Universidade Estadual de Campinas
Instituto de Computação

Marcelo Pinheiro Leite Benedito

Approximation Algorithms for Hub Location Problems

Algoritmos de Aproximação para Problemas de Localização e Alocação de Terminais

CAMPINAS
2018

Marcelo Pinheiro Leite Benedito

Approximation Algorithms for Hub Location Problems

Algoritmos de Aproximação para Problemas de Localização e Alocação de Terminais

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Thesis presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Computer Science.

Supervisor/Orientador: Prof. Dr. Lehilton Lelis Chaves Pedrosa

Este exemplar corresponde à versão final da Dissertação defendida por Marcelo Pinheiro Leite Benedito e orientada pelo Prof. Dr. Lehilton Lelis Chaves Pedrosa.

CAMPINAS
2018

Agência(s) de fomento e nº(s) de processo(s): FAPESP, 2016/12006-1; CAPES

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

B434a

Benedito, Marcelo Pinheiro Leite, 1994-
Approximation algorithms for hub location problems / Marcelo Pinheiro Leite Benedito. – Campinas, SP : [s.n.], 2018.

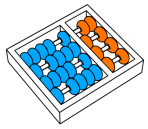
Orientador: Lehilton Lelis Chaves Pedrosa.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Algoritmos de aproximação. 2. Problema de localização e alocação de terminais. 3. Arredondamento de programa linear. 4. Otimização combinatória. I. Pedrosa, Lehilton Lelis Chaves, 1985-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Algoritmos de aproximação para problemas de localização e alocação de terminais
Palavras-chave em inglês:
Approximation algorithms
Hub location problem
Linear program rounding
Combinatorial optimization
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Lehilton Lelis Chaves Pedrosa [Orientador]
Luis Augusto Angelotti Meira
Cid Carvalho de Souza
Data de defesa: 31-07-2018
Programa de Pós-Graduação: Ciência da Computação

Universidade Estadual de Campinas
Instituto de Computação



Marcelo Pinheiro Leite Benedito

Approximation Algorithms for Hub Location Problems

Algoritmos de Aproximação para Problemas de Localização e Alocação de Terminais

Banca Examinadora:

- Prof. Dr. Lehilton Lelis Chaves Pedrosa (Orientador)
  Instituto de Computação - Unicamp
- Prof. Dr. Luis Augusto Angelotti Meira
  Faculdade de Tecnologia - Unicamp
- Prof. Dr. Cid Carvalho de Souza
  Instituto de Computação - Unicamp

A ata da defesa com as respectivas assinaturas dos membros da banca encontra-se no processo de vida acadêmica do aluno.

Campinas, 31 de julho de 2018

Dedicatória

À minha mãe Mitsi, que me ensinou o valor da educação.

Agradecimentos

Agradeço aos meus pais, Carlos e Mitsi, e irmãos, André e Henrique, que sempre apoiaram minhas aspirações e vontades, me dando suporte para continuar seguindo meus sonhos.

Ao meu orientador, Prof. Lehilton, pelos ensinamentos, paciência e amizade desenvolvidas ao longo desses anos.

A todos amigos que conheci neste período, obrigado por compartilharmos um capítulo das nossas histórias e tornar a vida mais divertida.

Aos servidores e professores do Instituto de Computação e de toda a Unicamp, por proporcionarem um ambiente de excelência para todos.

Por último, agradeço à Fundação de Amparo à Pesquisa do Estado de São Paulo (Fapesp) pelo apoio financeiro que possibilitou esta pesquisa (processo 2016/12006-1).

Resumo

No Problema de Localização e Alocação de Terminais, a entrada é um espaço métrico composto por clientes, localidades e um conjunto de pares de clientes; uma solução é um subconjunto das localidades, onde serão abertos terminais, e uma atribuição de cada par de clientes a uma rota, que começa no primeiro cliente, passando em um ou dois terminais, e terminando no segundo cliente. O objetivo é encontrar uma solução que minimize o tamanho de todas as rotas somado com o custo de abertura de terminais. Os algoritmos de aproximação da literatura consideram apenas o caso em que o conjunto de terminais abertos é dado como parte da entrada, e o problema se torna atribuir clientes aos terminais; ou então quando o espaço é definido em classes especiais de grafos. Neste trabalho, apresentamos o primeiro algoritmo de aproximação com fator constante para o problema de, simultaneamente, escolher localidades para abrir terminais e atribuir clientes a estes.

A primeira parte desta dissertação cria algoritmos de aproximação para diversas variantes do problema. A estratégia principal é reduzir os problemas de localização e alocação de terminais aos problemas clássicos de localidades, como o problema de localização de instalações e o problema das k-medianas. A redução transforma uma instância de localização e alocação de terminais em uma instância de um destes problemas, que então é resolvida usando algoritmos de aproximação já existentes na literatura. A saída do algoritmo induz uma solução para o problema original, com uma perda constante no fator de aproximação.

Na segunda parte, o foco é o Problema de Localização e Alocação Única de Terminais (SAHLP), que é uma variação em que cada cliente deve estar conectado a apenas um terminal, além de não haver limite na quantidade de terminais abertos. A principal contribuição é um algoritmo 2.48-aproximado para o SAHLP, baseado em arredondamento de uma nova formulação de programa linear para o problema. O algoritmo é composto por duas fases: na primeira, a solução fracionária é escalada e um subconjunto de terminais é aberto, e na segunda, atribuímos clientes aos terminais abertos. A primeira fase segue o formato padrão de *filtering* para problemas de localidades. A segunda, no entanto, exigiu o desenvolvimento de novas ideias e é baseada em múltiplos critérios para realizar a atribuição. A principal técnica atribui cada cliente ao terminal aberto mais próximo, se este estiver em sua vizinhança; caso contrário, o cliente se conecta ao terminal que melhor balanceia múltiplos custos, relacionados à distância entre eles.

Abstract

In the Hub Location Problem (HLP), the input is a metric space composed of clients, locations and a set of pairs of clients; a solution is a subset of locations to open hubs and an assignment for each pair of clients to a route starting in the first client, passing through one or two hubs and ending in the second client. The objective is to find a solution that minimizes the length of all routes plus the cost of opening hubs. The currently known approximation algorithms consider only the case in which the set of hubs is given as part of the input and the problem is assigning clients to hubs; or when the space is defined on special classes of graphs. In this work, we present the first constant-factor approximation algorithms for the problem of, simultaneously, selecting hubs and allocating clients.

The first part of the thesis derives approximation algorithms for several variants of the problem. The main strategy is to reduce the hub location problems to classical location problems, such as Facility Location and k-Median. The reduction transforms an instance of hub location into an instance of a corresponding location problem, which is then solved by known approximation algorithm. The algorithm's output induces a solution of the original problem within a constant loss in the approximation ratio.

In the second part, we focus on the Single Allocation Hub Location Problem (SAHLP), that is the variant in which a client must be connected to only one hub and there is no limit on the number of open hubs. Our main contribution is a 2.48-approximation algorithm for the SAHLP, based on the rounding of a new linear programming formulation. The algorithm is composed of two phases: in the first one, we scale the fractional solution and open a subset of hub locations, and in the second one, we assign clients to open hubs. The first phase follows the standard filtering framework for location problems. The latter, however, demanded the development of new ideas and is based on a multiple criteria assignment. The main technique is assigning a client to a closest open hub only if there are near open hubs, and otherwise selecting the hub which balances multiple costs.

Contents

Chapter 1

Introduction

Hub Location Problems (HLP) have been a topic of research for more than three decades, which spans various kinds of real life problems and solution techniques. The area started with O'Kelly [66, 67], who discussed several hub location system models and applications, which appear in aviation, telecommunication industries, logistic and postal companies and other types of problems of a communicating network, where nodes are not connected directly to the destination.

These networks have two kinds of interacting nodes, clients and hubs, that must be connected to transport a set of demands between two clients, using one or two hubs of the given set. There exists a cost to connect each pair of nodes and a cost to open a hub; the objective is to find a subset of hubs to be opened and an assignment function, that designates these hubs to clients while minimizing the total cost, that is, the length of all routes plus the cost of opening hubs. In some practical scenarios, the flow between hubs benefits from the economy of scale, and thus the cost of traveling between two hubs is multiplied by a discount factor $\alpha \leq 1$. Airlines and air cargo industries are real life examples of scenarios to which this discount factor is applied. In these applications, a person or shipment is transported using consolidated lines to travel long distances and then can take a smaller route to reach its destination.

As seen in a recent survey by Farahani et al. [38], most problems regarding HLP are based on exact or heuristic solutions, while only few papers focus on approximation algorithms. This type of algorithm rises to deal with NP-hard problems, a class of problems that are thought impossible to be solved in polynomial time — this is the famous conjecture that P $\neq$ NP. With this in mind, the objective of approximation algorithms is to produce in polynomial time a solution whose cost is at most a factor times the optimal one. An approximation algorithm differs from an exact algorithm because the latter does not necessarily run in polynomial time, but always seek for the best solution. Heuristics, on the other hand, do not commit to find a provably good solution, nor to run in polynomial time; the goal here is to yield acceptable results in practical experiments.

Although there are similarities between HLP and heavily studied location problems, such as the Facility Location Problem (FLP) and the k-Median Problem, in regard to approximation approaches, the HLP literature lacks studies involving different kinds of variants and algorithm design techniques. The majority of the works deal with a Hub-and-Spoke type of network, that assigns each client to exactly one hub and the set of

hubs is given in the input, which means that choosing the set of hubs to be opened is not part of the problem. For this variant, Iwasa et al. [50] present a 2-approximation algorithm, Ge et al. provide the same factor using geometric rounding techniques [44] and Ando et al. [4] present a $(1 + 2/\pi)$-approximation algorithm for the case in which nodes are in the Euclidean plane. Approximation algorithms and hardness results have also been given for the variant in which the topology of the hub network is a star [61, 20].

This work studies hub location problems whose objective is to, simultaneously, *select hubs and allocate clients*. A few variants are considered, depending on whether the number of open hubs is given in the input or there is a cost of opening a given hub; and on whether a client must be assigned to a single hub or may be assigned to different hubs. The obtained results may be summarized as follows: first, we present the first constant-factor approximation algorithm for each of the variants above by means of a reduction; second, we present an improved approximation factor for a particular single allocation variant, introducing a new linear programming (LP) formulation and using an LP-rounding algorithm.

In order to derive the constant-factor approximations, we observe that each of these variants is a generalization of a location problem which has been widely studied from the approximation perspective, a fact that we explore to provide both a lower bound on the solution cost and a reduction to the corresponding location problem. Namely, for the multiple allocation problems, we derive a $(\rho+1)$-approximation algorithm, where ρ is the approximation factor of an algorithm for the corresponding location problem; for single allocation problems, the reduction implies a $(2\rho + 1)$-approximation.

For instance, the variant named Single Allocation Median Hub Location Problem (SAHLP) — where each client must be assigned to exactly one hub and there are no restrictions on the number of open hubs — can be reduced to the classical Facility Location Problem, by using the same set of clients and hubs. Using the presented reduction, together with the best approximation factor for FLP, which is 1.488 [58], we obtain a 3.98-approximation algorithm. This is done by creating an instance for FLP based on the original instance for SAHLP and solving it with the algorithm for FLP. Then, we use the approximate solution to this newly created instance to produce a solution for our own problem. In this case, we assign the clients to the same hubs in both solutions and show that this generates an approximate solution for HLP with a slightly bigger factor than the used originally. The same idea is applied to obtain a first approximation for other 5 variants of HLP.

In the second part of this work, we focus on improving the approximation factor for the SAHLP, and present an LP-rounding 2.48-approximation algorithm based on a new linear formulation. The algorithm follows the generic framework of LP-rounding for well studied location problems and is composed of two phases: (i) a phase to scale the fractional solution and cluster the hubs; (ii) a specially crafted phase to assign clients to open hubs. The clustering phase is similar to standard algorithms for FLP (see [11]). However, allocating clients is non-trivial, since each client is not necessarily assigned to the closest hub, as in FLP. The main techniques are: assigning a client to an open hub using multiple criteria, depending on whether there is an open hub in its support set; and balancing the terms of the solution cost (i.e., inbound and outbound connection costs) by

exploiting the symmetries of the linear formulation.

The remaining chapters are organized as follows. Chapter 2 brings a brief introduction to combinatorial optimization problems and approximation algorithms, and also formally defines Hub Location Problems and the corresponding location problems used in the reduction algorithm. Chapter 3 reviews the literature of HLP and other relevant location problems, focusing on approximations and hardness results; furthermore, we briefly review Byrka and Aardal's [11] approximation algorithm for FLP, from which we drew a variety of ideas and results to analyze our algorithm. Chapter 4 presents the reduction algorithms for the various types of HLP. Chapter 5 brings our major contribution, which is an LP-rounding algorithm for SAHLP. Chapter 6 contains the concluding remarks.

Chapter 2

Preliminaries

2.1 Combinatorial Optimization

In a combinatorial optimization problem, a function that maps objects to values is given and the objective is to find the best object from a finite set, either minimizing or maximizing its associated value. In many cases, this set is too big, thus performing an exhaustive search is not viable. To study the complexity of these problems, one normally considers the corresponding decision version — where the question is whether there exists a solution of a given cost —, which can be included in various complexity classes. The most important classes are introduced next.

A problem is in P (polynomial-time) if there exists an algorithm which runs efficiently, i.e., in polynomial-time, and outputs the answer correctly. The NP (non-deterministic polynomial-time) class contains all decision problems that can be efficiently verified by a non-deterministic Turing machine in polynomial time, that is, a non-deterministic machine for which there is an accepting execution path if and only if the solution is valid and, when it does, this path takes polynomial time. For an optimization problem, when solving a problem, it is usual to ask for a solution with best value; and when validating a problem, checking whether a solution is valid and calculating its value.

Given problems P_1 and P_2, there exists a polynomial-time reduction (or Karp reduction) from P_1 to P_2 if, in polynomial time, it is possible to map an instance from P_1 to an instance of P_2, each yes-instance is mapped to a yes-instance, and each no-instance is mapped to a no-instance. In this way, an algorithm for P_2 can be used to solve P_1, and thus P_1 is not more "difficult" than P_2. A problem is said to be NP-hard if every problem from NP can be reduced to it in polynomial-time. This implies that, if there exists an efficient algorithm to solve any NP-hard problem, one can use it to solve all the problems in NP in polynomial time. If a problem is in both NP and NP-hard, then it is said to be NP-complete. A detailed guide to these classes is given in [43].

For an optimization problem, the notion of reduction is different. A reduction from P_1 to P_2 is a pair of polynomial-time transformations, such that the first transformation is given an instance of P_1 and constructs an instance of P_2, and the second transformation receives an optimal solution of the mapped instance of P_2, and constructs an optimal solution for the original instance.

The presented complexity classes are the foundation of many fields, including the

approximation algorithm, whose premise is the assumption that P $\neq$ NP, and thus the optimality requirement must be relaxed. This conjecture is one of the most important unsolved problems in Computer Science, and can be informally stated by the question: is the class of problems that can be efficiently verified the same as the class of problems that can be efficiently solved? Most researchers believe that the answer to this question is no [48]. Under this assumption, NP-hard problems are intractable and thus alternative ways to tackle these problems and find solutions must be studied. Among the possibilities, there are exact algorithms [65], heuristic methods [68] and approximation algorithms [78].

Exact algorithms focus on solving problems optimally as fast as possible. For many problems, the currently known algorithms have only exponential time guarantees in the worst case. In the real-world applications which demand that NP-hard problems are solved in a timely manner, the exponential must have a small base [40]; or, possibly, only a family of "easy" instances may be considered.

Heuristic methods try to solve problems by relaxing the optimality constraint of the solution and, sometimes, the requirement that algorithms are run in polynomial-time. These methods are applied when some insights of the problems are known, which guide the algorithms in the search for a good solution. Such insights may be gathered, for example, by performing empirical analysis, or using frameworks called metaheuristics, which are generic optimization processes commonly employed to solve similar problems.

Approximation algorithms give a mathematically-certified guarantee on the quality of the solution, while maintain the requirement that runtime is polynomial. This sort of algorithm investigates how much a solution obtained in polynomial time can approach the optimal value. It differs from common heuristics because the guarantee of the solution quality and polynomial running time are assured for every input of the problem. While many approximation algorithms are efficient in practice, they are mostly studied on the theoretical basis only, and some may have large polynomial degrees, which turn them impractical for use in real applications. Still, approximation algorithms provide valuable knowledge on the structure of the problems and provide a means of measuring their intrinsic difficulty.

2.1.1 Linear and Integer Programs

Linear Programming (LP) is an optimization technique that formulates problems into a set of n variables under linear constraints and an objective function, whose value must be minimized or maximized. To solve an LP formulation, an algorithm must find a solution (which is a point in the $\mathbb{R}^n$) in the feasible region bounded by the constraints. There exist polynomial-time algorithms to solve LP models (e.g., the interior-point method [54]), therefore the decision version is in P. This technique is widely used in numerous fields of science and is of great importance to the nature of the algorithms presented in this work. Detailed description of linear programming is not in the scope of the text. More information on LPs and their properties can be found in [23].

Given an LP formulation for a problem, called primal, there exists another formulation that is closely related to it, the dual formulation. If the primal has n variables and m constraints, then the dual has m variables, n constraints and a flipped objective goal. The

following results are used in latter sections, where we fix the primal formulation's goal to be minimization.

