Graph Pricing With Limited Supply

by

Maryam Mahboub

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

In this thesis, we study approximation algorithms for graph pricing where we have a set of items V and a set of customers X where each customer $i \in X$ has a budget b_i and is interested in a bundle of items $S_i \subseteq V$ with $|S_i| \leq 2$. However, there is a limited supply of each item: we only have μ_v copies of item v to sell for each $v \in V$. We should assign prices $p(v)$ to each $v \in V$ and chose a subset $Y \subseteq X$ of customers so that each $i \in Y$ can afford their bundle ($p(S_i) \leq b_i$) and at most μ_v chosen customers have item v in their bundle for each item $v \in V$. Each customer $i \in Y$ pays $p(S_i)$ for the bundle they purchased: our goal is to do this in a way that maximizes revenue.

Such pricing problems have been studied from the perspective of *envy-freeness* where we also must ensure that $p(S_i) \geq b_i$ for each $i \notin Y$. However, the version where we simply allocate items to customers after setting prices and do not worry about the envy-free condition has received less attention.

With unlimited supply of each $v \in V$, Balcan and Blum (2006) give a 4-approximation for graph pricing which was later shown to be tight by Lee (2015) unless the Unique Games conjecture fails to hold.

Our main result is an 8-approximation for the capacitated case via local search. If all capacities are bounded by a constant C, we further show a multi-

swap local search algorithm yields an $\left(4 \cdot \frac{2C-1}{C} + \epsilon\right)$-approximation. We also give a $(4+\epsilon)$-approximation in simple graphs through LP rounding when all capacities are very large as a function of ϵ.

The reduction by Balcan and Blum to the case of bipartite graphs where all items on one side must be assigned a price of 0 holds in this setting as well. However, unlike the unlimited supply setting, the resulting problem remains **APX**-hard even if all items have at most 4 copies to sell. We also show our multi-swap analysis is tight using an interesting construction based on regular, high-girth graphs.

Preface

I, Maryam Mahboub, declare that this thesis titled, 'Graph Pricing With Limited Supply' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

"No two things have been combined better than knowledge and patience."

Prophet Muhammad (peace be upon him)

To:

The great savior of the world

I think there is a world market for maybe five computers.

– Thomas J. Watson, IBM Chairman, 1943.

Acknowledgements

There are many people I want to thank for both supporting me through my degree as for their help with this thesis. First and foremost, I want to thank my supervisor, Zachary Friggstad. You have been a great teacher, mentor, and role-model who has pushed me to become a better student and researcher. While I have had my fair share of failures and stumbles along the way, you have shown me how much more I can achieve, and how much further I can still grow as a researcher.

I also want to thank Mohammad R. Salavatipour for not only providing insightful and constructive comments in this thesis, but for being great colleague. You have been a inspiration to me, and I have valued your guidance, especially during the more difficult times over the past two years.

I would also like to thank to my beloved husband, Mirmahdi Rahgoshay. Thank you for supporting me for everything, and especially I can't thank you enough for encouraging me throughout this experience.

Finally I thank my God, for helping me through all the difficulties. I have experienced Your guidance day by day. You are the one who let me finish my degree. I will keep on trusting You for my future. Thank you, Lord.

Contents

List of Figures

Chapter 1

Introduction

Choosing prices to sell items in order to maximize revenue is a complicated task even in environments where one can be certain of customer behaviour. Indeed, many so-called *pricing problems* have been studied in combinatorial optimization. One popular model is this: a collection of items V is available to be sold where we have $\mu_v \in \mathbb{Z}_{\geq 0} \cup \{\infty\}$ copies of item $v \in V$. Additionally, we are given a collection of customers X where each $i \in X$ has some budget $b_i \geq 0$. In the *single-minded* setting, each customer $i \in X$ is interested in a bundle $S_i \subseteq V$. We must assign prices $p : V \to \mathbb{R}_{\geq 0}$ to the items and sell them to some customers $Y \subseteq X$ while respecting two constraints:

- **Affordability**: $p(S_i) := \sum_{v \in S_i} p(v) \leq b_i$ for $i \in Y$, and
- **Supply Constraints**: $|\{i \in Y : v \in S_i\}| \leq \mu_v$ for $v \in V$.

That is, each customer that purchases their bundle can afford it and no item is oversold. Such a solution (p, Y) is said to be **feasible**, and the goal is to find a feasible (p, Y) maximizing revenue, *i.e.* $\sum_{i \in Y} p(S_i)$.

Much attention has been given to the **envy-free** setting, where a feasible solution must additionally satisfy the property $p(S_i) \geq b_i$ for $i \notin Y$ or to the **unlimited supply** setting where $\mu_v = \infty$ for each $v \in V$. Observe that in the unlimited supply setting, any pricing yields an envy-free solution by simply

choosing the customers that can afford the price. However, the problem still remains **APX**-hard in this relaxed setting (unlimited supply) and, further, is hard to approximate within a factor better than 4 unless the Unique Games Conjecture, an open problem in computer theory, fails to hold, see the prior works section.

We study single-minded pricing problems yet *without* the envy-free constraint. This is a natural variant of pricing problems where customer satisfaction is less of a concern than overall revenue generation. To the best of our knowledge, it seems that pricing problems without the envy-free condition like this have received virtually no attention so far except in simpler cases of unlimited supply where envy-freeness is a superfluous constraint (*i.e.* any solution can be trivially be made envy-free without losing revenue).

More specifically we mainly consider the case when $|S_i| = 2$ for each customer i. That is, the set of customers can be thought of as edges E in a graph $G = (V, E)$ with vertex capacities and, perhaps, parallel edges.

1.1 Preliminaries

We begin by formalizing the terminology we will use throughout this thesis. The definitions given here are adapted from [34], [36], and [35].

1.1.1 Graph

A graph G is defined by its finite edge set $E(G) = \{e_1, e_2, ..., e_m\}$ and finite vertex set $V(G) = \{v_1, v_2, ..., v_n\}$, where each edge $e \in E(G)$ is an unordered pair of vertices in $V(G)$. To simplify notation, we may drop the parameters of V and E when the graph is clear from context, and instead denote G as the pair (V, E). We also consider *directed* graphs; in a directed graph G, each edge $e \in E(G)$ is an *ordered* pair of vertices. We use the same notation as for undirected graphs.

In undirected graph for each edge $e = uv \in E(G)$, we say u and v are *adjacent* and e is *incident* to u and v. The *neighbours* of a vertex v are the vertices u such that u and v are *adjacent*; we denote this set as $N_G(v)$, or simply $N(v)$ when G is clear from context.

A *subgraph* of a graph G is a graph H, where H is obtained from G by *deleting* some edges and/or some vertices (and their incident edges) from G. We notate this relation as $H \subseteq G$, and may simply say that G *contains* H or H is *in* G. A *subgraph* $H \subseteq G$ is *spanning* if $V(H) = V(G)$.

Multigraph

A multigraph (in contrast to a simple graph) is a graph with a multiset of edges which is permitted to have multiple edges (also called parallel edges), that is, edges that have the same end nodes. Thus two vertices may be connected by more than one edge. Also, multigraphs could have edges that connects one vertex to itself which are called self loops.

Bipartite Graph

A bipartite graph is a graph whose vertices can be divided into two disjoint sets U and V such that every edge is incident to exactly one vertex in U and one vertex in V. Vertex sets U and V are usually called the parts of the graph.

Hypergraph

A hypergraph H is a pair $H = (X, E)$ where X is a set of elements called vertices, and E is a set of non-empty subsets of X called edges. Therefore, E is a subset of $P(X) \setminus \{\emptyset\}$, where $P(X)$ is the power set of X. The size of vertex set is called the order of the hypergraph, and the size of edges set is the size of the hypergraph. A k-hypergraph is a hypergraph such that all the edges have cardinality k (a subset of vertices with size k).

Walk in Graph

A walk in graph G is a finite non-empty sequence $W = v_0e_1v_1e_2v_2...e_kv_k$, whose terms are alternately vertices and edges, such that, for $1 \leq i \leq k$, the ends of

e_i are v_{i-1} and v_i. We say that W is a walk from v_0 to v_k. The vertices v_0 and v_k are called the origin and terminus of W, respectively, and $v_1, v_2, \ldots, v_{k-l}$ its internal vertices. The integer k is the length of W. A walk is closed if it has positive length and its origin and terminus are the same.

Eulerian Circuit

A Eulerian circuit in a graph is a walk that visits every edge exactly once and ends at the starting node (terminus is same as origin) of the walk.

Cycle in Graph

A closed walk whose origin and internal vertices are distinct is a cycle. Just as with paths we sometimes use the term 'cycle' to denote a graph corresponding to a cycle. A cycle of length k is called a $k - cycle$.

Girth of the Graph

The girth of a graph is the length of a shortest cycle contained in the graph. If the graph does not contain any cycles its girth is defined to be infinity.

1.1.2 Optimization Problems and Approximation Algorithms

Decision Problems and NP

A decision problem is a problem that can be answered with either "yes" or "no". We view decision problems as languages. A language is a subset of binary strings over the alphabet $\{0, 1\}$. Language L corresponding to some decision problem is the set of all strings in L that encode "yes" instances to the problem.

A language $L \in \mathbf{NP}$ if there are polynomials p, q and a Turing machine M (called a verifier) such that for each string $x \in \{0, 1\}^*$, the following holds. If $x \in L$, then a certificate string y of length at most $p(|x|)$ must exist such that $M(x, y)$ accepts in at most $q(|x|)$ steps. Otherwise, for all strings y of length at most $p(|x|)$, $M(x, y)$ rejects in at most $q(|x|)$ steps. **NP** is therefore the class

of all languages for which there are short and quickly verifiable yes-certificates.

Let L_1 and L_2 be two languages in **NP**. A language L_1 reduces to L_2 if there is a Turing machine that, given the string $x \in \{0,1\}^*$, outputs a string y such that $y \in L_2$ if and only if $x \in L_1$, and does so in $poly(|x|)$ steps. A language L is **NP**-hard if for every language $L_0 \in \mathbf{NP}$, L_0 reduces to L. A language L is **NP**-complete if L is **NP**-hard and $L \in \mathbf{NP}$.

Optimization problems

An **NP**-optimization problem Π consists of:

- A set of valid *instances* D_Π, where we can determine if some instance $I \in D_\Pi$ in time polynomial in $|I|$. We assume all *instances* $I \in D_\Pi$ can be expressed as finite binary strings; this implies all numeric values could be integer or rational. The *size* of an instance I, written $|I|$, is the number of bits needed to express it.

- A set of feasible solutions $\mathcal{S}_\Pi(I)$ for each instance $I \in D_\Pi$, where we can determine if $S \in \mathcal{S}$ in time $poly(|I|)$. The length of each solution must be polynomially bounded in the length of I.

- An *objective function* obj_Π that assigns each instance-solution pair (I, s) a non-negative value, computable in time that is polynomial in $|I|$.

We also specify whether Π is a *minimization problem* or a *maximization problem*. For a minimization/maximization problem Π and instance $I \in D_\Pi$, an *optimal solution* is a feasible solution $s \in S_\pi(I)$ that minimizes/maximizes the value of obj_Π ; that is, $argmin_{s \in S_\Pi}\ obj_\Pi(I, s)$ or $argmax_{s \in S_\Pi}\ obj_\Pi(I, s)$, respectively. We denote such a solution as $OPT_\Pi(I)$, or simply OPT if the problem and instance are clear from context. We slightly abuse this notation by using OPT to also refer to the objective value of the optimum solution, when the type of OPT is clear from context.

An **NP** optimization problem Π gives rise to a class of **NP** decision problems Π', by asking if a feasible solution of at most/at least some objective value exists (for minimization/maximization problems, respectively). A polynomial time algorithm that solves Π can thus be used to answer the decision problem Π'. On the other hand, proving that the decision version of a problem $I' \in \Pi'$ is hard in some sense, shows that the optimization version $I \in \Pi$ is at least as hard as I'. For example if we prove that I' in **NP**-hard, then it means that I is **NP**-hard too.

Approximation algorithms

Let Π be a minimization (maximization) problem, and let $\alpha : \mathbb{Z}^+ \to \mathbb{Q}^+$ be a function such that $\alpha(I) \geq 1$ for all inputs $I \in D_\Pi$. An algorithm A is an $\alpha - approximation$ for Π if, for all instances I, A returns a feasible solution $S \in S_\Pi(I)$ such that $obj_\Pi(I, S) \leq \alpha(|I|).OPT_\Pi(I)$ $\left(obj_\Pi(I, S) \geq \frac{OPT_\Pi(I)}{\alpha(|I|)}\right)$ and the running time is bounded by $\mathbf{poly}(|I|)$. The function α is called the approximation ratio of A.

It is sometimes difficult to obtain an algorithm that meets this definition exactly. We might need to relax the running time bound, for example to a quasi-polynomial factor, which is $O(|I|^{\log^c(|I|)})$, where c is a constant. Or, the algorithm makes random choices, and so the approximation ratio only holds in expectation over all random choices. We still loosely refer to these as approximation algorithms, although we will state such relaxations explicitly.

An algorithm A is an approximation scheme for the minimization (maximization) problem Π if for the valid instance I and error parameter $\epsilon > 0$, it returns a feasible solution S such that $obj_\Pi(I, S) \leq (1 + \epsilon).OPT_\Pi(I)$ $\left(obj_\Pi(I, S) \geq (1 - \epsilon).OPT_\Pi(I)\right)$. We call A a *polynomial time approximation scheme* ($PTAS$) if its running time is $\mathbf{poly}(|I|)$ for each fixed ϵ. We call A a *fully polynomial time approximation scheme* ($FPTAS$) if its running time is $\mathbf{poly}(|I|, \frac{1}{\epsilon})$ for each fixed ϵ.

Problem Π is said to be in the class **PTAS** or **FPTAS** if it admits the respective approximation scheme. It is said to be in the class **APX** if it admits *any* constant approximation.

Let Π and Π' be two optimization problems. Π **PTAS**-reduces to Π' if there exists an algorithm A and function $c : \mathbb{R}^+ \to \mathbb{R}^+$, where for each valid instance I of Π and each fixed $\epsilon > 0$,

- Algorithm A returns an instance $I' = A(I, \epsilon)$ of Π' in time $\mathbf{poly}(|I|)$, such that if I is feasible then I' is feasible, and
- Given any feasible solution $s' \in {S_\Pi}'(I')$, there exists a feasible solution $s \in S_\Pi(I)$ such that if ${obj_\Pi}'(I', s') \leq (1 + c(\epsilon)).{OPT_\Pi}'(I')$, then $obj_\Pi(I, s) \leq (1 + \epsilon).OPT_\Pi(I)$

An optimization problem Π is said to be **APX**-hard if for every other problem $\Pi' \in \mathbf{APX}$, Π' **PTAS**-reduces to Π. If in addition $\Pi \in \mathbf{APX}$, then Π is said to be **APX**-complete.

