c}}_{s+1} + \cdots + \ell m^{\mathrm{c}}_{\ell}. \end{aligned}$$

Then from the definition of the price vector $\boldsymbol{\tau}$ computed in Step 3, we have that

$$\begin{aligned} \frac{\textsc{Opt}_{\mathrm{sl}}(G)}{2} &\leq \textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\tau}_s) + \sum_{x=s+1}^{\ell} \frac{\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\tau}_x)}{x} \\ \textsc{Opt}_{\mathrm{c}}(G) &\leq \textsc{Profit}_{\mathrm{c}}(\boldsymbol{\tau}_s) + \sum_{x=s+1}^{\ell} \frac{\textsc{Profit}_{\mathrm{c}}(\boldsymbol{\tau}_x)}{x}. \end{aligned}$$

Since $\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\tau}_x) + \textsc{Profit}_{\mathrm{c}}(\boldsymbol{\tau}_x) = \textsc{Profit}(\boldsymbol{\tau}_x)$ for each $x \in [s, \ell]$, we have that

$$\begin{aligned} &\frac{\textsc{Opt}_{\mathrm{sl}}(G)}{2} + \textsc{Opt}_{\mathrm{c}}(G) \\ &\leq \textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\tau}_s) + \sum_{x=s+1}^{\ell} \frac{\textsc{Profit}_{\mathrm{sl}}(\boldsymbol{\tau}_x)}{x} \end{aligned}$$

$$
\begin{aligned}
&+ \textsc{Profit}_{\mathrm{c}}(\boldsymbol{\tau}_s) + \sum_{x=s+1}^{\ell} \frac{\textsc{Profit}_{\mathrm{c}}(\boldsymbol{\tau}_x)}{x} \\
&= \textsc{Profit}(\boldsymbol{\tau}_s) + \sum_{x=s+1}^{\ell} \frac{\textsc{Profit}(\boldsymbol{\tau}_x)}{x} \\
&\leq \textsc{Profit}(\boldsymbol{\tau}) + \sum_{x=s+1}^{\ell} \frac{\textsc{Profit}(\boldsymbol{\tau})}{x} \\
&= \left(1 + \sum_{k=s+1}^{\ell} \frac{1}{k}\right) \cdot \textsc{Profit}(\boldsymbol{\tau}) \\
&\leq \left(1 + \ln \frac{\ell}{s}\right) \cdot \textsc{Profit}(\boldsymbol{\tau}),
\end{aligned}
$$

where the 2nd inequality follows from the definition of the price vector $\boldsymbol{\tau}$ defined in Step 3. Thus it follows that

$$
\begin{aligned}
&\textsc{Profit}(\boldsymbol{p}) = \max \{\textsc{Profit}(\boldsymbol{\sigma}), \textsc{Profit}(\boldsymbol{\tau})\} \\
&\geq \max \left\{ \textsc{Opt}_{\mathrm{sl}}(G), \frac{\textsc{Opt}_{\mathrm{sl}}(G)/2 + \textsc{Opt}_{\mathrm{c}}(G)}{1 + \ln \frac{\ell}{s}} \right\} \\
&\geq \frac{1}{3 + 2 \ln \frac{\ell}{s}} \cdot \textsc{Opt}_{\mathrm{sl}}(G) \\
&\qquad + \frac{2\left(1 + \ln \frac{\ell}{s}\right)}{3 + 2 \ln \frac{\ell}{s}} \cdot \frac{\textsc{Opt}_{\mathrm{sl}}(G)/2 + \textsc{Opt}_{\mathrm{c}}(G)}{1 + \ln \frac{\ell}{s}} \\
&= \frac{2}{3 + 2 \ln \frac{\ell}{s}} \cdot \{\textsc{Opt}_{\mathrm{sl}}(G) + \textsc{Opt}_{\mathrm{c}}(G)\} \\
&= \frac{2}{3 + 2 \ln \frac{\ell}{s}} \cdot \textsc{Opt}(G),
\end{aligned}
$$

for the price vector $\boldsymbol{p}$ defined in Step 4. ■

5. Concluding Remarks

In this paper, we have shown that there exist approximation preserving reductions between any two of the problems such as the graph vertex pricing problem with selfloops GRAPH_SL, the graph vertex pricing problem with no selfloops GRAPH_NSL, the bipartite graph vertex pricing problem with no selfloops BPT_NSL, the bipartite oneway highway problem BPT_OWHW, the line highway problem LINE_HW, and the cycle highway problem CYC_HW (see Lemma 3.1–3.9). We have also shown that under the coupon model, there exists a $(1 + \ln \frac{\ell}{s})$-approximation algorithm $\textsc{Alg}^{\mathrm{nsl}}_{\mathrm{graph}}$ for GRAPH_NSL with $[s, \ell]$-valuations, which implies that the other problems have algorithms with good approximation ratio (see Corollaries 4.1–4.4).