Theorem 1 (Weak Duality)**.** *Given feasible solutions x and y for the primal and dual linear formulations, respectively, then the cost of x is always greater than or equal to the cost of y.*

Theorem 2 (Strong Duality)**.** *Given optimal solutions x and y for the primal and dual linear formulations, respectively, then the cost of x is equal to the cost of y.*

An Integer Linear Programming (ILP) is another modeling technique which is similar to LP, but such that some variables are restricted to integer values. This, seemingly small, difference turns the decision version of the problem of solving ILP NP-hard. Nonetheless, it is generally use modeling NP-hard problems as ILP formulations, as they describe a problem in an unambiguous language and provide helpful insights. Moreover, there are various approaches to design algorithms with the aid of ILP, as seen in the next subsection.

2.1.2 Approximation Algorithms

Let $\mathcal{A}$ be an algorithm for a minimization problem and I an instance of $\mathcal{A}$. Denote by $\mathcal{A}(I)$ the solution value for I given by the algorithm and by $OPT(I)$ the optimal value for this instance. We call $\mathcal{A}$ an *approximation algorithm* if it runs in polynomial time and the approximation ratio $r_{\mathcal{A}}$, defined as the supremum of the ratio between the produced solution and the optimal value over all instances I of size n,

$$r_{\mathcal{A}} \coloneqq \sup_I \left\{ \frac{\mathcal{A}(I)}{OPT(I)} \right\},$$

is bounded by a function of n. If $r_{\mathcal{A}}$ is bounded by a constant for every n, then we say that $\mathcal{A}$ is a constant approximation. Normally, $r_{\mathcal{A}}$ is not calculated directly, but an upper bound α; in this case, $\mathcal{A}$ is called an α-approximation.

This concept can be extended to define *randomized approximation algorithms*, that receive as input a function that generates random bits according to a particular probability distribution, and is used to create a solution. Since the algorithm is not deterministic, we are interested in giving the approximation factor in terms of the expected solution value, since it consists of a random variable. In many cases, it is possible to derandomize the algorithm, using the method of conditional probabilities [71], yielding a deterministic algorithm.

Approximation algorithms design techniques typically fall into the categories described next. A *combinatorial* algorithm builds a solution without any other algorithmic tool, whereas a *greedy* one performs a series of locally optimal choices, always trying to improve on the previously made decision. Linear programming plays an important role in the design of approximations, and there are many techniques that uses it. A *rounding* algorithm uses the relaxation of the problem's formulation as basis for building an answer, where the decisions on the algorithm are made by observing the values presented in the LP solution. When these variables are binary in the original ILP, they assume a value in

the $[0, 1]$ interval in the relaxation, thus, they can represent a probability which is often used in randomized rounding. An alternative to build this kind of algorithms is to use semidefinite programs, that generalize regular LPs, as seen in [45].

Other LP-based technique is the *primal-dual* method, which makes decisions to compose an integral solution (i.e. primal) by analyzing the dual formulation, typically using a dual-ascent algorithm, that raises the values of the dual variables from zero, maintaining feasibility, while the primal variables respond to those changes until a final solution is given. A variation of the primal-dual method is the *dual fitting*, where a primal integral solution is built, as well as a dual one, with the same cost as the first. This dual solution is infeasible, by weak duality, so the challenge is to find a scaling factor, such that multiplying a solution's variables make it feasible. When such factor is found, it becomes the approximation factor of the algorithm, again, by weak duality, because the dual solution value is a lower bound on the primal optimal.

2.2 Problem Definitions

2.2.1 Hub Location Problem

Consider the sets $\mathcal{C}$ of clients, $\mathcal{H}$ of hubs and $\mathcal{D}$ of unordered pairs of clients which represent demands and let $\mathcal{V} = \mathcal{C} \cup \mathcal{H}$. There is a function to represent the transportation cost to connect clients to hubs or hubs to hubs, $d\colon \mathcal{V}^2 \to \mathbb{R}$, which is assumed to form a metric, and another cost function $f\colon \mathcal{H} \to \mathbb{R}$ to open hubs. Given a pair of clients $\{i, j\} \in \mathcal{D}$, the demand between them needs to be transported through a pair of hubs that are directly connected to i and j, and, to simulate the economy of scale that exists in scenarios modeled by this problem, the transportation cost between hubs undergo a discount factor α, where $0 \leq \alpha \leq 1$. A limit on the number of opened hubs in a solution is dictated by the parameter p. For a client i, define $\mathcal{D}_i$ as the set of clients that are destinations from i. The hubs are assumed to have unlimited capacity, so they are capable of connecting and transporting any amount of clients and demands.

A solution to the HLP is composed of a subset of hubs $\mathcal{H}' \subseteq \mathcal{H}$ to be opened and a function ϕ that assigns clients to opened hubs, $\phi(i) \in \mathcal{H}'$, where $i \in \mathcal{C}$. The objective function is the sum of the costs to open the hubs of $\mathcal{H}'$ plus the cost to connect every pair of demands through a pair of hubs. Some restrictions can be imposed on the given problem definition, which specializes it to particular cases. There can be a limit on the number of hubs a client is connected to, such as only one, or no imposed limit at all. The number of opened hubs may be bounded, that is, at most a value p given in the input. The opening cost function can be defined as $f(h) = 0$, for every $h \in \mathcal{H}$, or given a specific value for each hub.

The hardness proofs of the Hub Location Problem focus on the single allocation case, that is known to be NP-hard. Sohn and Park [74] proved this result by polinomially transforming the Three-Terminal Cut Problem [30] into the single allocation HLP, when the number of hubs is equal to three, thus, the problem is NP-hard as soon as the number of hubs is three. They also proved that if the number of hubs is one or two, the problem is in P.

Each considered variant of HLP is formally defined next.

Definition 1. In the *Single Allocation Median p-Hub Location Problem* (SApHLP), given a tuple $(\mathcal{C}, \mathcal{H}, \mathcal{D}, d, f, \alpha, p)$, one must find a subset of hubs $\mathcal{H}'$ with $|\mathcal{H}'| \leq p$ and an assignment $\phi(i) \in \mathcal{H}'$ for each $i \in \mathcal{C}$. The objective is to find a solution which minimizes

$$\sum_{k \in \mathcal{H}'} f(k) + \sum_{\{i,j\} \in \mathcal{D}} \left[d(i, \phi(i)) + \alpha\, d(\phi(i), \phi(j)) + d(\phi(j), j) \right].$$

Definition 2. The *Single Allocation Median Hub Location Problem* (SAHLP) is the particular case of SApHLP in which there is no limit on the number of opened hubs.

Definition 3. The *Single Allocation Median Fixed p-Hub Location Problem* (SAFpHLP) is the particular case of SApHLP in which there are no costs to open hubs.

Definition 4. In the *Multiple Allocation p-Hub Location Problem* (MApHLP), given a tuple $(\mathcal{C}, \mathcal{H}, \mathcal{D}, d, f, \alpha, p)$, one must find a subset of hubs $\mathcal{H}'$ with $|\mathcal{H}'| \leq p$ and assign each demand $\{i, j\}$ to a pair of hubs $\psi(i, j), \psi(j, i) \in \mathcal{H}'$. The objective is to minimize

$$\sum_{k \in \mathcal{H}'} f(k) + \sum_{\{i,j\} \in \mathcal{D}} \left[d(i, \psi(i, j)) + \alpha\, d(\psi(i, j), \psi(j, i)) + d(\psi(j, i), j) \right].$$

Definition 5. The *Multiple Allocation Median Hub Location Problem* (MAHLP) is the particular case of MApHLP in which there is no limit on the number of opened hubs.

Definition 6. The *Multiple Allocation Median Fixed p-Hub Location Problem* (MAFpHLP) is the particular case of MApHLP in which $f(k) = 0$ for every $k \in \mathcal{H}$ and $|\mathcal{H}'| = p$.

2.2.2 Location Problems

In Chapter 4, we reduce hub location problems to a series of location problem. These problems are defined below.

Definition 7. In the *k-Facility Location Problem* (k-FLP), an instance is comprised of a set of facilities $\mathcal{F}$, a set of clients $\mathcal{C}$, as well as a distance function $d : \mathcal{F} \cup \mathcal{C} \to \mathbb{R}_{\geq 0}$ and an opening cost $f : \mathcal{F} \to \mathbb{R}_{\geq 0}$. Given an instance $(\mathcal{C}, \mathcal{F}, d, f, k)$, the objective is to find a subset of facilities $\mathcal{F}'$ such that $|\mathcal{F}'| \leq k$, which minimizes $\sum_{i \in \mathcal{F}'} f(i) + \sum_{j \in \mathcal{C}} \min_{f \in \mathcal{F}'} d(i, j)$.

Definition 8. The **Facility Location Problem** (FLP) is the case of k-FLP, in which there is no limit on the number of opened facilities.

Definition 9. The **k-Median Problem** is the case of k-FLP in which $f(i) = 0$ for every $i \in \mathcal{F}$ and $|\mathcal{F}'| = k$.

Chapter 3

Literature review

In this chapter we review the HLP literature, especially papers featuring approximation algorithms. Also, we briefly survey Byrka and Aardal's [11] work, which was an inspiration for our own algorithm.

3.1 Hub Location Problem

Hub Location Problems have been the research topics of many studies since the first formal study in the field by O'Kelly [66, 67], in which he brought a quadratic integer program and two simple heuristics, presenting the computational results in real instances composed of a few hubs and up to 25 cities in the USA. Before him, the spatial interaction has been explored by Hakimi [47], that studied the concept of graph centers in order to best locate service stations and gave procedures to find them, while Toh and Higgins [77] discussed the advantages to use hubs in airline industries. The interest in HLP grew in the following decade, with papers focusing on modeling new cases analogous to those found in the location theory literature, such as Campbell [14, 15] with p-Hub Median, p-Hub Center and Hub Covering problems. Aykin merged the HLP with a routing problem [7] and designed exact and heuristic methods to capacitated networks [6, 8].

It is safe to affirm that HLP has been extensively studied under exact and heuristic points of view, with papers exploring integer formulations [28, 3, 79, 36], and other exact methods [32, 33, 24, 42]. A wide range of heuristics were applied; to name a few: genetic algorithm [29, 39, 62], tabu search [64, 18, 80], simulated annealing [7, 37], Lagrangian relaxation [70, 25, 26], and greedy approaches [5, 15, 35, 9]. These and other works can be found in surveys from Campbell et al. [13], Alumur and Kara [2] and Farahani et al. [38], who also cover formulations for numerous variants, classifications and fundamental definitions.

Approximation algorithms for Hub Location Problems are not as abundant in the literature as the other methods presented, which indicates an opportunity to explore some of those variants from a more theoretical perspective and expand the knowledge on these problems. The most studied problem in this point of view is the Single Allocation Hub-and-Spoke Problem, where we are given sets of clients and opened hubs, with the objective to allocate each client to one of the hubs. Note that there is no location phase in

this problem, i.e., all hubs are already opened, thus, it is a subproblem of HLP for we only consider the allocation phase. In some practical cases, it is important to optimally decide the client allocation to hubs for a fixed time interval, where all hubs are available, because of the associated costs of moving equipment on hubs. This problem was first considered by Sohn and Park [73], where they showed how to transform a quadratic 0-1 integer program in the so called fixed two-hub system linear program. They prove that the two-hub variant is polynomially solvable by transforming it into a minimum 2-cut, which is in P [27]; in a later work, this problem is shown to be NP-hard when the number of fixed hubs is greater than or equal to 3, by using a minimum 3-cut reduction [74]. Iwasa et al. [50] introduced two algorithms for the metric case, using different approaches: a simple 3-approximation algorithm that connects clients to the nearest hubs and a 2-approximation algorithm based on a linear programming relaxation and a randomized rounding procedure. They showed that the Single Allocation Hub-and-Spoke Problem is a special case of Metric Labeling, introduced by Kleinberg and Tardos [55], concluding that their results are also valid for this problem when its distances form a metric. When fixed to three hubs/labels, they also gave a (5/4)-approximation algorithm for the single allocation and a (4/3)-approximation algorithm for the labeling.

Ando and Matsui [4] proposed an approximation algorithm for a hub-and-spoke structure when all nodes are embedded in a 2-dimensional plane, where the Euclidean distance is valid and the distance between hub nodes satisfy the Monge property [10], achieving a factor of $1 + 2/\pi$. They applied a dependent rounding procedure to a linear program, the same as Iwasa et al., where both used a tight linearization for general 0-1 quadratic integer program [1] in the problem's natural formulation. Ge et al. [44] also presented a 2-approximation algorithm to the hub-and-spoke structure and a $(\ln n)$-approximation algorithm for the problem with both location and allocation phases, the first of its kind, using a technique known as geometric rounding. The relevance of these algorithms is that they can be used for different problems, like metric labeling and winner determination, matching the best current bounds of the time.

In the p-Center Hub Location Problem, we are given sets of clients, demands and hubs with the objective to assign each client to an opened hub, minimizing the longest path of a demand pair. This version focus on time-sensitive, guaranteed time distribution systems, or on minimizing the maximum dissatisfaction of passengers. Here, the objective is not obtaining the lowest overall cost, but avoiding very expensive costs to demand between individual nodes. The problem was first proposed by Campbell [14], who showed a binary quadratic formulation and polynomial-time algorithms for special cases. Kara and Tansel [53] also worked on modeling the problem, performed experiments to determine a good way to optimally solve it and proved its NP-hardness. Chen et al. [20] showed that this problem is NP-hard to approximate to a ratio $(4/3 - \epsilon)$, unless P = NP, for any $\epsilon > 0$ and provided two algorithms, a linear-time 2-approximation algorithm and a cubic (5/3)-approximation algorithm. They obtained similar results using the same techniques in a variant presented below, which impose a certain topology on the resulting network. Pedrosa et al. [69] gave an initial 3-approximation algorithm for the multiple allocation variant, in which a client cannot be connected to more than one hub.

In 2012, Yaman and Elloumi [79] presented new HLP variants in designing two-level

star networks, considering service quality. A star network is a common topology in the field of communications and it consists in appointing a central hub that receives direct connections of all other peers; a two-level star is obtained by applying the same definition to the peers of the first level of the network, that now serve as a secondary hub for nodes in the second level. This configuration results in a tree, rooted in the central hub, with height of at most 2. In the Star p-Hub Center/Median Problem, a central hub is given in input, alongside with a set of hubs, clients and its demand points; the objective is to assign p hubs to be connected to the central hub, creating the network's first level, and connect each client to exactly one of those. They are interested in incorporating a measure of quality in the networks: in the first variant, called center, the objective is to create a network that minimizes the longest path between any two demand nodes; in the second, the aim is to find a network such that the total cost of routing is minimum and the length of the path connecting any two demand nodes doesn't exceed a predetermined value. Besides introducing these variants, they proved the problem's NP-hardness and also created formulations to computationally run them and report the performance of the techniques involved. Liang [61] came up with the first approximation algorithm for the center variant, with factor 3.5, that is based on a reduction to the k-Center Problem, using its simple 2-approximation algorithm. The first NP-hardness result for the problem, obtained using a reduction from the dominating set problem [43], was given in this paper: the problem does not admit a $(1.25 - \epsilon)$-approximation algorithm, for any $\epsilon > 0$, unless P = NP. Chen et al. [19] also studied this problem from the NP-hardness point of view, strengthening the lower bound to $(1.5 - \epsilon)$, for any $\epsilon > 0$, unless P = NP. Moreover, they provided two new combinatorial approximation algorithms answering Liang's question whether the gap could be diminished, with factors 2 and 5/3, running times $O(n)$ and $O(pn^4)$, respectively. In the following year, Chen et al. [21] enhanced their results creating a new approximation algorithm to a version of the problem where the triangle inequality is parameterized by β: given points a, b, c and cost function w, it follows that $w(a, c) \leq \beta(w(a, b) + w(b, c))$.

3.2 Location Problems

In this section we will review the main results of related location problems used to obtain our algorithms, either by direct reduction or applying similar concepts and techniques.

3.2.1 Facility Location Problem

The Facility Location Problem has been studied for more than 50 years [57, 75], with books spanning diverse topics and variations [34, 31]. Unlike the HLP, the FLP has received great attention in the approximation algorithms front that boosted this field altogether, creating and refining existing techniques over time. For general distance functions, Hochbaum [49] presented a $O(\log n)$-approximation algorithm that matches the problem's lower bound, meaning that there is no hope to create a better algorithm, unless P = NP, since the Set Cover Problem can be easily reduced to the FLP. Since then, people focused on the FLP with metric distances, i.e., the distance function obeys the

triangle inequality. Results in this variant's approximation hardness were given by Guha and Khuller [46] and Sviridenko [76]: there can be no algorithm with a better factor than 1.463, unless P = NP. Throughout the years a considerable amount of constant-factor approximation algorithms were created using different techniques, from which we can highlight the first LP rounding by Shmoys [72], with factor 3.16; Jain and Vazirani [52] primal-dual method, that achieved a factor of 3; Charikar and Guha [16] merged both methods and got a 1.728-approximation algorithm.