Hardness of approximation

Roughly speaking, a *hardness* proof shows that a certain optimization problem cannot be approximated better than some threshold assuming certain complexity assumptions. As an extreme example, it was shown in [37] that the maximum independent set problem cannot be approximated better than $O(n^{1-\epsilon})$ for any constant $\epsilon > 0$ assuming $\mathbf{P} \neq \mathbf{NP}$, ruling out all but the most trivial approximations. A less extreme example, implied by the PCP theorem, is that approximating Max-3SAT better than $(1 + \epsilon)$ for some $\epsilon > 0$ is **NP**-hard, ruling out a **PTAS** assuming $\mathbf{P} \neq \mathbf{NP}$ [34]. Since this problem is also **APX**-complete, a consequence of this hardness is that for any **APX**-hard optimization problem Π, $\Pi \notin \mathbf{PTAS}$ unless $\mathbf{P} = \mathbf{NP}$.

Unique Games Conjecture

In computational complexity theory, the unique games conjecture (often re-

ferred to as UGC) is a conjecture made by Subhash Khot in 2002 [26]. The conjecture postulates that the problem of determining the approximate value of a certain type of game, known as a unique game, is **NP**-hard. It has broad applications in the theory of hardness of approximation. If it is true, then for many important problems it is not only impossible to get an exact solution in polynomial time (as postulated by the **P** versus **NP** problem), but also impossible to get a good polynomial-time approximation.

However, UGC is not the only assumption to help for such an inapproximability results. For example it has been shown that the Minimum Vertex Cover problem is NP-hard to approximate to within a factor of $\sqrt{2}$, by assuming the traditional assumption of $\mathbf{P} \neq \mathbf{NP}$ [27], [28].

1.1.3 Linear Programming

Many problems in **NP** can be formulated as an *integer program* that describes the problem. Let $c \in \mathbb{Q}^n$, $b \in \mathbb{Q}^m$ be vectors, and $A = (a_{ij}) \in \mathbb{Q}^{m \times n}$ be a matrix. Let $u \cdot v$ denote the dot-product of two vectors u and v. The integer programming problem is to find a non-negative integer vector $x \in Z^{+n}$ maximizing the value $c \cdot x$, satisfying:

$$A \cdot x \leq b$$

Note that we can use this definition to define minimization problems as well (i.e. by maximizing $-c \cdot x$), and allow for $\geq$ and $=$ constraints.

Finding such a binary vector, or determining if such a vector even exists, is itself an **NP**-hard problem in general (otherwise, we could use integer programming to solve other **NP**-hard problems). Instead, suppose we relax this problem: instead of trying to find a binary vector x, we try to find a satisfying $x \in \mathbb{Q}^n$. This yields a *linear program*:

$$\begin{aligned} \textbf{maximize} \quad & c \cdot x & \textbf{(LP)} \\ \textbf{subject to} \quad & Ax \leq b, & (1.1) \\ & x \geq 0 & (1.2) \end{aligned}$$

It is usually more convenient to explicitly write out the constraints and the objective function rather than specifying A, b, c directly, as in the following (equivalent) **LP**:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j \leq b_j, \quad i = 1, ..., m \\ & x_j \geq 0, \quad j = 1, ..., n \end{aligned} \tag{1.3}$$

We say that we "solve" a linear program if we either determine no solution x exists, the value $c \cdot x$ is unbounded, or return a solution minimizing the objective c · x. Unlike integer programs, linear programs can be solved in time polynomial in n, m, and the number of bits Δ required to write the rational entries of A, b, and c; one such approach is the *interior point method* (see, for example, [24]).

Usefulness in approximations

Linear programming is a useful tool to build approximation algorithms with. The general procedure for a minimization (maximization) problem is to write the integer program, relax its constraints that force the variables to be non-negative integers, by allowing the variables to take any non-negative real numbers, solve it, and try to round the fractional result to an integer solution in polynomial time, without either violating constraints or increasing (decreasing) the objective value significantly. If we can do this while only increasing

(decreasing) the objective value by a factor of $f(n)$, where $n = |x|$, then we will have an $f(n)$-approximation to the original problem.

We say a linear program of a minimization (maximization) problem has an *integrality gap* of $f(n)$ if for an optimum solution x^* and optimum solution $\bar{x}$ for the corresponding integer program, $\frac{c \cdot x^*}{c \cdot \bar{x}} \leq f(n)$ $\left(\frac{c \cdot x^*}{c \cdot \bar{x}} \geq f(n)\right)$.

1.1.4 Local Search Algorithms

A local search algorithm starts with an arbitrary feasible solution, and then it iteratively improves the current solution by selecting a *neighboring solution* with a better objective function value. The algorithm stops when no further improvement is possible. Neighboring solutions of a given feasible solution are determined by a set of *local operations*.

In combinatorial optimization we look for a solution S from the solution space A, that optimizes an objective function $c : A \to \mathbb{Q}$. A local search algorithm is defined by its local operations. A local operation transforms a solution $S \in A$ into a new solution S' by for example, adding, removing or exchanging elements of S with elements not in S. Local operations are usually simple and they should be able to be performed quickly, in polynomial time. A *neighborhood function* N is defined by the local operations. For each solution $S \in A$, $N(S)$ includes all the solutions $S' \in A$ that can be obtained by performing a single local operation on S. $|N(S)|$ is polynomially bounded.

A local optimum solution S is a solution for which we have $c(S) \geq c(S')$ for all $S' \in N(S)$. Note that the actual optimum solution of the problem (global optimum) is a local optimum solution too, but we could have more than one local optimum solution as well. The main idea of a local search algorithm is to define the local operations such that we can prove any local optimum solution is not too far from the global optimum solution.

Locality Gap

The *locality gap* of a local search algorithm is the largest ratio of the value of a local optimal solution produced by the algorithm to the value of a corresponding global optimal solution. Let s_I be a local optimal solution produced by a local search algorithm for some instance I of a maximization problem P and let $s^*(I)$ be a global optimal solution for I, then the locality gap of the algorithm is defined as $min_{I \in P} \frac{c(s(I))}{c(s^*(I))}$, where $c(s(I))$ is the cost of $s(I)$ and $c(s^*(I))$ is the cost of $s^*(I)$. Similarly, for minimization problems P the locality gap is defined as $max_{I \in P} \frac{c(s(I))}{c(s^*(I))}$.

1.1.5 Probability Inequalities

Markov's Inequality

In probability theory, Markov's inequality gives an upper bound for the probability that a non-negative function of a random variable is greater than or equal to some positive value.

Lemma 1 (Theorem 3.2 in [30]). *Let Y be a random variable assuming only non-negative values. Then for all $t \in R^+$:*

$$\Pr\big[Y \geq t] \leq \frac{\mathbf{E}[Y]}{a}$$

Equivalently,

$$\Pr\big[Y \geq t\mathbf{E}[Y]] \leq \frac{1}{t}$$

Chernoff Bound

The Chernoff bound, named after Herman Chernoff but due to Herman Rubin, gives exponentially decreasing bounds on tail distributions of sums of independent random variables. there are many forms (inequalities) of Chernoff bound, but we mention the one we want to use in the thesis.

Lemma 2 (Theorem 1.1 in [13]). *Let $X = \sum_{i=1}^{n} X_i$ where X_i, $1 \leq i \leq n$ are* **independent** *random variables distributed in $[0, 1]$. Then for every $\epsilon > 0$:*

$$\Pr\big[X \geq (1+\epsilon)E[X]\big] \leq \exp\Big(-\frac{\epsilon^2}{3}E[X]\Big).$$

1.2 Problem Considered

We focus on the following problem.

Definition 1.2.1. Let $G = (V, E)$ be a graph with vertex capacities $\mu_v \in \mathbb{Z}_{\geq 0} \cup \{\infty\}$ where each $e = uv \in E$ has a budget $b_e \geq 0$ and is interested in the bundle of vertices $\{u, v\}$. In Capacitated Graph Pricing, we want to find a pricing $p : V \to \mathbb{R}_{\geq 0}$ and $F \subseteq E$ such that (p, F) is a feasible solution to the pricing problem with considering the capacity of each vertex and the budget of each edge. The goal is to maximize revenue: $\sum_{e=uv\in F} p(u) + p(v)$.

Note that, for edges, we use shorthand notation like $e = uv \in E$ when we want to consider an edge $e \in E$ in some graph $G = (V, E)$ and also want to name the endpoints u, v of e. This allows us to name distinct edges interested in the same bundle of items (*i.e.* $\{u, v\}$ and $\{w, v\}$).

All of our algorithmic results extend in a simple way to the case where each customer is interested in a bundle of size *at most* 2, but it is slightly simpler to describe the algorithms and their analysis for the case where each customer wants precisely two different items. Unless otherwise stated, the graph G may have parallel edges. We use the term *simple graph* to indicate it does not have parallel edges.

To get approximations for Capacitated Graph Pricing, we use the reduction from Balcan and Blum [4] to reduce to the case of a bipartite graph where all items on one side will be priced 0. Specifically, we consider the following problem.

Definition 1.2.2. In L-Sided Pricing, we are given a Capacitated Graph Pricing instance in a bipartite graph $(L \cup R, E)$. A feasible solution (p, F) must also have $p(v) = 0$ for $v \in R$.

1.3 Prior Work

The basic model of pricing problems of this sort were introduced by Guruswami *et al.* [21]. Among other results, they have given an $O(\log n + \log m)$-approximation for the case of single-minded pricing without item capacities if we have n items and m customers. Here, the bundle S_i for each customer i may be any subset of items (not just size 2). This was later improved by Briest and Krysta to an $O(\log D + \log k)$-approximation where each set has size at most k and each item appears in at most D sets [5]. Chalermsook *et al.* show that for any constant $\epsilon > 0$ there is no $O(\log^{1-\epsilon}(m+n))$-approximation unless $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{\mathbf{polylog}(n)})$ [7], so the logarithmic approximation of [5] is essentially tight.

If all customers are interested in a set of size at most k, Balcan and Blum give an $O(k)$-approximation for the uncapacitated pricing which specializes to a 4-approximation in the case $k = 2$ [4]. Amazingly, this may also be tight: building on work by Khadekar *et al.* [25], Lee showed that there is no $(4-\epsilon)$-approximation when $k = 2$ for any constant $\epsilon > 0$ unless the **Unique Games Conjecture** [29] fails.

Cheung and Swamy studied the envy-free variant of capacitated pricing problems [12]. As mentioned earlier, they show that **LP**-based approximations that choose the maximum-profit set of customers for given prices translate to approximation algorithms for envy-free pricing with capacities while losing an $O(\log \mu_{\max})$-factor. In particular, for envy-free Capacitated Graph Pricing they get an $O(\log \mu_{\max})$-approximation.

Hartline and Koltun design near-linear and near-cubic time approxima-

tion schemes under the assumption that the number of distinct items for sale is constant [22]. In the unlimited supply case, they give a near-linear time approximation schemes for both the problem. Specifically, for unit-demand consumers a $(1+\epsilon)$-approximation is achieved in time $O\big(n \log(\frac{1}{\epsilon}\log\frac{n}{\epsilon}) + (\frac{1}{\epsilon}\log\frac{n}{\epsilon})^{m(m+1)}\big)$ for n consumers, $m = O(1)$ items, and an arbitrarily small $\epsilon > 0$. For single-minded consumers a $(1+\epsilon)$-approximation is achieved in time $O\big((n+(\frac{1}{\epsilon}\log\frac{n}{\epsilon})^m)\log(\frac{1}{\epsilon}\log\frac{n}{\epsilon})\big)$. For the more general limited supply case and for unit-demand consumers they give a $(1+\epsilon)$-approximation algorithm for the limited supply (envy-free) pricing problem that runs in time $O(n^3 \log^m_{1+\epsilon} n)$.

Many other variants of envy-free pricing problems have been studied. For example, it could be that each customer is interested in acquiring just a single item from their subset (rather than all items). This was also studied in [21] and follow-up work [14] where they obtain fairly general results that relate the approximability of the profit-maximization problem to the corresponding *social-welfare-maximization* (SWM) problem, which is the problem of finding an allocation $(\{S_1, \ldots, S_n\})$ satisfying the capacity constraints that has maximum total value $\sum_j v_j(S_j)$. This yields an $O(\log c_{max})$-approximation for the profit-maximization problem, where c_{max} is the maximum item-supply.

Feldman *et al.* [16] study envy-free (EF) mechanisms for multi-unit auctions with budgeted agents that approximately maximize revenue. In an EF auction, prices are set so that every bidder receives a bundle that maximizes her utility amongst all bundles; They show that the problem of revenue-maximizing EF auctions is **NP**-hard, even for the case of identical items and additive valuations (up to the budget). They also provide a novel algorithm that runs in polynomial time and provides a approximation of 1/2 with respect to the revenue-maximizing EF auction.

Chen *et al.* [11] study the unit-demand envy-free pricing problem faced by a profit-maximizing seller with unlimited supply when there is metric substi-

tutability among the items. More precisely consumer i's value for item j is $v_i - c_{i,j}$, and the substitution costs, $\{c_{i,j}\}$, form a metric. They show that the problem of maximizing revenue with metric substitutability among items can be solved exactly in polynomial time.

Chen and Deng [9] study the revenue maximization envy-free pricing in multi-item markets where there are m items and n potential buyers where each buyer is interested in acquiring one item. The goal is to determine allocations (a matching between buyers and items) and prices of all items to maximize the total revenue given that all buyers are envy-free. They give a polynomial time algorithm to compute a revenue maximization envy-free pricing when every buyer evaluates at most two items a positive valuation, by reducing it to an instance of weighted independent set in a perfect graph and applying the Strong Perfect Graph Theorem. They also show that the problem becomes **NP**-hard if some buyers are interested in at least three items.

Another recent variation is studied by Chen *et al.* [10] where they consider markets consisting of a set of indivisible items, and buyers that have sharp multi-unit demand. This means that each buyer i wants a specific number d_i of items and a bundle of size less than d_i has no value. They focus on the case where each buyer i has a valuation $v_i q_j$ for item j, where v_i and q_j are positive quantities associated with buyer i and item j, respectively. They showed that for envy-free pricing, if the demand of each buyer is bounded by a constant, a revenue maximizing solution can be found efficiently, and the general demand case is shown to be **NP**-hard.

In a different version of the problem Balcan *et al.* [3] consider the problem of pricing n items to maximize revenue when faced with a series of unknown buyers with complex preferences, and show that a simple pricing scheme achieves surprisingly strong guarantees. They show that in the unlimited supply setting, a random single price achieves expected revenue within a loga-

rithmic factor of the total social welfare for customers with general valuation functions, which may not even necessarily be monotone. This generalizes work of Guruswami *et al.* [21], who show a logarithmic factor for only the special cases of single-minded and unit-demand customers. In the limited supply setting, they show that for subadditive valuations, a random single price achieves revenue within a factor of $2^{O(\sqrt{(\log n \log\log n)}}$ of the total social welfare, i.e., the optimal revenue the seller could hope to extract even if the seller could price each bundle differently for every buyer.

