Improved Approximation Algorithms for Facility Location Problems

A Thesis

Submitted in the fulfilment of the requirement

for the award of the degree of

Doctor of Philosophy

in

Computer Science and Engineering

Submitted by

Vaishali Mehta

(Regn. No. 951003004)

Under the Supervision of

Dr. Deepak Garg, Supervisor

Professor & Head, Computer Science Engineering, Bennett University

&

Dr. Maninder Singh, Administrative Guide

Professor & Head, CSED, Thapar Institute of Engineering & Technology

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY

Computer Science and Engineering Department
Thapar Institute of Engineering & Technology
Patiala – 147001 (Punjab), India.

Dedicated to
My family

Table of Contents

CERTIFICATE

I hereby certify that the work which is being presented in this thesis entitled IMPROVED APPROXIMATION ALGORITHMS FOR FACILITY LOCATION PROBLEMS, in fulfilment of the requirements for the award of the degree of DOCTOR OF PHILOSOPHY submitted in Computer Science and Engineering Department (CSED), Thapar Institute of Engg. & Tech., Patiala, Punjab, is an authentic record of my own work carried out under the supervision of Dr. Deepak Garg and refers the work of other researchers, which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(Vaishali Mehta)
Registration No. 951003004

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge and belief.

05/11/18

Deepak Garg
Professor & Head,
Department of Computer Science & Engg.
Bennett University
Greater Noida, India.

Supervisor

Maninder Singh
Professor & Head,
CSED
Thapar Institute of Engg. & Tech.
Patiala, India.

Supervisor (Administrative)

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY

Computer Science and Engineering Department
Thapar Institute of Engineering &Technology, Patiala

CANDIDATE'S DECLARATION

I hereby certify that the work which presented in this thesis entitled "**Improved Approximation Algorithms for Facility Location Problems**" is being submitted to the Computer Science and Engineering Department in fulfilment of the requirement for the award of the degree of **Doctor of Philosophy.** The work submitted in this thesis is my own work done under the supervision of **Dr. Deepak Garg, Professor & Head, Computer Science Engineering Department, Bennett University, Greater Noida.**

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other Institute/University.

Vaishali Mehta
Reg. No. 951003004

ACKNOWLEDGEMENT

I would like to express my special appreciation, gratitude and thanks to my advisor Professor Dr. Deepak Garg, who have been a tremendous mentor for me. I would like to thank him for encouraging my research. His advice on both research as well as on my career have been invaluable. A person with an amicable and positive disposition, he has always made himself available to clarify my doubts despite his busy schedules and I consider it as a great opportunity to do my doctoral programme under his guidance and to learn from his research expertise. Thank you Sir, for all your help and support.

I would like to acknowledge the Director and Dean (R&D), Thapar Institute of Engineering & Technology for the academic and technical support which has been indispensable. I also take the opportunity to express my sincere thanks to my administrative supervisor & Head of Department, Professor Dr. Maninder Singh. His motivational words and guidance at the time of need has been truly inspiring. I would also like to thank my committee members, Dr. Anil Kumar Verma, Dr. V. P. Singh and Dr. Sunil Singla for serving as my committee members even at hardship. I also want to thank you for your brilliant comments and suggestions for improvement of my Ph.D. work from time to time. I also thank the faculty and staff of Computer Science and Engineering Department in promoting a stimulating and welcoming academic and cooperative environment.

My hearty thanks to all the anonymous referees of several international research journals in the field of approximation algorithms and computational complexity. The research work presented in this thesis report has accomplished the eminence due to the constructive comments and ideas for improvement by them.

A special thanks to my family. Words can not express how grateful I am to my mother-in law, father-in-law, my mother, and father for all of the sacrifices that you've made on my behalf. Your prayers for me were what sustained me thus far. I would also like to thank to my beloved husband, Dr. Sachin Wadhwa. Thank you for supporting me for everything, and especially I can't thank you enough for encouraging me throughout this experience. A special acknowledgement goes to my friend Monika Mangla. She has been a true friend ever since we got enrolled in Ph.D. programme. She is an amazing person and has always been supportive in every way. Finally I thank my God, for letting me through all the difficulties. Thank you, Lord.

Date: NOVEMBER 05, 2018

Place: Thapar Institute of Engg. & Tech., Patiala

(Vaishali Mehta)

List of Figures

List of Algorithms

List of Abbreviations

FLP	Facility Location Problem
UFLP	Uncapacitated Facility Location problem
CFLP	Capacitated Facility Location Problem
FLPWP	Facility Location Problem With Penalty
FLPTP	Facility Location Problem with Time Dependent Penalty
FLPLP	Facility Location Problem with Linear Penalty
FLPSP	Facility Location Problem with Submodular Penalty
RFLP	Robust Facility Location Problem
FTFLP	Fault Tolerant Facility Location Problem
PFLP	Priority Facility Location Problem
PFLPWP	Priority Facility Location Problem with Penalty
PFLPTP	Priority Facility Location Problem with Time Dependent Penalty
MFLP	Multi Level Facility Location Problem
MFLPWP	Multi Facility Location Problem with Penalty
LP	Linear Programming
ILP	Integer Linear Programming
MIP	Mixed Integer Programming
APX	Approximable
PTAS	Polynomial Time Approximation Scheme
FPTAS	Fully Polynomial Time Approximation Scheme

List of Publications

Papers Published

1. **Vaishali, Deepak Garg "Approximation Algorithm for Resource Allocation Problems with Time Dependent Penalties"** accepted in *International Journal of Foundations of Computer Science*, WorldScientific, ISSN: 1793-6373, accepted on 15th Feb., 2017 (SCI Journal with impact factor – 0.63).
2. **Vaishali, Deepak Garg "Multiobjective Facility Location Problems : A Review"** *International Review on Modelling and Simulation*, ISSN: 1974-9821, vol. 4, no. 4, pp. 1871-1876, August 2011 (A peer reviewed journal indexed by SCOPUS with impact factor – 0.44).
3. **Vaishali, Deepak Garg "Facility Location Problem Using Genetic Algorithm: A Review"** *Research Journal of Computer Systems and Engineering (RJCSE)*, ISSN: 2230-8563; e-ISSN: 2230-8571 Vol 2 Issue 2 pp. 63-66 June, 2011.(Scopus Indexed).

Papers under Review in SCI Journals

4. **Vaishali, Deepak Garg** "**A constant factor Approximation Algorithm for Uncapacitated Priority Facility location problems with Time Dependent penalties**" in European Journal of Industrial Engineering (SCI).
5. **Vaishali, Deepak Garg** "**Approximation Algorithm for Uncapacitated Multi-level Facility location problems with Time Dependent penalties**" in Sadhna Journal (SCI).
6. **Vaishali, Deepak Garg** "**A (3+ϵ) Approximation Algorithm for Uncapacitated facility Location Problems with Delayed Service**" in Sadhna Journal (SCI).
7. **Vaishali, Deepak Garg** "**Approximation Algorithm for Logistics Systems with Critical Service Time Requirements**" in Turkish Journal of Electrcal Engineering and Computer Science (SCI).

ABSTRACT

The problem of locating facilities or resources within a specified region is one of the most well researched problems in the field of optimization and operations research. The problem is to identify an optimal set of locations (sites) to open facilities (warehouses) amongst a set of candidate locations, while minimizing the sum of the cost of installing (opening) the facilities plus the cost of allocating (assigning) request nodes (demand nodes) to open facilities. The assignment cost is generally the weighted sum of distances (Euclidean) between request nodes and facilities. Each facility also satisfies minimum or maximum capacity constraints. The most popular variant of facility location problem is the k-median problem. The k-median problem tries to minimize the allocation cost and allows at most k facilities to open or locate. Both FLP and k-median have been used in a wide range of applications including network design, data warehousing, and clustering to name a few. These are NP-hard problems, and have been extensively researched from the perspective of computational complexity i.e. average-case and worst-case performance guarantees. Such problems can be solved by using approximation algorithms. The approximation algorithms obtain a near optimal solution with the help of heuristics and techniques which include local search, linear programming rounding, and primal-dual algorithms. In this thesis, we have developed improved approximation algorithms for different variants of facility location problems. The algorithms we design are mainly based on two approximation techniques: linear programming rounding and the primal-dual approximation. We show that these techniques are very effective in solving variety of location-allocation problems. The major contributions of this thesis to the field of facility location literature are the following:

- The Uncapaciated Facility Location Problem with Time Dependent Penalties (UFLPTP): We develop an approximation algorithm for the UFLPTP which we prove

to be constant factor to the optimal solution. Using a simple linear programming formulation for this problem and subsequently using primal-dual algorithms, we show that the solution to this problem can be obtained within factor 4 approximation. This is the best known upper limit for UFLPTP.

- The Uncapacitated Facility Location Problem with Critical Service Time Requirements: We present a Primal-dual based greedy heuristic for this problem. We give first constant factor approximation for such problems with an upper bound of $(3 + \epsilon)$.
- The Uncapacitated Facility Location Problem with Parallel Supplies: we design an approximation algorithm using primal-dual approach. The computational complexity of our algorithm is an improvement over the algorithm available in literature, but the performance guarantee is larger. We give a performance guarantee of 6 which is quite larger and can be reduced but at the expense of increased running time.

Chapter 1

1. Introduction

Whenever any organization wishes to open a store, warehouse or set up a server, the most important question that needs to be addressed is the location. For example, if the company wants to open a store for maximizing the number of customers patronizing it, ware house needs to be opened so that cost of shipment between various stores and the warehouse is minimized. Thus whenever any location decision is to be made, it has an objective function associated with it. On the other hand if a company plans to open a factory that is intended to produce chemicals hazardous to human health, the effort will be to to maximize the distance of the factory from any site of residence.

Similarly location decisions are important for individual and public sectors also. For example, whenever any individual plans to buy a house, the most important point is to finalize the location that satisfies various demands of the aspirant. These requirements may include its proximity to the market, school and busy places of the city etc. Similarly whenever government bodies want to open any hospital or fire station, they need to select a location which reachable to all possible demand sites within some stipulated time. Furthermore location selection is the most important factor that determines success or failure of any business, service etc. and therefore it requires developing an efficient location model.

Mathematical location model addresses the various questions including:

- *How many facilities should be located?*
- *Where these facilities should be located?*
- *What should be the size of each facility?*
- *How demand sites should be allocated to facilities?*

All above stated questions are answered by the objective functions of the location model. For example, a hospital or fire station should be located as near as possible to the probable demand sites. While a factory producing hazardous chemicals should be located as far away from residential area.

Many effective strategies have been designed and developed in operations research theory to solve location problems optimally [1][2][3]. This involves simple heuristics such as *greedy heuristics* or *local search* [4][5][6]. But these heuristics have not been proven to be the best or cheapest solution. As location-allocation decisions are hard to make and FLPs are intractable, an optimal solution cannot be found efficiently. Therefore, researchers from past few decades are focusing more on finding a near optimal solution also called as *approximate solution*. The techniques for finding an approximate solution to the computationally intractable problems are known as *approximation algorithms*. The theory of approximation algorithms has attracted most researchers working on computational complexity. Moreover, approximation algorithms have also been applied effectively to different categories of FLPs. In the next section, we briefly discuss the existing findings on facility location model.

1.1. Facility Location Models and Classification

Facility location handles location of p facilities in a demand plane (open or closed) containing n demand sites while optimizing the objective function value. The facility location models may be categorized based on various parameters: nature of demand plane, type of facility to open, number of facilities to open, nature of facilities etc. The location models are classified into four categories:

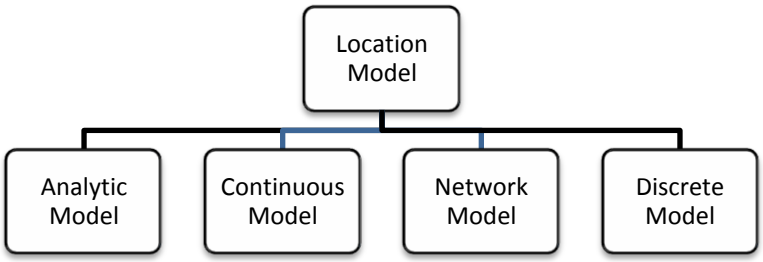


Figure 1.1: Classification of Location Model

In analytic models the demand nodes are distributed in some manner over the service region and the facilities can be located anywhere within the service region. This assumption of distribution of demand sites in the service area limits its application [7] [8][7][7][7][7].

Unlike analytic location models, in continuous location models demand arises at only discrete points in the service region and the facilities can be located anywhere in the demand plane. This necessitates consideration of infinite number of possible locations for facilities and thus becomes computationally expensive. Therefore it requires devising some specialized technique for continuous location models. In continuous location model, the location of facility is determined with reference to the location of existing demand sites in the service area. Classical Fermat Weber Problem is an example of continuous location model.

In Network location model, it is assumed that demands arise in the network and facilities can also be located on a network consisting of nodes and links. In network location model, it is assumed that transportation among nodes exists along edges in the graph only. Network location model can be applied for locating a server or a firewall in a network of organization [9]. Literature of network location model mainly focuses on finding polynomial time algorithm for specialized graph e.g trees etc. Polynomial time algorithm for locating p facilities on a network can be found in literature. Linear time algorithms also exist for non-weighted network location model for one or two facilities.

Another major classification of location model is discrete location model. Unlike continuous location model, in discrete location model, demand generates on the nodes and facilities are also restricted to be located on finite set of candidate locations. The discrete location models

are further classified into covering based location model, median based and other miscellaneous models etc. as shown in figure 1.2.

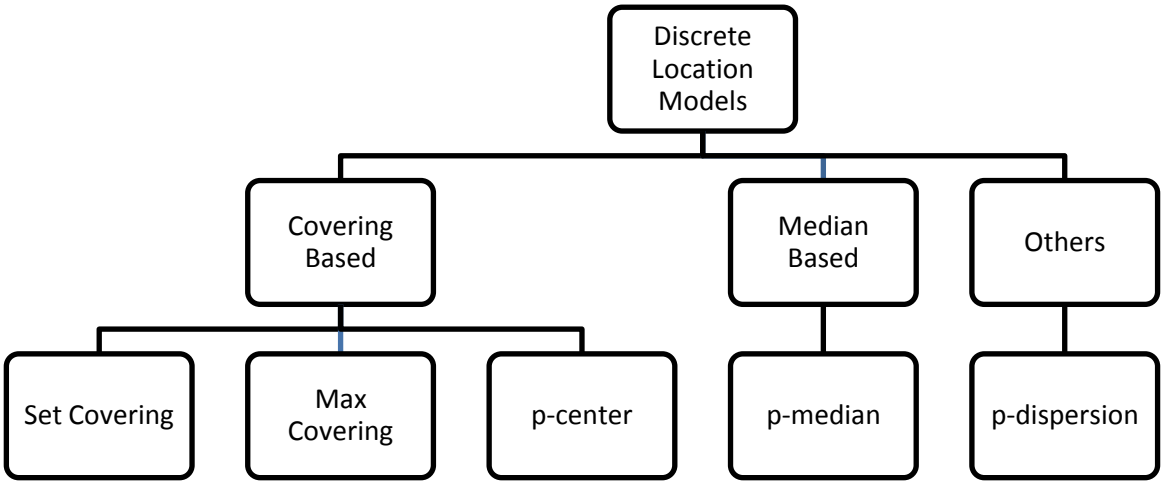


Figure 1.2: Categories of Discrete Location Model

Covering model is used for a service that necessitates some critical coverage standard (in terms of distance and/or time). Example of such services are emergency services like ambulance service, fire station service etc. In such services, a demand node is considered as served only if it receives the service within this service standard. Here the objective of location model can be for example: every demand site should receive the service within 6 minutes time otherwise it will not be considered as covered. The covering model can be implemented by set covering model, maximal covering model and *p-center* model. In covering model, a demand site is covered if it has at least one facility within coverage standard. The objective of set covering model is to cover each demand site within required service standard. Set covering model was introduced in [10] and was later addressed by Toregas et al. [11]. Another variant of covering location model is *Maximal covering problem,* which aims to cover maximal demand nodes. *p-center* location model is an additional well known variant of covering model. In *p-center*, the objective is to locate *p* resource in the

service area so that the longest distance for every demand site to its nearest resource is minimized and can be expressed as:

$$Z(X) = \min\left\{\max\ d\,(p_i, X)|\ 1 \leq i\ \leq n\right\}$$

It aims to locate a facility at X so that the distance of X from its farthest demand site $p_i|\ 1\ \leq i \leq n$ is minimized.

Furthermore, median based models aims at minimizing the sum of weighted distance between every request node and its nearest resource. Median based model is applicable for transportation and distribution context where travelled distance influences the total cost in a significant manner.

In median based model, *p-median* location model aims to locate p facilities so that sum of weighted distance of all demand points to respective nearest resource is minimized. The objective function is also termed as minisum. For example, *p-median* location model can be used for location of p warehouses of an organization in the service area so that the total transportation cost from and to warehouses is minimized. It can be mathematically formulated as [12]:

$$\text{minimize}\, Z(X) = \sum_{i=1}^{n} h_i d(X, p_i)$$

Here d represents Euclidean distance among the facility and the demand sites.

Unlike covering and median based location model, *p-dispersion* model is used for location of p obnoxious facilities like garbage dump, nuclear power plant, crematorium etc. in the service area. The objective here is to locate p facilities so that distance of each of p facility to its nearest demand site is maximized:

$$Z(X) = \max\{\min d(\,p_i, X)|\ 1 \leq i\ \leq n\}$$

The objective is to maximize the distance of resource to the nearest demand site in the service area.

As already discussed, *p-dispersion* is mainly used for obnoxious facilities but its usage can also be extended to locate outlets of an organization in competitive scenario. Furthermore, it can also be aptly used for locating additional stores of an industry when few stores already exist in the service area. In this scenario, the location model aims at maximizing the distance among its neighbouring stores and thus maximizing the covered region.

Location model can be categorized based on the nature of input: *static location model* and dynamic *location model*. Location model where the input does not vary along with time is called *static location model* while dynamism in the input along time results *dynamic location models* [13][14][15]. In all location models, facilities can have capacity constraint as well thus generating *capacitated location model* and *uncapacitated location model*. Capacity constraint of the resource limits the number of demand sites it can be allocated to. In contrast, uncapacitated model does not impose any limit on the number of demand nodes to be served [16].

Apart from above classification of location models, FLP can also be categorized on the basis of objective function to be optimized. Following are the most researched variants of FLP according the objective of the problem to be solved.

1.2. Variants of FLP based on Objective Function

As discussed earlier FLP aims at locating the set of facilities p in a demand plane having number of demand sites. In order to evaluate an optimal location of facility or facilities to open, we require objectives and constraints on the system to be modelled. A number of models have been proposed in the literature for a variety of different location-allocation problems. The reader is suggested to refer to survey paper by Klose and Drexl (2005) [17] which includes a detailed overview of FLPs used in distribution systems and states many mathematical programming formulations for a large variety of location problems. As

mentioned earlier, different types of facilities have different objective functions to be optimized (maximization or minimization). The FLP however aims at opening facilities at optimal locations so that the objective function can be optimized. There exist multiple variants of the FLPs. Following are some well known variations:

The *universal facility location problem (UniFLP)*[18] is the most general form of a facility (resource) location problem. In UniFLP, we are provided with a set *F* of facilities (or resources) and a set *C* of request nodes. For every facility $i \in F$ and request node $j \in C$, associated a *service cost* c_{ij} (also known as *connection cost or transportation cost*) between facility i and node *j*. Every facility $i \epsilon F$ is assigned installation cost f_i. Service costs are directly proportional to the distance between facility and the request node. In FLP we generally consider Euclidean distances which follow triangle inequality. The aim is to obtain a solution S so that the total facility opening cost and cost of allocating each facility to a request node $c(S) = c_f(S) + c_s(S)$ is minimized.

The *metric uncapacitated facility location problem* (UFLP) [19] is the most common class of FLP which is studied by most researchers. In this we have a set *F* of facilities or resources and a set *C* of demand points, a non-negative facility installation cost *fi*, a demand *hj* for each $j \epsilon C$ and a *transportation cost* c_{ij} between facility *i* and request node *j*. The aim is to find the location of a subset of facilities to open and allocate each demand point to one of the open facilities s.t. the total cost is minimized. The cost includes both service cost and installation cost. A request node can be allocated to one facility only. The effort therefore is made to allocate a request node to its nearest facility that is open. Thus the aim is to obtain a subset *S* of facilities in a way to minimize $\sum_{i \in S} fi + \sum_{j \in C} \min_{i \in S}\{c_{ij}\}$.

The *capacitated facility location problem* (CFLP) [19] is another variation of location problems where a limit is imposed on the number of request nodes that a facility can serve. In the CFLP, along with other input variables as in UFLP, we are provided a capacity u_i that

specifies the maximum number of request nodes that can be served by a facility. This means that a facility i can be allocated to maximum of u_i demand points. Similar to the UFLP there is a *service cost* c_{ij} between facility *i* and request node *j*. The aim is to find the location of a subset S of facilities to open in a way such that the sum of installation cost and the transportation cost is minimized without violating the capacity constraints.

The metric *soft-capacitated facility location problem* is a special class of general FLP where more than one facility can be opened at a given location. The total cost in SCFLP is a function defined as $f_i(k) = f_i\lceil k/u_i \rceil$ which specifies that if we want this facility to be allocated to *k* demand points, it has to be opened $\lceil k/u_i \rceil$ times for which the cost is $f_i\lceil k/u_i \rceil$.

In the *metric k-median problem* [1] we are provided *n* points in a metric space. Out of these we have to choose *k* points to be the centers to locate facilities, and then allocate each demand point *j* to the nearest chosen center. If request node *j* is allocated to a center *i*, a cost $d_j\, c_{ij}$ in incurred where d_j is the demand at point *j*. The aim is to choose the *k* centers in a way to minimize the total assignment costs. The k-median problem can be differentiated from the FLP in two ways: there is no facility opening cost, and there is an upper limit *k* on the number of facilities that are to be located. Here *k* is an input variable and is not fixed. Furthermore, the metric space ensures that the connecting edges from facility to request node follow the triangle inequality. The aim is to open a set *k* of facilities s.t. the sum of service costs for the request nodes is minimized.

The *metric fault tolerant facility location* (FTFLP) [20] extends the UFLP where each request node is assigned a positive integer specifying its coverage requirement r_j. It aims at finding a solution that opens some facilities and allocates each request node *j* to r_j different facilities that are open. In FTFLP every node *j* is to be allocated to r_j different facilities. The service cost in this case is a weighted sum of these r_j allocations. It aims at minimizing the sum of

the installation cost and the weighted sum of connection cost of each request node to its nearest open facility.

Facility location problem with linear/submodular penalties [21] is another variant of classical UFLP. Along with other problem variables as in UFLP, we have a penalty cost p_j. In FLPSP, every request node might not be served by some open facility. Instead, a request node j can either avail the service by a facility that is open or rejected with penalty cost p_j. The aim is to find the locations of a subset of facilities to open such that each request node $j \in D$ is either allocated to an open facility or rejected at all. The aim is to minimize the total cost, including the installation, transportation and the penalty costs. The service cost between request nodes and facilities follows a metric, i.e., $c_{ij} \leq c'_{ij} + c_{i'j'} + c_{i'j}$.

Facility location problem with time dependent penalties (FLPTP) is an extension of FLPSP where the penalty cost is a non-decreasing function of time. A function $f(.)$ is time dependent if $f(x, t_1) \leq f(x, t_2)$ for $t_1 \geq t_2$. Function f is monotonically increasing if $f(X) \leq f(Y)$ for any $X \subseteq Y$. FLPTP thus considers a non-decreasing penalty function with respect to time i.e. when a request node is waiting for a service, the penalty incurred increases in proportion to the waiting time. The objective here is to identify optimal locations of facilities to open and a set of request nodes to be penalized that are allocated to open facilities. The FLPTP has its applications in critical situations where timely delivery is the major concern instead of finding good facility locations [22].

In *stochastic facility location problem* [23] the objective is to minimize the expected cost. The SFLP is basically a two-stage optimization problem where locations are decided during first stage and allocation or assignment of request nodes is the second stage decision. However the demand of each request node is not known initially. During second stage, one of the k events may happen with a probability p_k. The cost of opening a facility at first stage is f_{i0}, but at the second stage it is f_{ik} if k^{th} event happens. The service cost c_{ij} however does

not change and remains same in both the stages. The goal is to find sets of facilities $S_0, S_1, \ldots\ldots, S_k$ for each event which minimizes the function $\sum_{i \in S_0} f_i^0 + \sum_{k=1}^{K} \sum_{i \in S_k} p_k f_i^k + \sum_{k=1}^{K} \sum_{j \in C} p_k \, min_{i \in S_0 \cup S_k} \{c_{ij}\}$.

In *priority based facility location problem*s [24] each request node has a priority of demand z_j and each facility has a non-decreasing cost function $f_i(z)$which specifies the facility cost according to the request node's priority of demand. These models generally follow a greedy approach for optimizing objective function value. The request nodes are scheduled for service on the basis of their priority. Higher priority request nodes are served first than the lower priority ones. The aim is to open a set of facilities and allocate them to a subset of request nodes in a way that the overall cost is minimized while preserving the priority constraints. Such models have their applications in many real life situations e.g. emergency services, reservation systems, networking [25], traffic management [26] etc.

All the variants of facility location models discussed above have their significance in different fields and applications. In this thesis, however we will discuss different variants of the *uncapacitated facility location problem (UFLP)* and their algorithmic aspects. Remainder of this thesis is focused on discussing and developing effective heuristics for different variants of UFLP. Our techniques try to improve the existing results for the problem under consideration.