In 2010, Byrka and Aardal [11] modified the $(1 + 2/e)$-approximation of Chudak [22], obtaining a bifactor $(1.6774, 1.3738)$-approximation algorithm, i.e., it achieves a cost of, at most, $1.6774 \cdot F^* + 1.3738 \cdot C^*$, where F^* and C^* are the optimal costs of opening facilities and connecting clients, respectively. It was the first algorithm to touch the approximability limit curve $(\gamma_f, 1 + 2e^{-\gamma_f})$ established by Jain et al. [51], meaning that, if an instance is dominated by connection costs then this algorithm performs as good as the best possible approximation for FLP. They also showed that, when combined with the $(1.11, 1.7764)$-approximation of Jain et al. [51], they produce a 1.5-approximation algorithm that is governed by the scaling factor γ, which was a fixed value in this work. A few years later, Li [59] presented an explicit distribution for this scaling factor, obtaining the best currently known factor guarantee, a 1.488-approximation algorithm. These are important papers in this work's arrangement, for the algorithm in Chapter 5 is build up on similar ideas as those.

3.2.2 k-Median Problem

The k-Median Problem can be applied to similar situations as the FLP, but with the difference on the number of opened facilities: in the latter, there is a cost on opening facilities and no limit on the number of open facilities, whereas in the k-Median an exact number k of facilities must be opened. This fact has led the first approximations to violate this constraint by opening $k + \Omega(k)$ facilities [63, 56]. Charikar et al. [17] came up with the first algorithm that opens exactly k facilities and has a constant approximation factor, using ideas inspired by those applied in FLP. Jain and Vazirani [52] also exploited the similarity between these problems to create an algorithm that uses their primal-dual algorithm to FLP as subroutine, noting that the Lagrangian relaxation of k-Median is the FLP, achieving a factor of 6. Whereas the best known factor was not improved, they introduced a generic framework to derive algorithms of this kind.

The currently best approximation algorithm for k-Median is due to Byrka et al. [12], using a framework composed of dependent rounding and a new way to handle positive correlation in these types of problems, resulting in an approximation factor of $2.675 + \epsilon$. There is an approximation hardness result for this problem, given by Jain et al. [51], which affirms that k-Median is hard to approximate within a factor 1.736.

3.3 Review of Byrka and Aardal's algorithm

This section reviews the algorithm for FLP by Jaroslaw Byrka and Karen Aardal, entitled "An Optimal Bifactor Approximation Algorithm for the Metric Uncapacitated Facility

Location Algorithm Problem" [11]. In Chapter 5, we develop a new algorithm for the Single Allocation Hub Location Problem, which is build on some ideas of the algorithm presented in this section.

The algorithm is an example of application of the LP-rounding technique. Recall that first, one solves relaxation of an integer program formulation, obtaining a fractional solution. This solution is then rounded to an integral one, using various strategies, so as to construct a feasible solution whose value can be bounded as a factor times the value of the fractional solution. This way, this factor expresses a bound on the approximation factor of the algorithm, since the fractional value is a lower bound on the value of an optimal solution.

In our algorithm, we will make use of some lemmas presented in this section, and which are now standard in the literature of FLP. While these lemmas are defined for FLP, in the corresponding steps of our analysis we have equivalent conditions. These proofs are accredited to Byrka and Aardal, but we also used arguments by Shi Li [59]. For the sake of completeness, we reproduce the proofs here.

The following integer program formulates the FLP, in which variable x_{ij} indicates whether client j is connected to facility i and y_i indicates if facility i has been chosen to be opened. Constraints (3.1) designate exactly one facility to each client, while constraints (3.2) ensure each designated facility is included in the set of opened facilities; constraints (3.3) correspond to the integrality of the solution.

$$\text{minimize } \sum_{i\in\mathcal{F}} y_i f(i) + \sum_{i\in\mathcal{F}, j\in\mathcal{C}} x_{ij} d(i,j)$$

$$\text{subject to } \sum_{i\in\mathcal{F}} x_{ij} = 1 \qquad \forall\, j \in \mathcal{C}, \tag{3.1}$$

$$x_{ij} - y_i \leq 0 \qquad \forall\, i \in \mathcal{F}, \quad \forall\, j \in \mathcal{C}, \tag{3.2}$$

$$x_{ij}, y_i \in \{0,1\} \qquad \forall\, i \in \mathcal{F}, \quad \forall\, j \in \mathcal{C}. \tag{3.3}$$

A clustering procedure is now discussed, that creates a structure from the fractional solution of the LP and helps to estimate the final connection cost. Many approximation algorithms for FLP are based on clustering procedures [72, 22, 76] and later we show that a similar strategy pays off for the HLP. The idea is to create a graph composed of clients and facilities, connecting a pair if they are used in the optimal LP solution. Formally, we create a bipartite graph $G = (\mathcal{C}, \mathcal{F}, E)$, called a *support graph*, and include an edge (i, j) in E if variable x_{ij} for client $j \in \mathcal{C}$ and facility $i \in \mathcal{F}$ is positive. The set of facilities which are adjacent to a client j is called the *support set* of j, and is denoted by $N(j)$. If two clients are connected to the same facility in the support graph they are called *neighbors*.

The clustering algorithm is as follows: while there are unclustered clients, choose a client as a new *cluster center* according to a greedy criterion (defined below), and build a cluster with this client as a center and including all of its neighbors. After the clustering is constructed, the algorithm opens facilities: for each cluster, the fractional opening in the support set of a center must sum to one (according to constraint (3.1)), thus each

facility i in this set is opened with probability x_{ij}; later, for each facility i not in a cluster, open it independently with probability y_i.

Since no two centers are neighbors in the support graph, the cost of each facility is charged to at most one client, which allows bounding the opening cost in terms of the fraction opening. Also, each client is close to one open facility, since it is either a center, or is a neighbor of a center. This situation is depicted in Figure 3.1, with cluster center j' and opened hub i, where client j can connect to i using a path of length three.

If the algorithm above is applied directly to the solution (x, y) of the relaxation, then the obtained solution is very economical, since each facility i is opened with probability y_i and the expected cost is at most the fractional opening cost, given by the relaxed solution of the problem. The connection cost, however, is bounded by a more expensive factor. Therefore, before running the clustering algorithm, two previous steps are executed, called *scaling* and *filtering*.

To balance opening and connection costs, in the *scaling* step, one multiplies the solution by a factor $\gamma > 1$, so that each facility i is opened with probability γy_i, but clients are connected to closer facilities with higher probability. In the *filtering* step, one returns a modified solution $(\bar{x}, \bar{y})$, where $\bar{y} = \gamma y$ and $\bar{x}$ is defined so that each client is (fractionally) connected to the closest opened facilities. If a variable $\bar{y}_i$ for a facility i is greater than one, this step also splits it in order to keep the variables $\bar{y}$ a valid probability distribution.

The support set of j with respect to the modified solution $(\bar{x}, \bar{y})$ is the set of *close* facilities, and is denoted by $Cl(j)$. The facilities which are in the support set according to the original solution, but are not close are the *distant* facilities, denoted by $Di(j)$. To simplify the algorithm and analysis, one assumes the sum of opening close facilities is exactly $1/\gamma$, that is, $\sum_{i \in Cl(j)} y_i = 1/\gamma$. If this is not the case, then one can create an equivalent instance and solution for which this is the case and has no larger value. These procedures are said to transform the solution into a *complete solution*, and is standard in approximation algorithms for FLP. Similar arguments are also used in our algorithm for HLP, so we refer to Chapter 5 and the original paper for the complete details.

It is useful to consider the distance from a client to a set of fractionally open facilities. Given a client $j \in \mathcal{C}$ and a subset of facilities $\mathcal{F}' \subseteq \mathcal{F}$, the average distance from j and to $\mathcal{F}'$ and the maximum distance from j and to $\mathcal{F}'$ are

$$d(j, \mathcal{F}') = \frac{\sum_{i \in \mathcal{F}'} c_{ij} y_i}{\sum_{i \in \mathcal{F}'} y_i}, \qquad d_{max}(j, \mathcal{F}') = \max_{i \in \mathcal{F}'} c_{ij}.$$

In particular, the average distance to close facilities, the average distance to distant facilities, and the maximum distance to a close facility are

$$c_j^c = d(j, Cl(j)), \qquad c_j^d = d(j, Di(j)), \qquad \tilde{c}_j^c = d_{max}(j, Cl(j)).$$

Once the solution is filtered, the clustering algorithm is run on the modified solution, selecting as the next center the unclustered client j whose sum $c_j^c + \tilde{c}_j^c$ is minimum. Observe that $\tilde{c}_j^c \leq c_j^d$ since the furthest close facility is not further than the closest distant facility.

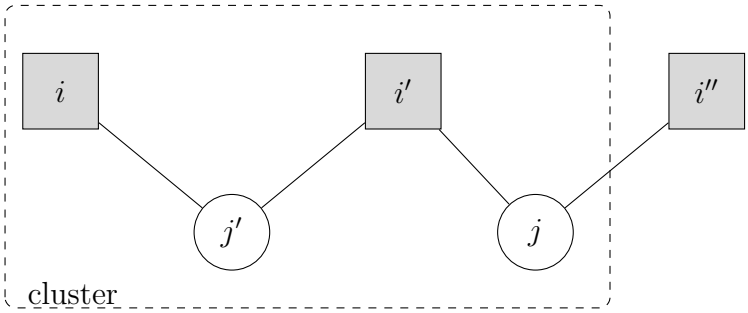


Figure 3.1: Cluster with center j', enclosing its connected facilities and neighbors in the support graph.

The algorithm is detailed next:

1. Solve the LP-relaxation and obtain a fractional solution (x, y);
2. Modify the solution with the procedures above, resulting in a new solution $(\bar{x}, \bar{y})$;
3. Cluster the clients, choosing as center the client j which minimizes the value of $d(j, Cl(j)) + d_{max}(j, Cl(j))$;
4. For each center j, open facility $i \in Cl(j)$ with probability $\bar{y}_i$;
5. For each facility i that is not close to some center, open i independently with probability $\bar{y}_i$;
6. For each client, connect it to the closest open facility.

Definition 10. For some subset $\mathcal{F}' \subseteq \mathcal{F}$ of facilities, define the *volume* of $\mathcal{F}'$, denoted by $\text{vol}(\mathcal{F}')$, to be the sum of the modified facility-opening variables, $\bar{y}_i$, over all facilities in this set: $\text{vol}(\mathcal{F}') = \sum_{i \in \mathcal{F}'} \bar{y}_i$.

The following lemma upper bounds the average distance from a client j to another the set of facilities that are close to its center j', but are not in the support of j. This lemma is due to Shi Li [59], and extends the result of Byrka and Aardal to any value of scaling parameter $\gamma \geq 1$.

Lemma 1. *If $j \in \mathcal{C}$ is a client and j' is the associated cluster center, then for any $\gamma \geq 1$*

$$d(j, Cl(j') \setminus N(j)) \leq (2 - \gamma)\tilde{c}^c_j + (\gamma - 1)c^d_j + \tilde{c}^c_{j'} + c^c_{j'}.$$

Proof. Since client j is in the cluster of j', then $Cl(j) \cap Cl(j') \neq \emptyset$ and $c^c_{j'} + \tilde{c}^c_{j'} \leq c^c_j + \tilde{c}^c_j$. We can assume that

$$d(j, j') \geq (2 - \gamma)\tilde{c}^c_j + (\gamma - 1)c^d_j + c^c_{j'}, \tag{3.4}$$

because if we sum the missing term, the maximum distance from j' to a facility in its close set, it is enough to reach any facility in $Cl(j')$ from j'. Applying the bound $\tilde{c}^c_j \leq c^d_j$

in this inequality, we get that $d(j, j') \geq \tilde{c}^c_j + c^c_{j'}$. Since $d(j, Cl(j) \cap Cl(j')) \leq \tilde{c}^c_j$, we have $d(j', Cl(j) \cap Cl(j')) \geq c^c_{j'}$.

If the average distance from j' to the intersection of both close sets of j and j' is greater than or equal to $c^c_{j'}$, then $d(j', Cl(j') \setminus N(j)) \leq c^c_{j'}$ and the lemma follows because $d(j, j') \leq \tilde{c}^c_j + \tilde{c}^c_{j'} \leq (2-\gamma)\tilde{c}^c_j + (\gamma - 1)c^d_j + \tilde{c}^c_{j'}$. If not, then

$$d(j', Di(j) \cap Cl(j')) = c^c_{j'} - z, \tag{3.5}$$

for some $z > 0$. In Figure 3.2 the sets involved in the proof are pointed.

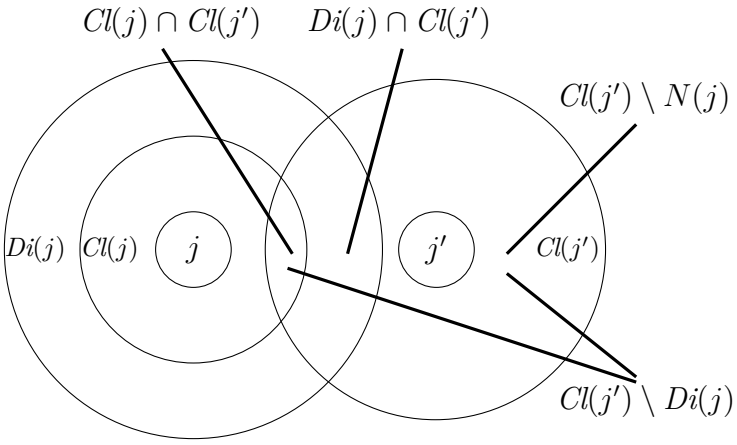


Figure 3.2: Groups of facilities of j and j'.

Let $\hat{y} = \text{vol}(Di(j) \cap Cl(j'))$, the volume of the given area. This value is, at most, $\max\{\gamma - 1, 1\}$, for the sum of the scaled variables related to the close facilities of j is equal to 1. Equation (3.5) implies that

$$d(j', Cl(j') \setminus Di(j)) = c^c_{j'} + \frac{\hat{y}}{1-\hat{y}} z. \tag{3.6}$$

Using the triangle inequality between j, j' and the set of facilities $Di(j) \cap Cl(j')$ as well as facts (3.4) and (3.5), we can subtract one distance from another and get a lower bound of the distance of j to $Di(j) \cap Cl(j')$:

$$\begin{aligned} d(j, Di(j) \cap Cl(j')) &\geq (2-\gamma)\tilde{c}^c_j + (\gamma - 1)c^d_j + z \\ &= c^d_j - (2-\gamma)(c^d_j - \tilde{c}^c_j) + z. \end{aligned} \tag{3.7}$$

Using the natural upper bound and (3.7), we'll find a lower bound on $c_j^d - \tilde{c}_j^c$:

$$\begin{aligned}
\tilde{c}_j^c &\leq d(j, Di(j) \setminus Cl(j')) \\
&\leq c_j^d - \frac{\hat{y}}{\gamma - 1 - \hat{y}}(z - (2-\gamma)(c_j^d - \tilde{c}_j^c)) \\
c_j^d - \tilde{c}_j^c &\geq \frac{\hat{y}}{\gamma - 1 - \hat{y}}(z - (2-\gamma)(c_j^d - \tilde{c}_j^c)) \\
&\geq \frac{\hat{y}}{\gamma - 1 - \hat{y}} z \Big/ \left(1 + \frac{(2-\gamma)\hat{y}}{\gamma - 1 - \hat{y}}\right) \\
&= \frac{\hat{y}z}{(\gamma - 1)(1 - \hat{y})}. \qquad (3.8)
\end{aligned}$$

We used the fact that the denominator of the last inequality is non-negative. Combining (3.4) and (3.8):

$$\begin{aligned}
d(j', Cl(j) \cap Cl(j')) &\geq d(j, j') - d(j, Cl(j) \cap Cl(j')) \\
&\geq (2-\gamma)\tilde{c}_j^c + (\gamma - 1)c_j^d + c_{j'}^c - \tilde{c}_j^c \\
&= (\gamma - 1)(c_j^d - \tilde{c}_j^c) + c_{j'}^c \\
&\geq \frac{\hat{y}}{1 - \hat{y}} z + c_{j'}^c. \qquad (3.9)
\end{aligned}$$

From (3.7) and (3.9):

$$d(j', Cl(j') \setminus N(j)) \leq c_{j'}^c + \frac{\hat{y}}{1 - \hat{y}} z.$$

Combining all those values, we complete the lemma:

$$\begin{aligned}
d(j, Cl(j') \setminus N(j)) &\leq \tilde{c}_j^c + \tilde{c}_{j'}^c + d(j', Cl(j) \cap Cl(j')) \\
&\leq (2-\gamma)\tilde{c}_j^c + (\gamma - 1)\left(c_j^d - \frac{\hat{y}z}{(\gamma - 1)(1 - \hat{y})}\right) \\
&\quad + \tilde{c}_{j'}^c + c_{j'}^c + \frac{\hat{y}}{1 - \hat{y}} \\
&= (2-\gamma)\tilde{c}_j^c + (\gamma - 1)c_j^d + \tilde{c}_{j'}^c + c_{j'}^c. \qquad \square
\end{aligned}$$

The following lemma provides an upper bound on the expected distance from a client to the closest facility opened by the algorithm within a certain subset of facilities.

