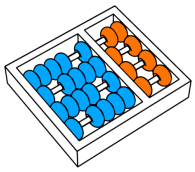
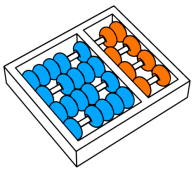


Lehilton Lelis Chaves Pedrosa

"Approximation Algorithms for Facility Location Problems and Other Supply Chain Problems"

"*Algoritmos de Aproximação para Problemas de Alocação de Instalações e Outros Problemas de Cadeia de Fornecimento*"

CAMPINAS
2014



University of Campinas
Institute of Computing

Universidade Estadual de Campinas
Instituto de Computação

Lehilton Lelis Chaves Pedrosa

"Approximation Algorithms for Facility Location Problems and Other Supply Chain Problems"

Supervisor/*Orientador*: **Prof. Dr. Flávio Keidi Miyazawa**
Co-Supervisor/*Coorientador*: **Prof. Dr. Maxim Sviridenko**

"Algoritmos de Aproximação para Problemas de Alocação de Instalações e Outros Problemas de Cadeia de Fornecimento"

Doctorate Thesis presented to the Post Graduate Program of the Institute of Computing of the University of Campinas to obtain a Doctor degree in Computer Science.

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Computação da Universidade Estadual de Campinas para obtenção do título de Doutor em Ciência da Computação.

This volume corresponds to the final version of the Thesis defended by Lehilton Lelis Chaves Pedrosa, under the supervision of Prof. Dr. Flávio Keidi Miyazawa.

Este exemplar corresponde à versão final da Tese defendida por Lehilton Lelis Chaves Pedrosa, sob orientação de Prof. Dr. Flávio Keidi Miyazawa.

Supervisor's signature / *Assinatura do Orientador*

CAMPINAS
2014

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

P343a

Pedrosa, Lehilton Lelis Chaves, 1985-
Approximation algorithms for facility location problems and other supply chain problems / Lehilton Lelis Chaves Pedrosa. – Campinas, SP : [s.n.], 2014.

Orientador: Flávio Keidi Miyazawa.
Coorientador: Maxim Sviridenko.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Otimização combinatória. 2. Algoritmos aproximados. 3. Cadeia de suprimentos. I. Miyazawa, Flávio Keidi,1970-. II. Sviridenko, Maxim. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Algoritmos de aproximação para problemas de alocação de instalações e outros problemas de cadeia de fornecimento
Palavras-chave em inglês:
Combinatorial optimization
Approximation algorithms
Supply chains
Área de concentração: Ciência da Computação
Titulação: Doutor em Ciência da Computação
Banca examinadora:
Flávio Keidi Miyazawa [Orientador]
Jayme Luiz Szwarcfiter
Luciana Salete Buriol
Cristina Gomes Fernandes
Luis Augusto Angelotti Meira
Data de defesa: 04-07-2014
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Defesa de Tese de Doutorado em Ciência da Computação, apresentada pelo(a) Doutorando(a) **Lehilton Lelis Chaves Pedrosa**, aprovado(a) em **4 de julho de 2014**, pela Banca examinadora composta pelos professores doutores:

Prof(ª). Dr(ª). Jayme Luiz Szwarcfiter
Titular

Prof(ª). Dr(ª). Luciana Salete Buriol
Titular

Prof(ª). Dr(ª). Cristina Gomes Fernandes
Titular

Prof(ª). Dr(ª). Luis Augusto Angelotti Meira
Titular

Prof(ª). Dr(ª). Flávio Keidi Miyazawa
Presidente

Institute of Computing /*Instituto de Computação*
University of Campinas /*Universidade Estadual de Campinas*

Approximation Algorithms for Facility Location Problems and Other Supply Chain Problems

Lehilton Lelis Chaves Pedrosa[1]

July 4, 2014

Board of Examiners/*Banca Examinadora*:

- Prof. Dr. Flávio Keidi Miyazawa (Supervisor/*Orientador*)
- Prof. Dr. Jayme Luiz Szwarcfiter
  COPPE - UFRJ
- Prof. Dr. Luciana Salete Buriol
  Institute of Informatics - UFRGS
- Prof. Dr. Cristina Gomes Fernandes
  Department of Computer Science - USP
- Prof. Dr. Luis Augusto Angelotti Meira
  Faculty of Technology - UNICAMP
- Prof. Dr. Eduardo Candido Xavier (Substitute/*Suplente*)
  Institute of Computing - UNICAMP
- Prof. Dr. Orlando Lee (Substitute/*Suplente*)
  Institute of Computing - UNICAMP
- Prof. Dr. Yoshiko Wakabayashi (Substitute/*Suplente*)
  Department of Computer Science - USP

[1]Financial support: CAPES (2010); FAPESP, grant number 2010/20710-4 (2011-2014).

Abstract

This thesis gives approximation algorithms for a series of NP-hard supply chain problems, that range from the packing, the distribution, and the inventory management of items, to the design of the supply network. Several novel techniques are employed in these approximations, and many of them may be extended to different problems.

In the Metric Facility Location Problem (FLP), given sets of clients and facilities in a metric space, the objective is to open a subset of facilities, such that the cost to open facilities, and the cost to connect each client to an open facility is minimum. This work considers the case when the distance function is the square of a metric, which is suitable for many applications that do not satisfy the metric assumption. A lower bound of 2.04 on the approximation factor is given, and it is shown that an LP-rounding algorithm matches this factor. More interestingly, a new technique to obtain factor-revealing programs is presented, and it is used to analyze primal-dual algorithms, giving tight factors under a squared metric, or improving known factors under a metric. Also, it is shown that the Continuous FLP (ConFL), when a facility location is any point of the Euclidean space, may be reduced to the FLP by using the so called center sets. Center sets for the L_2^α-norm are given, and thus approximations for ConFL with this norm are derived, for $\alpha \geq 1$. As a by-product, a PTAS for k-medians with the L_2^α-norm is obtained.

The Production and Distribution Problem (PDP) is the problem of minimizing ordering, transportation and inventory costs of a supply chain formed by a set of warehouses and retailers over a finite time horizon. This is a common generalization of the FLP, and the Joint Replenishment Problem (JRP), and allows the coordination of network design and inventory management decisions, thus leading to significant economy. A 2.77-approximation for the PDP is given under the assumption that holding and distribution cost functions satisfy a natural extension of the triangle inequality. Other generalizations of the JRP are also considered, such as the One-Warehouse Multiple-Retailer Problem (OWMR), for which a 5-approximation is given in the case that warehouse and retailer inventory costs are independent. Under this assumption, no approximation algorithm was known previously.

Finally, there are given APTAS's for the circle bin packing, and the circle strip packing.

Resumo

Esta tese apresenta algoritmos de aproximação para uma série de problemas NP-difíceis que surgem nas várias etapas de uma cadeia de fornecimento e vão desde o empacotamento, a distribuição e a gerência de estoque até o projeto da rede de abastecimento. Várias novas técnicas são apresentadas, das quais muitas podem ser estendidas a problemas diferentes.

No Problema de Alocação de Instalações Métrico (FLP), são dados conjuntos de clientes e instalações em um espaço métrico e o objetivo é alocar um subconjunto de instalações de modo que o custo de alocação e o custo de conexão de cada cliente a uma instalação seja mínimo. Este trabalho considera o caso em que a função de distância é o quadrado de uma métrica, que é adequado para várias aplicações que não pressupõe uma métrica. Obtém-se um limite inferior para aproximação de 2,04 e demonstra-se que um algoritmo baseado em arredondamento alcança esse valor. Um resultado mais significativo é uma nova técnica para obter programas reveladores de fator, que é utilizada para analisar algoritmos primais-duais, provendo fatores justos no caso da métrica ao quadrado, ou melhorando fatores conhecidos no caso métrico. Além disso, mostra-se que o FLP Contínuo (ConFL), variante em que um local de instalação é qualquer ponto do espaço euclidiano, pode ser reduzido ao FLP usando os chamados conjuntos de centros. Obtêm-se conjuntos de centros para a norma L_2^α e, daí, aproximações para o ConFL para $\alpha \geq 1$. Como um subproduto, obtém-se um PTAS para o problema das k-medianas nessa norma.

O Problema de Produção e Distribuição (PDP) tem como objetivo minimizar os custos de pedido, transporte e inventário de uma cadeia de fornecimento formada por armazéns e lojas durante um período determinado. Esse problema generaliza o FLP e o Problema de Reposição Conjunta (JRP) e permite a coordenação das decisões de projeto de rede e gerenciamento de estoque, conduzindo a uma economia significativa. Provê-se uma 2,77-aproximação quando os custos de armazenamento e distribuição satisfizerem a uma extensão natural da desigualdade triangular. Outras variantes do JRP também são consideradas, como o Problema Um-Armazém Múltiplas-Lojas, para o qual é dada uma 5-aproximação quando os custos de estoque dos armazéns e das lojas forem independentes. Sob essa hipótese, não se conhecia nenhuma aproximação anteriormente.

Finalmente, são apresentados APTASs para problemas de empacotamento de círculos.

Acknowledgements

I'd like to thank my supervisor Flávio Miyazawa, my co-supervisor Maxim Sviridenko, the co-authors Cristina Fernandes, Luis Meira, Rafael Schouery, and Yoshiko Wakabayashi, and all the other uncountable friends with whom I've spent some time during these joyful four years.

Thanks!

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Approximation algorithms have fascinated an ever-greater community of researchers working on optimization and theory of computing. Whereas the so called NP-hard problems cannot be efficiently solved under the largely believed assumption that P $\neq$ NP, the approximation algorithms provide a compromise between the accuracy of a solution, and the time used to compute it. The challenge of both practitioners and theorists is to obtain a solution as close as possible to the optimum, while maintaining the running time small. The study of approximation algorithms helps not only unveiling which problems can be solved efficiently, but also how well they can be solved.

In optimization problems, the objective is to minimize or maximize the value of a given function. In these situations, one has to distinguish between a feasible solution, that is any element of the function domain, from an optimal solution, that is a feasible solution with maximum or minimum value. For example, one might be interested in finding a path between two locations with minimum length, or selling items to several clients according to the distribution that maximizes the profit. While the best solution may be difficult to find, an acceptable solution can be easily obtained in many cases. The trade-off between the solution quality and the algorithm's running time is normally reflected on the ratio between the values of generated and optimal solutions. For an approximation algorithm, this ratio has a proven bound, and the running time is always polynomial.

Many optimization problems appear in the decision-making process of industries of several areas. In location problems, one looks for placing facilities in certain locations so as to minimize some defined client allocation cost function. For instance, a telecommunication company may wish to place concentrator nodes in a given network, or a supermarket network may need to locate a set of warehouses to serve its retailers. In the inventory management, the problems aim at minimizing the ordering and stock holding costs. In packing problems, one needs to arrange several items in available recipients minimizing the consumed resources. Although in different applications one deals with several kinds

of objects, it is common to define similar problems using simpler abstract models. For example, both the communication and retailer networks are modeled as graphs. Algorithms for these models can be analyzed formally, and then applied to practical scenarios without or with minor modifications.

This thesis investigates several optimization problems that appear in the management of an organization's supply chain and related activities. It comprises numerous tasks that extend from the configuration of a supply chain network, up to the packing, transportation, and inventory control of goods. This work is oriented from a theoretical point of view, and the objective is invariably obtaining approximation algorithms for the studied problems in their abstract forms. Several novel techniques are developed for these, as well as for many of their variants.

1.1 Approximation algorithms and thesis overview

An optimization problem is the task to find the best solution among a well-defined set of feasible solutions. This set of solutions is the domain of an associated real-valued function, that is called the objective function. Here, the best solution may refer either to one with minimum value among all solutions, or to one with maximum value. In the continuous optimization theory, the domain is a subset of the Euclidean space, and is normally defined by a set of inequalities and equalities. In combinatorial optimization, the domain is a discrete set. Typically, the domain is not defined explicitly, but defined as the set of elements in a finite enumerable set that satisfies some easily testable properties. However, the naive idea of evaluating the objective function for each element of the domain to find the optimal value is impractical, even for relatively moderated instances.

Several important optimization problems are NP-hard. This means that, under the hypothesis that P $\neq$ NP, there are no efficient algorithms that compute the optimal value. The idea of an efficient algorithm is subjective, so, in this work, it is defined as an algorithm that runs in polynomial time. Although having a polynomial running time is not a guarantee of being practical, running in superpolynomial time is certainly an indication of a slow algorithm. This is reasonable, since in most cases one is interested in the asymptotic analysis, that defines the behavior of the algorithm when it is fed with large instances. Such notion of efficiency is standard, and has been used since the 1960's [49].

The difficulty of solving large instances of a hard problem leads one to look for alternatives to inefficient exact algorithms. There are several options, such as using heuristics, or restricting the problem. In these cases, it is common to give up an optimal solution for a non-optimal acceptable solution. The concept of solution quality may vary with the problem, yet, in most cases, a good solution is one whose value is close to the op-

timum. An algorithm that runs in polynomial time, and that produces a solution with value guaranteed to be close to the optimal value is called an *approximation algorithm*.

The values of generated and optimal solutions can be compared either by their difference, or by their ratio. The measure used for most problems is the so called approximation factor, that is a bound on the value of the generated solution divided by the value of an optimal solution for any given instance. Therefore, an algorithm with approximation factor one is an exact algorithm, and the closer to one is the factor, the better is the algorithm. For some optimization problems, even the task to obtain an approximation algorithm whose factor is within a given bound is NP-hard. In this case, such a bound is said to be an approximation lower bound for a minimization problem, or an approximation upper bound for a maximization problem. On the other hand, an approximation algorithm gives an approximation upper bound for a minimization problem, or an approximation lower bound for a maximization problem.

Various problems have a rich history of approximation algorithms, such as the classical Metric Facility Location Problem, the k-means clustering, the packing of rectangular items, and many others. Several other important problems have just few or no works that give guarantees on the approximation factor, as is the case of some production and distribution problems in the inventory management, and the packing of non-rectangular items. Even for widely studied problems, there is a lengthy list of relevant variants that are underexplored from the perspective of approximation algorithms. Examples of variants are the Facility Location Problem with relaxed metric distance functions, and the clustering problem with penalty per cluster. This thesis aims at addressing this gap, by providing approximation algorithms for many of these problems

There is a variety of techniques used to obtain approximation algorithms, that are based on local search, linear programming, semidefinite programming, etc. Similar problems, or problems with similar combinatorial structures, can usually be tackled by the same or by comparable techniques. Most of this work is devoted to the study of approximation algorithms based on linear programming techniques, such as rounding of fractional solutions, and the primal-dual method. This approach is used, for example, on the Facility Location Problem, and on the inventory management problems. In the case of the considered clustering and packing problems, the used techniques are essentially specific to their geometrical nature.

This work spans over several topics of the supply chain management and related literature. For the class of location problems, the Facility Location Problem with squared metric distance functions, and a clustering problem called Continuous Facility Location Problem are studied. For the class of inventory management problems, the One-Warehouse Multiple-Retailer Problem with independent warehouse holding costs, and the integrated problem of production and distribution, that is a generalization of the Facility Location

Problem and the Joint Replenishment Problem, are studied. Additionally, the problems of packing circles into unit square bins, or into an unbounded unit width strip are considered. Approximation algorithms are given for problems in each of the investigated areas, by reviewing and applying known methods, while giving new insights, and developing new techniques.

Although this work views problems purely as abstract models, they have a wide range of applications. The Facility Location Problem appears in network projects, such as the installation of cache and routers [63, 92], or in data clustering and server replication [4, 63, 77, 117]. Other clustering problems, such as k-means, have diverse applications in image compression, distribution of resources, cellular biology, etc. [48]. Non-metric versions of classical problems are often considered, such as the variant of the Traveling Salesman Problem whose distance function is the Euclidean distance raised to some exponent, that finds application in the power attribution of wireless networks [56], or the version whose distance satisfies some relaxed triangle inequality [20]. The Joint Replenishment Problem appears in the inventory management of many multi-product environments [81].

1.2 Definitions and main techniques

In this section, approximation algorithms and related concepts are defined formally. Also, the techniques used in this thesis, and the main considered problems are described.

1.2.1 Approximation algorithms

Consider a minimization problem, and a corresponding algorithm $\mathcal{A}$. For a given problem instance I, the value of the solution obtained by the algorithm when it is applied to I is denoted by $\mathcal{A}(I)$. Also, the value of an optimal solution for I is denoted by $\mathrm{OPT}(I)$. The *approximation ratio* or *approximation factor* of $\mathcal{A}$ is defined as

$$\mathcal{R}_{\mathcal{A}} := \sup_I \{\mathcal{A}(I)/\mathrm{OPT}(I)\},$$

where I ranges over the complete set of instances. If $\mathcal{A}$ runs in polynomial time on the size of every instance, then it is called an approximation algorithm with factor $\mathcal{R}_{\mathcal{A}}$. In many cases, obtaining a bound on the approximation ratio is easier than computing the ratio itself. For a given number α, an approximation algorithm $\mathcal{A}$ is called an *α-approximation* if α is not smaller than $\mathcal{R}_{\mathcal{A}}$. Similarly, the asymptotic approximation ratio of $\mathcal{A}$ is defined as

$$\mathcal{R}^{\infty}_{\mathcal{A}} := \limsup_{h\to\infty} \sup_I \{\mathcal{A}(I)/\mathrm{OPT}(I) : \mathrm{OPT}(I) = h\},$$

and $\mathcal{A}$ is called an asymptotic approximation algorithm with factor $\mathcal{R}^{\infty}_{\mathcal{A}}$ if it runs in polynomial time on the size of I.

A family of algorithms $\{\mathcal{A}_\varepsilon\}$ with parameter ε for a given minimization problem is called a polynomial-time approximation scheme (PTAS) if, for every $\varepsilon > 0$, $\mathcal{A}_\varepsilon$ is a $(1+\varepsilon)$-approximation. Analogously, a family of algorithms $\{\mathcal{A}_\varepsilon\}$ is called an asymptotic polynomial-time approximation scheme (APTAS) if, for every $\varepsilon > 0$, $\mathcal{A}_\varepsilon$ is an asymptotic $(1+\varepsilon)$-approximation. Also, an efficient polynomial-time approximation scheme (EPTAS) is a PTAS whose running time is bounded by a polynomial on the size of the input, and the exponent of such polynomial does not depend on ε.

A randomized algorithm is an algorithm $\mathcal{A}$ that, in addition to the instance I, receives as input a random sequence of bits. In such algorithms, the decisions are taken according to probability distributions, and therefore the value of the generated solution is itself a random variable. In this setting, one is not interested in bounding the value of the worse case solution, but in the expected value of the generated solution. For a given number α, a polynomial-time algorithm $\mathcal{A}$ is a randomized α-approximation if $E[\mathcal{A}(I)] \leq \alpha\,\mathrm{OPT}(I)$ for every instance I, where $E[\mathcal{A}(I)]$ is the expected value of the generated solution.

1.2.2 Linear programming techniques

An integer linear program (ILP) is the task to minimize or maximize a linear function of integer variables that are constrained by a set of linear inequalities. In a mixed integer linear program (MILP), some of the variables are allowed to assume non-integer values. A common technique to solve combinatorial optimization problems is encoding them as ILP's or MILP's, and then using standard techniques, such as the branch-and-bound and branch-and-cut algorithms. Classical NP-hard problems, as the knapsack problem, can be easily formulated as ILP's, and thus solving ILP's in general is also NP-hard. Indeed, for many problems, solving the ILP directly is only possible for very small instances. For example, it has been shown that branch-and-bound algorithms must almost surely explore a superpolynomial number of nodes to obtain optimal solutions for k-medians [2].

Although it is impractical to solve ILP formulations directly, they are useful to obtain and analyze approximation algorithms. Often, one considers a linear programming (LP) relaxation of an ILP or of an MILP, which is obtained by relaxing the corresponding integrality constraints. The optimal value of this relaxation provides a bound on the value of the original problem, so that the value of a solution generated by the algorithm can be compared to the optimal value. There are standard techniques that take advantage of the relaxations. Among such methods are the primal-dual method, that considers the dual problem of the relaxation, and the LP-rounding technique, that solves the relaxation, and rounds the fractional solution to obtain an integer feasible solution.

The *primal-dual* method uses the dual problem of the relaxed linear program to obtain a bound on the optimal value, while generating a solution to the original problem. In such

algorithms, the costs of the obtained solution are associated with variables of a feasible dual solution. By the weak duality theorem, the value of such solution is a bound on the optimal value of the relaxation. A classical example is given by Jain and Vazirani [76], who use the primal-dual technique to obtain an approximation algorithm for the Metric Facility Location Problem. The technique has also been used in a long sequence of works for several problems [17, 58, 59, 74, 90, 118, 119, 132].

A particular class of primal-dual algorithms uses the dual-fitting technique. In such algorithms, one constructs an infeasible dual solution with value that matches the cost of the generated integral solution. To obtain a feasible dual solution, each dual variable is scaled by a certain number, so that obtaining an approximation factor reduces to calculating such a number. Dual fitting has been used to obtain approximation algorithms for the Facility Location Problem, whose analysis employs the so called families of factor-revealing linear programs [72].

The set of techniques that transform the solution of a linear programming relaxation into a feasible integer solution is commonly referred to as *LP-rounding*. Many of these techniques are applied to formulations whose integer variables are binary, that is, the goal is to round a fraction either to zero, or to one. Different approaches are used to achieve this goal, such as rounding up variables depending whether they are larger than a specified value. LP-rounding is also used in combination with randomized algorithms, in which fractional values are interpreted as probabilities. This technique has been used to obtain approximation algorithms for several problems, such as the Facility Location Problem [37], and the Joint Replenishment Problem [91].

1.2.3 Description of the main problems

The major problems investigated in this thesis have varied objectives, and may be grouped in three main sets: location and clustering problems, that include the Facility Location Problem, and the Continuous k-medians; inventory management problems, that include the Joint Replenishment Problem; and packing problems, that include the Circle Bin Packing. These problems are described in the following.

In the *Facility Location Problem* (FLP), one is given a set of facility locations, and a set of clients. Installing a facility at some location i incurs a fixed opening cost f_i, and assigning a client j to a facility at location i incurs a connection cost c_{ij}. The objective is to install a subset of facilities so that the total opening and connection costs are minimized. In the *Continuous k-medians*, one is given a set of points in the d-dimensional Euclidean space, and an integer $k \geq 1$. The objective is to find a set of k points in the space, that are called centers, while minimizing the sum of the distance from each point to its closest center. Traditionally, the considered distance is the Euclidean distance, but

other symmetric real-valued binary functions over the space may also be used. A widely considered particular case is the so called *k-means*, that uses the square of the Euclidean distance.

In the *Joint Replenishment Problem* (JRP), there are N kinds of items, that may be demanded in each of T periods. For each period t, the number d_{it} represents the quantity of items of kind i that are requested. This demand must be served by the currently held stock of this item, that is initially empty. Replenishing the stock of item i is done by ordering items to the warehouse, and each such order incurs a constant nonnegative cost K_i, independent of the quantity requested. Also, for every period a retailer places an order, a fixed setup joint ordering cost K_0 is charged, that is independent of the number of ordered items. For each unit of item i ordered at time s and delivered at a later period t, there is a nonnegative cost h_{ist} for holding the unit in the stock. The objective is to minimize the total ordering and holding costs, while satisfying all demands.

In the *Circle Bin Packing*, one is given a list of circles, and an unlimited number of unit squares, that are called bins. A circle is said to be packed if it is fully contained in one bin, and does not intersect any other circle. The objective of the problem is to pack all circles using the minimum number of bins. In the *Circle Strip Packing*, there is only one recipient of unit width and unbounded height, and the objective is to pack all circles using the strip of minimum height.

1.2.4 Variants and common assumptions

Many problems receive, as input, functions that represent some associated service cost. For example, in the FLP, the distance function corresponds to the cost to connect a client to a given facility. In the JRP, the holding cost function defines how much it will cost holding one unit of an item in the stock during a given period. Depending on the assumptions made on these functions, the corresponding problems become more or less general. As a consequence, their approximability is also affected by the hypotheses over the service cost. In the following, premises on the distance and holding cost functions commonly assumed in the literature are described.

Distance functions. Given a set of elements, the distance function is the measure that represents the dissimilarity between any two given elements, and may represent a physical distance, a time interval, etc. The FLP cannot be approximated by a constant factor with general distance functions, unless P = NP. Therefore, this problem is commonly studied under the assumption that the underling distance function is metric. A distance function c for the FLP is a *metric* if it satisfies the triangle inequality, that is, for every clients j and j', and facilities i and i', it holds $c_{ij} \leq c_{ij'} + c_{i'j'} + c_{i'j}$.

Although assuming the triangle inequality is reasonable in many cases, such as when costs correspond to physical distances, there are cases for which such restrictions cannot be made. Since the general case does not have good approximations, a natural alternative is to relax the triangle inequality in a controlled way. One possibility is using the so called τ-relaxed triangle inequality, which means that, for any clients j and j', and facilities i and i', it holds $c_{ij} \leq \tau(c_{ij'} + c_{i'j'} + c_{i'j})$. Another way of relaxing the triangle inequality is considering the distance function that is the square of a metric, that is specially useful in problems whose set of elements are points of the Euclidean space, such as in the k-means. A squared metric distance function satisfies the 3-relaxed triangle inequality.

Holding cost functions. In the JRP, the holding cost function may have one of several structures. The classical literature considers the additive holding cost, when for each time step t and item i, there is a fixed cost for holding one unit of item, h_{it}, and the total holding cost to keep one unit from time step r to time step s is given by the sum of partial holding costs, $h_{irs} = \sum_{t=r}^{s} h_{it}$. More general models consider holding cost functions under the assumption that h_{ist} is monotonically non-increasing in s for some fixed t. Another studied variant is the so called JRP with deadlines, that is the particular case in which the holding cost function is either zero or infinity, or, equivalently, in which each demand must be satisfied in a given time interval.

In the generalization of the JRP called the One-Warehouse Multiple-Retailer Problem (OWMR), the items may be held in the warehouse before being transported to a retailer. In this problem, the holding cost is given by a function h_{rs}^{it} that corresponds to the value of holding one unit of item i in the warehouse from time r to time s, and then holding the item in the retailer from time s to time t. This problem has been studied with additive holding cost functions, or with a more general monotonic holding structure. Specifically, there is an approximation algorithm for the case in which, for each item i, and fixed times r and t, the function h_{rs}^{it} is either non-increasing or non-decreasing in r.

1.3 Results and thesis organization

This thesis gives approximation algorithms for a broad list of problems, and is organized so that closely related problems are considered in the same chapter. The techniques used for a set of problems may vary significantly from those of others, and thus each chapter is made as self-contained as possible. While general notions have been given in this introduction, definitions and motivations that are specific to each problem are deferred to the corresponding chapters. In particular, in each chapter, a special section gives a literature synopses of the main problems being studied, and shows how the corresponding approximation algorithms have improved upon the previous works.

In Chapter 2, we address a variant of the FLP in which the considered distance function is the square of a metric. Although the Metric FLP is part of the combinatorial optimization folklore, and is widely studied from the approximation algorithms point of view, few works consider the case with more relaxed restrictions on the distance function. The square of a metric is an intermediate class of distance functions, that is not as restrictive as requiring the triangle inequality, but does not allow too general functions, for which there would be no constant approximations. As expected, this variant is harder to approximate, since, as we show in Chapter 2, it cannot be approximated by a factor smaller that 2.04, unless P = NP. This lower bound is a counterpart of the lower bound of 1.463 for the metric case [62].

To obtain approximation algorithms, we have taken the natural way, that is analyzing the existing algorithms for the Metric FLP, when they are applied to more relaxed distance functions. Algorithms based on LP-rounding and primal-dual techniques are analyzed. We verify that, for the squared metric variant, the algorithm of Chudak and Shmoys [38] has the best possible approximation factor, as it matches the lower bound of 2.04. This is a surprising result, since it extends in spirit the analysis made by Byrka and Aardal [28] for the metric case, for which this algorithm is only a 1.575-approximation, and thus does not match the corresponding lower bound of 1.463.

Perhaps more important is the study of the primal-dual algorithms of Jain *et al.* [72], that, although achieve slightly worse approximation factors, are very efficient when compared to the LP-rounding approach. These algorithms use the dual fitting technique, and are analyzed by the so called factor-revealing LP's. Solving one of such LP's with the aid of a computer gives a bound on the approximation factor for instances of a given size. The approximation factor, on the other hand, must be hand calculated by tedious proofs, that depend on guessing general dual solutions for the factor-revealing linear programs. Using the same approach for the squared metric case proved to be a very tough task, since the corresponding factor-revealing programs are not linear. Instead, we introduce a different approach. The main result of this chapter is a novel technique to derive a family of upper bound factor-revealing programs (UPFRP), which can be solved by a computer to give a bound on the approximation factor directly. Obtaining an UPFRP is mostly straightforward, and does not require guessing a dual solution. By using UPFRP's, tight approximation factors for the primal-dual algorithms under the squared metric are obtained, and the factors for the metric case are improved in some cases.

Also, other generalizations of the FLP are studied, such as the variant in which the distance function satisfies the τ-relaxed triangle inequality. This chapter contains joint work with Cristina G. Fernandes, Luis A. A. Meira, and Flávio K. Miyazawa. An extended abstract was presented at APPROX 2012 [53], and a full version article has been submitted for publication.

In Chapter 3, we address the Continuous FLP. The objective is to partition a set of points of the Euclidean space, while minimizing the cost to connect each point to its cluster center given by some known distance function. Differently from the Continuous k-medians problem, for which the number of clusters is limited by k, in the Continuous FLP, the number of clusters is controlled by a fixed opening cost. We show that this problem can be reduced to the corresponding discrete version of the FLP. Indeed, we obtain an $(\alpha + \varepsilon)$-approximation for the Continuous FLP from an α-approximation for the FLP whose clients and facilities are finite sets of points of the space. This reduction depends on the existence of the so called center sets for a given distance function. Informally, a center set is a small set of candidate centers including one point that is a $(1 + \varepsilon)$-approximated solution for the corresponding 1-median problem.

Combining center sets previously used for the Continuous k-medians with known approximation algorithms for the FLP, it is possible to obtain approximations for the Continuous FLP with several distance functions. For the Euclidean distance, an approximation scheme is given by combining the center sets used by Kumar *et al.* [86] for k-medians, and the approximation scheme by Arora *et al.* [7] for the Euclidean FLP. For the squared Euclidean distance, the center sets given by Inaba *et al.* [71] are used in combination with the 2.04-approximation presented in Chapter 2, thus obtaining a $(2.04+\varepsilon)$-approximation for ConFL with this distance.

Also, we show how to obtain center sets when the considered distance function is the Euclidean norm raised to some power α, for $\alpha \geq 1$. By noticing that this function satisfies the $3^{\alpha-1}$-relaxed triangle inequality, and using results from Chapter 2, we also derive approximations for the Continuous FLP in this case. As a by-product, by combining the obtained center sets with the framework of Kumar *et al.* [86], we obtain an approximation scheme for the Continuous k-medians problem with L_2^α-norm. This chapter contains joint work with Luis A. A. Meira, and Flávio K. Miyazawa, and a paper containing these results has been submitted for publication.

In Chapter 4, a series of supply chain problems is examined. First, we study a common generalization of the FLP and the JRP, here called the Production and Distribution Problem (PDP). This problem is considered, for example, by Pochet and Wolsey [116]. Unlike the JRP, for which the distribution network is fixed, or the FLP, that ignores the decisions of inventory management, in the PDP, transportation and holding costs must be minimized in an integrated manner. We present the first approximation algorithms for this problem under a natural assumption that combines metric distances, and monotonic holding costs. A 2.77-approximation based on LP-rounding is given. This algorithm is extended to the case with retailer ordering costs and additive holding cost functions, for which a 5-approximation is given. This chapter contains joint work with Maxim Sviridenko, and an extended abstract was presented at LATIN 2014 [115].

Also, we consider the OWMR with a more general holding cost structure than that considered in the previous known approximation, which is the LP-rounding 1.8-approximation by Levi *et al.* [91]. In their algorithm, they assume that the holding cost function satisfies certain monotonicity properties. Namely, they assume that the cost of holding one unit of item either decreases, or increases as the fraction of time that the item is held on the warehouse increases. For many applications, this is an unnatural assumption, since it implies that the warehouse holding cost is dependent on the retailer holding cost. In this chapter, we present a 5-approximation for OWMR with independent warehouse and retailers holding costs under natural monotonicity and subadditivity assumptions. This algorithm is based on a novel primal-dual technique that extends the wave mechanism used for the JRP [90], and answers positively an open question left by Levi *et al.* [91], that asks whether the primal-dual approach could be extended to work with OWMR.

Further, we study the Multilevel JRP, for which the supply chain can be any tree, opposed to the standard version, whose supply chain comprises of one warehouse and several retailers. We observe that the Multilevel JRP is equivalent to the Assembly Problem, that admits a 2-approximation via the primal-dual method [90].