1.3. Approximation Algorithms

A wide range of optimization problems are known to be NP-hard. It is widely accepted (however not proven) that all NP-hard problems are intractable, i.e. there does not exist any polynomial time algorithm that finds an optimal solution for such problems. Some well known examples of NP-hard optimization problems are the minimum travelling salesperson problem, the 0/1 knapsack problem, the minimum bin-packing problem and the location-

allocation problems, etc.

All the location problems as discussed above are NP-hard. Their solution approaches focus on finding the solution that optimizes some objective function. Also, it requires identifying the best solution. Finding best or exact solution for location problem has a complexity *which is not polynomial*. Therefore, it needs to devise an approach that finds approximate solution in polynomial time. Such algorithms that find approximate solution to NP problems are called approximation algorithms. For a minimization problem P, algorithm A is called ρ-approximate if for each instance x of P it generates a solution which is at most ρ times the cost of optimal solution for the same problem instance. We say that ρ is the *approximation factor* or *approximation ratio* of P. It is clear from the above definition that $\rho \geq 1$ for a problem with minimization objective and $\rho \leq 1$ for the maximization objective. As ρ approaches 1 the approximate solution equals the exact solution. These algorithms with $\rho \in R$ are known as constant factor approximation algorithms. Thus approximation algorithms basically find approximate solutions to NP problems in polynomial times because we are only interested in the algorithms which restrict the running time complexity at most to a polynomial. The well known NP problems are Knapsack problem, job scheduling, travelling salesperson problem, etc.

The design of good approximation algorithms for intractable problems is a continuous research practice where one continues to find outperforming/better solution methods and heuristics. These techniques have gained popularity for solving a number of real life problems. Good approximation algorithms have been proposed for many optimization problems.

In this thesis we mainly focus on the constant-factor approximation algorithms that run in polynomial time. The so called *APX (approximable)* complexity class includes the set of optimization problems that allow a polynomial-time approximation algorithms with a

performance ratio bounded by a constant factor. These are also known as *constant-factor-approximation algorithms.* An ε −approximation algorithm is the one that gives is a solution that is at most ε times worse than the optimal solution for some constant ε.

For many problems even better approximation algorithms can be designed. The *PTAS (polynomial-time approximation scheme)* class contains the set of optimization problems that allow a polynomial-time algorithm within every fixed percentage of the input size. That means it gives us a solution as close to the optimum as we like, provided that we are willing to compromise the running time as compared to quality of solution.

It has been proved that there are problems that are in APX but not in PTAS unless P=NP; i.e. problems that can be approximated within some constant factor, but not every constant.In some cases, we need approximation scheme that is polynomial both in terms of input size and the magnitude of approximate error. This is known as the *fully polynomial time approximation scheme (FPTAS).* Thus *FPTAS* is an approximation scheme that develops an algorithm whose time complexity is polynomial both in the input size as well as in $1/\varepsilon$.

We have $P = NP$ only if all NP class problems can be approximated within a constant factor. Moreover, if all optimization problems with constant approximations can be approximated arbitrarily, still $P = NP$. In other words, some of the NP-hard problems are hard to approximate as well. The detailed information of approximation and complexity theory can be found in [27][28][29][30].

1.4. Approximation Techniques

Following are some of the proven techniques and some heuristics for developing approximation algorithms for NP problems:

1.4.1 The Greedy Method

The greedy approximation is based on the idea that chooses locally optimized solution which may eventually lead to globally optimized solution. The greedy algorithms start from the scratch with an empty initial solution, process the elements in decreasing order of their weight and add each element to the current solution. The power of greedy approximation is that it is a generic algorithm that can be tried nearly on any problem. However, greedy may produce a good solution but does not guarantee a global optimal solution. Greedy approach based approximation algorithms for FLPs were first proposed by [31].

1.4.2 Local Search

Local search and iterative improvement algorithms are one of the most used heuristics in optimization. These algorithms repeatedly consider a neighbouring solution and search for a locally optimum solution. If the local operations are inexpensive, and input size is small, these algorithms are very efficient. The simplest neighbourhood of a solution is considering a solution with one extra facility. We need to determine if opening a facility at a new location would reduce the cost of the solution. It seems reasonable, if the assignment cost of our solution is high, the reduction of assignment cost would pay for the cost of the new facility. The next question is whether we can we delete facilities when the facility cost is high. If we can perform both, we would have a combinatorial approximation algorithm. The idea would be to use the `Add' strategy to reduce the assignment cost by increasing the facility cost; and subsequently trying to reduce the facility cost by paying increased assignment cost. The latter would be a `Delete' step. Observe that we would not be always able to delete facilities (for example, if it is the only open facility), but can always attempt an exchange. The quality of obtained solution depends on the fact that both the final facility installation and assignment costs of our solution are low, otherwise the solution need to be improved. The first

combinatorial approximation algorithm for the FLP was given by Korupolu et. al. [5]. They analyzed a heuristic named `Add and Drop', first proposed by Kuehn and Hamburger in [2]. They proved that if a facility had other facilities close by, the first facility can be closed and the assignments shifted to the nearby open facility; and if it cannot be done it would imply all facilities are far from each other. In this case the optimum must also have its facilities spread out. Consequently we can try exchanging facilities, adding one and deleting one, to improve the solution. This algorithm gives a performance guarantee of 5, in time $O(n^2k^2)$, on n locations and *k* possible facilities.

More precisely, the local search method starts with a feasible solution x, and compares it with "neighbouring" feasible solutions. If a better neighbour y is found, move to y, and repeat the same procedure. If no such neighbour is found, the process stops. A local optimum is found at the final stage of the procedure. Generally, the local search procedure follows following simple steps:

Algorithm 1.1 : The Local Search Procedure

- *Find an initial solution s to be the current solution and compute the value of objective function F(S).*
- *Select a neighbour S' of S and compute F(S').*
- *Test whether S' is better than S. If yes then replace S with S', otherwise S will remain as current solution.*
- *Report current best solution if termination condition reached, otherwise move to step 2.*

Example 2 Cost minimization of UFLP using local search algorithm.

Input: A set C of request nodes, a set F of facilities, service cost c_{ij} and installation cost f_i.

Output: Find a set $S \subseteq F$ to open s.t. the total cost of assignment is minimized i.e. *min(* $\sum_{i \in S} fi + \sum_{j \in C} \min_{i \in S}\{d_{ij}\}$).

Algorithm 1.2: Local Search Procedure for UFLP

- Start from arbitrary set $S \subset F$.
- In every iteration check if swapping one of the facilities in S with one in $F - S$ improves the solution.
- Terminate if no more swaps improve the solution.

This simple local search procedure gives a 5-approximation to the optimal solution. Later V. Pandit et. Al. [32] improved the solution to 3-approximation. Local search is the only known technique to provide a constant factor approximation for the CFLP. Assuming we have instance-specific lower bounds that are not derived from proof of approximation guarantee of the final solution, we can provide instance-based guarantees. Of course, we would only be able to prove worst-case bounds, but if we were faced with a real instance, an independent lower bound is of great value. It is also possible that the final solution constructed will be better than predicted by the exact worst-case of the analysis.

1.4.3 Integer and Linear Programming

Linear Programming [33] [34] [35] [36] has been one of the most important tools in optimization from the above viewpoint. Formulating a problem as an integer linear program and solving its fractional relaxation to get a lower bound has been a first step. This is useful even in the context of exact optimization. In the context of approximation algorithms, the fractional solution of a linear program provides a lower bound if we ensure that the optimum solution satisfies the constraints. The objective in this case is to construct an integral solution which is a small factor of the lower bound using the fractional solution. This step is referred

to as rounding. This approach bas been used for decades to bound performances of approximation algorithms [37] [38] [39].

In a linear programming problem, the goal is to either maximize or minimize a linear function subject to a finite set of linear constraints. A feasible solution must satisfy all of the linear constraints. If the solution does not satisfy any of the linear constraints then it is infeasible. The linear function which is to be optimized is also called as the objective function. An optimal solution is the feasible solution for which the objective function is to be optimized. A linear program is said to be unbounded if, for a maximization (minimization) problem we can achieve arbitrarily large (small) values of the objective function.

A simple linear program:

$$\text{maximize} \sum_{j=1}^{n} c_j x_j$$

$$\text{s.t.} \sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m$$

$$x_j \geq 0, \qquad j = 1, \ldots, n$$

and its matrix form is:

$$\text{maximize } C^t x$$

$$s.t. Ax \leq b$$

$$x \geq 0$$

This is the *standard form* of given LP.

However in certain cases, the specific linear program formulated may not be sufficiently strong. The fractional solution may be significantly smaller than the optimum solution. This is referred to as the *integrality gap* of the formulation, and is the best approximation factor we can hope to prove.

An *integer linear program* (ILP) is, by definition, a linear program with the additional constraint that all variables take integer values:

$$max\ c^T x\ s.t. Ax \leq b\ and\ x\ integral.$$

Example 3 An ILP formulation for UFLP

The Integer linear programming formulation for metric UFLP is as follows [47] **:**

$$minimize \sum_{i \in F} f_i x_i + \sum_{i \in F} \sum_{j \in D} c_{ij} y_{ij}$$

Subject to:

$$y_{ij} \leq x_i, \quad \forall i, j$$

$$\sum_{i \in F} y_{ij} = 1, \quad \forall j$$

$$y_{ij} \in \{0,1\} \quad \forall i, j$$

$$x_i \in \{0,1\} \quad \forall i$$

The inequality constraint states, for each request node *j*, we must choose at least one candidate location in I that can be allocated to *j*. This will take exponential time. Therefore, we need to relax the integrity requirements. The resulting LP is polynomial time solvable and is as follows:

$$minimize \sum_{i \in F} f_i x_i + \sum_{i \in F} \sum_{j \in D} c_{ij} y_{ij}$$

Subject to:

$$y_{ij} \leq x_i, \quad \forall i, j$$

$$\sum_{i \in F} y_{ij} = 1, \quad \forall j$$

$$y_{ij} \geq 0 \quad \forall i, j$$

$$x_i \geq 0 \quad \forall i$$

Let OPT be the optimal solution of ILP and $OPT(L)$ be the optimal solution of LP. Clearly, $OPT(L) \leq OPT$ as the solution space for the IP will be a subset of the solution space of the LP. Therefore, the minimum value of LP puts a limit on the minimum value of the IP and is the optimal solution of the problem. It can be observed that linear programming is a powerful

tool of providing a near optimal (approximate) solution to all optimization problems in polynomial time.

1.4.4 Linear Programming Duality

In the previous discussion we saw few approaches to solving the FLP. In the local search technique, we had an algorithm which was simple and efficient. However it did not provide us with a good lower bound. On the contrary, the linear programming relaxation gave us a good lower bound, but solving a linear program for large input variables and constraints is nontrivial. In further discussion we will have a combinatorial algorithm, as well as a lower bound for even larger input instances. *Duality* is another powerful feature of linear programming which may produce amazing results for some problems. In simple words duality states that every linear program has another linear program related to it and hence can be derived from it. The original linear program is called as primal and the derived one is termed as its dual.

For example, consider following LP formulation of a maximization problem:

$$\text{maximize} \sum_{j=1}^{n} c_j x_j$$

s.t.

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m$$

$$x_j \geq 0, \quad j = 1, \ldots, n$$

its *dual* is the LP:

$$\text{minimize} \sum_{i=1}^{m} b_i y_i$$

$$s.t. \sum_{i=1}^{m} a_{ij} y_i \geq c_j, \qquad j = 1, \ldots, n$$

$$y_i \geq 0, \qquad i = 1, \ldots, m$$

While solving LPP, we actually solve two problems – the primal resource allocation problem and the dual resource valuation problem. The dual of every maximization problem will be a

minimization one and vice-versa. If the primal problem has *n* variables and *m* constraints then its dual will have *m* variables and *n* constraints. The optimal value of dual problem provides an upper bound over the optimal value of primal for a maximization problem and a lower bound for minimization problem. Following important theorems are derived from duality theory.

Theorem 1: (Weak Duality) Let *x* and *y* be the primal and dual feasible solution resp., then we have $y^T b \leq c^T x$. *i.e.* the objective function value of the primal feasible solution can be no greater than the objective value of dual feasible solution.

Theorem 2: (Strong Duality) The primal problem has a finite optimal solution only if its dual too has a finite optimal solution. Moreover, if $xopt$ and $yopt$ are primal and dual optimal solutions respectively, then $c^T x = y^T b$.

The LP formulation has its drawbacks as well. If the number of constraints and/or variables are large, solving a linear program is not the trivial one. In a few cases we can reach a middle ground, where we have a combinatorial algorithm, as well as a lower bound somewhat independent of the solution constructed. For LPP with minimization objective, the FLP for example, its dual will have a maximization objective, and by the weak duality, any feasible solution to the dual linear program is a lower bound to the primal linear program. The feasible solution to a maximization dual problem can be used as a lower bound to the primal solution. The idea is to create a feasible dual solution, and an integral solution of the primal which is bounded by the dual solution. This was first explicitly demonstrated in [40], and subsequently the method has been used in a variety of problems [41].

1.5 Approximation Algorithms via Linear Programming

The theory of LP and duality has been proven to be very helpful in designing approximation algorithms for combinatorial optimization problems[42]. There are two efficient techniques for designing approximation algorithms using LP:

1. LP Rounding
2. Primal-dual approximation

All approximation algorithms based on linear programming follow 3 basic steps which are:

Algorithm 1.3: Approximation Algorithm Using Linear Programming

i) Formulate the given problem as an integer linear program (ILP). Let us denote its optimal objective value by OPT.

ii) Relax the integrity constraints in ILP formulation to obtain corresponding linear program (LP). Let optimal value of the relaxed program is denoted by OPT'.

iii) Use the relaxed LP to obtain a polynomial time feasible solution to the ILP. The value of obtained solution is bounded by $\rho.OPT$ for some $\rho > 1$.

As discussed earlier, the relaxation of ILP is the LP. Therefore, $OPT' \leq OPT$. Hence, an efficient polynomial time algorithm can be developed to obtain the optimal solution of ILP of value at most $\rho.OPT$.

For an ILP formulation of a problem, we solve it efficiently as a relaxed linear program over the real numbers. If the optimal solution happens to have integer values, then we have the optimal solution for our combinatorial problem of cost $opt(LP) \leq opt(ILP)$.

For some problems, in fact, this happens for every input. If, however, we solve the relaxed linear program efficiently over the real numbers and we find a solution x^* with fractional

values, but then we are able to *round* the fractional solution x^* into integral x' without changing the cost of the solution too much i.e. $cost(x^{'}) \leq c.cost(x^*)$ for some constant *c.* Then we have an efficient approximation algorithm for our problem. Thus the cost of the optimal solution is bounded by : $cost \leq c.opt(LP) \leq c.opt(ILP)$.

Subsequently fractional optima of linear programs can be rounded using LP rounding as follows:

1.5.1 LP Rounding

The solution to a LPP in most cases will not be integral. We therefore round the solution of the LPP to a feasible solution of corresponding ILP, with value within a factor $\rho.OPT$. This will produce a feasible solution of ILP of value less than or equal to $\rho.OPT$. By using properties of the problem, the rounding procedure is usually performed in polynomial time.

However, all the problems can not be solved using this rounding technique. For problems having decision variables which have values between 0 and 1, a special rounding technique called *randomized rounding* is used. It works as follows [87] : Let $x_j \in \{0,1\}$ be a decision variable for some $j \in J$ in an ILP and that in corresponding LP the integrality constraint of these variables is relaxed to $0 \leq x_j \leq 1$. Let $xopt$ be the optimal solution of given LP. Being in $[0,1]$, the decision variables $xopt_j$, of the optimal solution can be interpreted as probabilities. Now, we round each variable x_j to 1 with probability $p(xopt_j)$, where p is chosen such that it can be computed in polynomial time. The rounded solution is feasible for ILP and $\sum_{j=1}^{n} c_j Prob(x_j = 1) \leq \rho.OPT'$

Because of probabilistic properties of rounding procedure, there may be different outputs in different runs. We however are interested in the expected cost, which is the expected value of, $\sum_{j=1}^{n} c_j x_j$ $i.e., E(\sum_{j=1}^{n} c_j x_j) = \sum_{j=1}^{n} c_j Prob(x_j = 1)$. Now because we know that

$opt' \leq opt$, the expected cost of rounded solution is no more than ρ times the optimal solution.

1.5.2 Primal-dual Approximation

The rounding procedure discussed above though gives a near optimal solution to integer linear programming problems, is computationally expensive. Another approximation technique used as an alternate to solve linear program is called *primal-dual approximation technique.* It was originally introduced by Dantzig, Ford, and Fulkerson [43] in 1956 and then revised and used by many researchers for designing exact and approximate algorithms. The most powerful feature of primal-dual method is that it reduces the weighted optimization problems to easier unweighted ones. Following steps are performed to obtain an approximate solution using primal-dual procedure.

Algorithm 1.4: A General Primal-dual Approximation Algorithm

i) Formulate the problem as an ILP.

ii) Let us denote the optimal value of the solution by OPT.

iii) Relax the ILP to a primal linear program P and find its dual D.

iv) Let x and y be the primal and dual feasible solutions resp. Initially, $x = y = 0$.

v) Repeat following steps until x is feasible:

   a. Increase the value of y_i iteratively improve the qualityof dual solution.

   b. Select a subset of tight dual constraints and increase the value of primal variables corresponding to these constraints by an integral amount.

The cost of dual solution gives a lower bound on OPT.

This method will give us a α-approximation of the optimal solution of ILP.

1.6 Motivation of the thesis

Most variants of location problems are usually studied and researched with the assumption that *all* the demand nodes are to be serviced. A significant shortcoming of this kind of FLP formulation is that a few remote demand nodes, called *outliers*, may put a strong influence over the cost of the final solution. In this thesis we discuss a generalization and solution methods of various facility location problems (*k*-center, *k*-median, uncapacitated facility location etc) to the case when only a specified number of the customers are to be served. But the problem becomes harder when we also have to select the subset of clients that should get service. For many applications of facility location, such as express delivery, it may be that all clients must be serviced. However, for the majority of commercial applications of facility location, it becomes economically crucial to ignore very distant outliers. The simple formulations of facility location problems described above can lead to spurious solutions, in which facilities are placed in isolated areas just to satisfy the demands of a few outliers. This kind of arrangement may drastically increase the cost of overall solution and hence lower the performance. In this thesis we mainly focus on the notion of how to perform facility location in contexts where outliers may exist.

Our principal contribution is to formulate variations of the facility location problems so that outliers can be handled in a meaningful way, and to consider the computational consequences of these new formulations. The essential feature of our formulations is to provide additional parameters that allow a small subset of the clients to be denied service, thereby reducing costs drastically. These denied clients do not contribute to the final service cost.

The three types of problems that will be addressed are UFLP with delayed demand satisfaction, UFLP with time dependant penalty and Uncapacitated Priority facility location problem with penalty. In the first case, we assume that all the clients will be serviced but it

might get longer time for few remote clients which may influence the service cost of the solution. In the second case, a penalty is imposed for delayed service which increases with time. In this formulation not all the clients will avail the service, but only a subset is selected which can be served within the specified time period. In the third case, each client has an associated priority based on which it is decided that which client will get the service first. We propose primal-dual and rounding based approximation algorithms to deal with each of these situations. We also analyze and discuss about their computational consequences that how the added constraints affect the cost of overall solution.

1.7 Outline of the thesis

In this thesis we propose approximation algorithms for different variants of FLPs. In chapter 1 we discussed about classification of location models and different variants of FLPs. Then we gave a brief overview of computational complexity and approximation algorithms. Techniques of designing approximation algorithms including local search, greedy heuristic, LP rounding and primal-dual approximation were also explained along with appropriate examples. Remaining part of the thesis is organized as follows.

Chapter 2 presents the work done in the field of approximation algorithms for facility location models. In this the known approximation algorithms for FLPs are studied. It focuses mainly on the metric UFLP and its variants. We in this chapter study the existing results on approximation algorithms for metric UFLP, metric CFLP, metric fault tolerant UFLP, UFLP with penalties and UFLP with dynamic demands.

In chapter 3, a variant of UFLP is studied where rejection of service is associated with a penalty. This is known as FLP with penalties. The goal is to choose a subset of facilities to open permanently so that the total cost (which includes facility opening cost, service cost) as well as the penalty imposed due to delay in service is minimized. We consider the problem

where penalty increases with time at a linear rate. First a simple LP based relaxation is explained which a 4-approximation for this problem. Next an extension of this problem is presented where penalty is subjected to permitted time. A a primal-dual based 4-approximation algorithm for this problem is proposed in this chapter.

Chapter 4 is an extension of work done in chapter 3. It considers the priority facility location problem (PFLP) with critical service time requirements. Here, the services are provided on the basis of priority of demand. Also, there is time limit within which the service is to be delivered failing which the facility is subjected to a penalty. We first present a simple primal-dual approximation algorithm for PFLP. Next we present a greedy heuristic and primal-dual based $(3 + \varepsilon)$-approximation for PFLP with penalty.

Chapter 5 discusses the FLPs with parallel deliveries. In this problem there are *k* types of facilities to open: one type of service center and *(k − 1)* types of service providers. Each request must be shipped from the service center through service providers of type *k − 1, ..., 1* to the request nodes via a predetermined path. The transportation costs between demand points and service center, as well as transportation costs between service providers are known and they satisfy the triangle inequality. The aim is to identify the facilities to open and to allocate request nodes to paths along open facilities such that the total cost is minimized. Work done in this chapter is inspired by the 3−approximation algorithm based on LP−rounding technique proposed by Aardal, Chudak and Shmoys [44] for the metric uncapacitated multi-level FLP. We discuss a 6−approximation algorithm based on the primal-dual approach. Next we prove that there also exists a 6-approximation algorithm for penalty version of MFLP.

Finally in chapter 6, the concluding remarks are given and the scope of further research is discussed.

Chapter 2

2 Survey on Approximation Algorithms for FLP

FLP focuses on determining location for one or more facilities while optimizing one or more objective functions. The motive behind intensive research in FLP is its applications in numerous real life situations. In order to efficiently solve the location model, some mathematical model needs to be devised. In the literature, numerous models and algorithms have been proposed for solving various location problems efficiently.

These models may differ in the objective function, nature of solution space and few other factors that may affect the solution. Opening of a facility in facility location model incurs some cost. This cost may further be similar for each facility or may vary from facility to facility. Furthermore, the service cost is the cost of giving a service to demand site from some facility. Generally the objective function in these models is minimization of the total cost .

In this chapter, existing findings on the results of FLP have been presented. The facility location has been a luring area for numerous researchers since $60s$. Here we focus on the basic assumptions, models and solution methods. We start with the most researched UFLP and then present the existing results of various other facility location models.

2.4 Metric Uncapacitated Facility Location Problems

UFLP is a popular variant of location model [45]. The same has been an intensive area of interest among researchers since 1960s [46] [47] [48][49]. In Chapter 1 the mathematical

formulation of UFLP has been described. LP formulation by Balinski [50] is also presented in chapter 1.

From past few years several approaches have been used as a tool for providing approximate solution to FLP. Here, existing results for UFLP are discussed in terms of computational complexity.

The work by Shmoys and Aardal [51] has emerged as a breakthrough. J. Vygen in [52] focused on the algorithms that are based on LP rounding followed using primal-dual procedure proposed in [53]. Authors in [54] [55] also have developed a greedy heuristic which gives better approximation guarantee for the metric UFLP. This greedy method has been implemented for some examples and it has been verified that it helps in obtaining better performance guarantees. Shmoys et. al. [56] have used the LP rounding technique and gave a 4-factor approximation algorithm. Chudak in his paper [57] carried forward this research and reached at conclusion that the rounding itself can be improved quite easily. Guha et. al. [54] have provided an approximation guarantee of 1.46 for general metric service cost.

2.1.1 The primal-dual algorithm [58]

Authors in [58] have given an optimization algorithm for the UFLP. In this algorithm, primal-dual technique was used. According to algorithm of Jain and Vazirani, the relaxed complementary slackness conditions are:

$$\frac{1}{3}c_{ij} \leq v_j - t_{ij} \leq c_{ij}, \qquad \forall i \in F, j \in D \text{ and } x_{ij} > 0 \tag{2.1}$$

Once we get the values of primal variables (x, y) and dual variables (v, t), thereafter cost of the primal solution (x, y) is bounded by

$$c(x) + 3.f(y) = \sum_{i \in F, j \in D} c_{ij} x_{ij} + 3. \sum_{i \in F, j \in D} t_{ij} x_{ij} \tag{2.2}$$

$$= \sum_{j \in D} \sum_{i \in F} (c_{ij} + 3.t_{ij}) x_{ij} \leq 3. \sum_{j \in D} v_j \quad (2.3)$$

Furthermore, since

$$c(x) + 3.f(y) \leq 3. \sum_{j \in D} v_j \leq 3.C_{LP} \leq 3.C_{OPT}, \quad (2.4)$$

Authors in [58] gives a performance guarantee within a factor 3.

Mettu and Plaxton [59] have also proposed a similar combinatorial algorithm. In this algorithm, for facility cost f_i a quantity T_i is obtained in such a manner that:

$$\sum_{j \in D} max. \quad \{T_i - c_{ij}, 0\} = f_i \quad (2.5)$$

It is based on the clustering technique analogous algorithm in [58]. A 3−approximation algorithm is proved by this approach. Although the algorithms using dual fitting for the UFLP [60], [61], [62] follow a different approach, they produce the same result as by Jain and Vazirani algorithm. It is based on the following idea.

$$\sum_{j=1}^{k} v_j \leq \beta. \left(f_i + \sum_{j=1}^{k} c_{ij} \right) \quad (2.6)$$

then the pair $\beta^{-1}(v, \bar{t})$ with $\overline{t_{ij}} = \max\{v_j - \beta.c_{ij}, 0\}$ is a dual feasible solution. Here $\beta^{-1} \sum_{j \in D} v_j$ gives a lower bound for optimal solution by the weak duality and therefore the approximation guarantee of the algorithm is β.

