Improved Approximation Guarantees for Lower-Bounded Facility Location

by

Sara Ahmadian

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

We consider the *lower-bounded facility location* (LBFL) problem (, also known as *load-balanced facility location*), which is a generalization of *uncapacitated facility location* (UFL) problem where each open facility is required to serve a *minimum* number of clients. More formally, in the LBFL problem, we are given a set of clients $\mathcal{D}$, a set of facilities $\mathcal{F}$, a non-negative *facility-opening cost* f_i for each $i \in \mathcal{F}$, a lower bound M, and a distance metric $c(i, j)$ on the set $\mathcal{F} \cup \mathcal{D}$, where $c(i, j)$ denotes the cost of assigning client j to facility i. A feasible solution S specifies the set of open facilities $F^S \subseteq \mathcal{F}$ and the assignment of each client j to an open facility $i(j)$ such that each open facility serves *at least* M clients. Our goal is to find feasible solution S that minimizes $\sum_{i \in F^S} f_i + \sum_j c(i(j), j)$.

The current best approximation ratio for LBFL is 550 [20]. We substantially advance the state-of-the-art for LBFL by devising an approximation algorithm for LBFL that achieves a significantly-improved approximation guarantee of 83.

Acknowledgements

First and foremost, I would like to thank my supervisor, Chaitanya Swamy, for his generous and invaluable guidance during the course of research. Our weekly meetings have been a stimulating experience and a source of new ideas for me, and all the results presented in this thesis are achieved during my joint work with him. No words can express my gratitude towards him. His careful proofreading of the document is particularly appreciated.

I thank Shubham Gupta for the useful discussion on local search algorithm for *lower-bounded facility location* problem. I also wish to thank Abbas Mehrabian for taking care of some paper-works while i was absent.

Last but not least, I would like to express my love and gratitude to my parents and my fiance, for their love and support throughout the years.

Dedication

To my parents, and my fiance
for their patience and support.

Contents

List of Figures

Chapter 1

Introduction

Where should a franchisee open its stores? Where should the stations of a new subway in a city be located? Such problems are often encountered while setting up chain stores ,or other businesses with distributed production or supply centres. Typically, these centres, or *facilities* can be set up at strategic locations for a location-dependent *opening cost*. Each customer, or *client*, interacts with a facility at a price, which is called its *connection cost*. Mathematically, problems underlying the above examples are usually abstracted by a class of problems commonly referred to as *facility location problems*. The goal in problems is to efficiently determine the set of locations at which facilities should be opened so as to serve clients in a cost-efficient manner.

Facility location problems have been studied widely in Operations Research and Computer Science since the 1960s [2, 11, 14, 17, 18], and a number of different variations of facility location problems have been proposed in the literature. The simplest such problems is the *uncapacitated facility location* (UFL) problem, also known as the *simple plant location* problem, in which a set of candidate locations for facilities and a set of clients who need to be served are given. The goal is to decide which facilities to open to minimize the total costs of opening these facilities and servicing each client by its closest open facility. A more general problem is *capacitated facility location* (CFL) problem, where each facility can serve at most a specific number of clients. A further generalization is *universal facility location* (UniFL), in which the opening cost of a facility is a function of the number of clients assigned to it. Most versions of facility location are computationally intractable, so presumably, an optimal solution can not be found effectively.

A reasonable approach in such cases is to efficiently compute a solution to the facility location problem that is within some provable bound of the optimum. This approach falls in the framework of *approximation algorithm*, and the bound is called the *approximation ratio* or *approximation guarantee*. Approximation algorithms have been studied widely since the 1960s; see e.g. [5].

In the past few years, a variety of techniques that yield constant approximation ratios has been derived for the facility location problems. Initial attempts to solve these problems were greedy heuristics, which have been considerably refined. Algorithms for facility locations can be categorized in three major groups. One approach is *rounding algorithms*, which consider the linear programming (LP) relaxation of the integer programming formulation of the problem, and round the optimal LP solution to an integer solution to guarantee a good approximation. The maximum ratio of the cost of the optimal integer solution to the cost of the optimal fractional solution is called the *integrality gap* of the LP relaxation. There are various facility location problems, where the integrality gap of the "natural" LP relaxation of the problem could be arbitrarily large, so this approach would not be helpful in solving such problems (e.g., CFL). Another approach that relies implicitly on linear programming is that of *primal-dual algorithms*. In this approach, the aim is to simultaneously construct a feasible dual solution and a primal (integer) solution, and use the dual to bound the cost of the constructed primal solution. Finally, there are some techniques based on *local-search algorithms* that move from one solution to another solution in the space of candidate solutions and usually return the local optimum. The *locality gap* of a local-search algorithm is the maximum ratio of the cost of a locally optimum solution to the cost of a globally optimum solution. For certain facility location variants especially the variants with large integrality gap, the only technique that yields constant approximation ratios is based on local search algorithms.

In this thesis, we consider the *lower-bounded facility location* (LBFL) (also known as *load balanced facility location* [7]), which is the generalization of UFL where each facility needs to serve at least a certain number of clients. LBFL finds a direct application when a franchisee must open stores to serve clients and also needs each store to serve a minimum number of clients to be profitable. We devise an algorithm for the LBFL that improves the current best approximation ratio of the LBFL. Our approach involves reducing a special structured LBFL instance to the problem that we introduce, called *capacity-discounted UFL* (CDUFL). CDUFL is a special case of CFL where facilities are either uncapacitated or capacitated with a zero opening cost. We develop a local-search algorithm for the CDUFL and prove that it has a constant approximation ratio, and this is one the components that leads to improved approximation ratio.

1.1 Problem Definition and LP relaxation

In the LBFL problem, we are given a set of clients $\mathcal{D}$, a set of facilities $\mathcal{F}$, a non-negative *facility-opening cost* f_i for each $i \in \mathcal{F}$, a lower bound M, and a distance metric $c(i, j)$ on the set $\mathcal{F} \cup \mathcal{D}$, where $c(i, j)$ denotes the cost of assigning client j to facility i. A feasible solution S specifies the set of open facilities $F^S \subseteq \mathcal{F}$ and the assignment of each client j to an open facility $i(j)$ such that each open facility serves *at least* M clients. Our goal is

to find feasible solution S that minimizes:

$$\sum_{i \in F^S} f_i + \sum_j c(i(j), j).$$

We sometimes abuse the notation and use F^S to also denote the facility opening cost of the open facilities. Defining $C^S = \sum_j c(i(j), j)$, we can say that we are looking for solution S that minimizes $F^S + C^S$ such that $|\{j : i(j) = i\}| \geq M$ for each $i \in F^S$. We denote this LBFL instance by $\mathcal{I}$ and refer to its optimal solution by S^*, which has connection cost of C^* and facility cost (and set of open facilities) F^*. We sometimes use $OPT(\mathcal{I})$ to refer to the total cost of solution S^*, i.e., $OPT(\mathcal{I}) = C^* + F^*$.

We may consider the following natural integer linear programming (ILP) formulation for LBFL. In this program, y_i is an indicator variable denoting whether facility i is open, and x_{ij} is an indicator variable denoting whether client j is assigned to facility i.

$$\begin{aligned}
\min \quad & \sum_i f_i y_i + \sum_{j,i} c(i,j) x_{ij} && && \text{(LBFL LP)} \\
\text{s.t.} \quad & \sum_i x_{ij} = 1 && \forall j && (1.1) \\
& x_{ij} \leq y_i && \forall i, j && (1.2) \\
& \sum_j x_{ij} \geq M y_i && \forall i && (1.3) \\
& x_{ij}, y_i \in \{0, 1\} && \forall i, j.
\end{aligned}$$

Constraint 1.1 states that each client has to be assigned to a facility and constraint 1.2 ensures that this facility is open. Constraint 1.3 ensures that each open facility serves at least M clients. It is easy to see that an optimal solution to ILP corresponds to an optimum solution to the LBFL instance $\mathcal{I}$.

By relaxing the last two constraint to $x_{ij} \geq 0$ and $y_i \geq 0$, we get the natural linear programming of this ILP which can be solved efficiently. However, the natural LP-relaxation has a large integrality gap, i.e., the fractional solution can have arbitrarily small cost compared to the optimal integer solution (see Appendix A). Thus, we are led to consider alternate approaches for obtaining a good approximation algorithm for LBFL.

1.2 Related work

There is a large body of literature that deals with approximation algorithms for (metric) UFL, CFL and its variants; see [15] for a survey on UFL. The first constant-factor approximation guarantee for UFL was obtained by Shmoys, Tardos, and Aardal [16] via

an LP-rounding algorithm, and the current state-of-the-art for UFL is a 1.5-approximation algorithm due to Byrka [3]. Local-search techniques have also been utilized to obtain $O(1)$-approximation guarantees for UFL [1, 4, 10]. We apply some of the ideas of [1, 4] in our algorithm.

Lower-bounded facility location was introduced by Guha et al. [7] and Karger et al. [9] independently. Guha et al. [7] call it *load balanced facility location*, and use it as a subroutine for solving the *access network design* problem, which is a special case of the *single-sink buy-at-bulk* problem. Karger et al. [9] introduced the problem as *r-gathering* problem which was used to solve *maybecast* problem, which models a network design problem under uncertain or incomplete information (and reduces to the rent or buy problem). Both of them present a bicriteria approximation algorithm for LBFL that for each $\alpha \in [0, 1)$ finds a solution in which each facility serves at least αM clients, and has cost at most a constant-factor depending on α times the optimum. We sometimes refer to this solution as *bicriteria solution*. Lim et al. [12] studied the mixed integer programming for special case of LBFL where facility opening costs are zero. For this special case of LBFL, they devise a greedy approximation algorithm whose approximation ratio is $2M$. Svitkina [20] presents the first constant-factor approximation algorithm for LBFL problem. Her algorithm uses the bicriteria algorithm of [7, 9] to pre-process the instance and obtain a more structured LBFL instance (see Chapter 2).

LBFL is the opposite of CFL in some aspects. LBFL requires each facility to serve a certain minimum number of clients whereas CFL sets the bound on the maximum number of clients that can be served by a facility.Also, all the constant-factor approximation algorithms for CFL are based on local search, and the current best approximation for CFL is $5.83+\epsilon$ due to Zhang et al. [21]. However, we are not aware of any constant-factor local-search algorithms for LBFL. Actually, in Appendix B, we present an example which shows the large locality gap for local-search algorithm based on add, delete, and swap moves for LBFL.

A formulation that generalizes many facility-location variants is the universal facility location problem. This problem was introduced by Hajiaghayi et al. [8] who devise a constant-factor approximation for the special case where opening cost of a facility is a concave function. Mahdian et al. [13] gave a constant approximation for arbitrary monotone non-decreasing opening cost function. We are not aware of any work on UniFL with arbitrary non-increasing opening cost functions, which generalizes LBFL. Guha et al. [6] give a constant-factor approximation for the case where the cost-functions do not decrease too steeply (the constant depends on the steepness); notice that LBFL does not fall into this class so their results do not apply here.

1.3 Our contribution

We devise an approximation algorithm for LBFL that achieves a substantially-improved approximation guarantee of 82.5 and significantly improves the state-of-the-art for LBFL. Our improvement comes from a combination of ideas in algorithm design and analysis, and yields new insights about the approximability of LBFL.

As in Svitkina's algorithm, our algorithm includes a pre-processing step in which a bicriteria solution is obtained which is then used to construct a more structured LBFL instance $\mathcal{I}_2$. However, this bicriteria solution is obtained in a different fashion from Svitkina's algorithm. More precisely,similar to [7, 9], we construct a UFL instance from the LBFL instance $\mathcal{I}$, but we do so in a subtly different fashion, and then we use the local-search algorithm of [1, 4] for solving it. Our UFL instance is defined in a way that enables us to utilize techniques in [16, 19] in our analysis of LBFL. Also, local-search algorithms give asymmetric bounds on the facility-opening and connection cost which assists us in presenting a tighter analysis for LBFL.

We construct a special structured LBFL instance $\mathcal{I}_2$ in a similar fashion to [20] using the bicriteria solution, but we solve $\mathcal{I}_2$ by reducing it to CDUFL rather than CFL. We devise a local-search algorithm for CDUFL which has a better approximation ratio than CFL. This results in better approximation ratio for $\mathcal{I}_2$ and hence to $\mathcal{I}$.

1.4 Structure of the thesis

The rest of this thesis is structured as follows. In chapter 2, we first present the bicriteria algorithm by [7, 9]. The ideas in this algorithm, which lead to bicriteria guarantees for LBFL, play a preprocessing role in Svitkina's algorithm for LBFL [20] and (slightly indirectly) in our algorithm. Subsequently, we describe Svitkina's algorithm in Chapter 2. In Chapter 3, we present our algorithm and prove our approximation factor of 83 for LBFL. We highly encourage the reader to read Chapter 2 before Chapter 3 since we use some of the results in Chapter 2 in our algorithm. One of our improvements comes from a reduction of a structured LBFL problem to CDUFL. In Chapter 4, we present our algorithm for CDUFL which is based on local-search. Finally, we conclude in Chapter 5 with a summary of our results, and discuss possible future work.

Chapter 2

Lower-Bounded Facility Location

In this chapter, we focus on the *lower-bounded facility location* (LBFL) problem. LBFL is a generalization of UFL, which comes with an extra bound constraint on the *minimum* number of clients served by each open facility. We devise an algorithm for this problem which achieves approximation guarantee of 83. Our result improves significantly the previous best (and first) constant-factor approximation guarantee of 550 by Svitkina [20]. In order to be able to describe our ideas, we present the algorithm by Svitkina and its analysis in this chapter.

First, in Section 2.1, we discuss Svitkina's algorithm briefly. Then, in Section 2.2, we describe the bicriteria algorithm by [7, 9], which is a pre-processing step in Svitkina's algorithm. Svitkina shows that one can reduce LBFL instance to a special structured LBFL instance that is obtained by using bicriteria solution. We detail this reduction in Section 2.3. She solves the resulting LBFL instance by reduction to CFL, where we discuss these transformations in Section 2.4.

2.1 Sketch of Algorithm

In this section, we give a brief description of the algorithm in [20] and the ideas underlying it. The algorithm is shown in Algorithm 2.1. Svitkina's algorithm starts with using the algorithm in [7, 9] to obtain a bicriteria solution. This bicriteria solution is then used to construct a special structured LBFL instance $\mathcal{I}_2$ in step 2 with the following structure: (i) all clients are aggregated at a subset $\mathcal{F}_2 \subseteq \mathcal{F}$ of the facilities with each facility $i \in \mathcal{F}_2$ having $n_i \geq \alpha M$ co-located clients; (ii) all facilities in $\mathcal{F}_2$ have zero opening costs; and (iii) near-optimal solutions to $\mathcal{I}_2$ translate to near-optimal solutions to $\mathcal{I}$ (and vice versa). Now the goal is to close a subset of facilities in $\mathcal{F}_2$ such that transferring their co-located

clients to the remaining (open) facilities ensures that each open facility serves at least M clients (and the cost incurred is "small").