In Chapter 5, circle packing problems are studied. We consider the Circle Bin Packing Problem, whose objective is to pack a set of circles into the minimum number of unit square bins, and the Circle Strip Packing, whose objective is to pack a set of circles into a unit width strip of minimum length. Previously, such problems have been tackled mostly by means of heuristics, and other models. This chapter presents the first approximation algorithms. Indeed, we give an APTAS for the bin packing problem with resource augmentation in one dimension, when one side of the bin has length $1+\gamma$, for some arbitrarily small $\gamma > 0$. As a corollary, we also obtain an APTAS for the Circle Strip Packing Problem. This chapter contains joint work with Flávio K. Miyazawa, Rafael C. S. Schouery, Maxim Sviridenko, and Yoshiko Wakabayashi, and an extended abstract will be presented at ESA 2014 [111].

Chapter 6 summarizes the obtained results, discusses extensions, and lists problems that remain open.

Chapter 2

Upper Bound Factor-Revealing Programs Applied to Facility Location Problems

We consider a generalization of the Metric Facility Location Problem (MFLP), when the distance function is the square of a metric, that is named the Squared Metric Facility Location Problem (SMFLP). A deep investigation on this problem is undertaken, and we give approximations, as well as hardness results. The main result of this chapter, however, is new technique to systematically bound factor-revealing programs [72], that are used in the analysis of primal-dual algorithms. Previously, such programs were bounded through very long proofs, that depended on non-straightforward guessing steps. Applying the same strategy to the SMFLP proved impractical, and thus we developed a new and simpler alternative, which is presented in this chapter.

Problem's definition. Consider finite sets C and F representing *clients* and *facilities*, respectively. For each facility i and client j, let c_{ij} be a nonnegative number representing the cost to connect i to j. Additionally, let f_i be a nonnegative number representing the cost to open facility i. The objective of the Facility Location Problem (FLP) is finding a subset of facilities F' such that $\sum_{i \in F'} f_i + \sum_{j \in C} \min_{i \in F'} c_{ij}$ is minimum. We consider instances whose connection cost function c is the square of a metric. Specifically, a function c is a *squared metric*, if for all facilities i and i' and clients j and j' it holds

$$\sqrt{c_{ij}} \leq \sqrt{c_{ij'}} + \sqrt{c_{i'j'}} + \sqrt{c_{i'j}}.$$

The Squared Metric Facility Location Problem is the particular case of the FLP that only considers instances that satisfy this inequality.

Notice that any metric is also a squared metric, thus any approximation for the SMFLP is also an approximation for the MFLP, and the inapproximability results for the MFLP are also valid for the SMFLP. In general, the FLP is naturally formulated as a mixed integer linear programming. In the next program, a binary variable y_i indicates whether a given facility i is open, and variable x_{ij} indicates whether client j is connected to facility i.

$$\begin{array}{lll} \text{minimize} & \sum_i y_i f_i + \sum_{i \in F} \sum_{j \in C} c_{ij} x_{ij} & \\ \text{subject to} & \sum_{i \in F} x_{ij} = 1 & j \in C, \\ & x_{ij} \leq y_i & i \in F,\ j \in C, \\ & x_{ij} \geq 0 & i \in F,\ j \in C, \\ & y_i \in \{0,1\} & i \in F. \end{array} \tag{2.1}$$

By replacing each constraint $y_i \in \{0,1\}$ by constraint $y_i \geq 0$, for each facility i, one may obtain the so called linear programming relaxation. The dual program of this relaxation is given below.

$$\begin{array}{lll} \text{maximize} & \sum_{j \in C} \alpha_j & \\ \text{subject to} & \alpha_j \leq c_{ij} + \beta_{ij} & i \in F,\ j \in C, \\ & \sum_{j \in C} \beta_{ij} \leq f_i & i \in F, \\ & \alpha_j, \beta_{ij} \geq 0 & i \in F,\ j \in C. \end{array} \tag{2.2}$$

Motivations. Although there are several algorithms for the MFLP in the literature, few works consider the SMFLP. Nevertheless, one may try to solve an instance of the SMFLP using algorithms designed for the metric case. Since these algorithms and their analyses are based on the assumption of the triangle inequality, it is reasonable to expect that they generate good solutions also for squared metric instances. However, there is no trivial way to derive an approximation factor from the MFLP to the SMFLP, so each algorithm must be reanalyzed individually. Several techniques have been used to obtain approximations for the FLP, such as local search [84], LP-rounding [38], and primal-dual approach [72, 76, 104].

The original analysis of some primal-dual algorithms is based on the so called *factor-revealing linear programs* [72, 104]. The value of a computer calculated optimal solution for any such program is a lower bound on the approximation factor. An upper bound, however, is obtained analytically by bounding the value of every factor-revealing program. Here, a technique to obtain a family of *upper bound factor-revealing programs* is introduced, so that the upper bound on the approximation factor is also given by solving a single factor-revealing program. For the SMFLP, these programs have nonlinear constraints with square roots. Since such inequalities are convex, obtaining upper bound factor-revealing programs can also be done in this case.

Summary of results. There are two main contributions in this chapter. First, an important generalization of the MFLP is deeply studied. Approximation factors for the SMFLP are derived by reanalyzing known algorithms for FLP when applied to squared metric instances. We show that the primal-dual algorithms of Jain *et al.* [72], and of Mahdian *et al.* [104], and the LP-rounding algorithm of Chudak and Shmoys [38], that have approximation factor 1.861, 1.61, 1.52, and 1.575 for the MFLP, achieve ratios of 2.87, 2.43, 2.17, and 2.04 for the SMFLP, respectively. The last approximation factor is the best possible, as we show an inapproximability limit of 2.04 for the SMFLP, that extends the hardness result of 1.463 for the metric case by Guha and Khuller [62]. Second, and more importantly, we present a new technique to systematically bound factor-revealing programs. This technique is used in the dual-fitting analysis of the primal-dual algorithms for both the SMFLP and the MFLP.

In Section 2.1, the literature on the FLP and related problems is reviewed. In Section 2.2, we give a lower bound on the approximation of the SMFLP. The upper bound factor-revealing programs are introduced in Section 2.3, that analyzes the first algorithm of Jain *et al.* [72], and are further used in Section 2.4, that analyzes the second algorithm of Jain *et al.* [72], and the algorithm of Mahdian [104]. Experimental results of the obtained UPFRP's are given in Section 2.5. Finally, in Section 2.6, we show that the algorithm of Chudak and Shmoys [38] has the best possible approximation factor for the SMFLP, unless P = NP.

2.1 Literature review

Facility location problems have been studied since the 1960's [12, 85, 105, 125, 126]. Several heuristics and exact algorithms have been proposed, such as Erlenkotter's algorithm [50], that combines a dual heuristic in a branch and bound strategy. Approximations have appeared in the 1980's, and became object of extensive study over the years. Hochbaum [70] presented an $O(\log(n))$-approximation. This factor is best possible for general distance functions, since in this case the set cover can be reduced to the FLP, so it is unlikely to exist a better factor [51, 112]. The research has focused on the metric variant of the FLP, when the distance function satisfies the triangle inequality. For this version, Guha and Khuller [62] proved that no approximation has factor better than 1.463, unless NP $\subseteq$ DTIME$[n^{O(\log \log n)}]$. Sviridenko strengthened this result, by showing that this lower bound holds unless P $\neq$ NP (see [131]).

One of the first constant approximations for the Metric FLP is a 3.16-approximation, given by Shmoys *et al.* [122] in 1997. In 1998, Guha and Khuller [61] obtained a factor of 2.408, and showed that the problem is MaxSNP-hard. Also, Chudak and Shmoys [36] gave a 1.736-approximation for the Metric FLP, and Byrka *et al.* [29] reviewed this analysis

to obtain 1.575-approximation. Koropolu *et al.* [84] showed that a simple local search is a $(5+\varepsilon)$-approximation. In 1999, Charikar and Guha [33] gave a simple greedy method obtaining a 1.853-approximation, and showed an approximation with 1.728 factor using a rounding procedure.

In 2001, Jain and Vazirani [76] obtained a 3-approximation for the Metric FLP based on a primal-dual technique. Yet in 2001, Mahdian *et al.* [101] gave a 1.861-approximation. In 2002, Sviridenko [127] obtained factor 1.582, and Jain *et al.* [73] showed a greedy primal-dual algorithm with 1.61 approximation factor. Mahdian *et al.* [103] gave a 1.52, combining the 1.61-approximation with a greedy augmentation technique. Although near to the approximability limit, this algorithm did not close the gap, as showed by Byrka and Aardal [27], who presented instances for which the algorithm of Mahdian *et al.* obtained an approximation factor of at least 1.494.

Charikar and Guha [33] investigated approximation algorithms using a bi-criteria approximation. In this way, an algorithm for the FLP is a (λ_f, λ_c)-approximation if the obtained solution has total cost of at most $\lambda_f F^* + \lambda_c C^*$, where F^* and C^* denote, respectively, the opening cost and the connection cost of an optimal solution. Jain *et al.* [73] observed that $\lambda_c < 1 + 2e^{-\lambda_f}$, unless NP $\subseteq$ DTIME$[n^{O(\log\log n)}]$, what can be represented by an inapproximability curve. The result may be strengthened by the same arguments of Sviridenko (see [131]). In 2007, Byrka [26] presented the first algorithm that touches the inapproximability curve defined by Jain *et al.*, by combining the LP-rounding algorithm of Chudak and Shmoys [38], that is a $(1.6774, 1.3738)$-approximation, with a primal-dual algorithm of Jain *et al.* [73], that is a $(1.11, 1.7764)$-approximation, thus obtaining a 1.5-approximation for the Metric FLP.

Actually, Byrka *et al.* [28] showed that the algorithm of Chudak and Shmoys [38] is an approximation with bi-factor $(\gamma, \max\{1+2e^{-\gamma}, \frac{e^{-\gamma}+e^{-1}}{1-\frac{1}{\gamma}}\})$, for some parameter $\gamma \geq 1$. Byrka *et al.* [29] suggested that a random distribution on the choice of γ could be used to improve the 1.5-approximation, in opposition to the use of a fixed value $\gamma = 1.6774$. In 2011, Li [93] used a zero-sum game, and answered this conjecture positively, by presenting an explicit distribution for γ, and obtaining a 1.488-approximation for the Metric FLP, that is the best approximation currently known. In Table 2.1, a summary of the approximations for the FLP is presented.

Non-metric distance functions and related works. Many problems are studied under the assumption of an underling metric. Non-metric instances have also been considered in the literature. For example, in k-means, the distance function is defined as the squared Euclidean distance between two given points in the space. This kind of distance function is considered by Jain and Vazirani [76, pp. 292–293], and their approach implies a 9-approximation the SMFLP. To our knowledge, this is the best previously known

Factor	Reference	year	Technique
$O(\log n)$	Hochbaum [70]	1982	greedy augmentation
$5 + \varepsilon$	Koropulo *et al.* [84]	1998	local search
3.16	Shmoys *et al.* [122]	1997	LP-rounding
3	Jain and Vazirani [76]	2001	primal-dual
2.47	Guha and Khuller [62]	1998	LP-rounding + greedy augmentation
1.853	Charikar and Guha [33]	1999	primal-dual + greedy augmentation
1.736	Chudak [36]	1998	LP-rounding
1.728	Charikar and Guha [33]	1999	LP-rounding + primal-dual + greedy augmentation
1.861	Mahdian *et al.* [101]	2001	primal-dual
1.61	Jain *et al.* [73]	2002	primal-dual
1.582	Sviridenko [127]	2002	LP-rounding
1.575	Byrka *et al.* [29]	2010	primal-dual + greedy augmentation
1.52	Mahdian *et al.* [103]	2002	primal-dual + greedy augmentation
1.5	Byrka [26]	2007	LP-rounding + primal-dual
1.488	Li [93]	2011	LP-rounding + primal-dual

Table 2.1: Approximation factors for the MFLP

approximation factor for this problem. The choice of squared metrics discourages excessive distances in the solution. This effect is important in several applications, such as in k-means and in classification problems. Other kinds of distances are also considered. For instance, Charikar *et al.* [34] discussed that their algorithms for k-medians could also be analyzed under distance functions that obey the relaxed triangle inequality. Motivated by the power attribution in wireless networks, de Berg *et al.* [46] considered the TSP for the case that the distance between two given points is the Euclidean distance raised to some power α, and showed a 5-approximation for $\alpha = 2$.

Mahdian and Yan [102] introduced the *strongly factor-revealing linear programs*. A factor-revealing program is similar to a strongly factor-revealing program, however the techniques involved in obtaining such programs are different. To obtain a strongly factor-revealing linear program, one projects a solution of an arbitrarily large linear program into a linear program with a constant number of variables, and guesses how to adjust the restrictions to obtain a feasible solution. In our approach, we define a candidate dual solution for a program with a fixed number of variables, and obtain an UPFRP directly in the form of a minimization program using only straightforward calculations. For the case of the SMFLP, calculating the dual upper bound program is easier and more straightforward than projecting the solutions on the primal. Also, we have considered the case of the MFLP, for which the obtained lower and upper bound factor-revealing programs converge.

2.2 Inapproximability threshold for the SMFLP

As discussed above, there is no (γ_f, γ_c)-approximation for the MFLP, with $\gamma_c < 1+2e^{-\gamma_f}$, unless P = NP. By adapting the results of Guha and Khuller [62], one can obtain analogous lower bounds for the SMFLP, as in the following theorem.

Theorem 2.1. *Let γ_f and γ_c be positive constants with $\gamma_c < 1 + 8e^{-\gamma_f}$. If there is a (γ_f, γ_c)-approximation for the SMFLP, then* P = NP. *In particular, let $\alpha \approx 2.04011$ be the solution of the equation $\gamma = 1 + 8e^{-\gamma}$, then there is no α'-approximation with $\alpha' < \alpha$ for the SMFLP unless* P = NP.

Proof. For simplicity, here we adapt the proof of Guha and Khuller [62] to show that the lower bound holds unless NP $\subseteq$ DTIME$[n^{O(\log\log n)}]$. If we follow the lines of Sviridenko, the condition is changed to unless P = NP (see the lecture notes of Vygen [131, Section 4.4]).

Assume A is a (γ_f, γ_c)-approximation for the SMFLP with $\gamma_c < 1 + 8e^{-\gamma_f}$. Let $\mathcal{J} = (\mathcal{U}, \mathcal{S})$ be an instance of the Set Cover, with $\mathcal{U}$ being a set of elements, $\mathcal{S}$ a collection of subsets of $\mathcal{U}$ and $n = |\mathcal{U}|$. We will derive a $(d' \log n)$-approximation algorithm for the Set Cover problem, for some $d' < 1$. Also, let k be the optimal value of $\mathcal{J}$ for the Set Cover. If k is not known, then one can run the algorithm for $k = 1, \ldots, n$, and output the best solution found.

The algorithm will find a solution for $\mathcal{J}$ by iteratively solving a sequence of instances of the SMFLP of the form $\mathcal{I}^{(j)} = (C^{(j)}, F, c, f^{(j)})$, where $F = \mathcal{S}$ and the initial set $C^{(1)} = \mathcal{U}$. For each element $x_j \in S_i$, set $c_{ij} = 1$, and for each $x_j \notin S_i$, set $c_{ij} = 9$. Note that such c is a squared metric. Let $n_j = |C^{(j)}|$. In the jth instance, every facility cost is $f^{(j)} = \gamma \frac{n_j}{k}$, for some positive γ to be fixed later. For each j, let $S^{(j)}$ denote the solution for $\mathcal{I}^{(j)}$ produced by Algorithm A and let $C^{(j+1)}$ be the elements of $C^{(j)}$ not covered by any set in $S^{(j)}$. This process stops when $C^{(j+1)} = \emptyset$ and yields the solution $S^{(1)} \cup \cdots \cup S^{(j)}$ for $\mathcal{J}$.

Observe that an optimal solution for $\mathcal{J}$ is a solution for each $\mathcal{I}^{(j)}$ with total facility cost $k\, f^{(j)}$ and connection cost one for each of the n_j clients. Therefore, $S^{(j)}$ has cost at most $\gamma_f k f^{(j)} + \gamma_c n_j = (\gamma_f \gamma + \gamma_c) n_j$, because $f^{(j)} = \gamma \frac{n_j}{k}$. Let $\beta_j = |S^{(j)}|/k$ and d_j be such that $d_j n_j$ is the number of elements covered in iteration j, that is, the number of elements of $C^{(j)}$ in the union of the sets in $S^{(j)}$. Then the total facility cost of $S^{(j)}$ is $\beta_j k f^{(j)} = \beta_j \gamma\, n_j$. Moreover, $d_j n_j$ clients are connected with cost one and the other $n_j - d_j n_j = (1 - d_j) n_j$ clients are connected with cost nine. Hence the total cost of $S^{(j)}$ is $\beta_j \gamma n_j + d_j n_j + 9(1-d_j) n_j = (\beta_j \gamma + 9 - 8 d_j) n_j$. We conclude that $\gamma_f \gamma + \gamma_c \geq \beta_j \gamma + 9 - 8 d_j$. So we have that $\gamma_c \geq (\beta_j - \gamma_f)\gamma + 9 - 8 d_j$.

Let $d < 1$ be such that $1 + 8e^{-\gamma_f/d} > \gamma_c$. Suppose, for the sake of contradiction, that $d_j \leq 1 - e^{-\beta_j/d}$ for some j. Then

$$\gamma_c \geq (\beta_j - \gamma_f)\gamma + 9 - 8(1 - e^{-\beta_j/d}).$$

Considering γ_f, γ and d fixed, the minimum value of the right hand side is achieved when $\beta_j = d \log \frac{8}{d\gamma}$. Substituting β_j above, we get

$$\gamma_c \geq (d \log \frac{8}{d\,\gamma} - \gamma_f)\gamma + 1 + d\,\gamma.$$

Considering d and γ_f fixed, we choose the value of γ that maximizes the right hand side, that is, $\gamma = \frac{8}{d} e^{-\frac{\gamma_f}{d}}$. Replacing in the inequality, we obtain $\gamma_c \geq 1 + 8\, e^{-\frac{\gamma_f}{d}} > \gamma_c$, a contradiction. So $d_j > 1 - e^{-\beta_j/d}$ for every j, for this $d < 1$.

Now, we may simply follow the lines of Guha and Khuller [62], one can prove that the algorithm described above for the Set Cover is a $(d' \log n)$-approximation for some $d' < 1$. This implies that NP $\subseteq$ DTIME$[n^{O(\log \log n)}]$. □

2.3 Upper bound factor-revealing programs

We analyze the algorithms of Jain *et al.* [72] using a new systematic factor-revealing technique. For each algorithm, Jain *et al.* [72] analysis uses a family of factor-revealing LP's parameterized by some k. The optimal value z_k of the corresponding LP in the family is such that $\sup_{k\geq 1} z_k$ is the approximation factor of the algorithm. Thus each value z_k is a *lower* bound on the approximation factor and one has to analytically upper bound $\sup_{k\geq 1} z_k$ to obtain an approximation factor. This is a nontrivial analysis, since it is done by guessing a general suboptimal dual solution for the LP, usually inspired by numerically obtained dual LP solutions for small values of k.

In this section, we show how to derive a family of *upper bound factor-revealing programs* (UPFRP) parameterized by some t, so that, for any given t, the optimal value x_t of one such program is an upper bound on $\sup_{k\geq 1} z_k$. Obtaining an UPFRP and solving it using a computer is much simpler and more straightforward than using an analytical proof to obtain the approximation factor, since this does not include a guessing step and a manual verification of the feasibility of the solution. Additionally, as a property of the UPFRP's, we may tighten the obtained factor by solving the LP for larger values of t. In fact, in some cases (see Theorem 2.2 below), the lower and upper bound factor-revealing programs converge, that is, $\sup_{k\geq 1} z_k = \inf_{t\geq 1} x_t$.

We use an UPFRP to show that, when applied to the SMFLP instances, the first algorithm of Jain *et al.* [72], denoted by $A1$, is a 2.87-approximation. For the sake of completeness, the algorithm is described in the following.

*Algorithm A*1 (C, F, c, f) *[72]*

1. Set $U := C$, meaning that every facility starts unopened, and every client unconnected. Each client j has some budget α_j, initially 0, and, at every moment, the budget that an unconnected client j offers to some unopened facility i equals to $\max(\alpha_j - c_{ij}, 0)$.
2. While $U \neq \emptyset$, the budget of each unconnected client is increased continuously until one of the following events occur:
   (a) For some unconnected client j and some open facility i, $\alpha_j = c_{ij}$. In this case, connect client j to facility i an remove j from U.
   (b) For some unopened facility i, $\sum_{j \in U} \max(\alpha_j - c_{ij}, 0) = f_i$. In this case, open facility i and, for every unconnected client j with $\alpha_j \geq c_{ij}$, connect j to i and remove j from U.

The analysis presented by Jain *et al.* [72] uses the dual fitting method. That is, their algorithms produce not only a solution for the MFLP, but also a vector $\alpha = (\alpha_1, \ldots, \alpha_{|C|})$ such that the value of the solution produced is equal to $\sum_j \alpha_j$. Moreover, for the first algorithm, following the dual fitting method, Jain *et al.* [72] proved that the vector $\alpha/1.861$ is a feasible solution for the dual linear program presented as (3) in [72], concluding that the algorithm is a 1.861-approximation for the MFLP. To present a similar analysis for the SMFLP, we use the same definitions and follow the steps of Jain *et al.* analysis. We start by adapting Lemma 3.2 from [72] for a squared metric.

Lemma 2.1. *For every facility i, clients j and j', and vector α obtained by the first algorithm of Jain* et al. *[72] given an instance of the SMFLP, $\sqrt{\alpha_j} \leq \sqrt{\alpha_{j'}} + \sqrt{c_{ij'}} + \sqrt{c_{ij}}$.*

Proof. If $\alpha_j \leq \alpha_{j'}$, the inequality obviously holds. So assume $\alpha_j > \alpha_{j'}$. Let i' be the facility to which the algorithm connects client j'. Thus $\alpha_{j'} \geq c_{i'j'}$ and facility i' is open at time $\alpha_{j'} < \alpha_j$. If $\alpha_j > c_{i'j}$, then client j would have connected to facility i' at some time $t \leq \max(\alpha_{j'}, c_{i'j}) < \alpha_j$, and α_j would have stopped growing then, a contradiction. Hence $\alpha_j \leq c_{i'j}$. Furthermore, by the squared metric constraint, $\sqrt{c_{i'j}} \leq \sqrt{c_{i'j'}} + \sqrt{c_{ij'}} + \sqrt{c_{ij}}$. Therefore $\sqrt{\alpha_j} \leq \sqrt{\alpha_{j'}} + \sqrt{c_{ij'}} + \sqrt{c_{ij}}$. □

A facility i is said to be γ-*overtight* for some positive γ if, at the end of the algorithm,

$$\sum_j \max\left(\frac{\alpha_j}{\gamma} - c_{ij}, 0\right) \;\leq\; f_i. \tag{2.3}$$

Observe that, if every facility is γ-overtight, then the vector α/γ is a feasible solution for the dual linear program presented as (3) in [72]. Jain *et al.* proved that, for the MFLP, every facility is 1.861-overtight. We want to find a γ for the SMFLP, as close to one as possible, for which every facility is γ-overtight.

Fix a facility i. Let us assume without loss of generality that $\alpha_j \geq \gamma\, c_{ij}$ only for the first k clients. Following the lines of Jain *et al.* [72], we want to obtain the so called *(lower bound) factor-revealing* program. We define a set of variables f, d_j, and α_j, corresponding to facility cost f_i, distance c_{ij}, and client contribution α_j. Then, we capture the intrinsic properties of the algorithm using constraints over these variables. We assume without loss of generality that $\alpha_1 \leq \cdots \leq \alpha_k$. Also, we use Lemma 3.3 from [72], that states that the total contribution offered to a facility at any time is at most its cost, that is, $\sum_{l=j}^{k} \max(\alpha_j - d_l, 0) \leq f$. Additionally, we have the inequalities from Lemma 2.1. Subject to all of these constraints, we want to find the minimum γ such that the facility is γ-overtight. In terms of the defined variables, we want the maximum ratio $\sum_{j=1}^{k} \alpha_j / (f + \sum_{j=1}^{k} d_j)$. We obtain the following lower bound factor-revealing program:

$$
\begin{array}{rlll}
z_k^{A1} = & \max & \dfrac{\sum_{j=1}^{k} \alpha_j}{f + \sum_{j=1}^{k} d_j} & \\
& \text{s.t.} & \alpha_j \leq \alpha_{j+1} & 1 \leq j < k, \\
& & \sqrt{\alpha_j} \leq \sqrt{\alpha_l} + \sqrt{d_j} + \sqrt{d_l} & 1 \leq j,\, l \leq k, \\
& & \sum_{l=j}^{k} \max(\alpha_j - d_l, 0) \leq f & 1 \leq j \leq k, \\
& & \alpha_j, d_j, f \geq 0 & 1 \leq j \leq k.
\end{array}
\tag{2.4}
$$

The next lemma has an analogous statement to that of Lemma 3.4 in [72], but it refers to program (2.4). Since the proof is the same, we omit it.

Lemma 2.2. *Let $\gamma = \sup_{k \geq 1} z_k^{A1}$. Every facility is γ-overtight.*

Therefore $\sup_{k \geq 1} z_k^{A1}$ is an upper bound on the approximation factor of the algorithm for the SMFLP. A slight modification of the example presented in Theorem 3.5 of [72] shows that this upper bound is tight (take $c_{ij} = (\sqrt{d_i} + \sqrt{d_j} + \sqrt{\alpha_i})^2$ if $k \geq i \neq j$).

Although the constraints coming from Lemma 2.1 are defined by square roots, they are convex. This is shown in the following.

Lemma 2.3. *Let A, B, C, and D be nonnegative numbers. Then $\sqrt{A} \leq \sqrt{B} + \sqrt{C} + \sqrt{D}$ if and only if $A \leq (1 + \beta + \frac{1}{\gamma})B + (1 + \gamma + \frac{1}{\delta})C + (1 + \delta + \frac{1}{\beta})D$ for every positive numbers β, γ, and δ. In particular, if $\sqrt{A} \leq \sqrt{B} + \sqrt{C} + \sqrt{D}$, then $A \leq 3B + 3C + 3D$.*

Proof. First, suppose $\sqrt{A} \leq \sqrt{B} + \sqrt{C} + \sqrt{D}$. Since $(\sqrt{\beta B} - \sqrt{D/\beta})^2 \geq 0$, we have that $2\sqrt{BD} \leq \beta B + D/\beta$. Similarly, we obtain $2\sqrt{CB} \leq \gamma C + B/\gamma$ and $2\sqrt{DC} \leq \delta D + C/\delta$.

Therefore, if $\sqrt{A} \leq \sqrt{B} + \sqrt{C} + \sqrt{D}$, then

$$\begin{aligned} A &\leq (\sqrt{B} + \sqrt{C} + \sqrt{D})^2 \\ &= B + C + D + 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} \\ &\leq B + C + D + \beta B + D/\beta + \gamma C + B/\gamma + \delta D + C/\delta \\ &= (1 + \beta + \frac{1}{\gamma})B + (1 + \gamma + \frac{1}{\delta})C + (1 + \delta + \frac{1}{\beta})D. \end{aligned}$$

Choosing $\beta = \gamma = \delta = 1$, we obtain $A \leq 3B + 3C + 3D$.

Now suppose $\sqrt{A} > \sqrt{B} + \sqrt{C} + \sqrt{D}$. Let $d > 0$ be such that $A = B + C + D + 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} + d$. Then, $A > (1 + \beta + \frac{1}{\gamma})B + (1 + \gamma + \frac{1}{\delta})C + (1 + \delta + \frac{1}{\beta})D$ is equivalent to $(\beta + \frac{1}{\gamma})B + (\gamma + \frac{1}{\delta})C + (\delta + \frac{1}{\beta})D < 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} + d$. We will analyze the cases in which none, one, two or all numbers B, C and D are zero. Let ξ and ξ' be positive numbers such that $\xi + \xi' < 1$.

Case 1: $B, C, D > 0$. Let $\beta = \sqrt{\frac{D}{B}}$, $\gamma = \sqrt{\frac{B}{C}}$ and $\delta = \sqrt{\frac{C}{D}}$. Then $(\beta + \frac{1}{\gamma})B + (\gamma + \frac{1}{\delta})C + (\delta + \frac{1}{\beta})D = 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} < 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} + d$.

Case 2: $B = 0$ and $C, D > 0$. Let $\beta = \frac{D}{\xi d}$, $\gamma = \frac{\xi' d}{C}$ and $\delta = \sqrt{\frac{C}{D}}$. Then $(\beta + \frac{1}{\gamma})B + (\gamma + \frac{1}{\delta})C + (\delta + \frac{1}{\beta})D = 2\sqrt{DC} + (\xi + \xi')d < 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} + d$.

Case 3: $B, C = 0$ and $D > 0$. Let $\beta = \frac{D}{\xi d}$, $\gamma = 1$ and $\delta = \frac{\xi' d}{D}$. Then $(\beta + \frac{1}{\gamma})B + (\gamma + \frac{1}{\delta})C + (\delta + \frac{1}{\beta})D = (\xi + \xi')d < 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} + d$.

Case 4: $B, C, D = 0$. Let $\beta = 1$, $\gamma = 1$ and $\delta = 1$. Then $(\beta + \frac{1}{\gamma})B + (\gamma + \frac{1}{\delta})C + (\delta + \frac{1}{\beta})D = 0 < 2\sqrt{BD} + 2\sqrt{CB} + 2\sqrt{DC} + d$. □

From Lemmas 2.3 and 2.1, we can derive the following.

Lemma 2.4. *Given an instance of the SMFLP, for every facility i, clients j and j', the vector α produced by the first algorithm of Jain* et al. *[72] is such that, for every positive β, γ, and δ,*

$$\alpha_j \leq (1 + \beta + \frac{1}{\gamma})\alpha_{j'} + (1 + \gamma + \frac{1}{\delta})c_{ij'} + (1 + \delta + \frac{1}{\beta})c_{ij}.$$

Observe that the proof of the Lemma 2.3 is constructive in the sense that, if the given inequality with square roots is not satisfied, then it shows how to determine a linear inequality that is not satisfied. Therefore, the convexity of the constraints of Lemma 2.1 means that program (2.4) can be solved by linear programming packages.

2.3.1 An example: a first analysis

Our first step is to relax (2.4) into a linear program. For that, we adjust the objective function as in [72], and we approximate the inequalities with square roots using inequalities given by Lemma 2.4. For simplicity, here we will use only the inequalities corresponding to $\beta = \gamma = \delta = 1$. With this, we will prove that $\sup_{k\geq 1} z_k^{A1}$ is not greater than 3.236. Later, we will improve the obtained result by using a whole set of inequalities from Lemma 2.4, and using a more standard factor-revealing analysis for the SMFLP. The relaxed lower factor-revealing linear program is:

$$\begin{array}{rlll} \dot{w}_k = & \max & \sum_{j=1}^{k} \alpha_j & \\ & \text{s.t.} & f + \sum_{j=1}^{k} d_j \leq 1 & \\ & & \alpha_j \leq \alpha_{j+1} & 1 \leq j < k \\ & & \alpha_j \leq 3\alpha_l + 3d_j + 3d_l & 1 \leq j,\, l \leq k \\ & & x_{jl} \geq \alpha_j - d_l & 1 \leq j \leq l \leq k \\ & & \sum_{l=j}^{k} x_{jl} \leq f & 1 \leq j \leq k \\ & & \alpha_j, d_j, f, x_{jl} \geq 0 & 1 \leq j \leq l \leq k. \end{array} \tag{2.5}$$

As (2.5) is a relaxation of (2.4), we have that $z_k^{A1} \leq \dot{w}_k$ and thus an upper bound on $\sup_{k\geq 1} \dot{w}_k$ is also an upper bound on $\sup_{k\geq 1} z_k^{A1}$. Solving linear program (2.5) using CPLEX for $k = 540$, we obtain the next lemma.

Lemma 2.5. $\sup_{k\geq 1} \dot{w}_k \geq 3.220$.

To obtain an upper bound on their factor-revealing linear program, Jain *et al.* [72] presented a general dual solution of a relaxed version of the lower bound factor-revealing linear program. This solution is deduced from computational experiments and empirical results for small values of k. In their analysis, they guessed step functions over the indices of a set of dual variables, and used a long verification to show that the value of such solution was not greater than 1.861. For the squared metric case, if we use step functions for the dual variables, the bound on the factor would be as bad as 3.625. One can improve the obtained factor to 3.512 by guessing a piecewise function whose pieces are either constants or hyperboles.

Instead of looking for a good general dual solution, we use an alternative analysis and derive a linear minimization program from (2.5) whose feasible solutions are upper bounds on $\sup_{k\geq 1} \dot{w}_k$. Afterwards, we give an upper bound on the approximation factor by presenting a feasible solution for this program of value less than 3.236.

The idea is to determine a conical combination of the inequalities of (2.5) that imply inequality (2.3) for a γ as small as possible. The linear minimization program will help us to choose the coefficients of such conical combination.