Mahdian et. al. [60] have developed the first algorithm for the UFLP. They also implemented the Jain and Vazirani's algorithm in order to construct the pair (v, t) with the difference that a facility i is assumed to be fully paid if it keeps the resource until the end of the algorithm as follows:

$$\sum_{j\ unconnected} t_{ij} = f_i \qquad (2.7)$$

All paid facilities are opened and every request node is allocated to the nearest open facility. The above setting in the standard algorithm of Jain and Vazirani, shows that cost of a facility f_i is recovered only when the connected demand nodes contribute for it. It can be proved that the relation is satisfied for $\beta = 1.86$ of the above relation.

This algorithm was further extended by Jain et. al. [63]. The extension was motivated by interpretation of the dual program. The algorithm developed by Mahdian et. al. [64] used the algorithm proposed by Jain and Mahadian [63] and greedy technique. A performance guarantee of 1.52 is obtained for UFLP.

Arora and Raghwan in [65] have given a randomized polynomial approximation scheme for 2-dimension Euclidean space. It was later extended by Rao et. al. [66] by generalizing the result to metric spaces. A. Kolen [67] has considered a special case of UFLP on a tree. The suggested approach assumes that there exist r nodes on a tree. It assumes that the request nodes and the facilities are nodes. In network model, service cost c_{ij} is the length of path between i and j in the network. He proved that UFLP with these assumptions is solvable in $O(r^3)$. This work was extended by Barany [68] who implemented the UFLP on trees for partitioning solvable in $O(V^2)$. Apart from researchers like G. Cornuejols [69] and A. Ageev [70], V. L. Beresnev [71] has also proposed that the UFLP is solvable in polynomial time. Hochbaum [31] also proposed a greedy approach with $O(log\ n_c)$ approximation ratio. The authors here obtained the same performance guarantee.

2.5 Metric Capacitated Facility Location Problems

Sometimes it is realistic to impose a capacity limitation on the resources, thus limiting the number of demand sites it can serve to. This version of the location model is known as CFLP.

The mathematical formulation and basic notations have been discussed previously. CFLP aims to locate facilities in a manner that minimize the sum of installation cost and service cost

- A subset S of open facilities which are open
- An assignment $x : S \times D \rightarrow \mathbb{R}_+$ of customers' demand to open facilities

An optimum assignment x provides the solution to a transhipment problem. If u_i and d_j have integer values, it becomes an integral optimum assignment. At the same time if capacities are integer with all demands set to 1, the demands are said to be non-splittable among multiple facilities.

The CFLP was initially addressed by Kuehn and Hamburger [48] They presented a heuristic based procedure for CFLP. The next most influential paper following this was work done by Geoffrion and Graves [72]. Here authors considered a model of distribution network consisting of factories, distribution centres and customers. They devised an approach to locate distribution centres with an assumption that each request node receives its service by single distribution centre only. This model was tight bound for number of distribution centres.

Later Akinc [73] and Nauss [74] attempted the problem using linear programming and Lagrangean relaxation respectively. This was followed by the most effective techniques by Van Roy [75] and Beasley [76]. T. V. Roy in [75] devised the algorithm by considering dual of the capacity constraint. Jayaraman et. al. [77] incorporated the multi-commodity problem using Lagrangean relaxation as a solution framework.

Pal et. al. [78] established the turning point for CFLP and proposed the first approximation algorithm for its general case. This work was further extended by Korupolu et. al. [79] . The same was taken a step ahead by Zhang et. al. [80] who gave performance guarantee of 5.83. Levi et. al. [81] proposed a 5-factor approximation algorithm for a special a case where all

facilities have uniform opening cost. Authors in [18] has addressed work by Pal and Tardos [82] as the Universal FLP.

Shmoys et. al. [56] have attempted the CFLP using filtering and rounding and obtained an 7/2-approximation algorithm. Khuller and Sussman [83] later considered an assumption that k facilities of capacity u can be opened at k locations. This homogenous CFL model was also researched by Shmoys et. al. [56].

Various efficient and popular algorithms for CFLP include branch-and-bound Akinc et. al. [73], Lagrangian relaxation, Benders decomposition Davis [84]. Some researchers used combination of these methods in their algorithm like Van Roy [75], dual-based methods Guignard et. al. [85], and heuristics Melkote et. al. [86].

Levi et. al. [87] studied the CFLP in which a mobile service station with limited capacity is to be located. Daskin et. al. [88] had examined a capacitated location model with single source constraint. This location model was further intricated by considering customer demands that are stochastic.

Korupolu et. al. [5] used the triangle inequality and produced a solution which is polynomial in input size with a factor of 8+ϵ of optimal solution. Chudak and Williomson [89]. This result was further modified by Chudak and Shmoys [77] who devised an algorithm within a factor of $6(1+\epsilon)$

They considered the CFLP where $k \vee k > 1$ facilities are to be opened at same time. They also expressed that any α −approximation algorithm can be transformed to polynomial-time algorithm by paying an additional cost. J. Vygen [52] handled another variant of CFLP i.e. soft-capacitated FLP (SCFLP) where each facility has fixed capacity u_i. Here if demand for i exceeds the capacity u_i then it suggests opening of multiple copies of the facility. In the hard-capacitated FLP we can't open multiple copies of the facilities. Mahdian et. al. [90] This soft-capacitated FLP was addressed by Mahdian et. al. [91] and they devised a 2-

approximation algorithm by reducing it to the UFLP. Chudak and Shmoys [92] also worked on SCFLP with uniform capacities using LP rounding. On the contrary, [58] addressed non-uniform capacities by reducing it to UFLP followed by finding solution using primal-dual approach. It is proved with help of results given by all these researchers that SCFLP is solvable within a factor of 3.

Bumb [93] considered the CFLP where all facilities have uniform capacities. The first approximation algorithm for uniform CFLP was given by Korupolu, Plaxton and Rajaraman [79]. In this work, they used local search algorithm by Kuehn *et al.* [48] and proved that every locally optimal solution has its cost not more than $(8+\epsilon)$ times cost of an (global) optimum solution. This result was further progressed by Chudak and Williamson [94][89] by obtaining a $(5.83+\epsilon)$ −approximation.

Allocation of Demand Nodes

Allocation of demand nodes is the process of assigning demand sites to corresponding facilities in such a manner that whenever there exists a demand at any demand site, it is served by assigned facility. The allocation is preceded by determining the set of open facilities. Manisha et al. [95] have obtained that locally optimal solution is $(3+\epsilon)$ −approximation. Levi et al. [96] proposed an algorithm that decomposes the problem, which can be solved independently. They obtained that if opening cost for all facilities are equal, then this algorithm is 5-approximation algorithm.

Non-uniform capacities

Non-uniform CFLP was handled by Pal, Tardos and Wexler [82] initially and a $(8.53+\epsilon)-$ approximation was obtained. Mahdian and Pal [18] reduced this to $(7.88+\epsilon)$ approximation algorithm. This work was taken further by Zhang, Chen and Ye [97] and they succeeded to obtain a $(5.83+\epsilon)$ approximation algorithm. Bansal et. al. [98] modified the

result by introducing a multi operation in place of *add*, *open* and *close operation* and obtained a $(5+\epsilon)$ factor approximation.

The Single-source CFLP (SSCLP)

The SSCLP falls in the class of NP-hard problems. In SSCLP, it is required to obtain complete request of a demand node from single facility i.e. the demand for a node is not splittable. The purpose of the allocation is to minimize the connection cost. It can be mathematically modelled as: given a set $I = \{1,\ldots,n\}$ of possible facility locations, each with a maximum capacity $a_i, i \in I$, and a set $J = \{1,\ldots,m\}$ of demand nodes, each with a demand $d_j, j \in J$. It is required to choose facilities to open and to perform allocation in a way to minimize the total cost, and the capacity constraints are not violated. This is illustrated in figure 2.2.

Neebe and Rao [99] have transformed the SSCLP to set partitioning problem and then using branch and bound procedure. Cortinhal [100] has given computational results for two random data set with 25 facilities and 40 customers by De Maio et. al. [101]. Barcelo et. al. [102] relaxed the constraint of customer assignment and solved the problem in two phases: first phase finds the location of the plant and second phase focuses on the assignment of request nodes to the located plants. H. Luss et. al. [103] used dual of the capacity constraint for SSCLP.

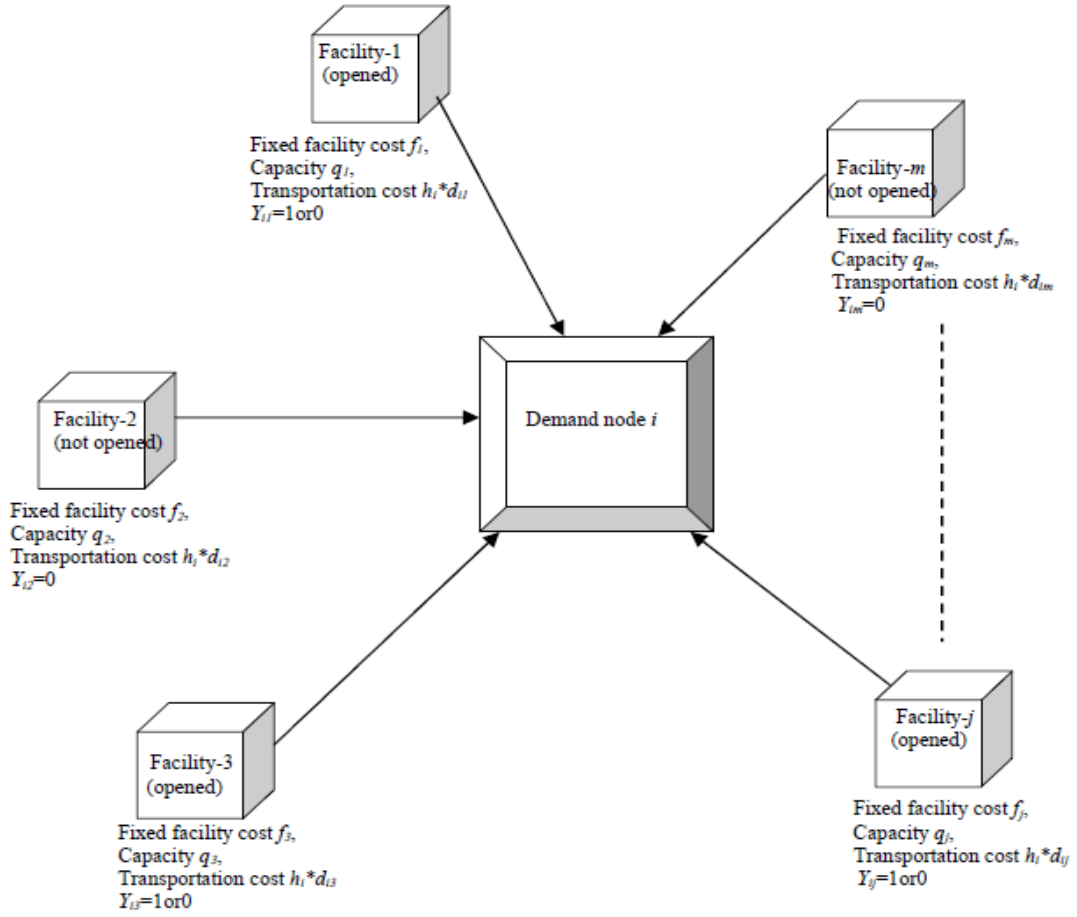


Figure 2.1: Single Source CFLP with uniform facility cost

Authors in [104] used the Lagrangean relaxation to establish relationship between fixed cost and assignment cost. they also claimed that optimal solution of relaxed problem is not a solution to the original problem.

Capaciatated Clustering facility Location Problem

For location models two distinct but relative objectives need to be determined: (1) where the facilities will be located (2) and then various demand nodes (customer regions) must be allocated to those facilities [105] [106]. The goal of CCFLP is to minimize dissimilarities between all items of cluster to its *centre*. The corresponding mathematical formulation is:

$$minimize \quad \sum_{i \in I} \sum_{j \in J} \left| a_i - z_j \right|^2 Y_{ij} + \sum_{j=1}^{n} X_j f_j \qquad (2.8)$$

Subject to constraints specified in metric UFLP and two additional constraints:

$$\sum_{i \in I} Y_{ij} = n_j, \quad \forall j \in J \tag{2.9}$$

$$\sum_{i \in I} a_i Y_{ij} \leq n_j z_j, \quad \forall j \in J \tag{2.10}$$

Where z_j is the centroid of j^{th} cluster, n_j is the number of demand nodes in j^{th} cluster such that $n_j \in N$ and a_i is the geometric position of i^{th} node in a two dimensional space. Constraint (2.9) specifies the number of demand nodes in each cluster and constraint (2.10) ensures that the centroid of each cluster is located at its geometric center in 2-d space. An illustration of above model is given in figure 2.3.

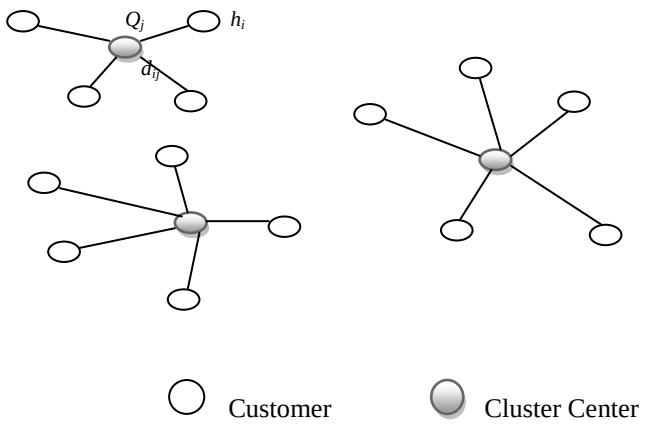


Figure 2.2: Capacitated Clustering Facility Location Problem

Solving exactly the CFLP is a difficult task. For variable number of facilities, this problem is NP-hard. So to solve it efficiently some heuristics have been investigated for getting some good solutions in acceptable time. Clustering is one of them which plays a significant role in finding optimal locations and allocations as well [107][108].

2.6 Metric Fault Tolerant Facility Location Problems

Fault-tolerant solutions are designed in order to provide protection against failures. For example caches are replicated in distributed networks to obtain resistance against caches becoming unavailable due to any possible reason [20][109]. This creates another

classification of the location model called fault tolerant FLP (FTFL). In FTFL a request node j having demand d_j is assigned to n distinct facilities in contrast to one facility. This assignment to multiple facilities results into FTFL. In FTFL, if the nearest facility fails to serve, the other facilities which are assigned may be used to serve the demand node. FTFL has observed its application for critical services.

In FTFL, the facilities have probability of failure to be p and each facility is required to provide service a request node j with probability at least q_j. Also each request node is to be allocated to r_j distinct facilities. Here the assignment cost for demand node j is distances from j to r_j. The aim is to allocate each request node j to r_j open facilities while minimizing the total cost. Authors in [20] have also considered FTFL for k-median problem with additional restriction. Jain and Vazirani in [53] have obtained a $4-$performance guarantee for FTFL using a langragean relaxation technique. Jain et. al. [53] have considered uniform requirement for all demand nodes.

Shmoys et. al. in [56] have given the 3.16 approximation algorithm for FTFL using the filtering technique of [110]. Furthermore, Chudak and Shmoys [92] used findings by [54] and and suggested LP rounding based $(1 + 2/e)$ – performance guarantee. This was further improved by Sviridenko [111] by devising a 1.58 approximation algorithm. Jain and Vazirani [58] have devised a combinatorial $3-$approximation algorithm using primal-dual approach. Mettu and Plaxton [112] were successful in devising an algorithm that gives the same performance guarantee and has linear time computational complexity. Authors in [113] gave a greedy heuristic with an approximation factor of 1.61 which was subsequently extended in [90] by giving a 1.52 approximation algorithm.

Authors in [114] have given a randomized LP-rounding $1.725-$approximation for same problem. This 1.725 is obtained as a result of usage of dependent rounding technique of A.

Srinivasan [115] and novel clustering method that does not require clusters to be disjoint. Following figure gives the recent findings of approximation algorithms for FTFL:

Figure 2.4 Approximation results for FTFLP

Jain and Vazirani	2000	$3 \ln max_j r_j$	Primal-dual
Guha et al.	2001	4	LP-rounding
Swamy, Shmoys	2008	2.076	LP-rounding
Byrka et al.	2010	1.7245	LP-rounding

There exist no primal-dual algorithm with constant ratio for FTFL.

2.7 UFLP with Penalties

Each request nodes is always allocated (assigned) to facilities so that total incurred cost is minimized. There exist a case when all request nodes are not allocated to a facility [21] [116]. In such case, any request node node *j* that remains unallocated incurs a cost known as penalty p_j . Such model is called facility location model with penalties (FLPWP), similar to UFLP except that not all request nodes are necessarily allocated. In FLPWP, a request node *j* may either receive the service by an open facility or gets rejected with penalty p_j .

FLPWP was introduced in [116], who proposed an algorithm with 3-perfomance ratio. Later, [117] used greedy algorithm for UFL and gave an approximation ratio of 2. This result was later improved by Xu and Xu [118] to 2.736 using LP-rounding. These findings were further improved by Geunes et al. [119] to 2.056-approximation algorithm.

Du, Lu and Xu [21] gave another extension under FLPLP called *facility location problem with submodular penalties* (FLPSP). In FLPSP, the penalty cost is a monotonically increasing submodular function $h(\cdot)$ defined on the set *D* of request nodes. FLPSP was first introduced by Hayrapetayan [120]. They presented $(1 + \gamma)$ approximate algorithm where γ is the performance guarantee of linear programming based approximation algorithm. Thereafter authors [121] offered an efficient algorithm with approximation ratio of $(1 + \varepsilon)(1 + \gamma)$.

Another variant of FLPWP is Multi level facility location with penalties (MLFLWP). According to Byrka [122], in k level UFLP with penalties, set C of request nodes and another set $F = \cup_{t=1}^{k} F_{l_t}$ of facilities is given in the demand plane. There exist k different types of facilities. Now each set F_{l_t} contains all facilities on level t and the sets F_{l_t} are pair wise disjoint. Each request node is connected to exactly one facility at each of k levels, otherwise a penalty p_j incurs for request node j. in MLFLWP, a request node j is connected to $i_1, i_2 \ldots . i_k$ where i_t is an open facility at level t. Here the goal is to minimize the sum of cost of opening the facilities, connection cost and the total penalty cost. If all penalties are same i.e $j_1, j_2 \in C$ we have $p_{j_1} = p_{j_2}$, it is classified as uniform version of the problem.

This MLFLWP was initially handled by Asadi et. al. [123] by presenting a 4-approximation algorithm. This was achieved by extending LP-based algorithm for UFLWP by Xu and Xu [118] to *k*-level. This is the best algorithm for MFLPWP till date.

The work on UFLPP extended the LP- based techniques like LP-rounding, primal-dual and dual-fitting. It was used to approximate UFLP (without penalties) after incorporating penalties. Xu and Xu used primal-dual technique of [124] followed by greedy local search to handle penalties. The local search technique of Chudak-Williamson (henceforth referred to as CW) [89] was extended further for uniform capacities and Pal-Tardos-Wexler (henceforth referred to as PTW) [78] in case the capacities are not uniform. Gupta and Gupta in [125] present (6+ǫ) factor for Uniform Facility Location Problem with Penalties (UnifFLPP).

2.8 UFLP with Dynamic Demands

There exist many parameters which may change during the lifespan of location modelling. Some of these parameters may be demand, travelling distance and installation costs [126]. These varying parameters have been addressed by various researchers in their work. Some of the work that considered change in travel time e.g., [127] [128] [129] [130] [131] and [23]

the availability of the facility for service e.g., [132] [133] [134], and the number of facilities to be sited.

In line to work by researchers, some researchers have relaxed the static demand assumptions in *UFLP* and *CFLP*, and developed models for *dynamic* FLP. Here the objective is to identify the locations of a subset of facilities to open in order to minimize the total cost for serving the customer demand for a specific duration of time. Dynamic demand was addressed in (e.g., [135] [136], [137]). Work done by Roy et. al. [13] is a milestone in the area of UFL with dynamic facility. Bean, et al. [138] formulated a deterministic problem for the uncertainty in demand. Later, Lim et. al. [139] and Canel et. al. [140] proposed methods for solving this problem for capacitated facilities.

While handling stochastic demand and congestion in facility location, two main factors are considered: (i) the service costs; and (ii) the quality of service. In such case, the objective is to maintain a balance between these two. Service cost is related to fixed cost for opening of facilities and the cost of providing these services to the demand sites. It is observed in the literature that there exist a relation between service costs and service quality ([141], [142], [143], [144], [145]). Authors in the literature focus on minimizing the service cost while maintaining the service quality up to a minimum set level ([146], [147], [148]). Berman and Krass and Boffey et al. [149] provided great reviews on this class of problems.

Majority of the literature have considered location of temporary deport for ambulance vehicles to implement such cases ([150], [151], [152], [153], [154]). However there exist studies on location of temporary relief facilities for distribution of various services. Tzheng et. al. in [155] and Lin et. al. [156] have considered integrated decisions of facility location, supply delivery and vehicle routing. Major focus in such allocation problem is to minimizing cost and maximizing serviceability in order to have effective response planning. These studies have also considered capacity limitations for located temporary facilities.

Dynamic nature of the demand is also considered in this model of disaster management. Authors also propose algorithms that allow the repositioning of facilities so that a minimum coverage standard is maintained. Lin et. al. in [156] considered the total number of facilities is fixed and does not vary with time, two other studies have considered it to be a time dependent decision variable.

In ILP formulation of this problem, we use the same variables described above and introduce a new indicator variable Z_j indicating whether a demand node is served or not. We set $Z_j = 1$ if the demand node j is selected for service and $Z_j = 0$ if demand node j is rejected or not served. The total penalty for a solution (X,Y,Z) is denoted by H, i.e.,

$$H = \sum_{j \in J} h_j Z_j \qquad (2.11)$$

The following formulation describes the MFLP with penalties:

$$minimize \sum_i \sum_j c_{ij} Y_{ij} + \sum_i f_i X_i + \sum_j h_j Z_j \qquad (2.12)$$

Subject to following constraints:

$$\sum_{p \in P} Y_{jp} + h_j \geq 1, \qquad \forall j \qquad (2.13)$$

$$\sum_{p:p \ni i} Y_{jp} \leq X_i, \qquad \forall i, j \qquad (2.14)$$

$$Y_{jp}, X_i, h_j \in \{0,1\}, \qquad \forall i, j, p \qquad (2.15)$$

Now we relax above LP by relaxing constraint (5.15) as $Y_{jp}, X_i, h_j \geq 0$. We now define the dual of relaxed LP as follows:

$$maximize \sum_j \alpha_j \qquad (2.16)$$

Subject to following constraints:

$$\alpha_j - \sum_{i \in p} \beta_{ij} \leq c(jp), \qquad \forall j, p \qquad (2.17)$$

$$\sum_j \beta_{ij} \leq f_i \,, \qquad \forall i \qquad (2.18)$$

$$\alpha_j \leq h_j \qquad (2.19)$$

$$\alpha_j \,, \beta_{ij} \geq 0 \,, \qquad \forall i, j \qquad (2.20)$$

The only difference between MFLP and MFLPWP is the constraint (2.19). Therefore, a dual feasible solution can be constructed in the similar way without any penalty added. Just like previous cases, the dual variables stop increasing whenever one of the following events occur:

$\alpha_j = h_j$ or j gets connected to a permanent open facility. The primal solution is now constructed as we did in MFLP above, with one exception that the frozen request nodes are not assigned any arbitrary path. They may contribute towards opening facilities along one *central* path.

Using analysis as discussed in section above, we can prove the following theorem:

Theorem 5.2 Applying primal-dual algorithm to case when penalties are added for demand rejection, will also yield a 6-approximation of the optimal solution for MFLP.

2.9 Conclusion

We have presented approximation algorithm for the metric UFLP with parallel supplies In this chapter. Our algorithm achieves a 6-approximation guarantee for the MFLP. The work can be further extended to obtain a performance guarantee better than 6. We also prove the optimality of our solution in terms of running time. We further proved that the same approximation guarantee can be obtained for penalty version of the problem. Work can also be further extended for the capacitated version of the problem. There are no approximation algorithms for the capacitated version of MFLP. We are also working on extending our research to MFLP with time dependant penalties.

Chapter 3

3 Approximation Algorithms for UFLP with Delayed Demand Satisfaction

It is generally assumed during study of FLP that the service is to be provided to all the request nodes. A major drawback of this kind of formulation is that some remote clients or request nodes, also called outliers, can greatly affect the final solution which is generally disproportionate to what is expected. This particularly applies to min-max problems like k-centers, where a single request node which is far apart from the other nodes may force a facility to be placed in its neighbourhood. However, this effect can be reduced to some extent with min-sum formulations, but it is still possible if the remote clients are sufficiently far away. Such clients adversely influence the cost of the final solution. Moreover, the service level is also not good enough for most of the clients. In many applications of FLPs it is desirable that all of the request nodes must be serviced. But for many commercial applications, it may be crucial to ignore such remote customers from monetary point of view. Here we focus on a generalized FLPs where only a specific number of the request nodes are to be served. This problem is NP-hard in the sense that subset of request nodes are required to be chosen to avail the service. Here we discuss two different versions of the problem here:

i. FLP with Penalty (FLPWP)
ii. FLP with Time Dependent Penalty (FLPTP)

We propose generalized approximation algorithms with these additional constraints.

In this chapter we explain about how to optimally solve FLP in situations where outliers may be present. Our primary focus is to present various formulations of the FLPs so that remote request nodes can be handled efficiently. While designing approximation algorithms we also discuss the computational consequences of these new formulations.