Lemma 2. *Let $y \in \{0,1\}^{|\mathcal{F}|}$ be a random binary vector encoding the facilities opened in steps 4 and 5 of the algorithm. The following inequality holds for any subset $A \subseteq \mathcal{F}$ of facilities, such that $\sum_{i \in A} \bar{y}_i > 0$ and any client $j \in \mathcal{C}$:*

$$\mathbb{E}\left[\min_{i \in A, y_i = 1} c_{ij} \Big| \sum_{i \in A} y_i \geq 1\right] \leq d(j, A).$$

Proof. Note that there exist disjoint sets $A_1, A_2, \ldots \subseteq A$ corresponding to the clusters,

that are negatively correlated. By construction, there is at most one facility in each subset A_k and the probability that one is opened if equal to $\sum_{i \in A_k} \bar{y}_i$.

We will analyze a suboptimal procedure to assign clients to open facilities: consider an assignment algorithm that replaces each set A_k by a new facility i_k, creating a new instance, with distance $d(j, A_k)$ from client j and fractional opening $\bar{y}_{i_k} = \sum_{i \in A_k} \bar{y}_i$. Then, for each client, the algorithm chooses the closest opened facility i_k, which is the same as choosing the only opened facility in the subset A_k, in the original instance.

Using the described algorithm, a client may be connected to a facility that is not the closest opened one: if a subset have a high mean value but a close facility to the client is opened, it will not be chosen. Nevertheless, we will show that the greedy assignment of j to the closest open facility in the modified instance results in the expected connection cost at most equal to $d(j, A)$, which translates to the suboptimal assignment in the original instance and therefore implies that in the optimal assignment in the original instance, the expected connection cost is also at most $d(j, A)$.

Consider the facilities from A in the order $i_1, i_2, \ldots$ of nondecreasing distance from j. Since their opening is independent, the probability that i_l counts as the closest among the open facilities is

$$\begin{aligned} p_l &= \prod_{i=i_1}^{i<i_l} \Pr[y_i = 0] \cdot \Pr[y_{i_l} = 1] \\ &= \prod_{i=i_1}^{i<i_l} (1 - \bar{y}_i) \cdot \bar{y}_{i_l}. \end{aligned}$$

The result follows from:

$$\begin{aligned} \mathbb{E}\left[\min_{i \in A, y_i = 1} c_{ij} \Big| \sum_{i \in A} y_i \geq 1 \right] &= \frac{\sum_{l=1}^{|A|} p_l c_{i_l j}}{\sum_{l=1}^{|A|} p_l} \\ &= \frac{\sum_{l=1}^{|A|} (\prod_{o=1}^{l-1} (1 - \bar{y}_{i_o})) \bar{y}_{i_l} c_{i_l j}}{\sum_{l=1}^{|A|} (\prod_{o=1}^{l-1} (1 - \bar{y}_{i_o})) \bar{y}_{i_l}} \\ &\leq \frac{\sum_{l=1}^{|A|} \bar{y}_{i_l} c_{i_l j}}{\sum_{l=1}^{|A|} \bar{y}_{i_l}} \\ &= d(j, A). \end{aligned}$$

In the third step, we used a result from Fortuin et al. [41], that extends Chebyshev's sum inequality. □

Since each facility is opened with probability γ, the expected opening cost of the the returned solution is $\sum_{i \in \mathcal{F}} \gamma y_i$, which is γ times the fraction opening in the objective function of the linear program. To complete the analysis of the algorithm, the expected connection cost of a given client j must be bounded. To this, one analyzes three disjoint events, depending on the set in which the closest facility to j is included: in the close set, in the distant set, or in neither (when one considers that j is connected to a facility in the close to its cluster center). The obtained expected cost has the form $f(\gamma) \sum_{i \in \mathcal{F}} c_{ij} x_{ij}$, where f is a function that depends only on γ.

From the obtained bounds, the total expected cost is bounded as $\gamma F^* + \gamma C^*$, where F^* is the fractional opening cost, and C^* is the fractional connection cost of the optimal fractional solution. Then, opening and connection costs are balanced, by making $\gamma = 1.68$, for which $\gamma = f(\gamma)$. Thus, the total expected cost is bounded by $1.68(C^*+F^*)$, obtaining a 1.68-approximation[1]. We omit the complete details to bound the opening and connection costs, as they are not used by our algorithm in Chapter 5.

[1]In Byrka and Aardal's work, they also balance this factors with the algorithm of Jain et al. [51], leading to a 1.5-approximation.

Chapter 4

Reduction-Based Approximation Algorithms for HLP

In this section, we derive the first approximation algorithm for the HLP variants presented in Chapter 2. The algorithm strategy is to reduce a HLP problem to a corresponding location problem. Recall that in a reduction, an instance of the problem we want to solve is transformed into an instance of another problem, which is known to have an approximation algorithm. Then, we solve the transformed instance using this algorithm, and use the returned solution to build a solution to the original problem. Formally, let A be the problem we want to solve and B the problem for which we have a ρ-approximation algorithm, called ALG_B. Also let I_A be an instance of A and I_B the mapped instance of B. In this section, A is a variant of HLP. The choice of B depends on the characteristics of the HLP problem: the number of hubs to which a client may be connected; the limit on the number of open hubs; and the cost to open the hubs.

In our reduction, the problem B will be k-FLP, or one of its particular cases, FLP or k-Median. The problem the SApHLP will be reduced to is the k-FLP; for SAHLP, the problem is FLP; and for the SAFpHLP, the k-Median variant.

The reduction scheme is summarized below.

1. Starting with I_A, build an instance I_B.
2. Solve I_B using ALG_B, obtaining a ρ-approximate solution S_B.
3. Starting with S_B, build a solution S_A.

Constructing I_B: Recall that I_A has the form of a tuple $I_A = (\mathcal{C}, \mathcal{H}, \mathcal{D}, d, f, \alpha, p)$. Create an instance $I_B = (\mathcal{H}, \mathcal{C}', d, f, p)$. Instance I_B contains the same set of facility locations $\mathcal{H}$, distance d, opening cost function f, and limit on the number of open facilities p as instance I_A. The set of clients $\mathcal{C}'$ is constructed as follows: for each demand $\{i, j\} \in \mathcal{D}$, add to $\mathcal{C}'$ a new client at the same location of i, and a new client at the same location of j.

To build the solution S_A, the clients must be connected to open hubs, respecting the nature of the considered HLP variant (according to the number of connections a client

can have). First, we consider the multiple allocation variants. Later, we will describe how to extend the results for single allocation case.

4.1 Multiple allocation variants

Observe that the set of open facilities in S_B corresponds to a set of open hubs in S_A, such that S_A and S_B have the same set of open hubs. From now on, the terms hub and facility will be used interchangeably.

Let S_A^* and S_B^* be optimal solutions of I_A and I_B, respectively. The value of S_A^* can be expressed as $OPT_A = C^* + \alpha E^* + H^*$, where C^* is the cost to connect clients to hubs, E^* is the connection cost of hub to hub link, and H^* is the opening cost of hubs. The value of S_B^* can be expressed as $OPT_B = G^* + F^*$, where G^* is the cost to connect clients to facilities, and F^* is the opening cost of facilities.

In the lemma below, the solution cost of the optimal solutions are related.

Lemma 3. *The optimal value for instance I_B is bounded by $C^* + H^*$, that is, $G^* + F^* \leq C^* + H^*$.*

Proof. Starting with S_A^*, let us build a viable solution for B, named S'_B, that uses the same set of opened hubs as S_A^* and assigns each client to the closest opened hub. For a client $i \in \mathcal{C}$, let $\phi(i)$ be the closest hub to i and $\phi^*(i)$ be the hub this client is connected to in an optimal solution S_A^*. Note that these hubs are not necessarily the same for a fixed client.

Let the cost of S'_B be $G + F$, where G corresponds to the connection cost and F corresponds to the opening cost. Using the definitions of functions ϕ and ϕ^*, it follows that:

$$\begin{aligned} G &= \sum_{i \in \mathcal{C}} |\mathcal{D}_i| \; d(i, \phi(i)) \\ &\leq \sum_{i \in \mathcal{C}} |\mathcal{D}_i| \; d(i, \phi^*(i)) && (4.1) \\ &= \sum_{\{i,j\} \in \mathcal{D}} (d(i, \phi^*(i)) + d(j, \phi^*(j)) && (4.2) \\ &= C^*. \end{aligned}$$

Note that we defined a weighted connection cost of a client i, meaning that the cost of a link between the client and the assigned hub will be payed for each time a demand point includes i. In step (4.1), the cost of connecting i to any hub cannot be less than connecting it to the closest hub and in (4.2) we simply split from a fixed client to a fixed demand point of view.

Seeing that S'_B opens the same set of hubs as S_A^*, we know that $F = H^*$, so the total cost of this solution is $cost(S'_B) = G + F \leq C^* + H^*$. Since S'_B is a viable solution to B, its cost is an upper bound to OPT_B. Therefore, the proof is complete. □

From the previous result, the cost of solution S_B for instance I_B can be written in terms of the optimal cost of I_A, since S_B is a ρ-approximate solution.

Corollary 1. *The cost of solution S_B is, at most, $\rho(C^* + H^*)$.*

We will now describe how to construct S_A given the set of hubs that were opened by S_B. Since the subset of hubs to be used is already defined, we must only connect the clients to hubs to create an answer to I_A. For a multiple allocation variant of HLP, the decision on where to connect clients can be made analyzing only the two clients in a demand point, for a client can be connected to any number of distinct hubs. In our algorithm, for each each demand $\{i, j\}$, the corresponding clients are connected to the pair of hubs that achieves minimum connection cost. The complete reduction is described in Algorithm 1.

Algorithm 1: Reduction algorithm for multiple allocation HLP.

Input : I_A
Output: S_A, ψ
$S_A \leftarrow \emptyset$
$I_B \leftarrow$ convert I_A into an instance of B
$S_B \leftarrow ALG_B(I_B)$ // obtains a ρ-approximation for I_B
foreach $\{i, j\} \in \mathcal{D}$ **do**
  $k^*, l^* \leftarrow \arg\min_{k,l \in S_B} \{d(i, k) + \alpha d(k, l) + d(l, j)\}$
  $\psi(i, j) \leftarrow k^*, \quad \psi(j, i) \leftarrow l^*$ // assign hubs k^* and l^* to $\{i, j\}$
  $S_A \leftarrow S_A \cup \{k^*, l^*\}$ // add hubs k^* and l^* to solution
return S_A, ψ

For the sake of simplicity, we analyze a (possibly suboptimal) algorithm in which i, j are connected through the hubs $\phi(i), \phi(j)$ that are the closest open hubs of S_B to i and j, respectively. The cost of S_B accounts for the links between clients and hubs for a demand $\{i, j\}$, which are the costs of connecting i to hub $\phi(i)$, and j to hub $\phi(j)$. However, the cost of S_B does not account the cost of interconnecting hubs. Since this value is not given by S_B, it must be bounded in terms of different lower bounds.

Recall that $\phi^*(i)$ represents the hub to which a client i is connected in the optimal solution S_A^*. For each demand $\{i, j\}$, Figure 4.1 depicts two paths to bound the route cost, where the dotted lines represent distances to which the discount factor α was applied. In π_1, clients i and j are assigned to $\phi(i)$ and $\phi(j)$, respectively, and the connection cost between these hubs is bounded by using the path from i to j in the optimal solution, considering the lengths multiplied by α. In π_2, both clients are connected to the same hub $\phi(i)$; here, one assumes w.l.o.g. that $d(i, \phi(i)) \leq d(j, \phi(j))$, and the cost from j to the hub is bounded using the optimal path with no discount factor.

Figure 4.1: Paths π_1 and π_2 used to bound the approximate solution value.

Observe that the connection cost of $\{i, j\}$ incurred by the solution returned by Algorithm 1 is not larger than the length of the smallest path between π_1 and π_2. The following lemmas calculate the cost of using paths π_1 and π_2 for every demand.

Lemma 4. *When all demand points use the assignment of π_1, the total transportation cost is at most $\alpha C^* + \alpha E^* + (1 + \alpha)G$.*

Proof. In π_1, all clients are connected to the same hub they were designated in S_B, given by the ϕ function. Define $c_{ij}^{(1)}$ as the cost to transport demand $(i, j) \in \mathcal{D}$ through hubs $\phi(i)$ and $\phi(j)$, as in Figure 4.1. We get:

$$\begin{aligned} c_{ij}^{(1)} &= d(i, \phi(i)) + \alpha d(\phi(i), \phi(j)) + d(j, \phi(j)) \\ &\leq d(i, \phi(i)) + \alpha \left(d(i, \phi(i)) + d(i, \phi^*(i)) + d(\phi^*(i), \phi^*(j)) + d(\phi^*(j), j) + d(j, \phi(j)) \right) \\ &\quad + d(j, \phi(j)) \\ &= (1 + \alpha) \left(d(i, \phi(i)) + d(j, \phi(j)) \right) + \alpha \left(d(i, \phi^*(i)) + d(\phi^*(i), \phi^*(j)) + d(\phi^*(j), j) \right). \end{aligned}$$

If all demands use the corresponding path π_1, the total transportation cost of the solution, defined as $T^{(1)}$, is obtained by adding the cost of each demand:

$$\begin{aligned} T^{(1)} &= \sum_{\{i,j\} \in \mathcal{D}} c_{ij}^{(1)} \\ &\leq (1 + \alpha) \sum_{\{i,j\} \in \mathcal{D}} \left(d(i, \phi(i)) + d(j, \phi(j)) \right) \\ &\qquad + \alpha \sum_{\{i,j\} \in \mathcal{D}} \left(d(i, \phi^*(i)) + d(\phi^*(i), \phi^*(j)) + d(\phi^*(j), j) \right) \\ &= \alpha C^* + \alpha E^* + (1 + \alpha) \sum_{\{i,j\} \in \mathcal{D}} \left(d(i, \phi(i)) + d(j, \phi(j)) \right) \\ &= \alpha C^* + \alpha E^* + (1 + \alpha) \sum_{i \in \mathcal{C}} |\mathcal{D}_i| \; d(i, \phi(i)) \\ &= \alpha C^* + \alpha E^* + (1 + \alpha) G. \end{aligned}$$

□

Lemma 5. *When all demand points use the assignment of π_2, the total transportation cost is at most $C^* + E^* + G$.*

Proof. Given a demand point and using the strategy applied in π_2, a client may be connected to the hub of its destination, which makes the assignment local, that is, a client may have more than one hub. The discount factor is not used in this case, since both clients are connected to the same hub. The transportation cost of an individual demand point $\{i, j\}$, defined as $c_{ij}^{(2)}$, is:

$$\begin{aligned} c_{ij}^{(2)} &= d(i, \phi(i)) + d(\phi(i), j) \\ &\leq 2d(i, \phi(i)) + d(i, \phi^*(i)) + d(\phi^*(i), \phi^*(j)) + d(\phi^*(j), j) \end{aligned}$$

The total cost of this route, when all demand points use π_2's assignment, is defined as

$T^{(2)}$ and is obtained by adding the costs of all demands:

$$\begin{aligned}
T^{(2)} &= \sum_{\{i,j\}\in\mathcal{D}} c_{ij}^{(2)} \\
&\leq \sum_{\{i,j\}\in\mathcal{D}} 2d(i,\phi(i)) + \sum_{\{i,j\}\in\mathcal{D}} (d(i,\phi^*(i)) + d(\phi^*(i),\phi^*(j)) + d(\phi^*(j),j)) \\
&= C^* + E^* + \sum_{\{i,j\}\in\mathcal{D}} 2d(i,\phi(i)) \\
&\leq C^* + E^* + \sum_{\{i,j\}\in\mathcal{D}} (d(i,\phi(i)) + d(j,\phi(j))) \qquad (4.3) \\
&= C^* + E^* + \sum_{i\in\mathcal{C}} |\mathcal{D}_i|\ d(i,\phi(i)) \qquad (4.4) \\
&= C^* + E^* + G.
\end{aligned}$$

In step (4.3), the fact that $d(i,\phi(i)) \leq d(j,\phi(j))$ is used, and in (4.4) the indices of the summation corresponding to a client are grouped. □

Note that the discount factor α is applied in the transportation cost between hubs for path π_1, but not for π_2. Thus, in the analysis of the algorithm, the paths are chosen according to a probability parameter $p \in [0,1]$, that expresses how likely is the use of path π_1. Thus, the total expected cost is p multiplied by the cost of using π_1 plus $(1-p)$ multiplied by the cost of using π_2. On the one hand, the closer α is to zero, the cheaper π_1 becomes, for it is mostly composed of discount links; on the other hand, if α is closer to 1, π_2 becomes a better choice. Thus, parameter p is set depending on the value of α in I_A, such that the expected connection cost is minimized. Notice that the deterministic algorithm which chooses the cheapest between the available paths for each demand point generates a solution of no larger cost, compared to choosing only one strategy for all demands.

Theorem 3. *If problem A is a variant of HLP with multiple allocation, and there is a ρ-approximation algorithm for the corresponding location problem B, then there exists a $(1+\rho)$-approximation algorithm for A.*

Proof. When $\alpha = 0$, there is no cost to interconnect hubs, so there is a mapping between solution of I_A and I_B which preserve the cost, and thus solution S_B induces a ρ-approximation for I_A. So, assume $\alpha > 0$.