First, rewrite the third inequality of program (2.5), so that the right-hand side is zero. For each j and l, we multiply the corresponding inequality by φ_{jl}. Denote by A the sum of all these inequalities, that is,

$$\sum_{j=1}^{k}\sum_{l=1}^{k}\varphi_{jl}(\alpha_j - 3\alpha_l - 3d_l - 3d_j) \leq 0.$$

The fourth and fifth inequalities of program (2.5) can be relaxed to the set of inequalities $\sum_{i=j}^{l}(\alpha_j - d_i) \leq f$, one for each l such that $j \leq l \leq k$. For each j and l, we multiply the corresponding inequality by θ_{jl} and denote by B the inequality resulting of summing them up, that is,

$$\sum_{j=1}^{k}\sum_{l=j}^{k}\theta_{jl}\sum_{i=j}^{l}(\alpha_j - d_i) \leq \left(\sum_{j=1}^{k}\sum_{l=j}^{k}\theta_{jl}\right) f.$$

The coefficients of α_j in A and B are, respectively,

$$\text{coeff}_A[\alpha_j] = \sum_{l=1}^{k}(\varphi_{jl} - 3\varphi_{lj}) \qquad \text{and} \qquad \text{coeff}_B[\alpha_j] = \sum_{l=j}^{k}(l - j + 1)\theta_{jl},$$

and the coefficients of $-d_j$ in A and B are, respectively,

$$\text{coeff}_A[-d_j] = \sum_{l=1}^{k}3(\varphi_{jl} + \varphi_{lj}) \qquad \text{and} \qquad \text{coeff}_B[-d_j] = \sum_{i=1}^{j}\sum_{l=j}^{k}\theta_{il}.$$

Now, we sum inequalities A and B and obtain a new inequality C:

$$\sum_{j=1}^{k}\text{coeff}_C[\alpha_j]\ \alpha_j - \sum_{j=1}^{k}\text{coeff}_C[-d_j]\ d_j \leq \text{coeff}_C[f]\ f. \tag{2.6}$$

We want to find values for γ, θ_{jl}, and φ_{jl} so that the corresponding coefficients of C are such that inequality (2.6) implies that

$$\sum_{j=1}^{k}\alpha_j - \gamma\sum_{j=1}^{k}d_j \leq \gamma f. \tag{2.7}$$

Moreover, we want γ as small as possible. To obtain inequality (2.7) from inequality (2.6), it is enough that, for each j, coefficient $\text{coeff}_C[\alpha_j] \geq 1$, $\text{coeff}_C[-d_j] \leq \gamma$, and $\text{coeff}_C[f] \leq \gamma$. Hence, this can be expressed by the following linear program.

$$\begin{array}{rcll} y_k = & \min & \gamma & \\ & \text{s.t.} & \text{coeff}_C[\alpha_j] \geq 1 & 1 \leq j \leq k \\ & & \text{coeff}_C[-d_j] \leq \gamma & 1 \leq j \leq k \\ & & \text{coeff}_C[f] \leq \gamma & \\ & & \varphi_{jl} \geq 0 & 1 \leq j, l \leq k \\ & & \theta_{jl} \geq 0 & 1 \leq j \leq l \leq k. \end{array} \tag{2.8}$$

The interested reader may observe that program (2.8) is the dual of a relaxed version of the lower bound factor-revealing linear program (2.5). Therefore, its optimal value is an upper bound on the optimal value of (2.5), that is, $\dot{w}_k \leq y_k$ for every k.

Lemma 2.6. $\sup_{k\geq 1} \dot{w}_k \leq 3.236$.

Proof. We start by observing that $\sup_{k\geq 1} \dot{w}_k$ does not decrease if we restrict attention to values of k that are multiples of a fixed positive integer t. Indeed, for an arbitrary positive integer p, by making t replicas of a solution of (2.5) for $k = p$, and scaling the variables by $1/t$, we obtain a solution of (2.5) for $k = pt$, that is, we deduce that $\dot{w}_p \leq \dot{w}_{pt}$. So we may assume that k has the form $k = pt$ with p and t positive integers, and our goal is to prove that $\dot{w}_k \leq 3.236$.

We will use program (2.8) to obtain a tight upper bound on $\dot{w}_k$. The size of this program however depends on k, which can be arbitrarily large. So we will use a scaling argument to create another linear minimization program with a fixed number (depending only on t) of variables, and obtain a feasible solution for program (2.8) from a solution for this smaller program. Then, we will show that the value of the generated solution for (2.8) is bounded by the value of the small solution.

Consider variables $\gamma' \in \mathbb{R}_+$, $\varphi'_{jl} \in \mathbb{R}_+$ for $1 \leq j, l \leq t$, and $\theta'_{jl} \in \mathbb{R}_+$ for $1 \leq j \leq l \leq t$. For simplicity of notation, we introduce the *hat* operator as follows: for an integer n, define $\hat{n} := \lceil \frac{n}{p} \rceil$. We will obtain a candidate solution for program (2.8) by taking

$$\varphi_{jl} = \frac{\varphi'_{\hat{j}\hat{l}}}{p}, \qquad \theta_{jl} = \frac{\theta'_{\hat{j}\hat{l}}}{p^2}, \qquad \text{and} \qquad \gamma = \gamma'. \tag{2.9}$$

Let us calculate each coefficient of C (inequality (2.6)) for this solution.

$$\begin{aligned}
\text{coeff}_C[\alpha_j] &= \sum_{l=1}^{k}(\varphi_{jl} - 3\varphi_{lj}) + \sum_{l=j}^{k}(l-j+1)\theta_{jl} \\
&= \sum_{l=1}^{k}(\frac{\varphi'_{\hat{j}\hat{l}}}{p} - 3\frac{\varphi'_{\hat{l}\hat{j}}}{p}) + \sum_{l=j}^{k}(l-j+1)\frac{\theta'_{\hat{j}\hat{l}}}{p^2} \\
&\geq \sum_{l=1}^{pt}(\frac{\varphi'_{\hat{j}\hat{l}}}{p} - 3\frac{\varphi'_{\hat{l}\hat{j}}}{p}) + \sum_{l=p\,\hat{j}+1}^{pt}(l - p\,\hat{j})\frac{\theta'_{\hat{j}\hat{l}}}{p^2} \\
&= \sum_{l'=1}^{t} p(\frac{\varphi'_{\hat{j}l'}}{p} - 3\frac{\varphi'_{l'\hat{j}}}{p}) + \sum_{l'=\hat{j}+1}^{t}\frac{\theta'_{\hat{j}l'}}{p^2}\sum_{i=0}^{p-1}(p\,l' - i - p\,\hat{j}) \\
&= \sum_{l'=1}^{t}(\varphi'_{\hat{j}l'} - 3\varphi'_{l'\hat{j}}) + \sum_{l'=\hat{j}+1}^{t}\frac{\theta'_{\hat{j}l'}}{p^2}(p^2\,l' - \frac{p(p-1)}{2} - p^2\,\hat{j}) \\
&\geq \sum_{l'=1}^{t}(\varphi'_{\hat{j}l'} - 3\varphi'_{l'\hat{j}}) + \sum_{l'=\hat{j}+1}^{t}(l' - \hat{j} - \frac{1}{2})\theta'_{\hat{j}l'}.
\end{aligned}$$

$$\begin{aligned}
\text{coeff}_C[-d_j] &= \sum_{l=1}^{k} 3(\varphi_{jl} + \varphi_{lj}) + \sum_{i=1}^{j}\sum_{l=j}^{k} \theta_{il} \\
&= \sum_{l=1}^{pt} 3(\frac{\varphi'_{\hat{j}\hat{l}}}{p} + \frac{\varphi'_{\hat{l}\hat{j}}}{p}) + \sum_{i=1}^{j}\sum_{l=j}^{pt} \frac{\theta'_{\hat{i}\hat{l}}}{p^2} \\
&\leq \sum_{l'=1}^{t} p \cdot 3(\frac{\varphi'_{\hat{j}l'}}{p} + \frac{\varphi'_{l'\hat{j}}}{p}) + \sum_{i'=1}^{\hat{j}} p \cdot \sum_{l'=\hat{j}}^{t} p \cdot \frac{\theta'_{i'l'}}{p^2} \\
&= \sum_{l'=1}^{t} 3(\varphi'_{\hat{j}l'} + \varphi'_{l'\hat{j}}) + \sum_{i'=1}^{\hat{j}}\sum_{l'=\hat{j}}^{t} \theta'_{i'l'}.
\end{aligned}$$

$$\text{coeff}_C[f] = \sum_{j=1}^{k}\sum_{l=j}^{k} \theta_{jl} = \sum_{j=1}^{pt}\sum_{l=j}^{pt} \frac{\theta'_{\hat{j}\hat{l}}}{p^2} \leq \sum_{j'=1}^{t} p \cdot \sum_{l'=\hat{j}}^{t} p \cdot \frac{\theta'_{j'l'}}{p^2} = \sum_{j'=1}^{t}\sum_{l'=\hat{j}}^{t} \theta'_{j'l'}.$$

Now, we want to find the minimum value of γ' and values for φ'_{jl} and θ'_{jl} such that the candidate solution for program (2.8) is feasible. We may define the following linear program, that is the *upper bound factor-revealing program*.

$$\begin{array}{lll}
\dot{x}_t = \min & \gamma' & \\
\text{s.t.} & \sum_{l=1}^{t}(\varphi'_{jl} - 3\varphi'_{lj}) + \sum_{l=j+1}^{t}(l - j - \frac{1}{2})\theta_{jl} \geq 1 & 1 \leq j \leq t \\
& \sum_{l=1}^{t} 3(\varphi'_{jl} + \varphi'_{lj}) + \sum_{i=1}^{j}\sum_{l=j}^{t} \theta'_{il} \leq \gamma' & 1 \leq j \leq t \\
& \sum_{j=1}^{t}\sum_{l=j}^{t} \theta'_{jl} \leq \gamma' & \\
& \varphi'_{jl} \geq 0 & 1 \leq j, l \leq t \\
& \theta'_{jl} \geq 0 & 1 \leq j \leq l \leq t.
\end{array} \tag{2.10}$$

Consider an optimal solution for program (2.10) given by variable θ', and vectors φ', γ', and the corresponding generated solution for program (2.8), given by variable γ, and vectors θ, φ. Replacing γ, θ, φ in (2.6), we obtain $\sum_{j=1}^{k} \alpha_j - \gamma \sum_{j=1}^{k} d_j \leq \gamma f$, and thus $\dot{w}_k = \sum_{j=1}^{k} \alpha_j \leq \gamma \left(\sum_{j=1}^{k} d_j + f\right) \leq \gamma$. Since $\gamma = \gamma' = \dot{x}_t$, we conclude that $\dot{w}_k \leq \dot{x}_t$, and that holds for every positive integer k.

Using CPLEX to solve program (2.10), we obtained $\dot{x}_{800} \approx 3.23586 < 3.236$, and this concludes the proof of Lemma 2.6. □

2.3.2 General technique: an improved analysis

In Lemma 2.6, we obtained the minimization program (2.10) from a conical combination of constraints from program (2.5) that bounds the approximation factor. This process is

similar to obtaining the dual and using a scaling argument. Indeed, we propose a general systematic way to obtain an UPFRP.

Consider the dual program of a traditional maximization factor-revealing linear program for some k. Take k in the form $k = pt$, for a fixed t. We want to create a minimization program that mimics the dual, but depends only on t and bounds the dual optimal value for every k. The idea is to constrain the variables of the small program to obtain a feasible solution for the dual program. To obtain a linear program independent of k, we scale the variables by p. For the sake of notation, the variables of the UPFRP will be called *block* variables, and they will be decorated with the *prime* symbol. The strategy to obtain an UPFRP may be summarized as follows:

1. obtain the dual $P(k)$ of the lower bound factor-revealing linear program;
2. consider a block variable x'_i for variables $x_{(i-1)p+1}, \ldots, x_{(i-1)p+p}$ of $P(k)$;
3. identify each variable x_i with the block variable $x'_{\lceil i/p \rceil}$ scaled by p;
4. replace variables of $P(k)$ by corresponding block variables, canceling factors p.

Denote the resulting program by $P'(t)$. If $P'(t)$ depends only on t, both in number of variables and constraints, then any feasible solution of $P'(t)$ is an upper bound on the solution of $P(pt)$ for every p. Also, if it is the case that the value of $P(k)$ is not greater than the value of $P(kt)$, for every t, then a solution of $P'(t)$ for any t is also a bound on the approximation factor. Therefore, we call $P'(t)$ an *upper bound factor-revealing program*.

Although program (2.4) is nonlinear, we can still use the presented strategy. If the nonlinear constraint is convex, we can approximate it by using a set of linear inequalities, and calculate the dual normally. In order to derive a better upper bound factor-revealing linear program, this time we will use a whole set of linear inequalities. Consider m tuples $(\beta_i, \gamma_i, \delta_i)$ of positive real numbers and $B_i = 1 + \beta_i + \frac{1}{\gamma_i}$, $C_i = 1 + \gamma_i + \frac{1}{\delta_i}$, $D_i = 1 + \delta_i + \frac{1}{\beta_i}$ for $1 \leq i \leq m$. Using Lemma 2.4, we insert inequalities corresponding to the given tuples, replacing the nonlinear constraint, and obtain that $z_k^{A1} \leq w_k^{A1}$, where w_k^{A1} is given by

$$
\begin{array}{lllll}
w_k^{A1} = & \max & \sum_{j=1}^{k} \alpha_j & & \\
& \text{s.t.} & f + \sum_{j=1}^{k} d_j \leq 1 & & \\
& & \alpha_j \leq \alpha_{j+1} & 1 \leq j < k & \\
& & \alpha_j \leq B_i \alpha_l + C_i d_j + D_i d_l & 1 \leq j, l \leq k, \quad 1 \leq i \leq m, & (2.11) \\
& & x_{jl} \geq \alpha_j - d_l & 1 \leq j \leq l \leq k & \\
& & \sum_{l=j}^{k} x_{jl} \leq f & 1 \leq j \leq k & \\
& & \alpha_j, d_j, f, x_{jl} \geq 0 & 1 \leq j \leq l \leq k. &
\end{array}
$$

The following lemma gives a lower bound on the approximation factor of the algorithm for the SMFLP using a cutting plane insertion strategy.

Lemma 2.7. $\sup_{k\geq 1} z_k^{A1} \geq 2.86$.

Proof. Although program (2.4) contains nonlinear constraints, we may use linear program packages to solve it. We start by solving program (2.11) with a fixed number of inequalities. Then, we employ a cutting plane insertion strategy: if the obtained solution violates some inequality with square roots of (2.4), we derive a cutting plane using Lemma 2.4, and resolve the linear program with this additional constraint. Using CPLEX with the cutting plane strategy, we obtained $z_{700}^{A1} \approx 2.86099 > 2.86$. □

Now, we can bound the approximation factor of the algorithm using an UPFRP.

Lemma 2.8. $\sup_{k\geq 1} z_k^{A1} \leq 2.87$.

Proof. It is easy to see that, for program (2.11), as in the proof of Lemma 2.6, we can restrict attention to values of k that are multiples of a fixed positive integer t, that is, $z_k^{A1} \leq w_{kt}^{A1}$, for every positive integer t. So we assume that k has the form $k = pt$, with p and t positive integers. The dual of the linear program (2.11) is

$$
\begin{array}{lll}
w_k^{A1} = \min & \gamma & \\
\text{s.t.} & a_j - a_{j-1} + \sum_{i=1}^{m}\sum_{l=1}^{k} c_{jli} - \sum_{i=1}^{m} B_i \sum_{l=1}^{k} c_{lji} + \sum_{l=j}^{k} e_{jl} \geq 1 & 1 \leq j \leq k, \\
& \sum_{i=1}^{m} C_i \sum_{l=1}^{k} c_{jli} + \sum_{i=1}^{m} D_i \sum_{l=1}^{k} c_{lji} + \sum_{l=1}^{j} e_{lj} \leq \gamma & 1 \leq j \leq k, \\
& \sum_{j=1}^{k} h_j \leq \gamma & \\
& e_{jl} \leq h_j & 1 \leq j \leq l \leq k, \\
& a_0 = a_k = 0, a_j, h_j, e_{jl}, c_{jli}, \gamma \geq 0 & \begin{array}{l} 1 \leq j, l \leq k, \\ 1 \leq i \leq m. \end{array}
\end{array}
\tag{2.12}
$$

We can derive the UPFRP. We would like to define variables as in equation (2.9). Just using a scale factor is not sufficient to preserve the variables a_j in program (2.12). The variables a_j correspond to the ordering restrictions of primal variables α_j in program (2.11), and computational experiments have indicated that removing such restrictions does not change the optimal value significantly, for large values of k. So, we could just set $a_j = 0$ for all j. However, we want to preserve such restrictions, as they will shortly be needed to prove Lemma 2.10. To do this, we can simply interpolate the variables of the UPFRP to obtain the variables of the lower bound program. Again, we group sets of variables based on their indices. For that, we denote the group of a variable of index n as $\hat{n}$. We define

$\hat{n} := \lceil \frac{n}{p} \rceil$ and consider block variables $\gamma', a'_j, c'_{jli}, e'_{jli}, h'_j$. We obtain a candidate solution for program (2.12) by defining

$$\gamma = \gamma',\ a_j = pa'_{\hat{j}} - (p\,\hat{j} - j)(a'_{\hat{j}} - a'_{\hat{j}-1}),\ c_{jli} = \frac{c'_{\hat{j}\hat{l}i}}{p},\ e_{jl} = \frac{e'_{\hat{j}\hat{l}}}{p},\ \text{and } h_j = \frac{h'_{\hat{j}}}{p}. \tag{2.13}$$

In the following, we will use definition (2.13) to obtain a candidate solution for program (2.12) from a small set of block variables. Then, for each constraint of program (2.12), we obtain the expression formed by the non-constant terms, and calculate it as a function of the considered variables. Notice that there is an expression for each primal variable of program (2.11). These expressions are analogous to the primal variables coefficients used in Lemma 2.6, thus, for each primal variable x, we say that this is the *coefficient expression* for x, and we will denote it by coeff$[x]$.

Now we create the minimization UPFRP. The objective value is obtained by applying definition (2.13) to the objective value of program (2.12). Then, for each group of coefficient expressions that has the same value, we include a constraint in the upper bound program that bounds the expression by the independent term. Notice that each UPFRP constraint may correspond to an arbitrarily large number of constraints of the factor-revealing linear program. In the following, we calculate and bound each coefficient expression.

First notice that $a_j - a_{j-1} = a'_{\hat{j}} - a'_{\hat{j}-1}$. To see this, it is enough to use definition (2.13) and consider the cases $\hat{j} = \widehat{(j-1)}$, and $\hat{j} = \widehat{(j-1)} + 1$. Now we have:

$$\begin{aligned}
\text{coeff}[\alpha_j] &= a_j - a_{j-1} + \sum_{i=1}^{m}\sum_{l=1}^{k} c_{jli} - \sum_{i=1}^{m} B_i \sum_{l=1}^{k} c_{lji} + \sum_{l=j}^{k} e_{jl} \\
&= a'_{\hat{j}} - a'_{\hat{j}-1} + \sum_{i=1}^{m}\sum_{l=1}^{pt} \frac{c'_{\hat{j}\hat{l}i}}{p} - \sum_{i=1}^{m} B_i \sum_{l=1}^{pt} \frac{c'_{\hat{l}\hat{j}i}}{p} + \sum_{l=j}^{pt} \frac{e'_{\hat{j}\hat{l}}}{p} \\
&\geq a'_{\hat{j}} - a'_{\hat{j}-1} + \sum_{i=1}^{m}\sum_{l'=1}^{t} p\frac{c'_{\hat{j}l'i}}{p} - \sum_{i=1}^{m} B_i \sum_{l'=1}^{t} p\frac{c'_{l'\hat{j}i}}{p} + \sum_{l'=\hat{j}+1}^{t} p\frac{e'_{\hat{j}l'}}{p} \\
&= a'_{\hat{j}} - a'_{\hat{j}-1} + \sum_{i=1}^{m}\sum_{l'=1}^{t} c'_{\hat{j}l'i} - \sum_{i=1}^{m} B_i \sum_{l'=1}^{t} c'_{l'\hat{j}i} + \sum_{l'=\hat{j}+1}^{t} e'_{\hat{j}l'} \geq 1.
\end{aligned}$$

$$\text{coeff}[f] = \gamma - \sum_{j=1}^{k} h_j = \gamma' - \sum_{j=1}^{pt} \frac{h'_{\hat{j}}}{p} = \gamma' - \sum_{j'=1}^{t} p\frac{h'_{j'}}{p} = \gamma' - \sum_{j'=1}^{t} h'_{j'} \geq 0.$$

$$\text{coeff}[x_{jl}] = h_j - e_{jl} = \frac{h'_{\hat{j}}}{p} - \frac{e'_{\hat{j}\hat{l}}}{p} \geq 0.$$

$$\begin{aligned}
\text{coeff}[d_j] &= \gamma - \sum_{i=1}^{m} C_i \sum_{l=1}^{k} c_{jli} - \sum_{i=1}^{m} D_i \sum_{l=1}^{k} c_{lji} - \sum_{l=1}^{j} e_{jl} \\
&= \gamma' - \sum_{i=1}^{m} C_i \sum_{l=1}^{pt} \frac{c'_{\hat{j}\hat{l}i}}{p} - \sum_{i=1}^{m} D_i \sum_{l=1}^{pt} \frac{c'_{\hat{l}\hat{j}i}}{p} - \sum_{l=1}^{j} \frac{e'_{\hat{j}\hat{l}}}{p} \\
&\geq \gamma' - \sum_{i=1}^{m} C_i \sum_{l'=1}^{t} p\frac{c'_{\hat{j}l'i}}{p} - \sum_{i=1}^{m} D_i \sum_{l'=1}^{t} p\frac{c'_{l'\hat{j}i}}{p} - \sum_{l'=1}^{\hat{j}} p\frac{e'_{\hat{j}l'}}{p} \\
&= \gamma' - \sum_{i=1}^{m} C_i \sum_{l'=1}^{t} c'_{\hat{j}l'i} - \sum_{i=1}^{m} D_i \sum_{l'=1}^{t} c'_{l'\hat{j}i} - \sum_{l'=1}^{\hat{j}} e'_{\hat{j}l'} \geq 0.
\end{aligned}$$

We notice that, for each primal variable, the constraint for its coefficient expression is equivalent to the constraint of any other primal variable in the same group. For example, for any pair α_j and α_l such that $\hat{j} = \hat{l}$, we need to add only one constraint to the UPFRP; therefore, we need only t constraints for this kind of primal variable. We remark that the constraint obtained for coeff$[x_{jl}]$ does not depend on p. Conjoining different constraints, and fixing variables a'_1 and a'_t to zero, we obtain:

$$\begin{aligned}
x_t^{A1} = \min \quad & \gamma \\
\max \quad & a_j - a_{j-1} + \sum_{i=1}^{m}\sum_{l=1}^{t} c_{jli} - \sum_{i=1}^{m} B_i \sum_{l=1}^{t} c_{lji} + \sum_{l=j+1}^{t} e_{jl} \geq 1 && 1 \leq j \leq t, \\
& \sum_{i=1}^{m} C_i \sum_{l=1}^{t} c_{jli} + \sum_{i=1}^{m} D_i \sum_{l=1}^{t} c_{lji} + \sum_{l=1}^{j} e_{lj} \leq \gamma && 1 \leq j \leq t, \\
& \sum_{j=1}^{t} h_j \leq \gamma \\
& e_{jl} \leq h_j && 1 \leq j \leq l \leq t, \\
& a_0 = a_t = 0,\ a_j,\ h_j,\ e_{jl},\ c_{jli} \geq 0 && 1 \leq j, l \leq t, \\
& && 1 \leq i \leq m.
\end{aligned}$$

Now, we want to use Lemma 2.4 and choose a set of tuples (β, γ, δ) so that the squared metric is minimally relaxed. To accommodate the premises of Lemma 2.4, we solve the dual of the UPFRP, and use the same strategy of Lemma 2.7. The dual is:

$$\begin{aligned}
x_t^{A1} = \max \quad & \textstyle\sum_{j=1}^{t} \alpha_j \\
\text{s.t.} \quad & f + \textstyle\sum_{j=1}^{t} d_j \leq 1 \\
& \alpha_j \leq \alpha_{j+1} && 1 \leq j < t \\
& \alpha_j \leq B_i\alpha_l + C_i d_j + D_i d_l && 1 \leq j, l \leq t, \quad 1 \leq i \leq m, \\
& x_{jl} \geq \alpha_j - d_l && 1 \leq j < l \leq t \\
& \textstyle\sum_{l=j}^{t} x_{jl} \leq f && 1 \leq j \leq t \\
& \alpha_j, d_j, f, x_{jl} \geq 0 && 1 \leq j \leq l \leq t.
\end{aligned} \tag{2.14}$$

Using the cutting plane strategy with CPLEX, we obtain $x_{700}^{A1} \approx 2.8697 < 2.87$. □

If we apply this analysis for the metric case, we obtain an UPFRP similar to program (2.14). The only difference is that, for the metric case, there are no coefficients B_l, C_l, and D_l. We use this modified linear program to tighten the approximation factor for the metric case.

Lemma 2.9. *For the MFLP, the approximation factor of A*1 *[72] is between* 1.814 *and* 1.816.

Proof. Let $\hat{z}_k^{A1}$ be the optimal value of the lower bound factor-revealing program (5) in [72]. The corresponding UPFRP is:

$$\begin{array}{rlll} \hat{x}_t^{A1} = & \max & \sum_{j=1}^{t} \alpha_j & \\ & \text{s.t.} & f + \sum_{j=1}^{t} d_j \leq 1 & \\ & & \alpha_j \leq \alpha_{j+1} & 1 \leq j < t \\ & & \alpha_j \leq \alpha_l + d_j + d_l & 1 \leq j, l \leq t \\ & & x_{jl} \geq \alpha_j - d_l & 1 \leq j < l \leq t \\ & & \sum_{l=j}^{t} x_{jl} \leq f & 1 \leq j \leq t \\ & & \alpha_j, d_j, f, x_{jl} \geq 0 & 1 \leq j \leq l \leq t. \end{array} \tag{2.15}$$

Numerical computations using CPLEX show that $\hat{z}_{1000}^{A1} \approx 1.81412 > 1.814$, and that $\hat{x}_{1000}^{A1} \approx 1.81584 < 1.816$. □

We notice that the only difference between the upper and lower bound factor-revealing programs is that the UPFRP does not contain the restrictions $\alpha_j - d_j \leq x_{jj}$ for all j. We explore the similarity between these programs to bound the gap between their optimal values. The following lemma is valid for both the metric and squared metric cases.

Lemma 2.10. *Let* z_k^{A1} *be the optimal value of the lower bound factor-revealing program* (2.11) *(program (5) in [72]) and let* $(\boldsymbol{\alpha}, \mathbf{d}, \mathbf{x}, \mathbf{f})$ *be an optimal solution for program* (2.14) *(respectively program* (2.15)*) with cost value* x_k^{A1}*. If* $\varepsilon = \max_j\{\alpha_j - d_j\}$*, then* $z_k^{A1} \geq \frac{1}{1+\varepsilon} x_k^{A1}$.

Proof. First, notice that we may assume $x_{jj} = 0$, for every j without loss of generality. Let $f' = f + \varepsilon$ and x' be such that $x'_{jl} = x_{jl}$ if $j \neq l$, and $x'_{jj} = \max\{0, \alpha_j - d_j\} \geq 0 = x_{jj}$. Observe that $(\boldsymbol{\alpha}, \mathbf{d}, \mathbf{x}', \mathbf{f}')$ has objective value x_k^{A1} and is a feasible solution for the lower bound factor-revealing program (2.11), except that it might violate the first restriction of program (2.11) (program (5) in [72], respectively). Indeed, it might be the case that $1 < f' + \sum_{j=1}^{k} d_j \leq 1 + \varepsilon$. Now, it is enough to multiply each variable by $\frac{1}{1+\varepsilon}$, and obtain a feasible solution. □

From the last lemma, one can see that the upper and lower bound factor-revealing programs yield very close values, as long as the error term $\varepsilon = \max_j\{\alpha_j - d_j\}$ is small.

Experimentally, we know that the error term decreases as the number of variables k increases, and thus it is reasonable to expect that the value of both factor-revealing programs become very close as k tends to infinity. Indeed, for the metric case, it is easy to show that this error vanishes as k goes to infinity and, therefore, the upper bound and the lower bound factor-revealing programs converge to the same value, as k goes to infinity.

Theorem 2.2. *Let $\hat{z}_k^{A1}$ be as in program (5) in [72] and let $\hat{x}_k^{A1}$ be as in program* (2.15)*. Then $\sup_{k\geq 1} \hat{z}_k^{A1} = \inf_{k\geq 1} \hat{x}_k^{A1}$.*

Proof. First notice that, for any dual solution of program (2.15) with parameter k, we may obtain a feasible solution for the same dual program with parameter $2k$ with same value, by simply duplicating the variables of the original solution, in a way similar to definition (2.13). Therefore, since the dual is a minimization program, we may assume that k is arbitrarily large. Consider an optimal solution of program (2.15). We have that $\alpha_j - d_j \leq \alpha_l + d_l$, for every j and l. Let j be such that $\varepsilon = \alpha_j - d_j$ is maximum and add up these inequalities for all l. We get $k\varepsilon = k(\alpha_j - d_j) = \sum_{l=1}^{k}(\alpha_j - d_j) \leq \sum_{l=1}^{k}(\alpha_l + d_l) \leq \hat{x}_k^{A1} + 1 \leq 1.816 + 1$. From Lemmas 2.9 and 2.10, we get that $\hat{x}_k^{A1} \geq \hat{z}_k^{A1} \geq \frac{1}{1+\varepsilon}\hat{x}_k^{A1} \geq \frac{1}{1+2.816/k}\hat{x}_k^{A1}$. Taking the limit as k goes to infinity, we get that $\sup_{k\geq 1} \hat{z}_k^{A1} = \inf_{k\geq 1} \hat{x}_k^{A1}$. □

It would be nice to bound the values of the variables of program (2.14), as this would suffice to show that the factor-revealing programs also converge for the squared metric case. Since the coefficients of the squared triangle inequality involved in program (2.14) are all greater than one, we cannot use the same approach as in Theorem 2.2. Although experiments suggest that the value of variable α_k in an optimal solution decreases as k increases, it does not seem trivial to determine whether α_k vanishes when k goes to infinity.

2.4 Further applications of UPFRP's

2.4.1 Analysis of improved greedy

We analyze the second algorithm of Jain *et al.* [72] for the squared metric case. The algorithm is essentially the same as Algorithm $A1$, but each connected client keeps contributing to unopened facilities. The contribution of a connected client j to an unopened facility i is the budget that the client would save if facility i were opened. The algorithm, that is denoted by $A2$, is described in the following.

Algorithm A2 (C, F, c, f) *[72]*

1. Set $U := C$, meaning that every facility starts unopened, and every client unconnected. Each client j has some budget α_j, initially 0. At every moment, for each unopened facility i, if client j is unconnected, then j offers $\max(\alpha_j - c_{ij}, 0)$ to i, and, if client j is connected to facility i', then j offers $\max(c_{i'j} - c_{ij}, 0)$ to i.
2. While $U \neq \emptyset$, the budget of each unconnected client is increased continuously until one of the following events occur:
   (a) For some unconnected client j and some open facility i, $\alpha_j = c_{ij}$. In this case, connect client j to facility i and remove j from U.
   (b) For some unopened facility i, the total offer i receives from the clients equals the cost f_i of opening i. In this case, open facility i, connect to i each client j with a positive offer to i, and remove each connected client from U.

For the metric case, the approximation factor is 1.61. With a completely analogous reasoning, we obtain the corresponding factor-revealing program (2.16). The variables are the same as in program (2.4). The new variable r_{jl} corresponds to the budget α_j if client j is connected at the same time as client l, or corresponds to the distance from j to the facility to which j is connected just before l is connected.

$$
\begin{array}{rlll}
z_k^{A2} = & \max & \frac{\sum_{j=1}^{k} \alpha_j}{f + \sum_{j=1}^{k} d_j} & \\
& \text{s.t.} & \alpha_j \leq \alpha_{j+1} & 1 \leq j < k \\
& & r_{jl} \geq r_{j,l+1} & 1 \leq j < l < k \\
& & \sqrt{\alpha_l} \leq \sqrt{r_{jl}} + \sqrt{d_l} + \sqrt{d_j} & 1 \leq j < l \leq k \\
& & \sum_{j=1}^{l-1} \max(r_{jl} - d_j, 0) + \sum_{j=l}^{k} \max(\alpha_l - d_j, 0) \leq f & 1 \leq l \leq k \\
& & \alpha_j,\, d_j,\, f,\, r_{j,l} \geq 0 & 1 \leq j \leq l \leq k.
\end{array}
\tag{2.16}
$$

We repeat the previous analysis to give lower and upper bounds on the approximation factor of the second algorithm for the SMFLP.

Lemma 2.11. $2.415 \leq \sup_{k \geq 1} z_k^{A2} \leq 2.425$.