FLP with penalties: A penalty p_j is associated with each request node j. For each request node it is to be decided whether to provide service from its closest facility or reject the service. If penalties are set to 1, we will obtain the standard formulation. A 3-approximation guarantee can be achieved in polynomial time for the FLPWP.

K-Median problem with penalties. This variant of FLP also involves associating a penalty h_j with each demand node j. As discussed in first chapter, in k-median problem we have to choose k centres to locate/open a subset of facilities. Every demand node is then allocated to the nearest center. The objective is weighted sum of distances between demand nodes to centres. In contrast, in the k-median with penalties, it is decided whether to connect a demand node to a center or declare it an outlier.The contribution to the objective function for a connected node j is its distance to related center. If demand node j on the other hand, is selected to be an outlier, the contribution to the objective function will be the penalty h_j . Similarly, a request node can be either allocated or declared as an outlier. The goal is to minimize cost consisting of installation costs, service cost of open facilities and penalties for outliers. We can obtain a performance guarantee of 4 times the optimal in polynomial time for k-median problem with penalties.

However, FLP in general sense with the penalty constraint is discussed. Its k-median version can be solved by applying a similar procedure.

Linear programming relaxations

LP relaxations are used for the FLPWP. We first introduce some basic notations used throughout as follows:

I = set of facilities to open

J = set of request nodes to be either served or penalized

f_i = facility opening cost corresponding to facility i

c_{ij} = connection/service cost from facility i to demand node j

h_j = penalty cost for rejection of service corresponding to demand node *j*

Decision Variables

X_i = 1 if facility *i* is open, 0 otherwise

Y_{ij} = 1 if request node *j* is served by facility *i* , 0 otherwise

Z_j = 1 if request node *j* is subjected to some penalty i.e. its demand is not met, 0 otherwise

We now give the LP formulation (LP-1) for FLPWP as follows:

$$\min \sum_i f_i X_i + \sum_i \sum_j c_{ij} Y_{ij} + \sum_j h_j Z_j \ , \qquad \forall ij \qquad (3.1)$$

$$Y_{ij} \leq X_i, \qquad \forall ij \qquad (3.2)$$

$\sum_i X_{ij} + h_j \geq 1 \ , \ \forall j$

(3.3)

$Y_{ij} , X_i , h_j \geq 0$ (3.4)

The dual of above LP is:

$$\max \sum_j \alpha_j \qquad (3.5)$$

Subject to following constraints,

$\sum_j \beta_{ij} \leq f_i , \ \forall i$ (3.6)

$\alpha_j \leq c_{ij} + \beta_{ij} \ , \ \forall ij$ (3.7)

$\alpha_j \leq h_j \ , \ \forall j$ (3.8)

$\alpha_j , \beta_{ij} \geq 0$ (3.9)

Primal-dual Algorithm for UFLP

We first explain the primal-dual algorithm for FLP by Jain and Vazirani in [53]. We brief their algorithm as it is the ground for FLPWP. This algorithm runs in two phases.

Algorithm 3.1 Phase 1 of Primal-dual Approximation Algorithm for UFLP

1. The first phase starts with initializing dual variables α_j and β_{ij} to 0 and then it updates

these variables at uniform rate in successive iterations.

2. If $\alpha_j > c_{ij}$, then β_{ij} is set to $\alpha_j - c_{ij}$.
3. As this process continues, check if any one becomes true:

   Event 1: $\sum_{ij} \beta_{ij} = f_i$, this is the case when total contribution of request nodes towards opening the facilities becomes equal to the installation cost and facility *i* is *paid for*. For all j with $\beta_{ij} > 0$ and j has not yet been directly connected, $dual\ variable\ \alpha_j$ does not increase anymore and *i* is said to be *service point* of *j*. Let facility *i* is *paid for* at time t_i.

   Event 2: In second case, the budget of a request node equals the service cost from request node to its connecting facility, i.e. $\alpha_j = c_{ij}$ and facility *i* is *paid for*. In this case, α_j stops incrementing and *i* is said to be *service point* of *j*.
4. The first phase terminates when all request nodes are assigned service points.

Phase 2: The second phase determines which facilities to open permanently.

Algorithm 3.2: Phase 2 of Primal-dual Approximation for UFLP

1. Arrange the facilities in increasing order of the time when they were paid for.
2. Next greedily pick first facility from the list, remove the facilities reachable from *i* by two directly connected edges and repeat the process.
3. A request node *j* is said to be *directly connected* if in the final solution there exists a facility *i* for which $\beta_{ij} > 0$ (*j* is now allocated to *i*). Facility *i* is now open.
4. On the contrary if there is no such facility for which $\beta_{ij} > 0$ in the final solution, then the request node *j* is either allocated to its service point or to the facility that caused the removal of its service point in the second phase. Such request nodes are called *indirectly connected.*

5. In figure 3.1, if $\beta_{i_4 j} > 0$, then i_4 is blocked by *j* otherwise it is blocked by *j'*.
6. Pick another facility from the list and repeat steps 1-4 until all the facilities are either permanently open or added to set of blocked facilities.
7. Report the final solution as the set of permanently open facilities.

Figure 3.1 illustrates *indirectly connected* demand nodes. Facilities $\{i_1, i_2, i_3, i_5, i_6\}$ are open where demand node j' is directly connected to i_1, i_2, i_3 and demand node j is directly connected to facilities i_5, i_6. As shown in figure the facility i_4 is closed. Moreover, if $\alpha_j \leq \alpha_{j'}$, then *j'* is said to be indirectly connected and vice-versa.

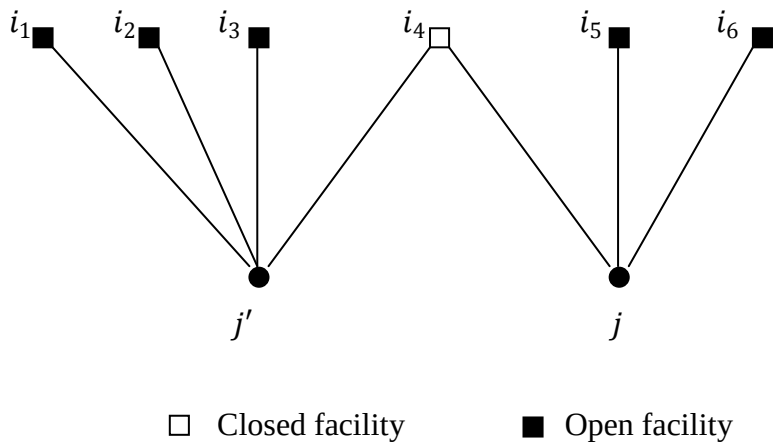


Figure 3.1 Example of indirectly connected demand nodes

Let *C** represents the service cost of OPT of above algorithm, Further assume that *F** and *OPT* represents setup cost and optimal solution respectively.

Lemma 3.1

$$C^* + 3.F^* \leq 3.\sum_j \alpha_j \leq 3.OPT$$

Proof: Recall that α_j is assigned to be 0 initially for all indirectly connected clients. Therefore only directly connected clients pay for the facility opening cost i,e., $\sum_i f_i = \sum_j \alpha_j$. Moreover, for every indirectly connected client the connection cost is at most $3.\alpha_j$ by using the properties of triangle inequality. Hence, $\sum_{ij} c_{ij} + 3.\sum_i f_i = \sum_{ij} c_{ij} + 3.\sum_j \alpha_j \leq 3.\sum_j \alpha_j \leq 3.OPT$.

We now see how primal-dual algorithm for FLP is adapted to address the FLPWP as well.

Primal-dual Approximation for FLP with Penalty

The above primal-dual algorithm is modified as follows:

Algorithm 3.3: Primal –dual Approximation Algorithm for FLPWP

1. The dual variables are initialized to 0.
2. The dual variable α_j is updated in the same way as in algorithm 3.1.
3. If node *j* does not have a connecting witness until α_j equals penalty h_j, then the value of dual variable α_j is *locked* at h_j. At this point a *timeout* occurs for request node *j*. The algorithm is now executed with the value of α_j fixed at h_j.
4. As soon as facility *i* gets paid for with $\beta_{ij} > 0$, the request node *j* becomes the service point. Similarly, run phase 2 as in algorithm 3.2 for this case also.
5. Some of the request nodes for which timeout occurs are selected for rejection (i.e. penalized) as follows: if timeout occurs for request node *j* and *j* has indirect connection to a facility, then it is selected for rejection.
6. During phase 2 of algorithm, all the remaining request nodes are connected either directly or indirectly to the facilities allocated to them.
7. Pick another facility in the greedy arrangement and repeat steps 2-6 until all the facilities are either permanently open or permanently closed.

8. Report final solution of set of permanently open facilities.

Let C^*, F^* and H^* represent service cost, installation cost, and penalty respectively in FLPWP. Following approximation can be proved about the solution of this algorithm:

Lemma 3.2

$$C^* + 3.F^* + 3.H^* \leq 3.\sum_j \alpha_j$$

$$\leq 3.OPT$$

Proof: Following two properties are used in the analysis of above algorithm:

1. $\beta_{ij} > 0 \rightarrow \alpha_j \leq t_i$,
2. i_1 is a service point for $j_1 \rightarrow \alpha_{j1} \geq t_{i1}$.

The first property is straightforward for modified algorithm of FLPWP. But the second one might not hold always. It becomes true if a timeout occurs for j_1. Second property is required in the case of indirectly connected request nodes only. Hence we obtain:

Theorem 3.1 A 3-approximation algorithm for the FLPWP is available.

Proof: Theorem 3.1 follows from lemma 3.1 and 3.2 respectively. For complete description of proof readers may refer [157] in which Jain and Vazirani proved above guarantee about their algorithm.

3.1 Proposed 4-approximation Algorithm for RAPTP

The RAPTP (Resource Allocation Problem with Time Dependent Penalties) is specialized URAP generally referred as uncapacitated facility allocation problems. This Work done is motivated by the work of Du.,Lu.,Xu [21] in which authors considered FLPs with submodular penalties and presented a $3-$approximation primal dual algorithm. This chapter considers that each unallocated demand point adds to penalty that increases as time passes

and is thus represented by function $x(t_i, p_i)$ where t_i and p_i are elapse time and priority of demand point d_i. As this problem has been considered for emergency service allocation, all demand points should be allocated to some facility or resource within some stipulated time limit beyond which it may lose its purpose. Thus penalty incurred by a demand point is considered till that threshold value only. Thus it is assumed that penalty contribution by a demand point remains constant after a specified threshold value. By exploiting the properties of time dependent penalties, a $4-$approximation primal-dual algorithm is proposed which is based on LP framework, and is the rst constant-factor approximation algorithm for RAPTP.

3.1.1 Background and Introduction

The challenge of finding an optimal resource has attracted many researchers who have their interest in location-allocation problems and complexity theory. From over many decades they are working on the issues involved in location-allocation and their applications. However, no polynomial time exact algorithm has been designed yet to solve location-allocation problems. Therefore, most of the research has been focusing finding efficient approximation algorithm. These are further combined with other approaches to give better approximation results such as randomized rounding, greedy strategy, evolutionary algorithms etc. [54] [113] [5] [158]. Among other variants of location problems, the metric UFLP is of maximum interest of researchers e.g. [44] [124] [159] [160] [161] [162]. Here metric means all straight line Euclidean distances that have been considered in the objective function must satisfy triangle inequality. We will be using the terms facility and resource interchangeably throughout the chapter.

We mainly focus on the resource allocation problem with penalties (RAPWP), which considers that not all the request nodes are allocated to a nearest resource and the demand of

a request node may be delayed or rejected due to many factors. Few factors may involve lack of resources, failure of the facility in demand, delay in request generation by a demand node or high connection cost leading to increase in overall cost etc. These request nodes may be allocated to the facility after some time or may not be connected at all.

Initially the problem was addressed in [124], who considered some additional parameters for cost reduction and obtained a 3- performance guarantee. Later Du, Lu and Xu considered in their paper FLP with sub-modular penalties ($FLPSP$) and gave a 3-approximation primal-dual algorithm using the features of submodular penalty functions [21]. This problem was first addressed by Hayrapetyan et al. [120]. They presented an $(2k + 1)$-approximation algorithm with $k -$level distribution tree. Authors in [121] presented an efficient algorithm where considered convex relaxations for combinatorial optimization problems with sub-modular penalties. The relaxations are obtained very naturally through a novel use of the Lovsz extension of a sub-modular function. Their work was further extended by Du, Lu, Xu [21] .

In the chapter we extend their research to resource allocation problems with time-dependent penalties and propose a 4-approximation algorithm for RAPTP by using a penalty function. Here, the penalty function depends upon many factors, one of which is the waiting time by a demand node before it is allocated to a resource. The RAPTP is an extension of RAPWP by considering penalty function to be a monotonically increasing.

In present work, an effort has been taken to extend a non-decreasing penalty function with respect to time i.e. when a request node is waiting for a service, the penalty incurred increases in proportion to the waiting time. Let $h(.)$ be the penalty function, then for two time periods t_1 and t_2, following condition will hold:

$$h(.)_{t_1} \geq h(.)_{t_2}$$

$RAPWP$ is mostly used to model the problems where a small portion of request nodes are at far locations and thus increasing the overall cost. Examples of such situations can be seen in commercial applications of UFLP where either facilitator or the request node or both have to pay a penalty if not connected.

For example, fire brigade service should reach the location within some stipulated time period otherwise it may loose its purpose. Providing service to a demand node after a threshold value may lose its purpose and in few cases service may not even be required. Therefore, in such cases, request nodes may not wait for a service for an indefinite time period. The penalty due to delayed service increases with time and thus penalty is in proportion to waiting time. The threshold value of a demand node depends upon demand node type described by its priority. For ex., for fire brigade request, the school demand node has much higher priority as compared to a less crowded area. Thus, for the same waiting time the penalty/damage incurred at school demand node will be higher in comparison to damage incurred at less crowded area. Thus if there are two demand nodes d_i and d_j where $priority(d_i) > priority(d_j)$, for same waiting time x the penalty incurred for demand node d_i will always be higher than penalty incurred by d_j.

In order to balance the trade-off between operating costs and the cost paid in terms of penalty due to waiting of service to request nodes, the resource allocation model should depend upon how likely the request node may be kept waiting for facilities. Practically all the request nodes may not have the same priority, which represents various considerable factors for allocation thus priority cannot be looked over. Here in this chapter we have considered priority of the demand point for allocation. Here we present a MIP formulation for $RAPTP$. Here, a primal-dual approximation model is presented which gives a 4-approximation for the $UFLP$ with time varying penalties. As discussed earlier that penalty for a demand node depends upon the priority of the demand node.

Problem Definition

Definition 3.1. For $RAPTP$ we assume a penalty function $h(.)_t$ that describes penalty incurred for a demand node at any time t on set J of demand nodes such that:

$$(1)\ h(.)_{t_2} > h(.)_{t_1} \ \forall\ t_2 > t_1$$

$$(2) h(.)_{t_2 - t_1} < h(.)_{t_3 - t_1} \ \forall\ t_3 > t_2$$

Above inequalities illustrates that the penalty for all demand point for any service increases with time. Following is the terminology used in this chapter:

Preliminaries

I =set of request nodes

J = Set of facility locations

c_{ij}^t = Connection / service cost from request node i to facility j at time t

f_j^t = Installation cost of facility j at the beginning of time period t

F^t =set of Frozen demand nodes at time t. These are the request nodes for which allocation is permanent and cannot be changed.

F^Δ = set of frozen demand nodes at the end of period Δ, where Δ represents the elapse time between a request has been made by a demand node and its allocation (or rejection)

L^t = set of locked demand nodes at time t. the allocation of these request nodes is tentative and may change during execution.

S^t = Set of penalized request nodes at time t

S^Δ = set of penalized request nodes at the end of time period Δ.

T^Δ = set of temporarily open facilities at the end of time period Δ.

O^Δ = set of permanently open facilities at the end of time period Δ. $O^\Delta = J \backslash T^\Delta$

h_i^t = penalty of request node i at time t

p_i^t = priority of demand node i at time t

w_i^t = waiting time of demand node penalty of request node i at time t. It is the elapsed time since the request has been made by a demand node and it gets the service.

Decision Variable

X_j^t = {1, if facility j is open at time t and remains open till Δ; 0, otherwise}

A_{ij}^t = {1, if facility j is open at time t and remains open till Δ; 0, otherwise }

Z_i^t = {1, if demand node i is penalized at time t ; 0 otherwise }

Now the aim is to identify set O of open facilities and a set S of penalized request nodes s.t every request nodes in S is either allocated to an open facilities or not allocated at all with some penalty. In other words, the request nodes which receive delay in service suffer a loss (penalty) and may be allocated to one of the nearest open facility. And the request node may remain unallocated if the delay reaches a $threshold$. In both situations, the request node will bear a penalty. We therefore define the total cost as:

$$\sum f_i + \sum c_{ij} + \sum h_i$$

We aim to find minimum cost solution including the penalty cost, and hence we also minimize the cost added due to delay in service. Therefore, the objective of our problem is two-fold. Such problems of providing in-time service (i.e. having critical service time requirements) have attracted the attention of researchers in past few years. We therefore, in this chapter focus on finding the optimal solution to the class of problems where along with minimizing the service cost, delay in service should also be minimized.

3.1.2 Proposed Approach

We first give the MIP formulation and LP relaxation, followed by defining the dual relaxed problem. IP solution and dual solutions are constructed simultaneously. This is inspired by

the fact that Primal-dual algorithms are extremely flexible and can provide good approximation bounds for location-allocation problems. The basic procedure is to set primal variables and raise the dual variable till an integral solution is obtained, which also satisfies the conditions for slackness. We present below a MIP formulation for RAPTP.

Linear Programming Relaxation

We define our objective function at time t as follows:

$$(RAPTP)\ \min\left(\sum_j f_j^t X_j^t + \sum_i \sum_j c_{ij}^t A_{ij}^t + \sum_{i\in I} h_i^t Z_i^t\right) \tag{3.10}$$

subject to the following constraints:

$$A_{ij}^t \leq X_j^t, \quad \forall\, i \in I, j \in J \tag{3.11}$$

$$\sum_j A_{ij}^t \geq 1, \quad \forall\, i \in I, j \in J \tag{3.12}$$

$$\sum_j A_{ij}^t \leq \sum_i Z_i^t, \quad \forall\, i \in I, j \in J \tag{3.13}$$

$$A_{ij}^t \in \{0,1\}, \quad X_j^t \in \{0,1\}, \quad Z_i^t \in \{0,1\} \tag{3.14}$$

Here inequality (3.11) ensures that a resource can be allocated only if it is open. Constraint (3.12) illustrates that a request node is allocated to at most one facility. Constraint 3.4 states that all allocated request nodes are subjected to some penalty (which will be counted since they have made a request). Now to obtain a linear program, we relax the integrity constraints in equation 3.5:

$$A_{ij}^t, X_j^t, Z_i^t\ 0\ s.t\ \sum_j A_{ij}^t = 1 \quad \forall\, i \in F^{\Delta}, where\ X_j^t = 1 \tag{3.15}$$

Primal-dual Approximation

We first consider that all opening costs are zero and all dual variables are also zero. The dual objective function for RAPTP in (3.10) can now be defined as:

$$\max \sum_{i \in I} \alpha_i^t \tag{3.16}$$

subject to constraints:

$$\alpha_i^t \leq \sum_j c_{ij}^t + \sum_j \beta_{ij}^t , \qquad \forall i, j \tag{3.17}$$

$$\alpha_k^t \leq c_{kj} + h_k^t \qquad \forall k \in S^t \tag{3.18}$$

$$\sum \beta_{ij}^t \leq f_j^t + h_i \tag{3.19}$$

$$\alpha_i^t \geq 0, \qquad \forall i \in I \tag{3.20}$$

$$\beta_{ij}^t \geq 0, \qquad \forall i \in I, j \in J \tag{3.21}$$

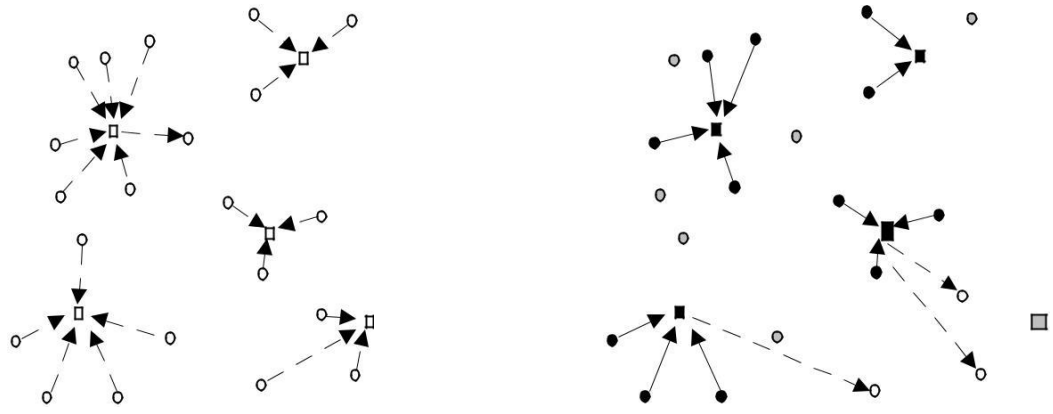
Here the dual variable α_i^t can be viewed as the total budget that request node i pays to avail the service and β_{ij}^t represents the contribution of request node i towards opening cost for facility j. Constraint 3.8 states that connection cost is first taken out from budget and the remaining budget is used towards facility opening cost. In inequality (3.18) S^t represents set of penalized request nodes and it implies that the total budget of penalized request nodes is less than their corresponding penalty. Constraint (3.19) imposes that all request nodes have to contribute for the opening cost for the facility to which they are connected. The primal-dual approximation algorithm that we present here runs in two phases, similar to idea proposed by Jain and Vazirani [163]. In phase one we determine which facilities to open permanently and in phase two we decide which demands nodes allocated to the facilities in O are penalized.

Description of Algorithm

Assume each request node places its request at time t and it has to be served within a specified time period $threshold_i$. Each request node has a waiting time w_i for which it waits

for service. The request nodes are subject to some penalty till they receive the service (i.e. during time w_i).

However, the penalty is added to the total cost till $threshold_i$ as explained further. The optimal solution would allocate a request node to one of the open facility if it serves all demand before threshold. This is accomplished in two phases of our algorithm. A dual feasible solution is identified in phase 1. At any time t_0, whenever a demand node requests for a service we set its request time $rt_i = t_0$. At this time the allocated facility is temporarily open. The request nodes which are allocated to temporarily open facilities are said to be locked and their allocation is tentative. The term frozen is used for permanent allocation. All locked demand nodes are frozen during phase 2. At the same time, all temporarily open facilities are also either permanently open or close during phase 2. Once a demand node is locked, it may be allocated to a permanently opened facility during phase 2. The other case can be that tentatively open facility is permanently open and thus link changes its state from locked to frozen as shown in Figure 3.1.



(a) (b)

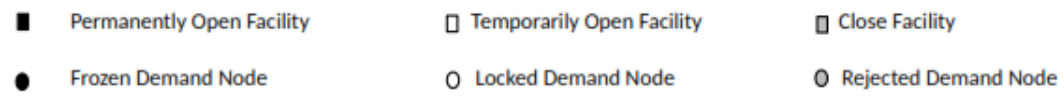


Figure 3 An Illusion of facilities after running proposed algorithm.

(a) Initialization in Phase 1 at time t_0. All demand nodes are initially awaiting service and unallocated (locked). All facilities are temporarily open (b) At time t_1, in Phase 2, all the facilities are either permanently open or closed. The demand nodes changed their state from locked to either frozen or rejected. The locked nodes of close facilities are now allocated to other facilities (shown by reverse arrows) which are open.

Phase 1.

For facility allocation problem with penalty under consideration, the penalty does not increase linearly with respect to time. For example for an ambulance request at some accident spot, the penalty incurred will rise rapidly in the beginning but after some time that penalty may become stable. Stability in penalty considers that cases where no more damage is possible for the demand point. For the example under discussion, stability may arise if victim dies after waiting for the service for some time. As discussed earlier in the chapter, another parameter that affects penalty is priority of the demand node. Therefore, penalty is a function of two variables: priority of demand node and elapsed time or waiting time of that node. Hence, we redefine our penalty function as:

$$h_i = f(p_i, w_i) \qquad (3.22)$$

Where, w_i is the elapse time since the request has been made by demand node i for the service and p_i is the priority for demand node i. We here assume that each demand node has an upper bound on the waiting time for which it can wait to avail the service. This is denoted by $threshold_i$, i.e. upper limit on waiting time for demand node i. It signifies that a demand node can be penalized till its $threshold$, after which the penalty will remain stable.

Algorithm 3 : Phase 1 of primal-dual Approximation for FLPTP

1. Phase one starts with initializing all dual variables to 0 while penalty for every demand node is also set to 0. Thereafter dual variable α_i is raised for all requests in

phase 1. Initially $t = 0$, As the time increases, the dual variable α_i is raised at a unit rate.

2. $F^{t_0} = L^{t_0} = \emptyset \; i.e.$ All facilities are assumed to be temporarily opened and none of the request node is locked or frozen during initialization.
3. For a new request at time $t_1 > t_0$, the demand node is added to the set S^{t_0} that was empty initially. So at any time t_1, we set:

$$S^{t_1} = S^{t_0} \cup \{i\} \; and \; request \; time \; rt_i = t_1$$

$$I = I - \{i\}$$

This is illustrated in Figure 3.1a.