Algorithm 2.1 Svitkina's Algorithm for LBFL

1: Get a bicriteria solution for LBFL instance $\mathcal{I}$.
2: Use bicriteria solution to modify LBFL instance,and get new instance $\mathcal{I}_2$.
3: Construct CFL instance $\mathcal{I}_{\mathsf{CFL}}$from $\mathcal{I}_2$.
4: Solve $\mathcal{I}_{\mathsf{CFL}}$.
5: Transfer obtained $\mathcal{I}_{\mathsf{CFL}}$-solution in step 4 to $\mathcal{I}_2$-solution.
6: Transfer obtained $\mathcal{I}_2$-solution in step 5 to $\mathcal{I}$-solution.

We solve $\mathcal{I}_2$ as follows. The idea for solving $\mathcal{I}_2$ is to reduce it to the CFL instance $\mathcal{I}_{\mathsf{CFL}}$ obtained in step 3 by reversing the role of facilities and clients. Roughly speaking, each group of co-located clients in $\mathcal{I}_2$ corresponds to a supply point in $\mathcal{I}_{\mathsf{CFL}}$ with capacity equal to the size of this group. A facility in $\mathcal{I}_2$ that needs to be filled to reach the lower bound M, corresponds to a demand point with demand equal to the number of empty slots at i. Now the goal is to find a solution that specifies how to fill the empty slots. The constructed CFL problem is solved by one of the constant-factor approximation algorithms for CFL problem in step 4. This near-optimal solution to $\mathcal{I}_{\mathsf{CFL}}$ is then converted to a near-optimal solution of $\mathcal{I}_2$ in step 5. Thus, we now have a solution to $\mathcal{I}_2$, which is finally transferred to a near-optimal solution to $\mathcal{I}$ (by property (iii) of $\mathcal{I}_2$).

2.2 Bicriteria Algorithm

We now describe the bicriteria algorithm by [7], and [9]. Svitkina [20] uses the bicriteria algorithm as a subroutine for converting $\mathcal{I}$ to a more structured LBFL instance $\mathcal{I}_2$. Also, our algorithm (described in Chapter 3) uses a subtle modification of this algorithm to obtain the instance $\mathcal{I}_2$. For any $\alpha \in [0, 1)$, the bicriteria algorithm finds a solution in which there are at least αM clients assigned to each open facility and the cost of this solution is within some factor, which depends on α, of the optimal solution cost to $\mathcal{I}$. Bicriteria algorithm gives a trade-off between the cost of the solution and the amount by which the lower-bound constraint is violated. However, they are unable to give a non-trivial bound for the cost of feasible LBFL solution.

For each facility $i \in \mathcal{F}$, we define $\mathcal{D}(i)$ to be the set of M closest clients to i. Consider the UFL instance $\mathcal{I}'$ with the same set of clients and facilities, and the same connection costs as in $\mathcal{I}$ but with facility opening costs:

$$f'_i = f_i + \lambda \sum_{j \in \mathcal{D}(i)} c(i, j),$$

where $\lambda = \frac{2\alpha}{1-\alpha}$ is a scaling parameter. The term $\sum_{j \in \mathcal{D}(i)} c(i,j)$ is a lower-bound on the connection cost of clients assigned to a facility i if i is opened in a solution to LBFL.

The bicriteria algorithm consists of two main steps. In the first step, we solve $\mathcal{I}'$ with a ρ-approximation algorithm for UFL and obtain a solution in which each client is assigned to its closest open facility (this only improves the cost). We maintain this invariant in all steps of algorithm. If each open facility serves at least αM clients, we stop and return this solution as a bicriteria solution.

In the second step, we reassign the clients that are assigned to facilities serving less than αM clients. For each open facility i that serves less than αM clients, do the following: Reassign each client j served by i to its closest open facility other than i. After reassigning all the clients served by i, close i. We do this until there is no facility that serves less than αM clients. We will show that the second step can be done without incurring any extra costs.

To bound the cost of solution S'' obtained by the algorithm, we state following lemmas:

Lemma 2.2.1. *There exists a feasible solution S to $\mathcal{I}'$ with facility cost bounded by $F^*+\lambda C^*$ and connection cost bounded by C^*.*

Proof. Let S be the solution to $\mathcal{I}'$ which opens the same set of facilities opened in (the optimal solution to $\mathcal{I}$) S^* and assigns clients as they are assigned in S^*. The connection cost of S is the same as S^*. The opening cost of facilities in F^* is:

$$\sum_{i \in F^*} f'_i = \sum_{i \in F^*} \big(f_i + \sum_{j \in \mathcal{D}(i)} c(i,j)\big) \leq F^* + \lambda C^*.$$

The inequality comes from the fact that the connection cost of clients assigned to each open facility i is at least $\sum_{j \in \mathcal{D}(i)} c(i,j)$. □

Corollary 2.2.2. *Solution S' obtained at end of first step of the algorithm costs at most $\rho F^* + \rho(1+\lambda)C^*$.*

The next lemma bounds the cost incurs by reassigning the clients in the second step:

Lemma 2.2.3. *Let i be a facility that is closed down in the second step. The increase in the cost of reassigning clients served by i in the second step is at most f'_i.*

Proof. We recall that there are less than αM clients assigned to i, so there are at least $(1-\alpha)M$ clients in $\mathcal{D}(i)$ that are not assigned to i. The distance of the closest client j' among them to i is bounded by the average distance of these clients to i; $c(i,j') \leq \frac{\sum_{j \in \mathcal{D}(i)} c(i,j)}{(1-\alpha)M}$.

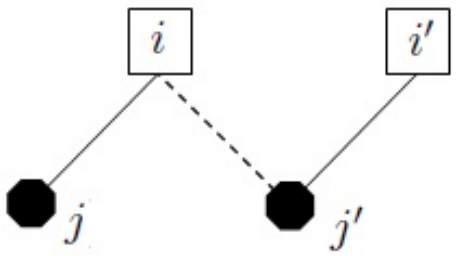

Figure 2.1: Use of the triangle inequality in Lemma 2.2.3

Let i' be the facility that serves j'. Since j' is not assigned to i we have $c(i', j') \leq c(i, j')$. So using the triangle inequality (see Figure 2.1), we can bound the distance between facilities i and i' by:

$$c(i, i') \leq c(i, j') + c(i', j') \leq 2c(i, j') \leq 2\frac{\sum_{j \in \mathcal{D}(i)} c(i, j)}{(1 - \alpha)M},$$

which is also an upper bound on the increase in the connection cost of each client that is reassigned from i to its nearest open facility. So the total additional cost for all such clients is at most:

$$\alpha M \cdot 2\frac{\sum_{j \in \mathcal{D}(i)} c(i, j)}{(1 - \alpha)M} = \lambda \sum_{j \in \mathcal{D}(i)} c(i, j) \leq f_i'.$$

□

Combining the results of Lemma 2.2.3 and corollary 2.2.2, we get the following theorem:

Theorem 2.2.4. *The above algorithm returns a solution to $\mathcal{I}'$ with facility cost F^b and connection cost C^b satisfying $F^b + C^b \leq \rho F^* + \rho \cdot \frac{1+\alpha}{1-\alpha} \cdot C^*$, where each open facility serves at least αM clients.*

Proof. From Corollary 2.2.2, we know that the cost of solution at the end of first step of algorithm is at most $F^b + C^b \leq \rho F^* + \rho \cdot \frac{1+\alpha}{1-\alpha} \cdot C^*$. By Lemma 2.2.3, we know that in the second step the total cost does not increase, since the increase in the connection cost is less than the decrease in the facility cost. So the result follows. □

2.3 Transformation to instance $\mathcal{I}_2$

Svitkina showed that if the LBFL instance has certain properties, then one can find a constant-factor solution for this instance. In this section, we show how to construct this special structured LBFL instance $\mathcal{I}_2$ given the bicriteria solution. We also prove that $OPT(\mathcal{I}_2)$ is bounded in terms of $OPT(\mathcal{I})$, and show how to convert a feasible solution to $\mathcal{I}_2$ to a feasible solution to $\mathcal{I}$ without incurring too much cost.

Let S^b be the solution found by the bicriteria algorithm for some parameter $\alpha > .5$ (to be specified later). Denote the connection cost of S^b by C^b and the facility opening cost (and set of open facilities) by F^b. Let $i^b(j)$ denote the facility to which j is assigned and $c_j^b = c(i^b(j), j)$. We define the LBFL instance $\mathcal{I}_2$ in the following way:

Definition 2.3.1. *Let $\mathcal{I}_2$ be an instance of LBFL with the same set of clients $\mathcal{D}$ and lower bound M as in $\mathcal{I}$. The set of facilities is $\mathcal{F}_2 = F^b$, and all facility costs are set to zero. The distance-metric is also modified as follows. Intuitively, we move each client j to the facility $i^b(j)$ that serves j in S^b. Formally, for any two clients j, j' and any two facilities i, i', we define: $c_2(i,j) = c(i, i^b(j))$; $c_2(j,j') = c(i^b(j), i^b(j'))$; and the distances between facilities remain unchanged, i.e., $c_2(i,i') = c(i,i')$.*

It is easy to see that the new distances form a metric. We will sometimes use the notation $\mathcal{I}_2(\alpha)$ to indicate explicitly that $\mathcal{I}_2$'s specification depends on the parameter α. Figure 2.2 depicts the transformation from $\mathcal{I}$ to $\mathcal{I}_2$ given a bicriteria solution.

In the next lemma, we bound the cost of an optimal solution to $\mathcal{I}_2$ in terms of $OPT(\mathcal{I})$, and the cost of S^b.

Lemma 2.3.2.

$$OPT(\mathcal{I}_2) \leq 2(C^b + C^*) \tag{2.1}$$

Proof. We construct a feasible solution S_2 to $\mathcal{I}_2$ from the optimal solution S^* to $\mathcal{I}$ in the following way. For each facility $i \in F^*$, transfer all clients served by i to its closest facility $i' \in \mathcal{F}_2$ (, and open i' if i' is not already open). Note that the above procedure returns a feasible solution since each open facility serves at least M clients. Now we just need to bound the connection cost of clients (the facility opening cost is zero). For any client j assigned to i', we have

$$\begin{aligned} c_2(i', j) = c(i', i^b(j)) &\leq c(i, i^b(j)) + c(i, i') \\ &\leq 2c(i, i^b(j)) \leq 2(c_j^b + c_j^*). \end{aligned} \tag{2.2}$$

The first inequality holds because of the triangle inequality (see Figure 2.3). Since $i^b(j) \in \mathcal{F}_2$, we can bound the distance $c(i,i') \leq c(i, i^b(j))$. Using the triangle inequality, the second inequality follows. Summing up (2.2) for all clients, the result follows.

□

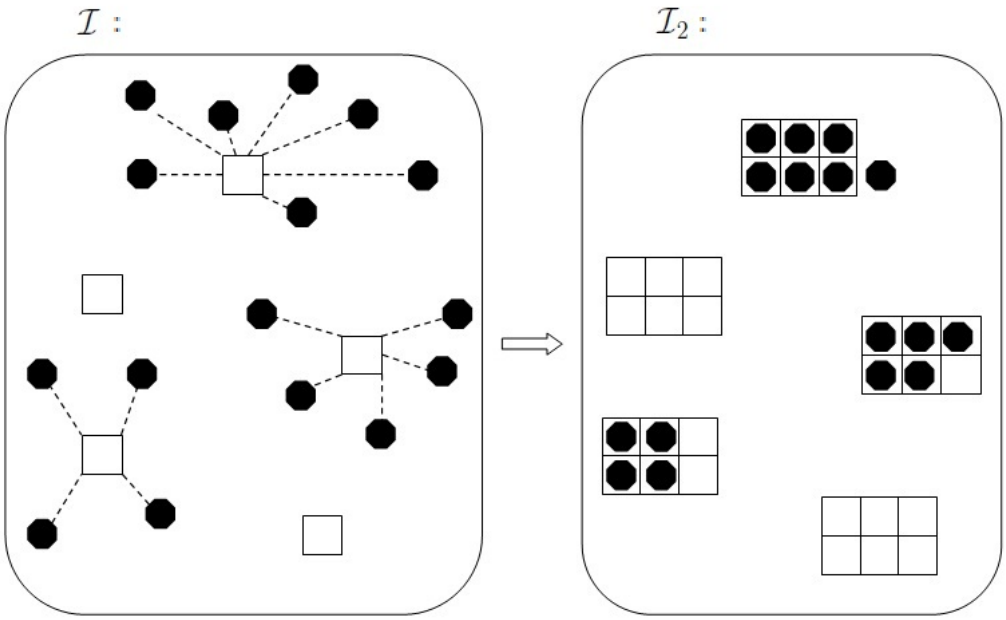

Figure 2.2: The octagons represent the clients, squares in instance $\mathcal{I}$ and the large rectangles in instance $\mathcal{I}_2$ represent the facilities, and $M = 6$. The dotted lines show the assignment of clients found by bicriteria algorithm for the original instance, for $\alpha = .65$.

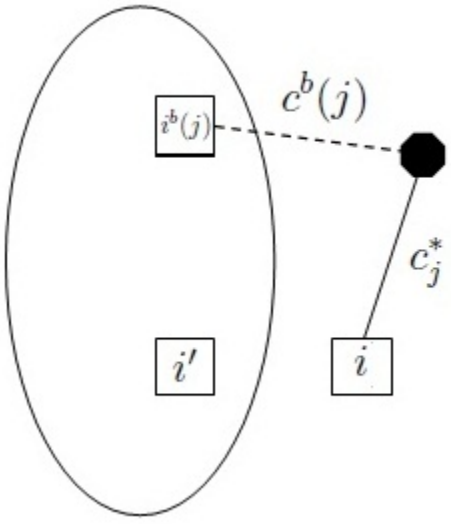

Figure 2.3: Triangle inequality in Lemma 2.3.2.

In the next section, we will describe how Svitkina's algorithm obtains a constant-factor approximation $\mathcal{I}_2$ solution. The next lemma shows how to construct a feasible solution to $\mathcal{I}$ using a feasible solution to $\mathcal{I}_2$ without incurring too much additional cost.