Graph pricing problems is another interesting directions that has gained so much attention during recent years. The items can be represented as the edges of an undirected (multi)graph G, where an edge multiplicity larger than one corresponds to multiple copies of the same item. Each customer is interested in purchasing a bundle of edges of G, and we assume that each bundle forms a simple path in G. Each customer has a known budget for her respective bundle, and is interested only in that particular bundle. The goal is to determine item prices and a feasible assignment of items to customers in order to maximize the total profit. Grigoriev *et al.* [20] show some early algorithms for special graph families along with some hardness results.

Other directions have considered more restricted subsets of items in single-minded pricing, for example the customers may be interested in the edges of sub-paths of a tree (the **tollbooth problem**) or a sub-path of a large path (the **highway problem**).

More precisely, An instance of the tollbooth problem consists of an undirected network and a collection of single-minded customers, each of which is interested in purchasing a fixed path subject to an individual budget constraint. The objective is to assign a per-unit price to each edge in a way that maximizes the collective revenue obtained from all customers. The revenue generated by any customer is equal to the overall price of the edges in her de-

sired path, when this cost falls within her budget; otherwise, that customer will not purchase any edge. A deterministic algorithm for the tollbooth problem on trees whose approximation ratio is $O(\log m / \log \log m)$, where m denotes the number of edges in the underlying graph, is provided in [18]. Elbassioni et. al. [15] also study a special case of the tollbooth problem, when all the paths that customers are interested in purchasing go towards a fixed root. In this case, they present an algorithm that returns a $(1-\epsilon)$-approximation, for any $\epsilon > 0$, and runs in quasi-polynomial time, which is $O(n^{\log^c n})$, where c is a constant.

In the highway problem, we are given an n-edge path graph (the highway), and a set of paths (the drivers), each one with its own budget. For a given assignment of edge weights (the tolls), the highway owner collects from each driver the weight of the associated path, when it does not exceed the budget of the driver, and zero otherwise. The goal is to choose weights so as to maximize the profit. The highway problem was shown to be strongly **NP**-hard [15]. In [19] Grandoni and Rothvoss present a polynomial-time approximation scheme (PTAS) for the highway problem, hence greatly improving the understanding of the complexity status of this problem. Their result is based on a novel randomized dissection approach.

Another related approach is the work done by Patrick Briest and Piotr Krysta [6], where they investigate non-parametric unit-demand pricing problems, in which we want to find revenue maximizing prices for products P based on a set of consumer profiles C. A consumer profile consists of a number of non-zero budgets for different products and possibly an additional product ranking. Once prices are fixed, each consumer chooses to buy one of the products she can afford based on some predefined selection rule.

1.4 Main Results

The main contributions of this thesis are the following:

In Chapter 2, we use the *Local Search* technique for L-Sided Pricing to show the following:

Theorem 1. *There is a polynomial-time 2-approximation for* L-Sided Pricing.

Theorem 2. *For any constant $C \geq 2, \epsilon > 0$, there is a polynomial-time $\left(\frac{2C-1}{C} + \epsilon\right)$-approximation for* L-Sided Pricing *if $\mu_v \leq C$ for all $v \in L$.*

In Chapter 3, we consider an alternative approach to get better approximation guarantees as well. Recall that the best known approximation algorithm for unbounded capacity version of the problem has approximation ration of 4. We also show that in the case that all the capacities are large enough we can have a $(4+\epsilon)$-approximation algorithm:

Theorem 3. *For any $\epsilon > 0$, let $C_\epsilon = 3\ln(1/\epsilon)/\epsilon^2 + 1 \geq 0$. Instances of* Capacitated Graph Pricing *in simple graphs satisfying $\mu_v \geq C_\epsilon$, admit a randomized, polynomial-time $(4+\epsilon)$-approximation.*

We also show that it is possible to get an $4k$-approximation for L-Sided Pricing in k-hypergraphs through straightforward rounding of a natural linear programming relaxation that is presented in Chapter 3, and then by using a reduction from Capacitated Graph Pricing to L-Sided Pricing which loses an other ke factor, we would have an $4k^2(\frac{k}{k-1})^{k-1} = O(k^2)$-approximation for Capacitated Graph Pricing:

Theorem 4. *k-Hypergraph* Capacitated Graph Pricing *problem admits a randomized, polynomial-time $(4k^2(\frac{k}{k-1})^{k-1})$-approximation.*

Note for $k = 2$ this is not better than the result which is obtained using the *Local Search* technique for L-Sided Pricing in Theorem 1.

In Chapter 4, we present a few different results that provide further insight into the Locality gaps for single-swap and multi-swap for L-Sided Pricing as following:

Theorem 5. *For any $C \geq 1$ and $\epsilon > 0$, there is an instance $\Phi = (G, \mu, b)$ of* L-Sided Pricing *where all $u \in L$ have capacity C and all $v \in R$ have capacity 1 such that the locality gap of Φ is at least $2 - \epsilon$ with respect to the single-swap heuristic.*

Theorem 6. *For all $C \geq 2, \rho \geq 1$ and $\epsilon > 0$, there is an instance $\Phi = (G, \mu, b)$ of* L-Sided Pricing *where all $u \in L$ have capacity C and all $v \in R$ have capacity 1 such that the locality gap Φ is at least $\frac{2C-1}{C} - \epsilon$ with respect to the simple ρ-swap algorithm.*

Also we show some **APX**-hardness results for L-Sided Pricing as following:

Theorem 7. L-Sided Pricing *is* **APX***-hard, even if all capacities are at most 4 and all customers have a budget of 1 or 2.*

At the end of Chapter 4 we will provide an instance of L-Sided Pricing for which the LP solution is slightly better than the optimal integral solution which gives us a lower-bound on the integrality gap.

Chapter 2

Graph Pricing with Limited Supply

2.1 Problem Overview

We consider multigraphs that may have parallel edges and loops, unless we explicitly specify that we are working with simple graphs. For a set of nodes S in a graph $G = (V, E)$, we let $N(S)$ denote all nodes not in S that are neighbours of some node in S. For $u \in V$ we let $\delta_G(u)$ be all edges having u as an endpoint. Often the subscript G is omitted when it is clear from the context. For a subset of edges B, we let $\delta_B(u) = \delta(u) \cap B$, again when the graph G is clear from the context.

We refer to an edge e by uv where u, v are the endpoints of e. For brevity, we may use notation like $e = uv \in E$ when we want to consider an edge $e \in E$ but also want to name the endpoints u, v of e as well. The reason for using this notation rather than simply saying $uv \in E$ is that our local search algorithms do work for graphs with parallel edges (i.e. customers interested in identical bundles), so e would be one particular customer and u, v would name the items that e is interested in.

Given a function $f : T \to \mathbb{R}$ on some finite set T, for any $S \subseteq T$ we let $f(S)$ denote $\sum_{x \in S} f(x)$. Similarly, if $p : V \to \mathbb{R}_{\geq 0}$ is a pricing of the

vertices of a graph $G = (V, E)$, for an edge $e = uv \in E$ we let $p(e)$ denote $p(u) + p(v)$. For two pricings $p, p' : V \to \mathbb{R}_{\geq 0}$ of the nodes of a graph, we let $\texttt{HW}(p, p') = |\{v \in V : p(v) \neq p'(v)\}|$.

Finally, consider an instance $G = (L \cup R, E)$ of L-Sided Pricing where edges have budgets b_e and vertices have capacities μ_v. For any pricing p of the vertices, let $\texttt{val}(p) = \max_{\substack{F \subseteq E \\ (p,F) \text{ feasible}}} \sum_{e \in F} p(e)$ be the maximum profit of a feasible solution with prices p. Note that $\texttt{val}(p)$ can be computed in polynomial time as it is merely asking for a maximum-weight μ-matching solution using only edges $e = uv$ with $p(e) \leq b_e$ (the weight of such an edge being $p(e)$) [31].

2.1.1 Reduction to L-Sided Pricing Problem

To begin, we use a reduction by Balcan and Blum [4] which was stated originally only for the uncapacitated case.

Lemma 3 (Balcan and Blum [4])**.** *If there is an α-approximation for* L-Sided Pricing *with unlimited supply, then there is a 4α-approximation for* Graph Pricing *with unlimited supply.*

We modify their proof to work for the limited capacity version:

Lemma 4. *If there is an α-approximation for* L-Sided Pricing *then there is a 4α-approximation for* Capacitated Graph Pricing.

Proof. Consider an optimal price-vector p^*, where each vertex v is assigned the price of p^*_v. Define $opt(e)$ to be the amount of profit that **OPT** makes from edge e. If e has only one endpoint v, $opt(e)$ could be either 0 or p^*_v, but if $e = uv$ has two different endpoints $v \neq u$, then $opt(e)$ is either 0 or $p^*_v + p^*_u$. We will think of $opt(e)$ as the *weight* of edge e, though it is unknown to our algorithm. Let E_2 be the set of edges that have two distinct endpoints, and let

E_1 be the set of self-loops. Let OPT_1 be the profit made by p^* on edges in E_1 and let OPT_2 be the profit made by p^* on edges in E_2 , so $\sum_{e \in E_i} opt(e) = OPT_i$ for $i = 1, 2$ and $OPT_1 + OPT_2 = OPT$.

Now, *randomly* partition the vertices into two sets L and R, by putting each vertex in L or R with probability equal to $\frac{1}{2}$. Since each edge $e \in E_2$ has a 50% chance of having its endpoints on different sides, in expectation $\frac{1}{2}OPT_2$ weight is on edges with one endpoint in L and one endpoint in R. Thus, if we simply ignore edges in E_2 whose endpoints are on the same side the profit we lose in expectation is no more than $\frac{1}{2}OPT_2$. Now suppose we set the price of all the vertices in L/R to zero, while gaining the profit from the same set of edges. Suppose this would give us the profit OPT_L/OPT_R:

$$\frac{OPT_2}{2} + OPT_1 = \mathbf{E}[OPT_L + OPT_R]$$

This means that if we take the better of the two choices we would have at least a quarter of the total profit:

$$\mathbf{E}[max(OPT_L, OPT_R)] \geq \frac{1}{2} \cdot \mathbf{E}[OPT_L + OPT_R] = \frac{OPT_2}{4} + \frac{OPT_1}{2} \geq \frac{OPT}{4}$$

This can be efficiently derandomized because we only require pairwise independence of the events $u \in L$ for various $u \in V$, see [30] for details behind this technique. This proves the desired result. □

2.1.2 Our Results

Based on Theorem 1, there is a polynomial-time 2-approximation for L-Sided Pricing. This 2-approximation is fairly simple to obtain using local search. But we think it nicely highlights a direction for designing approximations for pricing+packing problems. To expand on the potential for this technique, we consider a much more involved algorithm for L-Sided Pricing with bounded capacities.

Also, as we saw in Theorem 2, for any constants $C \geq 2, \epsilon > 0$, there is a polynomial-time $\left(\frac{2C-1}{C} + \epsilon\right)$-approximation for L-Sided Pricing if $\mu_v \leq C$ for all $v \in L$. Note that this does not require any bounds on capacities for nodes in R. For example with $C = 2$ this yields a $(1.5+\epsilon)$-approximation and in the case we prove is **APX**-hard (in Chapter 4), where $C = 4$, this yields a $(1.75+\epsilon)$-approximation. Observe if $C = 1$ then both Capacitated Graph Pricing and L-Sided Pricing reduce to maximum-weight matching because we can easily set prices to match the full budget of all edges in any matching.

Theorems 1 and 2 are proven using *local search* algorithms. That is, if we are given prices $p : L \to \mathbb{R}_{\geq 0}$ then the optimal customers $F \subseteq E$ can be computed using a maximum-weight μ-matching algorithm. The local search algorithm for Theorem 1 iteratively tries to change the price of one item in L to see if it yields a better matching. We prove with a simple argument that a local optimum is a 2-approximate solution for L-Sided Pricing.

To prove Theorem 2, we consider a local search algorithm that changes $O(1)$ prices at a time in each step. To analyze the performance of such an algorithm, we need a result about covering directed graphs by directed balls in a uniform way.

Let $H = (L, F)$ be a directed graph. For any $u \in L$ and $r \geq 0$ consider the "directed ball" $B^+(u, r) = \{v \in L : d_H(u, v) \leq r\}$ of nodes in L reachable from u in at most r steps. Similarly, let $\partial B^+(u, r) = \{v \in L : d(u, v) = r\}$ be nodes v such that the shortest $u - v$ path in H has length exactly r (the *boundary* of $B^+(u, r)$). We prove the following covering result for directed graphs.

Theorem 8. *Let $H = (L, F)$ be a directed graph where the indegree of each node is at most C and let $d \in \mathbb{Z}_{\geq 0}$. There is a "weighting" of directed balls $\tau : L \times \{0, 1, \ldots, d\} \to \mathbb{Z}_{\geq 0}$ with the following properties:*

- *For any* $v \in V$, $\displaystyle\sum_{\substack{u \in L, 0 \le r \le d \\ s.t.\ v \in B^+(u,r)}} \tau(u,r) = \sum_{i=0}^{d} C^i = \frac{C^{d+1}-1}{C-1}$.
- *For any* $v \in V$, $\displaystyle\sum_{\substack{u \in L, 0 \le r \le d \\ s.t.\ v \in \partial B^+(u,r)}} \tau(u,r) = C^d$.

Furthermore, $\tau(u,r) \le C^{d-r}$ *for each* $u \in V$ *and* $0 \le r \le d$.

That is, each $v \in L$ lies in these balls with total weight precisely $\frac{C^{d+1}-1}{C-1}$ and appears on the boundary of the balls with weight precisely C^d. The bound on $\tau(u,r)$ at the end of the statement is required to ensure the local search algorithm used to prove Theorem 2 runs in polynomial time.

We also show the analysis of both algorithms are tight. See Section 2.2 for definitions of the two local search algorithms mentioned in the results below and Section 4 for precise statements of how the analysis is tight.

2.2 Local Search Algorithm

We consider local-search algorithms for L-Sided Pricing. Recall we are given a bipartite graph $G = (L \cup R, E)$ where each $v \in L \cup R$ has a capacity $\mu_v \ge 0$, each $e \in E$ has a budget b_e, and we are restricted to setting $p(v) = 0$ for each $v \in R$.

It is clear that there is an optimal solution p such that for each $u \in L$ we have $p(u) = b_e$ for some $e \in \delta(u)$. Otherwise we could increase $p(u)$ to the next budget of an edge touching u (or decrease, if $p(u)$ exceeds all budgets of edges touching u) while not decreasing the value of the solution. So, for $u \in L$ we define $P_u = \{b_e : e \in \delta(u)\}$ to be the set of budgets of customers interested in item u.

We run a local-search approximation based on this observation. Here, a vector p over L is a **pricing** if $p(u) \in P_u$ for each $u \in L$. The local-search algorithm iteratively tries to improve a pricing by changing the price

of only one vertex until no such improvement is possible. The full algorithm is presented in Algorithm 1. Because a price $p(u)$ is chosen from P_u for each $u \in L$, it is clear that an iteration can be executed in polynomial time.