4. Assign tentative allocation to demand node to its nearest facility and the link remains locked until i is frozen.
5. Update dual variables α_i^t and β_{ij}^t at the same rate after regular interval until $w_i > threshold_i$ to ensure the feasibility of obtained solution.
6. The algorithm declares a demand node allocated when $\alpha_i^t \geq c_{ij}^t$. It also maintains that $\alpha_i^t - \sum_j \beta_{ij}^t = \sum_j c_{ij}^t$ is updated until there is no unfrozen request node. When a demand node is frozen, we set:

$$S^{t_1} = S^{t_1} - \{i\}; \; F^{t_1} = F^{t_0} \cup \{i\}$$

7. Check if one of the events takes place:
    a. **Event 1:** a request node $i \in S^{t_1}$ is locked if $\sum_j A_{ij}^{t_1} = 1$ at any time t_1. At this time allocate demand node i to facility j and set: $L^{t_1} = L^{t_1} \cup \{i\}$ and $lt_i = t_1$, where lt_i is the locking time of node i. Whenever a demand node is locked , this allocation is tentative and may change during phase 2. The demand node i waits as long as $w_i < threshold_i$. If $\{\exists \; i \in L^{t_1} | \; w_i > threshold_i\}$, then $L^{t_1} = L^{t_1} - \{i\}$, then no more penalty is added to it. Therefore if a demand

node i is frozen at time t_1, the penalty $h_i^{ft_i+\Delta} = \{0|\ \Delta > 0\}$. This is illustrated in Figure 3.2. We now raise α_i for unfrozen request node only. This process continues until all demands become frozen.

b. **Event 2:** The demand node $i \in S^t$ should not be locked if $\sum_j A_{ij}^t < 1$. From time t_1 in time period t, penalty is calculated after regular interval until waiting time reaches $threshold_i$ using equation (3.23). Once waiting time reaches the limit of $threshold_i$, it represents the case where penalty can no longer increase. Thus once a demand node waits for $threshold_i$, the penalty remains constant thereafter and is shown by equation (3.24). The process of calculating penalty is performed until all demand points are frozen or removed from S and is calculated as:

$$h_i^t = \int_{t=rt_i}^{lt_i} h_i^t . p_i, \qquad \forall\, i \in S^t \tag{3.23}$$

$$h_i^t = \int_{t=rt_i}^{threshold_i} h_i^t . p_i \quad \forall\ i \in I \backslash F^t \backslash S^t \backslash L^t \tag{3.24}$$

8. During successive iterations, dual variable α_i^t is modified using slack variables until following condition holds:

$$\sum\nolimits_{i \in S^t} \alpha_i^t \leq h_i^t \mid S^t \subseteq I \tag{3.25}$$

9. Repeat steps 3-8 until condition (3.25) holds true or all demand nodes are frozen.

Phase 2

During phase 2, algorithm decides whether to permanently open or permanently close the facilities that were tentatively open in phase 1. To decide if facility j is to be permanently open, $\sum_j \beta_{ij}^t$ is considered i.e a facility is made permanent open once it has been fully paid for otherwise it remains tentatively open and may be permanently closed later. Following inequality is used to make the decision.

$$f_j^t - \sum_i \beta_{ij}^t \leq 0 \quad facilities\ are\ permanently\ open \qquad (3.26)$$

$$otherwise \quad facilities\ are\ tentatively\ open$$

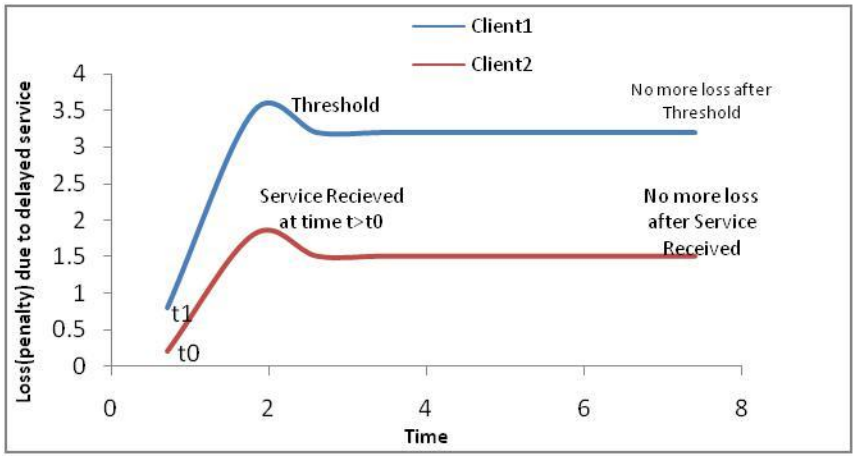


Figure 3.1: Penalty Due to Delayed Service.

In this Figure, x-axis represents the time values and y-axis represents the penalty. when time exceeds a threshold, the request node will be rejected and no further penalty will be imposed

To further find which other facilities are to be permanently open, we follow simple steps:

Algorithm 3: Phase 2 of Primal-dual Approximation for FLPTP

1. All tentatively open facilities are arranged in list according to non-decreasing order of $f_j^t - \sum_i \beta_{ij}^t$.
2. From each element in the list, we find set of dependent facilities. Two facilities j_1 and j_2 are said to be dependent $d(j_1, j_2)$ if there exists some demand point i for which $A_{i_1} > 0$ and $A_{i_2} > 0$. Such demand node is said to be connected indirectly.

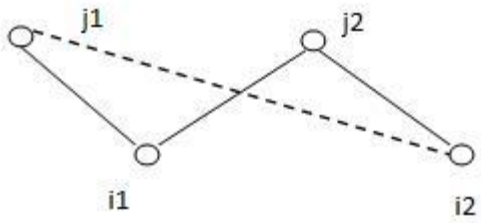


Figure 3.2: Dependent facilities

(3) Consider first facility m in the list. Now for each facility m find all dependent facilities. Start scanning the list from backward direction until facility is permanently open or closed. Remove the last dependent facility link in the list and allocate it to facility m. The process is repeated until condition in (3.26) becomes true.

(4) Now facility m is permanently open and all links to facility m are frozen. If (3.26) does not become true, the facility m is closed and links are frozen to neighboring facilities.

Now by the end of phase 2, all facilities are either permanently open or closed. Similarly all demand points are frozen to the corresponding facilities (1b).

3.1.3 Analysis

We consider each cost individually and then obtain the total cost by adding all costs. To begin with, let us assume that the number of times the penalty for demand node i is calculated is denoted by n_i. Clearly, $n_i = threshold_i \ / \ \Delta$.

Lemma 3.2.1: Proposed algorithm is solvable in polynomial time.

Proof. It is proved by demonstrating that next event takes place in polynomial time.

- **Case 1**: Let t_1 be time at which event 1 occurs. Then next closest time t_2 of event 2 will be given as $\max_{i \in I \setminus F}\{threshold_i \ / \ \Delta\}$
- Case 2: Time t_2 of event 2 occurs again in $\max_{i \in I \setminus F}\{n_i\}$ time.
- Case 3: To calculate time t_3 for event 3 above, we need to maintain inequality (3.9) in the dual objective function.

Let SOL be the solution to $RAPTP$. F_{sol} , C_{sol} and P_{sol} represent opening cost, connection cost and penalty respectively.

Lemma 3.2.2.

$$F_{sol} = \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} \beta_{ij}^{t} = \sum_{j \in O^{\Delta}} f_j^t = \sum_{i \in S^{\Delta}} \alpha_i$$

Proof: we know that only the allocated request nodes will contribute to the facility opening cost. Therefore, $\sum_{j \in O^{\Delta}} f_j^t = \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} \beta_{ij}^{t}$. where, the set of penalized and allocated request nodes i.e that belong to set $I \backslash F^{\Delta} \backslash S^{\Delta}$ do not contribute to the facility opening cost.

Lemma 3.2.3. At the end of phase 1, connection cost of any demand is at most $3\alpha_i$ $i.e$

$$C_{sol} \leq \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} c_{ij}^{t} \leq 3. \sum_{i \in F^{\Delta}} \alpha_i$$

Proof. This holds true if i is frozen and is assigned to a permanently open facility in O, otherwise i is assigned to j. If we consider j' to be an open facility that causes i to freeze. As described in Phase 2, j and j' must be dependent. Let us assume that there is a demand node k that is assigned to both j and j'. Let $t_{j'}$ and t_j be the time at which j' and j were tentatively open respectively. Now, $c_{ij} \leq c_{jk} + c_{kj'} + c_{j'i} \leq 2.\alpha_k + \alpha_i$. Also, $\alpha_k \leq t_{i'} \leq \alpha_i$. Therefore, $c_{ij} \leq 3.\alpha_i$.

We know that connection cost is paid by both frozen and penalized request nodes that are allocated to an *open facility*. Therefore, $C_{sol} \leq 3. \sum_{i \in F^{\Delta}} \alpha_{i.}$

Lemma 3.2.4:

$$P_{sol} \leq \sum_{i \in F^{\Delta}} h_i^{ft_i - rt_i} + \sum_{i \in S^{\Delta}} h_i^{t - rt_i} + \sum_{i \in I \backslash F^{\Delta} \backslash S^{\Delta}} h_i^{rt_i + threshold_i}$$

Proof. $\forall$ request node $i \in I$, there will be three possibilities:

(1) $i \in F^{\Delta}$. Request nodes that are frozen and allocated to some facility or facilities at time ft_i s.t. $\sum_{j \in near(i)} A_{ij}^{t} = 1$, where $near(i)$ denotes the facility which is in the

neighbourhood of i. These request nodes are penalized till their request has been accepted.

(2) $i \in S^{\Delta}$. These are request nodes which are penalized and allocated to some facility or facilities at time t subject to condition in case 1. For such request nodes, penalty is calculatd for the elapsed time between the request made by i^{th} request node and the time at which it was allocated.

(3) $i \in S^{'}$ These are the penalized request nodes which remain unallocated till the $threshold$ reached i.e $\sum_{j \in O^{\Delta}} A_{ij}^{t} = 0$ For these request nodes, penalty is calculated from the request of i^{th} request node and its *threshold*.

Using these lemmas, we now present final result:

Theorem3.4.5. we proposed a 4-approximation algorithm for $RAPTP$.

Proof. As we know that total cost Sol is defined as:

$$Sol = F_{sol} + C_{sol} + P_{sol}$$

It follows from lemma 1 to 4 that:

$$\begin{aligned} Sol &\leq \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} \beta_{ij}^{t} + \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} c_{ij}^{t} + \sum_{i \in F^{\Delta}} h_i^{ft_i - rt_i} \\ &+ \sum_{i \in S^{\Delta}} h_i^{t - rt_i} + \sum_{i \in I \setminus F^{\Delta} \setminus S^{\Delta}} h_i^{rt_i + threshold_i} \\ &\leq \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} \beta_{ij}^{t} + \sum_{j \in O^{\Delta}} \sum_{i \in F^{\Delta} \cup S^{\Delta}} c_{ij}^{t} + \sum_{i \in F^{\Delta}} \alpha_i^{t} + \sum_{i \in S^{\Delta}} \alpha_i^{t} + \\ &\quad \sum_{i \in I \setminus F^{\Delta} \setminus S^{\Delta}} \alpha_i^{t} \\ &\leq \sum_{i \in F^{\Delta} \cup S^{\Delta}} \alpha_i^{t} + 3. \sum_{i \in F^{\Delta} \cup S^{\Delta}} \alpha_i^{t} + \sum_{i \in I \setminus F^{\Delta} \setminus S^{\Delta}} \alpha_i^{t} \\ &\leq \sum_{i \in I} \alpha_i^{t} + 3. \sum_{i \in I} \alpha_i^{t} \\ &\leq 4 \sum_{i \in I} \alpha_i^{t} \\ &\leq 4. OPT \end{aligned}$$

3.2 Conclusion

The above analysis shows that the time dependent penalties may slightly increase the approximation factor as compared to penalties that do not change with the increase in time. However, in this chapter we have tried to find the approximate solution using conventional primal-dual approach. We may further consider usage of randomized rounding and local search to improve the approximation factor in our future work.

Chapter 4

4. Approximation Algorithm for UPFLP

Priority facility location problem (PFLP) was initially proposed by Ravi and Sinha [164] as a Multi Commodity Facility Location Problem (MCFLP), another variant of the UFLP. PFLP differs from UFLP as a result of level-of-service requirement in PFLP. Each facility also has a non-decreasing opening cost function at the level-of-service. Each requesting node must be satisfied. The goal of the location problem is to minimize the total cost consisting of opening cost and connection cost. Mahdian [165] offered a 3-approximation algorithm based on primal-dual for PFLP. Thereafter a primal-dual based 3-approximation algorithm for the stochastic PFLP has been presented by Li, et al. [24]. Using greedy augmentation, it was improved to 1.8526 which is the best ratio for PFLP and its stochastic version.

In PFLP, each requesting node has a level-of-service requirement. The level of service is indicated by an integer value $i|\ 1 \leq i \leq k$. Each facility also has a non-decreasing cost function in its level-of-service i.e $f_i(i) \geq f_i(j)|\ i \geq j$. The objective here is to construct a minimum cost model where requesting node j having priority p_j gets required service by a facility whose level-of-service is lower bounded by p_j. In order to understand the PFL, each priority level may be considered as a commodity. This association of a commodity with each priority level results in not linear facility cost functions. Therefore PFL can be considered as a special case of MCFL. Authors in [166] have studied priority Steiner tree with level-of-service requirements.

Priority algorithms can be classified based on whether ordering changes through the lifecycle of algorithm:

1. *Fixed Priority Algorithms* fixes the ordering prior to consideration of any input which remains same throughout.

2. *Adaptive Priority Algorithms* unlike fixed priority algorithm, permit to specify a new ordering after each input is processed. Updated ordering depends upon already considered inputs. This ordering and irrevocable decision depends on the current configuration i.e only on facilities which have already been considered.

We however focus on the first class of priority algorithms i.e. fixed priority algorithm.

Recall the definition of *PFLP* from Chapter 1, each demand node *j* has a *level-of-service* $p_j = \{1, \dots, K\}$ for some integer K and each facility *i* has a non-decreasing cost function $f_i(p)$ that specifies the installation cost of facility with priority of service *p*. The objective is to open each facility *i* with *level-of-service* say s_i, this is followed by assignment of request node j to a facility $i = \varphi(j)$ so that the total opening costs represented by $\sum_{i \in I} f_i(s_i)$ and connection costs represented by $\sum_{j \in J} c_{\varphi(j)j}$ is minimized.

In this chapter we see how the primal-dual algorithm for UFLP discussed in Chapter 3 gives a 3-approximation algorithm for PFLP as well. The LP formulation and algorithm has been presented below.

ILP Formulation for PFLP

PFLP can be formulated as :

$$min. \sum_i \sum_j c_{\varphi(j)j} Y_{ij} + \sum_i \sum_{s=1}^{K} f_i(s_i) X_i^s \qquad (4.1)$$

Subject to following constraints:

$$\sum_i Y_{ij} \geq 1\,, \qquad \forall j \qquad (4.2)$$

$$Y_{ij} \leq \sum_{s=p_j}^{K} X_i^s\,, \qquad \forall i, j \qquad (4.3)$$

$$Y_{ij}\,, X_i^s \in \{0,1\}, \qquad \forall i, j, s \in \{1, \dots, K\} \qquad (4.4)$$

Constraint (4.4) can be relaxed by replacing it with the constraint $Y_{ij}\,, X_i^s \geq 0$.

The dual of above relaxed LP is:

$$max. \sum_{j} \alpha_j \tag{4.5}$$

while having following constraints:

$$\beta_{ij} \geq \alpha_j - c_{ij} \,, \qquad \forall i,j \tag{4.6}$$

$$\sum_{j:p_j \leq s} \beta_{ij} \leq f_i(s) \,, \qquad \forall i, s \in \{1, \dots, K\} \tag{4.7}$$

$$\alpha_j \,, \beta_{ij} \geq 0 \,, \qquad \forall i,j \tag{4.8}$$

The dual variable α_j can be understood as budget of requesting node *j* towards opening cost of connected facility. This demand node contributes an amount of at most $\alpha_j - c_{ij}$ to opening facility *i* with priority of service p_j or higher. The sum of contributions of all demand nodes towards opening a facility *i* with priority *s* cannot exceed $f_i(s)$. Using these facts, we now explain the two-phase primal-dual algorithm for solving priority FLP. An important point here is that during first phase of the algorithm same facility might open with different priorities of service.

Primal-dual Algorithm for PFLP

The algorithm starts with initializing the dual variable α_j to 0 for each demand node *j*. All demand nodes are unallocated and all facilities are closed initially.

Phase 1:

Algorithm 4.1: Phase 1 of Primal-dual Approximation for PFLP

1. Similar to the procedure described in Chapter 3, start increasing the dual variable (i.e. budget) α_j of all demand nodes that are unallocated simultaneously at a uniform rate. Thus at any time *t* the value of $\alpha_j = t$ for any unallocated demand node *j*.
2. At any point of time, the contribution β_{ij} of demand node j towards facility *i* at

priority level *s* is defined as $\beta_{ij} = \max(0, \alpha_j - c_{\varphi(j)j})$ if $s \geq p_j$, and 0 if $s < p_j$.

3. Keep increasing the budget until any one event takes place:
    a. For a facility *i* and priority level *s*, the total contribution $\sum_{j:p_j \leq s} \beta_{ij}$ towards opening facility *i* at priority level *s* becomes equal to the opening cost $f_i(s)$. At this point of time, *i* is opened with priority level *s* , allocated to all unallocated demand nodes *j* in such a way that $p_j \leq s$ and $\beta_{ij} > 0$ and the dual variable is frozen (i.e. kept fixed at that value) corresponding to all demand nodes that are now allocated.
    b. For an unallocated demand node *j* and a open facility *i* at priority level $s \geq p_j$, the budget α_j becomes equal the connection cost $c_{\varphi(j)j}$. Thereafter, node *j* is assigned to facility *i.*
4. Phase 1 terminates when dual variable α_j can no longer be updated.

Phase 2:

Phase 2 will determine which facilities are to be opened permanently according to their priority level-of-service.

Algorithm 4.2: Phase 2 of Primal-dual Approximation for PFLP

1. Arrange the facilities that are open during phase 1 in non-increasing order of their priority of service level (if however, a facility *i* is opened multiple times at different priority levels, then multiple copies of *i* are kept one for each priority level).
2. For each facility *i* in this arrangement, if ∃ open facility *i'* and *j* contributes towards both *i* and *i'*, then close *i* (but is *i* is open multiple times, other copies of *i* will remain open).

3. At the end, allocate demand node j to the nearest open facility with *level-of-service* $s \geq p_j$.

Analysis

Let *C* be connection cost, *F* represent facility cost and *OPT* be optimal solution obtained from above algorithm.

Lemma 4.1

$$C + 3.F \leq 3.OPT$$

Proof:

1. From the description of algorithm, we understand that the values of α_j and β_{ij} will be a feasible solution at the end of the algorithm for primal linear program of problem. Therefore by duality theory, $OPT \geq \sum_j \alpha_j$. Hence, proof of above lemma requires to prove only $C + 3F \leq 3.\sum_j \alpha_j$.

   The proof is similar to the one described in Chapter 3 and also same as that proposed by Jain and Vazirani [58]. The demand node *j* is connected directly if there is a open facility *i* with priority level $s \geq p_j$ in such a manner that either $\beta_{ij} > 0$ or *j* is allocated to *i* during phase 1and $\beta_{i'j} > 0$ for all *i'*.

2. For each directly connected demand node j there is exactly one such facility *i* as during phase 2 the algorithm closes all remaining facilities and at most one is left open towards which j has a contribution. For each demand node j which is not connected directly, there exists is a facility *i* to which *j* is connected in phase 1 of the algorithm. This means that *i* is opened in the first phase of the algorithm with priority level $s \geq p_j$. Since it is closed during phase 2, there must be another facility *i'* prior to *i* in the arrangement (i.e. *i'* is opened with priority $s' \geq s$) and a city *j'* has a positive

contribution towards *i* and *i'*. Assume that *i* was opened at time *t* during phase 1. We must have $\alpha_j \geq t$, because *j* rises its budget (α_j) until the time $\max(t, c_{ij})$.

3. On the other hand, since *j'* contributes towards *i*, we have $c_{ij'} < t$. Similarly, we must have $c_{i'j'} < t$, otherwise *j'* will stop rising its budget at a time before $\max(t, c_{ij'})$ and thus cannot have a positive contribution towards both *i* and *i'*.

4. Also, $c_{ij} \leq \alpha_j$, since each demand node must be able to pay for its connection cost to the facility it is allocated. Therefore, we have,

   $c_{i'j} \leq c_{i'j'} + c_{ij'} + c_{ij} \leq 2t + \alpha_j \leq 3.\alpha_j$ (4.9) (triangulation inequality)

   A clear visualization of equation (4.9) is shown in figure 4.1.

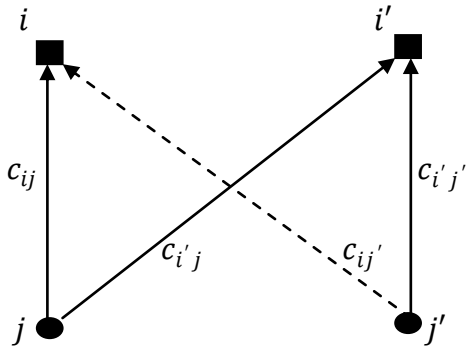


Figure 4.1 Pictorial Representation of equation (4.9)

   As explained earlier, *i'* is prior to *i* in the arrangement, therefore it must be opened with priority level $s' \geq s \geq p_j$. Hence, the connection cost of *j* in final solution is also bound by $3.\alpha_j$.

5. Finally, we observe that for each opened facility *i* with priority level *s*, each demand node that contributes towards its opening with this priority is directly connected. Thus, the total contribution from directly connected demand nodes towards opening *i* with priority level *s* is equal to the opening cost $f_i(s)$ which is either 0 or $\alpha_j - c_{ij}$. Therefore, the cost of facility is bounded by $F \leq \sum_{j \in J_{dir}} (\alpha_j - c_{ij})$, where J_{dir} denotes the set of demand nodes that are connected directly.

From 1 to 5 we conclude that,

$$3.F + C \leq 3.\sum_{i}\sum_{j \in J_{dir}} (\alpha_j - c_{ij}) + \sum_{i}\sum_{j \in J_{dir}} c_{ij} + \sum_{j \in J \setminus J_{dir}} 3.\alpha_j$$

$$\leq 3.\sum_{j \in J} \alpha_j$$

$$\leq 3.OPT$$

Theorem 4.1 For any instance of PFLP, the algorithm described above is a 3-approximation to the optimal solution.

4.1 A $(3 + \varepsilon)$-Approximation Algorithm for UFLP with Critical Service Time Requirements

This section presents an approximation algorithm for "logistics systems" containing various stochastic factors such as demand and travel time etc. focus of this work is emphasised on providing in-time delivery to the request nodes while minimizing the overall cost. This includes travelling cost, service installation cost and the cost incurred due to delayed service. This problem is represented as Mixed Integer Programming (MIP). An approximation algorithm based on primal-dual is proposed to obtain the solution to the problem which is $(3 + \varepsilon)$ -approximation to the optimal solution.

For many real life services like express delivery, home delivery etc., the most significant concern is in-time delivery, failing which the request node and/or the service provider may suffer a considerable loss. While the loss incurred due to delayed service in case of express delivery and home delivery by food court is bearable, the same in case of emergency service may turn into fatal loss.

This chapter examines the factors involved in first two scenarios mentioned above where late delivery can cause loss to the service provider only (also known as *penalty*). The problem under consideration takes into account the following factors: (i) number of service providers (ii) no. of requests to each service provider (iii) service time (response time) (iv) waiting time. Response time is the elapse between submission of request and receiving of service. There exists a threshold for each service within which the request should be served. This is called as maximum permissible waiting time (PWT) or threshold for which a request node may wait.

The problem involves assigning the request node to a service provider and serving the request node within permissible waiting time while minimizing the total cost. Some other applications of this model include supply chain, health services, and courier services etc. Assumptions on arrival pattern, demand allocation and travelling time (or distance) are crucial for problem formulation and its solution. For example, the maximum response time for a request should not exceed its PWT as well as the travel time between the service provider and the request node. We mainly focus on the demand allocation problem within PWT considering that all the request nodes may not be allocated to nearest service provider and the service to a request node may get delayed. This delay in allocation may be due to many factors. Few factors may involve lack of resources to the service providers, failure of assigned server or some unfavourable conditions (e.g. rain, storm, high traffic etc.). Now the request nodes getting delayed service may either accept the delayed service with some penalty imposed on the service provider (e.g. deduction in the demand cost) or may reject the service at all (in which case the server will bear the entire loss due to non-fulfilment of request). The aim of this model is minimizing the overall cost consisting of installation cost, transportation cost and penalty cost.

The critical service time location-allocation problem is combination of FLP and Travelling Repairman Problem (TRP). TRP is modified Travelling Salesmen Problem (TSP), in which tour for individual repairman (or repairmen in case of k-TRP) is determined. TRP ensures that overall waiting time for customers is minimized. In general, the location problem determines the location of facilities in a network and allocation of facilities. Here, the aim is minimizing the total weighted service cost. Aboolian et. Al. [167] has proposed an exact algorithm to minimize the service allocation cost. Although very few researchers [168] have focused on their research for minimizing the waiting time of customers. Related work on other spatial location problems with hard service time constraints can be found in the work of Charikar et. Al. [169], who considered some additional parameters of cost reduction and gave a 3-approximation primal-dual algorithm.

As already discussed in previous chapter that UFLP with penalties was also first introduced by Charikar et al. [116]. Subsequently, authors in [163] showed that their greedy algorithm for UFLP could also be tailored to UFLP with penalty with the approximation factor of 2. This ratio was further improved by authors in [161] [158] to 1.8526 using LP rounding and primal-dual. Later, Geunes et al. [119] in their paper presented a 2.056-approximation algorithm for UFLP with penalties. A related application of such problems lies in medical services. A systematic planning is required to handle such problems at large scale [170]. For planning of emergency medical services, Beraldi and Bruni [171] proposed an exact algorithm for minimizing the expected cost.