Theorem 2.3.3. *Let S_2 be a feasible solution to $\mathcal{I}_2$ with connection cost C_2. Then we can find a feasible solution to $\mathcal{I}$ which costs at most $C^b + F^b + C_2$. Thus, if S_2 is a β-approximation solution to $\mathcal{I}_2$, then, we can obtain a solution to $\mathcal{I}$ of the cost at most:*

$$(2h(\alpha) + 1)C^b + F^b + 2h(\alpha)C^*,$$

Proof. Consider a solution to $\mathcal{I}$ which opens the same set of facilities that are open in S_2, and assigns the clients as in S_2. Since the set of open facilities is subset of F^b, the facility cost is at most F^b. For each client j which is assigned to facility $i_2(j)$ in S_2, we have:

$$c(i_2(j), j) \leq c(i_2(j), i^b(j)) + c(i^b(j), j) = c_2(i_2(j), j) + c^b_j$$

If we sum this up for all clients, we get that the total connection cost is at most $C^b + C_2$. Thus, if S_2 is a β-approximation solution to $\mathcal{I}_2$, then $C_2 \leq \beta \cdot (C^b + C^*)$ and the result follows. □

2.4 Transformation to instance $\mathcal{I}_{\mathsf{CFL}}$

We recall that $\mathcal{I}_2$ has only points in $\mathcal{F}_2 \subseteq \mathcal{F}$ and there are $n_i \geq \alpha M$ co-located clients at each location i. The goal is to identify a subset of $\mathcal{F}_2$ to close such that transferring the clients at these locations to the remaining (open) facilities in $\mathcal{F}_2$ ensures that each such facility satisfies the lower bound constraint.

Since $\alpha \geq .5$, two facilities together have at least M clients. One initial idea is to find a matching between facilities and transfer all clients of these two facilities to one of the facilities. The following instance shows that the cost of solution found this way could be much larger than $OPT(\mathcal{I}_2)$. Consider an instance where we have $M - 1$ facilities located at unit distance from each other, and each of these facilities has $M - 1$ co-located clients. The optimal solution is to close an arbitrary facility, and transfer each of its clients to a different facility, which costs $M - 1$ in total. However, the solution found by a matching incurs cost $\frac{M}{2} \cdot (M - 1)$.

Svitkina shows that one can solve $\mathcal{I}_2$ by solving suitable CFL instance. The general idea is that a facility i that should be closed, corresponds to a *supply point* that has n_i units of supply to give out. On the other hand, a facility i' that remains open but has less than M clients corresponds to a *demand point* that requires $M - n_{i'}$ units of demand. Of course one does not know beforehand which facilities should be closed and which ones should remain open, but the reduction does not require this knowledge.

The CFL instance $\mathcal{I}_{\mathsf{CFL}}$ is defined as follows (see Figure 2.4). For each facility $i \in \mathcal{F}_2$ create a capacitated supply point with capacity M and opening cost $\delta \cdot \min\{n_i, M\} \cdot l(i)$, where $l(i) = \min_{i' \in \mathcal{F}_2, i \neq i'} c(i, i')$ and δ is a scaling parameter to be determined later. If $n_i < M$, we also create a demand point with $M - n_i$ amount of demand; if $n_i > M$ we create a supply point with capacity $n_i - M$ and zero opening cost. Distances in $\mathcal{I}_{\mathsf{CFL}}$ are the same as in $\mathcal{I}_2$.

We first bound the cost of the optimal solution to $\mathcal{I}_{\mathsf{CFL}}$ in terms of the optimal solution to $\mathcal{I}_2$:

Lemma 2.4.1. *There exists a solution to $\mathcal{I}_{\mathsf{CFL}}$ with facility cost bounded by δC_2^* and connection cost bounded by C_2^*.*

Proof. We transfer an optimal solution S_2^* of $\mathcal{I}_2$, to a feasible solution to $\mathcal{I}_{\mathsf{CFL}}$. We may assume, due to the triangle inequality, that if a facility i is open in S_2^*, then it serves all the clients co-located with it.

Solution S to $\mathcal{I}_{\mathsf{CFL}}$ is obtained in the following way (see Figure2.4). For each closed facility in S_2^*, open the corresponding supply point i with nonzero opening cost. We also select all the supply points with zero opening cost. For each selected supply point i, let it satisfy all the demands at location i (if there is any demand at location i, i.e., $n_i < M$). If k clients are assigned from i to a facility i' in S_2^* , we say k units of supply are sent from supply point i to i'. It is possible that the total supply sent to a demand point is larger than the actual demand at the demand point, but this only overestimates the connection cost incurred.

Now we need to show that the obtained solution is actually a feasible solution to $\mathcal{I}_{\mathsf{CFL}}$. We need to show all the demands are satisfied, and also none of the selected supply points exceeds its capacity. First, we show that all the demands in $\mathcal{I}_{\mathsf{CFL}}$ are satisfied. Note that we only have demands at supply points i where $n_i < M$. If a facility i is open in S_2^*, this means that at least $M - n_i$ clients of other facilities are transferred to i so in the constructed solution to $\mathcal{I}_{\mathsf{CFL}}$all the demand at location i are satisfied by supplies of other locations (like facilities on the left side of the bottom row in Figure2.4). If i is closed in S_2^*, then supply point i is opened in S, and it satisfies all the demand at its location (like facility i in Figure 2.4).

Second, we need to show that the capacity constraint is satisfied for all supply points. For a closed facility i, all n_i clients at this location should be assigned to other facilities. If $n_i \leq M$, the supply point i sends out n_i units of supply, and $M - n_i$ units of supply are used to satisfy the demand co-located at i. So overall such a facility uses M units of supply, which is equal to its capacity. If $n_i > M$, we open both supply points at i, which together have n_i units of supply, and we have no demands at this location. Since the two supply points send out n_i units of supply, they do not exceed their capacities. We also

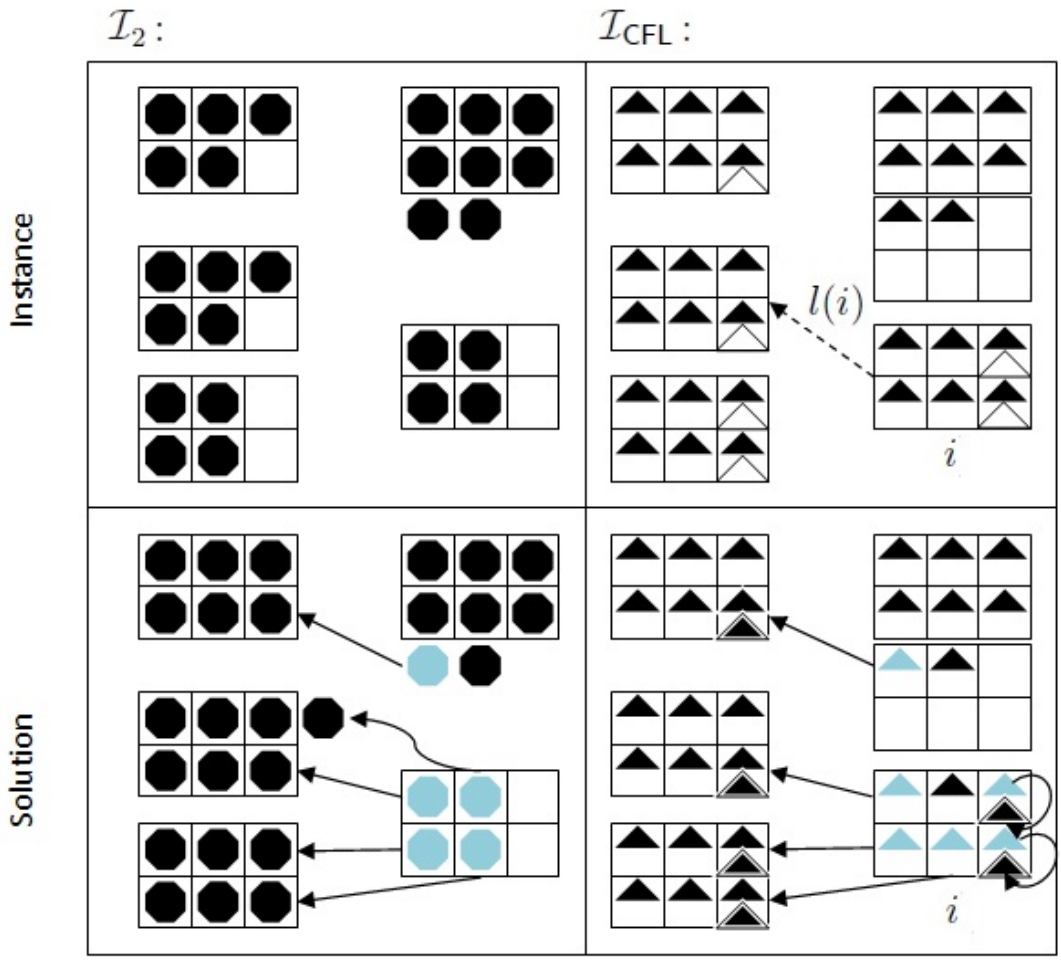

Figure 2.4: The top row shows the correspondence between the instances $\mathcal{I}_2$ and $\mathcal{I}_{\mathsf{CFL}}$ and the bottom row shows the correspondence between solutions to these instances. We have $M = 6$ and $\alpha = .6$. The octagons represent clients and large rectangles represent facilities in $\mathcal{F}_2$. The black triangles represent the capacity of the supply point and white triangles represent the demand at the co-located demand point. A closed facility i in the $\mathcal{I}_2$ solution correspond to an open supply point in $\mathcal{I}_{\mathsf{CFL}}$ solution. Three units of supply at i satisfy the demands of the other locations, and two units satisfy the demand located at i.

need to consider the case that a supply point i is opened in S, but corresponding facility is open in S_2^*. We only have this case for the supply points with zero selection cost. Since the facility i is open in S_2^* so at most $n_i - M$ of clients at this location are assigned to other facilities, which is equivalent to at most $n_i - M$ units of supply of supply point i being sent out to satisfy demand at other locations. So i does not exceed its total capacity.

We bound the cost of S as follows. The connection cost of S is at most C_2^* since each unit of supply transferred to a demand point corresponds to a reassigned client. Note that we only pay nonzero facility opening cost for supply points that correspond to facilities that are closed in S_2^*. All clients of such facilities should be reassigned to other facilities, so their connection cost is at least $n_i \cdot l(i)$. Since for each such a facility, we pay $\delta \min\{n_i, M\} l(i)$ as an opening cost so the overall facility cost is bounded by δC_2^* and result follows.

□

The next step is to solve $\mathcal{I}_{\mathsf{CFL}}$ and show how to transfer the obtained solution S to a feasible solution to $\mathcal{I}_2$. We can get a feasible solution to $\mathcal{I}_{\mathsf{CFL}}$by utilizing one of the constant-factor approximation algorithms for CFL. The best known factor for the general case is 5.83 [21]. Let γ denote the approximation guarantee of the algorithm we use for solving CFL. We get the following corollary using the results in lemma 2.4.1.

Corollary 2.4.2. *The cost of the solution S found for $\mathcal{I}_{\mathsf{CFL}}$ by γ-approximation algorithm is at most $(1+\delta)\gamma \cdot C_2^*$.*

Now we show how to transfer a near-optimal $\mathcal{I}_{\mathsf{CFL}}$-solution to a near-optimal $\mathcal{I}_2$-solution. We are given an $\mathcal{I}_{\mathsf{CFL}}$-solution S with facility cost (and set of open facilities) F^S and connection cost C^S. We may assume that:

- all zero-cost facilities are open in S.
- if S opens a supply point with non-zero opening cost located at some facility $i \in \mathcal{F}_2$ with $n_i > M$, then the demand assigned to the supply point at i with zero opening cost equals its capacity $n_i - M$, i.e., S never opens a supply point with non-zero facility opening cost unless all the capacity of a supply point with zero opening cost at the same location is used.
- for each $i \in \mathcal{F}_2$ with $n_i \leq M$, if the supply point at i is open then it serves the entire demand of the co-located demand point.

We briefly sketch parts of the procedure, omitting certain details since it is also described in Chapter 3 when we describe our procedure for solving $\mathcal{I}_2$. We reassign clients of S as follows. We transfer x clients from i to i' if x demand units of i' are satisfied by i where $i \neq i'$. Note that we can always perform this reassignment since the amount of supply i

sends out is at most n_i: if $n_i < M$, since i satisfies all demand at its location, it sends out at most n_i units of supply; if $n_i \geq M$, since the capacity constraint is satisfied, at most n_i units of supply are sent out from this location.

After doing the above procedure, each facility i that is closed in S has at least M clients. However, we may not yet have a feasible solution since a supply point i may send out less than n_i units of supply, in which case we may have some residual clients at i but their number may be less than M. So we need to reassign these residual clients. This issue also arises in our algorithm for solving $\mathcal{I}_2$, when we want to construct a feasible $\mathcal{I}_2$-solution, so to avoid redundancy, we describe this part in detail in Chapter 3 and just state the following result from [20] here.

Lemma 2.4.3. *[20] The cost of solution found by the algorithm for $\mathcal{I}_2$ is at most*

$$\frac{2\alpha}{2\alpha - 1} C^S + \frac{1}{\delta\alpha} F^S, \tag{2.3}$$

where C^S denotes the connection cost, and F^S denotes the facility opening cost of an $\mathcal{I}_{\mathsf{CFL}}$ solution S.

Combining the results of Theorem 2.2.4, Lemma 2.3.3, Corollary 2.4.2 and Lemma 2.4.3, we get the main theorem on cost of obtained solution for $\mathcal{I}$ in the algorithm :

Theorem 2.4.4. *[20] There is a constant-factor approximation algorithm for lower-bounded facility location problem.*

Proof. Combining the results of corollary 2.4.2 and lemma 2.4.3, we can get a solution to $\mathcal{I}_2$ which costs at most $\max(\frac{2\alpha}{2\alpha-1}, \frac{1}{\delta\alpha})(1+\delta) \cdot \gamma \cdot C_2^*$. Setting $\delta = \frac{2\alpha-1}{2\alpha^2}$, we get that the approximation ratio for solving $\mathcal{I}_2$ is $h(\alpha) = \gamma \cdot \frac{2\alpha}{2\alpha-1} \cdot (1 + \frac{2\alpha-1}{2\alpha^2})$. Using the results in lemma 2.3.3, now we only need to bound $F^b + (2h(\alpha)+1)C^b$. Using the results in Theorem 2.2.4, $F^b + C^b \leq \rho\frac{1+\alpha}{1-\alpha} \cdot OPT(\mathcal{I})$, so the overall bound on the cost of a solution to $\mathcal{I}$ is $(2h(\alpha)+1)(1+\rho\frac{1+\alpha}{1-\alpha}) + 2h(\alpha)$. We use the value $\alpha = .68$ and $\rho = 1.50$ which gives approximation guarantee of 550. □

Chapter 3

Our Improved Algorithm

In this chapter and next chapter (the CDUFL chapter), we describe our improved algorithm for LBFL and its analysis. We advance the state-of-the-art for LBFL significantly by devising an algorithm with approximation ratio 82.5. Initially, we tried to develop a local-search algorithm for LBFL, but we found bad locality gaps for various local-search algorithms for LBFL (see Appendix B). (Recall that the locality gap of a local-search algorithm is the maximum ratio of the cost of a locally optimum solution to the cost of a globally optimum solution.)