Consider the algorithm which uses paths π_1 with probability p and π_2 otherwise, for every demand. Let ALG be the expected cost of the selected paths, plus the cost to open

hubs. The expected cost of returned solution is:

$$\begin{aligned}
\mathbb{E}[ALG] &= pT^{(1)} + (1-p)T^{(2)} + F \\
&\leq p(\alpha C^* + \alpha E^* + (1+\alpha)G) + (1-p)(C^* + E^* + G) + F \\
&= C^*(\alpha p + 1 - p) + \alpha E^* \left(p + \frac{1-p}{\alpha}\right) + G\,(1+\alpha p) + F \\
&\leq \rho(1+\alpha p)(C^* + H^*) + C^*(\alpha p + 1 - p) + \alpha E^* \left(p + \frac{1-p}{\alpha}\right) \qquad (4.5) \\
&= C^*(1 - p + \rho + \alpha p + \alpha\rho p) + \alpha E^* \left(p + \frac{1-p}{\alpha}\right) + H^*(\rho + \alpha\rho p) \\
&\leq \gamma(C^* + \alpha E^* + H^*) \qquad (4.6) \\
&= \gamma OPT_A
\end{aligned}$$

In step (4.5), the fact that the optimal solution cost of S_A is an upper bound of the solution cost of S_B is applied, which puts the expected solution value in terms of HLP related costs. In step (4.6) the approximation factor γ is introduced, whose value is calculated below by analyzing the parameters' worst case performance.

$$\begin{aligned}
\gamma &= \max_{0<\alpha\leq 1} \left\{ \max \left\{ \begin{matrix} 1-p+\rho+\alpha p+\alpha\rho p, \\ p+(1-p)/\alpha, \\ \rho+\alpha\rho p \end{matrix} \right\} \right\} \\
&= \max_{0<\alpha\leq 1} \left\{ \max \left\{ \begin{matrix} 1-p+\rho+\alpha p+\alpha\rho p, \\ p+(1-p)/\alpha \end{matrix} \right\} \right\} \qquad (4.7) \\
&= 1 + \rho \qquad (4.8)
\end{aligned}$$

In (4.7), the third term of the maximum function was discarded for it is always smaller than the first one. In (4.8), we replaced p by the following definition:

$$p = \begin{cases} 0 & \text{if } \alpha \geq \frac{1}{1+\rho}, \\ 1 & \text{if } \alpha < \frac{1}{1+\rho}. \end{cases}$$

This completes the proof. □

4.2 Single allocation variants

When problem A is a single allocation HLP variant, each client j must be assigned to one unique hub and all demand from j must be transported through this hub. The modified reduction is described in Algorithm 2.

Since in the single allocation case every demand starting in a client i must be routed through the same hub, only path π_1 of Figure 4.1 can be used. Recall that in this path, each client is connected to the closest open hub.

Theorem 4. *If problem A is a variant of HLP with single allocation, and there is a ρ-approximation algorithm for the corresponding location problem B, then there exists a $(1+2\rho)$-approximation algorithm for A.*

Proof. The proof uses the same argument as the proof of Theorem 3, but in this case,

Algorithm 2: Reduction algorithm for single allocation HLP.

Input : I_A
Output: S_A, ϕ
$S_A \leftarrow \emptyset$
$I_B \leftarrow$ convert I_A into an instance of B
$S_B \leftarrow ALG_B(I_B)$ // obtains a ρ-approximation for I_B
foreach $i \in \mathcal{C}$ **do**
  $k^* \leftarrow \arg\min_{k \in S_B} d(i,k)$
  $\phi(i) \leftarrow k^*$ // assign hub k^* to i
  $S_A \leftarrow S_A \cup \{k^*\}$ // add hub k^* to solution
return S_A, ϕ

only path π_1 is used because clients must be assigned to only one hub, in this case, the closest one. □

4.3 Summary of reductions

From Theorems 3 and 4 we can derive various algorithms for variants of HLP, which are categorized depending if they have: single allocation (SA) or multiple allocation (MA); maximum number p of hubs to be opened or no limit at all; cost to open hubs, or no cost to open hubs (F). The results are summarized in Table 4.1, where given a variant of HLP, it shows the problem to which it is reduced, the used approximation algorithm and the obtained approximation factor.

Problem	**Reduced to**	**Factor used**	**Factor obtained**
SAHLP	FLP	1.488 [59]	3.98
MAHLP			2.488
SApHLP	k-FLP	$2+\sqrt{3}+\epsilon$ [81]	8.48
MApHLP			4.74
SAFpHLP	k-Median	$1+\sqrt{3}+\epsilon$ [60]	6.48
MAFpHLP			3.74

Table 4.1: Variants of HLP with the obtained approximation factors.

Chapter 5

Randomized Approximation Algorithm for SAHLP

In this section, a new approximation algorithm for the Single Allocation Hub Location Problem (SAHLP) is developed, in which there is no limit upon the number of open hubs and each client must be connected to exactly one of these. This algorithm builds on previous works on the Facility Location Problem, made through the last two decades. In particular, we refer to the 1.5-approximation algorithm by Byrka and Aardal [11] and the refinement of Shi Li, to achieve the 1.488-approximation [59], which are summarized in Section 3.3.

5.1 Problem Formulation

In the integer programming formulation each demand $\{i, j\} \in \mathcal{D}$ will be considered twice: a demand from i to j and a demand from j to i. Formally, let $\overrightarrow{\mathcal{D}}$ be the set of directed demands, defined as:

$$\overrightarrow{\mathcal{D}} = \{(i, j), (j, i) : \{i, j\} \in \mathcal{D}\}.$$

Given client i and hub k, let the variable $x_{ik} = 1$ indicate that client i is assigned to hub k and, otherwise, $x_{ik} = 0$. For each hub k, we consider a binary variable z_k such that $z_k = 1$ represents that hub k is opened and $z_k = 0$ indicates it is not opened. For a demand $(i, j) \in \overrightarrow{\mathcal{D}}$ and each pair of hubs $(k, l) \in \mathcal{H}^2$, let the binary variable $y_{ij}^{kl} = 1$ indicate whether this demand is routed through inbound hub k and outbound hub l, that is, k is connected to i and l is connected to j; otherwise, this variable assumes the value 0. We emphasize the order the indices assume in these variables matters, for a different route is taken if they are flipped. Note that variables x are forced to be binary, by constraints 5.2 and 5.3. Consider the following integer programming formulation of SAHLP.

$$\text{minimize} \sum_{k\in\mathcal{H}} z_k f(k) + \frac{1}{2} \sum_{(i,j)\in\overrightarrow{\mathcal{D}}} \sum_{(k,l)\in\mathcal{H}^2} y_{ij}^{kl} \left(d(i,k) + \alpha d(k,l) + d(l,j)\right)$$

$$\text{subject to} \sum_{k\in\mathcal{H}} x_{ik} = 1 \qquad \forall\, i \in \mathcal{C}, \qquad (5.1)$$

$$\sum_{l\in\mathcal{H}} y_{ij}^{kl} = x_{ik} \qquad \forall\, (i,j) \in \overrightarrow{\mathcal{D}}, \quad \forall\, k \in \mathcal{H}, \qquad (5.2)$$

$$\sum_{k\in\mathcal{H}} y_{ij}^{kl} = x_{jl} \qquad \forall\, (i,j) \in \overrightarrow{\mathcal{D}}, \quad \forall\, l \in \mathcal{H}, \qquad (5.3)$$

$$x_{ik} \leq z_k \qquad \forall\, i \in \mathcal{C}, \quad \forall\, k \in \mathcal{H}, \qquad (5.4)$$

$$y_{ij}^{kl} \in \{0,1\} \qquad \forall\, (i,j) \in \overrightarrow{\mathcal{D}}, \quad \forall\, (k,l) \in \mathcal{H}^2, \qquad (5.5)$$

$$z_k \in \{0,1\} \qquad k \in \mathcal{H}. \qquad (5.6)$$

Let (x, y, z) be a solution of the program. The set of variables z produces a set of open hubs O and x an assignment $\phi : \mathcal{C} \to O$. Regarding the objective function, the first term accounts for the cost of open hubs in O and the second term for the connection cost of each demand $(i, j) \in \overrightarrow{\mathcal{D}}$. Notice that $\overrightarrow{\mathcal{D}}$ counts each demand of $\mathcal{D}$ twice, and thus we have a factor $\frac{1}{2}$ that multiplies the sum.

By the first set of constraints (5.1), each client i is assigned to exactly one hub, say k, and constraint (5.4) ensures that $k \in O$, the set of open hubs. Constraints (5.2) and (5.3) make sure that there is a pair of hubs transporting the demand of a pair of clients, taking the route cost into account in the second term of the objective function. However, one could say that a demand $(i, j) \in \overrightarrow{\mathcal{D}}$ with corresponding variable y_{ij}^{kl} is not set to 1 and not correctly contributing to the final value; instead, a y variable associated with another hub is wrongly set to 1, so now we argue that for this demand, $y_{ij}^{kl} = 1$ if, and only if i is assigned to k and j is assigned to l, i.e., $x_{ik} = 1$ and $x_{jl} = 1$. Suppose that i is assigned to k and let $k' \neq k$, then by the second set of constraints $y_{ij}^{k'l'} = x_{ik'} = 0$ for every l', since i can be assigned to only one hub. In a similar way, if j is assigned to l, then $y_{ij}^{k'l'} = x_{jl'} = 0$ for every k' and $l' \neq l$. Given that only $y_{ij}^{kl'}$, for any l', and $y_{ij}^{k'l}$, for any k', can be different from 0 and the constraint sets of these variables must sum to 1, it implies that we must have $y_{ij}^{kl} = 1$.

For the development of the algorithm, we will require a relaxation of the program, in which we replace constraints $y_{ij}^{kl} \in \{0,1\}$ and $z_k \in \{0,1\}$ by $y_{ij}^{kl} \geq 0$ and $z_k \geq 0$. Denote the resulting linear program by (LP).

5.2 LP bounds and symmetries

Let (x, y, z) be a fixed optimal solution for (LP). In the following, we define a series of variables which represent parts of the cost of this given solution. The goal is the rewrite the objective function in simpler terms.

First, for each demand (i, j) we consider the corresponding (fractional) connection cost in the objective function of (LP). This cost can be broken into three parts: the average cost to go from i to the fractionally assigned inbound hubs k; the average cost to go from the fractionally assigned inbound hubs k to the fractionally assigned outbound hubs l; and the average cost to go from the fractionally assigned outbound hubs l to j. These costs are illustrated in Figure 5.1. When considering the costs associated with demands, we will use lower case letters for the name of the variables.

Definition 11. Let (x, y, z) be an optimal solution of (LP). Consider a demand $(i, j) \in \overrightarrow{\mathcal{D}}$. We define:

$$c_{ij} = \sum_{(k,l)\in\mathcal{H}^2} y_{ij}^{kl} d(i,k), \quad e_{ij} = \sum_{(k,l)\in\mathcal{H}^2} y_{ij}^{kl} \alpha d(k,l), \quad q_{ij} = \sum_{(k,l)\in\mathcal{H}^2} y_{ij}^{kl} d(l,j).$$

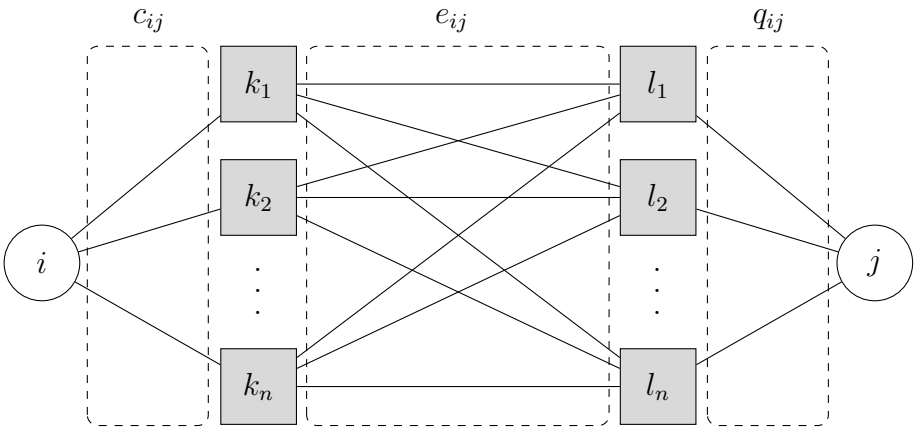


Figure 5.1: Illustration of which link costs are related to c_{ij}, e_{ij} and q_{ij}.

Consider a client $i \in \mathcal{C}$ and recall that $\mathcal{D}_i = \{j : \{i, j\} \in \mathcal{D}\}$. The set $\mathcal{D}_i$ is the set of endpoints of demands associated with a client i (excluding i). Since we duplicated the demands, each demand (i, j) of $\overrightarrow{\mathcal{D}}$ can be associated with a single client $i \in \mathcal{C}$. Now we associate part of the objective cost corresponding to a given client. When considering the costs associated with clients, capital letters will be used for the name of the variables.

Definition 12. Let (x, y, z) be an optimal solution of (LP). Consider a client $i \in \mathcal{C}$. We define:

$$C_i = \sum_{j\in\mathcal{D}_i} c_{ij}, \quad E_i = \sum_{j\in\mathcal{D}_i} e_{ij}, \quad Q_i = \sum_{j\in\mathcal{D}_i} q_{ij}.$$

Finally, we break the cost of the objective function in four different parts, namely: the opening cost, the inbound connection cost, the inter-hub connection cost and the outbound connection cost.

Definition 13. Let (x, y, z) be an optimal solution of (LP). We define:

$$H = \sum_{k\in\mathcal{H}} z_k f(k), \quad C = \sum_{i\in\mathcal{C}} C_i, \quad E = \sum_{i\in\mathcal{C}} E_i, \quad Q = \sum_{i\in\mathcal{C}} Q_i.$$

Observe that the objective function can now be written as $H + \frac{1}{2}(C + E + Q)$. When analyzing the approximation factor of the algorithm these terms' values will be bounded individually, which will make up the final achieved ratio. It will also be useful to define versions of the variables whose sums are restricted to one associated hub k.

Definition 14. Let (x, y, z) be an optimal solution of (LP). Consider a client $i \in \mathcal{C}$, and a fixed hub $k \in \mathcal{H}$. We define:

$$c_{ijk} = \frac{1}{x_{ik}} \sum_{l \in \mathcal{H}} y_{ij}^{kl} d(i,k), \qquad e_{ijk} = \frac{1}{x_{ik}} \sum_{l \in \mathcal{H}} y_{ij}^{kl}\, \alpha d(k,l), \qquad q_{ijk} = \frac{1}{x_{ik}} \sum_{l \in \mathcal{H}} y_{ij}^{kl} d(l,j),$$
$$C_{ik} = \sum_{j \in \mathcal{D}_i} c_{ijk}, \qquad E_{ik} = \sum_{j \in \mathcal{D}_i} e_{ijk}, \qquad Q_{ik} = \sum_{j \in \mathcal{D}_i} q_{ijk}.$$

Counting each demand twice is useful, as it provides many symmetries that will be exploited in the analysis. One such symmetry is given by the next auxiliary lemma, which follows directly from the set of constraints.

Lemma 6. *Let (x, y, z) be a feasible solution of (LP). Then for every demand $(i, j) \in \overrightarrow{\mathcal{D}}$ and hub $k \in \mathcal{H}$, the following holds:*

$$\sum_{l \in \mathcal{H}} y_{ij}^{kl} = \sum_{l \in \mathcal{H}} y_{ji}^{lk}$$

Proof. From constraint (5.2), we have $\sum_{l \in \mathcal{H}} y_{ij}^{kl} = x_{ik}$, and from constraint (5.3) we have $\sum_{l \in \mathcal{H}} y_{ji}^{lk} = x_{ik}$. The lemma follows. □

Using this result, it is possible to relate the inbound and outbound connection costs, showing they are equal, as in the following corollary.

Corollary 2. *For every feasible solution (x, y, z) of (LP) and demand (i, j), $c_{ij} = q_{ji}$*

Proof. By direct calculation and using Lemma 6,

$$\begin{aligned} c_{ij} &= \sum_{(k,l) \in \mathcal{H}^2} y_{ij}^{kl} d(i,k) \\ &= \sum_{k \in \mathcal{H}} \left(d(i,k) \sum_{l \in \mathcal{H}} y_{ij}^{kl} \right) \\ &= \sum_{k \in \mathcal{H}} \left(d(i,k) \sum_{l \in \mathcal{H}} y_{ji}^{lk} \right) \\ &= \sum_{(l,k) \in \mathcal{H}^2} y_{ji}^{lk} d(i,k) \\ &= q_{ji}. \end{aligned}$$

□

We make use of other properties of a solution to the (LP). For a fixed client i and hub k, every constraint of type (5.2) sum up to x_{ik}, implying that the constraint of demand points (i, j) and (i, j'), for any $j, j' \in \mathcal{D}_i$, are equal, thus, $\sum_{l \in \mathcal{H}} y_{ij}^{kl} = \sum_{l \in \mathcal{H}} y_{ij'}^{kl}$. Previously, we defined for a demand (i, j) the average cost to go from i to the fractionally assigned

inbound hubs k. Below, we define the unique value a client i uses to connect to every endpoint.