In small pick-up and delivery services as discussed above, service area (region) management is one of major concern. In such problems the aim is to minimize the travel time and service time which indirectly minimizes the total cost incurred. Several techniques are in existence to solve network design and location modelling like IP, tabu search, variable neighbourhood search, Simulated Annealing, Genetic Algorithm and meta-heuristics etc. Approximation

algorithms for network model include LP rounding [51][172], greedy heuristic [54], local search [5] and primal dual method [58] etc.

The aim is to design a near optimal solution to the location problems having critical service time requirement using a greedy heuristic followed by primal-dual approximation. Every service has a threshold value within which every demand point may be able to get service. We also consider permissible waiting time for all request nodes where request node c_i may wait for a time not more than PWT_i to acquire his service. A MIP formulation is proposed for considered problem. Each request is assigned a priority to decide about which request node is to be served first. The priority is generally a function of the permissible waiting time and quantum of demand of the request generated. For request nodes having equal priority (i.e. ties), a greedy heuristic on travelling distance (and hence travel time) is used to make the decision.

In this chapter, section 4.1 formulates the problem and initializes the required parameters. Proposed algorithm has been discussed in section 4.2 while main theorem has been given in section 4.3. Section 4.5 is dedicated to concluding remarks with suggestions for future work.

4.1.1 Problem Formulation

The problem is first described with a list of notations, basic assumptions and then mathematical formulation. For instance, logistics system (such as food delivery services) contains request nodes and service centres. A request node sends a request to its nearest server with a predetermined response time requirement. The response time for this request node should not exceed predetermined limit (i.e. the PWT). The objective here is to assign a unique service provider to each requesting node which provides immediate service while minimizing the service cost. A penalty is imposed on the service center if it responds beyond PWT. As discussed above the decision of which request node to serve first for multiple requests is based on the priority of requested node (or request node).

Preliminaries

Following are the terminology used.

C = set of request nodes

S = set of service providers

m = Input size of algorithm, i.e. $j = |C| \times |S|$

d_{cs} = service cost from service provider s to request node c

q_c = demand of request node c

o_s = opening cost of service provider

A = set of request nodes allocated to or seized by a service provider

L = set of request nodes that have availed the service (these request nodes are *locked*)

P = set of penalized service providers, $P \subseteq S$

Γ = set of temporarily open services, $\Gamma \subseteq S$

Π = set of permanently open services, $\Pi \subseteq S$

h_s = penalty to service s

τ_{sc} = service arrival time from service provider s to request node c

rt_{cs} = request arrival time of request node c to service s

wt_c = waiting time of request node c

tt_s = travel time/service time of service provider s

N = total number of service providers

Decision Variables

$$X_s = \begin{cases} 1, & for\ open\ services \\ 0, & otherwise \end{cases}$$

$$Y_{cs} = \begin{cases} 1, & if\ client\ c\ is\ alloted\ to\ service\ provider\ s \\ 0, & otherwise \end{cases}$$

$$Z_s = \begin{cases} 1, & if\ service\ provider\ s\ is\ penalized \\ 0, & otherwise \end{cases}$$

$$T_{s_i} = \begin{cases} 1, & if\ service\ for\ request\ i\ can\ be\ satisfied\ within\ PWT \\ 0, & otherwise \end{cases}$$

The request satisfaction depends upon two factors: (i) travel time from source to request node (d_{cs}) and (ii) waiting time of requested request node (wt_c). That is, if $\tau_{sc} - rt_{cs} \leq PWT_c$, then the service provider is not subjected to any penalty, otherwise service provider will have to bear a penalty. After satisfying the request, service provider is required to report to the service center in order to handle upcoming requests. Therefore, travel time is doubled every time a request is handled. This is depicted in figure 1. Dashed lines represent the service provider supplying demand to request node c after receiving the request and solid lines represent the service provider reporting to service center after serving the request node.

We can now formulate our objective function as follows:

$$\min \sum_{s \in S} o_s X_s + \sum_{c \in C} \sum_{s \in S} d_{cs} Y_{cs} + \sum_{s \in P} h_s Z_s \qquad (4.9)$$

Subject to constraints:

$$\sum_{s} X_s \leq N, \qquad \forall s \in S, \qquad (4.10)$$

$$\sum_{s} Y_{cs} \leq 1, \qquad \forall c \in L, where\ X_s = 1, \qquad (4.11)$$

$$Y_{cs} \leq X_s, \qquad \forall s \in S, c \in C \qquad (4.12)$$

$$Y_{cs} \in \{0,1\}, \qquad \forall s \in S, c \in C \qquad (4.13)$$

$$X_s \in \{0,1\}, \qquad \forall s \in S \qquad (4.14)$$

$$\tau_{sc} > rt_{cs} \qquad (4.15)$$

Here constraint (4.10) sates that the number of open service providers should not exceed the number of available service providers. Constraint (4.11) ensures that a service provider can be allocated only when it is open. Constraint (4.12) states that a request node will either fully avail the service or not at all.

Constraint (4.13) illustrates that a service is either fully open or fully closed and constraint (4.14) states that the service will be provided only after receiving a new request. h_s is the penalty imposed on service provider, which depends on the priority of request. That is, higher priority request nodes impose more penalty to the service provider than the lower priority request nodes. Consider following two cases:

Case 1: Higher priority request nodes are those which have high demand and low PWT. Such request node may refuse to avail the service after its PWT. Therefore the service center has to bear the penalty which is equal to the sum of total profit being earned by satisfying that request and $2.d_{sc}$ i.e.

$$h_s = \{\alpha_c + 2.d_{sc} | Y_{cs} = 1\}$$

Case 2: Lower priority request nodes are those which have low demand and high PWT. Such request nodes may avail the service after PWT, but with a little penalty on the service center i.e. penalty is some fraction of the profit earned say $\beta.q_c$ (q_c is the amount of demand and profit is directly proportional to q_c), such that $\beta < 1$ i.e.

$$h_s = \{\beta.\alpha_c | \beta < 1\}$$

Considering these scenarios and specified constraints, we present an approximate solution for stated problem. This solution is inspired by the work of Jain and Vazirani [58] and Du,Lu,Xu [158], [21], [162] on Approximation algorithms.

4.1.2 Proposed Algorithm

The algorithm starts with some assumptions and initializing the problem variables in the objective function in eq. (4.9). Different cases are considered to depict the problem with different scenario. Our algorithm runs in two phases. Phase I finds a dual feasible solution and also gets a set of open (temporarily) service providers. The request nodes which are allocated to temporary open services are said to be *seized* and their allocation is tentative. The term *locked* is used for permanent allocation. All *seized* request nodes are *locked* during

phase II. Any temporary open service can become permanent and thus allocation changes from *seized* to *locked*.

Phase I

We first define the dual of the above primal program as follows:

$$maximimze \sum_{c \in C} \alpha_c \qquad (4.16)$$

Subject to constraints:

$$\sum_{c} \lambda_{cs} \leq o_s \qquad (4.17)$$

$$\alpha_c - \sum_{s} \lambda_{cs} \leq \sum_{s} d_{cs} A_{cs} \qquad \forall S, c \in C \qquad (4.18)$$

$$\alpha_c \geq 0 \qquad (4.19)$$

$$\lambda_{cs} \geq 0 \qquad (4.20)$$

Here λ_{cs} represents the dual adjustment variable or the contribution of request node c towards opening cost of service provider s. Constraint (4.17) states that all request nodes have to contribute for the opening cost of the service provider from where they are receiving the service. Constraint (4.18) states that budget is partly used for the connection cost while rest is used towards service opening cost. To minimize the cost of objective function in eq. (4.9), each request node is assigned to an open service in its service region only (i.e. to the nearest service provider).

Initialization and Assumptions. To determine the initial solution to the stated problem, traditional location-allocation methods are used. Every service provider has a fixed service opening cost. All dual variables are initially set to zero. Penalty cost for all service providers is initially zero and no services are penalized initially i.e. $P = \Phi$. All service providers are assumed to open temporarily. All request nodes are unallocated during initialization i.e.

A= Φ. and $L = \Phi$. Each server is assigned a service region within which it will respond to incoming requests. No server is allowed to move out its service area. Therefore, every request node will send a request to its nearest server. We call it the *promising region*. As shown in fig 4.1.1, the promising region for server will be the convex hull of the request nodes in its vicinity. Let $\widehat{S}$ be the *promising region* or *promising set* for request nodes in set $\widehat{C}$, $\hat{C} \in C$. For any service provider $S^{'}$ inside promising region $\hat{S}$, $c^{'}$ will lie in the service region of $s^{'}$ if and only if $d_{cs'} \leq cost(\hat{s})$, where $cost(\hat{s})$ is defined as: $cost(\hat{s}) = (o_{\hat{s}} + \sum_{c \in \hat{C}} d_{c\hat{s}})/|\hat{C}|$

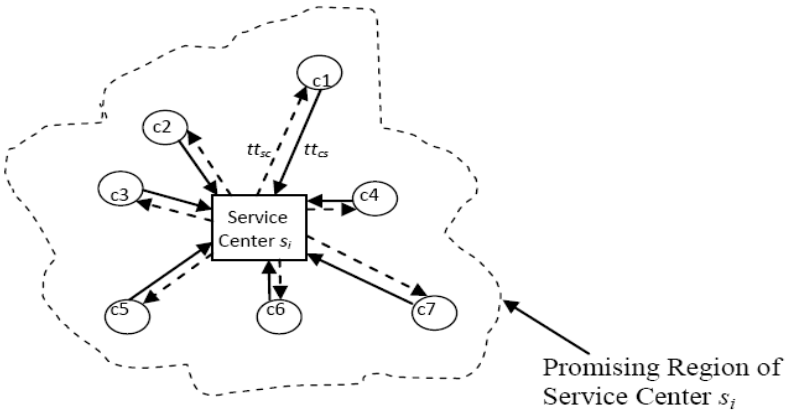


Figure 4.1.1: Illustration of Promising region for service center s_i

Now the requesting request node is tentatively allocated to its nearest service provider inside its promising region and it remains seized until it is locked. α_c is updated in successive iterations until $wt_c \geq PWT$. The algorithm declares a request node locked and allocated when $\sum_S Y_{cs} = 1$. It also maintains that $\alpha_c \leq \sum d_{cs} + \sum \lambda_{cs}$ and set $A = A \cup \{c\}$ is updated until there is no *unlocked* request node. To ensure the feasibility of the solution, primal and dual variables are updated accordingly.

Arrival of new request A greedy heuristic is used to tackle upcoming requests. A new request is assigned its nearest server according to its priority. The higher priority requests will be served first and then the lower priority ones in the same service region. Let there be

n requests arriving at time t, all these requests are in ascending order of their priority g_c. For more than one requests with equal priority, arrange them in descending order of travel time and then serve each request sequentially. Now for a new request from request node c to service provider s, we consider following cases to update the problem variables:

Algorithm 4.1.1 Updation of Problem Variables during Phase 1 of Primal-dual Algorithm for PFLPTP

1. If $Y_{cs} = 1$ and $wt_c \leq PWT$, then set $\tau_{sc} = t + wt_c$ and $T_s = 1$. Request node is allocated to service provider s and is *seized*. $A = A \cup \{c\}$ and $rt_c = t$, where t is the time of request. No penalty is imposed on s at this point. wt_c and tt_c are incremented at regular time intervals until $wt_c > PWT$.
2. If $\alpha_c \geq d_{cs}$ and $\sum Y_{cs} = 1$, then lock request node c and set $L = L \cup \{c\}, A = A - \{c\}, h_s = 0$. At this time, no penalty is imposed on service provider as request of request node is satisfied before waiting time of request node c i.e. wt_c exceeds the PWT.
3. If $wt_c > PWT$, then the service provider has to bear a penalty i.e. $h_s = 2.d_{cs} - \beta.q_c$ and $A = A - \{c\}$. During successive iterations, dual variable α_c is updated by a small factor say Δ and will continue to update until $\alpha_c - \lambda_{cs} < d_{cs}$ holds true.

Phase II

During Phase II, the algorithm decides whether to permanently open or permanently close a service provider that was temporarily open in Phase I.

A service is opened permanently if all of its opening cost is fully paid off by the revenue of incoming requests (e.g. food orders in case of food services). Each request contributes λ fraction of opening cost to service provider (until the service provider is penalty free; once

penalized, the contribution of that request node will become zero as it will not pay for the delayed service). Following equation is used to make this decision:

1. If $o_s \leq \sum_{s|c \in \hat{C}, s \in \hat{S}} (1+\varepsilon)\alpha_c - d_{cs} - h_s$, service provider is temporarily open.
2. If $o_s - \sum_c \lambda_{cs} \leq 0$, service provider is permanently open.

To further determine which facilities are permanently open, we follow following steps:

Algorithm 4.1.2: Phase 2 of Primal-dual Approximation for PFLPTP

1. All temporarily open service providers are arranged in list I according to non-decreasing order of $o_s - \sum_c \lambda_{cs}$.
2. If request node c contributes towards s i.e $(1+\varepsilon)\,\alpha_c - d_{cs} > 0$ and $c \in C$, then c is directly connected to s. For each request node c, $\exists$ atleast one service provider that it contributes to, therefore:

$$\sum_{s \in \hat{S}} o_s \leq \sum_{c \in \hat{C}} (1+\varepsilon)\,\alpha_c - d_{cs} - h_s$$

3. Next we find the set of dependent service providers say s_1 and s_2. Two service providers are said to be dependent if there exist some request node c which is receiving service both from s_1 and s_2 i.e. $Y_{cs_1} > 0$ and $Y_{cs_2} > 0$. If a request node is receiving service from more than one service providers at the same time, then all such service providers become dependent.
4. Consider first service provider i in I. Now for each i, find all the dependent service providers. Make sure that each service provider in I is either permanently open or permanently closed. Now for each $i \in I$ if:

$$\sum_{i \in I} \lambda_i \leq h_i \qquad (4.21)$$

then give i the total allocation of incoming request and remove it from set I. Repeat

this step until eq. (4.21) holds true. Now i is opened permanently and all its request nodes are seized.

5. If eq. (4.21) does not become true for all requests arriving during a predetermined time period, then close s and all its request nodes are assigned to neighboring service providers.

Now by the end of phase II, all service providers are either permanently open or permanently close. Similarly all upcoming requests are either locked to penalize. We also define here the notion of *indirectly connected* request nodes. The request nodes which are allocated to any of the dependent

service provider will become *indirectly connected* for the other service provider. In other words, indirectly connected request nodes are those which are not directly contributing to service provider s. Therefore they do not satisfy the eq. $(1 + \varepsilon)\alpha_c \geq d_{c's}$ for any s, where c' is neighbor of c which is directly connected to s. Also c' does not have any neighbor of promising service provider.

4.1.3 Analysis

Let Sol be the final solution to our problem. Additionally, OC^*, SC^* and PC^* be opening cost, service cost and penalty respectively for the optimal solution.

Lemma 4.1

$$OC^* \leq (1 + \varepsilon) \sum \alpha_C \qquad (4.22)$$

***Proof*:** We know that only the locked request nodes will contribute towards service opening cost. Therefore,

$$OC^* \leq \sum_{s \in \pi} o_s$$

$$\sum_{s \in \pi} o_s = \sum_{s \in \pi} \sum_{c \in L \cup P} \lambda_{cs}$$

$$\leq \sum_{c \in L \cup P} (1 + \varepsilon)\alpha_c - \sum d_{cs}$$

$$\leq (1 + \varepsilon). \sum \alpha_c$$

The unlocked request nodes (i.e. the request nodes which belong to the set $C \backslash A \backslash L$) on the other hand will not contribute towards service opening cost.

Lemma 4.2

$$P^* = \sum_{S \in \pi} h_{LT_s} - rt_s + \sum_{s \in P} h_t - rt_s + \sum_{s \in S \backslash \pi \backslash P} h_{rt_s} + threshold \qquad (4.23)$$

Proof.

1. Services that are opened completely are allocated to some request node and have been locked at time LT_s $s.t$ $\sum_{c \in near\,(s)} Y_{cs} = 1$, where $near(s)$ denotes the request nodes that are in neighborhood of s. These services are penalized till they satisfy the requests of their neighboring request nodes passed their PWT i.e during time period $(LT_s - rt_s)$.
2. For the services that are penalized, penalty is calculated for the elapsed time between arrival of request and delivery of service when request is fully satisfied i.e. during time period $(t - rt_s)$.
3. There are few services which remain unallocated till the threshold reached. For these services, penalty is calculated from the request arrival time till *threshold* i.e. during time period $(rt_s + threshold)$.

Lemma 4.3: For each indirectly connected request node c, we have $d_{cs} \leq 3.(1 + \varepsilon)\alpha_c$.

Proof. Because c is indirectly connected, there must exist a service provider $s^{'}$ and a request node $c^{'}$ such that $c^{'}$ has contributed to both s and $s^{'}$ and $(1+\varepsilon)\alpha_c \geq d_{c^{'}s}$.We also know that both $d_{c^{'}s^{'}}$ and $d_{c^{'}s}$ are upper bounded by $(1+\varepsilon)\alpha_c$. Now because $s^{'}$ is allocated to both c and $c^{'}$, therefore $d_{c^{'}s^{'}} \leq (1+\varepsilon)\alpha_{c^{'}}$ and $d_{cs^{'}} \leq (1+\varepsilon)\alpha_{c^{'}}$ Moreover, $s^{'}$ must be declared open before $c^{'}$ was declared *locked*.

As $(1+\varepsilon)\alpha_{c^{'}} > d_{c^{'}s^{'}}$, $c^{'}$ must be *locked* prior to c was *locked*. Using these facts and triangle inequality, we obtain:

$$d_{cs} \leq d_{c^{'}s} + d_{c^{'}s^{'}} + d_{cs^{'}}$$

$$\leq (1+\varepsilon)\alpha_c + 2(1+\varepsilon)\alpha_{c^{'}}$$

$$\leq 3(1+\varepsilon)\alpha_c$$

Now because $\{\alpha_c, \lambda_{cs}\}$ are dual feasible solution, it is upper bounded by primal solution. Therefore, $\sum_c \alpha_c \leq opt$. Combining this with lemma 3 and 4, it is known that the solution using primal dual algorithm becomes:

$$\sum_{s \in \pi} o_s + \sum_{c \in C} d_{cs} + \sum_s h_s \leq 3(1+\varepsilon^{'})opt \qquad (4.24)$$

for some $\varepsilon > 0$.

Lemma 4.4: Proposed algorithm runs in polynomial time.

Proof.

1. The number of iterations for suggested algorithm is polynomial. This is based on the fact that service opening cost and ratio of minimum node-server distance to maximum request node-server distance is polynomially bounded. Therefore, a polylogrithmic upper bound on the number of iterations is established i.e. $\log_{1+\varepsilon} m^k = O(\log_{1+\varepsilon} m),\ k \geq 1$.

2. We used a greedy heuristic to allocate a server to new request and to minimize the connection cost. It takes polynomial time to assign priorities to all incoming requests generating at the same time. It is a polynomial function of the request arrival rate.
3. The number of iterations in our algorithm are upper bounded by $O\ (\log_{1+\varepsilon} m)$ with a total of $O\ (\log_{1+\varepsilon} m)$ basic operations(as discussed earlier) over a matrix of size m. Hence, The proposed algorithm runs in $O\ (\log_{1+\varepsilon} m)$ matrix operations.

Putting all together we now present our final theorem.

Theorem 4.1: *for* $\varepsilon > 0$, *there is a primal-dual approximation algorithm giving a* $(3 + \varepsilon)$ *approximation factor for critical service time requirement problem in food courts.*

4.2 Concluding Remarks

In this work we focus on minimizing the cost for service allocation problems with critical time constraints. Our work focuses mainly on cost optimization in a way that the service center should not be underutilized. As we have come across the situations where food service centers are not properly utilized or bear a loss due to not providing in-time service to the customers. To handle such cases, we have categorized the service center as – temporarily open and permanently open. The temporarily open service centers may eventually be closed if they are underutilized and suffer a loss for a long period of time. To relate our problem more closely to real life problems such as food services, medical services etc., we consider the priority of request and other operational constraints. We presented a $(3 + \varepsilon)$ - approximation using a hybrid primal-dual approximation algorithm. We here consider the services without any capacity constraint.

Chapter 5

5 Approximation Algorithm for UFLP with Parallel Supplies

Many large-scale industries aim to satisfy customer demand with minimum effort and within stipulated time. The customer demand is to be satisfied while meeting all desired constraints such as minimizing time, maximizing profit etc. We present in this chapter a polynomial time constant factor approximation algorithm for scheduling customer orders over a specified path. An ILP formulation is presented for concerned problem. A greedy heuristic based primal-dual approximation algorithm is proposed to obtain the solution to the problem which is also proven to be asymptotically optimal.

In this chapter, we consider UFLP where each facility *k* different service points from where it starts supplying customer demands at the same time. This problem generalizes the FLP except that it requires that each request node is assigned to *k* facilities in a sequence, each of which belongs to a distinct class or type. In other words, each request node receives its demand from k different service nodes. As with the FLPs discussed in previous chapters, the solution to the stated problem must balance the cost of routing the request nodes through their assigned sequence of facilities, and their costs. The FLP with parallel deliveries may be capacitated or uncapacitated, metric or non-metric, just as FLP may be. We again concentrate on the UFLP with parallel supplies.

This work is inspired by access network design problem, which was initially proposed by Andrews and Zhang [173]. In network design problems, set of request nodes are connected to a central exchange, which installs communications links to carry request nodes' traffic. In such network, each communications link may have a fixed cost regardless of number of

request nodes or variable cost based on the number of request nodes that uses it. There exists a trade-off between fixed cost and variable cost that imposes a rigid structure on the network being constructed.

In this problem *k* types of facilities need to be located. Each unit of demand needs to be shipped from a service center through service stations and finally to the demand points. Authors in [174] has extended their previous work to accommodate multiple levels with a performance guarantee of 3.16. This was further improved to 3 by [44]. Here we also present a simple greedy method for MFLP using work in [157]. The proposed work obtains solution within 6 times of the optimum.

For UFLP with parallel deliveries from k service stations simultaneously, we assume that for set I of facilities, the service is provided from k different service stations at the same time. Let I be partitioned into k classes denoted by $I_1, \dots, I_k$. Let J cotains demand nodes, f_i be the facility cost for each $i \in I$ and d_{ij} be the service cost (or the distance traveled) for each $i, j \in I \cup J$. Here, the objective is to assign a sequence of k facilities to each requesting node in a manner such that the overall cost (opening cost and service cost) is minimized. The service cost for a request node is the sum of service cost from each of the facilities in the sequence to that request node. Figure 5 give an illusion of multi level facility location problem with k=3.

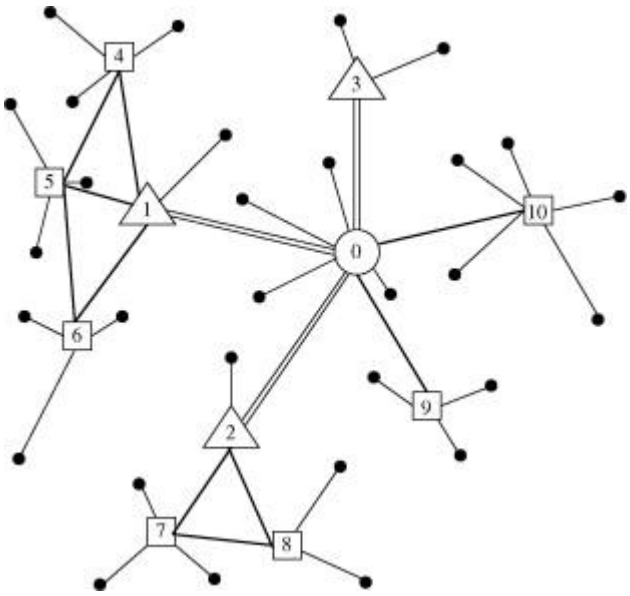


Figure 5.1 Multi level Facility Location Problem with k=3.

Facility 0 is class-1, facilities 1,2,3 are class-2 and facilities 4-10 are class-3 facilities respectively. Black solid dots represent demand nodes at different locations.

For convenience, we consider a directed graph G with nodes V = $I \cup J$. The edges E of G are represented as $i_l i_{l-1}$ for $i_l \in I_l, i_{l-1} \in I_{l-1}, i = 2, \dots, k \; or \; j i_k \; for \; j \in J, i_k \in I_k$. For each $ij \in E$, the service cost of that edge will be d. Suppose there exists a set S of facilities containing at least one facility of each type (class). Each request node j is assigned shortest path with reference to the service cost, from j to one of the facilities of $I_1 \cap S$, where I_1 represents the class 1 (we call them top level) facilities. We define few more variables here.

$P = I_k, \dots I_1$ valid path of length k-1.

$d_{pj} = d_{j\,i_k} + d_{i_k i_{k-1}} + \cdots . + d_{i_2 i_1}$, *the service cost of* j *to path* $p = (i_k, \dots, i_1) \in P$

X_i = 1 if facility i is allocated (or open), 0 otherwise.

Y_{pj} = 1 if request node j is assigned to path p, 0 otherwise.

The ILP formulation of objective function is now given as:

$$ILP - 1 \quad min \sum_{i \in I} f_i X_i + \sum_{p \in P} \sum_{j \in J} d_{pj} Y_{pj} \qquad (5.1)$$

Subject to following constraints:

$\sum_{p \in P} Y_{jp} = 1, \ \forall j \in J$ (5.2)

$\sum_{p \ni i} Y_{jp} \leq X_i, \ \forall i \in I, j \in J$ (5.3)

$Y_{pj} \in \{0,1\}, X_i \in \{0,1\}$

Constraints (5.2) ensure that each *j* is connected through some path and (5.3) ensures that only open facilities are used in the path.