Proof. Consider tuples $(\beta_i, \gamma_i, \delta_i) \in {\mathbb{R}_+^*}^3$ and $B_i = 1+\beta_i+\frac{1}{\gamma_i}$, $C_i = 1+\gamma_i+\frac{1}{\delta_i}$, $D_i = 1+\delta_i+\frac{1}{\beta_i}$ for $1 \leq i \leq m$. Using Lemma 2.3, we insert inequalities corresponding to these tuples,

replacing the nonlinear constraint, and obtain $z_k^{A2} \leq w_k^{A2}$, where w_k^{A2} is given by

$$\begin{array}{rlll} w_k^{A2} = & \max & \sum_{j=1}^{k} \alpha_j & \\ & \text{s.t.} & f + \sum_{j=1}^{k} d_j \leq 1 & \\ & & \alpha_j \leq \alpha_{j+1} & 1 \leq j < k \\ & & r_{jl} \geq r_{j,l+1} & 1 \leq j < l < k \\ & & \alpha_l \leq B_i r_{jl} + C_i d_l + D_i d_j & 1 \leq j < l \leq k, \quad 1 \leq i \leq m \\ & & r_{jl} - d_j \leq x_{jl} & 1 \leq j < l \leq k \\ & & \alpha_l - d_j \leq x_{jl} & 1 \leq l \leq j \leq k \\ & & \sum_{j=1}^{k} x_{jl} \leq f & 1 \leq l \leq k \\ & & \alpha_j, d_j, f, r_{jl} \geq 0 & 1 \leq j \leq l \leq k \\ & & x_{jl} \geq 0 & 1 \leq j, l \leq k. \end{array} \tag{2.17}$$

Now, we calculate the dual of program (2.17) to derive the UPFRP. After that, we calculate its dual program (2.21), in order to use Lemma 2.3, and solve the UPFRP inserting cutting planes. We proceed the same way as done in Lemma 2.8. With similar arguments, we may see that $z_k^{A2} \leq z_{kt}^{A2}$, for any t, and we assume that k has the form $k = pt$, for some integer t. The dual of linear program (2.17) is given in the following.

$$\begin{array}{rlll} w_k^{A2} = & \min & \gamma & \\ & \text{s.t.} & a_l - a_{l-1} + \sum_{i=1}^{m} \sum_{j=1}^{l-1} c_{jli} + \sum_{j=l}^{k} e_{jl} \geq 1 & 1 \leq l \leq k \\ & & \gamma - \sum_{i=1}^{m} C_i \sum_{j=1}^{l-1} c_{jli} - \sum_{i=1}^{m} D_i \sum_{j=l+1}^{k} c_{lji} - \sum_{j=1}^{k} e_{lj} \geq 0 & 1 \leq l \leq k \\ & & \gamma - \sum_{l=1}^{k} h_l \geq 0 & \\ & & b_{j,l-1} - b_{jl} + e_{jl} - \sum_{i=1}^{m} B_i c_{jli} \geq 0 & 1 \leq j < l \leq k \\ & & h_l - e_{jl} \geq 0 & 1 \leq j, l \leq k \\ & & a_0 = a_k = b_{ll} = b_{lk} = 0 & 1 \leq l \leq k \\ & & a_l, h_l, e_{jl} \geq 0 & 1 \leq l, j \leq k \\ & & b_{jl}, c_{jli}, \gamma \geq 0 & 1 \leq j < l \leq k \\ & & & 1 \leq i \leq m. \end{array} \tag{2.18}$$

Now, we may derive the UPFRP. Let $\hat{n} = \lceil \frac{n}{p} \rceil$ and consider block variables γ', a'_l, b'_{jl}, c'_{jli}, e'_{jl}, and h'_l. We obtain a candidate solution for program (2.18) by defining:

$$\begin{array}{c} \gamma = \gamma', \quad a_l = p\, a'_{\hat{l}} - (p\,\hat{l} - l)(a'_{\hat{l}} - a'_{\hat{l}-1}), \quad b_{jl} = b'_{\hat{j},\hat{l}} - \frac{p\,\hat{l}-l}{p}(b'_{\hat{j}\hat{l}} - b'_{\hat{j},\hat{l}-1}), \\ c_{jll} = \frac{c'_{\hat{j}\hat{l}l}}{p}, \qquad e_{jl} = \frac{e'_{\hat{j}\hat{l}}}{p}, \qquad \text{and} \qquad h_l = \frac{h'_{\hat{l}}}{p}. \end{array} \tag{2.19}$$

In the following, we apply definition (2.19) and calculate each coefficient expression for program (2.18). Again, notice that $a_l - a_{l-1} = a'_{\hat{l}} - a'_{\hat{l}-1}$, and that $b_{j,l-1} - b_{jl} = (b'_{\hat{j},\hat{l}-1} - b'_{\hat{j}\hat{l}})/p$. Also, fix variables c'_{lli} at zero.

$$\begin{aligned}
\text{coeff}[\alpha_l] &= a_l - a_{l-1} + \sum_{i=1}^{m}\sum_{j=1}^{l-1} c_{jli} + \sum_{j=l}^{k} e_{jl} \\
&= a'_{\hat{l}} - a'_{\hat{l}-1} + \sum_{i=1}^{m}\sum_{j=1}^{l-1} \frac{c'_{\hat{j}\hat{l}i}}{p} + \sum_{j=l}^{pt} \frac{e'_{\hat{j}\hat{l}}}{p} \\
&\geq a'_{\hat{l}} - a'_{\hat{l}-1} + \sum_{i=1}^{m}\sum_{j'=1}^{\hat{l}-1} p\frac{c'_{j'\hat{l}i}}{p} + \sum_{j'=\hat{l}+1}^{t} p\frac{e'_{j'\hat{l}}}{p} \\
&= a'_{\hat{l}} - a'_{\hat{l}-1} + \sum_{i=1}^{m}\sum_{j'=1}^{\hat{l}-1} c'_{j'\hat{l}i} + \sum_{j'=\hat{l}+1}^{t} e'_{j'\hat{l}} \geq 1.
\end{aligned}$$

$$\begin{aligned}
\text{coeff}[d_l] &= \gamma - \sum_{i=1}^{m}\sum_{j=1}^{l-1} C_i c_{jli} - \sum_{i=1}^{m}\sum_{j=l+1}^{k} D_i c_{lji} - \sum_{j=1}^{k} e_{lj} \\
&= \gamma' - \sum_{i=1}^{m} C_i \sum_{j=1}^{l-1} \frac{c'_{\hat{j}\hat{l}i}}{p} - \sum_{i=1}^{m} D_i \sum_{j=l+1}^{k} \frac{c'_{\hat{l}\hat{j}i}}{p} - \sum_{j=1}^{k} \frac{e'_{\hat{l}\hat{j}}}{p} \\
&\geq \gamma' - \sum_{i=1}^{m} C_i \sum_{j'=1}^{\hat{l}} p\frac{c'_{j'\hat{l}i}}{p} - \sum_{i=1}^{m} D_i \sum_{j'=\hat{l}}^{t} p\frac{c'_{\hat{l}j',i}}{p} - \sum_{j'=1}^{t} p\frac{e'_{\hat{l}j'}}{p} \\
&= \gamma' - \sum_{i=1}^{m} C_i \sum_{j'=1}^{\hat{l}-1} c'_{j'\hat{l}i} - \sum_{i=1}^{m} D_i \sum_{j'=\hat{l}+1}^{t} c'_{\hat{l}j'i} - \sum_{j'=1}^{t} e'_{\hat{l}j'} \geq 0.
\end{aligned}$$

$$\text{coeff}[f] = \gamma - \sum_{l=1}^{k} h_l = \gamma' - \sum_{l=1}^{k} \frac{h'_{\hat{l}}}{p} = \gamma' - \sum_{l'=1}^{t} p \cdot \frac{h'_{l'}}{p} = \gamma' - \sum_{l'=1}^{t} h'_{l'} \geq 0.$$

$$\text{coeff}[r_{j,l}] = b_{j,l-1} - b_{jl} + e_{jl} - \sum_{i=1}^{m} B_i\, c_{jli} = \frac{b'_{\hat{j},\hat{l}-1} - b'_{\hat{j}\hat{l}}}{p} + \frac{e'_{\hat{j}\hat{l}}}{p} - \sum_{i=1}^{m} B_i \frac{c'_{\hat{j}\hat{l}i}}{p} \geq 0.$$

$$\text{coeff}[x_{jl}] = h_l - e_{j,l} = \frac{h'_{\hat{l}}}{p} - \frac{e'_{\hat{j}\hat{l}}}{p} \geq 0.$$

Conjoining all constraints, the obtained UPFRP is:

$$
\begin{array}{rll}
x_t^{A2} = \min & \gamma & \\
\text{s.t.} & a_l - a_{l-1} + \sum_{i=1}^{m}\sum_{j=1}^{l-1} c_{jli} + \sum_{j=l+1}^{t} e_{jl} \geq 1 & 1 \leq l \leq t \\
& \gamma - \sum_{i=1}^{m} C_i \sum_{j=1}^{l-1} c_{jli} - \sum_{i=1}^{m} D_i \sum_{j=l+1}^{t} c_{lji} - \sum_{j=1}^{t} e_{lj} \geq 0 & 1 \leq l \leq t \\
& \gamma - \sum_{l=1}^{t} h_l \geq 0 & \\
& b_{j,l-1} - b_{jl} + e_{jl} - \sum_{i=1}^{m} B_i c_{jli} \geq 0 & 1 \leq j < l \leq t \\
& h_l - e_{jl} \geq 0 & 1 \leq j, l \leq t \\
& a_0 = a_t = b_{ll} = b_{lt} = 0 & 1 \leq l \leq t \\
& a_l, h_l, e_{jl} \geq 0 & 1 \leq l, j \leq t \\
& b_{jl}, c_{jli} \geq 0 & \begin{array}{l} 1 \leq j < l \leq k \\ 1 \leq i \leq m. \end{array}
\end{array}
\tag{2.20}
$$

Finally, calculating the dual of program (2.20), we obtain program (2.21).

$$
\begin{array}{rll}
x_t^{A2} = \max & \sum_{j=1}^{t} \alpha_j & \\
\text{s.t.} & f + \sum_{j=1}^{t} d_j \leq 1 & \\
& \alpha_j \leq \alpha_{j+1} & 1 \leq j < t \\
& r_{jl} \geq r_{j,l+1} & 1 \leq j < l < t \\
& \alpha_l \leq B_i r_{jl} + C_i d_l + D_i d_j & 1 \leq j < l \leq t, 1 \leq i \leq m \\
& r_{jl} - d_j \leq x_{jl} & 1 \leq j < l \leq t \\
& \alpha_l - d_j \leq x_{jl} & 1 \leq l < j \leq t \\
& \sum_{j=1}^{t} x_{jl} \leq f & 1 \leq l \leq t \\
& \alpha_j, d_j, f, r_{jl} \geq 0 & 1 \leq j \leq l \leq t \\
& x_{jl} \geq 0 & 1 \leq j, l \leq t.
\end{array}
\tag{2.21}
$$

Notice that we can replace the forth set of constraints in program (2.21) by constraints with square roots, if we consider an infinite set of tuples $(\beta_i, \gamma_i, \delta_i)$, obtaining an equivalent nonlinear program. This program is exactly program (2.16), except that the fourth constraint is replaced with

$$
\sum_{j=1}^{l-1} \max(r_{jl} - d_j, 0) + \sum_{j=l+1}^{k} \max(\alpha_l - d_j, 0) \leq f.
$$

Let x_k^{A2} be the optimal value of such a program. With CPLEX we get that $z_{500}^{A2} \approx 2.41565 > 2.415$, and that $x_{500}^{A2} \approx 2.42473 < 2.425$. □

Solving the UPFRP obtained for the MFLP for $k = 500$, we may show that the approximation factor of $A2$ [72] is 1.602. The lower bound factor-revealing program and

the maximization UPFRP are essentially the same, except that, in the lower bound factor-revealing program, the second summation of the fourth constraint contain terms of the kind $\max(\alpha_l - d_l, 0)$, that are not present in the UPFRP. Therefore, Lemma 2.10 also holds for such programs. For the metric case, using a similar analysis to that of Theorem 2.2, one can show that the lower and the upper bound factor-revealing programs converge.

Theorem 2.3. *Let $\hat{z}_k^{A2}$ be as in program (25) in [72] and let $\hat{x}_k^{A2}$ be the optimal value of the corresponding UPFRP obtained by removing the terms of the kind* $\max(\alpha_l - d_l, 0)$ *from the fourth restriction. Then* $\sup_{k\geq 1} z_k^{A2} = \inf_{k\geq 1} x_k^{A2}$.

Proof. Recall that program (25) in [72] is similar to (2.16), but does not contain the square roots. Consider a solution of the UPFRP, for a sufficiently large k. Without loss of generality, we assume $f + \sum_{j=1}^{k} d_j = 1$. For a fixed l, the constraint $\sum_{j=1}^{l-1} \max(r_{jl} - d_j, 0) + \sum_{j=l+1}^{k} \max(\alpha_l - d_j, 0) \leq f$ implies that $\sum_{j=1}^{l-1} r_{jl} \leq f + \sum_{j=1}^{l-1} d_j \leq 1$. We consider two cases. First, suppose $l \leq k/2$, then, $(k/2)\cdot\max(\alpha_l - d_l, 0) \leq k/2\alpha_l \leq \sum_{j=1}^{k} \alpha_j \leq 1.62$. Now, suppose $l > k/2$, then, summing the constraint $\alpha_l - d_l \leq r_{jl} + d_j$ for $j = 1, \ldots, l-1$, then $(k/2-1)\cdot\max(\alpha_l - d_l, 0) \leq \sum_{j=1}^{l-1}(r_{jl} + d_j) \leq 1+1$. In either case, $\varepsilon = \max_l\{\alpha_l - d_l\}$ vanishes as k tends to infinity. The theorem follows by arguments similar to Theorem 2.2. □

2.4.2 Combining with scaling and greedy augmentation

Algorithm $A2$ can be analyzed as a bi-factor approximation algorithm. The analysis uses a factor-revealing linear program, and is similar to the previous analysis. Mahdian *et al.* [104] observed that, due to the asymmetry between the approximation guarantee for the opened facilities cost and the connections cost, Algorithm $A2$ may be used to open facilities that are very economical. This gives rise to a two-phase algorithm, denoted here by $A3(\delta)$, based on scaling the cost of facilities by a constant $\delta \geq 1$, and on the greedy augmentation technique introduced by Guha and Khuller [61]. The first phase opens the most economical facilities, and the second phase greedily

Algorithm $A3(\delta)$ (C, F, c, f) [104]

1. *Scaling*:
   (a) Scale the facility costs by a factor δ.
   (b) Run Algorithm $A2$ on the scaled instance.

2. *Greedy augmentation*:
   While there are facilities that, if open, reduce the total cost:
   (a) Compute the gain g_i of opening each unopened facility i.
   (b) Open a facility i that maximizes the ratio $\frac{g_i}{f_i}$.

In [104], a factor-revealing linear program is used to analyze Algorithm $A3(\delta)$ with a somewhat different, but equivalent, greedy augmentation procedure. This was used to balance a bi-factor from Algorithm $A2$ for the MFLP. As noticed by Byrka and Aardal [28], this analysis is not restricted to Algorithm $A2$, and applies to any bi-factor approximation for the FLP. Therefore, since it does not depend on the cost function being a metric, we can use it to balance a bi-factor approximation for the squared metric case. This result is precisely stated as follows.

Lemma 2.12 ([104])**.** *Consider a (γ_f, γ_c)-approximation for the FLP. Then, for every $\delta \geq 1$, Algorithm $A3(\delta)$ is a $(\gamma_f + \log \delta + \varepsilon, 1 + \frac{\gamma_c - 1}{\delta})$-approximation for the FLP, for $\varepsilon > 0$.*

For the metric case, it has been shown that Algorithm $A2$ is a $(1.11, 1.78)$-approximation. This and Lemma 2.12 give a 1.52-approximation for the MFLP. For the SMFLP, we present an analysis based on an UPFRP. Using straightforward calculations, we may obtain the following:

Lemma 2.13. *Let $\gamma_f \geq 1$ be a fixed value and let $\gamma_c = x_k^{A2_c}$, where*

$$
\begin{array}{rlll}
x_k^{A2_c} = & \max & \frac{\sum_{j=1}^{k} \alpha_j - \gamma_f f}{\sum_{j=1}^{k} d_j} & \\
& \text{s.t.} & \alpha_l \leq \alpha_{l+1} & 1 \leq l < k \\
& & r_{jl} \geq r_{j,l+1} & 1 \leq j < l < k \\
& & \sqrt{\alpha_l} \leq \sqrt{r_{jl}} + \sqrt{d_l} + \sqrt{d_j} & 1 \leq j < l \leq k \\
& & \sum_{j=1}^{l-1} \max(r_{jl} - d_j, 0) + \sum_{j=l+1}^{k} \max(\alpha_l - d_j, 0) \leq f & 1 \leq l \leq k \\
& & \alpha_j, d_j, f, r_{jl} \geq 0 & 1 \leq j \leq l \leq k.
\end{array}
\tag{2.22}
$$

Then, if $\gamma_c < \infty$, Algorithm $A2$ is a (γ_f, γ_c)-approximation for the SMFLP.

The only difference between program (2.22) and the corresponding lower bound factor-revealing program is the extra term $\max(\alpha_l - d_l, 0)$ in the lower bound program, which is not in the fourth constraint of program (2.22). Again, having a bound for this term that vanishes as k goes to infinity would be sufficient to show convergence of the upper and lower bound factor-revealing programs.

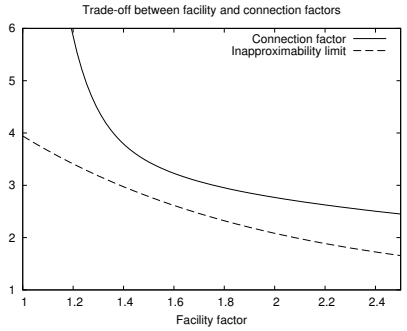
We observe that program (2.22) is unbounded for values of γ_f close to one. This happens also for the corresponding lower bound factor-revealing program. This is in contrast to the factor-revealing programs obtained for the metric case, for which we know that Algorithm $A2$ is a $(1, 2)$-approximation. In this case, the lower bound program is always bounded, but the upper bound program is unbounded for $\gamma_f = 1$, or for values close to one. It would be interesting to strengthen this UPFRP, so that it could also be used in the analysis also for $\gamma_f = 1$.

Theorem 2.4. *Algorithm A3 is a 2.17-approximation for the SMFLP.*

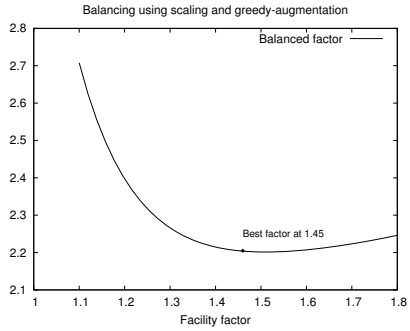
Proof. Consider program (2.22) for $\gamma_f = 1.45$. Numerical computations using CPLEX show that $x_{300}^{A2_c} \approx 3.40339 < 3.4034$. From Lemma 2.13, we know that Algorithm $A2$ is a $(1.45, 3.4034)$-approximation for the SMFLP. Now, by letting $\delta = 2.0543$, and using Lemma 2.12, we obtain that Algorithm $A3$ is a $(2.169\ldots, 2.169\ldots)$-approximation for the SMFLP. □

2.5 Experimental results

In this section, we summarize the results obtained with CPLEX for the analysis of algorithms $A1$, $A2$, and $A3$. In Table 2.2, we present the computational results using CPLEX for the lower bound in the column labeled z_k^{A1} and the upper bound in the column labeled x_k^{A1} on the approximation factor of Algorithm $A1$. In Table 2.3, we present lower and upper bounds on the approximation factor of Algorithm $A2$, that correspond to columns labeled z_k^{A2} and x_k^{A2}, respectively. In Table 2.4, we present computational results for program (2.16) when $\gamma_f = 1.45$, and the approximation factor obtained from Lemma 2.12. We set δ as the value given by the solution of equation $\gamma_f + \log\delta = 1 + \frac{\gamma_c - 1}{\delta}$, that is, $\delta = e^{W_0((\gamma_c-1)e^{\gamma_f - 1})-(\gamma_f-1)}$, where W_0 stands for the Lambert W-function. Figure 2.1a shows the trade-off between connection and facility costs approximation guarantees for the Algorithm $A2$ and the inapproximability curve given by Jain *et al.* [73], and Figure 2.1b shows the trend of obtained factor for Algorithm $A3$ as we vary the value of γ_f, when $k = 50$.



(a) Trade-off between connection and facility approximation factors



(b) Trend of the obtained balanced approximation factors

Figure 2.1: Experimental results

k	z_k^{A1}	x_k^{A1}
10	2.57261	3.18162
20	2.71704	3.01717
50	2.80540	2.92579
100	2.83534	2.89553
200	2.85034	2.88046
300	2.85532	2.87543
400	2.85782	2.87292
500	2.85930	2.87142
600	2.86029	2.87041
700	2.86099	2.86970

Table 2.2: Solutions of the factor-revealing programs for Algorithm $A1$

k	z_k^{A2}	x_k^{A2}
10	2.20702	2.65131
20	2.30987	2.53301
50	2.37551	2.46544
100	2.39773	2.44278
200	2.40894	2.43150
300	2.41267	2.42775
400	2.41453	2.42586
500	2.41565	2.42473

Table 2.3: Solutions of the factor-revealing programs for Algorithm $A2$

k	$x_k^{A2_c}$	best δ	factor
10	4.02931	2.33433	2.29772
20	3.64790	2.16561	2.22270
50	3.48465	2.09159	2.18792
100	3.43524	2.06895	2.17704
200	3.41127	2.05793	2.17170
300	3.40339	2.05430	2.16993

Table 2.4: Solutions of connection factor-revealing programs for Algorithm $A2$, and factors for Algorithm $A3$

2.6 An optimal approximation algorithm

Byrka and Aardal [28] gave a 1.5-approximation for the MFLP, by combining the approximation of Jain *et al.* [73], and a new analysis of the algorithm of Chudak and Shmoys [38], that is based on the rounding of a fraction solution of program (2.1). The algorithm of Chudak and Shmoys, denoted by $CS(\gamma)$, receives parameter $\gamma > 1$, and applies the filtering technique [97] to balance between opening and connection costs. Byrka showed that it achieves the optimal bi-factor approximation $(\gamma, 1 + 2e^{-\gamma})$ for $\gamma \geq 1.6774$. We show that $CS(\gamma)$, when applied to the SMFLP, touches its optimal bi-factor approximation curve $(\gamma, 1+8e^{-\gamma})$ for $\gamma \geq \gamma_0 \approx 2.00492$. Since the solution of $\gamma = 1+8e^{-\gamma}$ is $\alpha = 2.04011 > \gamma_0$, we obtain an (α, α)-approximation for the SMFLP, and thus $CS(\alpha)$, solely used, is an optimal approximation for the SMFLP.

Algorithm $CS(\gamma)$ may be summarized as follows. First, a solution (x^*, y^*) of program (2.1) is obtained. Then, the fractional opening variables y_i^* are scaled by a factor $\gamma \geq 1$, $\overline{y}_i = \gamma\, y_i^*$, and variables $\overline{x}_{ij}$ are defined so that client j is served entirely by its closest facilities, obtaining a new solution $(\overline{x}, \overline{y})$. We may assume that this solution is *complete*, *i.e.* for every client j and facility i, if $\overline{x}_{ij} > 0$, then $\overline{x}_{ij} = \overline{y}_i$, and that for every i, $\overline{y}_i \leq 1$, since, in either case, we can split facility i, and obtain an equivalent instance with these properties. Finally, a clustering of some of the facilities is obtained according to a given criterion, and a probabilistic rounding procedure is used to obtain the final solution. For a detailed description of the algorithm, see [28] (also [29]).

A facility i with $\overline{x}_{ij} > 0$ is called a *close facility* of client j, and the set of such facilities is denoted by C_j. Similarly, a facility i with $\overline{x}_{ij} = 0$ but $x_{ij}^* > 0$ is called a *distant facility* of j, and the set of such facilities is denoted by D_j. Let $F_j = C_j \cup D_j$. The analysis of $CS(\gamma)$ uses the notion of *average distance* between a client $j \in C$ and a subset of facilities $F' \subseteq F$ such that $\sum_{i \in F'} \overline{y}_i > 0$, defined as $d(j, F') = \frac{\sum_{i \in F'} c_{ij} \cdot \overline{y}_i}{\sum_{i \in F'} \overline{y}_i}$. For a client j, we also use some definitions from [29]: the average connection cost, $d_j = d(j, F_j)$; the average distance from close facilities, $d_j^{(c)} = d(j, C_j)$; the average distance from distant facilities, $d_j^{(d)} = d(j, D_j)$; the maximum distance from close facilities, $d_j^{(\max)} = \max_{i \in C_j} c_{ij}$; and the irregularity parameter ρ_j, defined as $\rho_j = (d_j - d_j^{(c)})/d_j$ if $d_j > 0$, and $\rho_j = 0$ otherwise.

With these definitions, we can describe the clustering of the facilities. In each iteration, greedily select a client j, called the *cluster center*, such that the sum $d_j^{(c)} + d_j^{(\max)}$ is minimum, and build a cluster formed by j and its close facilities C_j. Remove j and every other client j' such that $C_j \cap C_{j'}$ is not empty, and repeat this process until every client is removed. The set of facilities opened by $CS(\gamma)$ is given by the following rounding procedure: for each cluster center j, open one facility i from C_j with probability $\overline{x}_{ij} = \overline{y}_i$, and, for each unclustered facility i, open it independently with probability $\overline{y}_i$. Each client is connected to its closest opened facility.

The following lemma of Byrka and Aardal [28] is used to bound the expected connection cost between a client and the closest facility from a set of facilities.

Lemma 2.14 ([28])**.** *Consider a random vector $y \in \{0,1\}^{|F|}$ produced by Algorithm $CS(\gamma)$, a subset $A \subseteq F$ of facilities such that $\sum_{i\in A} \bar{y}_i > 0$, and a client $j \in C$. Then, the following holds:*

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

For a given client j, if one facility in C_j or D_j is opened, then Lemma 2.14 states that the expected connection cost is bounded by $d_j^{(c)}$ and $d_j^{(d)}$, respectively. If no facility in $C_j \cup D_j = F_j$ is opened, then client j can always be connected to one of the close facilities $C_{j'}$ of the associated cluster center j', with expected connection cost $d(j, C_{j'} \setminus F_j)$. Byrka and Aardal [28] showed that, for the MFLP, when $\gamma < 2$, this cost is at most $d_j^{(d)} + d_{j'}^{(\max)} + d_{j'}^{(c)}$. Since for the SMFLP we need $\gamma > 2$, we will use an improved version of this lemma by Li [94]. The adapted lemma for the squared metric is given in the following. The proof is the same, except that we use the squared metric property, instead of the triangle inequality.

Lemma 2.15. *Let j be a client and j' be the associated cluster center. If $C_j \cap C_{j'} \neq \emptyset$, and $C_{j'} \setminus F_j \neq \emptyset$, then we get*

$$d(j, C_{j'} \setminus F_j) \leq 3 \cdot \left((2-\gamma)d_j^{(\max)} + (\gamma-1)d_j^{(d)} + d_{j'}^{(\max)} + d_{j'}^{(c)}\right).$$

Proof. Let $d_{jj'} = \min_{l\in F}(c_{lj} + c_{lj'})$, that is, the minimum connection cost of a path of length two from j to j'.[1] Fix a facility l such that $c_{lj} + c_{lj'} = d_{jj'}$. For each facility i in $C_{j'} \setminus F_j$, we say that a path (j, l, j', i) is the *center-path* to i. The cost of such center-path to i is defined as $d_{jj'} + c_{ij'}$. Notice that, using the squared metric property, $c_{ij} \leq 3(d_{jj'} + c_{ij'})$, and therefore

$$\begin{aligned} d(j, C_{j'} \setminus F_j) &= \frac{\sum_{i\in C_{j'}\setminus F_j} c_{ij} \cdot \overline{y}_i}{\sum_{i\in C_{j'}\setminus F_j} \overline{y}_i} \\ &\leq \frac{\sum_{i\in C_{j'}\setminus F_j} 3(d_{jj'} + c_{ij'}) \cdot \overline{y}_i}{\sum_{i\in C_{j'}\setminus F_j} \overline{y}_i} \\ &= 3 \cdot (d_{jj'} + d(j', C_{j'} \setminus F_j)). \end{aligned}$$

That is, $d(j, C_{j'} \setminus F_j)$ is at most three times the average center-path cost. Following the lines of Li [94, Lemma 12] we know that $d_{jj'} + d(j', C_{j'} \setminus F_j) \leq (2-\gamma)d_j^{(\max)} + (\gamma-1)d_j^{(d)} + d_{j'}^{(\max)} + d_{j'}^{(c)}$. Therefore, the lemma holds. □

[1] In [94], the connection cost c is extended to a distance between j and j', and the triangle inequality is then used to bound this distance with the connection cost of any path of length two. Here, we make a more explicit definition to avoid confusion, since the squared metric property is not sufficient for this purpose.

The next lemma follows from Lemma 2.15, and is straightforward.

Lemma 2.16. $d(j, C_{j'} \setminus F_j) \leq 3\left(\gamma d_j + (3-\gamma)d_j^{(\max)}\right)$.

Now, we can bound the expected facility and connection cost of a solution generated by $CS(\gamma)$. The next theorem is an adapted version of Theorem 2.5 from [29].

Theorem 2.5. *For $3 \geq \gamma \geq 1$, Algorithm $CS(\gamma)$ produces a solution (x, y) for the integer program corresponding to (2.1) with expected facility and connection costs*

$$E[y_i f_i] = \gamma \cdot F_i^*, \quad \textit{and} \quad E\left[\min_{i \in F, y_i = 1} c_{ij}\right] \leq \max\left\{1 + 8e^{-\gamma}, \frac{5e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\right\} \cdot C_j^*,$$

where $F_i^* = y_i^* f_i$ and $C_j^* = \sum_{i \in F} x_{ij}^* c_{ij}$.

Proof. The expected cost of facility i is $E[y_i f_i] = \overline{y}_i f_i = \gamma \cdot y_i^* f_i = \gamma \cdot F_i^*$.

If j is a cluster center, one of its close facilities is open, then the expected connection cost is $d_j^{(c)} \leq d_j = C_j^*$. We may assume that j is not a cluster center. Let p_c be the probability that the closest facility to j is in C_j, and p_d the probability that it is in D_j. If neither case occurs, then, with probability $p_s = 1 - p_c - p_d$, the closest facility is in $C_{j'} \setminus F_j$, where j' is the cluster center associated with j. From the definitions, we have that $d_j^{(c)} = (1 - \rho_j)d_j$, $d_j^{(d)} = (1 + \frac{\rho_j}{\gamma - 1})d_j$, and $\rho_j \leq 1$. Also, from [28], we know that $p_s \leq e^{-\gamma}$ and $p_c \geq 1 - e^{-1}$. Combining these facts with Lemmas 2.14 and 2.16, and using $d_j^{(\max)} \leq d_j^{(d)}$, we obtain

$$\begin{aligned}
E\left[\min_{i \in F, y_i = 1} c_{ij}\right] &\leq p_c \cdot d_j^{(c)} + p_d \cdot d_j^{(d)} + p_s \cdot 3\left(\gamma d_j + (3-\gamma)d_j^{(d)}\right) \\
&= \left((p_c + p_d + 9p_s) + \frac{(p_c + p_d + 9p_s) - (p_c + 3p_s)\gamma}{\gamma - 1}\rho_j\right) d_j \\
&= \left((1 + 8p_s) + \frac{(1 + 8p_s) - (p_c + 3p_s)\gamma}{\gamma - 1}\rho_j\right) d_j \\
&= \left((1 + 8p_s)(1 - \rho_j) + \frac{5p_s + 1 - p_c}{1 - \frac{1}{\gamma}}\rho_j\right) d_j \\
&\leq \left((1 + 8e^{-\gamma})(1 - \rho_j) + \frac{5e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\rho_j\right) d_j \\
&\leq \max\left\{1 + 8e^{-\gamma}, \frac{5e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\right\} \cdot C_j^*.
\end{aligned}$$

□

Let γ_0 be the solution of equation

$$\left(\frac{5e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\right) = \left(1 + 8e^{-\gamma}\right).$$

For γ, with $3 \geq \gamma \geq \gamma_0 \approx 2.00492$, the maximum connection cost factor is $1 + 8e^{-\gamma}$, so $CS(\gamma)$ touches the inapproximability curve $(\gamma, 1 + 8e^{-\gamma})$ of Theorems 2.1. Therefore, the approximation factor is the best possible for the SMFLP, unless P = NP. The next theorem follows immediately.

Theorem 2.6. *Let $\alpha \approx 2.04011$ be the solution of the equation $\gamma = 1+8e^{-\gamma}$. Then $CS(\alpha)$ is an α-approximation for the SMFLP and the approximation factor is the best possible unless* P = NP*.*

2.6.1 The Facility Location Problem with relaxed metrics

We notice that the analysis of Subsection 2.3.1, and that of Lemma 2.15 apply to a more general case of the FLP when the connection cost function satisfies $c_{ij} \leq 3(c_{ij'} + c_{i'j'} + c_{i'j})$ for all facilities i and i', and clients j and j'. We say that a connection cost function c for the FLP satisfies a τ-relaxed triangle inequality if $c_{ij} \leq \tau \cdot (c_{ij'} + c_{i'j'} + c_{i'j})$, for all $i, i' \in F$, and $j, j' \in C$. Also, we say that the subset of the FLP that contains only instances that satisfy the τ-relaxed triangle inequality is the *τ-relaxed FLP*.