Algorithm 1 Single-Swap Algorithm for L-Sided Pricing.

```
let p be any pricing
while val(p') > val(p) for some pricing p' with HW(p, p') = 1 do
  p ← p'
return p
```

Call a pricing p *locally optimal* if it cannot be improved by changing the price for any $u \in L$, note Algorithm 1 returns a locally-optimal pricing. As is common in local search, we analyze the quality of a locally-optimal solution. In the next subsection we show $\texttt{val}(p) \geq \texttt{val}(p^*)/2$ where p^* is an optimal pricing for the L-Sided Pricing instance.

The main concern is then the efficiency of the algorithm. Clearly each iteration can be executed in polynomial time but the number of iterations is not apparently bounded. We use a more recent observation from [17] to find a solution which may not be a local optimum but is still guaranteed to have value at least 1/2 of the optimum value with no ϵ-loss in the guarantee, unlike in older standard tricks where an ϵ-loss is necessary to have a polynomial running time (See [2] for a specific example of this approach). A simple application of this trick is discussed in Section 2.2.4.

2.2.1 Single Swap Analysis

We fix p^* to be some particular optimal pricing.

Theorem 9. *For any locally-optimal pricing p, $\texttt{val}(p) \geq \texttt{val}(p^*)/2$.*

Proof. Let $B \subseteq E$ be the edges that are bought in the local optimum solution, and $B^* \subseteq E$ the edges that are bought in the global optimum solution. Thus, $\texttt{val}(p) = \sum_{u \in L} p(u) \cdot |\delta_B(u)|$ and $p(e) \leq b_e$ for each $e \in \delta_B(u)$.

For each $u \in L$, consider the local search step that changes the price of u from $p(u)$ to $p^*(u)$. That is, consider p^u where $p^u(u) = p^*(u)$ and $p^u(u') = p(u')$ for $u' \in L - \{u\}$. We refer to this swap as the $p \to p^u$ swap. For brevity, let $\Delta_u := \texttt{val}(p^u) - \texttt{val}(p)$ and note $\Delta_u \leq 0$ because p is a local optimum. We provide a lower bound on Δ_u in a way that relates part of the global optimum with part of the local optimum.

First, construct a subset $B' \subseteq B^*$ and an injective mapping $\sigma : B' \to B$ iteratively as follows in Algorithm 2. Intuitively, it greedily pairs edges in B^* with edges in B sharing the same endpoint in R until no more pairs can be made. After this pairing, for each $v \in R$ we either have $\delta_{B^*}(v) \subseteq B'$ or $\delta_B(v) \subseteq \sigma(B')$ (or both).

Algorithm 2 Constructing B' and σ.

$B' := \emptyset$
for each $e^* = uv \in B^*$ where $v \in R$ **do**
  if there is some $e \in \delta_B(v)$ such that no $e' \in B^*$ has $\sigma(e') = e$ **then**
    set $B' := B' \cup \{e^*\}$ and $\sigma(e^*) := e$

Now we bound Δ_u. One possible matching with the modified prices p^u is

$$B^u := B \cup \delta_{B^*}(u) - \delta_B(u) - \{\sigma(e) : e \in \delta_{B'}(u)\}.$$

To show this is feasible, note than no vertex capacities are violated and each edge $e \in B^u$ has $p^u(e) \leq b_e$. That is, it alters B by swapping $\delta_B(u)$ for $\delta_{B^*}(u)$ and removes edges paired, via σ, with $\delta_{B^*}(u)$ to make room across nodes in R for these new edges. It could be that some edges in $\delta_{B^*}(u)$ are not paired by σ but this indicates their right-endpoints already have enough room to accommodate these edges without removing other edges from B. So, B^u respects the vertex capacities.

Now, Δ_u represents the cost change when using the maximum value matching with the new profits. This can be bounded as follows, based on the fact

that B^u is a feasible solution under prices p^u:

$$0 \geq \Delta_u \geq p^*(u) \cdot |\delta_{B^*}(u)| - p(u) \cdot |\delta_B(u)| - \sum_{e' \in \delta_{B'}(u)} p(\sigma(e')).$$

Summing over all $u \in L$ and noting each $e \in B$ has its corresponding term appearing in the last sum for at most one $u \in L$ because σ' is one-to-one shows $0 \geq \texttt{val}(p^*) - 2 \cdot \texttt{val}(p)$. □

2.2.2 An Improved Multi-Swap Algorithm for Bounded Capacities

Here we consider the restriction of L-Sided Pricing to instances where $\mu_u \leq C$ for each $u \in L$ for some fixed constant $C \geq 2$. Note we do not require capacities of $v \in R$ to be bounded by C.

Let $d \geq 1$ be a fixed integer: larger d will result in better approximation guarantees with a slower, but still polynomial-time, algorithm. The multi-swap algorithm we consider is given in Algorithm 3. Let $\rho = 1 + C + C^2 + \ldots + C^d = \frac{C^{d+1}-1}{C-1}$. An iteration runs in polynomial time because ρ is a constant.

Algorithm 3 Multi-Swap Algorithm For L-Sided Pricing.

```
let p be any pricing
while there is a pricing p' with HW(p, p') ≤ ρ and val(p') > val(p) do
  p ← p'
return p
```

As before, call a pricing p *locally optimal* if $\texttt{val}(p') \leq \texttt{val}(p)$ for any pricing p' with $\texttt{HW}(p, p') \leq \rho$. Recall P_u for $u \in L$ is the set of distinct budgets of the edges incident to u and that, in L-Sided Pricing, we can assume any pricing p has $p(u) \in P_u$ for all $u \in L$. So, as C and d are constants, a single iteration can be executed in polynomial time by trying all subsets $S \subseteq L$ of bounded size and, for each of those, trying all $\prod_{u \in S}(|P_u| - 1) \leq |E|^{O(1)}$ ways to change the prices of all $u \in S$. We prove the following approximation guarantee.

Theorem 10. *Let p be a locally-optimal solution and p^* a global optimum solution. Then $\texttt{val}(p) \geq \frac{C-C^{-d}}{2C-1-C^{-d}} \cdot \texttt{val}(p^*)$.*

So for any fixed $C \geq 2$ and $\epsilon > 0$, and large enough d we see there is a $\left(\frac{C}{2C-1} - \epsilon\right)$-approximation for instances of L-Sided Pricing where all capacities of nodes in L are bounded by C. We can again use the same trick from Section 2.2.4 to ensure the number of iterations is polynomially-bounded.

We will soon prove Theorem 8 stated in Section 1.2. For now, we show how to complete the local search analysis using this result. Let p^* denote an optimal pricing, $B \subseteq E$ the edges bought in the local optimum p, and $B^* \subseteq E$ the edges bought under p^*. Let $\sigma : B' \to B$ be a pairing constructed in the same way as in the single swap analysis (using Algorithm 2) where $B' \subseteq B^*$.

To describe the swaps used in the analysis, first consider the following auxiliary directed graph $H = (L, F)$ whose nodes are the same as the left-side of this L-Sided Pricing instance and whose edges are given as follows. For any $e^* = uv \in B'$, let $w \in L$ be the left-endpoint of $\sigma(e^*)$. Add a directed edge from u to w in F.

Observe that both the indegree and outdegree of a vertex in H is at most C by this construction, so Theorem 8 applies. Let $\tau : L \times \{0, 1, \ldots, d\} \to \mathbb{Z}_{\geq 0}$ be the given weighting of directed balls in H. These weights will be used to combine inequalities generated by the test swaps below.

Test Swaps

For any $u \in L$ and any $0 \leq i \leq d$, consider the prices $p^{u,i}$ defined by

$$p^{u,i}(v) = \begin{cases} p^*(v) & \text{if } d_H(u, v) \leq i \\ p(v) & \text{otherwise} \end{cases}$$

Note $\texttt{HW}(p, p^{u,i}) = |B^+(u, i)| \leq C^0 + C^1 + \ldots + C^i \leq \rho$ because the outdegree of each vertex is at most C, so $p \to p^{u,i}$ is a valid test swap. Let $\Delta_{u,i} = \texttt{val}(p^{u,i}) - \texttt{val}(p)$ and note $\Delta_{u,i} \leq 0$ by local optimality. We bound the

difference by explicitly describing a feasible set of edges $B^{u,i}$, namely:

$$B^{u,i} = B \cup \delta_{B^*}(B^+(u,i)) - \delta_B(B^+(u,i)) - \sigma(\delta_{B'}(\partial B^+(u,i))).$$

That is, add all edges from B^* touching a vertex in the directed ball $B^+(u,i)$ and remove all edges from B that either touch $B^+(u,i)$ or are paired (via σ) with an edge in B' that touches $\partial B^+(u,i)$. It is again easy to check that $(p^{u,i}, B^{u,i})$ is a feasible solution: across $u \in L$ we simply exchanged edges in B touching U for edges in B^* touching u and we ensured any new $e^* \in B'$ has $\sigma(e^*)$ removed to make room for e^* across its right-endpoint. Observe for any $e^* \in \delta_{B'}(B^+(u,i-1))$ that $\sigma(e^*)$ is already removed when $\delta_B(B^+(u,i))$ is removed from B, which is why the last part of the definition of $B^{u,i}$ only uses the boundary $\partial B^+(u,i)$ instead of all of $B^+(u,i)$ to remove the remaining edges of B that are paired with $\delta_{B'}(B^+(u,i))$.

Weighting the inequalities by $\tau(u,i)$,

$$\begin{aligned} 0 &\geq \tau(u,i) \cdot \Delta_{u,i} \geq \tau(u,i) \cdot \left(\sum_{e \in B^{u,i}} p^{u,i}(e) - \sum_{e \in B} p(e) \right) \\ &= \tau(u,i) \cdot \sum_{e \in B^* \cap B^{u,i}} p^*(e) - \tau(u,i) \cdot \sum_{e \in B - B^{u,i}} p(e). \end{aligned} \tag{2.1}$$

It remains to consider the contribution of each edge in B^* and B to this bound if we sum over all $u \in L, 0 \leq i \leq d$. Observe an edge $e = vw \in B^*$ is "swapped in" in this analysis for the swap $p \to p^{u,i}$ if and only if $v \in B^+(u,i)$. So by Theorem 8, the total contribution of $p^*(e)$ to $\sum_{u,i} \tau(u,i) \cdot \Delta_{u,i}$ is precisely $\frac{C^{d+1}-1}{C-1}$.

On the other hand, an edge $e = vw \in B$ is "swapped out" in this analysis for the swap $p \to p^{u,i}$ if and only if $v \in B^+(u,i)$ or $\sigma^{-1}(e) \in \partial B^+(u,i)$ (if e is indeed paired by σ). Again by Theorem 8, the total τ-weight of the first event is exactly $\frac{C^{d+1}-1}{C-1}$ and, if $\sigma^{-1}(e)$ is defined, the total τ-weight of the second

event is exactly C^d. Thus,

$$0 \geq \sum_{\substack{u \in L \\ 0 \leq i \leq d}} \tau(u,i) \cdot \Delta(u,i) \geq \frac{C^{d+1}-1}{C-1} \cdot \texttt{val}(p^*) - \left(\frac{C^{d+1}-1}{C-1} + C^d \right) \cdot \texttt{val}(p),$$

which proves Theorem 10.

2.2.3 Proof of Theorem 8

Before presenting the full proof to conclude the analysis, we consider a simpler setting to develop intuition. Suppose, for each $0 \leq i \leq d$ and each $u \in L$ there are precisely C^i nodes $w \in L$ with $d_H(w,u) = i$. This would happen if, say, H has indegree and outdegree exactly C at each vertex and the undirected version of H has girth $> 2d$. Then setting $\tau(u,i) = 1$ if $i = d$ and 0 otherwise for each $u \in L$ would suffice.

In the general setting without this assumption, we have to consider other directed balls $B^+(u,i)$ for different $0 \leq i \leq d$ and with, perhaps, larger weights than 1. This is because the radius-d balls $B^+(u,d)$ themselves for various $u \in L$ do not cover each $v \in L$ precisely $\sum_{i=0}^{d} C^i$ times.

Inductively define $\tau(u,i)$ for $u \in L$ and $0 \leq i \leq d$ as follows:

$$\tau(u,i) = \begin{cases} 1 & \text{if } i = d, \\ C^{d-i} - \displaystyle\sum_{j=i+1}^{d} \sum_{\substack{v \in L \\ d_H(v,u) = j-i}} \tau(v,j) & \text{otherwise.} \end{cases}$$

The inspiration behind this construction is that in general we would have $d_H(u,v) = i$ for only *at most* C^i nodes u. So we consider smaller directed balls to make up this deficiency. If we think that the distance i requirement for each $v \in V$ is exactly C^i, then for each $u \in L$ the ball $B^+(u,j)$ contributes to the distance $d - j + d_H(u,v)$ requirement for each $v \in B^+(u,j)$. See Figure 2.1 for an illustration.

The recurrence above ensures the total contribution to the distance i requirement for each v by all all directed balls is exactly C^i. We formalize this idea and show the τ values are nonnegative in Lemma 5 below.

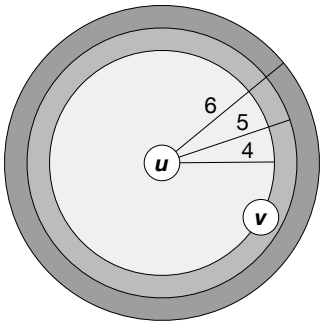

Figure 2.1: Illustration of the case $d = 6$ where $d_H(u, v) = 4$. The directed ball $B^+(u, 6)$ contributes to the "distance 4" requirement for v, $B^+(u, 5)$ contributes to the "distance 5" requirement for v, and $B^+(u, 4)$ contributes to the "distance 6" requirement for v.

Lemma 5. *For each* $u \in L, 0 \leq i \leq d$ *we have* $\sum_{j=i}^{d} \sum_{\substack{v \in L \\ d_H(v,u)=j-i}} \tau(v, j) = C^{d-i}$ *and* $0 \leq \tau(u, i) \leq C^{d-i}$.

Proof. The equality is by construction and the observation that $d_H(v, u) = 0$ if and only if $v = u$. The inequalities are proven inductively with the base case $i = d$ being given. Now suppose for $i < d$ we know $0 \leq \tau(u, j) \leq C^{d-j}$ for any $i < j \leq d$ and any $u \in L$. By the recurrence for $\tau(u, i)$ and because $\tau(v, j) \geq 0$ for any $i < j \leq d$ and $v \in V$, we see $\tau(u, i) \leq C^{d-i}$. Next, we prove $\tau(u, i) \geq 0$ for each $u \in L$.

For any $i < j \leq d$ and any $v \in L$ with $d_H(v, u) = j - i$, there is some $w \in L$ such that $d_H(v, w) = i - j - 1$ and $d_H(w, u) = 1$. That is, consider a shortest $v - u$ path P in H, as $i < j$, we have $v \neq u$ so the second-last node on this path is a node w whose distance to u is 1 (it could be $w = v$, if $j - i = 1$).