Definition 15. Let (x, y, z) be an optimal solution of (LP), i be a client and $j \in \mathcal{D}_i$. We define c_i as the unique cost that this client uses to connect to the fractionally assigned inbound hubs:

$$c_i = c_{ij},$$

Note the c_i does not depend on which endpoint is being used in the definition, and thus c_i is well defined. Indeed, let $j, j' \in \mathcal{D}_i$. Then

$$c_i = c_{ij} = \sum_{k\in\mathcal{H}} \sum_{l\in\mathcal{H}} y_{ij}^{kl} d(i,k) = \sum_{k\in\mathcal{H}} \sum_{l\in\mathcal{H}} y_{ij'}^{kl} d(i,k) = c_{ij'}.$$

We also observe that the overall inbound and outbound connection costs are equal.

Lemma 7. *For any feasible solution (x, y, z), $C = Q$.*

Proof. From Corollary 2, we calculate

$$C = \sum_{i\in\mathcal{C}} C_i = \sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{D}_i} c_{ij} = \sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{D}_i} q_{ji} = \sum_{j\in\mathcal{C}} \sum_{i\in\mathcal{D}_j} q_{ji} = \sum_{j\in\mathcal{C}} Q_j = Q. \qquad \square$$

5.3 Modifying the fractional solution

A commonly used technique for FLP approximation algorithms is the notion of clustering and the neighborhood of a client, that allow us to bound connection costs and connect clients to open hubs with values we know how to calculate. Inspired by those ideas, we consider the support of the solution, i.e., the set of nonzero variables. One way to interpret the support is to think of the bipartite graph between $\mathcal{C}$ and $\mathcal{H}$, where there is an edge between a client i and a hub k if $x_{ik} > 0$.

Definition 16. Let (x, y, z) be an optimal solution of (LP). For a client i, we define the *neighborhood* of this client as the set of hubs to which it is connected:

$$N(i) = \{k \in \mathcal{H} : x_{ik} > 0\}.$$

Before explaining the clustering algorithm, we must introduce several concepts that involves how to connect clients to hubs.

The *filtering technique* was introduced by Lin and Vitter [63], that considered problems with packing and covering constraints. After solving the LP relaxation, some variables that have large objective coefficients are modified and then a rounding procedure is applied. Shmoys et al. [72] used the technique but also scaled some variables in the process. We will use the filtering as seen in the FLP literature [76, 11] to bound the distance between a client i and a hub k of $N(i)$, instead of using the dual variable that corresponds to the set of constraints (5.1), as in primal-dual based algorithms.

Let $\gamma > 1$ be a scaling parameter for our algorithm. The first modification in the solution we will execute is scale up the hub opening variables: for all $k \in \mathcal{H}$, let $z'_k = \gamma z_k$. Note that several of these variables related to hubs can now be greater than one. As discussed by Byrka and Aardal [11], we do not round these variable down to 1 (as done by Shmoys et al. [72]), but rather split hubs that exceeded the value to maintain feasibility. This technique is describe more precisely later in this section.

We highlight that variables z_k always refer to the non-scaled solution, and are not replaced by variables z'_k.

Definition 17. For a client i, consider a permutation $k_1, k_2, \ldots, k_{|N(i)|}$ of the neighborhood $N(i)$ in nondecreasing order of $d(i, k_s)$. Let t be the minimum index such that the sum surpasses the given value: $\sum_{s=1}^{t} z_{k_s} \geq 1/\gamma$. The γ-*neighborhood* of client i is defined as:

$$N_\gamma(i) = \{k_1, k_2, \ldots, k_t\}.$$

Filtering is used in many location problems whose aim is connecting clients to some kind of structure. The idea is that for $\gamma > 1$ only the hubs that are closest to i are considered and thus the distance between i and a hub in $N_\gamma(i)$ can be bounded in terms of the primal objective function. In FLP, this technique has been used to bound the connection cost of a client to its closest open facility, which produces a good solution, for the client-facility link is the only cost that needs to be analyzed. In the SAHLP, however, filtering cannot be used this way. The reason is that the total connection cost of a demand is the sum of different terms, and not only the distance of a client to the assigned hub and, thus, the cost of serving the demands starting in i through the closest hub of i could be larger than serving the same demands though a different hub. In our problem, in the general case, we might use a more expensive connection to the inbound hub, but save in other links to transport the demands. To tackle this difficulty, in our algorithm, filtering will only be used to bound the distance from i to the assigned hub, the overall demand connection cost will be calculated using different techniques.

Next, we describe more precisely the splitting process, and define the complete solution.

5.3.1 Splitting hubs

For a given client i, we defined its γ-neighborhood as containing almost the same hubs as $N(i)$, until the sum of their z_k variables exceeds $1/\gamma$, if the opening of each hub is summed by nondecreasing order of the distance to i. We assume that, for each client, the volume of opening in $N_\gamma(i)$ equals to γ, that is,

$$\sum_{k \in N_\gamma(i)} z_k = 1/\gamma.$$

If this is not the case, it is possible to create an equivalent instance of the problem and a corresponding solution, whose value is the same as the value of (x, y, z). This is done by successively *splitting* hubs. The idea is to find the last hub in the sum related to $N_\gamma(i)$ and systematically modify the variables to perfectly achieve the volume of γ.

Let $i \in \mathcal{C}$ and t be as in the definition of $N_\gamma(i)$, that is, let t such that k_t is the first hub for which the sum of fractional opening exceeds $1/\gamma$. Define $\epsilon = \sum_{s=1}^{t} z_{k_s} - 1/\gamma$, the exceeding amount. If $\epsilon = 0$, then we say that $N_\gamma(i)$ is tight; otherwise, $\epsilon > 0$, and we say that it is loose. In the latter case, we split k_t at value $z_{k_t} - \epsilon$: replace k_t by two new hubs, say k' and k'', in the same location. Now, define $z_{k'} = z_{k_t} - \epsilon$ and $z_{k''} = \epsilon$. Since we remove a hub that is connected to multiple clients and replaced it with two others, we must fix the connection variables for every client: for each $i \in \mathcal{C}$, let $x_{ik'} = \min\{z_{k'}, x_{ik_t}\}$ and $x_{ik''} = \max\{0, x_{ik_t} - z_{k'}\}$. Note that splitting takes polynomial time, since each time that we split a hub, the neighborhood of clients considered in previous iterations remain tight. Moreover, the number of created hubs is polynomial and the objective value does not change. Thus, in the following, assume that $\sum_{k \in N_\gamma(i)} z_k = 1/\gamma$ for every $i \in \mathcal{C}$.

5.3.2 Complete solution

There is also the notion of *complete* solution: a solution (x, y, z) of (LP) is complete if, for every $(i, j) \in \overrightarrow{\mathcal{D}}$ and every $(k, l) \in \mathcal{H}^2$, if $y_{ij}^{kl} > 0$, then $y_{ij}^{kl} = z_k = z_l$. Once again, we may assume that (x, y, z) is complete, since otherwise we can create an equivalent instance of the problem together with a solution of (LP) of no larger cost. Suppose that a solution is not complete. Then, there is a variable y_{ij}^{kl} that is greater than zero and at least one of the variables z_k or z_l are not equal to y_{ij}^{kl}: w.l.o.g., assume $z_k > y_{ij}^{kl}$. The procedure to fix these variables is the same as before, splitting hub k into hubs k' and k''.

Let $\epsilon = z_k - y_{ij}^{kl}$, the value at which k must be split. Create k' and k'' in the same location as k, define the new variables and fix the value of the variables x that were already split in the previous step: $z_{k'} = x_{ik'} = y_{ij}^{k'l}$ and $z_{k''} = x_{ik''} = \epsilon$. As long as there is a variable y_{ij}^{kl} that is not in the needed standard, we split z_k or z_l in a similar way as described above. Notice that the number of these variables always decreases, thus this process takes polynomial time. From now on, suppose that (x, y, z) is a complete solution.

In this modified solution, after splitting hubs and transforming it into a complete solution, we partition the support $N(i)$ of a client i into two sets of hubs: the *close hubs* $Cl(i) = N_\gamma(i)$ and the *distant hubs* $Di(i) = N(i) \setminus Cl(i)$. In order to calculate the cost a demand expends, we will make use of the average distances from a client to particular sets of hubs it is connected to.

Definition 18. Given a client $i \in \mathcal{C}$, define c_i^c as the average distance from i to the close hubs in $Cl(i)$ and c_i^d as the average distance from i to the distant hubs in $Di(i)$:

$$c_i^c = \frac{\sum_{k \in Cl(i)} z_k d(i,k)}{\sum_{k \in Cl(i)} z_k} = \gamma \sum_{k \in Cl(i)} z_k d(i,k),$$

$$c_i^d = \frac{\sum_{k \in Di(i)} z_k d(i,k)}{\sum_{k \in Di(i)} z_k} = \frac{\gamma}{\gamma - 1} \sum_{k \in Di(i)} z_k d(i,k).$$

From the average distances of a client to these sets of hubs, we introduce helpful distance

measures of how the hubs of an instance are related to the clients:

$$Q^c = C^c = \sum_{i \in \mathcal{C}} c_i^c, \qquad Q^d = C^d = \sum_{i \in \mathcal{C}} c_i^d.$$

Observe that c_i can be defined as the weighted arithmetic mean using the terms of the last definition and thus $c_i = \frac{1}{\gamma} c_i^c + (1 - \frac{1}{\gamma}) c_i^d = \sum_{k \in N(i)} z_k d(i,k)$, which agrees with the previously defined versions. Also we define the *irregularity* of the distances, which is defined as

$$\rho = \frac{C - C^c}{C} = \frac{Q - Q^c}{Q}.$$

Note that $0 \leq \rho \leq 1$, since $0 \leq C^c \leq C$.

5.4 LP-rounding algorithm

In this section we describe our randomized algorithm. Given a complete solution of (LP), we need to perform two steps: (i) locate which hubs will be selected and (ii) create an allocation of clients to hubs. Notice that we must explicitly return an assignment $\phi : \mathcal{C} \to O$. Each step is detailed next.

5.4.1 Clustering

After solving (LP), the obtained fractional solution is modified by scaling the opening variables, splitting hubs and turning it into a complete solution with the same cost of the original solution. To select which hubs will be opened, we will use a *clustering algorithm*, such that each cluster is associated to a special client, called the *cluster center*, for which a hub will be opened. The clustering has the property that not two cluster centers are neighbors, which are defined as follows:

Definition 19. Given clients $i, j \in \mathcal{C}$, we say that i and j are *neighbors* if $Cl(i) \cap Cl(j) \neq \emptyset$.

Now, we create a partition of clients by using the following greedy strategy: as long as there are clients that are not in a cluster, select an unclustered client i whose sum $c_i^c + c_i^d$ is minimum, create a new cluster whose center is i and that contains all neighbors of i. By construction, note that for the chosen centers i, the sets $Cl(i)$ are disjoint and that $\sum_{k \in Cl(i)} \gamma z_k = 1$. Thus, the variables in this set represent a probability distribution on $Cl(i)$. For each center i, we dependently open exactly one hub k in $Cl(i)$, such that k is opened with probability γz_k. For each hub k that is not a close hub of some center, we open k independently with probability γz_k. Note that these strategies differ in the sense that the former chooses one hub from $Cl(i)$ to be opened and the latter tries to open each of the others with that independent probability, therefore, there will always be open hubs. Let O be the set of all open hubs.

After executing this procedure, a client i will either have an open hub in its set of close hubs or an open hub that is not too far away, since it is close to the cluster center of i. This situation is depicted in Figure 5.2.

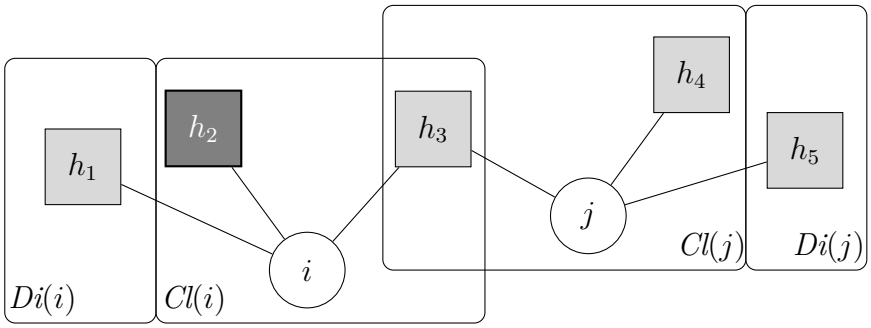


Figure 5.2: Cluster with open hub h_2 and center i, that shares hub h_3 with client j.

5.4.2 Assignment

Now that the set of open hubs is defined, we must explicitly return an assignment $\phi : \mathcal{C} \to O$. Notice that, contrary to what happens in FLP, a client is not necessarily assigned to the closest open hub. Note that the problem of assigning a hub to a client is NP-hard, even if the set of open hubs were given [74], then one must be careful on which hubs are allocated to clients. To allocate a hub to a client i, our algorithm will consider two possibilities, depending on whether there is a hub open in $N(i)$ or not.

We process each client i independently. Fixing a client i, there are two possibilities. First, suppose that one or more hubs of $N(i)$ were opened. In this case, we will connect i to one hub $N(i) \cap O$. If we connected i to the closest hub, then the overall demand cost could be potentially very large, therefore we balance between the inbound connection cost, the inter-hub connection cost and the outbound connection cost. More precisely, we assign i to the hub $k \in N(i) \cap O$ which minimizes

$$V_{ik} := 3C_{ik} + E_{ik} + Q_{ik}.$$

In this case, we set $\phi(i) = k$. Otherwise, suppose no hub in $N(i)$ was opened. In this case, we simply assign i to the closest open hub, i.e., let $k \in O$ be the hub which minimizes $d(i, k)$, then we se $\phi(i) = k$. The rationale for choosing the value for V_{ik} is that it appears in the analysis as the expected cost a client pays to connect to an open hub in its neighboring set.

The whole process is presented in Algorithm 3.

5.5 Analysis of the LP-rounding algorithm

In this section, we analyze LP-rounding algorithm and show that the expected cost of the generated solution is at most 2.48 times the value of the objective function of (LP), and therefore our algorithm is a randomized 2.48-approximation, since the objective value is a lower bound on the optimal value.

First, we bound the expected cost of opening hubs.

Lemma 8. *If O is the set of hubs opened by the algorithm, then the expected cost to open hubs is $\mathbb{E}[O] = \gamma H$.*

Algorithm 3: Rounding algorithm for SAHLP.

Input : (x, y, z)
Output: O, ϕ

$S \leftarrow \mathcal{C}$
$C \leftarrow \emptyset$ // The set of cluster centers
$O \leftarrow \emptyset$ // The set of open hubs
// Cluter the clients
while $S \neq \emptyset$ **do**
    Let $i \in S$ be the client with minimum $c_i^c + c_i^d$.
    $C \leftarrow C \cup \{i\}$.
    Create a cluster $Q = \{i\} \cup \{j \in S : Cl(i) \cap Cl(j) \neq \emptyset\}$ centered at i.
    $S \leftarrow S - Q$.
// Open a hub for each cluster
foreach $i \in C$ **do**
    Choose $k \in Cl(i)$ with probability γz_k.
    $O \leftarrow O \cup \{k\}$.
// Open remaining hubs
foreach $k \in \mathcal{H}$ *such that* $k \notin Cl(i)$ *for every* $i \in C$ **do**
    Open hub k with probability γz_k.
// Connect clients to hubs
foreach $i \in \mathcal{C}$ **do**
    if $N(i) \cap O \neq \emptyset$ **then**
        Let $k \in N(i) \cap O$ be the hub that minimizes V_{ik}.
    else
        Let $k \in O$ be the hub that minimizes $d(i, k)$.
    $\phi(i) = k$.
return O, ϕ

Proof. In the algorithm, the opening conditions for a hub $k \in \mathcal{H}$ occur in two cases: if $k \in Cl(i)$ for some cluster center i, then k is selected with probability γz_k; otherwise, k is not in any cluster and thus was opened with probability γz_k. Therefore, the expected cost to open hubs is:

$$\mathbb{E}\left[\sum_{k\in O} f(k)\right] = \sum_{k\in O} \Pr(k \in O) f(k) = \sum_{k\in O} \gamma z_k f(k) = \gamma H. \qquad \square$$

Consider now the connection cost: recall that for a fixed demand $(i,j) \in \overrightarrow{\mathcal{D}}$, if i is assigned to hub k and j is assigned to hub l, then the cost to serve (i,j) is $d(i,k) + \alpha d(k,l) + d(l,j)$, with the discount factor applied in the hub-hub link. The main intuition of the analysis is using the fact that $\alpha \leq 1$ and bounding the connection cost of a demand (i,j) as follows:

$$\begin{aligned} cost(i,j) &= d(i,\phi(i)) + \alpha d(\phi(i),\phi(j)) + d(\phi(j),j) \\ &\leq d(i,\phi(i)) + \alpha d(\phi(i),j) + \alpha d(j,\phi(j)) + d(\phi(j),j). \end{aligned}$$

Observe that we do not know how to calculate the distance between hubs $\phi(i)$ and $\phi(j)$, so we bound this link using only connections of clients to hubs, pertinently multiplying the discount factor to one of the links. This situation is demonstrated in Figure 5.3, where the left scheme is the definition of such cost and the right scheme is the value we bound it to. This connection cost can be split in two parts: (I) $d(i,\phi(i)) + \alpha d(\phi(i),j)$, that accounts for the cost of connecting i to its designated hub plus the distance of this hub to j with the discount factor; and (II) $\alpha d(j,\phi(j)) + d(\phi(j),j)$, that accounts for the cost of going from j to the designated hub of j and back, using the discount factor in one of the links.