We begin by presenting the sketch of our algorithm in section 3.1. As we mentioned in the Introduction chapter, our high-level approach is similar to Svitkina's algorithm [20]; we first transform our instance to a more structured LBFL instance $\mathcal{I}_2$ such that (i) all clients are aggregated at a subset $\mathcal{F}_2 \subseteq \mathcal{F}$ of the facilities with each facility $i \in \mathcal{F}_2$ having $n_i \geq \alpha M$ co-located clients; (ii) all facilities in $\mathcal{F}_2$ have zero opening costs; and (iii) near-optimal solutions to $\mathcal{I}_2$ translate to near-optimal solutions to $\mathcal{I}$ (and vice versa), then we solve $\mathcal{I}_2$. But our algorithm differs from Svitkina's algorithm in (a) How we solve $\mathcal{I}_2$ and, (b) how we obtain $\mathcal{I}_2$. Our key insight is that instead of solving $\mathcal{I}_2$ by reducing it to a CFL instance as in [20], one can solve LBFL instance $\mathcal{I}_2$ by reducing it to an instance of *capacity-discounted UFL* (CDUFL), which is a special case of CFL, and a generalization of UFL that we introduce.

In section 3.2, we will describe our approach for obtaining a bicriteria solution to LBFL, which is then used to construct $\mathcal{I}_2$ instance. We describe our our local-search algorithm for CDUFL, and the proof of its approximation guarantee to Chapter 4. In Section 3.3, we use our results on the approximation ratio of CDUFL and show how this leads to an improved approximation to $\mathcal{I}_2$ and hence for $\mathcal{I}$. We describe this reduction, that is, how we construct the CDUFL instance $\widehat{\mathcal{I}}$, how to transfer an $\widehat{\mathcal{I}}$-solution to an $\mathcal{I}_2$-solution. Finally we wrap up our improvements in our main theroem stated in section 3.4.

3.1 Sketch of the algorithm

In this section, we briefly present our algorithm (see Algorithm 3.1).

Algorithm 3.1 Our Algorithm for LBFL

1: Get a bicriteria solution for LBFL instance $\mathcal{I}$.
2: Use the bicriteria solution to modify LBFL instance $\mathcal{I}$, and get new instance $\mathcal{I}_2$.
3: Construct CDFL instance $\widehat{\mathcal{I}}$ from $\mathcal{I}_2$.
4: Solve $\widehat{\mathcal{I}}$.
5: Transfer the $\widehat{\mathcal{I}}$-solution obtained in step 4 to an $\mathcal{I}_2$-solution.
6: Transfer the $\mathcal{I}_2$-solution obtained in step 5 to an $\mathcal{I}$-solution.

The first step of our algorithm is to obtain a bicriteria solution. Our approach for obtaining this solution varies from [7, 9], which show that given an LBFL instance, if one modifies the facility costs in a certain way, solves the resulting UFL instance, and post-processes the solution so that no single facility deletion move improves the solution cost, then one obtains a bicriteria solution to the LBFL instance. We define slightly distinct UFL instance where the facility opening costs are defined in a subtly different way. The opening cost of facility i in $\widehat{\mathcal{I}}$ is now set to $\hat{f}_i = f_i + 2\alpha M R_i(\alpha)$ where $R_i(\alpha)$ is the distance between i and the $\lceil \alpha M \rceil$-closest client to i. As in [7, 9], one can show a delete-optimal solution is a bicriteria solution. Eventually, the overall cost of the solution to $\mathcal{I}$ includes various $R_i(\alpha)$ terms. Instead of plugging the (weak) bound $MR_i(\alpha) \leq \frac{\sum_{j \in \mathcal{D}(i)} c(i,j)}{1-\alpha}$ in the analysis, we choose α from a suitable distribution and leverage the fact that $M \int_0^1 R_i(\alpha) = \sum_{j \in \mathcal{D}(i)} c(i,j)$ for tighter analysis. Also, we use the local-search algorithm of [1, 4] to solve the resulting UFL instance. The obtained UFL solution is already post-processed, and gives asymmetric bounds on the facility-opening and assignment costs. This property also yields a tighter analysis since we can bound the the term $2(h(\alpha)+1)C^b + F^b$ in Theorem 2.3.3 more tightly (as compared to $2(h(\alpha)+1)(C^b + F^b)$).

We construct $\mathcal{I}_2$ from the bicriteria solution as in [20] (See Section 2.3). Our chief algorithmic novelty is that one can solve $\mathcal{I}_2$ by reducing it to CDUFL. CDUFL is a special case of CFL, where facilities are either uncapacitated, or capacitated with zero facility-opening cost. In step 3, the CDUFL instance $\widehat{\mathcal{I}}$ is constructed in a way similar to the way the CFL instance is constructed in section 2.4, except that supply points with nonzero opening cost are now uncapacitated. In the fourth step, we solve $\widehat{\mathcal{I}}$ with the local-search algorithm that we devise (See Chapter 4). Our approximation ratio for CDUFL is better than the best known approximation ratio for CFL. The next step is to transfer this solution to $\mathcal{I}_2$-solution without incurring too much cost. Here, in addition to the problem faced in [20], where supply point may send out less supply than the number of clients at its corresponding facility in $\mathcal{I}_2$, since we now have uncapacitated facilities, a supply point i

may send out supply more than the number of clients at its corresponding facility in $\mathcal{I}_2$. We resolve this problem by opening subset of supply points at locations of demand points served by i. We show that the overall incurred cost for transferring a feasible $\widehat{\mathcal{I}}$-solution to a feasible $\mathcal{I}_2$-solution is small. Finally, we transfer the obtained $\mathcal{I}_2$-solution to a feasible solution to $\mathcal{I}$ as described in Chapter 2.

3.2 Bicriteria Algorithm

We now present our method for obtaining a bicriteria solution to LBFL. Consider the UFL instance $\widehat{\mathcal{I}}$ with the same set of facilities and clients, and the same assignment costs as $\mathcal{I}$. The opening cost of facility i is set to $\hat{f}_i = f_i + 2\alpha M R_i(\alpha)$ where $R_i(\alpha)$ denote the distance between i and the $\lceil \alpha M \rceil$-closest client to i; that is, if $\mathcal{D}(i) = \{j_1, \ldots, j_M\}$, where $c(i, j_1) \leq \ldots \leq c(i, j_M)$, then $R_i(\alpha) = c(i, j_{\lceil \alpha M \rceil})$ (for $0 < \alpha \leq 1$). Let $R^*(\alpha) = \sum_{i \in F^*} R_i(\alpha)$. Observe that each $R_i(\alpha)$ is an increasing function of α, $M \int_0^1 R_i(\alpha) d\alpha = \sum_{j \in \mathcal{D}(i)} c(i, j)$, and $R_i(\alpha) \leq \frac{\sum_{j \in \mathcal{D}(i)} c(i,j)}{M - \lceil \alpha M \rceil + 1} \leq \frac{\sum_{j \in \mathcal{D}(i)} c(i,j)}{M(1-\alpha)}$. Hence, $R^*(\alpha)$ is an increasing function of α. The same argument as in Lemma 2.2.1 yields the following lemma:

Lemma 3.2.1. *There exists a feasible solution to $\widehat{\mathcal{I}}$ with facility cost bounded by $F^* + 2\alpha M R^*(\alpha)$ and connection cost bounded by C^*.*

We use the local search by [1,4] to solve $\widehat{\mathcal{I}}$. [1] argues that to get a polynomial running time, the only local moves that should be considered are those that improve the cost by some factor of the cost of current solution. However, one can execute all delete moves that improves the cost of the solution upon termination and there by obtain an "approximate" local optimum that is "delete-optimal"[1] . The same arguments as in Lemma 2.2.3 show that in a delete-optimal solution, each facility serves at least αM clients. So our solution does not need the post-processing step as in [7, 9] (and hence [20]). For simplicity, we assume that we obtain a local optimum; standard arguments in [1, 4] show that dropping this assumption increases the cost by at most a $(1 + \epsilon)$ factor.Thus combining the results of [1, 4] and Lemma 3.2.1 yeilds the following :

Lemma 3.2.2. *For any parameter $\gamma > 0$, executing the local-search algorithm in [1, 4] on $\widehat{\mathcal{I}}$ returns a solution with facility cost F^b and assignment cost C^b satisfying $F^b \leq \big(F^* + 2\alpha M R^*(\alpha)\big) + 2C^*/\gamma, C^b \leq \gamma\big(F^* + 2\alpha M R^*(\alpha)\big) + C^*$, where each open facility serves at least αM clients.*

[1] We call a solution with no improving delete move a *delete optimal* solution

3.3 Solving $\mathcal{I}_2$

In this section, we describe our algorithm for solving $\mathcal{I}_2$. As we mentioned earlier, our key insight is that one can solve $\mathcal{I}_2$ by reducing it to CDUFL. We present our algorithm for solving CDUFL in chapter 4. Here, we first describe how the CDUFL instance space,$\widehat{\mathcal{I}}$ is constructed. Then, in Section 3.3.1, we show how to transfer an $\widehat{\mathcal{I}}$-solution to an $\mathcal{I}_2$-solution, and in Section 3.3.2, we bound the cost of the obtained solution for $\mathcal{I}_2$.

Following [20], to avoid confusion, we refer to the facilities and clients in the CDUFL instance as supply points and demand points respectively. The CDUFL instance $\widehat{\mathcal{I}}$ is constructed from $\mathcal{I}_2$ in a way similar to the way in which $\mathcal{I}_{\mathsf{CFL}}$ is constructed (in Chapter 2), except that the supply points with non-zero facility costs are now uncapacitated. More precisely, for each facility i put an uncapacitated supply point with opening cost $\delta l(i)\min(n_i, M)$ where $l(i)$ is the distance between i and the closest facility in $\mathcal{F}_2 \setminus \{i\}$ and δ is a scaling parameter. If $n_i > M$ also put a supply point with capacity $n_i - M$ and zero opening cost; If $n_i \leq M$ put a demand point with demand $M - n_i$. In $\widehat{\mathcal{I}}$, let $\widehat{\mathcal{F}}^c$ denote the set of capacitated supply points and let $\widehat{\mathcal{F}}^u$ denote the set of uncapacitated supply points. Arguing as Lemma 2.4.1 leads to the following lemma:

Lemma 3.3.1. *There exists a solution to $\widehat{\mathcal{I}}$ with facility cost at most δC_2^* and connection cost bounded by C_2^*.*

We will prove the following theorem in Chapter 4 on the approximation ratio of our local-search algorithm for CDUFL:

Theorem 4.3.5. Given any CDUFL instance, one can efficiently compute a solution with facility-opening cost $\widehat{F} \leq (\widehat{F}^{sol} + 2\widehat{C}^{sol})(1+\epsilon)$ and connection cost $\widehat{C} \leq (\widehat{F}^{sol} + \widehat{C}^{sol})(1+\epsilon)$, where $\widehat{F}^{sol}$ and $\widehat{C}^{sol}$ are the facility and connection costs of an arbitrary solution to the CDUFL instance, and ϵ is a factor that effects the running time of the algorithm.

As in [1,4], for notational simplicity, we ignore the ϵ-terms in the sequel since the effect of all such ϵ-terms can be captured by incurring an additional $(1+\epsilon)$-multiplicative factor in the approximation. Combining the results in the above theorem and Lemma 3.3.1, we arrive at the following theorem:

Theorem 3.3.2. *One can compute a solution to $\widehat{\mathcal{I}}$ with facility cost $\widehat{F} \leq (2+\delta)C_2^*$ and assignment cost $\widehat{C} \leq (1+\delta)C_2^*$.*

3.3.1 Mapping CDUFL solution to $\mathcal{I}_2$ solution

Suppose that are given a solution $\widehat{S}$ to $\widehat{\mathcal{I}}$ with facility cost $\widehat{F}^S$ and assignment cost $\widehat{C}^S$. Again, we abuse notation and use $\widehat{F}^S$ to also denote the set of supply points that are opened in $\widehat{S}$. We may assume that: (i) all capacitated supply points are open (i.e., $\widehat{\mathcal{F}}^c \subseteq \widehat{F}^S$); (ii) if $\widehat{S}$ opens an uncapacitated supply point located at some $i \in \mathcal{F}_2$ with $n_i > M$, then the demand assigned to the capacitated supply point at i equals its capacity $n_i - M$; (iii) for each $i \in \mathcal{F}_2$ with $n_i \leq M$, if the supply point at i is open then it serves the entire demand of the co-located demand point; and (iv) at most one *uncapacitated* supply point serves, maybe partially, the demand of any demand point; we say that this uncapacitated supply point satisfies the demand point.

Let N_i, initialized to n_i, keep track of the number of clients at location $i \in \mathcal{F}_2$. During the process, we *always* keep N_i updated. Our goal is to reassign clients (using $\widehat{S}$ as a template) so that at the end we have $N_i = 0$ or $N_i \geq M$ for each $i \in \mathcal{F}_2$. Observe that once we have determined which facilities in $\mathcal{F}_2$ will have $N_i \geq M$ (i.e., the facilities to open in the $\mathcal{I}_2$-solution), one can find the best way of (re)assigning clients by solving a min-cost flow problem. However, for purposes of analysis, it will often be convenient to explicitly specify a (possibly suboptimal) reassignment. We reassign clients in three phases.

A1. **(Removing capacitated supply points)** For each location $i \in \mathcal{F}_2$ with $n_i > M$, if the capacitated supply point i^1 at i supplies x units to the demand point at location i', we transfer x clients from location i to i'. Now if i^1 has $y > 0$ leftover units of capacity in $\widehat{S}$, then we "move" y clients to the uncapacitated supply point at i, i^2 (which must not be open in $\widehat{S}$ due to property (ii)). Note that this reassignment effectively gets rid of all capacitated supply points. Thus, there is now exactly one uncapacitated supply point and at most one demand point at each location $i \in \mathcal{F}_2$; we refer to these simply as supply point i and demand point i below.

Let X_i be the total demand from other locations assigned to supply point i. Let $\mathcal{F}^R = \{i \in \mathcal{F}_2 : N_i \geq X_i > 0\}$, $\mathcal{F}^G = \{i \in \mathcal{F}_2 : N_i < X_i\}$, and $\mathcal{F}^B = \{i \in \mathcal{F}_2 : X_i = 0\}$, which is the set of supply points that are not opened in $\widehat{S}$. Note that $N_i \geq \min\{n_i, M\} \geq \alpha M$ for all $i \in \mathcal{F}_2$, and $N_i = \min\{n_i, M\}$ for all $i \in \mathcal{F}^R \cup \mathcal{F}^G$ (because of properties (ii) and (iii) above).