From this and using the equality from the first part of the theorem statement, we bound the double sum in the recurrence defining $\tau(u,i)$ by

$$\begin{aligned}\sum_{j=i+1}^{d} \sum_{\substack{v \in L \\ d_H(v,u)=j-i}} \tau(v,j) &\leq \sum_{w:d_H(w,u)=1} \sum_{j=i+1}^{d} \sum_{\substack{v \in L \\ d_H(v,w)=j-(i+1)}} \tau(v,j) \\ &= \sum_{w:d_H(w,u)=1} C^{d-(i+1)} \\ &\leq C^{d-i}.\end{aligned}$$

The last bound follows as each $v \in L$ has indegree at most C in H. Thus, from the recurrence again, we see $\tau(u,i) \geq 0$. □

Lemma 5 finishes the proof of Theorem 8 as follows. The first bullet point in Theorem 8 follows by summing over all $0 \leq i \leq d$. The second point follows by fixing $i = 0$.

2.2.4 Efficient Versions of Local Search

The standard trick to make local search algorithms efficient is to only make an improvement if it is somewhat noticeable. That is, a swap is performed only if it improves the cost by a factor of at least $1 + \epsilon/\Delta$ where Δ is the total "weight" of all inequalities generated by test swaps to complete the analysis (typically, Δ is polynomial in the input size). See [2] for a specific example of this approach.

However, such analysis typically "loses an ϵ" in the approximation guarantee. We adapt an alternative approach outlined in [17] that avoids this ϵ-loss while still achieving the same approximation guarantee that a true local optimum is proven to have. We consider the single-swap algorithm first, the extension to the multi-swap algorithm is in Section 2.2.5.

Recall that the proof of Theorem 9 described a set of *test swaps* and placed a bound on the cost change. That is, for each $u \in L$ the swap $p \to p^u$ is

considered and a bound Δ_u on the change in $\texttt{val}()$ was given as

$$\Delta_u \geq p^*(u) \cdot |\delta_{B^*}(u)| - p(u) \cdot |\delta_B(u)| - \sum_{e \in \delta_{B'}(u)} p(\sigma(e)).$$

Observe this bound holds even if p is not a local optimum solution. The only place in the proof of Theorem 9 that used the fact that p was a local optimum was in asserting $0 \geq \Delta_u$, which is not required here.

Summing the above over all $u \in L$ shows

$$\sum_{u \in L} \Delta_u \geq \texttt{val}(p^*) - 2 \cdot \texttt{val}(p).$$

Thus, the $u \in L$ with largest Δ_u satisfies

$$\Delta_u \geq \frac{\texttt{val}(p^*) - 2 \cdot \texttt{val}(p)}{|L|}.$$

So if we take the best improvement in each step of the algorithm, the next price p' then satisfies

$$\texttt{val}(p') \geq \texttt{val}(p) + \frac{\texttt{val}(p^*) - 2 \cdot \texttt{val}(p)}{|L|}.$$

Consider the potential function $\Phi(p) := \texttt{val}(p^*) - 2 \cdot \texttt{val}(p)$. If $\Phi(p) > 0$, then $\Phi(p') \leq \left(1 - \frac{2}{|L|}\right) \cdot \Phi(p)$ follows from the expression above. That is, $\Phi(p)$ decreases by a factor of $\exp(-1)$ after every $|L|/2$ iterations as long as the current price p satisfies $\Phi(p) > 0$.

With the standard assumption that the budgets b_e are expressed as rational numbers in the input, there is a big integer $\mathcal{M}$, whose bit complexity is polynomial in the total bit complexity of the input, such that $\Phi(p)$ could not be less than $\frac{1}{\mathcal{M}}$, unless it is zero. This means that after a polynomial number of iterations, we will reach a solution p with $\Phi(p) \leq 0$, *i.e.* $\texttt{val}(p) \geq \texttt{val}(p^*)/2$ as required, provided we take the best improvement in each step.

2.2.5 Extension to Multi-Swap

Each swap of the form $p \to p^{u,r}$ for $0 \leq i \leq d$ and $u \in L$ in the analysis was weighted with a value $0 \leq \tau(u,r) \leq C^{d-r}$. Let $\kappa = \frac{C^{d+1}-1}{C-1} \cdot |L|$, so κ is an upper bound on the total weight of all test swaps and $\kappa = O(|L|)$ as C and d are constants.

Again, even if p is not a local optimum our analysis still shows

$$\sum_{u \in L, 0 \leq r \leq d} \tau(u,r) \cdot (\texttt{val}(p^{u,r}) - \texttt{val}(p)) \geq \frac{C^{d+1}-1}{C-1} \cdot \texttt{val}(p^*) - \left(\frac{C^{d+1}-1}{C-1} + C^d \right) \cdot \texttt{val}(p).$$

Local optimality of p was only used to show the left-hand side was not positive. Without local optimality, we may still conclude the most improving swap $p \to p'$ satisfies

$$\texttt{val}(p') \geq \texttt{val}(p) + \frac{1}{\kappa} \cdot \left(\frac{C^{d+1}-1}{C-1} \cdot \texttt{val}(p^*) - \left(\frac{C^{d+1}-1}{C-1} + C^d \right) \cdot \texttt{val}(p) \right).$$

Consider the potential function

$$\phi(p) = \frac{C^{d+1}-1}{C-1} \cdot \texttt{val}(p^*) - \left(\frac{C^{d+1}-1}{C-1} + C^d \right) \cdot \texttt{val}(p).$$

The above bound shows if $\phi(p) > 0$ then choosing the best improving swaps will result in a solution p' with $\phi(p') \leq \left(1 - \left(\frac{C^{d+1}-1}{C-1} + C^d \right) \cdot \frac{1}{\kappa} \right) \cdot \phi(p)$. So $\phi(p)$ decreases geometrically every $O(\kappa)$ iterations as long as it remains positive. As $\kappa = O(|L|)$ and by using rationality of the input values, the potential $\phi(p)$ will become nonpositive after a polynomial number of iterations in the total bit complexity of the input as long as we take the most improving swap.

Chapter 3

Linear Programming Based Algorithms

3.1 Our Results

So far, our focus has been on approximations based on local search. Here, we consider linear programming relaxations for L-Sided Pricing. Recall for each $u \in L$ that $P_u = \{b_e : e \in \delta(u)\}$ is a set of possible prices for vertex u: there is an optimal solution that selects $p(u)$ from P_u for each $u \in L$.

For $u \in L$ and $p \in P_u$, we let $y_{u,p}$ be a variable indicating we select price p for u. Similarly, for each $e = uv \in E$ and $p \in P_u$ we let $x_{e,p}$ be a variable indicating edge e is selected and vertex u is assigned price p (so e buys their bundle at price p). The following relaxation provides an upper bound on the optimal solution to the given instance of the L-Sided Pricing.

$$\textbf{maximize} \quad \sum_{e=uv} \sum_{p \in P_u} p \cdot x_{e,p} \qquad \textbf{(LP-Pricing)}$$

$$\textbf{subject to} \quad \sum_{p \in P_u} y_{u,p} = 1 \qquad \forall\, u \in L \tag{3.1}$$

$$\sum_{e \in \delta(u)} x_{e,p} \leq y_{u,p} \cdot \mu_u \quad \forall\, u \in L, p \in P_u \tag{3.2}$$

$$\sum_{e=uv \in \delta(v)} \sum_{p \in P_u} x_{e,p} \leq \mu_v \qquad \forall\, v \in R \tag{3.3}$$

$$x_{e,p} \leq y_{u,p} \qquad \forall\, u \in L, e \in \delta(u), p \in P_u \text{ s.t. } p \leq b_e \tag{3.4}$$

$$x_{e,p} = 0 \qquad \forall e = uv, p \in P_u \text{ s.t. } p > b_e \tag{3.5}$$

$$x, y \geq 0$$

Constraints (3.1) indicate one price must be selected for each $u \in L$, (3.2) ensures the capacity constraints for $u \in L$ are satisfied and (3.3) ensures the capacity constraints for $v \in R$ are satisfied, (3.4) ensures we must set the price of u to p if we are to have e pay p, and (3.5) ensures a chosen edge does not pay more than it can afford.

3.1.1 Randomized Rounding Algorithms

In this section we are going to prove Theorem 3 by first showing that in simple graphs with large capacities for nodes in R, the integrality gap is close to 1. This will help us to use the reduction described in Lemma 4 and provide a $(4+\epsilon)$-approximation for instances of Capacitated Graph Pricing in simple graphs satisfying $\mu_v \geq C_\epsilon$ which completes the proof of Theorem 3. More specifically we prove the following:

Theorem 11. *For any $\epsilon > 0$, the integrality gap of* (**LP-Pricing**) *is $1 - 2\epsilon$ in simple graphs when $\mu_v \geq 3\ln(1/\epsilon)/\epsilon^2 + 1$ for all $v \in R$.*

Proof. Consider the following randomized rounding algorithm. For each $u \in L$, sample a price $p'(u) \in P_u$ from the distribution with $\mathbf{Pr}[p'(u) = p] = y_{u,p}$. This is a distribution by (3.1) and non-negativity of y. For brevity, we will let $p'(e) = p'(u)$ for an edge $e = uv$.

Then define a fractional matching in G as follows. The idea is that we want to assign a value of $x_{e,p'(e)}/y_{u,p'(e)}$ to each edge (using 0 if $y_{u,p'(e)} = 0$), this is at most 1 by (3.4). By (3.2) this fractional matching would always satisfy the capacity constraints for nodes in L. But it may violate constraints for nodes in R. The obvious solution would be to scale each of these fractional values to be a feasible matching satisfying all vertex constraints. We take a simpler view which is sufficient for our purposes, we scale all resulting values by $1 - \epsilon$, and then outright discard edges $e = uv$ where the capacity of v is still violated after this scaling.

More precisely, for each $e = uv$ we first let $x''_e = (1-\epsilon) \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}$ (again using 0 if $y_{u,p'(e)} = 0$). Then for each edge $e = uv$, we define

$$x'_e = \begin{cases} x''_e & \text{if } \sum_{e'=u'v \in \delta(v)} x''_e \leq \mu_v, \\ 0 & \text{otherwise.} \end{cases}$$

Now $x'(\delta(w)) \leq \mu_w$ for each $w \in L \cup R$. Also, (3.5) ensures any $e \in E$ with $x'_e > 0$ has $p'(e) \leq b_u$. So, considering the fact that the bipartite μ-matching polytope is unimodular [33], there would be an integral matching p' obtaining at least as much value as the fractional matching x': $\texttt{val}(p') \geq \sum_e p \cdot x'_e$. It remains to show that the fractional matching x' has good profit in expectation.

For any $e = uv \in E$ let $\mathcal{B}_e$ be the *bad event* that $\sum_{e'=u'v \in \delta(v), e' \neq e} x''_{e'} \geq \mu_v - 1$. Notice that the second case in the definition of x'_e applies only if event $\mathcal{B}_e$ happens. We show $\mathbf{Pr}[\mathcal{B}_e] \leq \epsilon$. If so, for each $e = uv \in E$ the fact that $\mathcal{B}_e$ is independent of the choice of $p'(e)$ (as G is a simple graph) we then have

$$\mathbf{E}[p'(e) \cdot x'_e] \geq (1 - \mathbf{Pr}[\mathcal{B}_e]) \cdot (1-\epsilon) \cdot \mathbf{E}\left[p'(e) \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}\right] \geq (1-\epsilon)^2 \cdot \sum_{p \in P_u} p \cdot x_{e,p}.$$

Summing over all edges:

$$\mathbf{E}\left[\sum_{e\in E} p'(e)\cdot x'_e\right] \geq (1-2\epsilon)\cdot \sum_{e=uv}\sum_{p\in P_u} p\cdot x_{e,p}$$

To bound $\mathbf{Pr}[\mathcal{B}_e]$, for an edge e' let $X_{e'}$ denote the random variable with value $(1-\epsilon)\cdot x_{e',p'(u)}/y_{u',p'(e')}$ and let $X^e = \sum_{e'\in\delta(v),e'\neq e} X_{e'}$. Then $\mathbf{E}[X_{e'}] = (1-\epsilon)\cdot\sum_{p\in P_{u'}} x_{e',p}$ so by (3.3) we have $\mathbf{E}[X^e] \leq (1-\epsilon)\cdot\mu_v$.

Again by simplicity of G, the random variables $X_{e'}$ are independent for different $e'\in\delta(v), e'\neq e$. Let $Y = (1-\epsilon)\cdot(\mu_v - 1)$. As $Y \geq \mathbf{E}[X^e]$ and $0<\epsilon<1$, by using Lemma 2 (Chernoff Bound) we have

$$\mathbf{Pr}[X^e > (1+\epsilon)\cdot Y] \leq \exp(-Y\epsilon^2/3) \leq \epsilon$$

Finally, since event $\mathcal{B}_e$ implies $X^e \geq \mu_v - 1 \geq (1+\epsilon)\cdot Y$, we have $\mathbf{Pr}[\mathcal{B}_e]\leq\epsilon$, as required. □

The fact that G was simple was used in the application of the Chernoff bound. The random variables $X_{e'}$ for edges in $\delta(v)$ for some $v\in R$ are independent if G is simple.

3.1.2 Extension to k-Hypergraphs

To begin, we use a reduction by Balcan and Blum [4] which was stated originally only for the uncapacitated case.

Lemma 6 (Balcan and Blum [4])**.** *If there is an α-approximation for* L-Sided Pricing *with unlimited supply, then there is a $O(k\cdot\alpha)$-approximation for* Graph Pricing *problem with unlimited supply in k-Hypergraph.*

We modify their proof to work for the limited capacity version:

Lemma 7. *If there is an α-approximation for* L-Sided Pricing *then there is a $O(k\cdot\alpha)$-approximation for k-Hypergraph* Capacitated Graph Pricing *problem.*

Proof. We can use the following procedure.

Step 1: Randomly partition V into V_L and V_{rest} by independently placing each node into V_L with probability $\frac{1}{k}$.

Step 2: Let E' be the set of edges with *exactly* one endpoint in V_L . Ignore all edges in $E \setminus E'$.

Step 3: To analyze this algorithm, let $OPT_{i,e}$ denote the profit made by p^* selling item i to bidder e. (So $OPT_{i,e} \in \{0, p_i^*\}$ where p_i^* is the price of item i in the optimal solution p^* and $OPT = \sum_{i \in V, e \in E} OPT_{i,e}$).