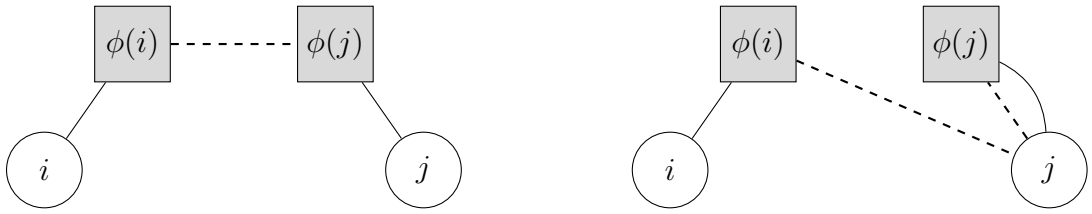


Figure 5.3: Bounding the connection cost of demand (i,j).

In the algorithm, a client i can be connected to an open hub in its neighboring set $N(i)$, choosing the hub that minimizes a special sum of some components, or connect to the closest one, in the case that there are not any open hubs in $N(i)$. So, because the assignment of each client considers two cases, there are four possibilities to analyze. If a client i is assigned to some hub in its support set $N(i)$, then we will bound part (I) very economically, meaning that we do not need to bound this cost; otherwise we will consider the path from i to $\phi(i)$ and back to i, and then bound the distance from i to j. For part (II), we will always bound the cost of going from j to $\phi(j)$ and returning.

These bounds cover well the possibility that neither $N(i)$ nor $N(j)$ has an open hub, or that there is some open hub in $N(i)$ and no open hub in $N(j)$. The possibility regarding

the case that there is one open hub in $N(j)$ and no open hub in $N(i)$ is also balanced in our strategy, as we consider each demand $\{i, j\} \in \mathcal{D}$ twice, one for (i, j) and other for (j, i), in the summation of total connection cost. We remark that our strategy will give only an upper bound on the cost of the solution, since it does not consider the possibility that both $N(i)$ and $N(j)$ have open hubs k and l for some $y_{ij}^{kl} > 0$.

Definition 20. Consider a client $i \in \mathcal{C}$ and let O be the set of open hubs. Denote by a_i the event that there is an open hub in the neighboring set hubs of i, i.e., $N(i) \cap O \neq \emptyset$, and by $\bar{a}_i$ the complement of a_i. Also, let p_i be the probability of event a_i.

The next lemma bounds the probability of event a_i occurring for a client $i \in \mathcal{C}$, which we will use to analyze the four possibilities aforestated.

Lemma 9. *For a given client $i \in \mathcal{C}$, the probability that there is an open hub in its support set, i.e., event a_i occurring, is: $p_i \geq 1 - 1/e^{\gamma}$.*

Proof. For a fixed client $i \in \mathcal{C}$, we consider a sequence of sets associated with the support of i and the close hubs of cluster centers: let $A_s = Cl(i_s) \cap N(i)$, for all cluster centers i_s. For any pair of hubs k, l in $N(i)$, there can be several cases: k is not close to any center, k is close to a center or k and l are both close to a center. In the first case, hub k is opened independently with probability $\bar{z}_k$; in the second case, since it is the only hub of $N(i)$ in this close set, the same probability applies. In the third case, when there are two or more hubs of $N(i)$ in the same close set, their probability to be opened is negatively correlated, meaning that, the fact that one is not chosen to be opened increases the chance of opening the others, since exactly one hub of this set is chosen. However, we consider the possibly suboptimal analysis of independent hub opening in this third case.

Now, since all hubs of $N(i)$ are opened independently and knowing that the probability of hub k not being opened is $1 - \bar{z}_k$, the probability p that $N(i)$ do not have an open hub is:

$$p = \prod_{k \in N(i)} (1 - \bar{z}_k) \leq \prod_{k \in N(i)} e^{-\bar{z}_k} = e^{-\sum_{k \in N(i)} \bar{z}_k} = e^{-\sum_{k \in N(i)} \gamma z_k} = e^{-\gamma},$$

where we used the fact that $1 + x \leq e^x$, for $x \in \mathbb{R}$, and that the sum of z variables over the neighboring hubs of i is equal to 1. Therefore, there is an open hub in $N(i)$ with probability at least $1 - e^{-\gamma}$. □

Next lemma bounds the average cost of the closest hub of a client i in the case that no hub in $N(i)$ was opened. Recall that in this case, there is an open hub that is near to its cluster center. The proof of the next two lemmas are featured in Section 3.3, where we reviewed an important FLP algorithm that uses similar techniques as ours. As said before, these results are due to Byrka and Aardal, as well as Shi Li.

Lemma 10. *Let $i \in \mathcal{C}$, i' be the corresponding cluster center and $A = Cl(i') \setminus N(i)$. If $\tilde{c}_i^c$ represents the maximum distance from client i to one of its close hubs, then, for any $\gamma \geq 1$:*

$$\frac{\sum_{k \in A} z_k d(i, k)}{\sum_{k \in A} z_k} \leq (2 - \gamma)\tilde{c}_i^c + (\gamma - 1)c_i^d + \tilde{c}_{i'}^c + c_{i'}^c.$$

Lemma 11. *Let i be a client, O be the set of opened hubs by the clustering algorithm, and $A \subseteq \mathcal{H}$ be a set of hubs such that $\sum_{k\in A} z_k > 0$. Also, let $h_k \geq 0$ be a real value associated with a hub k. Then*

$$\mathbb{E}\left[\min_{k\in A\cap O} h_k \middle| A \cap O \neq \emptyset\right] \leq \frac{\sum_{k\in A} z_k h_k}{\sum_{k\in A} z_k}.$$

Lemma 12. *Let i be a client, then $\mathbb{E}[V_{i\phi(i)} \,|\, a_i] \leq 3C_i + E_i + Q_i$.*

Proof. This follows directly from Lemma 11 by defining $h_k = V_{ik}$, for each k. □

Using Lemmas 10 and 11, we obtain the following.

Lemma 13. *Let i be a client. Then, $\mathbb{E}[d(i,\phi(i)) \,|\, \bar{a}_i] \leq 2c_i^d + c_i^c$.*

Proof. Notice that conditioned on the event that i is not assigned to a hub in $N(i)$, i is assigned to the closest open hub. In particular, in this case $d(i,\phi(i))$ is smaller than the distance from i to the closest open hub in $A = Cl(i') \setminus N(i)$, where i' is the cluster center corresponding to i. Observe that $A \cap O \neq \emptyset$ since we opened a hub in $Cl(i')$, and we suppose no hub in $N(i) \cap A$ was opened. Letting $h_k = d(i,k)$ for each k and using Lemma 11, we obtain that

$$\begin{aligned}
\mathbb{E}[d(i,\phi(i)) \,|\, \bar{a}_i] &\leq \frac{\sum_{k\in A} z_k h_k}{\sum_{k\in A} z_k} \\
&\leq (2-\gamma)\tilde{c}_i^c + (\gamma-1)c_i^d + \tilde{c}_{i'}^c + c_{i'}^c \\
&= (3-\gamma)\tilde{c}_i^c + (\gamma-1)c_i^d + c_i^c \\
&\leq 2c_i^d + c_i^c
\end{aligned}$$

In the second inequality, Lemma 10 was applied; we used the fact that $\tilde{c}_i^c \leq c_i^d$, for any $i \in \mathcal{C}$ and in the last inequality the following bound was used $c_{i'}^c + c_{i'}^d \leq c_i^c + c_i^d$ since i' is the cluster center associated with i. □

In the next lemma, we bound the cost a given demand (i,j) considering all cases of events a_i and a_j occurring or not.

Lemma 14. *Let $(i,j) \in \overrightarrow{\mathcal{D}}$ and $cost(i,j) = d(i,\phi(i)) + \alpha d(\phi(i),\phi(j)) + d(\phi(j),j)$. The expected cost of (i,j) is:*

$$\begin{aligned}
\mathbb{E}[cost(i,j)] = {} & p_i \mathbb{E}[d(i,\phi(i)) + e_{ij\phi(i)} + q_{ij\phi(i)} \,|\, a_i] \\
& + (1-p_i)\left(2(2c_i^d + c_i^c) + c_i + e_{ij} + c_j\right) \\
& + p_j \mathbb{E}[2d(j,\phi(j)) \,|\, a_j] \\
& + (1-p_j)2(2c_j^d + c_j^c).
\end{aligned}$$

Proof. Let $k = \phi(i)$. First, we calculate a series of bounds:

$$\begin{aligned}\alpha d(\phi(i), j) &= \alpha d(k, j) \\ &\leq \min_{l \in \mathcal{H}} \{\alpha(d(k, l) + d(l, j))\} \\ &\leq \min_{l \in \mathcal{H}} \{\alpha d(k, l) + d(l, j)\} \\ &\leq \frac{1}{x_{ik}} \sum_{l \in \mathcal{H}} y_{ij}^{kl} (\alpha d(k, l) + d(l, j)) \\ &= e_{ij\phi(i)} + q_{ij\phi(i)},\end{aligned}$$

where in the last inequality we used $\sum_{l \in \mathcal{H}} y_{ij}^{kl} = x_{ik}$, and the fact that the minimum of $\alpha d(k, l) + d(l, j)$, for some l, is not larger than its average. By similar arguments, we also have:

$$\alpha d(i, j) \leq \alpha(c_i + e_{ij} + c_j).$$

Lemma 13 states that:

$$\begin{aligned}\mathbb{E}[d(i, \phi(i)) \,|\, \bar{a}_i] &\leq 2c_i^d + c_i^c \quad \text{and} \\ \mathbb{E}[d(j, \phi(j)) \,|\, \bar{a}_j] &\leq 2c_j^d + c_j^c.\end{aligned}$$

Recall that we want to bound $d(i, \phi(i)) + \alpha d(\phi(i), \phi(j)) + d(\phi(j), j)$ as the sum of two parts: (I) $d(i, \phi(i)) + \alpha d(\phi(i), j)$ and (II) $\alpha d(j, \phi(j)) + d(\phi(j), j)$:

- Bounding (I): conditioned on the event a_i, i.e., that $\phi(i) \in N(i)$, the cost of the first part is $d(i, \phi(i)) + \alpha d(\phi(i), j) \leq d(i, \phi(i)) + e_{ij\phi(i)} + q_{ij\phi(i)}$. Otherwise, since $\alpha \leq 1$, we have:

$$\begin{aligned}d(i, \phi(i)) + \alpha d(\phi(i), j) &\leq 2d(i, \phi(i)) + \alpha d(i, j) \\ &\leq 2(2c_i^d + c_i^c) + c_i + e_{ij} + c_j\end{aligned}$$

- Bounding (II): conditioned on a_j, we simply use $d(j, \phi(j))$, otherwise, conditioned on $\bar{a}_j$, the value of the part is at most:

$$\mathbb{E}[2d(j, \phi(j)) \,|\, \bar{a}_j] \leq 2(2c_j^d + c_j^c)$$

Combining all the conditional expectations, we complete the lemma. □

Now we are ready to bound the overall connection cost, summing up the costs of all demands.

Lemma 15. *Suppose $\gamma \leq 3$. The expected connection cost to serve demands in $\overrightarrow{\mathcal{D}}$ is:*

$$\mathbb{E}\left[\sum_{(i,j) \in \overrightarrow{\mathcal{D}}} cost(i, j)\right] \leq \left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) C + E + \left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) Q$$

Proof. Let T be the total connection cost. Using Lemma 14, we get that the total expected cost is:

$$\begin{aligned}\mathbb{E}[T] \leq &\sum_{i\in\mathcal{C}}\sum_{j\in D_i}\Big(p_i\mathbb{E}[d(i,\phi(i)) + e_{ij\phi(i)} + q_{ij\phi(i)} \mid a_i] \\ &\qquad +(1-p_i)\left(2(2c_i^d + c_i^c) + c_i + e_{ij} + c_j\right)\Big) \\ &+\sum_{j\in\mathcal{C}}\sum_{i\in D_j}\left(p_j\mathbb{E}\left[2d(j,\phi(j)) \mid a_j\right] + (1-p_j)2(2c_j^d + c_j^c)\right)\end{aligned}$$

Notice that indices in the sums in both lines range the set of demands $\overrightarrow{\mathcal{D}}$. If we rename the indices of the sum in the last line, we obtain:

$$\begin{aligned}\mathbb{E}[T] \leq &\sum_{i\in\mathcal{C}}\sum_{j\in D_i}\Big(p_i\mathbb{E}[d(i,\phi(i)) + e_{ij\phi(i)} + q_{ij\phi(i)} \mid a_i] \\ &\qquad +(1-p_i)\left(2(2c_i^d + c_i^c) + c_i + e_{ij} + c_j\right)\Big) \\ &+\sum_{i\in\mathcal{C}}\sum_{j\in D_i}\left(p_i\mathbb{E}\left[2d(i,\phi(i)) \mid a_i\right] + (1-p_i)2(2c_i^d + c_i^c)\right)\end{aligned}$$

Now we join the lines and distribute the inner sum:

$$\begin{aligned}\mathbb{E}[T] \leq \sum_{i\in\mathcal{C}}\Bigg(&p_i\mathbb{E}\left[\sum_{j\in D_i} 3d(i,\phi(i)) + e_{ij\phi(i)} + q_{ij\phi(i)} \,\middle|\, a_i\right] \\ &+ (1-p_i)\sum_{j\in D_i}\left(4(2c_i^d + c_i^c) + c_i + e_{ij} + c_j\right)\Bigg)\end{aligned}$$

Using the identities from the previous subsections, we get:

$$\begin{aligned}\mathbb{E}[T] &\leq \sum_{i\in\mathcal{C}}\Big(p_i\mathbb{E}\left[3C_{i\phi(i)} + E_{i\phi(i)} + Q_{i\phi(i)} \mid a_i\right] \\ &\qquad + (1-p_i)\left(4(2C_i^d + C_i^c) + C_i + E_i + Q_i\right)\Big) \\ &= \sum_{i\in\mathcal{C}}\left(p_i\mathbb{E}\left[V_{i\phi(i)} \mid a_i\right] + (1-p_i)\left(4(2C_i^d + C_i^c) + C_i + E_i + Q_i\right)\right).\end{aligned}$$

Now we use Lemma 12,

$$\mathbb{E}[T] \leq \sum_{i\in\mathcal{C}}\left(p_i(3C_i + E_i + Q_i) + (1-p_i)\left(4(2C_i^d + C_i^c) + C_i + E_i + Q_i\right)\right).$$

Let $p = 1 - 1/e^\gamma$. By Lemma 9, we know that $\min_{i\in\mathcal{C}} p_i \geq p$. Since $C_i^d \geq C_i$ for each i, the factor that multiplies p_i is not larger than the factor that multiplies $(1-p_i)$. Therefore, if we replace p_i by p, we get an upper bound on the summation.

$$\begin{aligned}\mathbb{E}[T] &\leq \sum_{i\in\mathcal{C}}\left(p(3C_i + E_i + Q_i) + (1-p)\left(4(2C_i^d + C_i^c) + C_i + E_i + Q_i\right)\right) \\ &= p(3C + E + Q) + (1-p)\left(4(2C^d + C^c) + C + E + Q\right) \\ &= p(2C + E + 2Q) + (1-p)\left(2(2C^d + C^c) + 2(2Q^d + Q^c) + C + E + Q\right).\end{aligned}$$

In the last equality used $Q = C$, $Q^d = C^d$ and $Q^c = C^c$. If we rearrange the terms,

we get:

$$\begin{aligned}\mathbb{E}[T] \leq\ & (p+1)C + (1-p)2(2C^d + C^c) \\ & + E \\ & + (p+1)Q + (1-p)2(2Q^d + Q^c),\end{aligned}$$

which is a sum of three parts, say $T_C + T_E + T_Q$, where the first and the last parts are equal. Next, we will bound the first part, T_C. Note that $C^d = \frac{\gamma C - C^c}{\gamma - 1}$ and that $C^c = (1-\rho)C$, where ρ is the irregularity factor. Substituting those into T_C:

$$\begin{aligned}T_C &= (p+1)C + (1-p)2(2C^d + C^c) \\ &= \left(\rho\left(\frac{4e^{-\gamma}}{\gamma - 1} - 2e^{-\gamma}\right) + 2 + 5e^{-\gamma}\right) C \\ &= \left(\rho\left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right)(1-\rho)(2 + 5e^{-\gamma})\right) C \\ &\leq \left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) C\end{aligned}$$

where we used the fact that $0 \leq \rho \leq 1$ and $1 < \gamma \leq 3$. Now, since $T_C = T_Q$ and $Q = C$, we have

$$\mathbb{E}[T] \leq T_C + T_E + T_Q \leq \left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) C + E + \left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) Q. \qquad \square$$

Theorem 5. *The LP-rounding algorithm described in Chapter 5 is a randomized* 2.48*-approximation algorithm to the Single Allocation Hub Location Problem.*

Proof. By Lemma 8, the expected cost of open hubs is at most γH. By Lemma 15, the overall expected cost of all demands $(i,j) \in \overrightarrow{\mathcal{D}}$ is, at most, $(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma-1})C + E + (2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma-1})Q$. Since we counted each demand twice, the expected connection cost of the algorithm is half of this value. By selecting the best possible value of γ, the expected cost of the solution is bounded as:

$$\begin{aligned}\mathbb{E}[cost] &= \mathbb{E}\left[\sum_{k \in O} f(k) + \sum_{\{i,j\} \in \mathcal{D}} (cost(i,j))\right] \\ &\leq \gamma H + \frac{1}{2}\left(\left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) C + E + \left(2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right) Q\right) \\ &\leq \min_{\gamma}\left\{\max\left\{\gamma, 2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma - 1}\right\}\right\}\left(H + \frac{C + E + Q}{2}\right) \\ &\leq 2.48\left(H + \frac{C + E + Q}{2}\right).\end{aligned}$$

The value $\gamma_0 = 2.48$ was obtained numerically by finding the point in which the functions $g_1(\gamma) = \gamma$ and $g_2(\gamma) = 2 + 3e^{-\gamma} + \frac{4e^{-\gamma}}{\gamma-1}$ intersect, which is the point the minimizes the expression above. The functions and the found value are depicted in Figure 5.4.