A2. **(Taking care of $\mathcal{F}^R$ and demand points satisfied by $\mathcal{F}^R$)** For each $i \in \mathcal{F}^R$, if i supplies x units to demand point i', we move x clients from i to i'.

We now have $N_i = \min(n_i, M) - X_i$ residual clients at each $i \in \mathcal{F}^R$, which we must reduce to 0, or increase to at least M. We follow the same procedure as in [20], which we sketch below.

For each $i \in \mathcal{F}^R$, we include an edge (i, i') where $i' \in \mathcal{F}_2$ is the facility nearest to i (recall that $c(i, i') = l(i)$). We use an arbitrary but fixed tie-breaking rule here, so we have two types of components in G:

- (i) A tree rooted at facility $i \in \mathcal{F}_2 \setminus \mathcal{F}^R$ where all the edges of the tree are directed toward the root.
- (ii)A tree rooted at 2-cycle $(r, r'), (r', r)$, where $r, r' \in \mathcal{F}^R$ and all edges of the tree are directed toward this 2-cycle.

Note that a facility in $\mathcal{F}_2 \setminus \mathcal{F}^R$ is either a root of a type (i) tree, or a singleton since it does not have any out-edges. Essentially, we move the residual clients of supply points in a component bottom-up from the leaves up to the root. For each internal node i (i.e., the node is neither the root, nor one of the nodes of the 2-cycle): If i has at least M clients cut off the edge going up to its parent i'; otherwise transfer all clients at i to i'. And for 2-cycle $(r, r'), (r', r)$: if $N_r + N_{r'} \geq M$ and $\min(N_r, N_{r'}) < M$, say $N_{r'} < M$, transfer all clients of r' to r; if $N_r + N_{r'} < M$, transfer all clients at r and r' to the demand point i in $\mathcal{F}^B$ that is nearest to $\{r, r'\}$; otherwise do nothing. After each of such above transfers, we update N_is.

At this point, the only facilities that may have $0 < N_i < M$ correspond to eithe supply points in $\mathcal{F}^G$ or demand points satisfied by a supply point in $\mathcal{F}^G$. All other facilities have $N_i = 0$ or $N_i > M$.

A3. **(Taking care of $\mathcal{F}^G$ and demand points satisfied by $\mathcal{F}^G$)** For $i \in \mathcal{F}^G$, let $D(i)$ be the set of demand points $j \in \mathcal{F}_2,\ j \neq i$ that are satisfied by i, and let $D'(i) = \{j \in D(i) : N_j < M\}$. Note that $D(i) \subseteq \mathcal{F}^B$. Phase A2 only increases N_j for all j in $\mathcal{F}^B \cup \mathcal{F}^G$, so $N_j \geq \alpha M$ for all $j \in \mathcal{F}^G \cup \left(\bigcup_{i \in \mathcal{F}^G} D(i)\right)$.

Fix $i \in \mathcal{F}^G$. We reassign clients so that $N_j = 0$ or $N_j \geq M$ for all $j \in \{i\} \cup D'(i)$, without decreasing N_j for $j \in D(i) \setminus D'(i)$. Applying this procedure to all supply points in $\mathcal{F}^G$ will complete our task. Define $Y_j = M - N_j$ (which is at most $M - n_j$) for $j \in D'(i)$. First, suppose $\sum_{j \in D'(i)} Y_j \leq N_i$. For each $j \in D'(i)$, if i supplies x units to j, we transfer x clients from i to j. If i is now left with less than M residual clients, we move these residual clients to the location in $D(i)$ nearest to i.

Now suppose $\sum_{j \in D'(i)} Y_j > N_i$. Let $i_0 = i$, and $D'(i) = \{i_1, \ldots, i_t\}$, where $c_{i_1 i} \leq \ldots \leq c_{i_t i}$. Let $\ell = t - \left\lfloor \frac{\sum_{r=0}^{t} N_{i_r}}{M} \right\rfloor = \left\lceil \frac{\sum_{r=1}^{t} Y_{i_r} - N_{i_0}}{M} \right\rceil$, so $\ell \geq 1$ (and $\ell < t$ since $N_{i_0} + N_{i_1} \geq M$). We first transfer Y_{i_q} clients to each $i_q,\ q = \ell + 1, \ldots, t$ from the locations $i_\ell, \ldots, i_1, i_0$, where we transfer all clients of i_r (where $1 \leq r \leq \ell$) before moving to i_{r-1}. (This is always possible since $(t - \ell)M \leq \sum_{r=0}^{t} N_{i_r}$.) We argue that this leaves at most M residual clients in $\{i_0\} \cup D'(i)$, which are all concentrated at i_0 and i_1, with i_1 having at most $(1 - \alpha)M$ residual clients. We transfer these residual clients to $i_{\ell+1}$.

3.3.2 Bounding the cost of the constructed $\mathcal{I}_2$-solution

In this section, we bound the cost of the feasible solution to $\mathcal{I}_2$obtained above. We first bound the cost of this solution in terms of the $\widehat{\mathcal{I}}$-solution $\widehat{S}$ that we started from. Then, we determine our approximation ratio for solving $\mathcal{I}_2$.

Lemma 3.3.3. *The above algorithm returns an $\mathcal{I}_2$-solution of cost at most $\frac{\widehat{F}^S}{\delta\alpha} + \widehat{C}^S\big(\frac{1}{\alpha} + \frac{2\alpha}{2\alpha-1}\big)$ where $\widehat{S}$ is a feasible $\widehat{\mathcal{I}}$-solution.*

Proof. Let S_2 denote the solution computed for $\mathcal{I}_2$. For a supply point i opened in $\widehat{S}$, we use $\widehat{C}_i^S$ to denote the cost incurred in supplying demand from i to the demand points satisfied by i; so $\widehat{C}^S = \sum_{i\in F^S} \widehat{C}_i^S$. At various steps, we transfer clients between locations according to the assignment in the CDUFL solution S, and the cost incurred in this reassignment can be charged against the $\widehat{C}_i^S$s of the appropriate supply points. So the cost of phase A1 is $\sum_{i\in\widehat{\mathcal{F}}^c} \widehat{C}_i^S$, and the cost of the first step of phase A2 is $\sum_{i\in\mathcal{F}^R} \widehat{C}_i^S$.

We can bound the remaining cost of A2, by $\widehat{F}^S/\delta\alpha + \big(\sum_{i\in\mathcal{F}^R} \widehat{C}_i^S\big)/(2\alpha-1)$. For the out-edge of facility i in its corresponding tree, we pay at most $Ml(i)$ for (possibly) transferring (at most M) clients along this edge. Since i is an open supply point in $\widehat{S}$, it pays at least $\delta l(i)\alpha M$ for its opening cost. So the cost of transferring clients through edges of tree can be bounded by $\frac{\widehat{F}^S}{\delta\alpha}$. We also need to bound the cost of transferring clients from 2-cycles $(r,r'),(r',r)$to a demand point in $\mathcal{F}^B$. We incur this cost in case that $N_r + N_{r'} < M$. This means that at least $(2\alpha-1)M < n_r + n_{r'} - (N_r + N_{r'})$ supplies of these location were sent to the other demand points. Let demand point i in $\mathcal{F}^B$ be the location nearest to $\{r,r'\}$. So the cost transferring the residual clients of these two facilities to i is at most $M\cdot\frac{\widehat{C}_r^S+\widehat{C}_{r'}^S}{X_r+X_{r'}} \le \big(\widehat{C}_r^S + \widehat{C}_{r'}^S\big)/(2\alpha-1)$. Summing up this for all 2-cycles, we can bound the cost by $\sum_{i\in\mathcal{F}^R} \widehat{C}_i^S/(2\alpha-1)$.

Finally, consider phase A3 and some $i\in\mathcal{F}^G$. If $\sum_{j\in D'(i)} Y_j \le N_i$, then the cost incurred is at most $\widehat{C}_i^S + M\cdot\frac{\widehat{C}_i^S}{X_i} \le \widehat{C}_i^S\big(1+\frac{1}{\alpha}\big)$. Now consider the case $\sum_{j\in D'(i)} Y_j > N_i$. For any $i_q \in \{i_{\ell+1},\ldots,i_t\}$ and any $i_r\in\{i_0,\ldots,i_\ell\}$, we have $c(i_r,i_q) \le 2c(i,i_q)$, so the cost of transferring $Y_{i_q} \le M - n_{i_q}$ clients to each i_q, $q=\ell+1,\ldots,t$ is at most $2\widehat{C}_i^S$. Observe that $(t-\ell+1)M > \sum_{r=0}^t N_{i_r}$, i.e., $M + \sum_{q=\ell+1}^t Y_{i_r} > \sum_{r=0}^{\ell} N_{i_r}$, so after this reassignment, there are less than M residual clients in $i_0,\ldots,i_\ell$. By our order of transferring clients, all these residual clients are at i_0,i_1 (otherwise we would have at least $N_{i_0}+N_{i_1}\ge M$ residual clients) with at most $M - N_{i_0} \le (1-\alpha)M$ of them located at i_1. The cost of reassigning these residual clients is at most $(1-\alpha)Mc(i,i_1) + Mc(i,i_{\ell+1}) \le (1-\alpha)M\cdot\frac{\widehat{C}_i^S}{\sum_{r=1}^t (M-n_{i_r})} + M\cdot\frac{\widehat{C}_i^S}{\sum_{r=\ell+1}^t (M-n_{i_r})}$, which is at most $\le \widehat{C}_i^S\big(\frac{1-\alpha}{\alpha} + \frac{1}{2\alpha-1}\big)$, since $M - n_j \ge Y_j$ for all $j\in D'(i)$.

Thus, the cost of S_2 is at most

$$\frac{\widehat{F}^S}{\delta\alpha} + \sum_{i\in\widehat{\mathcal{F}}^c} \widehat{C}_i^S + \sum_{i\in\mathcal{F}^R} \widehat{C}_i^S \cdot \Big(1+\tfrac{1}{2\alpha-1}\Big) + \sum_{i\in\mathcal{F}^G} \widehat{C}_i^S \cdot \max\Big\{1+\tfrac{1}{\alpha}, 2+\tfrac{1-\alpha}{\alpha}+\tfrac{1}{2\alpha-1}\Big\}$$
$$\leq \frac{\widehat{F}^S}{\delta\alpha} + \widehat{C}^S\Big(\tfrac{1}{\alpha}+\tfrac{2\alpha}{2\alpha-1}\Big).$$

□

Combining the results in Theorem 3.3.2 and above lemma yields the following theorem on the approximation bound of our algorithm for solving $\mathcal{I}_2$:

Theorem 3.3.4. *For any $\alpha \geq 0.5$, there is a $g(\alpha)$-approximation for $\mathcal{I}_2(\alpha)$, where $g(\alpha) = \frac{2}{\alpha} + \frac{2\alpha}{2\alpha-1} + 2\sqrt{\frac{2}{\alpha^2} + \frac{4}{2\alpha-1}}$.*

Proof. Letting $\widehat{S}$ be the solution given by Theorem 3.3.2, the cost of S_2 is at most

$$\big(\frac{2}{\delta\alpha} + \frac{1}{\alpha} + (1+\delta)(\frac{1}{\alpha} + \frac{2\alpha}{2\alpha-1})\big)C_2^*.$$

Setting $\delta = \sqrt{\frac{2/\alpha}{1/\alpha+(2\alpha)/(2\alpha-1)}}$ yields the bound in the theorem. □

Remark Our $g(\alpha)$-approximation ratio for $\mathcal{I}_2(\alpha)$ improves upon the approximation obtained in [20] by a factor of roughly 2 *for all* α. Thus, plugging in our algorithm for solving $\mathcal{I}_2$ in the LBFL-algorithm in [20] (and choosing a suitable α), already yields an improved approximation factor of 264 for LBFL.

3.4 Combining improvements

In this section, we bound the cost of solution obtained by our algorithm. The overall bound that we obtain includes certain $R^*(\alpha)$ terms. We will argue how choosing a random variable from suitable distribution will lead to tighter analysis. This can be easily de-randomized since we have only M choices for α. Combining the results of Theorem 2.3.3 and Lemma 3.2.2 leads to the following theorem on the approximibility of LBFL.

Theorem 3.4.1. *For any parameter $\alpha \in (0.5, 1]$, we can compute efficiently a solution to $\mathcal{I}$ of cost at most*

$$F^*\,(4.125) + C^*\big(5.28h(\alpha) + 1.64\big) + 8.25\alpha M R^*(\alpha)$$

where $h(\alpha) = \frac{2}{\alpha} + \frac{2\alpha}{2\alpha-1} + 2\sqrt{\frac{6}{2\alpha-1}}$. Thus, choosing $\alpha \in [0.67, 1]$ randomly according to the density function $p(x) = \frac{1}{\ln(1/0.67)x}$ concentrated on $[0.67, 1]$, yields a solution of cost at most $83 \cdot OPT(\mathcal{I})$.

Proof. Note that our approximation factor for solving $\mathcal{I}_2$ $g(\alpha)$ is smaller than $h(\alpha)$ for all $\alpha \in (0,1)$; we use this upper bound throughout below. Combining the bounds in Lemma 3.2.2, and Theorems 2.3.3 and 3.3.4, we obtain a solution to $\mathcal{I}$ of cost at most $F^b + \big(2h(\alpha)+1\big)C^b + 2h(\alpha)C^*$

$$\begin{aligned} &\leq F^* + 2\alpha MR^*(\alpha) + \frac{2C^*}{\gamma} + \big(2h(\alpha)+1\big)\gamma\Big(F^* + 2\alpha MR^*(\alpha)\Big) + \big(4h(\alpha)+1\big)C^* \\ &\leq F^*\Big(1+\gamma\big(2h(\alpha)+1\big)\Big) + C^*\Big(4h(\alpha)+1+\tfrac{2}{\gamma}\Big) + 2\gamma\alpha MR^*(\alpha)\big(2h(\alpha)+1\big) + 2\alpha MR^*(\alpha). \end{aligned}$$

Setting $\gamma = 3.125/(2h(\alpha)+1)$ yields the expression in the theorem statement.

We bound the expected cost incurred when one chooses α randomly according to the stated density function as follows :

$$\begin{aligned} \mathrm{E}_\alpha\left[h(\alpha)\right] &= c_2(\beta) := \Big[\frac{2}{\beta} - 2 + 4\sqrt{6}\big(\pi/4 - \tan^{-1}(\sqrt{2\beta-1})\big) + \ln\Big(\frac{1}{2\beta-1}\Big)\Big]/\ln(1/\beta), \\ \mathrm{E}_\alpha\left[\alpha MR^*(\alpha)\right] &= M\Big(\int_\beta^1 R^*(x)dx\Big)/\ln(1/\beta) \leq C^*/\ln(1/\beta). \end{aligned}$$

Plugging in these bounds, we get that the total cost is at most

$$F^*(4.125) + C^*\Big(5.28c_2(\beta) + 1.64 + 8.25/\ln(1/\beta)\Big) \leq 83(F^*+C^*).$$

□

Chapter 4

Capacity Discounted Facility Location Problem

In this chapter, we focus on the capacity-discounted UFL (CDUFL) problem and describe the algorithm we developed for it. CDUFL is a special version of CFL, where facilities are either uncapacitated, or capacitated with zero facility cost. We introduced this problem as we realized that using it as a subroutine instead of CFL when solving the LBFL instance $\mathcal{I}_2$ yields a better better approximation guarantee.