All of the results in this paper depend on $r = \ell/s$ (the ratio between the largest valuation ℓ and the smallest valuation s). This implies that our approximation algorithms given in this paper may work well for smaller r (but may not behave well for larger r). Thus the following issues remain as future works.

(1) Under all the price models, design algorithms with better approximation ratio for the item pricing problems such as GRAPH_SL, GRAPH_NSL, BPT_NSL, BPT_OWHW, LINE_HW, and CYC_HW.

(2) Under all the price models, derive lower bounds on the approximation ratio for the item pricing problems such as GRAPH_SL, GRAPH_NSL, BPT_NSL, BPT_OWHW, LINE_HW, and CYC_HW.

Acknowledgments

The authors wish to thank anonymous referees for valuable comments and suggestions to improve the quality and readability of the paper.

References

[1] M.F. Balcan and A. Blum, "Approximation algorithms and online mechanisms for item pricing," Proc. 7th ACM Conference on Electronic Commerce, pp.29–35, 2006.

[2] M.F. Balcan, A. Blum, T-H.H. Chan, and M.T. Hajiaghayi, "A theory of loss leaders: Making money by pricing below cost," School of Computer Science, Carnegie Mellon University, Technical Report, CMU-CS-07-142, 2007.

[3] M.F. Balcan, A. Blum, T-H.H. Chan, and M.T. Hajiaghayi, "A theory of loss leaders: Making money by pricing below cost," Proc. 3rd International Workshop on Internet and Network Economics, Lecture Notes in Computer Science 4858, pp.293–299, 2007.

[4] P. Briest and P. Krysta, "Single-minded unlimited supply pricing on sparce instances," Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.1093–1102, 2006.

[5] P. Briest and P. Krysta, "Buying cheap is expensive: Hardness of non-parametric multi-product pricing," Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.716–725, 2007.

[6] P. DeGraba, Volume Discounts, Loss Leaders, and Competition for More Profitable Customers, Federal Trade Commission Bureau of Economics, Working Paper 260, 2003.

[7] E.D. Demaine, U. Feige, M.T. Hajiaghayi, and M.R. Salavatipour, "Combination can be hard: Approximability of the unique coverage problem," Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.162–171, 2006.

[8] A. Grigoriev, J. van Loon, R. Sitters, and M. Uetz, "How to sell a graph: Guidelines for graph retailers," Proc. 32nd International Workshop on Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science 4271, pp.125–136, 2006.

[9] V. Guruswami, J.D. Hartline, A.R. Karlin, D. Karger, C. Kenyon, and F. McSherry, "On profit-maximizing envy-free pricing," Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.1164–1174, 2005.

[10] M.G. Luby and A. Wigderson, "Pairwise independence and derandomization," Technical Report: CSD-95-880, 1995.

[11] M.G. Luby and A. Wigderson, Pairwise Independence and Derandomization, Now Publishers, 2006.

[12] N. Robertson and P.D. Seymour, "Graph minors I: Excluding a forest," Combin. Theory Ser. B, vol.35, pp.39–61, 1983.

Ryoso Hamane was born in Hiroshima, Japan, in 1985. He received the B.E. degree in 2007 from Tokyo Institute of Technology, and since 2007, he has been a master course student in the Department of Information Processing at Tokyo Institute of Technology. His current research interests are discrete algorithms, approximation algorithms, and complexity theory.

Toshiya Itoh was born in Urawa, Japan, in 1959. He received the B.E., M.E., and Dr. Eng. degree in electronic engineering in 1982, 1984, and 1988, respectively from Tokyo Institute of Technology, Tokyo, Japan. From 1985 to 1990, he was an Assistant Professor in the Department of Electrical and Electronic Engineering at Tokyo Institute of Technology, and from 1990 to 1992, he was a Lecturer in the Department of Information Processing at Tokyo Institute of Technology. From 1992 to 2001, he was an Associate Professor in the Department of Information Processing at Tokyo Institute of Technology, and since 2001, he has been a Professor in the Global Scientific Information and Computing Center at Tokyo Institute of Technology. His current interests are discrete algorithms, combinatorics, and complexity theory. Dr. Itoh is a member of the Information Processing Society of Japan, the Association for Computing Machinery, and LA.

Kouhei Tomita was born in Kanagawa, Japan, in 1986. He received the B.E. degree in 2008 from Tokyo Institute of Technology, and since 2008, he has been a master course student in the Department of Computational Intelligence and System Science at Tokyo Institute of Technology. His current research interests are discrete algorithms, approximation algorithms, and complexity theory.