Notice that the total profit made in Step 3 is at least $\sum_{i \in V_L, e \in E'} OPT_{i,e}$ because setting prices in V_{rest} to 0 can only increase the number of sales made by p^* to bidders in E'. Thus, we simply need to analyze the quantity $E[\sum_{i \in V_L, e \in E'} OPT_{i,e}]$

Define indicator random variable $X_{i,e} = 1$ if $i \in V_L$ and $e \in E'$, and $X_{i,e} = 0$ otherwise. We have:

$$\mathbf{E}[X_{i,e}] = \mathbf{Pr}[i \in V_L \text{ and } e \in E'] \geq \frac{1}{k}\left(1 - \frac{1}{k}\right)^{k-1}$$

Therefore,

$$\begin{aligned}
\mathbf{E}\left[\sum_{i \in V_L, e \in E'} OPT_{i,e}\right] &= \mathbf{E}\left[\sum_{i \in V, e \in E} X_{i,e} OPT_{i,e}\right] \\
&= \sum_{i \in V, e \in E} \mathbf{E}[X_{i,e}] OPT_{i,e} \\
&\geq \frac{1}{k}\left(1 - \frac{1}{k}\right)^{k-1} OPT \\
&\geq \frac{OPT}{ke}
\end{aligned}$$

Where e is the base of the natural logarithm. □

Now we are going to prove Theorem 4 which says k-Hypergraph Capacitated Graph Pricing problem admit a randomized, polynomial-time $(4k^2(\frac{k}{k-1})^{k-1})$-approximation.

Proof. We first use the reduction to a k-hypergraph where we are only allowed to use nonzero prices on one part losing a $(k(\frac{k}{k-1})^{k-1}) \leq ke$ factor in the guarantee of lemma 7 and then use a natural rounding of **LP-Pricing** to this setting while losing only an additional $4k$ factor.

Algorithm 4 Randomized Rounding **LP-Pricing** for L-Sided Pricing in k-Hypergraphs.

Solve **LP-Pricing** for L-Sided Pricing
For each $u \in L$ sample price $p'(u) \in P_u$ with $\mathbf{Pr}[p'(u) = p] = y_{u,p}$
For each $e \in E \cap \delta(u)$ where $u \in L$, sample e with probability $\frac{1}{2k} \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}$ (0 if $y_{u,p'(e)} = 0$)
For each $w \in L \cup R$, if edges picked from $\delta(W)$ are more than its capacity drop all of them
return the remaining set of edges

To prove $4k$ additional loss, we use Algorithm 4. More specifically, for each $u \in L$, sample a price $p'(u) \in P_u$ from the distribution with $\mathbf{Pr}[p'(u) = p] = y_{u,p}$. This is a distribution by (3.1) and non-negativity of y. For brevity, we will let $p'(e) = p'(u)$ for an edge $e \in \delta(u)$, where $u \in L$. Then for each edge $e \in E$, where $e \in \delta(u)$ and $u \in L$, set $x''_e = 1$ with probability $\frac{1}{2k} \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}$ (0 if $y_{u,p'(e)} = 0$) and $x''_e = 0$ otherwise. Note that $\frac{x_{e,p'(e)}}{y_{u,p'(e)}}$ is at most 1 by (3.4). More precisely we are picking each edge e into our integral solution x'' with probability $\frac{1}{2k} \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}$ (again zero if $y_{u,p'(e)} = 0$).

But the problem with solution x'' is that it may violate the capacity constraints for vertices on both sides L and R. Our approach is to convert integral solution x'' to x' by going through all the vertices in L and R and if the capacity constraint is violated for some $u \in L$ ($v \in R$), we would remove all the edges connected to u (v). For each edge $e \in E$ we set $x'_e = x''_e$ only if

the capacity constraints of all its k endpoints are not violated in x''. Now $x'(\delta(w)) \leq \mu_w$ for each $w \in L \cup R$. Also, (3.5) ensures any $e \in E$ with $x'_e > 0$ has $p'(e) \leq b_u$. The only remaining thing is to show that it has good profit in expectation.

For any $e \in E$, where $e \in \delta(u)$ and $u \in L$, let $\mathcal{B}_e$ be the *bad event* that there is a vertex $w \in \{u\} \cup R$ where $e \in \delta(w)$ and $\sum_{e' \in \delta(w)} x''_{e'} > \mu_w$. Notice that $x'_e = x''_e$ if event $\mathcal{B}_e$ does not happen and $x'_e = 0$ otherwise. We show $\mathbf{Pr}[\mathcal{B}_e] \leq \frac{1}{2}$. If so, we then have

$$\mathbf{E}[p'(e) \cdot x'_e] \geq (1 - \mathbf{Pr}[\mathcal{B}_e]) \cdot \frac{1}{2k} \cdot \mathbf{E}\left[p'(e) \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}\right] \geq \frac{1}{4k} \cdot \sum_{p \in P_u} p \cdot x_{e,p}.$$

Summing over all edges:

$$\mathbf{E}\left[\sum_{e \in E} p'(e) \cdot x'_e\right] \geq \frac{1}{4k} \cdot \sum_{e=uv} \sum_{p \in P_u} p \cdot x_{e,p}$$

So, it only remains to show $\mathbf{Pr}[\mathcal{B}_e] \leq \frac{1}{2}$ for any $e \in E$. For each vertex $w \in L \cup R$ and each edge $e \in E \cap \delta(w)$, let $\mathcal{B}^w_e$ be the *bad event* that $\sum_{e' \in \delta(w)} x''_{e'} > \mu_w$. It is enough to prove that $\mathbf{Pr}[\mathcal{B}^w_e] \leq \frac{1}{2k}$ for each e and w. Then, because each edge has exactly k endpoints in k-hypergraphs by using a simple union bound $\mathbf{Pr}[\mathcal{B}_e] \leq \frac{1}{2}$.

Now fix a vertex $v \in R$ and edge $e \in E \cap \delta(v)$. For all $e' \in \delta(v)$ let $X_{e'}$ denote the random variable with value $\frac{1}{2k} \cdot x_{e',p'(e')}/y_{u',p'(e')}$, where $u' \in L$ is the only endpoint of e' in L, and let $X^e = \sum_{e' \in \delta(v)} X_{e'}$. Then $\mathbf{E}[X_{e'}] = \frac{1}{2k} \cdot \sum_{p \in P_{u'}} x_{e',p}$ so by (3.3) we have $\mathbf{E}[X^e] \leq \frac{1}{2k} \cdot \mu_v$. Since event $\mathcal{B}^v_e$ implies $X^e > \mu_v \geq 2k \cdot \mathbf{E}[X^e]$, using a simple Markov's inequality we have $\mathbf{Pr}[\mathcal{B}^v_e] \leq \mathbf{Pr}[X^e > 2k \cdot \mathbf{E}[X^e]] \leq \frac{1}{2k}$ as required.

Similarly, fix a vertex $u \in L$ and edge $e \in E \cap \delta(u)$. For all $e' \in \delta(u)$ let $X_{e'}$ denote the random variable with value $\frac{1}{2k} \cdot x_{e',p'(e')}/y_{u,p'(e')}$ and let $X^e = \sum_{e' \in \delta(v)} X_{e'}$. Then $\mathbf{E}[X_{e'}] = \frac{1}{2k} \cdot \sum_{p \in P_u} x_{e',p}$ so by (3.3) we have $\mathbf{E}[X^e] \leq$

$\frac{1}{2k} \cdot \mu_u$. Since event $\mathcal{B}_e^u$ implies $X^e > \mu_u \geq 2k \cdot \mathbf{E}[X^e]$, using Lemma 1 (Markov's inequality) we have:

$$\mathbf{Pr}[\mathcal{B}_e^u] \leq \mathbf{Pr}[X^e > 2k \cdot \mathbf{E}[X^e]] \leq \frac{1}{2k}$$

This means that For each vertex $w \in L \cup R$ and each edge $e \in E \cap \delta(w)$, $\mathbf{Pr}[\mathcal{B}_e^w] \leq \frac{1}{2k}$ which completes the proof.

□

This proof also shows that the integrality gap of the (**LP-Pricing**) would be no more than $4k$. However, for the case of graphs, where $k = 2$, we can have even better:

Lemma 8. *The integrality gap of* (**LP-Pricing**) *is no worse than* $1/4$ *in any instance of* L-Sided Pricing.

Proof. We can change randomized rounding we used for general k and use a rounding algorithm similar to the proof of Theorem 11. For each $u \in L$, sample a price $p'(u) \in P_u$ from the distribution with $\mathbf{Pr}[p'(u) = p] = y_{u,p}$. This is a distribution by (3.1) and non-negativity of y. For brevity, we will let $p'(e) = p'(u)$ for an edge $e = uv$.

Then define a fractional matching in G as follows. The idea is that we want to assign a value of $x_{e,p'(e)}/y_{u,p'(e)}$ to each edge (using 0 if $y_{u,p'(e)} = 0$), this is at most 1 by (3.4). By (3.2) this fractional matching would always satisfy the capacity constraints for nodes in L. But it may violate constraints for nodes in R. We scale all resulting values by $\frac{1}{2}$, and then outright discard edges $e = uv$ where the capacity of v is still violated after this scaling.

More precisely, for each $e = uv$ we first let $x''_e = \frac{1}{2} \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}$ (again using 0 if $y_{u,p'(e)} = 0$). Then for each edge $e = uv$, we define

$$x'_e = \begin{cases} x''_e & \text{if } \sum_{e'=u'v \in \delta(v)} x''_e \leq \mu_v, \\ 0 & \text{otherwise.} \end{cases}$$

Now $x'(\delta(w)) \leq \mu_w$ for each $w \in L \cup R$. Also, (3.5) ensures any $e \in E$ with $x'_e > 0$ has $p'(e) \leq b_u$. So, considering the fact that the bipartite μ-matching polytope is unimodular [33], there would be an integral matching p' obtaining at least as much value as the fractional matching x': $\texttt{val}(p') \geq \sum_e p \cdot x'_e$. It remains to show that the fractional matching x' has good profit in expectation.

For any $e = uv \in E$ let $\mathcal{B}_e$ be the *bad event* that $\sum_{e'=u'v \in \delta(v)} x''_{e'} > \mu_v$. Notice that if event $\mathcal{B}_e$ happens then $x'_e = 0$. We show $\mathbf{Pr}[\mathcal{B}_e] \leq \frac{1}{2}$. If so, for each $e = uv \in E$ the fact that $\mathcal{B}_e$ is independent of the choice of $p'(e)$ (as G is a simple graph) we then have

$$\mathbf{E}[p'(e) \cdot x'_e] \geq (1 - \mathbf{Pr}[\mathcal{B}_e]) \cdot \frac{1}{2} \cdot \mathbf{E}\left[p'(e) \cdot \frac{x_{e,p'(e)}}{y_{u,p'(e)}}\right] \geq \frac{1}{4} \cdot \sum_{p \in P_u} p \cdot x_{e,p}.$$

Summing over all edges:

$$\mathbf{E}\left[\sum_{e \in E} p'(e) \cdot x'_e\right] \geq \frac{1}{4} \cdot \sum_{e=uv} \sum_{p \in P_u} p \cdot x_{e,p}$$

To bound $\mathbf{Pr}[\mathcal{B}_e]$, for an edge e' let $X_{e'}$ denote the random variable with value $\frac{1}{2} \cdot x_{e',p'(u)}/y_{u',p'(e')}$ and let $X^e = \sum_{e' \in \delta(v)} X_{e'}$. Then $\mathbf{E}[X_{e'}] = \frac{1}{2} \cdot \sum_{p \in P_{u'}} x_{e',p}$ so by (3.3) we have $\mathbf{E}[X^e] \leq \frac{1}{2} \cdot \mu_v$.

Let $Y = \frac{1}{2} \cdot \mu_v$. As $Y \geq \mathbf{E}[X^e]$ and $0 < \epsilon < 1$, by using Markov's inequality we have

$$\mathbf{Pr}[X^e > 2 \cdot Y] \leq \mathbf{Pr}[X^e > 2 \cdot \mathbf{E}[X^e]] \leq \frac{1}{2}$$

Finally, since event $\mathcal{B}_e$ implies $X^e \geq \mu_v \geq \frac{1}{2} \cdot Y$, we have $\mathbf{Pr}[\mathcal{B}_e] \leq \frac{1}{2}$, as required. □

Note that this approximation guarantee is even worse than our single-swap algorithm.

Chapter 4

Locality Gaps and Hardness

4.1 Single-Swap

In this section we are going to provide a proof for Theorem 5 which says for any $C \geq 1$ and $\epsilon > 0$, there is an instance $\Phi = (G, \mu, b)$ of L-Sided Pricing where all $u \in L$ have capacity C and all $v \in R$ have capacity 1 such that the locality gap of Φ is at least $2 - \epsilon$ with respect to the single-swap heuristic.

Our single-swap analysis is tight. While this is most striking when $C = 1$, we remark it is still interesting for larger C because it is not obvious, *a priori*, that the single-swap algorithm's analysis cannot be improved as the capacities in L increase.

Proof. For $n \geq 2$, consider the graph $G^{n,C} = (L \cup R, E)$, $L = \{u_i : 1 \leq i \leq n\}$ and $R = \{v_{i,j} : 1 \leq i \leq n, 1 \leq j \leq C\}$. The edges are the union of the edges on the paths $P_j = \{u_1, v_{1,j}, u_2, v_{2,j}, \ldots, u_n, v_{n,j}\}$ for $1 \leq j \leq C$. We use $\mu_u = C$ for $u \in L$ and $\mu_v = 1$ for $v \in R$.

The budgets are given as follows. First let $E_{LOC} = \{u_i v_{i,j} : 1 \leq i \leq n, 1 \leq j \leq C\}$ and $E_{OPT} = \{u_i v_{i-1,j} : 2 \leq i \leq n, 1 \leq j \leq C\}$. All edges in E_{LOC} have a budget of 1 and all edges in E_{OPT} have a budget of 2.

Using $p^*(u) = 2$ for every vertex in L and corresponding edges E_{OPT} is a solution with value of $2C(n-1)$. Now consider the solution with $p(u) = 1$ for

each vertex in L. This solution is just E_{LOC} with a value of Cn, which can be seen to be the optimal matching under these prices because the capacity of every vertex in L is saturated by E_{LOC}. Note $\texttt{val}(p) = \left(\frac{1}{2-2/n}\right) \cdot \texttt{val}(p^*)$. We claim this is a local optimum with respect to the single-swap heuristic.

The only possible swap is to change the price of some u_i to 2. If $i = 1$, this is clearly not an improving swap because no edge incident to u_1 can afford the new price and all other vertices are priced 1 so no matching has value $\geq n-1$. So suppose $i \geq 2$.

The only edges incident to u_i that can afford this new price are $(u_i, v_{i-1,j})$ for all $1 \leq j \leq C$. Furthermore, for any μ-matching B that does not use an edge $e \in \delta_{E_{OPT}}(u_i)$, we can get a better μ-matching (with respect to the new prices) by adding e to B and, if necessary, removing some edge of $E_{LOC} \cap B$ sharing the right-endpoint with e.