Circumventing the difficulty that CDUFL inherits from CFL with respect to LP-based approximation algorithms (see Appendix C), we give a simple local-search algorithm based on add, delete ,and swap moves (Section 4.2). Our analysis is inspired by the analysis of the algorithm in [1], which uses the same moves for UFL, but the presence of capacitated facilities calls for the use of other techniques for analysis as well. Surprisingly the bounds that we get for facility opening cost and connection cost of our solution are the same as the bounds in [1] (Section 4.3).

The locality gap of a local-search algorithm is the maximum ratio of the cost of a global optimum to the cost of a local optimum. For UFL, Arya et al. [1] argued that any procedure that permits add, delete ,andswap has a locality gap of 3. Since our local-search algorithm is based on the same moves for CDUFL which generalizes UFL problem, the the example in [1] also shows that the locality gap of our algorithm is at least 3. We include this example in Section4.4 for completeness.

We use the standard scaling technique in [4] to improve our approximation guarantee to $1 + \sqrt{2}$. As in the case of various other local-search algorithms for UFL, we can bound the facility cost and the connection cost of our solution in terms of the facility cost and the connection cost of *any* feasible solution to CDUFL. This property has a significant role in giving tighter bounds when CDUFL is used as subroutine for solving other problems (as we use it here for solving LBFL).

4.1 Problem Definition

In the capacity-discounted UFL problem, we are given a set of clients $\widehat{\mathcal{D}}$, a set of facilities $\widehat{\mathcal{F}}$, which is the union of two disjoint set $\widehat{\mathcal{F}}^u$ and $\widehat{\mathcal{F}}^c$, non-negative *facility opening cost* $\hat{f}_i$ for each $i \in \widehat{\mathcal{F}}^u$ ($\hat{f}_i = 0$ for each facility $i \in \widehat{\mathcal{F}}^c$), positive (finite) capacity u_i for each $i \in \widehat{\mathcal{F}}^c$, and a distance metric $\hat{c}(i,j)$ on the set $\widehat{\mathcal{F}} \cup \widehat{\mathcal{D}}$. Here $\hat{c}(i,j)$ denotes the cost of assigning a demand at location j to a facility i. In certain settings, a client j may have non-unit demand d_j at its location, in which case the cost of assigning this demand to an open facility i is $d_j\hat{c}(i,j)$.

A feasible solution S specifies the set of open facilities $\widehat{F}^S \subseteq \widehat{\mathcal{F}}$ and the assignment of each client j to an open facility such that the capacity constraints for facilities in $\widehat{\mathcal{F}}^c$ are satisfied. For a feasible CDUFL solution S, let $\hat{c}_j^S$ denote the connection cost of client j and $\widehat{\mathcal{D}}_S(i)$ denote the set of clients assigned to facility i. Our goal is to find feasible solution S that minimizes :

$$\sum_{i \in \widehat{F}^S} \hat{f}_i + \sum_j \hat{c}_j^S,$$

subject to $|\widehat{\mathcal{D}}_S(i)| \leq u_i$ for all $i \in \widehat{\mathcal{F}}^c$. We sometimes abuse the notation and use $\widehat{F}^S$ to also denote the facility cost of open facilities. Defining $\widehat{C}^S = \sum_j \hat{c}_j^S$, we can say we are looking for solution S that minimizes $\widehat{F}^S + \widehat{C}^S$.

4.2 A local search algorithm

Since we can find the best assignment of the clients to open facilities by solving a suitable network flow problem, we focus on determining the set of facilities to open. Our local-search algorithm consists of three moves : add(i'), delete(i), and swap(i, i'), which respectively open a not-currently-open facility i', close an open facility i, and open a not-currently-open facility i', and close an open facility i. Note that all (local-search) algorithms for CFL use moves more complicated than above for non-uniform capacities.

To guarantee polynomial running-time for our local-search algorithm, we only consider a move in a local step if it significantly improves the cost of the solution. We call such a local move an admissible move. More formally, let $cost(S)$ denote the total cost of a solution S and $op(S)$ denote the solution obtained after applying local move $op \in \{\text{add}, \text{delete}, \text{swap}\}$. We consider an operation op, if $cost(op(S)) \leq (1 - \frac{\epsilon}{N})cost(S)$ where ϵ is a constant factor and N is a suitable integer which is polynomial in the size of the input. Thus, the cost of the current solution is improved in each local step by at least a factor of $\frac{\epsilon}{N}$. Let $\widehat{S}_0$ be an initial solution to the CDUFL problem and $\widehat{S}^*$ be an optimal solution to CDUFL, then the algorithm terminates after at most $\log(cost(\widehat{S}_0)/cost(\widehat{S}^*))/\log\frac{1}{1-\epsilon/N}$ number of steps. As

$\log(cost(\widehat{S}_0))$, $\log(cost(\widehat{S}^*))$, and N are polynomial in the size of the input of the algorithm, and there are polynomial number of possible moves in each step of the local-search, the algorithm has a polynomial running time.

The algorithm terminates in the polynomial time with an "approximate" local optimum. For the purpose of analysis, first we show how to bound the facility cost and the connection cost of the optimum, and in the end we show how to modify the analysis to get bounds on the facility cost and the connection cost of the approximate local optimum. For simplicity, we assume that we have unit-demand clients. In the cases of non-unit-demand clients, we can still find an admissible move in the polynomial time, hence run the algorithm, and for the purposes of analysis, we can always treat a client with demand d as d co-located unit-demand clients.

4.3 Analysis

In this section, we show how to bound the facility cost and the connection cost of the solution obtained by our local-search algorithm. We first show how to bound the facility cost $\widehat{F}^S$ and the connection cost $\widehat{C}^S$ of a local-optimal solution S in terms of the facility cost $\widehat{F}^{sol}$ and the connection cost $\widehat{C}^{sol}$ of a feasible CDUFL solution sol. (Recall that we also use $\widehat{F}^S$ and $\widehat{F}^{sol}$ to denote respectively the set of open facilities in S and sol, and we may assume $\widehat{\mathcal{F}}^c \subseteq \widehat{F}^S \cap \widehat{F}^{sol}$.) We do this by using the fact that S is a local optimum and no local-search operation will improve the cost of the solution. We borrow some ideas from the analysis of local search based algorithm for UFL by [1]. Similar to the analysis for UFL, when we use a local search move, we also specify the reassignment of clients to order to (upper) bound the change in the connection cost, but since we have capacitated facilities now, we do this in a more careful way. Particularly, upon removal of a facility s in the analysis of local-search for UFL, we only reassign the clients assigned to s, but in our case we may have to reassign some other clients to satisfy the capacity constraints of the facilities in $\widehat{\mathcal{F}}^c$.

We specify the reassignment of clients for local-search moves in the analysis by means of a suitable graph. We construct a directed graph G with node-set $\widehat{\mathcal{F}} \cup \widehat{\mathcal{D}}$, and arcs from i to all clients in $\widehat{\mathcal{D}}_S(i)$ and arcs from all clients in $\widehat{\mathcal{D}}_{sol}(i)$ to i, for every facility i. Thus each client has one incoming arc and one outgoing arc.

We decompose the arc-set of G into a set of (simple) paths, $\mathcal{P}$, and cycles $\mathcal{R}$ via standard flow decomposition, so that (i) each client j appears on a unique path P_j or on a unique cycle, (ii) each facility appears as the starting point of $\max\{0, |\widehat{\mathcal{D}}_S(i)| - |\widehat{\mathcal{D}}_{sol}(i)|\}$ paths and as the ending point of $\max\{0, |\widehat{\mathcal{D}}_{sol}(i)| - |\widehat{\mathcal{D}}_S(i)|\}$ paths. Let $\mathcal{P}^{st}(s) \subseteq \mathcal{P}$ and $\mathcal{P}^{end}(o) \subseteq \mathcal{P}$ respectively denote the set of paths starting in s and ending in o, and $\mathcal{P}(s,o) := \mathcal{P}^{st}(s) \cap \mathcal{P}^{end}(o)$. For a path $P = \{i_0 := s, j_0, i_1, j_1, ..., i_k, j_k, i_{k+1} := o\} \in \mathcal{P}(s,o)$

define $\widehat{\mathcal{D}}(P) = \{j_0, j_1, ..., j_k\}$, $head(P) = j_0$, and $tail(P) = j_k$. A *shift* along this path is a reassignment of clients so that j_t which is earlier assigned to i_t now assigned to i_{t+1}(opening o if necessary). Note that this is feasible since if $o \in \widehat{\mathcal{F}}^c$ then we know that $|\widehat{\mathcal{D}}_S(o)| \leq |\widehat{\mathcal{D}}_{sol}(o)| - 1 < u_o$. Let $shift(P) = \sum_{j \in \widehat{\mathcal{D}}(P)} (\hat{c}_j^{sol} - \hat{c}_j^S)$ denote the increase in service cost due to shift along path P. We can similarly define shift along a cycle $R \in \mathcal{R}$ similarly, letting $shift(R) = \sum_{j \in \widehat{\mathcal{D}} \cap \mathcal{R}} (\hat{c}_j^{sol} - \hat{c}_j^S)$. Also let $cost(P) = \sum_{j \in \widehat{\mathcal{D}}(P)} (\hat{c}_j^{sol} + \hat{c}_j^S)$.

4.3.1 Bounding the connection cost

We will show that

$$\widehat{C}^S \leq \widehat{C}^{sol} + \widehat{F}^{sol} \tag{4.1}$$

Consider adding a facility $o \in \widehat{F}^S \setminus \widehat{F}^{sol}$ and shifting along all the paths in $\mathcal{P}^{end}(o)$. The change in the objective value consists of the facility-opening cost of o and the change in the connection costs of the clients that are reassigned, i.e., all clients in $\bigcup_{P \in \mathcal{P}^{end}(o)} \widehat{\mathcal{D}}(P)$. From local optimality of S, we get:

$$0 \leq \hat{f}_o + \sum_{P \in \mathcal{P}^{end}(o)} \sum_{j \in \widehat{\mathcal{D}}(P)} (\hat{c}_j^S - \hat{c}_j^{sol}). \tag{4.2}$$

Note that the above argument assumed that the facility o was not present in the local optimum solution. If o is in the local optimal solution, shifting along paths in $\mathcal{P}^{end}(o)$ results in a change in cost of $\sum_{P \in \mathcal{P}^{end}(o)} \sum_{j \in \widehat{\mathcal{D}}(P)} \hat{c}_j^S - \hat{c}_j^{sol}$ which is non-negative (from local optimality) and so the above inequality continues to hold.

For every cycle $R \in \mathcal{R}$, we have $shift(R) \geq 0$. Thus, adding all these inequalities, we get that $\widehat{C}^S \leq \widehat{F}^{sol} + \widehat{C}^{sol}$. Using the argument above we can state the following lemma :

Lemma 4.3.1. *For every $o \in \widehat{F}^{sol}$ and any $\mathcal{Q} \subseteq \mathcal{P}^{end}(o)$, we have $\sum_{P \in \mathcal{Q}} shift(P) \geq$*
$\begin{cases} -\hat{f}_o & \text{if } o \in \widehat{F}^{sol} \setminus \widehat{F}^S, \\ 0 & \text{otherwise.} \end{cases}$

4.3.2 Bounding the facility cost

To bound the facility cost of facilities in $\widehat{F}^S \setminus \widehat{F}^{sol}$, we only need the paths that start at a facility in $\widehat{F}^S \setminus \widehat{F}^{sol}$. Note that all facilities in $\widehat{F}^{sol} \triangle \widehat{F}^S := (\widehat{F}^{sol} \setminus \widehat{F}^S) \cup (\widehat{F}^S \setminus \widehat{F}^{sol})$ are uncapacitated. To avoid excessive notation, for a facility $o \in \widehat{F}^{sol} \setminus \widehat{F}^S$, we now use $\mathcal{P}^{end}(o)$ to refer to the collection of paths ending in o and starting in $\widehat{F}^S \setminus \widehat{F}^{sol}$. (As before $\mathcal{P}(s, o)$ is

the set of paths that start at s and end at o.) Consider a bijection $\pi : \mathcal{P}^{end}(o) \to \mathcal{P}^{end}(o)$ for $o \in \widehat{F}^{sol} \setminus \widehat{F}^{S}$ such that if $P \in \mathcal{P}(s,o)$ and $\pi(P) = P' \in \mathcal{P}(s',o)$ (so $s, s' \in \widehat{F}^{S} \setminus \widehat{F}^{sol}$ by our new notation) then we have the following properties:

- if $|\mathcal{P}(s,o)| \leq \frac{|\mathcal{P}^{end}(o)|}{2}$, we have $s \neq s'$.
  Therefore π maps every "small" set to a different region in $\mathcal{P}^{end}(o)$. So if P belongs to a small set $\mathcal{P}(s,o)$, then the path to which it is mapped starts at a facility different from s.
- If $s = s'$, then $P = P'$.
  Now we cannot guarantee that a path in a "large" set $\mathcal{P}(s,o)$ is mapped to outside the set.
- $\pi(\pi(P)) = P$
  If a path P is mapped to path P', it must be that P' is mapped to P.

Say that o is captured by s if $|\mathcal{P}(s,o)| > \frac{|\mathcal{P}^{end}(o)|}{2}$. Observe that we can have at most one set $\mathcal{P}(s,o)$ of cardinality strictly greater than $|\mathcal{P}^{end}(o)|/2$, so each facility $o \in \widehat{F}^{sol} \setminus \widehat{F}^{S}$ is captured by at most one facility $s \in \widehat{F}^{S} \setminus \widehat{F}^{sol}$. Call a facility s in $\widehat{F}^{S} \setminus \widehat{F}^{sol}$ *good* if it does not capture any facility, and *bad* otherwise.