The following generalizes Theorems 2.1 to obtain a lower bound on the approximation factor for the τ-relaxed FLP.

Theorem 2.7. *Let γ_f and γ_c be positive constants with $\gamma_c < 1 + (3\tau - 1)e^{-\gamma_f}$. If there is a (γ_f, γ_c)-approximation for the τ-relaxed FLP, then* P = NP*.*

Proof sketch. We use the same reduction to set cover as in Theorem 2.1, but associate elements and sets not connected directly with cost 3τ (note that this connection cost function satisfies the τ-relaxed triangle inequality). The proof remains the same, except that terms 9 become 3τ, and coefficients 8 become $3\tau - 1$. □

To obtain approximations for this variant, we extend Theorem 2.5. First, we need the following result, that is completely analogous to Lemma 2.16.

Lemma 2.17. $d(j, C_{j'} \setminus F_j) \leq \tau \cdot \left(\gamma d_j + (3 - \gamma) d_j^{(\max)}\right)$.

Now, we can obtain approximation for the τ-relaxed FLP.

Theorem 2.8. *For every $\gamma \geq 1$, the algorithm $CS(\gamma)$ obtain a bi-factor approximation of $\left(\gamma, \max\left\{1 + (3\tau - 1)e^{-\gamma}, \frac{(2\tau-1)e^{-\gamma}+e^{-1}}{1-\gamma^{-1}}, \frac{((\gamma-1)\tau-1)e^{-\gamma}+e^{-1}}{1-\gamma^{-1}}\right\}\right)$ for the τ-relaxed FLP.*

Proof. The proof adapts the proof of Theorem 2.5, but we use Lemma 2.17, instead of Lemma 2.16. First, we consider $\gamma \leq 3$, and use $d_j^{(\max)} \leq d_j^{(d)}$, so we obtain

$$
\begin{aligned}
E\left[\min_{i\in F, y_i=1} c_{ij}\right] &\leq p_c \cdot d_j^{(c)} + p_d \cdot d_j^{(d)} + p_s \cdot \tau\left(\gamma d_j + (3-\gamma) d_j^{(d)}\right) \\
&= \left((p_c + p_d + 3\tau p_s) + \frac{(p_c + p_d + 3\tau p_s) - (p_c + \tau p_s)}{\gamma - 1}\rho_j\right) d_j \\
&= \left((1 + (3\tau - 1)p_s) + \frac{(1 + (3\tau - 1)p_s) - (p_c + \tau p_s)\gamma}{\gamma - 1}\rho_j\right) d_j \\
&= \left((1 + (3\tau - 1)p_s)(1 - \rho_j) + \frac{(2\tau - 1)p_s + 1 - p_c}{1 - \frac{1}{\gamma}}\rho_j\right) d_j \\
&\leq \left((1 + (3\tau - 1)e^{-\gamma})(1 - \rho_j) + \frac{(2\tau - 1)e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\rho_j\right) d_j \\
&\leq \max\left\{1 + (3\tau - 1)e^{-\gamma}, \frac{(2\tau - 1)e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\right\} \cdot C_j^*.
\end{aligned}
$$

Now, we consider $\gamma > 3$, and use $d_j^{(\max)} \geq d_j^{(c)}$. Analogously, we obtain

$$
\begin{aligned}
E\left[\min_{i\in F, y_i=1} c_{ij}\right] &\leq p_c \cdot d_j^{(c)} + p_d \cdot d_j^{(d)} + p_s \cdot \tau\left(\gamma d_j + (3-\gamma) d_j^{(c)}\right) \\
&\leq \max\left\{1 + (3\tau - 1)e^{-\gamma}, \frac{((\gamma - 1)\tau - 1)e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\right\} \cdot C_j^*. \qquad \square
\end{aligned}
$$

Let $\alpha(\tau)$ be the solution of equation $\gamma = 1 + (3\tau - 1)e^{-\gamma}$. We will verify that for τ in the interval $(2.62, 31.02)$, Theorem 2.8 implies a $(\alpha(\tau), \alpha(\tau))$-approximation for the τ-relaxed FLP, and thus, for this interval, the algorithm $CS(\alpha(\tau))$ has the best possible approximation, unless P = NP. Let $\beta(\tau)$ be the solution of equation $\gamma = \frac{(2\tau-1)e^{-\gamma}+e^{-1}}{1-\gamma^{-1}}$ (considering $\gamma \leq 3$), and $\delta(\tau)$ be the solution of equation $\gamma = \frac{((\gamma-1)\tau-1)e^{-\gamma}+e^{-1}}{1-\gamma^{-1}}$ (considering $\gamma > 3$). First notice that for a fixed τ we obtain a balance in the bi-factor of the Theorem 2.8 at the at the γ value that is maximum among $\alpha(\tau), \beta(\tau), \delta(\tau)$. It is not hard to show that all $\alpha(\tau)$, $\beta(\tau)$ are $\delta(\tau)$ are increasing continuous functions, and that each pair intersects at a single point (see Figure 2.2).

From Theorem 2.8, we know that $\max\{\alpha(\tau), \beta(\tau), \delta(\tau)\}$ is an approximation for the τ-relaxed FLP. Also, Theorem 2.7 states that there is no approximation better than $\alpha(\tau)$ for the τ-relaxed FLP, unless P = NP. Solving numerically the equations $\alpha(\tau) = \beta(\tau)$ and $\alpha(\tau) = \delta(\tau)$, we obtain that $\alpha(\tau)$ is maximum over the three functions for $\tau \in (2.62, 31.02)$, and so for this interval, the LP-rounding algorithm has the best approximation factor.

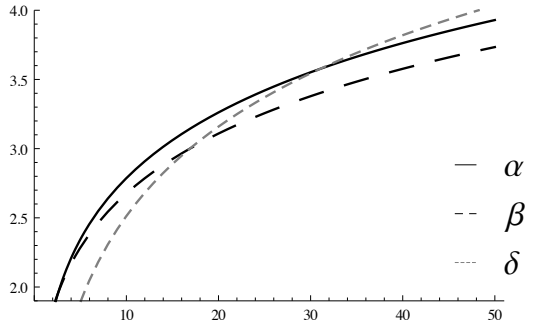


Figure 2.2: Intersections of approximation and inapproximability curves

We say that the FLP with a metric raised to α (M$^\alpha$FLP), is the variant of the FLP that considers instances such that the connection cost function is the α^{th} power of a given metric. We may use the following known fact to derive approximations for the M$^\alpha$FLP using approximations for τ-relaxed FLP's.

Lemma 2.18. *Let A, B, C, and D be nonnegative numbers such that $A \leq B + C + D$, and let $\alpha \geq 1$, then $A^\alpha \leq 3^{\alpha-1}(B^\alpha + C^\alpha + D^\alpha)$.*

This implies that the connection cost function that is the α^{th} power of a metric satisfies the $3^{\alpha-1}$-relaxed triangle inequality, and therefore the M$^\alpha$FLP is a particular case of the $3^{\alpha-1}$-relaxed FLP.

Chapter remarks

The majority of the literature on the FLP considers the metric case. While the assumption of the triangle inequality is natural for many applications, for other applications, more relaxed distance functions are necessary, and so the difficulty of approximating the FLP in these cases remained open. This work addressed this gap, by studying several relaxed distance functions for the FLP. As expected, it was shown that, for the particular case that the distance is the square of a metric, it is NP-hard to obtain an approximation factor better than 2.04, while the corresponding known hardness result for the metric case gives a lower bound of 1.463 [62]. It might be surprising, therefore, that the LP-rounding algorithm of Chudak and Shmoys [38] has the best possible factor under the squared metric, while for the metric case the gap between the best known approximation, with factor of 1.488 [94], and the known approximation lower bound of 1.463 is still open.

For the analyzed primal-dual algorithms [72], the obtained results for the squared metric were more consistent with the results known for the metric case, since the algorithms with best factors for the squared metric case are also the algorithms with best factors

for the metric case. These factors were originally obtained by using the so called factor-revealing programs. Since the non-linear behavior of the non-metric restriction turned such factor-revealing programs much more complex, repeating the analysis employed in the metric case became impractical. Thus, the upper bound factor-revealing programs were introduced, as an alternative to obtain the approximation factor. It is worth noticing that, although the independently obtained strongly factor-reveling programs [102] are similar, in the sense that the approximation factor is given by solving an LP directly, we presented a more systematic approach, that avoids guessing steps, and it indeed proved easier in the analysis of the algorithms for the FLP.

Chapter 3

The Continuous Facility Location Problem

Meira and Miyazawa [108] introduced a variant of the Facility Location Problem (FLP), that is called the Continuous Facility Location Problem (ConFL). In this problem, a facility can be opened at any point of the Euclidean space, while, in the traditional Euclidean FLP, one can only open facilities at points of a given finite set of locations. This is a problem with clustering applications, and is closely related to k-medians and k-means problems. We consider continuous versions of both the Euclidean FLP, and the Squared Euclidean FLP, when the cost to connect a client to a facility is given by the Euclidean distance, and the squared Euclidean distance, respectively.

The ConFL may be interpreted as a penalty clustering problem. In this way, the problem is to find a partition of the clients in any number of clusters with a fixed cost per cluster, whereas the k-medians and k-means problems aim to find a partition of the set of clients in at most k parts. Therefore, ConFL may be used as an alternative to k-clustering problems when the number of clusters is not known, but there is an estimate for the cost of a cluster. Notice that solving k-medians and k-means to the optimality for each value of k is sufficient to solve ConFL under the L_2 and L_2^2 norms, respectively.

Problem's definition. In the ConFL problem, one is given a set of points C of the d-dimensional space, and a positive number f. A facility may be opened in any point of the space at cost f, and a client j is connected to a facility i at cost $c(i, j)$. The goal is to open a set of facilities F' that minimizes

$$|F'|f + \sum_{j \in C} \min_{i \in F'} c(i, j).$$

The ConFL with distance μ is denoted by μ-ConFLP, so that L_2-ConFLP and L_2^2-ConFLP are the variants of ConFL with Euclidean and squared Euclidean distances, respectively.

Summary of results. We reduce the ConFL to the corresponding discrete version of the FLP, by using the so called *center sets*, which are discrete sets points that approximate the centers in a continuous optimal solution. This leads to approximations for ConFL problems, provided that there exists an efficient procedure to obtain these center sets, and approximation algorithms for the corresponding discrete facility location problems. Using this reduction, we obtain approximations with factors $1.488+\varepsilon$ and $2.04+\varepsilon$ for the L_2 and L_2^2 norms, for the case in which the dimension is part of input, and a PTAS for the case of L_2-norm and fixed dimension. Solving the L_2-ConFLP and the L_2^2-ConFLP by using the corresponding discrete k-clustering problems cannot improve these factors, since there is a lower bound on the approximation factor of 1.735 for the Metric k-medians [73], and we show that there is no approximation with factor better than 3.943 for the k-medians problem with squared metric. Moreover, we obtain center sets for the more general L_2^α-norm, that are used to give approximation algorithms for the corresponding ConFL problems, for any $\alpha \geq 1$. Using such center sets, we also obtain a PTAS for the k-clustering problems with L_2^α norms, for fixed k.

In Section 3.1, related works are reviewed. In Section 3.2, a generalized version of the FLP is described, and additional definitions are given. In Section 3.3, we show how to reduce ConFL to its corresponding discrete version, and give approximations for the case of L_2 and L_2^2 dissimilarity measures. In Section 3.4, a procedure to obtain center sets for the distance L_2^α is described, and approximations for the L_2^α-ConFLP are obtained.

3.1 Literature review

A few works give approximations for the ConFL. Meira and Miyazawa [108] presented a $(1.861+\varepsilon)$-approximation for the L_2-ConFLP, and a $(9+\varepsilon)$-approximation for L_2^2-ConFLP, and Czumaj *et al.* [43] presented an approximation scheme for the special case of the ConFL problem for the L_2 distance function in the plane. For the discrete Euclidean FLP, when the set of facility locations is finite, and the connection cost is the Euclidean distance, there is an approximation scheme by Arora *et al.* [7], who extended previous approximation schemes for the TSP and other problems [5, 6]. For the variant whose connection cost is the square of the Euclidean distance, the work of Jain and Vazirani [76] implies a 9-approximation, and the best known factor is a 2.04-approximation for the more general Squared Metric FLP, presented in Chapter 2,

Other clustering problems have been largely investigated. Several algorithms and techniques have been proposed for k-means and k-medians (for both discrete and continuous versions). In particular, some algorithms make use of center sets to obtain approximation schemes. These works are summarized in the following.

The k-means problem. One of the first algorithms for the k-means problem is a local search heuristic proposed by Lloyd for *Pulse Code Modulation* (PCM) in 1957, although this technique was only published in 1982 [98]. The term k-means is commonly used to refer to both the problem and Lloyd's method, and was used for the first time by MacQueens [99] in 1967. In 2008, Dasgupta [44] showed that k-means is NP-hard for the particular case where $k = 2$, and d is part of the input. The same result was obtained independently by Aloise *et al.* [3]. In 2009, Mahajan *et al.* [100] showed that the planar k-means is NP-hard, that is, k-means is also NP-hard for the particular case where $d = 2$, and k is part of the input.

Despite the popularity of Lloyd's algorithm, due to its relative simplicity, and good performance in practice, it does not give any guarantee on the approximation factor, nor on its running time. Indeed, there are examples for which Lloyd's algorithm can return arbitrarily bad solutions. Some works have analyzed this method, and some of its variations, considering both the algorithm running time [8, 67], and the quality of the returned solution [9, 114].

In 2005, Har-Peled and Sadri [67] analyzed the running time of k-means method, and gave instances for the unidimensional case with $k = 2$ for which this algorithm runs for at least $\Omega(n)$ iterations. They also considered the running time of Lloyd's algorithms as a function of the point spread Δ, which is defined as the ratio between the largest and the smallest distance between two given points. Later, Artur and Vassilvitskii [8] showed that, even for randomly selected initial centers, the running time is superpolynomial with high probability. Also, they showed that, if the dimension of the points is not much smaller than the total number of points, that is, if $d = \Omega(n/\log n)$, then the k-means method runs in polynomial time with high probability.

In 2006, Ostrovsky *et al.* [114] showed that, ensuring that initial centers are selected properly, a randomized variation of Lloyd's algorithm has bounded approximation factor. In 2007, Arthur and Vassilvitskii [9] revisited the importance of the good selection of initial centers, and showed that a variation of Lloyd's algorithm, that selects initial centers probabilistically according to the inverse distance from a point to current centers, has expected approximation factor of $O(\log k)$.

In 1993, Hasegawa *et al.* [68] presented one of the first algorithms with bounded approximation factor. They noticed that the set of input points that are closest to the centers of an optimal clustering gives a Voronoi partition with cost of at most twice the optimal value. By enumerating all Voronoi partitions of the n points in k parts, they gave a 2-approximation for k-means with running time $O(n^{k+1})$. In 1994, Inaba *et al.* [71] showed that the optimal partition for k-means is obtained from a set of Voronoi diagrams of size $O(n^{O(kd)})$. Therefore, they solved k-means exactly in time $O(n^{kd+1})$ enumerating such partitions. This is polynomial for the particular case in which k and d are fixed.

Several approximation schemes have been proposed, both for the case of low dimension [66, 71, 106], and for the case of high dimension [35, 52, 54, 86]. In 1994, Inaba *et al.* [71] presented a $(1+\varepsilon)$-approximation for the 2-means problem with running time $O(n\varepsilon^{-d})$. In 2000, Matoušek [106] obtained $(1+\varepsilon)$-approximations for the case of constant d, or constant k, with running times $O(n \log n)$ and $O(n \log^k n)$, respectively. In 2003, Fernandez de La Vega *et al.* [54] showed an approximation scheme for constant k, with time complexity $O(n^{3k})$, that is independent of the dimension.

In 2004, Har-Peled and Mazumdar [66] showed that small subsets that are denominated *coresets*, can be used to obtain good approximations for clustering problems. They used coresets to obtain approximation schemes for k-means and k-medians problems. In the same year, Kumar *et al.* [86] showed an approximation scheme for constant k, with linear running time $O(nd)$. Their algorithm is relatively simple, and is based on small random samples of points.

When both d and k are considered part of the instance, Kanungo *et al.* [78] used a local search to give a $(9+\varepsilon)$-approximation for the discrete version of k-means, when one is given a discrete set of candidate centers C. To obtain a solution to the (continuous) k-means, one may simply use an ε-approximate centroid set as the set of candidate centers. The size of this centroid set may have exponential dependency on d [106], or may be chosen independently of d [71]. Jain and Vazirani [76] gave a 108-approximation using a primal-dual algorithm for the (discrete) Squared Euclidean FLP, using a binary search over the cost of opening a facility, and using a rounding algorithm to obtain exactly k facilities. Meira and Miyazawa [108] showed how such algorithm could be used to directly solve the L_2^2-ConFLP when d is fixed, improving the approximation factor to $54+\varepsilon$.

Table 3.1 summarizes the algorithms described for k-means.

The k-medians problem. The continuous k-medians is analogous to k-means, but uses Euclidean distance as the connection cost function. There are polynomial-time approximation schemes for the case in which k and d are fixed [7, 66, 83], and for the case in which d is part of the input, but k is fixed [11, 35, 86]. When k is part of the input and $d \geq \Omega(\log n)$, it is NP-hard to obtain a PTAS to k-medians [64]. In the discrete version of k-medians, the set of clients and centers are vertices of a graph, and the connection cost between two clients is given by the edge weight. The special case whose graph is a tree can be solved in polynomial time [79]. In 1999, Charikar *et al.* [33] obtained a 6.77-approximation for the Metric k-medians, when the distance function is a metric. Using the so called Lagrangian Multiplier Preserver algorithms for the FLP, Jain e Vazirani [76] obtained a 6-approximation for the Metric k-medians, and Jain *et al.* [73] obtained a 4-approximation. Arya *et al.* [10] gave a local search $(3+\varepsilon)$-approximation, and the best approximation factor currently known is $1+\sqrt{3}+\varepsilon$ by Li and Svensson [95].

Factor	Reference	year	Notes
-	Lloyd [98]	1982	$O(n^{O(k)})$ amortized time [8]
2	Hasegawa *et al.* [68]	1993	Voronoi partitions, $O(n^{k+1})$ time
1	Inaba *et al.* [71]	1994	Voronoi partitions, $O(n^{kd+1})$ time
$1+\varepsilon$	Inaba *et al.* [71]	1994	$O(n\varepsilon^{-d})$ time for $k=2$
$1+\varepsilon$	Matoušek [106]	2000	$O(n\varepsilon^{-2k^2d}\log^k n)$ time
108	Jain and Vazirani [76]	2001	primal-dual
$9+\varepsilon$	Kanungo *et al.* [78]	2002	local search
$1+\varepsilon$	Fernandez de La Vega *et al.* [54]	2003	$O(n^{O(k/\varepsilon^2)})$ time
$1+\varepsilon$	Har-Peled and Mazumdar [66]	2004	$O(n+k^{k+2}\varepsilon^{-(2d+1)k}\log^{k+1} n\log^k \frac{1}{\varepsilon})$ time
$1+\varepsilon$	Kumar *et al.* [86]	2004	$O(2^{(\frac{k}{\varepsilon})^{O(1)}}dn)$ time for small k
$O(\log k)$	Arthur and Vassilvitskii [9]	2007	variant of Lloyd's algorithm
$54+\varepsilon$	Meira and Miyazawa [108]	2008	primal-dual

Table 3.1: Algorithms for k-means

Center sets. To obtain approximation schemes for k-means, Matoušek [106] used the notion of approximate centroid sets. The idea is to obtain a set of candidate centers containing a subset of k elements that is an approximate solution for k-means, and deviates by at most an ε fraction of the optimum. The term centroid set comes from the fact that, for k-means, the optimal center of a cluster is given by the centroid of its points. Analogous sets are used for other k-clustering problems, and thus they are called generically *center sets* in this work. For k-means, there are center sets with size $O(n\varepsilon^{-d}\log(1/\varepsilon))$, given by Matoušek [106], and with size $O(k\varepsilon^{-2d}\log n\log 1/\varepsilon)$, given by Har-Peled and Muzumdar [66]. Approximate center sets of size independent of n and with $O(k^3\varepsilon^{-(2d+2)}\log 1/\varepsilon)$ elements were given by Har-Peled and Kushal [65]. Also, center sets of size independent of d with $O(n^{1/\varepsilon})$ elements are implicitly given by Inaba *et al.* [71]. For k-medians, there are ε-approximate center sets with size $O(k^2\varepsilon^{-2d}\log^2 n)$ by Har-Peled and Muzumdar [66]. Kumar *et al.* [86, 87] gave center sets with size $O(2^{(1/\varepsilon)^{O(1)}})$ independent of d with constant probability, which implies center sets for k-medians of size $O(n^{(1/\varepsilon)^{O(1)}})$.

Kumar *et al.* [86, 87] formalized the existence of a *random sampling procedure* for certain k-clustering problems, and used a superset sampling algorithm to obtain linear approximation algorithms for k-means and k-medians. The idea is that, for certain clustering problems, one may obtain, with constant probability, a center set for a given set of points using only a constant sized subset of these points. Ackermann *et al.* [1] extended the algorithm of Kumar *et al.*, using a combinatorial analysis, and avoiding the need of the triangle inequality, therefore obtaining linear approximation schemes for k-clustering problems with different distance functions, such as Kullback-Leibler divergence and Mahalanobis distances.

3.2 Definitions

The following definition generalizes the continuous and the discrete facility location problems, using arbitrary distance functions. Also, this generalizes other versions of the FLP, such as the soft-capacitated, and the concave cost facility location problems.

Definition 3.1. *In the* Generalized FLP*, we are given a finite set C of clients, and a (possibly infinite) set F of facilities. We are given a dissimilarity measure $c : F \times C \to \mathbb{R}_+$, and a facility opening cost function $f : F \times \mathbb{N} \to \mathbb{R}_+$. The cost to serve a given client $j \in C$ using a facility i is given by $c(i, j)$, and the cost to open a facility $i \in F$ if it serves r clients is $f(i, r)$. We require that $f(i, 0) = 0$ for each facility $i \in F$. We want to obtain a function $\phi : C \to F$ that associates each client to a certain facility. The objective is to minimize the sum of opening and connection costs, $\sum_{i \in F} f(i, r(i)) + \sum_{j \in C} c(\phi(j), j)$, where $r(i) = |\{j : \phi(j) = i\}|$.*

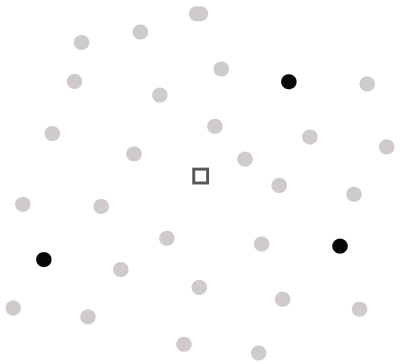
We say that a facility location problem is discrete if F is finite. In this case, we assume an oracle distance function c. If F is not finite, then we assume that the distance function may be calculated in constant computational time. For example, irrational distances $c(i, j)$ can be truncated at some precision.

We consider a generalization of k-means and k-medians, which we denote by (L_2^α, k)-clustering, where $\alpha \geq 1$. In this problem, we are given a set P of n points in $\mathbb{R}^d$, and want to find a set $K \subseteq \mathbb{R}^d$ of k cluster centers, such that the sum of the α power of Euclidean distance from every point to its nearest center is minimized. Using this notation, the k-means problem corresponds to the (L_2^2, k)-clustering, and the k-medians problem to the (L_2, k)-clustering.

Usually, the notion of centers is associated to a k-clustering problem. We say that a center of a point set $C \subset \mathbb{R}^d$, denoted by $center(C)$, is a point $i \in \mathbb{R}^d$ such that $\sum_{j \in C} c(i, j)$ is minimum. An approximate center set for a k-clustering problem, with respect to a dissimilarity measure c, is defined below. In the following, we represent the total cost to connect a set of clients C to a center set $K \subseteq F$ as $c(K, C) = \sum_{j \in C} \min_{i \in K} c(i, j)$.

Definition 3.2. *A (k, ε)-approximate center set for a set of clients C is a set $S \subseteq F$ which contains a subset $K \subseteq S$ of size $|K| = k$ such that $c(K, C) \leq (1 + \varepsilon)c(K', C)$, for every $K' \subset F$, with $|K'| = k$. In other words, the set S contains a center set whose cost deviates by an ε fraction of the optimal value.*

The problem to find a $(1, \varepsilon)$-approximate center set of a set $C \subseteq \mathbb{R}^d$ is the most basic clustering problem, since $k = 1$. Kumar *et al.* [86] showed that it is possible to obtain, with constant probability, $(1, \varepsilon)$-center sets from a constant sized subset of clients. Their main idea is that, to solve a k-clustering problem, one could obtain an approximate center



This example shows an example of an approximate center set for the squared Euclidean distance: given a set of points (gray points), and a multisample of the points (black points), the random sample procedure returns as the center set the mean of the sample points (black square). With constant probability, this center is close to the mean of all points (white square).

Figure 3.1: Center set example

for one cluster of the optimal k-clustering, remove the corresponding clients, and solve the remaining $(k-1)$-clustering problem recursively. To obtain an approximation for the optimal center of such cluster, even if such cluster is not known, Kumar *et al.* [86] showed the existence of *random sampling procedures*. Next, we give a slightly adapted definition of a random sampling procedure. We denote by $opt_k(Q)$ the value of an optimal k-clustering of $Q \subset \mathbb{R}^d$.

Definition 3.3 (adapted from [86])**.** *A* random sampling procedure $\mathcal{A}$ *with a constant parameter* ε *takes as input a multiset* R *of size* λ_ε *with elements of* C*, and produces as output a set* $core(R) \subseteq F$ *of size* β_ε*, where* λ_ε *and* β_ε *are constants for* $\mathcal{A}$*, that depend only on* ε*. If* R *is a multisample obtained by drawing elements a set* $Q \subseteq C$ *with uniform probability, then with constant probability there is at least one* $p \in core(R)$ *such that* $c(\{p\}, Q) \leq (1+\varepsilon)opt_1(Q)$*. Further, the time taken by* $\mathcal{A}$ *is polynomial on the size of* R*.*

The idea of the random sampling procedure is that we may solve exactly the 1-cluster problem for a multisample of constant size (or approximate its solution when it is not possible to give an exact solution), and obtain an approximate center for one of the centers of the optimal solution. Figure 3.1 illustrates one approximate center set.

3.3 Approximations for ConFL

In this section we show how to deal with a generalized facility location problem such that the set of facilities is infinite. In the Subsection 3.3.1 we present a discretization

lemma that reduces the Generalized FLP to the corresponding discrete version. The idea is to use a random sampling procedure, and transform an instance with an infinite set of facility locations into an instance with bounded number of location, and with approximate optimal value. In Subsection 3.3.2, we use this lemma to give approximations for the L_2-ConFLP and the L_2^2-ConFLP, respectively. In Subsection 3.3.3 we show that a natural and straightforward way to solve the L_2^2-ConFLP — solving the corresponding k-clustering problem with squared metric — cannot improve the approximation obtained in Subsection 3.3.2.

3.3.1 A discretization lemma

Kumar *et al.* [86] use a random sampling procedure to obtain approximate centers of one optimal cluster, and use this information to obtain an algorithm for k-medians, by removing one cluster at a time, and recursively solving the remaining subproblem. The algorithm is linear in the number of points, but is exponential in k. One way to solve facility location problems with uniform facility cost would be solving the corresponding k-clustering problem for each possible number of open facilities, k. Since, in the FLP, the number of open facilities is not known a priori, it is not possible to obtain an approximation using the approach of Kumar *et al.* Instead, we use a random sampling procedure to obtain approximate centers for all optimal clusters at once, and solve the reduced problem using known combinatorial algorithms.

Since each multisample has a constant size, we may enumerate all of them in polynomial time. By joining the center sets obtained for each multisample, we ensure that at least one point that is near to each optimal cluster center is known. The following lemma shows that if there is a random sampling procedure set for a given infinite FLP, then we may readily reduce it to the corresponding discrete version. This reduction is approximation preserving, except for an ε error. We assume that in such infinite versions of the FLP, every facility has the same opening cost.

Lemma 3.1. *Consider an instance of the Generalized FLP such that for any pair $j, l \in F$, we have $f(j, r) = f(l, r)$ for every integer $r \geq 0$. Suppose that there exists a random sampling procedure for the set of clients C, facilities F, and dissimilarity measure c. Also, suppose that there exists a β-approximation for the corresponding discrete version of the problem. Then, there exists a $(\beta + \varepsilon)$-approximation for every $\varepsilon > 0$.*

Proof. Define $\varepsilon' = \frac{\varepsilon}{\beta}$, and let $\lambda_{\varepsilon'}$ and $core(\cdot)$ be as in Definition 3.3. Construct the set of all multisamples of size at most $\lambda_{\varepsilon'}$, $\mathcal{R} = C^{\lambda_{\varepsilon'}}$. That is, one element in $\mathcal{R}$ is a tuple $(c_1, \ldots, c_{\lambda_{\varepsilon'}})$ of points in C. Then, obtain

$$\mathcal{F} = \bigcup_{R \in \mathcal{R}} core(R),$$

by applying the random sampling procedure to each set R. Notice that there are $n^{\lambda_{\varepsilon'}}$ distinct multisamples, where $n = |C|$, and that the size of each $core(R)$ is polynomial in the size of R, $|R| = \lambda_{\varepsilon'}$. Consider the instance of the discrete FLP given by facilities $\mathcal{F}$, clients C, and facility and connection cost functions f and c, respectively. We show that an approximate solution for instance $\{\mathcal{F}, C, f, c\}$ is also an appropriate solution for instance $\{F, C, f, c\}$.

Let $S_c = \{c_1, \ldots, c_t\} \subseteq F$ be the set of open facilities in an optimal solution of the original FLP instance, with solution cost OPT_c. Let P_i be the set of clients served by c_i, for $1 \leq i \leq t$. Also, let $S_d = \{c'_1, \ldots, c'_l\} \subseteq \mathcal{F}$ be the set of open facilities in an optimal solution of the discrete problem with solution cost OPT_d. Similarly, let P'_i be the set of clients served by c'_i, for $1 \leq i \leq l$.

First, we notice that, for any center c_i in S_c, there is a point c''_i in set $\mathcal{F}$ that is close to c_i. Indeed, consider the set P_i of clients assigned to c_i in solution S_c. Also, let R be a random multisample of P_i of size $\lambda_{\varepsilon'}$. Using Definition 3.3, by instantiating $Q = P_i$, we obtain that with constant probability there exists at least one $c \in core(R)$ such that $c(\{c\}, P_i) \leq (1 + \varepsilon')opt_1(P_i) = c(\{c_i\}, P_i)$. Since the algorithm considers all such multisamples, there must be some R such that, for some $c''_i \in core(R) \subseteq \mathcal{F}$, we have $c(\{c''_i\}, P_i) \leq (1 + \varepsilon')c(\{c_i\}, P_i)$.

Now consider the solution ϕ (not necessarily optimal) for the discrete instance that opens facilities $S_\phi = \{c''_1, \ldots, c''_t\}$, and that assigns center $\phi(j) = c''_i$ for every $j \in P_i$, and $1 \leq i \leq t$. From the optimality of solution S_d, we obtain

$$\begin{aligned}
\mathrm{OPT}_d &= c(S_d, C) + \sum_{i=1}^{l} f(c'_i, |P'_i|) \\
&\leq c(S_\phi, C) + \sum_{i=1}^{t} f(c''_i, |P_i|) \\
&= \sum_{i=1}^{t} c(S_\phi, P_i) + \sum_{i=1}^{t} f(c_i, |P_i|) \\
&\leq \sum_{i=1}^{t} c(c''_i, P_i) + \sum_{i=1}^{t} f(c_i, |P_i|) \\
&\leq \sum_{i=1}^{t} (1 + \varepsilon')c(c_i, P_i) + \sum_{i=1}^{t} f(c_i, |P_i|) \\
&\leq (1 + \varepsilon')\mathrm{OPT}_c.
\end{aligned}$$

Finally, using the β-approximation for the discrete FLP, we obtain a solution for the ConFL with value SOL, such that $\mathrm{SOL} \leq \beta\mathrm{OPT}_d \leq \beta(1 + \varepsilon')\mathrm{OPT}_c = (\beta + \varepsilon)\mathrm{OPT}_c$. □

3.3.2 Euclidean and squared Euclidean ConFL

The ConFL is the particular case of the generalized facility location in which the set of clients C is a subset of the space $\mathbb{R}^d$, for an integer dimension d, and a facility is any point of the continuous space $\mathbb{R}^d$. We assume that the dissimilarity measure is a function $c : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$, computed in time that depends only on d. The costs to open any two facilities in $\mathbb{R}^d$ to serve a number r of clients are the same, that is, for any $j, l \in \mathbb{R}^d$, $f(j, r) = f(l, r)$ for every integer $r \geq 0$. This gives some flexibility on the cost of a cluster. For example, the cost can be either constant, or proportional to the number of elements.