Thus, the optimum matching after changing $p(u)$ to 2 uses all of $\delta_{E_{OPT}} \cap B$. After fixing these edges, which have total value $2C$, it is easy to see the best matching we can get in the graph obtained by removing the endpoints of edges $\delta_{E_{OPT}}(u) \cap B$ (plus all edges incident to these endpoints) has value at most $C(n-2)$. So this is not an improving swap. □

4.2 Multi-Swap

The construction for the multi-swap analysis is much more involved than the one for the single-swap case. As a starting point for the construction, we require simple graphs of constant degree but arbitrarily large girth. Such graphs were shown to exist by Sachs [32]. Before presenting the lower bound, we describe a construction of a layered graph with high girth and particular degree bounds. The construction is depicted in Figure 4.1.

Lemma 9. *For any $C \geq 2, \rho \geq 1$ and $t \geq 1$ there is a simple, layered, and*

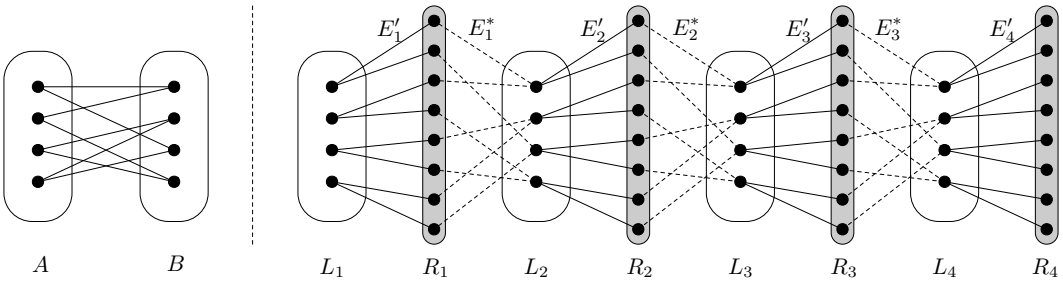

Figure 4.1: An example with $C = 2$. **Left**: the bipartite graph H constructed from H' in the proof of Lemma 9. **Right**: the resulting graph G^t with $t = 4$. Edges of E' are solid and the edges of E^* are dashed.

bipartite graph $G^t = (L \cup R, E)$ with consecutive layers $L_1, R_1, L_2, \ldots, L_t, R_t$ where the subgraph induced by L_i and R_i is a $(C, 1)$-biregular bipartite graph and the subgraph induced by R_i and L_{i+1} is a $(1, C)$-biregular bipartite graph (for each relevant i). Further, G^t has girth exceeding $2t \cdot \rho$.

Note, this implies $|L_i| = |L|/t$ and $|R_i| = |R|/t = C \cdot |L_i|$ for each $1 \leq i \leq t$.

Proof. In [32], it is shown that for any $C', g \geq 3$ there is a simple connected C'-regular graph whose girth (*i.e.* shortest cycle length) is at least g. In our setting, this means a $(2C)$-regular graph with girth exceeding $\rho \cdot t$ exists where ρ, t are as in the statement of Lemma 9. Call this graph $H = (V, F'')$.

As H is $(2C)$-regular it contains an Eulerian circuit. Direct all edges along this circuit so that each vertex of H has indegree C and outdegree C. Finally, build a bipartite graph $H = (A \cup B, F)$ where A and B are disjoint copies of V, and where $u \in A$ and $v \in B$ is an edge of F if uv is a directed edge obtained when we directed the Eulerian circuits.

Build the following layered graph $G^t = (L \cup R, E)$. For each $1 \leq i \leq t$, let L_i be a set of size V and R_i be a set of size $|F|$. Recall that both A and B in H are viewed as copies of V in H', so each L_i can be viewed either as a copy of A or as a copy of B, when appropriate. Now, for each $u \in A$, each $e \in \delta_H(u)$, and each layer $1 \leq i \leq t$, add an edge in G^t from the copy of u in

L_i to the copy of e in R_i. Call the set of all such edges added for a given i E'_i. Similarly, for each $v \in B$, each $e \in \delta_H(u)$, and each layer $1 \leq i \leq t-1$, add an edge in G^t from the copy of e in layer R_i to the copy of v in L_i. Call the set of all such edges added for a given i E^*_i.

Then let $L = \cup_{i=1}^{t} L_i$, $R = \cup_{i=1}^{t} R_i$ and $E = (\bigcup_{i=1}^{t-1} E'_i \cup E^*_i) \cup E'_t$. This construction is depicted in Figure 4.1 in Section 4.

To complete the analysis, consider a simple cycle C in G^t. Note that C alternates between using nodes in L and nodes in R. Furthermore, if C uses nodes consecutive nodes $a \in L_i, b \in R_j, c \in L_k$ (where $|j-i| \leq 1$ and $|k-j| \leq 1$) then the nodes of H corresponding to a and c are connected by an edge in H that corresponds to node b. Thus, the cycle C corresponds to a circuit C' of H^t with $|C'| = |C|/2$. Here, C' may use an edge more than once so $|C'|$ measures the steps taken by the circuit C'.

Consider any node $a \in C \cap L$ and say it is a copy of node v of H. Because the cycle C is simple in G^t, then the two adjacent nodes b, b' to a on C correspond to distinct edges in H incident to v. This is true for every $a \in C \cap L$, so the set of nodes of H corresponding to nodes in $C \cap L$ are incident to at least two distinct edges traversed by C'. That is, the edges used on the circuit C' contain a cycle. As the girth of H is at least $\rho \cdot t$, then $|C| = 2 \cdot |C'| \geq 2 \cdot \rho \cdot t$. $\square$

Now we are going to prove Theorem 6 which says for all $C \geq 2, \rho \geq 1$ and $\epsilon > 0$, there is an instance $\Phi = (G, \mu, b)$ of L-Sided Pricing where all $u \in L$ have capacity C and all $v \in R$ have capacity 1 such that the locality gap Φ is at least $\frac{2C-1}{C} - \epsilon$ with respect to the simple ρ-swap algorithm. That is, our bound on the locality gap for the multi-swap algorithm on instances with bounded capacity is tight.

Proof. Fix $C \geq 2, \rho \geq 1, \epsilon > 0$ and let t be such that $\frac{2C-1}{C} \cdot \frac{t-1}{t} \geq \frac{2C-1}{C} - \epsilon$. Let $G^t = (L \cup R, E)$ be the graph from Lemma 9 for these parameters C, ρ, t. For

each $1 \leq i \leq t$, let E'_i be the edges connecting L_i to R_i and for each $1 \leq i < t$ let E^*_i be the edges connecting R_i to L_{i+1}. Naturally, let $E' = \cup_{i=1}^{t} E'_i$ and $E^* = \cup_{i=1}^{t-1} E^*_i$. See Figure 4.1 for an illustration. Let n be such that $|L_i| = n$ and $|R_i| = C \cdot n$ for all $1 \leq i \leq t$.

The optimum is at least $(2C-1) \cdot C \cdot (|L| - n)$, which can be seen by using edges E^* where each vertex in L has a price of $2C-1$. Now consider the pricing p that uses price C for each vertex in L. The optimum set of edges to buy with these prices is E' with a value of $C \cdot C \cdot |L|$.

Claim 1. *The pricing p is locally optimal with respect to the ρ-swap procedure.*

If so, then the locality gap of this instance is as bad as $\frac{2C-1}{C} \cdot \frac{|L|-n}{|L|} = \frac{2C-1}{C} \cdot \frac{t-1}{t}$, as required.

Proof. Consider any pricing p' with $\texttt{HW}(p, p') \leq \rho$. Let $X \subseteq L$ be the nodes v with $p(v) \neq p'(v)$. So $p'(v) = 2C-1$ for $v \in X$. If some vertex $v \in X$ lies in $L_1 \neq \emptyset$ then no edge incident to v can afford the price $2C-1$, so we may assume that $X \cap L_1 = \emptyset$.

We show $\texttt{val}(p') \leq \texttt{val}(p)$. We first claim any optimal set of edges M^* under this price includes all of $\delta_{E^*}(X)$ and excludes all of $\delta_{E'}(X)$. The latter is simple, no edge in $\delta_{E'}(X)$ can afford the price of its endpoint in L. Then if any $e \in \delta_{E^*}(X)$ is missing from M^*, we can get an even better solution by adding e and removing, if necessary, an edge of $E' \cap M$ sharing its R-endpoint with e. The value increases by at least $(2C-1) - C = C - 1$.

So M^* contains all edges of $\delta_{E^*}(X)$ with value $2C-1$ each plus some edges in $\delta(L - X)$ (which could be in either E' or E^*) with value C each. We then see the value of M^* is

$$(2C-1) \cdot C \cdot |X| + C \cdot |\delta_{M^*}(L - X)|. \tag{4.1}$$

The rest of the proof focuses on showing the following:

$$|\delta_{M^*}(L - X)| \leq C \cdot |L| - (2C - 1) \cdot |X|. \tag{4.2}$$

If this holds, we can bound (4.1) by $C \cdot C \cdot |L|$ thus showing $\texttt{val}(p') \leq \texttt{val}(p)$.

To show (4.2), first consider the graph G' obtained from G^t by directing all edges to higher layers: so an edge in E'_i is directed from L_i to R_i and an edge in E^*_i is directed from L_i to R_{i+1}. Let S consist of R_t plus all nodes reachable from X in G', including X itself. We claim $|\delta^{in}_{G'}(S)| = C \cdot n$. This can be seen easily:

$$|\delta^{in}_{G'}(S)| = |\delta^{in}_{G'}(S)| - |\delta^{out}_{G'}(S)| = \sum_{v \in S} |\delta^{in}_{G'}(v)| - \sum_{v \in S} |\delta^{out}_{G'}(S)| = \sum_{v \in S \cap R_t} |\delta^{in}(v)| = n \cdot C.$$

The first equality holds because $\delta^{out}_{G'}(S) = \emptyset$ by construction of S, the second holds for any cut of any directed graph, and the third holds because $S \cap L_1 = \emptyset$ (as $X \cap L_1 = \emptyset$) and because every vertex not in L_1 or R_t has equal in- and out-degree.

Now let Y be all endpoints of edges in $\delta_{E^*}(X)$ and let G'' be the subgraph of G' obtained by deleting Y and incident edges. Let $S' = S - Y$, we claim $|\delta_{G''}(S')| \leq n \cdot C - (C - 1) \cdot |X|$. One should think that $\delta_{G''}(S')$ is obtained by deleting edges of $\delta^{in}_{G'}(X) \cap \delta^{in}_{G'}(S)$ from $\delta^{in}_{G'}(S)$. There are precisely $C \cdot |X|$ edges in $\delta^{in}_{G'}(X)$, we show at least $(C - 1) \cdot |X|$ of there were also in $\delta^{in}_{G'}(S)$.

To that end, consider an edge $e \in \delta^{in}_{G'}(x)$ for some $x \in X$ that does not lie in $\delta^{in}_{G'}(S)$. Then v is reachable from some other node of X in G' by construction of S, pick the deepest such node and call this node $\tau(e)$. By this choice for $\tau(e)$, there is a $\tau(e) - x$ path in G' that avoids every other vertex in X. Also, the length of this path is at most $2t$ because the paths are monotone with respect to the layers of G'. Also note for two different $e, e' \in \delta^{in}_{G'}(x) - \delta^{in}_{G'}(S)$ that $\tau(e) \neq \tau(e')$, or else we have two different $\tau(e) - x$ walks implying there is a cycle of length at most $4t$ in G^t which is not possible.

Build an auxiliary graph $\mathcal{T} = (X, F)$ where for each $e \in \delta^{in}_{G'}(x) - \delta^{in}_{G'}(S)$ for some $x \in X$ we include an undirected edge from $\tau(e)$ to x in F. By the above discussion, this is a simple graph. We also claim it is a forest, otherwise consider a cycle C in $\mathcal{T}$. Focus on some edge $xy \in C$ and let $z \notin \{x, y\}$ be another node in C. As the xy-path from the construction in the last paragraph avoids z, we get two different $x - y$ walks in G^t by following the paths corresponding to the two directions around C from x to y. But this is impossible because G^t has no cycle of length at most $2t \cdot |X| \leq 2t \cdot \rho$. So, $|F| \leq |X| - 1$ meaning $|\delta^{in}_{G'}(X) - \delta^{in}_{G'}(S)| \leq |X| - 1$. Thus,

$$|\delta^{in}_{G''}(S')| \leq C \cdot n - (C-1) \cdot |X|. \tag{4.3}$$

Now we can prove (4.2). Let $\overline{G}''$ be the undirected version of G'', so $\overline{G''}$ is obtained from G^t by deleting Y and its incident edges from G^t. Call a subset of edges of $\overline{G}''$ a matching if they satisfy the capacity constraints of nodes in $\overline{G}''$. Note $\delta_{M^*}(L - X)$ is a matching.

We bound the size of a maximum matching in $\overline{G}''$. First, observe $M := E^* - \delta(X)$ is a matching in $\overline{G}''$ and that G'' is the directed graph we get by directing edges along this matching. That is, the set of $L_1 - R_t$ paths in G'' are exactly the set of M-alternating path. By the max-flow/min-cut theorem, the maximum number of edge-disjoint M-alternating paths is at most $|\delta^{in}_{G''}(S')| \leq C \cdot n - (C-1) \cdot |X|$. So the maximum size of a matching in $\overline{G}''$ is at most

$$|M| + C \cdot n - (C-1) \cdot |X| = C \cdot (L - n) - C \cdot |X| + C \cdot n - (C-1) \cdot |X| \leq C \cdot |L| - (2C-1) \cdot |X|.$$

This proves (4.2) and completes the analysis of the locality gap.

□

□

4.3 APX-Hardness for L-Sided Pricing

In this section we provide a proof for Theorem 7 which says L-Sided Pricing is **APX**-hard, even if all capacities are at most 4 and all customers have a budget of 1 or 2.

Proof. We reduce from the Vertex Cover problem for 3-regular graphs, which is known to be **APX**-hard [1]. Let $G = (V, E)$ be a 3-regular graph, with $|V| = n$ nodes and $|E| = m = \frac{3n}{2}$ edges.

Construct the following bipartite graph $G' = (L \cup R, E')$ from G. Here, L is a copy of V and R is a copy of V plus a copy of E. Each $v \in L$ has capacity 4 and each vertex in R has a capacity of 1. For a node $v_i \in V$, let l_i denote its copy in L and r_i denote its copy in R. Similarly, for each edge $e_j \in E$ let d_j denote its copy in R.

All customers have budget equal to 1 or 2, and they fall into two classes: node customers and edge customers. For each $v_i \in V$, we have a node customer who is interested in l_i and r_i with budget 2. For each edge $e_j = v_i v_k \in E$, we define two edge customers interested in $l_i d_j$ and $l_k d_j$ respectively, both with budget 1. We claim that the optimal solution to this L-Sided Pricing instance G' has profit $m + 2n - k$ where k is the size of the smallest vertex cover of G.

First, suppose S is a vertex cover of G with $|S| = k$. Consider the pricing p with $p(l_i) = 1$ if $l_i \in S$ and $p(l_i) = 2$ if $l_i \notin S$. As S is a vertex cover in G, for each $e_j = v_i v_k \in E$ we have at least one of $l_i d_j$ or $l_k d_j$ is incident to a vertex with price 1. Form $F' \subseteq E'$ by adding one such edge from each e_j and adding all node customers. We get profit m from edge customers, profit $2(n - k)$ from all node customers $l_i r_i$ such that $v_i \notin S$, and profit k from all node customers $l_i r_i$ with $v_i \in S$ for a total profit of $m + 2n - k$.