Lemma 4.3.2. *For any good facility s, we have :*

$$\hat{f}_s \leq \sum_{P \in \mathcal{P}^{st}(s)} \mathit{shift}(P) + \sum_{o \notin \widehat{F}^{S}, P \in \mathcal{P}(s,o)} \mathit{cost}\big(\pi(P)\big). \tag{4.3}$$

Proof. Consider the operation delete(s). We upper bound the increase in the connection cost as follows. Let $j \in \widehat{\mathcal{D}}_S(s)$ and $P_j \in \mathcal{P}(s,o)$. (Recall that P_j is the unique path containing j.) Note that j cannot lie on a cycle since $s \in \widehat{F}^{S} \setminus \widehat{F}^{sol}$. We do the reassignment as follows :

- If $o \in \widehat{F}^{S} \cap \widehat{F}^{sol}$, then we reassign clients on P_j by shifting along P_j.
- If $o \in \widehat{F}^{S} \setminus \widehat{F}^{sol}$, then let $\pi(P_j) = P' \in \mathcal{P}(s',o)$ where $s' \neq s$. We reassign all clients in in $\widehat{\mathcal{D}}(P_j)$ except $\mathit{tail}(P_j)$ by shifting along P_j and reassign $\mathit{tail}(P_j)$ to s'. Let $k = \mathit{tail}(P_j)$ (see Figure 4.1). Since $\hat{c}(k,s') \leq \hat{c}^{sol}_k + \hat{c}(o,s') \leq \hat{c}^{sol}_k + \mathit{cost}(\pi(P_j))$, we can bound the reassignment cost of clients on P_j by $\mathit{cost}\big(\pi(P_j)\big) + \hat{c}^{sol}_k - \hat{c}^{S}_k + \sum_{j' \in \widehat{\mathcal{D}}(P_j) \setminus \{k\}} \big(\hat{c}^{sol}_{j'} - \hat{c}^{S}_{j'}\big) = \mathit{cost}\big(\pi(P_j)\big) + \mathit{shift}(P_j)$.

So we can bound the change in the objective value performing the above operation by

$$0 \leq -\hat{f}_s + \sum_{o \in \widehat{F}^S, P \in \mathcal{P}(s,o)} \mathit{shift}(P) + \sum_{o \notin \widehat{F}^S, P \in \mathcal{P}(s,o)} \Big[\mathit{shift}(P) + \mathit{cost}\big(\pi(P)\big) \Big]. \tag{4.4}$$

which leads to the inequality 4.3.

□

Now consider a bad facility s. Let $\mathrm{cap}_s \subseteq \widehat{F}^{sol} \setminus \widehat{F}^S$ be the set of facilities captured by s, and $o_s \in \mathrm{cap}_s$ be the facility nearest to s.

Lemma 4.3.3. *For any bad facility s, we have*

$$\hat{f}_s \leq \sum_{o \in \mathrm{cap}_s} \hat{f}_o + \sum_{P \in \mathcal{P}^{st}(s)} \mathit{shift}(P) + \sum_{\substack{o \notin \widehat{F}^S \\ P \in \mathcal{P}(s,o): \pi(P) \neq P}} \mathit{cost}\big(\pi(P)\big) + \sum_{\substack{o \in \mathrm{cap}_s \\ P \in \mathcal{P}(s,o): \pi(P) = P}} \mathit{cost}(P). \tag{4.5}$$

Proof. We consider the move $\mathrm{swap}(s, o_s)$ in the local optimum. We upper bound the increase in the connection cost as follows. Consider client $j \in \widehat{\mathcal{D}}_S(s)$ and let $P_j \in \mathcal{P}^{st}(s)$. We do the reassignment as follows:

- If $o \in \widehat{F}^S \cap \widehat{F}^{sol}$, or $o = o_s$ and $\pi(P_j) = P_j$, then we reassign clients on P_j by shifting along P_j. The increase in the cost is at most $\mathit{shift}(P_j)$.

  Otherwise, let $\pi(P_j) \in \mathcal{P}(s', o)$.

- If $\pi(P_j) \neq P_j$(so $s \neq s'$), We reassign $\widehat{\mathcal{D}}(p) \setminus \{\mathit{tail}(P_j)\}$ as in the *shift*operation, and assign $\mathit{tail}(P_j)$ to s'. As in the proof of lemma 4.3.2, the increase in this case can be bounded by $\mathit{shift}(P_j) + \mathit{cost}(\pi(P_j))$.

- If $\pi(P_j) = P_j$ (so $o \neq o_s$), we assign j to o_s. Note that $\hat{c}(j, o_s) \leq \hat{c}^S_j + \hat{c}(s, o_s) \leq \hat{c}^S_j + \hat{c}(s, o) \leq \hat{c}^S_j + \mathit{cost}(P_j)$, so the increase in the cost can be bounded by $\mathit{cost}(P_j)$.

Thus we get the following inequality :

$$\begin{aligned} 0 \leq \hat{f}_{o_s} - \hat{f}_s &+ \sum_{\substack{P \in \mathcal{P}(s,o): o \in \widehat{F}^S \text{ or} \\ o = o_s,\ \pi(P) = P}} \mathit{shift}(P) \\ &+ \sum_{o \notin \widehat{F}^S} \sum_{P \in \mathcal{P}(s,o): \pi(P) \neq P} \Big[\mathit{shift}(P) + \mathit{cost}\big(\pi(P)\big) \Big] \\ &+ \sum_{o \notin \widehat{F}^S: o \neq o_s} \sum_{P \in \mathcal{P}(s,o): \pi(P) = P} \mathit{cost}(P). \end{aligned} \tag{4.6}$$

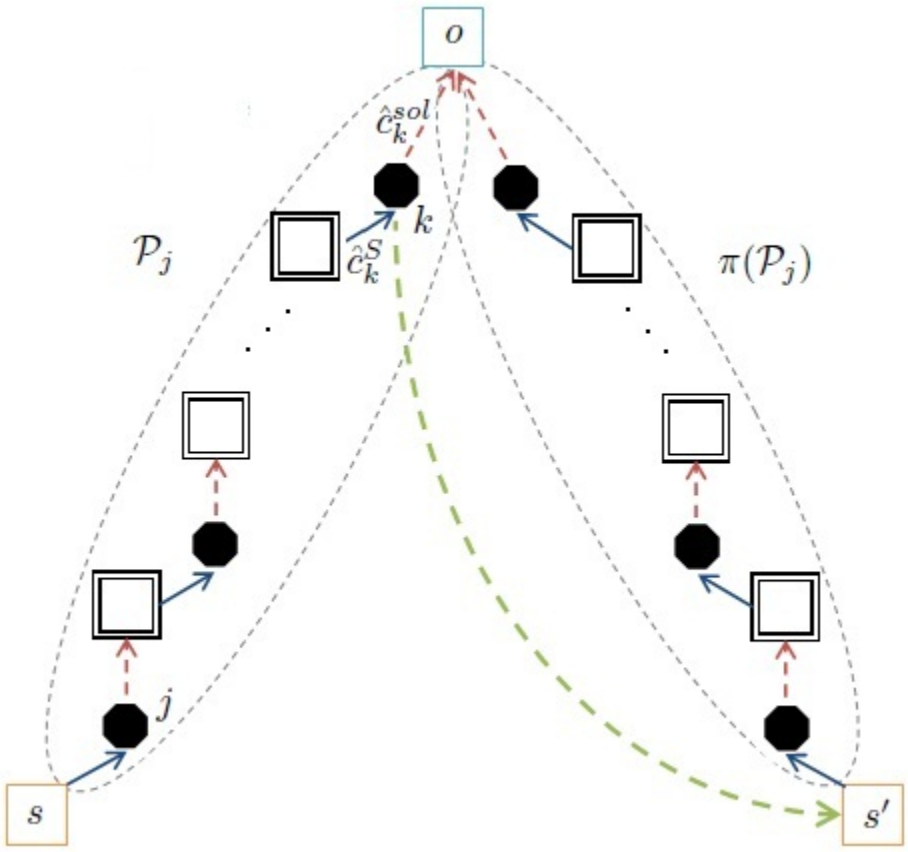

Figure 4.1: The octagons represent clients, and the squares represent facilities (double border squares represent facilities in $\widehat{F} \cap \widehat{F}^{sol}$) . The dashed outgoing arc shows the assignment of a client in solution *sol*, and the incoming arc shows the assignment of a client in solution S.

Now consider an operation in which a facility $o' \in \text{cap}_s \setminus \{o_s\}$ is added. Applying lemma 4.3.1 with $\mathcal{Q} = \{P \in \mathcal{P}(s,o') : \pi(P) = P\}$, we get $0 \leq \hat{f}_o + \sum_{P \in \mathcal{P}(s,o):\pi(P)=P} \mathit{shift}(P)$ for each $o' \in \text{cap}_s \setminus \{o_s\}$. Adding these inequalities and $0 \leq \mathit{cost}(P)$, for each $P \in \mathcal{P}(s,o_s)$ that $\pi(P) = P$, to 4.6 and rearranging proves the lemma. □

Finally we add up inequality 4.3 for all good facilities and inequality 4.5 for all bad facilities and we get the following:

$$\begin{aligned} 0 \;\leq \sum_{s:good} \;& \Big(-\hat{f}_s + \sum_{P \in \mathcal{P}^{st}(s)} \mathit{shift}(P) + \sum_{o \notin \widehat{F}^S, P \in \mathcal{P}(s,o)} \mathit{cost}\big(\pi(P)\big)\Big) \\ + \sum_{s:bad} \;& \Big(-\hat{f}_s + \sum_{o \in \text{cap}_s} \hat{f}_o + \sum_{P \in \mathcal{P}^{st}(s)} \mathit{shift}(P) + \sum_{\substack{o \notin \widehat{F}^S \\ P \in \mathcal{P}(s,o):\pi(P) \neq P}} \mathit{cost}\big(\pi(P)\big) \\ & + \sum_{\substack{o \in \text{cap}_s \\ P \in \mathcal{P}(s,o):\pi(P)=P}} \mathit{cost}(P)\Big) \end{aligned} \tag{4.7}$$

We define three categories on the paths that start from some facility $s \in \widehat{F}^S \setminus \widehat{F}^{sol}$. Let $\mathcal{Q}^1$ denote the set of paths that end in $o \in \widehat{F}^S \cap \widehat{F}^{sol}$. Hence, the rest of paths end in $o \in \widehat{F}^{sol} \setminus \widehat{F}^S$, let $\mathcal{Q}^2 = \{P : \pi(P) \neq P\}$. Let $\mathcal{Q}^3$ be the set of the remaining paths (i.e., $\{P : \pi(P) = P\}$). Using these definition inequality 4.7 leads to:

$$\begin{aligned} \sum_{s \in \widehat{F}^S \setminus \widehat{F}^{sol}} \hat{f}_s \;\leq\; & \sum_{o \text{ is captured}} \hat{f}_o + \sum_{P \in \mathcal{Q}^1} \mathit{shift}(P) \\ & + \sum_{P \in \mathcal{Q}^2} \big(\mathit{shift}(P) + \mathit{cost}(\pi(P))\big) \\ & + \sum_{P \in \mathcal{Q}^3} \big(\mathit{shift}(P) + \mathit{cost}(P)\big) \end{aligned}$$

Using the fact that $\pi(\pi(P)) = P$, we get:

$$\begin{aligned} \sum_{s \in \widehat{F}^S \setminus \widehat{F}^{sol}} \hat{f}_s \;\leq\; & \sum_{o \text{ is captured}} \hat{f}_o + \sum_{P \in \mathcal{Q}^1} \sum_{j \in \widehat{\mathcal{D}}(P)} (\hat{c}_j^{sol} - \hat{c}_j^S) \\ & + \sum_{P \in \mathcal{Q}^2} \sum_{j \in \widehat{\mathcal{D}}(P)} 2\hat{c}_j^{sol} \\ & + \sum_{P \in \mathcal{Q}^3} \sum_{j \in \widehat{\mathcal{D}}(P)} 2\hat{c}_j^{sol} \end{aligned}$$

Which leads to our desired inequality $\widehat{F}^S \leq \widehat{F}^{sol} + 2\widehat{C}^{sol}$.

4.3.3 Bounding the total cost

We can wrap up the results we got so far in the following lemma:

Lemma 4.3.4. *The local optimal solution to a local search algorithm based on* add*,* delete*, and* swap *moves has the facility-opening cost $\widehat{F} \leq \widehat{F}^{sol} + 2\widehat{C}^{sol}$ and connection cost $\widehat{C} \leq \widehat{F}^{sol} + \widehat{C}^{sol}$, where $\widehat{F}^{sol}$ and $\widehat{C}^{sol}$ are the facility and connection costs of an arbitrary solution to the CDUFL instance.*

Now we describe how to modify the analysis to obtain bounds on the facility opening cost and the connection cost of an "approximate" local optimum obtained by our algorithm. To avoid excessive notation, we denote this solution by S. For solution S, we know that no operation improves its cost significantly. More formally, the change in the cost of solution for any operation is larger than $-\frac{\epsilon}{N}(\widehat{C} + \widehat{F})$. (Recall that for a local optimum, we say this change is positive.) We use the same approach in the analysis, so we get similar inequalities but instead having 0 we have $-\frac{\epsilon}{N}(\widehat{C} + \widehat{F})$ for lower bound on the change in the cost (in inequalities 4.1,4.4 and 4.6). Since at most N number of these inequalities add up at the end, we get $-\epsilon(\widehat{C} + \widehat{F}) + \widehat{F} \leq \widehat{F}^{sol} + 2\widehat{C}^{sol}$. Similarly, for the connection cost of S, we get $-\epsilon(\widehat{C} + \widehat{F}) + \widehat{C} \leq \widehat{F}^{sol} + \widehat{C}^{sol}$. Combining these two results, we get the following bounds for S:

$$(1 - \epsilon - \frac{\epsilon^2}{1-\epsilon})\widehat{F} \leq (1 + \frac{\epsilon}{1-\epsilon})\widehat{F}^{sol} + (2 + \frac{\epsilon}{1-\epsilon})\widehat{C}^{sol},$$

$$(1 - \epsilon - \frac{\epsilon^2}{1-\epsilon})\widehat{C} \leq (1 + \frac{\epsilon}{1-\epsilon})\widehat{F}^{sol} + (1 + \frac{2\epsilon}{1-\epsilon})\widehat{C}^{sol}.$$

Dividing inequalities by $1 - \epsilon - \frac{\epsilon^2}{1-\epsilon}$, we get the following theorem:

Theorem 4.3.5. *Given any CDUFL instance, one can efficiently compute a solution with facility-opening cost $\widehat{F} \leq (\widehat{F}^{sol} + 2\widehat{C}^{sol})(1+\epsilon)$ and connection cost $\widehat{C} \leq (\widehat{F}^{sol} + \widehat{C}^{sol})(1+\epsilon)$, where $\widehat{F}^{sol}$ and $\widehat{C}^{sol}$ are the facility and connection costs of an arbitrary solution to the CDUFL instance, and ϵ is a factor that effects the running time of the algorithm.*

Using standard scaling techniques [4], we can scale the facility opening costs by $\sigma > 0$ so we get a solution where $\widehat{F} + \widehat{C} \leq (\widehat{F}^{sol} + \frac{2}{\sigma}\widehat{C}^{sol}) + (\sigma\widehat{F}^{sol} + \widehat{C}^{sol})$. By setting $\sigma = \sqrt{2}$, we get the following corollary:

Corollary 4.3.6. *There is a 2.45-approximation algorithm for CDUFL.*

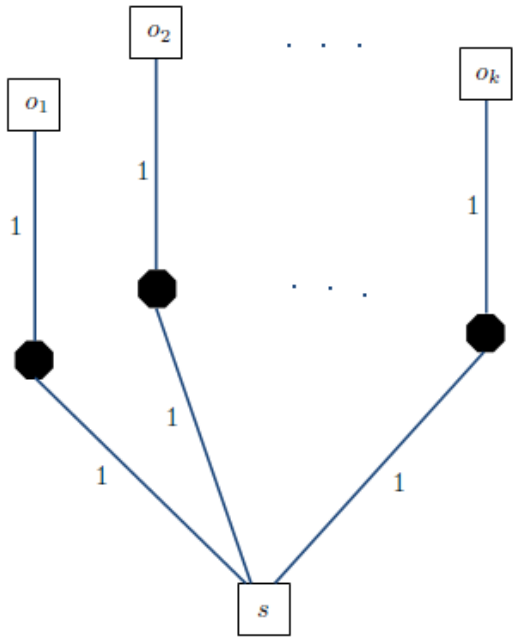

Figure 4.2: Tight example for CDUFL algorithm.