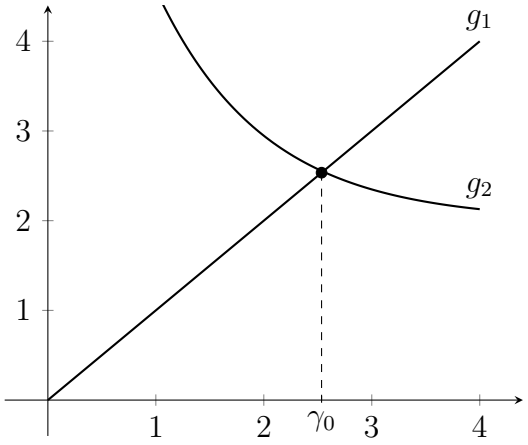


Figure 5.4: Intersection point of functions g_1 and g_2.

Since the second term of the cost is the value of the objective function of (LP), the expected cost is at most 2.48 the value of an optimal solution, and therefore the theorem holds. □

Chapter 6

Concluding remarks

In this thesis, we studied variations of Hub Location Problems from the perspective of approximation algorithms. The literature on such subject is not as rich as other location problems, such as the Facility Location Problem and the k-Median Problem. With respect to HLP, the existing approximation algorithms attempt to solve the problem of designating clients to hubs, but not the problem of locating the hubs to be opened and assigning clients to hubs as a single decision process. In our effort to improve the literature, we handled both location and allocation decisions simultaneously.

The first attempt to do so was tackling some variations of the problem by using state-of-the-art algorithms of location problems as subroutines. This is done by transforming an instance of the hub location problem into one instance of a corresponding location problem for which we know a good approximation. To our knowledge, the approximation algorithms derived using the reduction described in Chapter 4 are the first approximations for the six listed variants of HLP.

Our main contribution is a 2.48-approximation algorithm for the Single Allocation Hub Location Problem, based on LP-rounding and other strategies applied by famous algorithms for location problems. This algorithm is based on a new linear formulation to produce an initial fractional solution. The consecutive steps use both techniques that are standard in the location literature (to select and open hubs), and new techniques, which are specially crafted to deal with the assignment of clients to hubs. The former is based on the clustering algorithm for FLP [22, 11]; the latter required the development of new methods because a client is not always connected to the closest opened hub, as is the case of FLP.

The adopted strategy to assign clients to hubs uses different criteria to allocate hubs, depending on whether there is an open hub in its support set. This allows balancing the terms of the solution cost by exploiting the symmetries of the linear formulation. The achieved factor improves on the results obtained by the previous reduction for this problem. We believe the method to balance connection costs described in Chapter 5 can be extended for other hub location problems or similar problems whose objective functions are composed of multiple terms, in the same way that standard techniques of FLP algorithms are extended to other location problems.

While this LP-rounding algorithm improves on the previously obtained factor, we think that the analysis of the algorithm can be strengthened: our strategy only gives an

upper bound on the solution cost, since it does not consider the case where both demand endpoints have opened hubs in their neighborhood. Future works might also investigate new methods to balance the components of the objective function, extending our analysis of the LP-rounding algorithm. Several research opportunities remain in the field, and we hope that this work serves as a starting point to deepen the understanding of hub location problems.

Bibliography

[1] Warren P Adams and Hanif D Sherali. A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, 32(10):1274–1290, 1986.

[2] Sibel Alumur and Bahar Y. Kara. Network hub location problems: The state of the art. *European Journal of Operational Research*, 190(1):1–21, 2008.

[3] Sibel A. Alumur, Stefan Nickel, and Francisco Saldanha da Gama. Hub location under uncertainty. *Transportation Research Part B: Methodological*, 46(4):529–543, 2012.

[4] Ryuta Ando and Tomomi Matsui. Algorithm for Single Allocation Problem on Hub-and-Spoke Networks in 2-Dimensional Plane. In *Algorithms and Computation*, pages 474–483, 2011.

[5] Turgut Aykin. On the Location of Hub Facilities. *Transportation Science*, 22(2):155–157, 1988.

[6] Turgut Aykin. Lagrangian relaxation based approaches to capacitated hub-and-spoke network design problem. *European Journal of Operational Research*, 79(3):501–523, 1994.

[7] Turgut Aykin. The hub location and routing problem. *European Journal of Operational Research*, 83(1):200–219, 1995.

[8] Turgut Aykin. Networking Policies for Hub-and-Spoke Systems with Application to the Air Transportation System. *Transportation Science*, 29(3):201, 1995.

[9] Ramesh Bollapragada, Jeffrey Camm, Uday S Rao, and Junying Wu. A two-phase greedy algorithm to locate and allocate hubs for fixed-wireless broadband access. *Operations Research Letters*, 33(2):134–142, 2005.

[10] Peter Brucker. *"Monge"-property and efficient algorithms*, pages 137–141. Physica-Verlag HD, Heidelberg, 1992.

[11] J. Byrka and K. Aardal. An Optimal Bifactor Approximation Algorithm for the Metric Uncapacitated Facility Location Problem. *SIAM Journal on Computing*, 39(6):2212–2231, 2010.

[12] Jarosław Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An Improved Approximation for k-median, and Positive Correlation in Budgeted Optimization. In *Proc. of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 737–756, 2015.

[13] James Campbell, Andreas Ernst, and Mohan Krishnamoorthy. Hub location problems. *Facility location: application and theory*, 2002.

[14] James F. Campbell. Integer programming formulations of discrete hub location problems. *European Journal of Operational Research*, 72(2):387–405, 1994.

[15] James F. Campbell. Hub Location and the p-Hub Median Problem. *Operations Research*, 44(6):923–935, 1996.

[16] Moses Charikar and Sudipto Guha. Improved Combinatorial Algorithms for the Facility Location and k-Median Problems. In *Proc. of the 40^{th} Annual IEEE Symposium on Foundations of Computer Science*, pages 378–388, 1999.

[17] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A Constant-Factor Approximation Algorithm for the k-Median Problem. *Journal of Computer and System Sciences*, 65(1):129–149, 2002.

[18] Jeng-Fung Chen. A hybrid heuristic for the uncapacitated single allocation hub location problem. *Omega*, 35(2):211–220, 2007.

[19] Li-Hsuan Chen, Dun-Wei Cheng, Sun-Yuan Hsieh, Ling-Ju Hung, Chia-Wei Lee, and Bang Ye Wu. Approximation algorithms for single allocation k-hub center problem. In *Proc. of the CMCT*, pages 13–18, 2016.

[20] Li-Hsuan Chen, Dun-Wei Cheng, Sun-Yuan Hsieh, Ling-Ju Hung, Chia-Wei Lee, and Bang Ye Wu. *Approximation Algorithms for the Star k-Hub Center Problem in Metric Graphs*, pages 222–234. Springer International Publishing, Cham, 2016.

[21] Li-Hsuan Chen, Sun-Yuan Hsieh, Ling-Ju Hung, Ralf Klasing, Chia-Wei Lee, and Bang Ye Wu. *On the Complexity of the Star p-hub Center Problem with Parameterized Triangle Inequality*, pages 152–163. Springer International Publishing, Cham, 2017.

[22] Fabián A Chudak. Improved approximation algorithms for uncapacitated facility location. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 180–194. Springer, 1998.

[23] Vasek Chvatal. *Linear programming*. Macmillan, 1983.

[24] Ivan Contreras, Jean-François Cordeau, and Gilbert Laporte. Benders decomposition for large-scale uncapacitated hub location. *Operations Research*, 59(6):1477–1490, 2011.

[25] Ivan Contreras, Juan A Díaz, and Elena Fernández. Lagrangean relaxation for the capacitated hub location problem with single assignment. *OR Spectrum*, 31(3):483–505, 2009.

[26] Ivan Contreras, Juan A Díaz, and Elena Fernández. Branch and price for large-scale capacitated hub location problems with single assignment. *INFORMS Journal on Computing*, 23(1):41–55, 2011.

[27] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

[28] Isabel Correia, Stefan Nickel, and Francisco Saldanha da Gama. The capacitated single-allocation hub location problem revisited: A note on a classical formulation. *European Journal of Operational Research*, 207(1):92–96, 2010.

[29] Claudio B Cunha and Marcos Roberto Silva. A genetic algorithm for the problem of configuring a hub-and-spoke network for a ltl trucking company in brazil. *European Journal of Operational Research*, 179(3):747–758, 2007.

[30] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[31] Mark S Daskin. *Network and discrete location: models, algorithms, and applications*. John Wiley & Sons, 2011.

[32] Ricardo Saraiva de Camargo, Gilberto de Miranda Jr, and Henrique Pacca Luna. Benders decomposition for the uncapacitated multiple allocation hub location problem. *Computers & Operations Research*, 35(4):1047–1064, 2008.

[33] Ricardo Saraiva de Camargo, Gilberto de Miranda Jr, and Henrique Pacca Luna. Benders decomposition for hub location problems with economies of scale. *Transportation Science*, 43(1):86–97, 2009.

[34] Zvi Drezner and Horst W Hamacher. *Facility location: applications and theory*. Springer Science & Business Media, 2001.

[35] Jamie Ebery, Mohan Krishnamoorthy, Andreas Ernst, and Natashia Boland. The capacitated multiple allocation hub location problem: Formulations and algorithms. *European Journal of Operational Research*, 120(3):614–631, 2000.

[36] Andreas T. Ernst, Horst Hamacher, Houyuan Jiang, Mohan Krishnamoorthy, and Gerhard Woeginger. Uncapacitated single and multiple allocation p-hub center problems. *Computers & Operations Research*, 36(7):2230–2241, 2009.

[37] Andreas T Ernst and Mohan Krishnamoorthy. Solution algorithms for the capacitated single allocation hub location problem. *Annals of Operations Research*, 86:141–159, 1999.

[38] Reza Zanjirani Farahani, Masoud Hekmatfar, Alireza Boloori Arabani, and Ehsan Nikbakhsh. Hub location problems: A review of models, classification, solution techniques, and applications. *Computers & Industrial Engineering*, 64(4):1096–1109, 2013.

[39] Vladimir Filipović, Jozef Kratica, Dušan Tošić, and Djordje Dugošija. Ga inspired heuristic for uncapacitated single allocation hub location problem. *Applications of Soft Computing*, pages 149–158, 2009.

[40] Fedor V Fomin and Petteri Kaski. Exact exponential algorithms. *Communications of the ACM*, 56(3):80–88, 2013.

[41] Cees M Fortuin, Pieter W Kasteleyn, and Jean Ginibre. Correlation inequalities on some partially ordered sets. *Communications in Mathematical Physics*, 22(2):89–103, 1971.

[42] Sergio García, Mercedes Landete, and Alfredo Marín. New formulation and a branch-and-cut algorithm for the multiple allocation p-hub median problem. *European Journal of Operational Research*, 220(1):48–57, 2012.

[43] Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. W. H. Freeman New York, 2002.

[44] Dongdong Ge, Simai He, Yinyu Ye, and Jiawei Zhang. Geometric rounding: a dependent randomized rounding scheme. *Journal of Combinatorial Optimization*, 22(4):699–725, 2010.

[45] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[46] Sudipto Guha and Samir Khuller. Greedy Strikes Back: Improved Facility Location Algorithms. *Journal of Algorithms*, 31(1):228–248, 1999.

[47] S. L. Hakimi. Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph. *Operations Research*, 12(3):450–459, 1964.

[48] Lane A Hemaspaandra. Sigact news complexity theory column 74. *SIGACT News*, 43(2):51–52, 2012.

[49] Dorit S Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on computing*, 11(3):555–556, 1982.

[50] Masaru Iwasa, Hiroo Saito, and Tomomi Matsui. Approximation algorithms for the single allocation problem in hub-and-spoke networks and related metric labeling problems. *Discrete Applied Mathematics*, 157(9):2078 – 2088, 2009. Optimal Discrete Structures and Algorithms ODSA 2006.

[51] Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In *Proceedings of the thiry-fourth annual ACM Symposium on Theory of Computing*, pages 731–740. ACM, 2002.

[52] Kamal Jain and Vijay V. Vazirani. Approximation Algorithms for Metric Facility Location and k-Median Problems Using the Primal-dual Schema and Lagrangian Relaxation. *J. ACM*, 48(2):274–296, March 2001.

[53] Bahar Y Kara and Barbaros Ç Tansel. On the single-assignment p-hub center problem. *European Journal of Operational Research*, 125(3):648–655, 2000.

[54] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

[55] Jon Kleinberg and Éva Tardos. Approximation Algorithms for Classification Problems with Pairwise Relationships: Metric Labeling and Markov Random Fields. *J. ACM*, 49(5):616–639, September 2002.

[56] Madhukar R Korupolu, C Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems. *Journal of Algorithms*, 37(1):146–188, 2000.

[57] Alfred A Kuehn and Michael J Hamburger. A heuristic program for locating warehouses. *Management Science*, 9(4):643–666, 1963.

[58] Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. In *Automata, Languages and Programming*, pages 77–88, 2011.

[59] Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 222(0):45–58, 2013.

[60] Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. In *Proceedings of the Symposium on the Theory of Computing*, pages 901–910. ACM, 2013.

[61] Hongyu Liang. The hardness and approximation of the star -hub center problem. *Operations Research Letters*, 41(2):138–141, 2013.

[62] Cheng-Chang Lin, Jr-Yung Lin, and Yin-Chieh Chen. The capacitated p-hub median problem with integral constraints: An application to a chinese air cargo network. *Applied Mathematical Modelling*, 36(6):2777–2787, 2012.

[63] Jyh-Han Lin and Jeffrey Scott Vitter. Approximation algorithms for geometric median problems. *Information Processing Letters*, 44(5):245–249, 1992.

[64] Vladimir Marianov and Daniel Serra. Location models for airline hubs behaving as m/d/c queues. *Computers & Operations Research*, 30(7):983–1003, 2003.

[65] George L Nemhauser and Laurence A Wolsey. Integer programming and combinatorial optimization. *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992). Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin*, 20:8–12, 1988.

[66] Morton E. O'Kelly. The Location of Interacting Hub Facilities. *Transportation Science*, 20(2):92–106, 1986.

[67] Morton E. O'Kelly. A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, 32(3):393–404, December 1987.

[68] Judea Pearl. Heuristics: intelligent search strategies for computer problem solving. *Reading, MA*, 1984.

[69] Lehilton LC Pedrosa, Vinicius F dos Santos, and Rafael CS Schouery. Uma aproximação ao para o problema de alocação de terminais. In *Anais do CSBC 2016 (1º ETC - 2016)*, 2016.

[70] Hasan Pirkul and David A Schilling. An efficient procedure for designing single allocation hub and spoke systems. *Management Science*, 44(12-part-2):S235–S242, 1998.

[71] Prabhakar Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2):130–143, 1988.

[72] David B. Shmoys, Éva Tardos, and Karen Aardal. Approximation Algorithms for Facility Location Problems (Extended Abstract). In *Proc. of the Twenty-ninth Annual ACM Symposium on Theory of Computing*, pages 265–274, 1997.

[73] Jinhyeon Sohn and Sungsoo Park. A linear program for the two-hub location problem. *European Journal of Operational Research*, 100(3):617–622, 1997.

[74] Jinhyeon Sohn and Sungsoo Park. The single allocation problem in the interacting three-hub network. *Networks*, 35(1):17–25, 2000.

[75] John F Stollsteimer. A working model for plant numbers and locations. *Journal of Farm Economics*, 45(3):631–645, 1963.

[76] Maxim Sviridenko. An Improved Approximation Algorithm for the Metric Uncapacitated Facility Location Problem. In *Proceedings of the Integer Programming and Combinatorial Optimization*, pages 240–257, 2002.

[77] Rex S Toh and Richard G Higgins. The impact of hub and spoke network centralization and route monopoly on domestic airline profitability. *Transportation Journal*, pages 16–27, 1985.

[78] David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge University Press, 2011.

[79] Hande Yaman and Sourour Elloumi. Star p-hub center problem and star p-hub median problem with bounded path lengths. *Computers & Operations Research*, 39(11):2725–2732, 2012.

[80] Hande Yaman, Bahar Y Kara, and Barbaros Ç Tansel. The latest arrival hub location problem for cargo delivery systems with stopovers. *Transportation Research Part B: Methodological*, 41(8):906–919, 2007.

[81] Peng Zhang. A new approximation algorithm for the k-facility location problem. *Theoretical Computer Science*, 384(1):126–135, 2007.