The ConFL can be used for clustering points. In such applications, the considered dissimilarity measure is usually either the Euclidean distance, or the squared Euclidean distance. Meira and Miyazawa [108] considered the case in which the cost to open a facility is fixed, and may be interpreted as a penalty for each created cluster. More precisely, they considered the particular cases of ConFL in which $f(i, r) = f$ for every $i \in F$ and $r > 0$, and the dissimilarity measure is either the L_2-norm, or the squared L_2-norm. These variants are denoted here by L_2-ConFLP and L_2^2-ConFLP, respectively.

To obtain approximation algorithms for the ConFL problem, we need random sampling procedures for the corresponding k-clustering problems as stated in Lemma 3.1. For k-means, Inaba *et al.* [71] showed that the centroid of a uniformly (uniformly sampled) random multiset of P with m points is a $(1, \frac{1}{\delta m})$-approximate centroid with probability at least $1 - \delta$ for any $\delta > 0$. For k-medians, Kumar *et al.* [86] showed that a random multisample of size $O((\frac{1}{\varepsilon})^{O(1)})$ may be used to obtain an ε-approximate center set of size $O(2^{(1/\varepsilon)^{O(1)}})$ with constant probability. This satisfies the premises of a random sampling procedure for both the L_2 and L_2^2 versions of ConFL.

Now, we may obtain approximations for ConFL using the best known approximations for the discrete versions of Euclidean FLP and Squared Euclidean FLP. For the Euclidean FLP, the best algorithm is a PTAS of Arora *et al.* [7]. We notice, however, that this algorithm assumes that d is a small constant, since it is exponential in d. For the case that d is part of the input, we use the best algorithm for the Metric FLP, that is a 1.488-approximation by Li [93]. For the Squared Euclidean FLP, we use the best algorithm for Squared Metric FLP, that the 2.04-approximation obtained in Chapter 2.

Theorem 3.1. *There is a PTAS for the L_2-ConFLP for fixed dimension d, and there is a $(1.488 + \varepsilon)$-approximation for arbitrary d. Also, there is a $(2.04 + \varepsilon)$-approximation for the L_2^2-ConFLP for every $\varepsilon > 0$.*

One may also consider the PTAS's for k-medians to derive approximation schemes for the L_2-ConFLP. For this, it would be necessary to run the k-medians algorithm for each $k \in \{1, \ldots, n\}$. In this approach, one must use an algorithm with time that is polynomial on k. Kolliopoulos and Rao [83], and Har-Peled and Muzumdar [66] present

approximation schemes for k-medians, all for fixed d. Such results lead to alternative approximation schemes to L_2-ConFLP for fixed d.

It is also natural to solve L_2^2-ConFLP running k-means for each $k \in \{1, \ldots, n\}$. The approximation schemes for k-means whose running time is polynomial in d have exponential dependency on k, according to a result in [64]. So, the running time of this approach leads to exponential time algorithms. If a PTAS is not aimed, then it is possible to use k-means algorithms with worse approximation guarantees. One possibility is to solve the more general k-clustering problem that satisfies a squared metric, that we call the *Squared Metric k-medians*. This problem is discussed in the next subsection, and it is shown that the 2.04-approximation for the L_2^2-ConFLP cannot be improved using this approach.

3.3.3 Discrete squared metric k-medians

The k-medians problem aims at finding a k-clustering of a set of points C, that minimizes the overall sum of the Euclidean distance from points to the center of each cluster. That is, in k-medians, a facility is any point of the continuous space. An often considered variant is the discrete k-medians, where the sets of clients and facilities are given by a weighted bipartite graph, and the distance function is given by the edge weights. The most studied problem is the so called *Metric k-medians*, for which the distance function satisfies the triangle inequality. We consider another variant of the discrete k-medians, when the square root of the distance function satisfies the triangle inequality, that is, the distance function is a squared metric. Formally the *Squared Metric k-medians* problem is a particular case of the discrete k-medians in which, for all facilities i and i', and clients j and j', we have $\sqrt{c(i,j)} \leq \sqrt{c(i,j')} + \sqrt{c(i',j')} + \sqrt{c(i',j)}$. This kind of distance function is a relaxation of a metric, and was considered in [53].

The Squared Metric k-medians can be thought as the discrete version of k-means. In fact, if we use center sets, we can even solve k-means using algorithms for the Squared Metric k-medians, as discussed before, and thus also solve the L_2^2-ConFLP. One advantage of considering this discrete version is that, since the distance function is given by an oracle, the algorithm running time has no dependency on the dimension d. While there are approximation schemes for k-means, the algorithms of Jain and Vazirani [76], based on LP relaxations, and Kanungo *et al.* [78], based on local search, have large approximation factors of 108[1] and $9 + \varepsilon$, respectively. However, these algorithms consider both d and k as part of the instance. In fact, we can use them to solve the more general Squared Metric k-medians problem, and obtain 54 and $(9 + \varepsilon)$-approximation algorithms, respectively.

[1]Jain and Vazirani only consider as candidate centers the points of the instance. Using better approximate center sets, as those of Inaba *et al.* [71], one can readily improve this factor to obtain a $(54 + \varepsilon)$-approximation.

The 54-approximation of Jain and Vazirani is based on the Lagrangian relaxation of k-medians, that corresponds to the FLP with uniform facility cost. The algorithm is based on the idea of guessing (by a binary search) an appropriate value for this cost and approximately solving the FLP so that only k facilities are open. If the binary search is successful, the algorithm yields a 9-approximation for the squared k-medians. Since, finding (in polynomial time) such a facility cost is not always possible, there is an additional rounding step, that multiplies the approximation factor by an extra factor. For the squared metric instances, Jain and Vazirani gave a preliminary analysis with factor 6, using a relaxed triangle inequality, and thus obtained factor 9×6.

The Metric k-medians is related to the Metric FLP, and, analogously, the Squared Metric k-medians is related to the Squared Metric FLP. There is a known lower bound of 1.463 for Metric FLP [61], and $1+2/e \approx 1.73$ [73] for the Metric k-medians. Similarly, the Squared Metric FLP has a lower bound of 2.04 [53]. For the Squared Metric k-medians, it is straightforward to extend these results, and obtain the following theorem. For the sake of completeness, we include a proof sketch.

Theorem 3.2. *There is no* $(1+8/e-\varepsilon \approx 3.943)$*-approximation algorithm for the Squared Metric* k*-medians for any* $\varepsilon > 0$*, unless* P = NP*.*

Proof sketch. First, notice that the statement of Theorem 2.1 also applies to the particular version of the Squared Metric FLP, so that all facilities have the same cost (it is enough to notice that in the proof of Theorem 2.1, all facilities of a given instance have the same cost).

The key observation is that an α-approximation for the Squared Metric k-medians implies a $(1, \alpha)$-approximation for the Squared Metric FLP with uniform facility cost f: given an instance of the FLP, create an instance of k-medians with the same set of clients and set of facilities, for each $k = 1, \ldots, m$, where m is the total number of facilities; solve each such instance using the α-approximation, and return the solution with smallest connection cost. This solution is also a solution for the original FLP.

For some k, let OPT_k be the value of an optimal solution for the k-medians instance, and let SOL be the value of the returned solution for the facility location. Also, let F_{OPT}, and C_{OPT} be the facility and connection costs of an optimal solution for the facility location. Clearly, if such solution opens k' facilities, then $F_{\mathrm{OPT}} = k'f$, and $\mathrm{OPT}_{k'} = C_{\mathrm{OPT}}$. We derive that $\mathrm{SOL} \leq k'f + \alpha\,\mathrm{OPT}_{k'} = F_{\mathrm{OPT}} + \alpha\, C_{\mathrm{OPT}}$. Therefore, the reduction is a $(1, \alpha)$-approximation for the FLP with uniform facility cost.

The theorem follows from the modified Theorem 2.1 (for the Squared Metric FLP with uniform facility cost), using $\gamma_f = 1$. □

This theorem implies that it is unlike that an algorithm for the Squared Metric

k-medians can be used to improve the $(2.04+\varepsilon)$-approximation for the L_2^2-ConFLP obtained using an algorithm for the discrete Squared Metric FLP.

3.4 Continuous FLP with powers of Euclidean distances

We also consider a generalization of the Euclidean and squared Euclidean ConFL when the considered dissimilarity measure is the α-th power of the Euclidean distance for some $\alpha \geq 1$. The considered problem is the L_2^α-ConFLP, which is the particular case of ConFL such that $f(i,r) = f$, where $f \geq 0$ is a given constant, for every $i \in F$ and $r > 0$, and the dissimilarity measure L_2^α is the α-th power of the Euclidean distance. That is, for two points $p_i, p_j \in \mathbb{R}^d$,

$$L_2^\alpha(p_i, p_j) = \|p_i - p_j\|^\alpha = \left(\sqrt{\sum_{t=1}^{d}(p_{it} - p_{jt})^2}\right)^\alpha .$$

When $\alpha = 1$, L_2^α is the Euclidean distance. For this case, Kumar *et al.* [87] proposed a random sampling procedure extending the result of Bādoiu *et al.* [11]. For $\alpha = 2$, the function L_2^α is the widely used L_2^2 norm, and Inaba *et al.* [71] proposed a random sampling procedure for this distance function. For a general α, we do not know whether it is possible to obtain a constant sized $(1,\varepsilon)$-center set from a random multisample with high probability, and thus we do not give a random sampling procedure. However, we may extend the results of Bādoiu *et al.* [11] and Kumar *et al.* [87] to obtain a $(1,\varepsilon)$-center set of polynomial size to the L_2^α dissimilarity measure. This is sufficient to derive approximations for L_2^α-ConFLP from approximations for the discrete case.

We start with an auxiliary lemma, that gives a rough bound on the generalized binomial. For the sake of completeness, we include a proof.

Lemma 3.2. *Let* $0 < x < \frac{1}{2}$ *and* $\alpha \geq 1$. *It holds* $(1+x)^\alpha \leq 1 + 2x\alpha^\alpha$.

Proof.

$$(1+x)^\alpha = 1 + \sum_{k=1}^{\infty}\binom{\alpha}{k}x^k \leq 1 + \alpha^\alpha \sum_{k=1}^{\infty} x^k \leq 1 + \alpha^\alpha x \sum_{k=0}^{\infty}\frac{1}{2}^k = 1 + 2x\alpha^\alpha,$$

where the first inequality is valid because $\binom{\alpha}{k} \leq \alpha^\alpha$. □

The following lemma extends Theorem 3.2 of Bādoiu *et al.* [11]. The proof is the same, except that for the L_2^α we may not assume triangle inequality, so we have to be careful when considering an alternative center candidate.

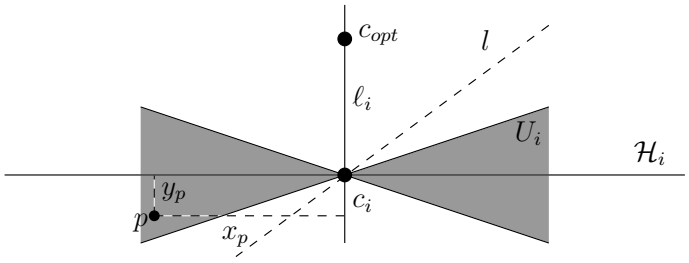


Figure 3.2: Definitions of Lemma 3.3 ([11])

In the following, consider a set of points P of size n, let c_{opt} be an optimal center of the corresponding 1-median problem, $\text{OPT} = \sum_{p \in P} ||p - c_{opt}||^\alpha$ be the cost of an optimal solution, and T be such that $T^\alpha = \frac{1}{n}\text{OPT}$. The value $\frac{1}{n}\text{OPT}$ is the average cost of connecting clients to the optimal center. Also, define $M_\varepsilon = \lceil 1/\varepsilon^{2\alpha} \cdot \log_{1-\varepsilon^2/1024} \varepsilon^2/2 \rceil$ for some $\varepsilon > 0$. We say that the *flat span* of a subset of points $X \subset \mathbb{R}^d$, denoted by $span(X)$, is the affine subspace spanned by points of X.

Lemma 3.3. *Let $1/2 > \varepsilon > 0$, and let X be a random sample of points (uniformly drawn) from P of size M_ε. Then, with constant probability, (i) the projection x of c_{opt} on the flat span of X is a $(1+4\varepsilon)^\alpha$-approximate 1-median for P, and (ii) X contains a point y such that $||y - c_{opt}|| \leq 2T$.*

Proof. Let $\beta = \varepsilon/16$. We will randomly sample a set of points $s_1, \ldots, s_u$. For each i, let F_i be the flat span of the first i sampled points $s_1, \ldots, s_i$, let c_i be the projection of c_{opt} on F_i, and let $d_i = ||c_{opt} - c_i||$ be the (Euclidean) distance from the optimal center c_{opt} to the flat span F_i. Denote by ℓ_i the line that pass through c_{opt} and c_i, and define $U_i = \left\{x \in \mathbb{R}^d \mid \pi/2 - \beta \leq \angle c_{opt} c_i x \leq \pi/2 + \beta\right\}$ as the complement of the cone of angle $\pi - \beta$ emanating from c_i and axis ℓ_i. Let $Q_i = P \setminus U_i$ be the set of clients in such cone. Also, denote by $\mathcal{H}_i$ the $(d-1)$-dimensional hyperplane orthogonal to ℓ_i and passing through c_i (see Figure 3.2).

Now, we partition the sampled points in rounds. The first round is finished when a point s_i such that $||s_i - c_{opt}|| \leq 2T$ is sampled. By Markov inequality, $P(||s_i - c_{opt}|| \geq 2T) = P(||s_i - c_{opt}||^\alpha \geq (2T)^\alpha) \leq (\frac{1}{n}\text{OPT})/(2T)^\alpha \leq 1/2$. Therefore, this event occurs with probability at least $1/2$. Suppose that the sample s_i has just finished a round, then the next round is finished when one of three cases happen:

Case (a): The distance from c_{opt} to F_i is $d_i \leq \varepsilon^2 T$.

We divide P in two parts, $P_1 = \{x \in P \mid ||x - c_{opt}|| \leq \varepsilon T\}$, and $P_2 = P \setminus P_1$. If $p \in P_1$, then $||p - c_i|| \leq (\varepsilon + \varepsilon^2)T$, and if $p \in P_2$, then $||p - c_i|| \leq ||p - c_{opt}|| + \varepsilon^2 T \leq (1+\varepsilon)||p - c_{opt}||$.

We calculate the cost of using c_i as a center.

$$\begin{aligned}\sum_{p\in P}||p-c_i||^\alpha &= \sum_{p\in P_1}||p-c_i||^\alpha + \sum_{p\in P_2}||p-c_i||^\alpha \\ &\leq (\varepsilon+\varepsilon^2)^\alpha \sum_{p\in P_1} T^\alpha + (1+\varepsilon)^\alpha \sum_{p\in P_2}||p-c_{opt}||^\alpha \\ &\leq (\varepsilon+\varepsilon^2)^\alpha \sum_{p\in P} T^\alpha + (1+\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha \\ &\leq (1+\varepsilon+\varepsilon+\varepsilon^2)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha \\ &\leq (1+3\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha.\end{aligned}$$

The third inequality follows since both sums are lower bounds on the optimal value. In this case, we are done.

Case (b): The number of points in Q_i is at most $\varepsilon^{2\alpha}n$.

We divide P in three parts, $P_1 = P \cap U_i$, $P_2 = \{p \in Q_i \mid ||p-c_{opt}|| \leq \frac{1}{\varepsilon}T\}$, and $P_3 = Q_i \setminus P_2$. Consider a point $p \in P_1$, let x_p be the distance of p to the line ℓ_i, and let y_p be the distance between p and $\mathcal{H}_i$. We have $y_p \leq x_p \tan\beta \leq x_p \frac{\sin\beta}{\cos\beta} \leq 4\beta x_p \leq \frac{\varepsilon x_p}{4} \leq \frac{\varepsilon}{4}||p-c_{opt}||$, since $\beta < 1/16$. So we obtain $||p-c_i|| \leq x_p + y_p \leq (1+2\frac{\varepsilon}{4})||p-c_{opt}||$. Now, consider $p \in Q_i$, and recall $||c_i - c_{opt}|| \leq 2T$. If $p \in P_2$, then $||p-c_i|| \leq ||c_i-c_{opt}|| + ||p-c_{opt}|| \leq (2+\frac{1}{\varepsilon})T$, and if $p \in P_3$, then $||p-c_i|| \leq ||p-c_{opt}|| + 2T \leq ||p-c_{opt}|| + 2\varepsilon||p-c_{opt}|| = (1+2\varepsilon)||p-c_{opt}||$. We calculate the cost of using c_i as a center.

$$\begin{aligned}\sum_{p\in P}||p-c_i||^\alpha &= \sum_{p\in P_1}||p-c_i||^\alpha + \sum_{p\in P_2}||p-c_i||^\alpha + \sum_{p\in P_3}||p-c_i||^\alpha \\ &\leq (1+\frac{\varepsilon}{2})^\alpha \sum_{p\in P_1}||p-c_{opt}||^\alpha + (2+\frac{1}{\varepsilon})^\alpha \sum_{p\in P_2} T^\alpha + (1+2\varepsilon)^\alpha \sum_{p\in P_3}||p-c_{opt}||^\alpha \\ &\leq (1+\frac{\varepsilon}{2})^\alpha \sum_{p\in P_1}||p-c_{opt}||^\alpha + (2+\frac{1}{\varepsilon})^\alpha \varepsilon^{2\alpha} n T^\alpha + (1+2\varepsilon)^\alpha \sum_{p\in P_3}||p-c_{opt}||^\alpha \\ &= (1+\frac{\varepsilon}{2})^\alpha \sum_{p\in P_1}||p-c_{opt}||^\alpha + (2\varepsilon^2+\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha \\ &\qquad + (1+2\varepsilon)^\alpha \sum_{p\in P_3}||p-c_{opt}||^\alpha \\ &\leq (1+2\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha + (2\varepsilon^2+\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha \\ &\leq (1+2\varepsilon+2\varepsilon^2+\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha \\ &\leq (1+4\varepsilon)^\alpha \sum_{p\in P}||p-c_{opt}||^\alpha.\end{aligned}$$

We are done also in this case.

Case (c): There is at least $\varepsilon^{2\alpha}n$ points in Q_i, and $d_i \geq \varepsilon^2 T$.

The new round continues until a random sample point $s_j \in Q_i$ is picked. Let l be the line segment connecting c_i to s_j, then l must be in F_j. Also, the angle between l and ℓ_i is smaller than $\pi/2 - \beta$, so $d_j = \text{dist}(F_j, c_{opt}) \leq \text{dist}(l, c_{opt}) \leq \sin(\pi/2 - \beta)||c_{opt} - c_i|| = \cos(\beta)d_i \leq (1-\beta^2/4)d_i$, where $\text{dist}(X, p) = \min_{x \in X} ||x-p||$ for some $X \subseteq \mathbb{R}^d$. This means that, in this case, the distance from the optimal center c_{opt} and the flat span of successive rounds shrinks by a factor of $(1 - \beta^2/4)$. Since the probability of sampling a point in Q_i is at least $\varepsilon^{2\alpha}$, the expected number of samples in this round is at most $1/\varepsilon^{2\alpha}$.

For the first round we have $d_i \leq ||s_i - c_{opt}|| \leq 2T$, and we stop sampling as soon as the distance from c_{opt} to the flat span is smaller than $\varepsilon^2 T$. Therefore the maximum number of rounds finished by Case (c) is bounded by $\log_{1-\beta^2/4}(\varepsilon^2 T)/(2T)$, and the expected total number of sampled points is at most $M_\varepsilon = \lceil 1/\varepsilon^{2\alpha} \cdot \log_{1-\beta^2/4} \varepsilon^2/2 \rceil$. The lemma follows from Markov inequality. □

The next lemma obtains a center set from the flat span obtained from Lemma 3.3, assuming that bounds on T are known. This uses the construction of Kumar *et al.* [87] and Bādoiu *et al.* [11].

Lemma 3.4. *Let $1/4 > \varepsilon > 0$, and let X be a random sample of points from P of size M_ε. Suppose that we know numbers a, b such that $a \leq T \leq b$. Then, we can construct a set Y of $O(2^{(1/\varepsilon)^{O(1)}} \log(b/a))$ points that, with constant probability, is a $(1, 16\varepsilon\alpha^\alpha)$-center set of P. Further, it takes $O(2^{(1/\varepsilon)^{O(1)}} \log(b/a)d)$ time to construct Y from X.*

Proof. First, notice that

$$\begin{aligned} M_\varepsilon &= \lceil 1/\varepsilon^{2\alpha} \log_{1-\varepsilon^2/1024} \varepsilon^2/2 \rceil \\ &= O(1/\varepsilon^{2\alpha} \log(\varepsilon/2) \,/\, \log(1 - \varepsilon^2/1024)) \\ &= O(1/\varepsilon^{2\alpha} \log(2/\varepsilon) \,/\, \log(1 - \varepsilon^2/1024)^{-1})) \\ &= O(1/\varepsilon^{2\alpha} \log(2/\varepsilon) \,/\, \varepsilon^2/1024)) \\ &= O(1/\varepsilon^{2+2\alpha} \log 1/\varepsilon), \end{aligned}$$

where the penultimate equality is valid since $\log(1-x)^{-1} \geq x$, by Taylor expansion.

We construct exponential grids $G_p(t)$ for each point $p \in X$, and for each $t = 2^i$, with i in the range $[\lfloor \log a \rfloor, \lceil \log b \rceil]$. The grid $G_p(t)$ has side length $l = \varepsilon^2 t/(4|X|)$ in $span(X)$, and is centered at p. The candidate set Y is the union of vertices of every $G_p(t)$, that are at distance at most $4t$ from p. The number of considered points in each such a grid $G_p(t)$ is at most $(8t/l)^{|X|} = O(2^{(1/\varepsilon)^{O(1)}})$, so the total number of points in Y is $O(|X| \cdot \log(b/a) \cdot 2^{(1/\varepsilon)^{O(1)}}) = O(2^{(1/\varepsilon)^{O(1)}} \log(b/a))$.

The statements of Lemma 3.3 hold with constant probability, so assume they are true. Let x and y be as in Lemma 3.3. Since x is the projection of c_{opt} in $span(X)$ and $y \in span(X)$, we have $||c_{opt} - x|| \leq ||c_{opt} - y|| \leq 2T$. By the triangle inequality, we have $||y-x|| \leq 4T$. Let $t = 2^i$ be such that $t/2 \leq T \leq t$, then $||y-x|| \leq 4t$. Now, let $p \in G_y(t)$ be a grid vertex closest to x such that $||p - y|| \leq 4t$. The distance between p and x is $||p - x|| \leq |X|l \leq \varepsilon^2 t/2 \leq \varepsilon^2 T$.

We divide P in two parts, $P_1 = \{z \in P \mid ||z - x|| \leq \varepsilon T\}$, and $P_2 = P \setminus P_1$. If $z \in P_1$, then $||z - p|| \leq 2\varepsilon T$, and if $z \in P_2$, then $||z - p|| \leq ||z - x|| + \varepsilon^2 T \leq (1 + \varepsilon)||z - x||$. We calculate the cost of using p as a center.

$$\begin{aligned}
\sum_{z \in P} ||z - p||^\alpha &= \sum_{z \in P_1} ||z - p||^\alpha + \sum_{z \in P_2} ||z - p||^\alpha \\
&\leq (2\varepsilon)^\alpha \sum_{z \in P_1} T^\alpha + (1 + \varepsilon)^\alpha \sum_{z \in P_2} ||z - x||^\alpha \\
&\leq (2\varepsilon)^\alpha \mathrm{OPT} + (1 + \varepsilon)^\alpha (1 + 4\varepsilon)^\alpha \mathrm{OPT} \\
&\leq (1 + 8\varepsilon)^\alpha \mathrm{OPT} \\
&\leq (1 + 16\varepsilon\alpha^\alpha)\mathrm{OPT},
\end{aligned}$$

where the second inequality follows from Lemma 3.3, and the last inequality from Lemma 3.2.

Since we must have considered $G_y(t)$ when constructing Y, we are done. □

The following gives logarithmic sized center sets for k-medians with L_2^α dissimilarity measure.

Lemma 3.5. *Let $1/4 > \varepsilon > 0$, and let X be a random sample of $O(1/\varepsilon^{2+2\alpha} \log 1/\varepsilon)$ points from P. Then, we can construct a set Y of $O(2^{(1/\varepsilon)^{O(1)}} \log n)$ points that, with constant probability, is a $(1, 16\varepsilon\alpha^\alpha)$-center set of P. Further, it takes $O(2^{(1/\varepsilon)^{O(1)}} \log n\, d + nd)$ time to construct Y from X.*

Proof. Denote by D the diameter of P, and let $r, s \in P$ be such that $D = ||r - s||$. Fix a point $p \in P$. Find the point $q \in P$ such that the distance $||p-q||$ is maximum, and denote by d this distance. Then, we have $D \leq ||p-r|| + ||p-s|| \leq 2d$. Clearly $2d$ is an upper bound for T. For the lower bound, we know that $nT^\alpha = \mathrm{OPT} \geq \max\{||p - c_{opt}||^\alpha, ||q - c_{opt}||^\alpha\} \geq (d/2)^\alpha$. Therefore, we can instantiate Lemma 3.4 with $a = d/(2n^{1/\alpha})$ and $b = 2d$. □

The center set constructed using this lemma is not of constant size, and so does not guarantee the existence of a random sampling procedure for k-medians with L_2^α distance function. Therefore, it does not satisfy the requirements for the Kumar *et al.* [87] algorithm, that is used to obtain a linear-time approximation scheme for k-medians. If we use this center set with their algorithm, however, we obtain an approximation scheme for the L_2^α k-clustering.

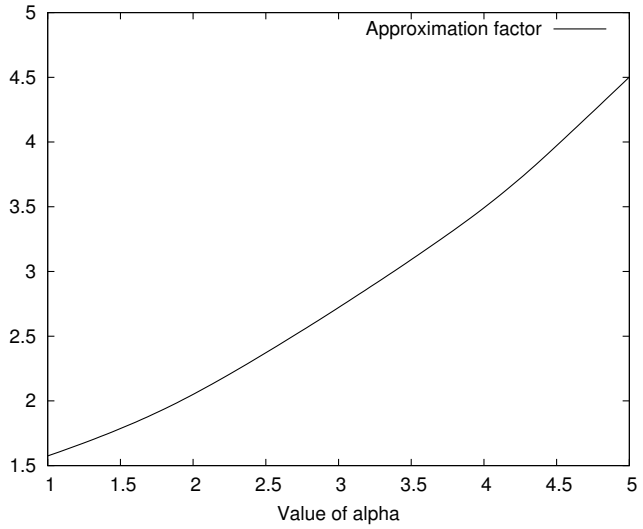


Figure 3.3: Approximation for the L_2^α-ConFLP

Corollary 3.1. *For any $\varepsilon > 0$ and for a constant k, there is a $(1+\varepsilon)$-approximation for the L_2^α k-clustering. This algorithm runs in $O(2^{(k/\varepsilon)^{O(1)}} dn \log^k n)$-time.*

For the L_2^α-ConFLP, we may use Lemma 3.5 instead of a random sampling procedure, in the same way done in Lemma 3.1. Although the obtained center set does not have a constant size, it is obtained from a constant sized random sample. Therefore, we still may enumerate all such samples and obtain a discrete set of candidate centers in polynomial time. We use this to derive an approximation for the L_2^α-ConFLP from an approximation for the corresponding discrete case. Since the proof is analogous to that of Lemma 3.1, we omit it.

Theorem 3.3. *Suppose that there exists a β-approximation for the discrete L_2^α FLP. Then, there exists a $(\beta+\varepsilon)$-approximation L_2^α-ConFLP for every $\varepsilon > 0$.*

Now we use approximations for the M$^\alpha$FLP, that is a variant of the Metric FLP, such that the distance function is the α-th power of a metric. The discrete L_2^α FLP is therefore a particular case of the M$^\alpha$FLP. The following theorem is derived directly from Theorem 3.3, and from results of Subsection 2.6.1. Figure 3.3 shows the approximation factor obtained using this theorem for α between 1 and 5.

Theorem 3.4. *There are constant approximations for the L_2^α-ConFLP, for every $\alpha \geq 1$. In particular, for $\alpha \leq 5$, the approximation factor is not greater than 5.*

Chapter remarks

In the k-clustering problems, one has an estimate k on the number of clusters of a given set of points of the Euclidean space. Another possibility is considering an estimate f on the cost to create each cluster, if an optimal number of clusters is not known. This problem corresponds to the FLP whose set of locations is the complete Euclidean space, and thus is called the Continuous FLP (ConFL). Meira and Miyazawa [108] showed how to apply primal-dual algorithms for the FLP in the continuous set of facilities, leading to constant approximations for ConFL. This chapter showed that, to obtain an approximation for ConFL, it is enough to obtain an approximation for the corresponding discrete Euclidean version of the problem, provided the existence of approximate center sets for the specific dissimilarity measure used. We provide approximate center sets for the case that the distance function is the Euclidean distance raised to some power $\alpha \geq 1$, that is, the L_2^α-norm. Therefore, we improve the results of Meira and Miyazawa, by giving a PTAS for $\alpha = 1$, and constant approximations for every $\alpha > 1$. The approximate center sets were introduced in the context of k-medians and k-means, that correspond to the k-clustering problems with L_2^α-norm for $\alpha = 1$ and $\alpha = 2$, respectively. Kumar *et al.* [87] showed how to derive PTAS's for k-clustering problems from approximate center sets for a given distance function, and thus we also give the first PTAS for the k-clustering problem with L_2^α-norm, for general $\alpha \geq 1$.

Chapter 4

Supply Chain Problems

This chapter considers several generalizations of the Joint Replenishment Problem (JRP), either by considering more general settings, or more relaxed assumptions. First, we introduce a generalization of the Facility Location Problem (FLP), called the Production and Distribution Problem (PDP), which is the problem of minimizing ordering, distribution and inventory holding costs of a supply chain formed by a set of warehouses and retailers. Each retailer can face a demand in each time of a discretized planning horizon that must be satisfied by the items currently held in the inventory, and that were previously ordered and transported from any of the warehouses. The objective is to determine an inventory replenishment policy for each retailer, minimizing the overall cost of stock and transportation. These costs are balanced by a fixed ordering setup cost that depends on the warehouse, but is independent of the time or of the number of items produced.

Next, the One-Warehouse Multiple-Retailer Problem (OWMR) is considered. As in the JRP, the objective is to determine an inventory replenishment policy of a set of retailers, that may place orders to a single retailer. In this problem, however, items may be held in the warehouse before being transported to a retailer. We consider the case that holding cost functions of retailers and warehouse may be independent. Finally, we consider the Multilevel JRP, for which the supply chain is given by a tree with a special root node. For example, a supply chain may be comprised of retailers, warehouses, and a producer. Each leaf of the tree can face demands over time. In this problem, a non-root node may place an order at a given time only if each element in the path from this node to the root also places an order at the same time.

In Section 4.1, related works are reviewed. In Section 4.2, the PDP is described. In Section 4.3, we present an approximation for the PDP, and in Section 4.4 we extend the previous algorithm to the variant of the PDP with retailer ordering costs. In Section 4.5, we present a primal-dual approximation for OWMR. In Section 4.6, we note that the Multilevel JRP may be reduced to the Multistage Assembly Problem.

4.1 Literature review

The JRP has a 2-approximation based on a primal-dual approach by Levi *et al.* [90]. In this algorithm, the dual variables are increased according to the so called "wave mechanism", rather than uniformly as in traditional algorithms. For the more general OWMR problem, when items can be held in the warehouse, there is a 1.8-approximation through LP-rounding, based on a new random shift procedure introduced by Levi *et al.* [91]. They considered the particular case in which the holding cost either decreases, or increases as the fraction of time that the item is held on the warehouse increases. The best known approximation algorithm for the JRP has performance guarantee 1.791 [22]. For the special case that holding costs are either zero or infinity, also known as the JRP with deadlines, there is a randomized 1.574-approximation [21]. An on-line variant of the JRP has been considered by Buchbinder *et al.* [25], who gave a deterministic 3-competitive algorithm, and showed a lower bound of 2.64 on the competitiveness.

There is an efficient approximation scheme (EPTAS) for the JRP with stationary demands and linear holding costs, and a PTAS for the JRP with non-stationary demands, but with soft capacitated single item orders, where each retailer order has a capacity, but several retailer orders at the same time are permitted [113]. The similar multistage assembly problem, that considers the problem of assembling items according to a bill of material, and paying echelon holding costs has a 2-approximation based on the primal-dual approach by Levi *et al.* [90]. They also showed how to solve in polynomial time the particular case of the JRP with only one retailer, called the single-item lot-sizing problem. A capacitated version of the multi-item lot-sizing problem has been considered by Levi *et al.* [89], when there is a hard capacity on the total number of items ordered. They showed that this problem is strongly NP-hard, and gave a 2-approximation based on the rounding of an LP relaxation that is similar to that of the FLP.

Integrated supply chain management. The literature on integrated supply chain problems has considered several design and inventory problems, with different network structures, objective functions, and constraints. Reviews on integrated supply chain are given by Shen [120] and Melo *et al.* [109]. Depending on the considered model, a retailer must be assigned to a unique supplier [96, 120, 130], or it may be assigned to different suppliers at different times [24, 116]. Several approaches have been used to deal with these problems, such as Lagrangian relaxation [45], column generation [121, 130], and metaheuristics [24, 32]. Pochet and Wosley [116, Section 13.4] have discussed valid inequalities for a mixed integer linear programming formulation of a generalized version of the PDP. In their problem, they also considered warehouse stocks, production capacity, and transportation cost depending on time.