4.4 Tight example

Since UFL is an special case of CDUFL the tight example given by [1] also works for our algorithm. In this example, we have $\widehat{\mathcal{F}} = \widehat{\mathcal{F}}^u = \{s, o_1, o_2, ..., o_k\}$ where the facility opening cost are $\hat{f}_s = 2k$ and $\hat{f}_{o_i} = \epsilon$ for $i \in \{1, 2, ..., k\}$ for some integer k and constant factor ϵ. We have $|\widehat{\mathcal{D}}| = k$ and the distances between the facilities and clients are shown in Figure 4.2. Consider a feasible solution with open facility set $\{s\}$. Clearly this solution is a local optimum since deleting s, or adding any facility, or swapping s and some other facility does not improve the cost of this solution. This solution has cost 3 times the cost of the optimal solution in which the set of open facilities is $\{o_1, o_2, ..., o_k\}$. More precisely, the cost of this solution is $3k$ while the optimal solution has a cost $k(1 + \epsilon)$.

Chapter 5

Conclusion and future work

In this thesis, we presented our approximation algorithm for solving LBFL which achieves the constant-factor approximation guarantee of 83. The running time of our algorithm is dominated by the running time of local-search algorithms for a) UFL instance $\mathcal{I}'$, and b)CDUFL instance $\widehat{\mathcal{I}}$, since other transformation steps can be done efficiently. As we mentioned earlier, to ensure polynomial running time, we only consider the moves with significant improvement in local-search, and hence the algorithm terminates at an approximate local optimum.

Although our approximation factor is much better than the current best approximation factor for LBFL, our results are not yet practical. Finding an algorithm with much better approximation factor which is useful in practice is an interesting future work. Also, we are not aware of any LP-based or local-search algorithm for LBFL, so devising such algorithms is another future work.

Our algorithm can be extended to handle the case that clients have non-unit splittable demands, since for non-unit demands one can still compute the best local search move (for CDUFL), and hence run the algorithm. In this case, our algorithm finds a solution where the demand of a client might be satisfied by different facilities, i.e., our algorithm finds a solution with splittable demand. If splitting the demand is not allowed, one can check if there is a feasible solution to LBFL; but Svitkina [20] claims that approximating the solution within any factor, independent of the demand value, is not possible unless $P = NP$.

Unfortunately, our algorithm (or the algorithm in [20]) does not extend to the generalization of LBFL where each facility i has its own lower bound M_i on the number of clients that should be served by i. In this case, we are not able to provide bounds on the cost of the optimal solution to $\mathcal{I}_2$ (the approach in Lemma 2.3.2 may fail to return a feasible solution). Svitkina shows that there is a reduction from UniFL with monotone non-increasing

facility costs to LBFL with non-uniform lower bounds. We leave the problem of solving LBFL with non-uniform lower bounds as future work. Another interesting open problem is universal facility location problem with non-monotone opening cost functions.

APPENDICES

Appendix A

Integrality gap of LBFL

Let $\big(\mathcal{F}, \mathcal{D}, \{f_i\}, M, \{c(i,j)\}\big)$ be an LBFL instance with facility-set $\mathcal{F}$, and client-set $\mathcal{D}$. Now consider the following simple LBFL instance. We have two facilities i and i' at distance d from each other. There is a set T_i of $M-1$ clients at location i, and a set $T_{i'}$ of $M-1$ clients at location i'. Both of the facilities have zero facility opening cost. The only feasible integer solution is to transfer clients at one of the facilities to the other facility and open the latter, and therefore the cost incurred is $d(M-1)$. However, there is a feasible solution to (LBFL LP) of cost $2d \cdot \frac{M-1}{M}$: we set $y_i = y_{i'} = \frac{M-1}{M}$, $x_{ij} = \frac{M-1}{M}$, $x_{i'j} = \frac{1}{M}$ for each $j \in T_i$, and $x_{i'j} = \frac{M-1}{M}$, $x_{ij} = \frac{1}{M}$ for each $j \in T_{i'}$. Thus, the integrality gap of (LBFL LP) is at least $\frac{M}{2}$.

Appendix B

The locality gap for a local-search algorithm for LBFL

Our initial attempt for solving LBFL was to design a local-search algorithm for it. We are not aware of any constant-factor local-search algorithm for LBFL. For the local-search algorithm based on add, delete, and swap moves we found the following example showing the large locality gap for this algorithm. Consider an LBFL instance $\big(\mathcal{F}, \mathcal{D}, \{f_i\}, M, \{c(i,j)\}\big)$ with facility-set $\mathcal{F} = \{o, s_1, s_2, ..., s_M\}$, and client-set $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup ... \cup \mathcal{D}_M$, where the $\mathcal{D}_i$s are disjoint sets of size M (see Figure B.1). The facility opening costs are defined as follows : $f_o = M^2 + \epsilon$ and $f_{s_i} = M$ for each $i \in \{1, 2, ..., M\}$. All clients are located at unit-distance from o. For all clients, we have $c(s_i, j) = M$ for $j \in \mathcal{D}_i$ and $c(s_i, j) = M + 2$ for each $j \notin \mathcal{D}_i$. Note that these distances can be extended to yield a metric on $\mathcal{F} \cup \mathcal{D}$. The solution S with set of open facilities $F^S = \{s_1, s_2, ..., s_M\}$ is a local optimum since no improving add, delete,or swap move exists. The cost of this solution is $M^2 + M^3$. However, the optimal solution opens facility $\{o\}$, and it has a total cost of $2M^2 + \epsilon$. So the locality gap for this algorithm is at least $M/2$.

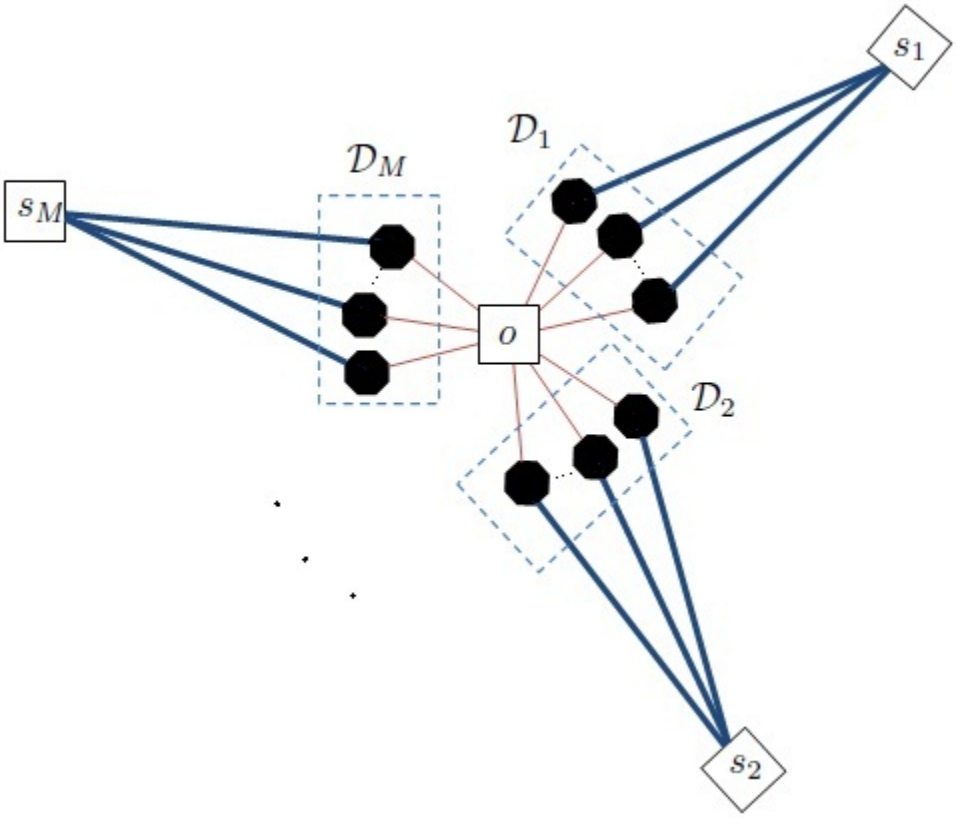

Figure B.1: An example showing large locality gap for a local-search algorithm based on add,delete, and swap moves for LBFL.

Appendix C

Integrality gap of CDUFL

Let $\left(\widehat{\mathcal{F}} = \widehat{\mathcal{F}}^u \cup \widehat{\mathcal{F}}^c, \widehat{\mathcal{D}}, \{\hat{f}_i\}, \{u_i\}, \{\hat{c}(i,j)\}\right)$ be a CDUFL instance with facility-set $\widehat{\mathcal{F}}$ (where $u_i = \infty$ for all $i \in \widehat{\mathcal{F}}^u$, and $\hat{f}_i = 0$ for all $i \in \widehat{\mathcal{F}}^c$), and client-set $\widehat{\mathcal{D}}$. We propose the following natural linear programming for the CDUFL. Let y_i be indicator variables denoting whether facility i is open, and x_{ij} is an indicator variable denoting whether client j is assigned to facility i.

$$\begin{aligned}
\min \quad & \sum_i \hat{f}_i y_i + \sum_{j,i} \hat{c}(i,j) x_{ij} && && \text{(CDUFL LP)}\\
\text{s.t.} \quad & \sum_i x_{ij} \geq 1 && \forall j \in \mathcal{D} && \text{(C.1)}\\
& x_{ij} \leq y_i && \forall i \in \mathcal{F}, j \in \mathcal{D} && \text{(C.2)}\\
& \sum_j x_{ij} \leq u_i && \forall i \in \widehat{\mathcal{F}}^c && \text{(C.3)}\\
& x_{ij}, y_i \geq 0 && \forall i \in \mathcal{F}, j \in \mathcal{D}.
\end{aligned}$$

Constraint C.1 states that each client has to be assigned to a facility and constraint C.2 ensures that this facility is open. Constraint C.3 ensures that at most u_i clients are assigned to a capacitated facility i. Now consider the following simple CDUFL instance. We have two facilities i and i', and $u+1$ clients, all present at the same location. Facility i is uncapacitated and has opening cost f, and facility i' has capacity u (and zero opening cost). Any solution to CDUFL must open facility i and therefore incur cost at least f. However, there is a feasible solution to (CDUFL LP) of cost $\frac{f}{u+1}$: we set $y_i = \frac{1}{u+1}$, and $x_{ij} = \frac{1}{u+1}$, $x_{i'j} = \frac{u}{u+1}$. Thus, the integrality gap of (CDUFL LP) is at least $u+1$.

Bibliography

[1] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristic for k-median and facility location problems. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, page 29. ACM, 2001. 4, 5, 18, 19, 20, 26, 28, 35

[2] ML Balinski. On finding integer solutions to linear programs, 1964. 1

[3] J. Byrka. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 29–43, 2007. 4

[4] M. Charikar and S. Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *focs*, page 378. Published by the IEEE Computer Society, 1999. 4, 5, 18, 19, 20, 26, 34

[5] R.L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, 45(9):1563–1581, 1966. 1

[6] S. Guha, A. Meyerson, and K. Munagala. Facility Location with Demand Dependent Costs and Generalized Clustering, 2000. 4

[7] S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. In *focs*, page 603. Published by the IEEE Computer Society, 2000. 2, 4, 5, 6, 7, 18, 19

[8] M.T. Hajiaghayi, M. Mahdian, and V.S. Mirrokni. The facility location problem with general cost functions. *Networks*, 42(1):42–47, 2003. 4

[9] DR Karget and M. Minkoff. Building Steiner trees with incomplete global knowledge. In *focs*, page 613. Published by the IEEE Computer Society, 2000. 4, 5, 6, 7, 18, 19

[10] M.R. Korupolu, C.G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. In *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, page 10. Society for Industrial and Applied Mathematics, 1998. 4

[11] A.A. Kuehn and M.J. Hamburger. A heuristic program for locating warehouses. *Management Science*, pages 643–666, 1963. 1

[12] A. Lim, F. Wang, and Z. Xu. A transportation problem with minimum quantity commitment. *Transportation Science*, 40(1):117–129, 2006. 4

[13] M. Mahdian and M. Pal. Universal facility location. *Algorithms-ESA 2003*, pages 409–421, 2003. 4

[14] A.S. Manne. Plant location under economies-of-scale-decentralization and computation. *Management Science*, 11(2):213–235, 1964. 1

[15] D.B. Shmoys. The design and analysis of approximation algorithms: facility location as a case study. *Trends in Optimization: American Mathematical Society Short Course, January 5-6, 2004, Phoeniz, Arizona*, 61:85, 2004. 3

[16] D.B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 265–274. ACM, 1997. 3, 5

[17] J.F. Stollsteimer. *The effect of technical change and output expansion on the optimum number, size, and location of pear marketing facilities in a California pear producing region.* PhD thesis, University of California, Berkeley, 1961. 1

[18] J.F. Stollsteimer. A working model for plant numbers and locations. *Journal of Farm Economics*, 45(3):631, 1963. 1

[19] M. Sviridenko. An improved approximation algorithm for the metric uncapacitated facility location problem. *Integer programming and combinatorial optimization*, pages 240–257, 2006. 5

[20] Z. Svitkina. Lower-bounded facility location. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1154–1163. Society for Industrial and Applied Mathematics, 2008. iii, 4, 5, 6, 7, 16, 17, 18, 19, 20, 21, 24, 36

[21] J. Zhang, B. Chen, and Y. Ye. A multi-exchange local search algorithm for the capacitated facility location problem. *Integer Programming and Combinatorial Optimization*, pages 1–4, 2004. 4, 15