Conversely, consider an optimal pricing p, so each price is 1 or 2. For

$e_j = v_i v_k \in E$, we claim that either $p(v_i) = 1$ or $p(v_k) = 1$. If not, then consider changing $p(v_i)$ to 1. We lost a profit of 1 from the node customer $l_i r_i$ but have gained a profit of 1 by adding $v_i d_j$, which remains feasible because neither $v_i d_j$ nor $v_k d_j$ could afford the price of their left-endpoint before (i.e. d_j is not used by any edge that can afford their price under pricing p, so we may add $v_i d_j$ after adjusting prices).

Set $S = \{v_i : p(l_i) = 1\}$. By the above argument, S is a vertex cover of G. Also observe that the optimal set of edges of G' under prices p will include every node customer plus exactly one from each pair $\{l_i d_j, l_k d_j\}$ for each $e_j = v_i v_k \in E$. So the profit of p is $m + 2n - |S|$.

Therefore, the optimal profit in G' is exactly $\frac{5}{2} \cdot n - k$ where k is the size of a minimum vertex cover of G. There are constants $0 < \alpha < \beta < 1$ such that it is **NP**-hard to distinguish between 3-regular graphs having vertex covers of size $\leq \alpha \cdot n$ and 3-regular graphs requiring vertex covers of size $\geq \beta \cdot n$. So it is **NP**-hard to distinguish between L-Sided Pricing instances that have optimal profit at least $\left(\frac{5}{2} - \alpha\right) \cdot n$ or at most $\left(\frac{5}{2} - \beta\right) \cdot n$. □

4.4 A Lower Bound on the Integrality Gap of (LP-Pricing)

Consider the following example of L-Sided Pricing where we have 2 items a and b on the left side which we have to assign a price and three items c, d and e on the right side for which we have to set the price to zero. The capacities of all items are 1 except for the item b which has capacity 2. Also, as you can see in the picture we have 5 customers (edges) as well each with budget 2 except the one interested in items b and e which has budget 1.

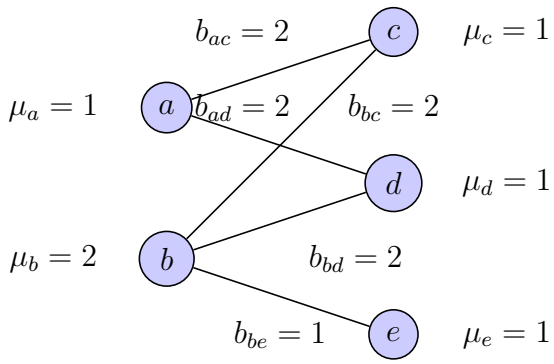

Figure 4.2: An Example Instance Showing A Lower Bound on the Integrality Gap of (**LP-Pricing**)

The optimal integral solution for this instance would gain the total profit of 4 by setting the price for items a and b to 2 and picking the customers interested in bundles $\{a, c\}$ and $\{b, d\}$ as buyers. Of course, there could be some other optimal solutions with the same total profit of 4.

On the other hand, consider this assignment of fractional values to the variables of (**LP-Pricing**):

$$y_{a,2} = 1, \quad y_{b,1} = \frac{1}{2}, \quad y_{b,2} = \frac{1}{2}$$

$$x_{ac,2} = \frac{1}{2}, \quad x_{ad,2} = \frac{1}{2}, \quad x_{bc,2} = \frac{1}{2}, \quad x_{bd,2} = \frac{1}{2}, \quad x_{be,1} = \frac{1}{2}$$

One can simply verify that this assignment does not violate any constraint of (**LP-Pricing**) and the total profit is 4.5. This means that the integrality gap of (**LP-Pricing**) is at least $\frac{9}{8} = 1.125$.

Corollary 1. *The integrality gap of* (**LP-Pricing**) *is no better than* 9/8 *in any instance of* L-Sided Pricing.

Note that the best rounding algorithm for (**LP-Pricing**) we know has approximation ratio much bigger than 9/8 and there is a big gap between the lower bound and the upper bound of (**LP-Pricing**).

Chapter 5

Conclusion

5.1 Incorporating Loops

The algorithms presented in this paper assumed every customer was interested in a bundle with precisely two distinct items. This was done for notational simplicity. However, the algorithmic results extend very easily to the case where some customers may be only interested in a single item. The reduction to L-Sided Pricing is valid in this case as well and we only have to consider singleton customers interested in an item in L. One can still compute an optimum matching for a given pricing in this case, so the local search algorithm can still be executed. The analysis of the local search algorithms using test swaps can then be adapted in a straightforward way by removing singleton customers from the local optimum and adding singleton customers from the global optimum who are interested in an item whose price changed when constructing the matching used to generate the inequality for this swap.

Similarly, the LP-based $(1 + \epsilon)$-approximation for L-Sided Pricing with large capacities from Chapter 3 is trivial to adapt. The "edge-variables" for singleton customers interested only an item in L do not contribute to the load of any constraint for any $v \in R$. The randomized rounding algorithm is identical.

5.2 Future Directions

We presented an 8-approximation for Capacitated Graph Pricing. If all capacities were bounded from above by a constant or, in simple graphs, were bounded from below by a sufficiently large constant then we get better approximations. It would be nice to combine these two cases to beat the 8-approximation in any Capacitated Graph Pricing instance even if only for simple graphs. But the techniques we use are quite different and it is not clear how to combine them in a single algorithm that works in the presence of both small and large capacities.

It would also be interesting to know if the hardness lower bound for Capacitated Graph Pricing is worse than 4. Intuitively, this could be the case as the L-Sided Pricing problem we reduce to is **APX**-hard in the capacitated case.

We have proved both upper and lower bound for (**LP-Pricing**). But there is still a big gap between these two bounds. Closing this gap by finding either a better rounding algorithm or another instance with larger difference between the optimal integral and fractional solution would be another interesting direction to follow.

We also briefly remark that the generalization of Capacitated Graph Pricing in hypergraphs, where each hyperedge has size $\leq k$, is a common generalization of the uncapacitated case which has a hardness of $\Omega(k^{1-\epsilon})$ [8], and the k-Set Packing problem which has a hardness of $\Omega(k/\log k)$ [23]. One then wonders if Capacitated Graph Pricing in hypergraphs could be hard to approximate better than $\Omega(k^{2-\epsilon})$. It would be interesting to determine if this is the case or to see if there is a noticeably better approximation than our $O(k^2)$ approximation, perhaps even $O(k)$.

References

[1] P. Alimonti and V. Kann, "Some apx-completeness results for cubic graphs.", *Theoretical Computer Science*, vol. 237, no. 1, pp. 123–134, 2000. 51

[2] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit, "Local search heuristics for k-median and facility location problems", *SIAM J. Comput*, vol. 33, no. 3, pp. 544–562, 2004. 25, 32

[3] M.-F. Balcan, A. Blum, and Y. Mansour, "Item pricing for revenue maximization", in *Proceedings 9th ACM Conference on Electronic Commerce (EC-2008), Chicago, IL, USA, June 8-12, 2008*, 2008, pp. 50–59. [Online]. Available: `https://doi.org/10.1145/1386790.1386802`. 15

[4] M.-F. Balcan and A. Blum, "Approximation algorithms and online mechanisms for item pricing", *Theory of Computing*, vol. 3, no. 9, pp. 179–195, 2007. 12, 13, 21, 38

[5] P. Briest and P. Krysta, "Single-minded unlimited supply setting pricing on sparse instances", in *In Proceedings of SODA*, 2006, pp. 1093–1102. 13

[6] P. Briest and P. Krysta, "Buying cheap is expensive: Hardness of non-parametric multi-product pricing", in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, 2007, pp. 716–725. [Online]. Available: `http://dl.acm.org/citation.cfm?id=1283383.1283460`. 17

[7] P. Chalermsook, J. Chuzhoy, S. Kannan, and S. Khanna, "Improved hardness results for profit maximization pricing problems with unlimited supply", in *In Proceedings of APPROX*, 2012, pp. 73–84. 13

[8] P. Chalermsoon, B. Laekhanukit, and D. Nanongkai, "Independent set, induced matching, and pricing: Connections and tight (subexponential time) approximation hardnesses", in *In Proceedings of FOCS*, 2013, pp. 370–379. 56

[9] N. Chen and X. Deng, "Envy-free pricing in multi-item markets", in *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part II*, 2010, pp. 418–429. [Online]. Available: https://doi.org/10.1007/978-3-642-14162-1%5C_35. 15

[10] N. Chen, X. Deng, P. W. Goldberg, and J. Zhang, "On revenue maximization with sharp multi-unit demands", *J. Comb. Optim.*, vol. 31, no. 3, pp. 1174–1205, 2016. [Online]. Available: https://doi.org/10.1007/s10878-014-9817-y. 15

[11] N. Chen, A. Ghosh, and S. Vassilvitskii, "Optimal envy-free pricing with metric substitutability", in *Proceedings 9th ACM Conference on Electronic Commerce (EC-2008), Chicago, IL, USA, June 8-12, 2008*, 2008, pp. 60–69. [Online]. Available: https://doi.org/10.1145/1386790.1386803. 14

[12] M. Cheung and C. Swamy, "Approximation algorithms for single-minded envy-free profit-maximization problems with limited supply", in *In Proceedings of FOCS*, 2008, pp. 35–44. 13

[13] D. P. Dubhashi and A. Panconesi, *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. 11

[14] K. Elbassioni, M. Fouz, and C. Swamy, "Approximation algorithms for non-single-minded profit-maximization problems with limited supply", in *In Proceedings of the International Workshop on Internet and Network Economics (WINE)*, 2012, pp. 462–472. 14

[15] K. M. Elbassioni, R. Raman, S. Ray, and R. Sitters, "On profit-maximizing pricing for the highway and tollbooth problems", in *Algorithmic Game Theory, Second International Symposium, SAGT 2009, Paphos, Cyprus, October 18-20, 2009. Proceedings*, 2009, pp. 275–286. [Online]. Available: https://doi.org/10.1007/978-3-642-04645-2%5C_25. 17

[16] M. Feldman, A. Fiat, S. Leonardi, and P. Sankowski, "Revenue maximizing envy-free multi-unit auctions with budgets", in *Proceedings of the 13th ACM Conference on Electronic Commerce, EC 2012, Valencia, Spain, June 4-8, 2012*, 2012, pp. 532–549. [Online]. Available: https://doi.org/10.1145/2229012.2229052. 14

[17] Z. Friggstad, K. Khodamoradi, and M. R. Salavatipour, "Exact algorithms and lower bounds for stable instances of euclidean k-means", in *In Proceedings of SODA*, 2019, pp. 2958–2972. 25, 32

[18] I. Gamzu and D. Segev, "A sublogarithmic approximation for highway and tollbooth pricing", in *In Proceedings of ICALP*, 2010, pp. 582–593. 17

[19] F. Grandoni and T. Rothvoss, "Pricing on paths: A ptas for the highway problem", in *In Proceedings of SODA*, 2011, pp. 675–684. 17

[20] A. Grigoriev, J. van Loon, R. Sitters, and M. Uetz, "How to sell a graph: Guidelines for graph retailers", in *Graph-Theoretic Concepts in Computer Science, 32nd International Workshop, WG 2006, Bergen, Norway, June 22-24, 2006, Revised Papers*, 2006, pp. 125–136. [Online]. Available: https://doi.org/10.1007/11917496%5C_12. 16

[21] V. Guruswami, J. D. Hartline, A. R. Karlin, D. Kempe, C. Kenyon, and F. McSherry, "On profit-maximizing envy-free pricing", in *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, 2005, pp. 1164–1173. 13, 14, 16

[22] J. D. Hartline and V. Koltun, "Near-optimal pricing in near-linear time", in *Algorithms and Data Structures, 9th International Workshop, WADS 2005, Waterloo, Canada, August 15-17, 2005, Proceedings*, 2005, pp. 422–431. [Online]. Available: https://doi.org/10.1007/11534273%5C_37. 14

[23] E. Hazan, S. Safra, and O. Schwartz, "On the complexity of approximating k-set packing", *Computational Complexity*, vol. 15, no. 1, pp. 20–39, 2006. 56

[24] N. Karmarkar, "A new polynomial time algorithm for linear programming", in *PALP*, ser. Combinatorica, vol. 4(4):373–395, 1984. 9

[25] R. Khandekar, T. Kimbrel, K. Makarychev, and M. Sviridenko, "On hardness of pricing items for singleminded bidders", in *In Proceedings of APPROX*, 2009, pp. 202–216. 13

[26] S. Khot, "On the power of unique 2-prover 1-round games", in *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, 2002, pp. 767–775. [Online]. Available: https://doi.org/10.1145/509907.510017. 8

[27] S. Khot, D. Minzer, and M. Safra, "On independent sets, 2-to-2 games, and grassmann graphs", in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, 2017, pp. 576–589. [Online]. Available: https://doi.org/10.1145/3055399.3055432. 8

[28] ——, "Pseudorandom sets in grassmann graph have near-perfect expansion", in *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, 2018, pp. 592–601. [Online]. Available: https://doi.org/10.1109/FOCS.2018.00062. 8

[29] E. Lee, "Hardness of graph pricing through generalized max-dicut", in *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, 2015, pp. 391–399. 13

[30] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge University Press, 1995. 11, 22

[31] F. Rajabi-Alni, A. Bagheri, and B. Minaei-Bidgoli, "An o(nˆ3) time algorithm for the maximum weight b-matching problem on bipartite graphs", *CoRR*, vol. abs/1410.3408, 2014. arXiv: 1410.3408. [Online]. Available: http://arxiv.org/abs/1410.3408. 21

[32] H. Sachs, "Regular graphs with given girth and restricted circuits", *Journal of the London Mathematical Society*, vol. s1-32, no. 1, pp. 423–429, 1963. 45, 46

[33] A. Schrijver, "Combinatorial optimization : Polyhedra and efficiency. algorithms and combinatorics", *Springer-Verlag*, 2003. 37, 43

[34] V. Vazirani, "Approximation algorithms.", in *AA*, ser. Lecture Notes in Computer Science, vol. 380, Springer, 2003. 2, 7

[35] D. West, "Introduction to graph theory", in *ITGT*, ser. Lecture Notes in Computer Science, vol. 504, Prentice-Hall, 2001. 2

[36] D. Williamson and D. Shmoys, "The design of approximation algorithms", in *DAA*, ser. Lecture Notes in Computer Science, vol. 504, Cambridge University Press, 2011. 2

[37] D. Zuckerman, "Linear degree extractors and the inapproximability of max clique and chromatic number", in *LDIMCCN*, ser. THEORY OF COMPUTING, vol. 3:103–128, 2007. 7