Approximation algorithms have been proposed for few problems with integrated costs, such as the Warehouse-Retailer Network Design Problem (WRND) [96, 130], and the Stochastic Transportation-Inventory Network Design Problem (STIND) [96, 123]. In the WRND, one is given a set of warehouses, and a set of retailers. Each retailer faces a constant-rate demand for items, and can be served by one of the warehouses. All the warehouses are served by a single external supplier. The sets of warehouses, retailers, and supplier form a metric space. The objective is to partition the retailers, and associate each partition to a warehouse, such that the overall partition cost is minimized over the infinite time horizon. The cost of each partition includes three components: the yearly operational cost of the warehouse; the annual transportation cost from the supplier to the warehouse, and from the warehouse to the retailer; and the two-echelon inventory cost of the system formed by the warehouse and associated retailers. The STIND is similar to the WRND, but the demand of each retailer is uncertain, with some known mean and variance, and the warehouses should keep some safety stock levels. In this problem, the inventory cost is given by two concave functions: the storage and material handling cost, and the safety stock inventory cost.

Both WRND and STIND admit set covering formulations, and therefore approximations with logarithmic factors, based on the primal-dual greedy algorithm, can be readily obtained. Li *et al.* [96] gave a 1.861-approximation for the WRND, based on the greedy primal-dual algorithm by Jain *et al.* [72], for the case that warehouse holding costs are greater than retailer holding costs, and a 3-approximation based on primal-dual techniques for the case in which all the warehouse holding costs are identical. For the STIND, a primal-dual 3-approximation algorithm was also designed by Li *et al.* [96]. Compared to the WRND and the STIND, the time horizon of the PDP is discretized and finite, and for each time step, each retailer has a known demand. This contrasts to the STIND, for which the demand is uncertain, and to the WRND, for which the demand rate is constant. Also, in the setting of PDP, a retailer is not bonded to a single warehouse, but can choose to order from different warehouses, depending on which is more economical at each time.

The particular case of the PDP with only one time step corresponds to the FLP, that is extensively discussed at Chapter 2. Also, the variant of the PDP with a setup cost for each retailer order is a generalization of the JRP. In general, one can obtain a $\log(n)$-approximation for the PDP, where n is the number of demands, by encoding the problem as an exponential minimum set cover problem, and modifying the $\log(n)$-approximation for the FLP by Hochbaum [70]. This factor can also be obtained by a primal-dual greedy algorithm, and using a factor-revealing based analysis [101]. When the transportation cost function is arbitrary, set cover can be reduced to FLP, so it is unlikely to exist a better factor [51, 112]. Also, set cover would still be encoded by the PDP if the holding cost is dependent on the warehouses, even if transportation costs were zero.

4.2 The Production and Distribution Problem

Traditionally, network design, distribution and inventory replenishment decisions are made separately. In the location theory, the literature focuses on the strategic decisions of network design, such as where to place facilities and how to assign one facility to each client. On the other hand, in the inventory theory, a static network design is usually assumed. Such a static network design has defined assignments between facilities and clients, and the decisions are concentrated on determining replenishment policies for the inventory. However, the lack of coordination between inventory and shipment costs when determining the network design leads to sub-optimality [120]. In this section, rather than assuming a previously defined distribution network design, we consider a pre-established set of locations, and allow a dynamic distribution design that integrates the decisions of shipment, ordering and inventory replenishment policies. For instance, one retailer's stock can be replenished by different warehouses at different times.

Problem's definition. In the PDP, one is given a set of warehouses P, and a set of retailers Q. Each retailer q may face a demand for $d_{qt} \in \mathbb{Z}_+$ units of item in each time t of a discretized planning horizon with steps $\{1, \ldots, T\}$. Each demand can only be satisfied with items that are currently in the stock of the corresponding retailer, *i.e.*, we have a make-to-stock scenario. The stock is initially empty, and may be replenished by placing orders to any warehouse. There is no stock at warehouse facilities, so every time s that a warehouse p receives an order, the demanded items are produced at a setup cost k_p, that is independent of the number of items produced, or the number of retailers participating in the order. Once the items have been produced, each unit is transported to the requesting retailer q at a cost c_{pq}. We assume that the transportation time is negligible, so each unit of item is held in the stock of retailer q from the time s it was produced until the time t it was delivered. The holding cost incurred for this item is h_{qst}. The objective is to minimize the overall sum of ordering, distribution, and holding costs.

Summary of results. We study the PDP under a natural assumption that the transportation and holding cost functions satisfy a generalization of the triangle inequality. The intuition for this assumption is that in many applications it is cheaper to transport one item from the warehouse to the retailer of destination directly, rather than using other retailers as storage midpoints. The main contribution of this chapter is a 2.77-approximation for the PDP. Our algorithm is based on the randomized rounding of the natural LP relaxation, and uses clustering of demand points, in the spirit of the FLP algorithms of Sviridenko [127] and Chudak and Shmoys [38], but has to carefully select the order to be placed for each cluster, due to the additional temporal restriction. This

extra step leads to a worse approximation factor for the service cost, when compared to the standard FLP. To balance the ratios for ordering and service costs, we use two different approaches. In the first approach, we place two orders for each cluster. This results in high ordering cost, but reduces significantly the expected service cost. In the second approach, we use the filtering technique parameterized by some α to obtain the opposite imbalance on the approximation guarantee. Combining the two approaches is done by the use of a probability distribution over the choice of parameter α, or the use of the first approach.

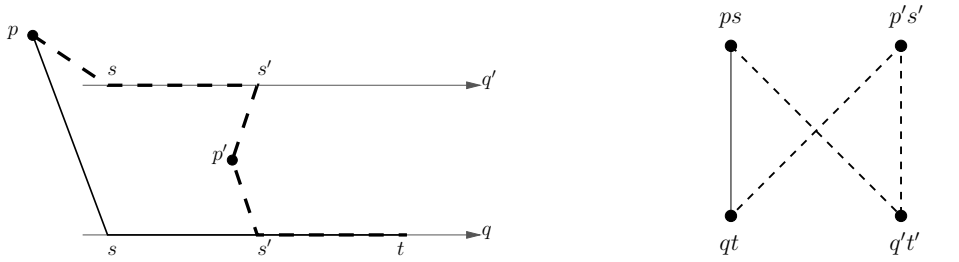
4.2.1 Holding and transportation costs model

For the PDP, the cost incurred to serve a demand point is the combination of two different contributions: the cost to transport one item from a warehouse facility to a given retailer, and the cost to keep one item in the inventory of a retailer until it is delivered. We refer to the sum of distribution and holding costs as the *service cost* of this demand point. In the following, we describe assumptions on these cost functions.

For most inventory problems considered in the literature, the holding cost is modeled on a *per unit* and *per time step* basis, that is, in traditional inventory models a nonnegative cost h_{qt} is incurred to hold one unit of item from time step t to time step $t+1$ in the stock of retailer q. For the PDP, the holding cost is modeled by the more general function h_{qst}. We assume that this holding cost function is monotone, that is, the holding cost can only decrease if the period that an item is kept in the stock is shortened. This is formalized in the next assumption.

Assumption 4.1. *[Monotonicity] Fix a retailer q. Let s, s', t, t' be time steps such that $s \leq t$ and $s' \leq t'$. If $[s,t] \subseteq [s',t']$, then $h_{qst} \leq h_{qs't'}$.*

For location problems, such as the FLP and k-medians, it is common to make the assumption that facilities and clients are in a metric space, that is, the distance between facilities and clients is a symmetric function that satisfies the triangle inequality. Indeed, if no restrictions on the distance function are made, then the FLP is hard to approximate by a factor better than $O(\log n)$. For many distribution problems, however, the assumption of triangle inequality can be made without loss of generality. The reason is that one can create a modified instance where the new distance function is defined by the lengths of the shortest paths in the original graph. This new instance satisfies the triangle inequality, and a solution of non-greater cost for the original problem can be obtained from a solution to this new instance by rerouting direct routes through shortest paths in the graph. We define an analogous notion for the PDP.



(a) Rerouting (b) Triangle inequality

Figure 4.1: Mixed holding and transportation costs metric

Assumption 4.2. *[Generalized Triangle Inequality] For all retailers q and q', warehouses p and p', and time steps s, s' and t such that $s \leq s' \leq t$, it holds:*

$$c_{pq} + h_{qst} \leq c_{pq'} + h_{q'ss'} + c_{p'q'} + c_{p'q} + h_{qs't}.$$

This assumption states that the cost to transport one item from the warehouse p directly to the retailer q, and holding it in q until the delivery time t is cheaper than the following alternative route: transporting the item from the warehouse p to the retailer q', holding it from time s to time s', then transporting it again from retailer q' to retailer q through warehouse p', and holding it until time t. This alternative route is depicted in Figure 4.1a. This inequality can also be interpreted in the following way: the cost to serve demand point (q, t) directly by order (p, s) is not greater than the overall cost to serve demand (q, t) by order (p', s') and some demand (q', t') by orders (p, s) and (p', s'). Since this inequality resembles the triangle inequality, as illustrated in Figure 4.1b, we say that a pair of holding and transportation costs that satisfies Assumption 4.2 forms a metric service cost for the PDP.

4.2.2 A linear programming relaxation

The PDP has a natural formulation as a mixed integer linear program (MILP). This is analogous to that of facility location problems, where a client is a demand point, and a facility is a warehouse order. The difference between the FLP and the PDP formulations is that, in the PDP, a demand point can only be served by orders placed before its arrival time, whilst, in the FLP, a client point can be served by any facility. In the following, let $\mathcal{D} \subseteq Q \times [T]$ be the set of all positive demand points. Also, let $\mathcal{P} = P \times [T]$ be the set of all potential warehouse orders, and for each $t \in [T]$, let $\mathcal{P}_t = P \times [t]$ be the set of potential warehouse orders that can serve a demand point at time t. Variable x_{ps}^{qt} indicates that demand (q, t) is served by warehouse order (p, s), and variable y_{ps} indicates whether the

order (p, s) is placed. The MILP is given in the following.

$$\begin{array}{lll} \text{minimize} & \sum_{(p,s)\in\mathcal{P}} y_{ps}\, k_p + \sum_{(q,t)\in\mathcal{D}} \sum_{(p,s)\in\mathcal{P}_t} x_{ps}^{qt}\, d_{qt}(h_{qst} + c_{pq}) & \\ \text{subject to} & \sum_{(p,s)\in\mathcal{P}_t} x_{ps}^{qt} \geq 1 & (q,t) \in \mathcal{D} \\ & x_{ps}^{qt} \leq y_{ps} & (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P}_t \\ & x_{ps}^{qt} \geq 0 & (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P}_t \\ & y_{ps} \in \{0,1\} & (p,s) \in \mathcal{P} \end{array} \tag{4.1}$$

A linear relaxation can be obtained by replacing the integrality constraints $y_{ps} \in \{0, 1\}$ by constraints $y_{ps} \geq 0$, for each $(p, s) \in \mathcal{P}$. The dual program corresponding to the relaxation is given in the following.

$$\begin{array}{lll} \text{maximize} & \sum_{(q,t)\in\mathcal{D}} b_{qt} & \\ \text{subject to} & b_{qt} \leq d_{qt}(h_{qst} + c_{pq}) + z_{ps}^{qt} & (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P}_t \\ & \sum_{(q,t)\in\mathcal{D}:t\geq s} z_{ps}^{qt} \leq k_p & (p,s) \in \mathcal{P} \\ & z_{ps}^{qt}, b_{qt} \geq 0 & (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P} \end{array} \tag{4.2}$$

Consider any feasible solution (x, y) of program (4.1). We define the *service set* of a demand $j = (q, t)$, denoted by $\mathcal{S}_j$, as the set of all orders $(p, s) \in \mathcal{P}_t$ such that $x_{ps}^{qt} > 0$. Also, the *service window* of demand j is the time interval $[\min_{(p,s)\in\mathcal{S}_j} s,\ t]$. The fractional service cost of demand j is $S_j = C_j + H_j$, where

$$C_j = \sum_{(p,s)\in\mathcal{S}_j} x_{ps}^{qt}\, d_{qt} c_{pq} \quad \text{and} \quad H_j = \sum_{(p,s)\in\mathcal{S}_j} x_{ps}^{qt}\, d_{qt} h_{qst}$$

are the fractional distribution cost, and the fractional holding cost of j, respectively. Also, the total cost of the solution is divided between the ordering cost $K = \sum_{(p,s)\in\mathcal{P}} y_{ps}\, k_p$, and the service cost $S = C + H$, where

$$C = \sum_{j\in\mathcal{D}} C_j \quad \text{and} \quad H = \sum_{j\in\mathcal{D}} H_j$$

are the total distribution cost, and the total holding cost, respectively. We represent an optimal solution of the LP by (x^*, y^*), and define K^*, S^*, C^*, H^*, S_j^*, C_j^*, H_j^* accordingly.

4.2.3 Complete solutions and filtering

In the following, we describe the notion of *complete* solutions for the PDP, which are commonly used in the FLP literature [38]. Then, we describe how to apply *filtering* [97] to modify a fractional solution of the LP. This technique has been used in many algorithms for the FLP [28, 127], and for the Multilevel FLP [30].

Complete solutions. In a complete solution, each demand point is fully served by the most economical orders. This is formalized as following: for a fixed demand point $j = (q, t)$, consider a permutation π_j of warehouse orders in $\mathcal{P}_t$ such that the elements are listed in non-decreasing order of service cost, that is, if $\pi_j = ((p_1, s_1), \ldots, (p_k, s_k))$, where $k = |\mathcal{P}_t|$, then $c_{p_1 q} + h_{q s_1 t} \leq \cdots \leq c_{p_k q} + h_{q s_k t}$. Assume that for each demand point j, permutation π_j is unique, by breaking ties arbitrarily, but in a fixed way (two elements appear in the same order in all permutations in which they tie). A solution (x, y) of the LP is said to be *complete* if for every demand point (q, t) there is an index l, such that $x^{qt}_{p_i s_i} = y_{p_i s_i}$ if $i \leq l$, and $x^{qt}_{p_i s_i} = 0$ if $i > l$.

Any feasible solution can be transformed into a complete solution of no greater cost. Indeed, let (x, y) be a feasible solution of the LP. We create a new solution $(\bar{x}, \bar{y})$, such that $\bar{y} = y$, and $\bar{x}$ is given by serving each demand point $j = (q, t)$ greedily by the orders in the permutation π_j. More precisely, for each $(q, t) \in \mathcal{D}$, let l be minimum such that $\sum_{i=1}^{l} y_{p_i s_i} \geq 1$. Since (x, y) is feasible, we know that there is such an l. Now, we define $\bar{x}^{qt}_{p_i s_i} = y_{p_i s_i}$ for each $i \leq l$, and $\bar{x}^{qt}_{p_i s_i} = 0$ for each $i > l$. We assume without loss of generality that $(\bar{x}, \bar{y})$ is complete, that is, $\sum_{i=1}^{l} y_{p_i s_i} = 1$. In the case that the solution is not complete, we can always replace p_l by two warehouses p'_l and p''_l at the same location, and split its fractional ordering $y_{p_l s_l}$ between p'_l and p''_l, such that $y_{p'_l s_l} = 1 - \sum_{i=1}^{l-1} y_{p_i s_i}$, and $y_{p''_l s_l} = y_{p_l s_l} - y_{p'_l s_l}$. Repeating this for each demand point, we obtain an equivalent instance with a corresponding complete solution (the arguments are completely analogous to Lemma 1 of Sviridenko [127]).

Filtering. The idea of filtering is that, if (x, y) is a complete solution, then for each demand $j = (q, t)$ we can consider only the subset of orders in $\mathcal{S}_j$ that is the most economical. This subset is formed by the orders in the minimal prefix of permutation π_j that serves an α fraction of the demand. Formally, given a parameter $\alpha \in (0, 1]$, let l be the minimal index such that $\sum_{i=1}^{l} y_{p_i s_i} \geq \alpha$. The *$\alpha$-neighborhood* of a demand point j is the set $\mathcal{N}_j(\alpha) = \{(p_i, s_i) : i \leq l\}$. The radius $R_j(\alpha)$ of this neighborhood is the maximum cost paid to serve j by an order in $\mathcal{N}_j(\alpha)$, that is,

$$R_j(\alpha) = \max_{(p,s) \in \mathcal{N}_j(\alpha)} d_{qt}(c_{pq} + h_{qst}).$$

Intuitively, if we increase the value of some fractional order variables y_{st} in the LP solution, then the average service cost of a demand should decrease. Indeed, given a solution (x, y), the filtering technique consists of scaling up the y variables by $1/\alpha$ for some $\alpha \in (0, 1]$, then defining x such that each demand is fully satisfied by orders in its α-neighborhood. We obtain a complete solution $(\bar{x}, \bar{y})$ (by splitting warehouse fractional ordering if necessary). For a demand $j = (q, t)$, we denote the average service cost by

$$W_j(\alpha) = \sum_{(\hat{p}, \hat{s}) \in \mathcal{N}_j(\alpha)} \bar{x}^{qt}_{\hat{p}\hat{s}}\, d_{qt}(c_{\hat{p}q} + h_{q\hat{s}t}).$$

4.3 Approximation for the Metric PDP

4.3.1 Clustering

Many algorithms for the Metric FLP are based on a clustering technique. In such algorithms, we are given an optimal solution for the LP relaxation, and construct the support graph corresponding to this solution, that is, the bipartite graph that contains an edge for each pair of client and facility such that the client is fractionally served by the facility in the LP solution. In the support graph, two clients are called *neighbors* if they are adjacent to a common facility. A partition of the clients is then obtained, such that any client in a given cluster is neighbor to a leading client, that is called the *cluster center*. It is required that no two cluster centers are neighbors. The algorithms for the FLP use the following greedy procedure: while not all clients are clustered, choose a cluster center with a certain greedy criterion, and create a new cluster with this center and all its neighbors. Different greedy criteria lead to different algorithms and analyses.

We use a clustering algorithm for the PDP. However, in the PDP, we are not aiming to locate facilities to be opened, rather, the warehouses are already established, and we want to select the set of time steps at which we place orders for each warehouse. Therefore, we can think of an order formed by a pair of warehouse and time step as a single facility. Analogously, each demand can be thought of as a single client, that is willing to be connected to this "facility". We can then construct the corresponding support graph, and proceed to the clustering algorithm, in a way similar to facility location algorithms.

Formally, the support graph G is the bipartite graph such that the vertices are formed by the disjoint union $\mathcal{P} \cup \mathcal{D}$, and there is an edge between order $(p, s) \in \mathcal{P}$ and demand $(q, t) \in \mathcal{D}$ if $(p, s) \in \mathcal{S}_j$. Notice that, contrary to the case of the FLP, where a non-center client could always be indirectly connected to any facility that served the cluster center, for the PDP, it can happen that a non-center demand (q, t) cannot be served by an order adjacent to its cluster center (q', t'). This happens when demand (q, t) arrives before (q', t'), that is $t < t'$, and cluster center (q', t') is adjacent to some order (p, s) with $s > t$. To guarantee that every demand in a cluster is served, we place orders at the beginning of the cluster center's service window.

We will use the following clustering algorithm for the PDP.

Algorithm 4.1 (Clustering algorithm)

We are given a complete solution (x, y) for the LP relaxation, and an ordered list of demand points L. The algorithm returns a set F' of placed orders, and a clustering $\mathcal{C}$ of L.

1. Construct the support graph G.
2. While there are unclustered demands points:
   (a) Create cluster D with the next unclustered element j' in L as center.
   (b) Add all unclustered demand points that are neighbors of j' to D.
   (c) Add D to clustering $\mathcal{C}$.
3. For each cluster D with center j':
   (a) Choose one order $(\bar{p}, \bar{s}) \in \mathcal{S}_{j'}$ with probability $y_{\bar{p}\bar{s}}$.
   (b) Let $s' = \min_{(\hat{p},\hat{s})\in\mathcal{S}_{j'}} \hat{s}$, and $p' = \bar{p}$.
   (c) Add (p', s') to set F'.

Different choices of the list L lead to algorithms with different approximation guarantees. In Subsection 4.3.2, list L will be the set of demands in increasing order of $(C^*_{j'} + 2b^*_{j'})/d_{j'}$, where $b^*_{j'}$ corresponds to an optimal solution of (4.2) and in Subsection 4.3.3 the demands will be chosen (using filtering) by order of $(W_{j'}(\alpha) + 2R_{j'}(\alpha))/d_{j'}$, for some parameter α.

Suppose that we run Algorithm 4.1 for an optimal LP solution (x^*, y^*), and some arbitrary list L. Let $K_{F'} = \sum_{(p,s)\in F'} k_p$ be the total cost of the orders in the set F'. The next lemmas calculates the expected value of $K_{F'}$, and the expected cost to serve one unit of a demand point.

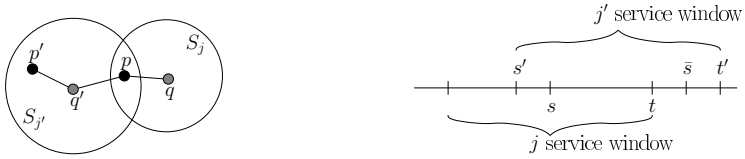
Lemma 4.1. *Let $K_{F'}$ be the ordering cost Algorithm 4.1, then $E[K_{F'}] \leq K^*$.*

Proof. For a cluster $D \in \mathcal{C}$, let j_D be its cluster center, and (p_D, s_D) be the order included in F' at step 3(c) of the algorithm. We obtain

$$\begin{aligned} E[K_{F'}] = E[\sum_{(p,s)\in F'} k_p] &= E[\sum_{D\in\mathcal{C}} k_{p_D}] \\ &= \sum_{D\in\mathcal{C}} \sum_{(p,s)\in\mathcal{S}_{j_D}} y^*_{ps}\, k_p \\ &\leq \sum_{(p,s)\in\mathcal{P}} y^*_{ps}\, k_p = K^*, \end{aligned}$$

where the inequality follows from the fact that the sets $\mathcal{S}_{j_D}$ are disjoint. □

In the following, we bound the expected cost to serve one demand point by the order placed for its corresponding cluster.



(a) In space (b) In time

Figure 4.2: Possible configuration for Lemma 4.2

Lemma 4.2. *Let $j = (q, t)$ be a demand point, and $j' = (q', t')$ be the corresponding cluster center. Then, $E\left[\min_{(p,s)\in F'}(c_{pq} + h_{qst})\right] \leq (C^*_{j'} + 2b^*_{j'})/d_{j'} + b^*_j/d_j$.*

Proof. Let (p', s') be the order placed by Algorithm 4.1 at step 3(c) corresponding to cluster center j', and $(\bar{p}, \bar{s})$ be the order drawn in step 3(a). It is enough to bound the expected cost to serve one unit of j by order (p', s').

Since j' is the cluster center corresponding to j, we know that there is an order $(p, s) \in \mathcal{S}_j \cap \mathcal{S}_{j'}$. Thus $s' \leq s$, since s is in the service window of j' and s' is the minimum time step in this service window. Also, we get $s \leq t$, because demand j is fractionally served by (p, s). Similarly, demand j' is fractionally served by (p, s), so $s \leq t'$ (see Figure 4.2). Using Assumption 4.2, we obtain

$$\begin{aligned} c_{p'q} + h_{qs't} &\leq c_{p'q'} + h_{q's's} + c_{pq'} + c_{pq} + h_{qst} \\ &\leq c_{p'q'} + h_{q's't'} + c_{pq'} + c_{pq} + h_{qst}, \end{aligned}$$

where the last inequality follows from Assumption 4.1 and the fact that $s \leq t'$. Since $(p, s) \in \mathcal{S}_j$, we obtain $x^{qt}_{ps} > 0$. By complementary slackness, it follows that $b^*_j = d_j(h_{qst} + c_{pq}) + z^{qt}_{ps}$, and thus $h_{qst} + c_{pq} \leq b^*_j/d_j$. Similarly, we also get $h_{q's't'} \leq b^*_{j'}/d_{j'}$, and $c_{pq'} \leq b^*_{j'}/d_{j'}$. Finally, the expected value of $c_{p'q'}$ is

$$E[c_{p'q'}] = E[c_{\bar{p}q'}] = \textstyle\sum_{(\hat{p},\hat{s})\in\mathcal{S}_{j'}} y^*_{\hat{p}\hat{s}}\, c_{\hat{p}q'} = \sum_{(\hat{p},\hat{s})\in\mathcal{S}_{j'}} x^{*q't'}_{\hat{p}\hat{s}}\, c_{\hat{p}q'} = C^*_{j'}/d_{j'},$$

where the third equality holds because the solution is complete. Adding up all terms, we obtain the desired statement. □

4.3.2 Balancing using extra orders

Lemmas 4.1 and 4.2 give bounds to the ordering and service costs of the solution yielded by Algorithm 4.1. One can observe the imbalance between the low value of ordering cost and the high value of the service cost. Indeed, Lemma 4.2 bounds the demand service cost in the worst case, when a demand point is served through its cluster center. For the FLP, the algorithms of Sviridenko [127] and Chudak and Shmoys [38] would first try to

connect a client to its close facilities. In fact, they would open the unclustered facilities after the clustering phase, so that facilities in service sets are opened independently, or with negative correlation, with probability equal to the fractional opening. This implies that there is, with constant probability, an open close facility.

For the PDP, however, this approach cannot be applied directly. The reason is that the orders placed for each cluster are moved to earlier times, and thus serving demand points by the corresponding orders would incur an extra holding cost, that is potentially unbounded. Instead, for the PDP, we place an extra set of orders, in their original time positions. This increases the total ordering cost, but such increase is compensated by the decrease of the service cost.

Next algorithm uses the clustering algorithm and places an extra set of warehouse orders in their original time steps.

Algorithm 4.2 (Balancing algorithm)

We are given an instance of the PDP. The algorithm returns a solution for this instance, formed by a set of orders $F' \cup F''$.

1. Solve the LP relaxation, and obtain solution (x^*, y^*) and b^*.
2. Make the solution complete, splitting fractional ordering if necessary.
3. Run Algorithm 4.1 using (x^*, y^*), and L as the set of demand points j' in increasing order of $(C^*_{j'} + 2b^*_{j'})/d_{j'}$. Obtain a set of orders F'.
4. For each $(p, s) \in \mathcal{P}$, add order (p, s) to set F'' with probability y^*_{ps}.
5. Place an order for each element of $F' \cup F''$.
6. Serve each demand point (q, t) by the order (p, s) that minimizes $c_{pq} + h_{qst}$.

The following lemma bounds the expected service cost of the cheapest placed order in a subset A of $\mathcal{P}$, conditioned to the event that there is one placed order in A. Since versions of this lemma have appeared in several LP rounding algorithms for the FLP (for instance, see Lemma 4.2 of Byrka and Aardal [28]), we omit the proof.

Lemma 4.3. *If A be a set of orders such that $\sum_{(p,s)\in A} y^*_{ps} > 0$, then*

$$E\left[\min_{(p,s)\in A\cap F''}(c_{pq} + h_{qst})\middle| A \cap F'' \neq \emptyset\right] \leq \frac{\sum_{(p,s)\in A} y^*_{ps}\,(c_{pq} + h_{qst})}{\sum_{(p,s)\in A} y^*_{ps}}.$$

Now, we may bound the cost of the solution produced by Algorithm 4.2.

Theorem 4.1. *The balancing algorithm produces a solution for the PDP with expected cost at most* $(2+\frac{3}{e})\,K^* + (1+\frac{3}{e})\,C^* + (1+\frac{2}{e})\,H^*$.

Proof. Let $K_{F''}$ be the cost of orders in F''. We obtain

$$E[K_{F''}] = E[\textstyle\sum_{(p,s)\in F''} k_p] = \sum_{(p,s)\in\mathcal{P}} y^*_{ps} k_p = K^*.$$

Consider a demand point $j=(q,t)$. We calculate the service cost c_j to serve j by a placed order in $F'\cup F''$ if we used the following, suboptimal, algorithm: if $\mathcal{S}_j \cap F''$ is not empty, then we serve j by the closest order in $\mathcal{S}_j\cap F''$, otherwise we serve it indirectly through its cluster center j'. Let p_c be the probability that $\mathcal{S}_j\cap F''\neq\emptyset$. We have $p_c = 1-\prod_{(p,s)\in\mathcal{S}_j}(1-y^*_{ps}) \geq 1-e^{-\sum_{(p,s)\in\mathcal{S}_j} y^*_{ps}} = 1-e^{-1}$. Now, combining with Lemmas 4.2 and 4.3, we obtain

$$\begin{aligned}
E[c_j] &\leq p_c \textstyle\sum_{(p,s)\in\mathcal{S}_j} y^*_{ps}\, d_j(c_{pq}+h_{qst}) + (1-p_c)\, d_j((C^*_{j'}+2b^*_{j'})/d_{j'} + b^*_j/d_j)\\
&\leq p_c\,(C^*_j+H^*_j) + (1-p_c)\, d_j((C^*_j+2b^*_j)/d_j + b^*_j/d_j)\\
&\leq (1-e^{-1})(C^*_j+H^*_j) + e^{-1}(C^*_j+3b^*_j),
\end{aligned}$$

where the second inequality follows since j' was chosen as cluster center, thus $(C^*_{j'}+2b^*_{j'})/d_{j'} \leq (C^*_j+2b^*_j)/d_j$, and the last inequality follows since, by complementary slackness, $C^*_j+H^*_j \leq C^*_j+3b^*_j$, and $p_c\geq 1-e^{-1}$. Finally, we get

$$\begin{aligned}
&E[K_{F'}] + E[K_{F''}] + \textstyle\sum_{j\in D} E[c_j]\\
&\leq K^* + K^* + \textstyle\sum_{j\in\mathcal{D}} \Big((1-e^{-1})(C^*_j+H^*_j) + e^{-1}(C^*_j+3b^*_j)\Big)\\
&= K^* + K^* + (1-e^{-1})(C^*+H^*) + e^{-1}(C^*+3(K^*+C^*+H^*))\\
&= (2+\frac{3}{e})\,K^* + (1+\frac{3}{e})\,C^* + (1+\frac{2}{e})\,H^*.
\end{aligned}$$

□

In particular, Theorem 4.1 implies that there is a randomized approximation for the PDP with factor $2+3/e \approx 3.10$.

4.3.3 Balancing using filtering

Other way to fix the imbalance between the production and the service cost is using the filtering technique: the fractional ordering y_{ps} is scaled by a factor $1/\alpha$, for some $\alpha\in(0,1]$. Notice that y_{ps} can become larger than 1; in this case, a copy of the warehouse p is made, and the fractional ordering y_{ps} is split in the filtering step. Intuitively, placing "more times" each warehouse order should increase the probability of a client being served by a cheap order.

Algorithm 4.3 (Filtering algorithm)

Given an instance of the PDP and a parameter $\alpha \in (0, 1]$, the algorithm returns a solution for this instance, formed by warehouses orders F'.

1. Solve the LP relaxation and obtain solution (x^*, y^*).
2. Scale up the ordering variables y^* by $1/\alpha$. Change variables x^*, and obtain a complete solution $(\bar{x}, \bar{y})$, splitting orders if necessary.
3. Run Algorithm 4.1 with solution $(\bar{x}, \bar{y})$, passing as list L the set of demand points j' in increasing order of $(W_{j'}(\alpha) + 2R_{j'}(\alpha))/d_j$, and obtain set F'.
4. Serve each demand point (q, t) by the order (p, s) that minimizes $h_{qst} + c_{pq}$.

The following lemmas are similar to Lemmas 4.1 and 4.2. Recall that $R_j(\alpha)$ is the maximum service cost in the α-neighborhood of demand j, and $W_j(\alpha)$ is the average service cost in this neighborhood.