LP −relaxation of (ILP-1) is achieved by relaxing the integrality constraints i.e.:

$$Y_{jp} \geq 0, \ X_i \geq 0$$

5.1 Proposed Algorithm

The proposed algorithm has two phases. Dual formulation of the primal program in ILP-1 is proposed in phase 1. At the end of phase 1, we get a subset of facilities that are fully paid (and hence open) and a subset of connected demand nodes to the open facilities. We also introduce some more terminology during this phase such as neighboring demand nodes, predecessor facilities and the central path. Then in phase 2, by using the lemmas and results obtained in phase 1 we provide the solution to primal program which is proved to be within constant factor of the optimal solution.

Phase 1:

We introduce two dual variables α_j and β_{ij}. The dual of (5.1) is given below:

$$minimize \sum_{j \in J} \alpha_j$$

Subject to following constraints:

$$\alpha_j - \sum_{i \in P} \beta_{ij} \, d(jp), \qquad \forall p \in P, j \in J \qquad (5.4)$$

$$\sum_{j \in J} \beta_{ij} \leq f_i, \qquad \forall i \in I \qquad (5.5)$$

$$\beta_{ij} \geq 0, \qquad \forall i \in I, j \in J \qquad (5.6)$$

The dual variable α_j represents how much $j \in J$ chooses to pay as connection cost. Similarly, β_{ij} indicates how much $j \in J$ is ready to pay towards the opening cost f_i. Therefore a facility *i* is said to be fully paid if:

$$\sum_{j \in J} \beta_{ij} = f_i \qquad (5.7)$$

A request node $j \in J$ reaches $i_l \in V_l$ if there is a path p from V_1 to i_l and each facility $i_k \in p \mid k$ is fully paid and:

$$\alpha_j = d_{jp} + \sum_{i \in p} \beta_{ij} \qquad (5.8)$$

Our algorithm starts with constructing a dual solution as in chapter 3 where we applied the same procedure for solving an instance of FLP with penalties. As previously discussed our work is inspired by procedure suggested by Jain & Vazirani [157].

Algorithm 5.1: Phase 1 of Primal-dual Approximation for MFLP

1. We start by initializing both the dual variables as $\alpha = \beta = 0$. α_j is incremented uniformly (at unit rate) until j gets connected. β_{ij} is also increased at unit rate as soon as $j \in J$ ris locked by a non-fully paid node $i \in I$ until f_i is fully paid and j leaves *i*.
2. Until all $j \in J$ becomes *frozen* do,

a) Increase α_j, $\forall j \in J$ not yet frozen

b) Increase β_{ij}, $\forall i \in I, j \in J$ satisfying (i) – (iii):

i. *j* is locked by *i*

ii. *j* is not yet assigned a facility

iii. *i* is not yet fully paid.

The algorithm ensures that at any time *t*, if α_j is increased then $\alpha_j = t$. Let (α, β) is the final dual solution. Before developing a primal solution (X,Y), some facts about the dual variables are given:

For each fully paid facility $i \in V_l, l \geq 2$, let t_i is the time at which facility i becomes fully paid. Furthermore, *predecessor* of i is the facility of type $l-1$ through which i reached demand node for the first time.

The *predecessor* of a fully paid facility $i \in V_1$ is its closest demand point and $t_{Pred\,(i)} = 0$. The set of all demand nodes contributing towards p_i are known as the *adjacent* demand points of *i* and represented by Adj_i.

Now $\forall\, j \in J$ which is connected, let $\overline{p_j} \in P$ represents the corresponding connecting path of fully paid facilities. Clearly, $d\left(j\overline{p_j}\right) \leq \alpha_j$, for all $j \in J$. Moreover, according the algorithm (Fig. 5.2) for all fully paid facilities $i \in \overline{p_j}$, either $\alpha_j = t_i \; and \; \beta_{ij} > 0 \; or \; \alpha_j > t_i \; and \; \beta_{ij} = 0$. The condition holds true for all fully paid facilities $i \in V_k$. It also remains true for corresponding paths $p_i = (i_1, \dots, i_k)$ that, $t_{i_1} \leq \cdots \leq t_{i_k}$. Based on these observations we present following lemmas.

Lemma 5.1 If $i, i^{'} \in V_k$ are two fully paid facilities having common neighbors, then $\forall\, j^{'} \in J \; and \; i^{'} \in \overline{p_{j'}}$,

$$d_{j_i j'} \leq 4.max\{t_i, \alpha_{j'}\}$$

Proof:

Let $j \in Adj_i \cap Adj_{i'}$. Since $j \in Adj_i$, there exists $a\; i_l \in p_i | \, \beta_{i_l j} > 0$. Then a path q from V_1 to i_l is available s.t $d(jq) \leq t_i$. Suppose $p_{i'} = (i_1^{'}, \dots, i_k^{'})$. Similarly, there a path q' from V_1 to $i_r^{'}$ *s.t.* $d(jq^{'}) + \sum_{s=r}^{k-1} d_{i'_s i'_{s+1}} \leq t_{i'}$. Now using triangle inequality,

$$d_{j_i j'} \leq d(j_i p_i) + d(jq) + d(jq^{'}) + \sum_{s=r}^{k-1} d_{i'_s i'_{s+1}} + d(j'\overline{p_{j'}})$$

$$\leq 2.t_i + t_{i'} + \alpha_{j'}$$

$$\leq 2.t_i + 2.\alpha_{j'}$$

$$\leq 4.max\{t_i, \alpha_{j'}\}.$$

Phase 2:

The primal solution (X,Y) is constructed in Phase 2:

Algorithm 5.2: Phase 2 of Primal-dual Algorithm for MFLP

1. Assume that there are r facilities that are fully paid in the last category (type). Order them according to non decreasing order of the time at which they become fully paid.
2. Pick greedily the facilities from above list and create a set of cluster centers of indirectly connected edges of these facilities and assign each $j \in J$ to some cluster center $i_0 \in C$ as follows:
   a. *Initialize* $C = \phi$
   b. *For* $i = 1, \ldots, r$ *do*

      If $Adj_i \cap Adj_{i_0} \neq \phi$ *for some* $i_0 \leq i$,

      allocateto p_{i_0} *all request nodes* $j \in J$ *where* i_0 *is predecessor of* p_{i_0}.

      Else

      $C = C \cup \{i\}$ *and allocate to* p_i *all request nodes* $j \in J$ for which i_0 *is predecessor of* p_{i_0}.

Note that each demand node j is assigned to one cluster center and thus it contributes to at most one such path where $i \in C$. Let us call these paths as the central paths. Now the primal solution (X,Y) is obtained by connecting request nodes along corresponding *central* paths

Theorem 5.1 *The primal solution (X,Y) satisfies:*

$$f(X) + d(Y) \leq 6. \sum_{j \in J} \alpha_j$$

Proof:

Now using observations we get $t_{i_0} \leq \alpha_j$ and,

$$d_{j\, p_{i_0}} \leq d_{j_{i_0} j} + d_{j_{i_0} p_{i_0}} \leq 4. \alpha_j + t_{i_0} \leq 5. \alpha_j \qquad (5.9)$$

The installation cost of facilities along a central path p_{i_0} can be bounded as :

$$\sum_{i \in p_{i_0}} f_i = \sum_{i \in p_{i_0}} \sum_{j \in Adj_i} \beta_{ij} \leq \sum_{j \in Adj_i} \alpha_j$$

We can further conclude that,

$$f(X) = \sum_{i_0 \in C} \sum_{i \in p_{i_0}} f_i \leq \sum_{j \in J} \alpha_j \qquad (5.10)$$

It follows now from (5.9) and (5.10),

$$f(X) + d(Y) \leq 6. \alpha_j .$$

5.2 Running Time Analysis

We now prove that our algorithm runs in polynomial time. There exists three types of events that can happen and they lead to the following updates:

Event 1: A request node *j* is locked by a facility $i_l \in V_l$. If i_l is not fully paid, it requires one more request node that will contribute towards its opening. The expected time of facility i_l to be fully paid can be calculated in polynomial time. This expected time is updated for every request node *j'* that is locked by i_l when it is locked by other facilities in V_{l+1}.

Event 2: When a request node *j* is locked by a fully paid facility $i_k \in V_k$, then *j* is declared frozen and the dual variable α_j is not raised further. Next for each unpaid facility, the expected running time at which it gets fully paid is recomputed.

Event 3: For each fully paid facility $i_k \in V_k$, all request nodes *j* locked by i_k are frozen. For all such request nodes, the running time is computed as we discussed in event 2 above.

Each edge $(j, i_l) \in J \times V_l,\ 1 \leq l \leq k$ is considered for running time evaluation in following situations:

I. *j* is locked by i_l

II. Another request node is locked by a non fully paid facility $i_{l-1} \in V_{l-1}$.

III. *j* becomes frozen.

In all three situations *O (log n)* operations are performed. Hence, the total number of operations performed are $O(n^4 logn)$ which is polynomial in the input size.

5.3 Proposed Approximation Algorithm for UMFLP with Penalty

In this version of MFLP, just like FLPWP, each demand node $j \in J$ is assigned a penalty h_j if its request is rejected. Each request node will either receive the service and pay the service cost or will be rejected and we pay the penalty. In this formulation only limited number of facilities are opened as it avoids opening the facilities that are at distant locations and will serve only a few number of clients. We here illustrate how the primal-dual algorithm presented above is reconstructed in order to obtain a 6-approximation for this version of MFLP as well.

In ILP formulation of this problem, we use the same variables described above and introduce a new indicator variable Z_j indicating whether a demand node is served or not. We set $Z_j = 1$ if the demand node *j* is selected for service and $Z_j = 0$ if demand node *j* is rejected or not served. The total penalty for a solution (X,Y,Z) is denoted by *H*, i.e.,

$$H = \sum_{j \in J} h_j Z_j \qquad (5.11)$$

The following formulation describes the MFLP with penalties:

$$minimize \sum_i \sum_j c_{ij} Y_{ij} + \sum_i f_i X_i + \sum_j h_j Z_j \qquad (5.12)$$

Subject to following constraints:

$$\sum_{p \in P} Y_{jp} + h_j \geq 1 , \qquad \forall j \qquad (5.13)$$

$$\sum_{p:p \ni i} Y_{jp} \leq X_i , \qquad \forall i, j \qquad (5.14)$$

$$Y_{jp} , X_i , h_j \in \{0,1\}, \qquad \forall i, j, p \qquad (5.15)$$

Now we relax above LP by relaxing constraint (5.15) as $Y_{jp} , X_i , h_j \geq 0$. We now define the dual of relaxed LP as follows:

$$maximize \sum_j \alpha_j \qquad (5.16)$$

Subject to following constraints:

$$\alpha_j - \sum_{i \in p} \beta_{ij} \leq c(jp), \qquad \forall j, p \qquad (5.17)$$

$$\sum_j \beta_{ij} \leq f_i , \qquad \forall i \qquad (5.18)$$

$$\alpha_j \leq h_j \qquad (5.19)$$

$$\alpha_j , \beta_{ij} \geq 0 , \qquad \forall i, j \qquad (5.20)$$

The only difference between MFLP and MFLPWP is the constraint (5.19). Therefore, a dual feasible solution can be constructed in the similar way without any penalty added. Just like previous cases, the dual variables stop increasing whenever one of the following events occur:

$\alpha_j = h_j$ or j gets connected to a permanent open facility. The primal solution is now constructed as we did in MFLP above, with one exception that the frozen request nodes are not assigned any arbitrary path. They may contribute towards opening facilities along one *central* path.

Using analysis as discussed in section 5.1, we can prove the following theorem:

Theorem 5.2 Applying primal-dual algorithm to case when penalties are added for demand rejection, will also yield a 6-approximation of the optimal solution for MFLP.

5.4 Conclusion

We have presented approximation algorithm for the metric UFLP with parallel supplies In this chapter. Our algorithm achieves a 6-approximation guarantee for the MFLP. The work can be further extended to obtain a performance guarantee better than 6. We also prove the optimality of our solution in terms of running time. We further proved that the same approximation guarantee can be obtained for penalty version of the problem. Work can also be further extended for the capacitated version of the problem. There are no approximation algorithms for the capacitated version of MFLP. We are also working on extending our research to MFLP with time dependant penalties.

Chapter 6

6 Summary and Future Directions

The main focus of this thesis was to design and analyze better approximation algorithms for the UFLP and its variants. We developed several algorithms for some of the variants of the UFLP, all of which achieves best known approximation factor in polynomial time. The main tools we used in designing and analyzing our algorithms are primal-dual and filtering and rounding based methods. We showed the versatility of these techniques by applying them to real life scenario (such as logistics and express delivery services). Firstly, we considered FLP for time based services and proved tight lower bounds obtained in polynomial time. Next we extended our work to FLP with multiple requests under different scenario.

As discussed in chapter 1, Linear Programming is a very powerful tool in designing approximation algorithms. Its two main approaches: LP rounding and the primal-dual are used for designing efficient approximation algorithms thesis. In both cases, first an IP formulation of the problem is to be prepared and then an LP relaxation of it is presented. For a minimization (maximization) problem, the solution to the LPP gives a lower bound (upper bound) on the optimal value. In LP rounding, the optimal solution of the relaxed LP is rounded to an integer value, such that the rounded solution is feasible and has an objective value of at most ρ times the optimal solution value of the relaxed LPP. Its main drawback is that for larger number of constraints and variables, the computational efforts increase rapidly at exponential rate. The primal-dual technique is used to overcome this. It is a very powerful approach and gives the solution in polynomial time. For minimization problems, the dual of the linear program is a maximization problem. By weak duality theory, any feasible solution to the dual objective function has a value smaller than the objective value of the primal feasible solution. Hence, the objective value of dual feasible solution works as a lower bound

of the optimal value of the original problem. For a simple LPP, its dual feasible solution can be constructed by using a simple combinatorial algorithm. After which an integral solution of the primal problem can be created which is bounded in terms of the dual solution.

We give some basic definitions and preliminaries in Chapter 1 and also discuss different strategies for designing approximation algorithms. First we discuss greedy strategy and local search procedure followed by linear programming based approaches such as LP rounding, primal-dual and filtering. These techniques have been explained very clearly by giving suitable examples. For clear understanding, we also present the integer and linear programming formulation of general case of UFLP.

In Chapter 2 we study the existing approximation algorithms for the metric UFLP, metric CFLP, UFLP with penalties, Fault Tolerant UFLP and UFLP with stochastic demands. We mainly focus on the algorithms by Chudak and Shmoys [92] and Jain and Vazirani [157]. Their algorithms are very helpful to understand how LP rounding and the primal-dual technique can be used for solving UFLP. The procedures proposed by Guha and Khuller [54] and Charikar and Guha [175] are also very useful, which in many cases provides a much improvement in the performance guarantee for many versions of FLPs. Later, the results of Du & Xu [21], [176], [177] were also discussed for fault tolerant and penalty based location models. Their results are the base and motivation for our research and we tried to extend their work in our thesis in chapters 3 and 4. We also reviewed the findings on facility location with varying demands and used their ideas in chapter 5 to formulate a linear program for FLP with multiple requests having different priorities.

Chapter 3 is mainly dedicated to the FLP with time dependent penalties, the FLPTP. FLPTP is an extension over FLP with penalty (FLPWP). The FLPWP is similar to the metric version of the UFLP, the only difference is that each unallocated request node adds to some penalty. The approximation algorithm for this problem, proposed by Charikar et al. [124] which make

use of LP rounding has an approximation ratio 3. By using and primal-dual and filtering, Jain et al. [58] proved that the performance guarantee can be reduced to 2. Later Du, Lu and Xu considered in their paper facility location problems with sub-modular penalties ($FLPSP$) and gave a $3-$approximation primal-dual algorithm using the features of submodular penalty functions. We however extend their work to facility location problem with time dependent penalty (FLPTP), where penalty is a monotonically increasing function of time. We in our work propose a simpler $4-$ approximation algorithm for the FLPTP, based on LP rounding and primal-dual approximation as well. We presented the more realistic version of the problem as in many real life situations penalty due to delayed service increases with passage of time.

In Chapter 4 we further extended our work in previous chapter and focused on uncapacitated priority facility location problems with critical service time requirements. The aim is finding a set facilities to open and allocating request nodes to paths along these facilities such that the total cost is minimized. We showed that how travel time and hence the route chosen to deliver the service plays a major role in the overall cost. A carefully chosen path reduces the service delivery time and hence reduces the penalty imposed due to delayed service. We also considered a priority function to make a decision on which request node to serve first. Unlike previous case, priority is not only a function of waiting time but also the quantum of demand requested by the request node. The problem therefore is closely related to the conventional vehicle routing or location routing problem with some additional constraints. Numerous algorithms exist for general case of VRP. Bazgan et. al. [178] presented a $(\frac{197}{99}+\varepsilon)$-approximation algorithm for kVRP. Viswanath & Ravi gave a $O(logD)$-approximation for distance constrained VRP based on LP relaxation and tree properties. We however considered location routing with time constraints and proposed $(3+\varepsilon)$-approximation algorithm which is based on LP rounding, primal-dual and a greedy heuristic. We show how

an extension of algorithm in previous chapter for the FLPTP, combined with greedy improvement, gives a performance guarantee of $(3 + \varepsilon)$ where ε is arbitrarily small positive constant.

In Chapter 5 we focus on the MFLP with parallel demands (or requests). The requests are assumed to be generated at random time periods. Work done in this chapter is closely related to conventional packet routing problem under different constraints. Facilities are categorized into different classes that are considered to handle customer requests at the same time. Each demand must be shipped from a service center through different service stations. There might be more than one supplies in a single shipping. The transportation costs between demand points and facilities are known and they satisfy the triangle inequality. Our aim is now to minimize the sum of overall shipping cost (or distance) and the facility installation cost. We present LP formulations for stated problem and claim that our solutions is asymptotically optimal i.e. produces a solution which is almost optimal as the number of requests approach to infinity. For each service station we also assume that the demand supply will follow a predetermined path. We used greedy path finding algorithm to route and schedule the deliveries. We use LP rounding framework and propose a primal-dual based 6-approximation algorithm for the given problem.

Future Scope

1. There is a close gap between lower bound (1.46 by Guha and Khuller) and upper bound (1.52 by chudak and shmoys) for the approximability of UFLP. The question is whether this is a tight bound or can be further improved by a combination of LP rounding and greedy techniques?
2. Find better lower bound estimates for Capacitated Facility Location Problems via linear programs. Is there a polynomial time-computable relaxation R of the CFLP? Or, what's the approximate min-max relaxation?

3. Similarly, although the 4-approximation bound is tight for the Time Dependent Penalty, it is not known whether a bound better than this can be obtained for FLPTP for metric facility location algorithms. Our algorithm used a waiting time based priority function for calculating penalty cost. Can we improve the results by considering arbitrary penalties where irrespective of the waiting time penalties are added?
4. For logistics systems with critical service time requirements we considered the services without having any upper limit on the number of request nodes it can serve. An extension of our setting is the withdrawal of requests by request nodes before completion.
5. Another area of research can be to obtain an efficient solution for multi facility location problem with penalties and multi facility location problem with time dependant penalties.

References

[1] M. S. Daskin, “Network and discrete location: models, algorithms, and applications,” no. 63968, pp. 1–4, 1995.

[2] A. A. Kuehn and M. J. Hamburger, “A Heuristic Program for Locating Warehouses,” *Manage. Sci.*, vol. 9, no. 4, pp. 643–666, Jul. 1963.

[3] O. Kariv and S. L. Hakimi, “An Algorithmic Approach to Network Location Problems. II: The *p* -Medians,” *SIAM J. Appl. Math.*, vol. 37, no. 3, pp. 539–560, Dec. 1979.

[4] D. B. Shmoys, “Approximation Algorithms for Facility Location Problems,” *Approx. Algorithms Comb. Optim.*, vol. 1913, no. 20244, pp. 27–32, 2000.

[5] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, “Analysis of a Local Search Heuristic for Facility Location Problems,” *J. Algorithms*, vol. 37, no. 1, pp. 146–188, 2000.

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

[7] M. S. Daskin, “What You Should Know About Location Modeling,” no. January, 2008.

[8] K. T. Tse and J. Song, “An analytical model of linked twin tall buildings and modal property analysis,” in *Proceedings of the Sixth European-African Conference on Wind Engineering*, 2013.

[9] J. Sathishkumar and D. R. Patel, “Enhanced location privacy algorithm for wireless sensor network in internet of things,” in *Internet of Things and Applications (IOTA), International Conference on*, 2016, pp. 208–212.

[10] S. L. Hakimi, “Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems,” *Oper. Res.*, vol. 13, no. 3, pp. 462–475, Jun. 1965.

[11] C. Toregas, R. Swain, C. ReVelle, and L. Bergman, “The Location of Emergency Service Facilities,” *Oper. Res.*, vol. 19, no. 6, pp. 1363–1373, 1971.

[12] H. A. Eiselt and V. Marianov, *Foundations of Location Analysis*. Springer US, 2013.

[13] T. J. Van Roy and D. Erlenkotter, “A dual-based procedure for dynamic facility location,” *Manage. Sci.*, vol. 28, no. 10, pp. 1091–1105, 1982.

[14] D. J. Sweeney and R. L. Tatham, “An Improved Long-Run Model for Multiple Warehouse Location,” *Manage. Sci.*, vol. 22, no. 7, pp. 748–758, Mar. 1976.

[15] G. O. Wesolowsky and W. G. Truscott, “The Multiperiod Location-Allocation Problem with Relocation of Facilities,” *Manage. Sci.*, vol. 22, no. 1, pp. 57–65, Sep. 1975.

[16] M. S. Daskin, “Extensions of Location Models,” in *Network and Discrete Location*, John Wiley & Sons, Inc., 1995, pp. 309–382.

[17] M. Objectives, “Discrete Location Problems – Theory , Algorithms , and Extensions to,” 2016.

[18] M. Mahdian and M. Pál, “Universal facility location,” in *European Symposium on Algorithms*, 2003, pp. 409–421.

[19] V. Verter, “Foundations of Location Analysis,” in *Foundations of Location Analysis*, vol. 155, 2011, pp. 25–38.

[20] C. Swamy and D. B. Shmoys, “Fault-tolerant facility location,” vol. 4, no. 4, pp. 735–736, 2003.

[21] D. Du, R. Lu, and D. Xu, “A primal-dual approximation algorithm for the facility location problem with submodular penalties,” *Algorithmica*, vol. 63, no. 1–2, pp. 191–200, 2012.

[22] S. Barat, A. Pal, and T. De, “A load balanced approach of multicast routing and wavelength assignment in WDM networks,” *Int. J. Commun. Networks Distrib. Syst.*, vol. 15, no. 1, pp. 1–21, 2015.

[23] F. V Louveaux, “Discrete stochastic location models,” *Ann. Oper. Res.*, vol. 6, no. 2, pp. 21–34, 1986.

[24] G. Li, Z. Wang, and C. Wu, “Approximation algorithms for the stochastic priority facility location problem,” *Optimization*, vol. 62, no. 7, pp. 919–928, Jul. 2013.

[25] S. Ram, K. R. Ramakrishnan, P. K. Atrey, V. K. Singh, and M. S. Kankanhalli, “A design methodology for selection and placement of sensors in multimedia surveillance systems,” in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, 2006, pp. 121–130.

[26] M. Saini, P. K. Atrey, S. Mehrotra, and M. Kankanhalli, “W 3-privacy: understanding what, when, and where inference channels in multi-camera surveillance video,” *Multimed. Tools Appl.*, vol. 68, no. 1, pp. 135–158, 2014.

[27] V. V. Vazirani, “Approximation Algorithms,” pp. 1–380, 2003.

[28] C. P. Gomes and R. Williams, “Charpter 18 - APPROXIMATION ALGORITHMS,” *Forbes*, 2005.

[29] D. P. Williamson and D. B. Shmoys, “The Design of Approximation Algorithms,” p. 504, 2011.

[30] G. Ausiello, A. Marchetti-Spaccamela, P. Crescenzi, G. Gambosi, M. Protasi, and V. Kann, “Design Techniques for Approximation Algorithms,” in *Complexity and Approximation*, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 39–85.

[31] D. S. Hochbaum, “Heuristics for the fixed cost median problem,” *Math. Program.*, vol. 22, no. 1, pp. 148–162, 1982.

[32] V. Pandit, “Local Search Heuristics for Facility Location Problems,” *PhD Thesis*,

Indian Inst. Technol. Delhi, no. July, 2004.

[33] H. P. Williams, “Integer programming,” *Constraints*, vol. 11 Suppl 1, no. 1, pp. 272–319, 1980.

[34] L. Trevisan, “Combinatorial Optimization : Exact and Approximate Algorithms,” 2011.

[35] F. Malucelli, “Integer Linear Programming,” pp. 1–50.

[36] A. Schrijver, *Theory of linear and integer programming*. Wiley, 1998.

[37] L. Lovász, “On the ratio of optimal integral and fractional covers,” *Discrete Math.*, vol. 13, no. 4, pp. 383–390, 1975.

[38] G. Cornuejols, M. L. Fisher, and G. L. Nemhauser, “Exceptional Paper—Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms,” *Manage. Sci.*, vol. 23, no. 8, pp. 789–810, Apr. 1977.

[39] L. A. Wolsey, “Heuristic analysis, linear programming and branch and bound,” Springer Berlin Heidelberg, 1980, pp. 121–134.

[40] A. Agrawal, P. Klein, and R. Ravi, “When trees collide: An approximation algorithm for the generalized Steiner Problem on Networks,” *SIAM J. Comput.*, 1994.

[41] M. X. Goemans, D. P. Williamson, and A. W. Tucker, “THE PRIMAL-DUAL METHOD FOR APPROXIMATION ALGORITHMS AND ITS APPLICATION TO NETWORK DESIGN PROBLEMS.”

[42] M. L. BALINSKI, “ON FINDING INTEGER SOLUTIONS TO LINEAR PROGRAMS,.” 1964.

[43] H. W. (Harold W. Kuhn, A. W. (Albert W. Tucker, and G. B. (George B. Dantzig, *Linear inequalities and related systems*. .

[44] K. Aardal, F. A. Chudak, and D. B. Shmoys, “A 3-approximation algorithm for the -level uncapacitated facility location problem,” *Inf. Process. Lett.*, vol. 72, no. 5–6, pp. 161–167, Dec. 1999.

[45] B. Korte, J. Vygen, B. Korte, and J. Vygen, *Combinatorial optimization*, vol. 2. Springer, 2012.

[46] J. F. Stollsteimer, “A working model for plant numbers and locations,” *J. Farm Econ.*, vol. 45, no. 3, pp. 631–645, 1963.

[47] M. L. Balinski and P. Wolfe, “On Benders decomposition and a plant location problem,” *ARO-27, Math. NJ, 1963)*, p. 45, 1963.

[48] A. A. Kuehn and M. J. Hamburger, “A heuristic program for locating warehouses,” *Manage. Sci.*, vol. 9, no. 4, pp. 643–666, 1963.

[49] A. S. Manne, “Plant location under economies-of-scale—decentralization and computation,” *Manage. Sci.*, vol. 11, no. 2, pp. 213–235, 1964.

[50] M. L. Balinski, “Integer programming: methods, uses, computations,” *Manage. Sci.*,

vol. 12, no. 3, pp. 253–313, 1965.

[51] D. B. Shmoys, É. Tardos, and K. Aardal, “Approximation algorithms for facility location problems,” in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, 1997, pp. 265–274.

[52] J. Vygen, “Approximation Algorithms for Facility Location Problems,” pp. 1–59, 2005.

[53] K. Jain and V. V Vazirani, “Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation,” *J. ACM*, vol. 48, no. 2, pp. 274–296, 2001.

[54] S. Guha and S. Khuller, “Greedy strikes back: Improved facility location algorithms,” *J. algorithms*, vol. 31, no. 1, pp. 228–248, 1999.

[55] M. Charikar and S. Guha, “Improved combinatorial algorithms for the facility location and k-median problems,” pp. 378–388, 1999.

[56] D. B. Shmoys, E. Tardos, and K. Aardal, “Approximation algorithms for facility location problems,” *Proc. 3rd Int. Work. Approx. Algorithms Comb. Optim.*, vol. 5, pp. 265–274, 1997.

[57] F. A. Chudak, “Improved approximation algorithms for uncapacitated facility location,” in *International Conference on Integer Programming and Combinatorial Optimization*, 1998, pp. 180–194.

[58] K. Jain and V. V Vazirani, “Approximation algorithms for metric facility location and k-Median problems using the primal-dual schema and Lagrangian relaxation,” *J. ACM*, vol. 48, no. 2, pp. 274–296, 2001.

[59] R. R. Mettu and C. G. Plaxton, “The online median problem,” *Proc. 41st Annu. Symp. Found. Comput. Sci.*, pp. 339–348, 2000.

[60] M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani, “A greedy facility location algorithm analyzed using dual fitting,” in *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, Springer, 2001, pp. 127–137.

[61] K. Jain, M. Mahdian, and A. Saberi, “A new greedy approach for facility location problems,” in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 2002, pp. 731–740.

[62] M. Mahdian, Y. Ye, and J. Zhang, “Improved approximation algorithms for metric facility location problems,” in *International Workshop on Approximation Algorithms for Combinatorial Optimization*, 2002, pp. 229–242.

[63] K. Jain, M. Mahdian, and A. Saberi, “A new greedy approach for facility location problems,” *STOC 02 Proc. thiryfourth Annu. ACM Symp. Theory Comput.*, pp. 731–740, 2002.

[64] M. Mahdian, Y. Ye, and J. Zhang, “A 1.52-Approximation Algorithm for the Uncapacitated Facility Location Problem,” pp. 127–137, 2002.

[65] S. Arora, P. Raghavan, and S. Rao, “Approximation schemes for Euclidean k -medians and related problems,” *Proc. thirtieth Annu. ACM Symp. Theory Comput. - STOC ’98*, pp. 106–113, 1998.

[66] S. G. Kolliopoulos and S. Rao, “A nearly linear-time approximation scheme for the Euclidean k-median problem,” in *European Symposium on Algorithms*, 1999, pp. 378–389.

[67] A. Kolen, “Solving covering problems and the uncapacitated plant location problem on trees,” *Eur. J. Oper. Res.*, vol. 12, no. 3, pp. 266–278, 1983.

[68] I. Barany, L. A. Wolsey, and J. Edmonds, “Packing and covering a tree by subtrees,” *Combinatorica*, vol. 6, no. 3, pp. 221–233, 1986.

[69] G. Cornuéjols, G. L. Nemhauser, and L. A. Wolsey, “The uncapacitated facility location problem,” 1983.

[70] A. A. Ageev and V. L. Beresnev, “Polynomially solvable cases of the simple plant location problem,” in *Proceedings of the 1st Integer Programming and Combinatorial Optimization Conference*, 1990, pp. 1–6.

[71] V. L. Beresnev, “An efficient algorithm for the uncapacitated facility location problem with totally balanced matrix,” *Discret. Appl. Math.*, vol. 114, no. 1, pp. 13–22, 2001.

[72] A. M. Geoffrion and G. W. Graves, “Multicommodity distribution system design by Benders decomposition,” *Manage. Sci.*, vol. 20, no. 5, pp. 822–844, 1974.

[73] U. Akinc and B. M. Khumawala, “An efficient branch and bound algorithm for the capacitated warehouse location problem,” *Manage. Sci.*, vol. 23, no. 6, pp. 585–594, 1977.

[74] R. M. Nauss, “An improved algorithm for the capacitated facility location problem,” *J. Oper. Res. Soc.*, vol. 29, no. 12, pp. 1195–1201, 1978.

[75] T. J. Van Roy, “A cross decomposition algorithm for capacitated facility location,” *Oper. Res.*, vol. 34, no. 1, pp. 145–163, 1986.

[76] J. E. Beasley, “A lagrangian heuristic for set-covering problems,” *Nav. Res. Logist.*, vol. 37, no. 1, pp. 151–164, 1990.

[77] V. Jayaraman and H. Pirkul, “Planning and coordination of production and distribution facilities for multiple commodities,” *Eur. J. Oper. Res.*, vol. 133, no. 2, pp. 394–408, 2001.

[78] M. Pal, T. Tardos, and T. Wexler, “Facility location with nonuniform hard capacities,” in *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, 2001, pp. 329–338.

[79] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, “Analysis of a local search heuristic for facility location problems,” *J. algorithms*, vol. 37, no. 1, pp. 146–188, 2000.

[80] J. Zhang, B. Chen, and Y. Ye, “A multi-exchange local search algorithm for the capacitated facility location problem,” in *International Conference on Integer Programming and Combinatorial Optimization*, 2004, pp. 219–233.

[81] R. Levi, D. B. Shmoys, and C. Swamy, “LP-based approximation algorithms for capacitated facility location,” in *International Conference on Integer Programming and Combinatorial Optimization*, 2004, pp. 206–218.

[82] M. Pal, T. Tardos, and T. Wexler, “Facility location with nonuniform hard capacities,” *Proc. 2001 IEEE Int. Conf. Clust. Comput.*, 2001.

[83] S. Khuller and Y. J. Sussmann, “The capacitated k-center problem,” *SIAM J. Discret. Math.*, vol. 13, no. 3, pp. 403–418, 2000.

[84] P. S. Davis and T. L. Ray, “A branch-bound algorithm for the capacitated facilities location problem,” *Nav. Res. Logist.*, vol. 16, no. 3, pp. 331–344, 1969.

[85] M. Guignard and K. Spielberg, “A direct dual method for the mixed plant location problem with some side constraints,” *Math. Program.*, vol. 17, no. 1, pp. 198–228, 1979.

[86] S. Melkote and M. S. Daskin, “Capacitated facility location/network design problems,” *Eur. J. Oper. Res.*, vol. 129, no. 3, pp. 481–495, 2001.

[87] D. Simchi-Levi and O. Berman, “Minimizing the total flow time of n jobs on a network,” *IIE Trans.*, vol. 23, no. 3, pp. 236–244, 1991.

[88] M. S. Daskin, A. P. Hurter, and M. G. VanBuer, “Toward an integrated model of facility location and transportation network design,” *Northwest. Univ.*, 1993.

[89] F. Chudak and D. P. Williamson, “Improved approximation algorithms for capacitated facility location problems,” *Math. Program.*, vol. 102, no. 2, pp. 207–222, 2005.

[90] M. Mahdian, Y. Ye, and J. Zhang, “Approximation Algorithms for Metric Facility Location Problems,” *SIAM J. Comput.*, vol. 36, no. 2, pp. 411–432, 2006.

[91] M. Mahdian, Y. Ye, and J. Zhang, “A 2-approximation algorithm for the soft-capacitated facility location problem,” in *Approximation, Randomization, and Combinatorial Optimization.. Algorithms and Techniques*, Springer, 2003, pp. 129–140.

[92] F. A. Chudak and D. B. Shmoys, “Improved approximation algorithms for the uncapacitated facility location problem,” *SIAM J. Comput.*, vol. 33, no. 1, pp. 1–25, 2003.

[93] A. F. Bumb, *Approximation algorithms for facility location problems*. 2002.

[94] F. A. Chudak and D. P. Williamson, “Improved approximation algorithms for capacitated facility location problems,” *Integer Program. Comb. Optim.*, vol. 102, no. 2, pp. 99–113, 1999.

[95] A. Aggarwal *et al.*, “A 3-approximation algorithm for the facility location problem with uniform capacities,” *Math. Program.*, vol. 141, no. 1–2, pp. 537–547, 2013.

[96] R. Levi, D. B. Shmoys, and C. Swamy, “LP-based approximation algorithms for capacitated facility location,” *Math. Program.*, vol. 131, no. 1–2, pp. 365–379, 2012.

[97] J. Zhang, B. Chen, and Y. Ye, “A multiexchange local search algorithm for the

capacitated facility location problem," *Math. Oper. Res.*, vol. 30, no. 2, pp. 389–403, 2005.

[98] M. Bansal, N. Garg, and N. Gupta, "A 5-approximation for capacitated facility location," in *European Symposium on Algorithms*, 2012, pp. 133–144.

[99] A. W. Neebe and M. R. Rao, "An algorithm for the fixed-charge assigning users to sources problem," *J. Oper. Res. Soc.*, vol. 34, no. 11, pp. 1107–1113, 1983.

[100] M. J. Cortinhal and M. E. Captivo, "Upper and lower bounds for the single source capacitated location problem," *Eur. J. Oper. Res.*, vol. 151, no. 2, pp. 333–351, 2003.

[101] A. De Maio and C. Roveda, "An all zero-one algorithm for a certain class of transportation problems," *Oper. Res.*, vol. 19, no. 6, pp. 1406–1418, 1971.

[102] J. Barceló and J. Casanovas, "A heuristic Lagrangean algorithm for the capacitated plant location problem," *Eur. J. Oper. Res.*, vol. 15, no. 2, pp. 212–226, 1984.

[103] J. G. Klincewicz and H. Luss, "A Lagrangian relaxation heuristic for capacitated facility location with single-source constraints," *J. Oper. Res. Soc.*, vol. 37, no. 5, pp. 495–500, 1986.

[104] K. Darby-Dowman and H. S. Lewis, "Lagrangian relaxation and the single-source capacitated facility-location problem," *J. Oper. Res. Soc.*, vol. 39, no. 11, pp. 1035–1040, 1988.

[105] C. S. ReVelle, H. A. Eiselt, and M. S. Daskin, "A bibliography for some fundamental problem categories in discrete location science," *Eur. J. Oper. Res.*, vol. 184, no. 3, pp. 817–848, 2008.

[106] V. Arya *et al.*, "Local search heuristic for k-median and facility location problems," *Proc. thirty-third Annu. {ACM} Symp. Theory Comput.*, vol. 33, no. 3, pp. 21–29, 2001.

[107] P. Bose, A. Maheshwari, and P. Morin, "Fast approximations for sums of distances, clustering and the Fermat-Weber problem," *Comput. Geom. Theory Appl.*, vol. 24, no. 3, pp. 135–146, 2003.

[108] S. Rani and G. Sikka, "Recent techniques of clustering of time series data: a survey," *Int. J. Comput. Appl.*, vol. 52, no. 15, 2012.

[109] E. Pitoura and B. Bhargava, "Maintaining consistency of data in mobile distributed environments," in *Distributed Computing Systems, 1995., Proceedings of the 15th International Conference on*, 1995, pp. 404–413.

[110] J.-H. Lin and J. S. Vitter, "e-Approximations with minimum packing constraint violation," in *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, 1992, pp. 771–782.

[111] M. Sviridenko, "An improved approximation algorithm for the metric uncapacitated facility location problem," in *International Conference on Integer Programming and Combinatorial Optimization*, 2002, pp. 240–257.

[112] R. R. Mettu and C. G. Plaxton, "The online median problem," *SIAM J. Comput.*, vol.

32, no. 3, pp. 816–832, 2003.

[113] K. Jain *et al.*, “Greedy Facility Location Algorithms Analyzed Using Dual Fitting with Factor-Revealing LP,” *Lect. Notes Comput. Sci.*, vol. 2129, no. 6, pp. 127–137, 2002.

[114] J. Byrka, A. Srinivasan, and C. Swamy, “Fault-tolerant facility location: A randomized dependent LP-rounding algorithm,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6080 LNCS, no. i, pp. 244–257, 2010.

[115] A. Srinivasan, “Distributions on level-sets with applications to approximation algorithms,” in *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, 2001, pp. 588–597.

[116] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, “Algorithms for Facility Location Problems with Outliers.”

[117] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V Vazirani, “Greedy Facility Location Algorithms Analyzed using Dual Fitting with Factor-Revealing LP,” 2002.

[118] G. Xu and J. Xu, “An LP rounding algorithm for approximating uncapacitated facility location problem with penalties,” *Inf. Process. Lett.*, vol. 94, no. 3, pp. 119–123, 2005.

[119] J. Geunes, R. Levi, H. E. Romeijn, and D. B. Shmoys, “Approximation algorithms for supply chain planning and logistics problems with market choice,” *Math. Program.*, vol. 130, no. 1, pp. 85–106, Nov. 2011.

[120] A. Hayrapetyan, C. Swamy, and É. Tardos, “Network design for information networks,” pp. 933–942, 2005.

[121] F. A. Chudak and K. Nagano, “Efficient solutions to relaxations of combinatorial problems with submodular penalties via the Lov{á}sz extension and non-smooth convex optimization,” in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 79–88.

[122] J. Byrka, S. Li, and B. Rybicki, “Improved approximation algorithm for k-level UFL with penalties, a simplistic view on randomizing the scaling parameter,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8447 LNCS, pp. 85–96, 2014.

[123] M. Asadi, A. Niknafs, and M. Ghodsi, “An Approximation Algorithm for the k-Level Uncapacitated Facility Location Problem with Penalties,” Springer, Berlin, Heidelberg, 2008, pp. 41–49.

[124] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, “Algorithms for Facility Location Problems with Outliers,” *Soda*, pp. 642–651, 2001.

[125] N. Gupta and S. Gupta, “Approximation algorithms for Capacitated Facility Location Problem with Penalties,” pp. 1–16, 2014.

[126] J. Current, M. S. Daskin, and D. Schilling, *Discrete Network Location Models*. 2002.

[127] P. B. Mirchandani and A. R. Odoni, “Locations of medians on stochastic networks,” *Transp. Sci.*, vol. 13, no. 2, pp. 85–97, 1979.

[128] P. B. Mirchandani, “Locational decisions on stochastic networks,” *Geogr. Anal.*, vol. 12, no. 2, pp. 172–183, 1980.

[129] O. Berman and A. R. Odoni, “Locating mobile servers on a network with Markovian properties,” *Networks*, vol. 12, no. 1, pp. 73–86, 1982.

[130] J. R. Weaver and R. L. Church, “Computational procedures for location problems on stochastic networks,” *Transp. Sci.*, vol. 17, no. 2, pp. 168–180, 1983.

[131] O. Berman and B. LeBlanc, “Location-relocation of mobile facilities on a stochastic network,” *Transp. Sci.*, vol. 18, no. 4, pp. 315–330, 1984.

[132] M. S. Daskin, “Application of an expected covering model to emergency medical service system design,” *Decis. Sci.*, vol. 13, no. 3, pp. 416–439, 1982.

[133] C. ReVelle and K. Hogan, “The maximum availability location problem,” *Transp. Sci.*, vol. 23, no. 3, pp. 192–200, 1989.

[134] V. Marianov and C. ReVelle, “The queueing maximal availability location problem: a model for the siting of emergency vehicles,” *Eur. J. Oper. Res.*, vol. 93, no. 1, pp. 110–120, 1996.

[135] A. S. Manne, “Capacity expansion and probabilistic growth,” *Econom. J. Econom. Soc.*, pp. 632–649, 1961.

[136] O. Berman, R. C. Larson, and S. S. Chiu, “Optimal Server Location on a Network Operating as an *M* / *G* /1 Queue,” *Oper. Res.*, vol. 33, no. 4, pp. 746–771, Aug. 1985.

[137] R. Carbone, “Public Facilities Location Under Stochastic Demand,” *INFOR Inf. Syst. Oper. Res.*, vol. 12, no. 3, pp. 261–270, Oct. 1974.

[138] J. C. Bean, J. L. Higle, and R. L. Smith, “Capacity Expansion Under Stochastic Demands,” *Oper. Res.*, vol. 40, no. 3-NaN-2, pp. S210–S216, Jun. 1992.

[139] S.-K. Lim and Y.-D. Kim, “An integrated approach to dynamic plant location and capacity planning,” *J. Oper. Res. Soc.*, vol. 50, no. 12, pp. 1205–1216, Dec. 1999.

[140] C. Canel and B. M. Khumawala, “International facilities location: A heuristic procedure for the dynamic uncapacitated problem,” *Int. J. Prod. Res.*, vol. 39, no. 17, pp. 3975–4000, Jan. 2001.

[141] A. Amiri, “Solution procedures for the service system design problem,” *Comput. Oper. Res.*, vol. 24, no. 1, pp. 49–60, Jan. 1997.

[142] A. Amiri, “The design of service systems with queueing time cost, workload capacities and backup service,” *Eur. J. Oper. Res.*, vol. 104, no. 1, pp. 201–217, Jan. 1998.

[143] Q. Wang, R. Batta, and C. M. Rump, “Algorithms for a Facility Location Problem with Stochastic Customer Demand and Immobile Servers,” *Ann. Oper. Res.*, vol. 111, no. 1/4, pp. 17–34, 2002.

[144] S. Elhedhli, “Service System Design with Immobile Servers, Stochastic Demand, and Congestion,” *Manuf. Serv. Oper. Manag.*, vol. 8, no. 1, pp. 92–97, Jan. 2006.

[145] I. Castillo, “Socially Optimal Location of Facilities with Fixed Servers, Stochastic

Demand and Congestion," 2002.

[146] V. Marianov and D. Serra, "Probabilistic, Maximal Covering Location—Allocation Models forCongested Systems," *J. Reg. Sci.*, vol. 38, no. 3, pp. 401–424, Aug. 1998.

[147] V. Marianov and D. Serra, "Location–Allocation of Multiple-Server Service Centers with Constrained Queues or Waiting Times," *Ann. Oper. Res.*, vol. 111, no. 1–4, pp. 35–50.

[148] F. Silva and D. Serra, "Locating emergency services with different priorities: the priority queuing covering location problem," *J. Oper. Res. Soc.*, vol. 59, no. 9, pp. 1229–1238.

[149] Z. Drezner and H. Hamacher, *Facility location : applications and theory*. Springer, 2002.

[150] M. Gendreau, G. Laporte, and F. Semet, "A dynamic model and parallel tabu search heuristic for real-time ambulance relocation," *Parallel Comput.*, vol. 27, no. 12, pp. 1641–1653, Nov. 2001.

[151] S. Ichoua, M. Gendreau, and J.-Y. Potvin, "Vehicle dispatching with time-dependent travel times," *Eur. J. Oper. Res.*, vol. 144, no. 2, pp. 379–396, Jan. 2003.

[152] L. Ozdamar, "Planning helicopter logistics in disaster relief," *OR Spectr.*, vol. 33, no. 3, pp. 655–672, Jul. 2011.

[153] V. Schmid and K. F. Doerner, "Ambulance location and relocation problems with time-dependent travel times," *Eur. J. Oper. Res.*, vol. 207, no. 3, pp. 1293–1303, Dec. 2010.

[154] V. Schmid, "Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming," *Eur. J. Oper. Res.*, vol. 219, no. 3, pp. 611–621, Jun. 2012.

[155] G.-H. Tzeng, H.-J. Cheng, and T. D. Huang, "Multi-objective optimal planning for designing relief delivery systems," *Transp. Res. Part E Logist. Transp. Rev.*, vol. 43, no. 6, pp. 673–686, Nov. 2007.

[156] Y.-H. Lin, R. Batta, P. A. Rogerson, A. Blatt, and M. Flanigan, "Location of temporary depots to facilitate relief operations after an earthquake," *Socioecon. Plann. Sci.*, vol. 46, no. 2, pp. 112–123, Jun. 2012.

[157] K. Jain and V. V Vazirani, "Primal-Dual Approximation Algorithms for Metric Facility Location and -Median Problems," *J. ACM*, vol. 48, no. 2, pp. 274–296, 2001.

[158] G. Xu and J. Xu, "An improved approximation algorithm for uncapacitated facility location problem with penalties," *J. Comb. Optim.*, vol. 17, no. 4, pp. 424–436, May 2009.

[159] D. Du, X. Wang, and D. Xu, "An approximation algorithm for the k-level capacitated facility location problem," *J. Comb. Optim.*, vol. 20, no. 4, pp. 361–368, 2010.

[160] D. Xu and D. Du, "The k-level facility location game," *Oper. Res. Lett.*, vol. 34, no. 4,

pp. 421–426, 2006.

[161] G. Xu and J. Xu, “An LP rounding algorithm for approximating uncapacitated facility location problem with penalties,” *Inf. Process. Lett.*, vol. 94, no. 3, pp. 119–123, May 2005.

[162] D. Xu and S. Zhang, “Approximation algorithm for facility location with service installation costs,” *Oper. Res. Lett.*, vol. 36, no. 1, pp. 46–50, 2008.

[163] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani, “Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP,” *J. ACM*, vol. 50, no. 6, pp. 795–824, Nov. 2003.

[164] R. Ravi and a. Sinha, “Multicommodity facility location,” *Proc. fifteenth Annu. ACM-SIAM Symp. Discret. algorithms*, pp. 342–349, 2004.

[165] M. Mahdian, “Facility location and the analysis of algorithms through factor-revealing programs,” vol. 546, 2004.

[166] M. Charikar, J. Naor, and B. Schieber, “Resource Optimization in QoS Multicast Routing of Real-Time Multimedia,” *IEEE/ACM Trans. Netw.*, vol. 12, no. 2, pp. 340–348, Apr. 2004.

[167] R. Aboolian, Y. Sun, and G. J. Koehler, “A location-allocation problem for a web services provider in a competitive market,” *Eur. J. Oper. Res.*, vol. 194, no. 1, pp. 64–77, 2009.

[168] I. Averbakh, O. Berman, Z. Drezner, and G. O. Wesolowsky, “The uncapacitated facility location problem with demand-dependent setup and service costs and customer-choice allocation,” *Eur. J. Oper. Res.*, vol. 179, no. 3, pp. 956–967, 2007.

[169] M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys, “A constant-factor approximation algorithm for the k -median problem (extended abstract),” *Proc. thirty-first Annu. ACM Symp. Theory Comput. - STOC ’99*, vol. 2, pp. 1–10, 1999.

[170] B. B. Eva K. Lee, Chien-Hung Chen, Ferdinand Pietz, “Modeling and Optimizing the Public-Health Infrastructure for Emergency Response,” *Informs*, vol. 39, no. 5, pp. 476–490, 2009.

[171] R. T. Wong, “Vehicle Routing for Small Package Delivery and Pickup Services,” in *The Vehicle Routing Problem: Latest Advances and New Challenges*, Boston, MA: Springer US, 2008, pp. 475–485.

[172] M. Sviridenko, “An Improved Approximation Algorithm for the Metric Uncapacitated Facility Location Problem,” Springer, Berlin, Heidelberg, 2002, pp. 240–257.

[173] M. Andrews and L. Zhang, “The access network design problem,” in *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280)*, pp. 40–49.

[174] D. B. Shmoys, E. Tardos, and K. Aardal, “Approximation algorithms for facility location problems,” 1998.

[175] M. Charikar and S. Guha, “Improved combinatorial algorithms for facility location

problems," *SIAM J. Comput.*, vol. 34, no. 4, pp. 803–824, 2005.

[176] Y. Li, D. Du, N. Xiu, and D. Xu, "A combinatorial 2.375-approximation algorithm for the facility location problem with submodular penalties," *Theor. Comput. Sci.*, vol. 476, pp. 109–117, 2013.

[177] Y. Li, D. Xu, D. Du, and N. Xiu, "Improved approximation algorithms for the robust fault-tolerant facility location problem," *Inf. Process. Lett.*, vol. 112, no. 10, pp. 361–364, 2012.

[178] C. Bazgan, R. Hassin, and J. Monnot, "Approximation algorithms for some vehicle routing problems *."