Lemma 4.4. *Let $K_{F'}$ be the ordering cost of Algorithm 4.3, then $E[K_{F'}] \leq 1/\alpha\, K^*$.*

Proof. Similar to Lemma 4.1, but we use $\bar{y}_{ps}$, instead of y^*_{ps}. □

Lemma 4.5. *Let $j = (q, t)$ be a demand point, and let $j' = (q', t')$ be the cluster center corresponding to j. Then, $E\left[\min_{(p,s)\in F'}(c_{pq} + h_{qst})\right] \leq (W_{j'}(\alpha)+2R_{j'}(\alpha))/d_{j'} + R_j(\alpha)/d_j$.*

Proof. Let (p', s') be the order placed for j', and (p, s) be the common order in $\mathcal{N}_{j'}(\alpha)$ and $\mathcal{N}_j(\alpha)$. Similarly to Lemma 4.2, we can obtain $c_{p'q} + h_{qs't} \leq c_{p'q'} + h_{q's't'} + c_{pq'} + c_{pq} + h_{qst}$. Since $(p, s) \in \mathcal{N}_j(\alpha)$, we get $h_{qst} + c_{pq} \leq R_j(\alpha)/d_j$. Also, there is a $\hat{p}$ such that $(\hat{p}, s') \in \mathcal{N}_{j'}(\alpha)$, and $(p, s) \in \mathcal{N}_{j'}(\alpha)$, we get $h_{q's't'} \leq R_{j'}(\alpha)/d_{j'}$, and $c_{pq'} \leq R_{j'}(\alpha)/d_{j'}$. Finally, we bound the expected value of $c_{p'q'}$. We have

$$E[c_{p'q'}] = \sum_{(\hat{p},\hat{s})\in\mathcal{N}_{j'}(\alpha)} \bar{y}_{\hat{p}\hat{s}}\, c_{\hat{p}q'} = \sum_{(\hat{p},\hat{s})\in\mathcal{N}_{j'}(\alpha)} \bar{x}^{q't'}_{\hat{p}\hat{s}}\, c_{\hat{p}q'} \leq W_{j'}(\alpha)/d_{j'}.$$

The first equality is true since the service set of j' is $\mathcal{N}_{j'}(\alpha)$, and the second inequality comes from the fact that $(\bar{x}, \bar{y})$ is complete. The inequality is due the definition of $W_{j'}(\alpha)$. Now, we sum all terms, and obtain the lemma. □

Lemma 4.6. *If $j = (q, t)$, then $E\left[\min_{(p,s)\in F'} d_j(c_{pq} + h_{qst})\right] \leq W_j(\alpha) + 3R_j(\alpha)$.*

Proof. By Lemma 4.5, and then by the order of list L in Algorithm 4.3,

$$\begin{aligned} E\left[\min_{(p,s)\in F'}(c_{pq} + h_{qst})\right] &\leq (W_{j'}(\alpha) + 2R_{j'}(\alpha))/d_{j'} + R_j(\alpha)/d_j \\ &\leq (W_j(\alpha) + 3R_j(\alpha))/d_j. \end{aligned}$$

□

Lemma 4.4 shows that the approximation factor of the filtering algorithm for the ordering cost depends only on the parameter α. On the other hand, as it can be seen in Lemma 4.6, the service cost depends on the neighborhood radius function of each demand point j. Such dependency may be characterized by the summation of such radius functions, as in the following definition.

Definition 4.1. *Given an instance of the PDP, and an optimal solution* (x^*, y^*) *of* (4.1)*, the* characteristic function $r : [0, 1] \to \mathbb{Z}_+$ *is* $r(\alpha) = \sum_{j \in \mathcal{D}} R_j(\alpha)/S^*$.

Remark 4.1. *The characteristic function* $r(\alpha)$ *satisfies* $\int_0^1 r(t)\mathrm{d}t = 1$.

Proof. Notice that for each demand j, R_j is a piece-wise constant function. We have

$$\begin{aligned}\int_0^1 r(t)\mathrm{d}t &= (1/S^*) \sum_{(q,t)\in\mathcal{D}} \int_0^1 R_{qt}(t)\mathrm{d}t \\ &= (1/S^*) \sum_{(q,t)\in\mathcal{D}} \sum_{(p,s)\in\mathcal{S}_{qt}} x^{*qt}_{ps}\, d_{qt}(h_{qst} + c_{pq}) \\ &= (1/S^*) \sum_{(q,t)\in\mathcal{D}} \sum_{(p,s)\in\mathcal{P}_t} x^{*qt}_{ps}\, d_{qt}(h_{qst} + c_{pq}) = (1/S^*)S^* = 1. \qquad \square\end{aligned}$$

The expected service cost of can be bounded in a more concise way than in Lemma 4.6.

Corollary 4.1. *The expected service cost of Algorithm 4.3 is* $\left((1/\alpha)\int_0^\alpha r(t)\mathrm{d}t + 3r(\alpha)\right) S^*$.

Proof. From Lemma 4.6, we obtain

$$\begin{aligned}&E\left[\sum_{(q,t)\in\mathcal{D}} \min_{(p,s)\in F'} d_{qt}(c_{pq} + h_{qst})\right] \\ &\le \sum_{(q,t)\in\mathcal{D}} (W_{qt}(\alpha) + 3R_{qt}(\alpha)) \\ &= \sum_{(q,t)\in\mathcal{D}} \left(\left(\sum_{(\hat{p},\hat{s})\in\mathcal{N}_{qt}(\alpha)} \bar{x}^{qt}_{\hat{p}\hat{s}}\, d_{qt}(h_{q\hat{s}t} + c_{\hat{p}q})\right) + 3R_{qt}(\alpha)\right) \\ &= \sum_{(q,t)\in\mathcal{D}} \left(\left(\sum_{(\hat{p},\hat{s})\in\mathcal{N}_{qt}(\alpha)} (1/\alpha) x^{*qt}_{\hat{p}\hat{s}}\, d_{qt}(h_{q\hat{s}t} + c_{\hat{p}q})\right) + 3R_{qt}(\alpha)\right) \\ &= \sum_{(q,t)\in\mathcal{D}} \left((1/\alpha)\int_0^\alpha R_{qt}(t)\mathrm{d}t + 3R_{qt}(\alpha)\right) \\ &= (1/\alpha)\int_0^\alpha \sum_{(q,t)\in\mathcal{D}} R_{qt}(t)\mathrm{d}t + 3\sum_{(q,t)\in\mathcal{D}} R_{qt}(\alpha) \\ &= \left((1/\alpha)\int_0^\alpha r(t)\mathrm{d}t + 3r(\alpha)\right) S^*.\end{aligned}$$

The second equality holds because we can assume without loss of generality that $(\bar{x}, \bar{y})$ and (x^*, y^*) are complete. The third equality holds because $R_{qt}(\alpha)$ is piece-wise constant. $\square$

4.3.4 Combining different algorithms

Algorithm 4.2 is a bi-factor approximation algorithm, that achieves factor $2 + 3/e$ for the ordering cost, and factor $1 + 3/e$ for the service cost (*i.e.*, holding and transportation costs). Similarly, for each value of parameter α, Algorithm 4.3 is a bi-factor approximation with factors $1/\alpha$ and $(1/\alpha)\int_0^\alpha r(t)\mathrm{d}t + 3r(\alpha)$ for ordering and service costs, respectively.

To combine the two algorithms, we use the following strategy: with a given probability δ, we run Algorithm 4.2, and with probability $1-\delta$ we run Algorithm 4.3 with parameter α drawn from a probability density function $f : (0, 1] \to \mathbb{R}_+$. A similar approach has been done in the algorithm by Li [93], for the FLP. Let SOL be the cost corresponding to solution produced by this algorithm, thus

$$E[\mathrm{SOL}] \leq (A_2)\ \delta + \left(\int_0^1 A_3(\alpha) f(\alpha)\mathrm{d}\alpha\right)\ (1-\delta), \tag{4.3}$$

where $A_2 = (2 + \frac{3}{e})K^* + (1 + \frac{3}{e})S^*$ is the expected cost of the solution produced by Algorithm 4.2, and $A_3(\alpha) = 1/\alpha\, K^* + ((1/\alpha)\int_0^\alpha r(t)\mathrm{d}t + 3r(\alpha))\, S^*$ is the expected cost of the solution produced by Algorithm 4.3 when it is run with parameter α.

For simplicity, we let $g(\alpha) = (1-\delta)f(\alpha)$, so that $\delta + \int_0^1 g(\alpha)\mathrm{d}\alpha = 1$. We may rewrite (4.3) as $E[\mathrm{SOL}] \leq \beta(\delta, g)K^* + \gamma(\delta, g, r)S^*$, where $\beta(\delta, g)$ and $\gamma(\delta, g, r)$ are the obtained approximation factors for ordering and service costs of the combining algorithm, respectively. We obtain

$$\beta(\delta, g) = (2 + \frac{3}{e})\delta + \int_0^1 \frac{1}{\alpha}\, g(\alpha)\mathrm{d}\alpha, \tag{4.4}$$

$$\gamma(\delta, g, r) = (1 + \frac{3}{e})\delta + \int_0^1 \left((1/\alpha)\int_0^\alpha r(t)\mathrm{d}t + 3r(\alpha)\right)\, g(\alpha)\mathrm{d}\alpha. \tag{4.5}$$

We define $g(\alpha)$ and δ as follows, where $\alpha_0 \in (0, 1]$, and $c > 0$ are constants to be defined later:

$$g(\alpha) = \begin{cases} 0 & \alpha < \alpha_0 \\ c\,\alpha^{\frac{1}{3}} & \alpha \geq \alpha_0 \end{cases} \quad \text{and} \quad \delta = 1 - \int_0^1 g(\alpha)\mathrm{d}\alpha. \tag{4.6}$$

Substituting $g(\alpha)$ (notice that $g(\alpha)$ is non-zero only for $\alpha \in [\alpha_0, 1]$), we can simplify the integral in (4.5) by using integration by parts:

$$\int_0^1 \left((1/\alpha)\int_0^\alpha r(t)\mathrm{d}t + 3r(\alpha)\right) g(\alpha)\mathrm{d}\alpha = \int_{\alpha_0}^1 \int_0^\alpha r(t)\mathrm{d}t\ c\alpha^{-\frac{2}{3}}\mathrm{d}\alpha + \int_{\alpha_0}^1 3r(\alpha)c\alpha^{\frac{1}{3}}\mathrm{d}\alpha$$
$$= \left(\int_0^\alpha r(t)\mathrm{d}t\right) \cdot 3c\alpha^{\frac{1}{3}}\Big|_{\alpha_0}^1 - \int_{\alpha_0}^1 3c\alpha^{\frac{1}{3}}\, r(\alpha)\mathrm{d}\alpha + \int_{\alpha_0}^1 3r(\alpha)c\alpha^{\frac{1}{3}}\mathrm{d}\alpha = 3c\int_0^\alpha r(t)\mathrm{d}t \cdot \alpha^{\frac{1}{3}}\Big|_{\alpha_0}^1,$$

Now, we obtain calculating the definite integrals that

$$\beta(\delta, g) = (2 + \frac{3}{e})\delta + 3c(1^{\frac{1}{3}} - \alpha_0^{\frac{1}{3}}), \tag{4.7}$$

$$\begin{aligned}\gamma(\delta, g, r) &= (1 + \frac{3}{e})\delta + 3c\left[\int_0^1 r(t)\mathrm{d}t \cdot 1^{\frac{1}{3}} - \int_0^{\alpha_0} r(t)\mathrm{d}t \cdot \alpha_0^{\frac{1}{3}}\right] \\ &\leq (1 + \frac{3}{e})\delta + 3c, \end{aligned} \tag{4.8}$$

where the inequality comes from Remark 4.1, and the fact that $r(t)$ is nonnegative. Notice that the last expression bounds $\gamma(\delta, g, r)$, and does not depend on r, so it is independent of the input instance. By appropriately choosing c and α_0, we are ready to give an approximation factor for the combining algorithm.

Theorem 4.2. *There exists a pair (δ, g) such that, for any characteristic function r, we obtain that* $\max\{\beta(\delta, g), \gamma(\delta, g, r)\} \leq 2.77$.

Proof. Equating (4.7) and (4.8), and substituting δ from (4.6), we can obtain c as a function of α_0, and get

$$c = \frac{4/3}{1 + 4\alpha_0^{1/3} - \alpha_0^{4/3}}.$$

Replacing this in (4.7), we thus obtain

$$\gamma(\delta, g, r) \leq \beta(\delta, g) = \frac{4\left(1 + \alpha_0^{1/3}(1 + 3/e)\right)}{1 + 4\alpha_0^{1/3} - \alpha_0^{4/3}}.$$

Choosing $\alpha_0 = 0.23947$, we get $c \approx 0.399771$, and $\beta(\delta, g) \approx 2.76602 < 2.77$. □

4.4 The PDP with retailer ordering costs

In this section, we consider a variant of the PDP that includes a possibly non-zero cost k_q for each time the retailer q places an order. This captures the situation when there is a setup cost associated with an order at a retailer. In addition to Assumption 4.2, we also assume traditional holding costs.

Assumption 4.3. *For each retailer q, there are nonnegative numbers h_{qi}, for $i = 1, \ldots, T$, such that for each s, t, it holds $h_{qst} = \sum_{i=s}^{t} h_{qi}$.*

A minor modification of the program (4.1) can formulate the PDP with retailer ordering cost. For the sake of clarity, we give the complete formulation here. In the following,

let $\mathcal{Q} = Q \times [T]$ be the set of all potential retailer orders.

$$\begin{aligned}
\text{minimize} \quad & \sum_{(p,s)\in\mathcal{P}} y_{ps}\, k_p + \sum_{(q,s)\in\mathcal{Q}} y_{qs}\, k_q + \sum_{(q,t)\in\mathcal{D}} \sum_{(p,s)\in\mathcal{P}_t} x^{qt}_{ps}\, d_{qt}(h_{qst} + c_{pq}) \\
\text{subject to} \quad & \sum_{(p,s)\in\mathcal{P}_t} x^{qt}_{ps} \geq 1 && (q,t) \in \mathcal{D} \\
& x^{qt}_{ps} \leq y_{ps} && (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P}_t \\
& \textstyle\sum_{p\in P} x^{qt}_{ps} \leq y_{qs} && (q,t) \in \mathcal{D}, s \in [t] \\
& x^{qt}_{ps} \geq 0 && (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P}_t \\
& y_{ps}, y_{qs} \in \{0,1\} && (q,t) \in \mathcal{D}, (p,s) \in \mathcal{P}, (q,s) \in \mathcal{Q}
\end{aligned} \tag{4.9}$$

A linear relaxation can be obtained by replacing the integrality constraints by constraints $y_{ps} \geq 0$ and $y_{qs} \geq 0$ for all $(p,s) \in \mathcal{P}$ and $(q,s) \in \mathcal{Q}$. Besides the set of placed warehouse orders, a solution to (4.9) also includes the sets of orders placed for each retailer. Notice that a solution for (4.1) induces a solution for (4.9) such that a retailer order (q,s) is placed if a warehouse order (p,s) serves a demand point (q,t). Therefore, one may consider using Algorithm 4.1 to solve the PDP with retailer ordering costs, by returning such induced solution. However, the incurred cost to place a retailer order for each demand is potentially unbounded. This suggests that, to use Algorithm 4.1, we need to preprocess the input, so that each retailer order can serve a group of demand points. In Subsection 4.4.1, we review the notion of complete solutions considering retailer orders, in Subsection 4.4.2, we describe the preprocessing procedure, and, in Subsection 4.4.3, we give the modified clustering and filtering algorithm for the PDP with retailer orders.

4.4.1 Complete solutions

The notion of complete solutions for the PDP with non-zero retailer ordering costs is slightly different from the basic notion of complete solutions described in Subsection 4.2.3. Unlikely the case of program (4.1), in an optimal fractional solution of program (4.9), a demand point is not necessarily served by the cheapest warehouses orders. The reason is that the total fractional warehouse ordering in a given time step can be strictly greater than the fractional retailer ordering in this time. We extend the notion of complete solutions in the following.

Consider a solution (x,y) of the relaxation of (4.9). For each retailer q, we consider the permutation $p_1, \ldots, p_{|P|}$ of warehouses P in non-decreasing order of distance from retailer q, that is, $c_{p_1 q} \leq \cdots \leq c_{p_{|P|} q}$. Also, for a given time step s, let l be the minimum index such that $\sum_{i=1}^{l} y_{p_i s} \geq y_{qs}$. If there is no such l, then we can decrease the value y_{qs} and maintain feasibility. We assume that $\sum_{i=1}^{l} y_{p_i s} = y_{qs}$, since otherwise we could make

copies of warehouse p_l, and split its fractional warehouse ordering, obtaining an equivalent solution for which this is true (this is similar to the process described in Subsection 4.2.3). Now we may define a new variable y_{p_iqs}, for each q and s, as follows:

$$y_{p_iqs} = \begin{cases} y_{p_is} & i \leq l, \\ 0 & i > l. \end{cases}$$

Intuitively, a demand point (q,t) will only be served by the warehouse orders (p,s) for which $y_{pqs} = y_{ps}$, since it cannot be fractionally served by more than a fraction y_{qs} at time s. Therefore, for each retailer q, we only need to consider the warehouse orders (p,s) such that $y_{pqs} > 0$, so we will concentrate on the set $\mathcal{P}_{qt} = \{(p,s) \in \mathcal{P}_t : y_{pqs} > 0\}$.

Now, for a given demand $j = (q,t)$, consider a permutation of $\mathcal{P}_{qt}$ in non-decreasing order of service cost, $\pi_j = ((p_1,s_1),\ldots,(p_k,s_k))$, where $k = |\mathcal{P}_{qt}|$. A fractional solution (x,y) of (4.9) is said to be *complete* if for every demand point (q,t) there is an index l, such that $x^{qt}_{p_is_i} = y_{p_is_i}$ if $i \leq l$, and $x^{qt}_{p_is_i} = 0$ if $i > l$. Here again we can always transform any feasible solution into a complete solution by making copies of warehouses and splitting fractional warehouse ordering, as done in Subsection 4.2.3.

For a given parameter $\alpha \in (0,1]$, we also define the α-neighborhood set, its radius, and its average service cost, $N_j(\alpha)$, $R_j(\alpha)$, and $W_j(\alpha)$, respectively, as in Subsection 4.2.3, but considering set $\mathcal{P}_{qt}$, instead of $\mathcal{P}_t$.

4.4.2 Preprocessing

To bound the retailer ordering cost, we will use a preprocessing phase, that performs a grouping of demand points. The idea is to group demand points in each retailer, so that each group shares a positive fraction of the retailer ordering cost. After this pre-clustering step, we choose a representative demand point for each group, and run Algorithm 4.1 using only representative demand points. After the clustering algorithm is run, each representative demand point is served by one place warehouse order. Finally, we place a retailer order for each representative demand point, at the same time of its associated warehouse order. Since each representative demand point "owns" a share of the LP fractional retailer ordering cost, the cost of such placed retailer orders can be bounded. Non-representative demand points are served indirectly through their representatives, and thus no additional retailer orders are necessary.

The grouping algorithm is inspired by the *random shift points procedure*, that is used in the approximation for the OWMR problem by Levi *et al.* [91]. The shift point procedure places a retailer order at the so called shift points. Two shift points are away from each other by a time interval such that the total fractional retailer ordering in it adds up to a constant fraction $\alpha \in (0,1]$. Next, the grouping algorithm is described precisely.

Algorithm 4.4 (Grouping algorithm)

We are given as input the LP solution (x, y) and a value $\alpha \in (0, 1]$. The output of this procedure is a partition $\mathcal{G}^q$ of demand points for each retailer q. Each such group $G \in \mathcal{G}^q$ is associated with a representative demand point, namely r_G. The set of representative demand points is $\mathcal{R} = \bigcup_{q \in Q} \mathcal{R}^q$, where $\mathcal{R}^q$ is the set of all representatives of retailer q.

For each retailer q, we consider the interval $(0, \sum_{t=1}^{T} y_{qt}]$, that corresponds to the total sum of fractional retailer ordering, and, for each time $s \in [T]$, we consider the associated subinterval $Y_{qs} = (\sum_{t=1}^{s-1} y_{qt}, \sum_{t=1}^{s} y_{qt}]$. Let also V be the largest integer multiple of α that is not greater than $\sum_{t=1}^{T} y_{qt}$, that is, $V = \lfloor 1/\alpha \sum_{t=1}^{T} y_{qt} \rfloor$. Then we consider the set of points $S^q = \{\alpha, 2\alpha, \ldots, V\alpha\}$. Notice that these points are contained in the interval $(0, \sum_{t=1}^{T} y_{qt}]$, and thus each point $x \in S^q$ is contained in exactly one subinterval Y_{qm}, for some time $m = 1, \ldots, T$. For each $l = 1, \ldots, V$, let m_l be the unique index such that $l\alpha \in Y_{qm_l}$. For simplicity, we also let $m_{V+1} = T + 1$. We will create a cluster containing all demand points that are between two consecutive index points m_l and m_{l+1}. Formally, for each $l = 1, \ldots, V$, we define $G_l = \{(q, t') \in \mathcal{D} : m_l \leq t' < m_{l+1}\}$, and add G_l to $\mathcal{G}^q$ if G_l is not empty. We let the representative r_G of a group $G \in \mathcal{G}^q$ be the demand point $(q, t) \in G$ with minimum t.

The grouping algorithm associates to each group G a share α of the fractional retailer ordering cost. Since we will place at most one retailer order per cluster, the total ordering cost is bounded by a factor of the LP ordering cost. However, this preprocessing step requires that all demand points $j \in G$ are served by the same order, that is, the order chosen for the group representative. The intuition is that the orders that are cheap for the group representative r_G should also be cheap for a demand point j in the same group G. This is formalized by the following lemma. Recall that $\mathcal{N}_j(\alpha)$ is the α-neighborhood of a demand point j.

Lemma 4.7. *Let $\beta \leq 1 - \alpha$, and G be a group obtained by Algorithm 4.4. If $j = (q, t)$ is a demand point in G, and $j' = (q, t')$ is the representative of G, then $\mathcal{N}_{j'}(\beta) \subseteq \mathcal{N}_j(\alpha + \beta)$.*

Proof. Let $j = (q, t)$ and $j' = (q, t')$. Since $j, j' \in G$, and j' is the representative of G, we know that $t' \leq t$. Let $\pi_{j'} = (p_1, s_1), \ldots, (p_k, s_k)$, where $k = |\mathcal{P}_{qt'}|$, be a permutation of orders $\mathcal{P}_{qt'}$ in non-decreasing order of the service cost of j'.

Let $S_1 = \mathcal{N}_j(\alpha + \beta) \setminus \mathcal{P}_{qt'}$ be the set of orders in neighborhood of j that cannot serve demand point j', and $S_2 = \mathcal{N}_j(\alpha + \beta) \setminus S_1$ be the complementary set (see Figure 4.3). We get $\sum_{(p,s) \in S_1} y_{pqs} \leq \sum_{s \in (t', t]} \sum_{p \in P} y_{pqs} \leq \sum_{s \in (t', t]} y_{qs} \leq \alpha$, where the last inequality holds since demand points j and j' are in the same group G. By definition, we have that

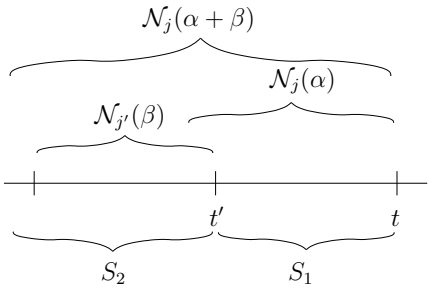


Figure 4.3: Shared neighborhood in retailer q over time

$\sum_{(p,s)\in\mathcal{N}_j(\alpha+\beta)} y_{pqs} = \sum_{(p,s)\in S_1} y_{pqs} + \sum_{(p,s)\in S_2} y_{pqs} \geq \alpha+\beta$. It follows that $\sum_{(p,s)\in S_2} y_{pqs} \geq \beta$. Since S_2 is a subset of $\mathcal{P}_{qt'}$ (and thus of $\pi_{j'}$), it follows that j is served by a fraction of at least β of orders in $\pi_{j'}$. In the following, we show that these orders correspond to the first orders of permutation $\pi_{j'}$.

First notice that Assumption 4.3 implies that $\pi_{j'}$ is also ordered in non-decreasing order of service cost of demand j. Indeed, for each pair of indices i, i', such that $1 \leq i < i' \leq k$, we have

$$\begin{aligned} c_{p_iq} + h_{qs_it} &= c_{p_iq} + h_{qs_it'} + h_{q,t'+1,t} \\ &\leq c_{p_{i'}q} + h_{qs_{i'}t'} + h_{q,t'+1,t} \\ &= c_{p_{i'}q} + h_{qs_{i'}t}. \end{aligned}$$

Since S_2 corresponds to the orders of $\mathcal{P}_{qt'}$ with smallest service cost of j, we know that S_2 must be the first orders of $\pi_{j'}$, that is, $S_2 = \{(p_1, s_1), \ldots, (p_m, s_m)\}$, for some $m \leq k$.

Finally, we obtain $\beta \leq \sum_{(p,s)\in S_2} y_{pqs} = \sum_{i=1}^{m} y_{p_iqs_i}$. From the definition, we have $\mathcal{N}_{j'}(\beta) = \{(p_1, s_1), \ldots, (p_l, s_l)\}$, where $l \leq k$ is minimum index such that $\sum_{i=1}^{l} y_{p_iqs_i} \geq \beta$. So, by the minimality of l, we must have $m \geq l$, and thus $\mathcal{N}_{j'}(\beta) \subseteq S_2 \subseteq \mathcal{N}_j(\alpha+\beta)$. □

4.4.3 Filtering and clustering

Lemma 4.7 shows that the β-neighborhood of a representative demand point is contained in the $(\alpha+\beta)$-neighborhood of a non-representative demand point in the same group. The algorithmic consequence of this lemma is that, if the representative demand point j' of a group G is served by some order with service cost bounded by a constant factor of $R_{j'}(\beta)$, then a demand point $j \in G$ can be served by the same order, with service cost bounded by a factor of $R_j(\alpha+\beta)$. To guarantee that each representative j' is served paying at most a factor of $R_{j'}(\beta)$, a filtering step will be used, and then the clustering algorithm is used to serve representative demand points. Notice that using filtering is a necessary step of the algorithm, since we must have β strictly less than 1 to use Lemma 4.7.

The complete algorithm is detailed in the following.

Algorithm 4.5 (Filtering and clustering algorithm)

Given an instance of the PDP, the algorithm returns a solution formed by warehouses orders F', and retailer orders $E = \bigcup_{q \in Q} E^q$, where E^q is the set of orders placed for retailer q.

1. Solve the LP relaxation and obtain solution (x^*, y^*).
2. Run Algorithm 4.4, and obtain the set of representatives $\mathcal{R}$.
3. Scale up the ordering variables y by $1/\beta$ and change variables x to obtain a complete solution $(\bar{x}, \bar{y})$, splitting fractional warehouses ordering if necessary.
4. Run Algorithm 4.1 over the set of representatives $\mathcal{R}$, passing as list L the set of representatives j' in order of $(W_{j'}(\beta) + 2R_{j'}(\beta))/d_{j'}$. Obtain a set of warehouse orders F', and a clustering $\mathcal{C}$ of representative demand points.
5. Place a warehouse order for each element of F'.
6. For each cluster $C \in \mathcal{C}$, and for each representative demand $(q, t) \in C$:
   - Let (p, s) be the warehouse order placed for cluster C.
   - Add retailer order (q, s) to E^q.
7. Serve each demand point $(q, t) \in \mathcal{D}$ by an order $(p, s) \in F'$ such that $(q, s) \in E^q$ that minimizes the sum $h_{qst} + c_{pq}$.

Let $K_{F'} = \sum_{(p,s) \in F'} k_p$ be the cost of warehouse orders in F', and $K_E = \sum_{q \in Q} |E_q| \cdot k_q$ be the cost of retailer orders in E. Let K^* and E^* be the warehouse and retailer ordering costs in the LP solution, respectively. The next lemma calculates the expected ordering cost.

Lemma 4.8. *$E[K_{F'}] \leq 1/\beta \, K^*$ and $K_E \leq 1/\alpha \, E^*$.*

Proof. For each retailer q, we have $|E_q| \leq |\mathcal{R}^q|$, since each retailer order corresponds to a representative demand point. Each representative demand point in $\mathcal{R}^q$ corresponds to one shift point in S^q in Algorithm 4.4, so $|E_q| \leq |S^q| = \lfloor 1/\alpha \sum_{t=1}^{T} y_{qt} \rfloor$. It follows that $K_E \leq \sum_{q \in Q} |E_q| \cdot k_q \leq \sum_{q \in Q} 1/\alpha \sum_{t=1}^{T} y_{qt} k_q = 1/\alpha \, E^*$.

Now let $\bar{K}$ be the cost of warehouse ordering by the scaled solution obtained in step 3 of Algorithm 4.5, so $\bar{K} = 1/\beta \, K^*$. Using the same arguments of Lemma 4.1, we obtain $E[K_{F'}] \leq \bar{K} = 1/\beta \, K^*$. □

Now we bound the service cost of a demand point. Recall that S_j^* is the fractional service cost of j in the solution (x^*, y^*), and $R_j(\alpha+\beta)$ is the maximum service cost of j in the $(\alpha+\beta)$-neighborhood of j.

Lemma 4.9. *Let $j=(q,t)$ be a demand point. Then, the expected service cost is such that $E\left[\min_{(p,s)\in F':(q,s)\in E^q} d_j(c_{pq}+h_{qst})\right] \le 1/\beta\, S_j^* + 3R_j(\alpha+\beta)$.*

Proof. Let $j'=(q,t')$ be the representative demand point of group G that contains demand point j, and let $j''=(q'',t'')$ be the cluster center corresponding to representative j'.

Now let $(p',s')\in F'$ be the warehouse order placed by the clustering algorithm for cluster center j''. By step 6 of Algorithm 4.5, there is a retailer order $(q,s')\in E^q$ placed to satisfy representative demand point j'. We have $s'\le t'\le t$, so demand point j can be served by the order (p',s'). Therefore, it is enough to bound the cost of serving demand j by this order.

Since j' and j'' are neighbors in the support graph, there exists a common (p,s) in the service sets of j' and j''. Using Assumption 4.2, we obtain

$$\begin{aligned} c_{p'q}+h_{qs't} &\le c_{p'q''}+h_{q''s's}+c_{pq''}+c_{pq}+h_{qst} \\ &\le c_{p'q''}+h_{q''s't''}+c_{pq''}+c_{pq}+h_{qst}, \end{aligned}$$

where the last inequality follows by the monotonicity of h. Since $(p',s')\in\mathcal{N}_{j''}(\beta)$, we get $h_{q''s't''}\le R_{j''}(\beta)/d_{j''}$. Similarly, we also get $c_{pq''}\le R_{j''}(\beta)/d_{j''}$. As $(p,s)\in\mathcal{N}_{j'}(\beta)$, by Lemma 4.7, we obtain $(p,s)\in\mathcal{N}_j(\alpha+\beta)$, and thus $c_{pq}+h_{qst}\le R_j(\alpha+\beta)/d_j$. The expected value of $c_{p'q''}$ is

$$E[c_{p'q''}] = \sum_{(\hat{p},\hat{s})\in\bar{\mathcal{N}}_{j''}(\beta)} \bar{y}_{\hat{p}q\hat{s}}\, c_{\hat{p}q''} = \sum_{(\hat{p},\hat{s})\in\bar{\mathcal{N}}_{j''}(\beta)} \bar{x}_{\hat{p}\hat{s}}^{q''t''} c_{\hat{p}q''} \le W_{j''}(\beta)/d_{j''},$$

where the last equality comes from the fact that $(\bar{x},\bar{y})$ is complete. Adding up, and using the fact that j' is a cluster center, we obtain

$$\begin{aligned} E[c_{p'q}+h_{qs't}] &\le W_{j''}(\beta)/d_{j''} + 2R_{j''}(\beta)/d_{j''} + R_j(\beta+\alpha)/d_j \\ &\le W_{j'}(\beta)/d_{j'} + 2R_{j'}(\beta)/d_{j'} + R_j(\beta+\alpha)/d_j. \end{aligned} \tag{4.10}$$

Using Lemma 4.7 and $t'\le t$, we can bound the first term of (4.10) as

$$\begin{aligned} W_{j'}(\beta)/d_{j'} &= \left(\textstyle\sum_{(\hat{p},\hat{s})\in\mathcal{N}_{j'}(\beta)} \bar{y}_{\hat{p}q\hat{s}}\, d_{j'}(c_{\hat{p}q}+h_{q\hat{s}t'})\right)/d_{j'} \\ &\le \textstyle\sum_{(\hat{p},\hat{s})\in\mathcal{N}_j(\alpha+\beta)} \bar{y}_{\hat{p}q\hat{s}}\,(c_{\hat{p}q}+h_{q\hat{s}t}) \\ &= \textstyle\sum_{(\hat{p},\hat{s})\in\mathcal{N}_j(\alpha+\beta)} 1/\beta\, x^{*qt}_{\hat{p}\hat{s}}\,(c_{\hat{p}q}+h_{q\hat{s}t}) \\ &\le 1/\beta\, \textstyle\sum_{(\hat{p},\hat{s})\in\mathcal{P}_t} x^{*qt}_{\hat{p}\hat{s}}\,(c_{\hat{p}q}+h_{q\hat{s}t}) \\ &\le 1/\beta\, S_j^*/d_j, \end{aligned}$$

where the second equality comes from the fact that solutions $(\bar{x}, \bar{y})$ and (x^*, y^*) are complete. Similarly, we can bound the second term,

$$\begin{aligned} 2R_{j'}(\beta)/d_{j'} &= 2\left(\max_{(\hat{p},\hat{s})\in\mathcal{N}_{j'}(\beta)} d_{j'}(c_{\hat{p}q} + h_{qst'})\right)/d_{j'} \\ &\leq 2 \max_{(\hat{p},\hat{s})\in\mathcal{N}_j(\alpha+\beta)} (c_{\hat{p}q} + h_{qst}) \\ &= 2R_j(\alpha+\beta)/d_j. \end{aligned}$$

Summing all terms, we obtain the lemma. □

Theorem 4.3. *There is a randomized 5-approximation for the PDP with retailer ordering costs.*

Proof. We set $\alpha = \beta = 1/2$ and run Algorithm 4.5. Since $\alpha + \beta \leq 1$, for each demand point j, $\mathcal{N}_j(\beta+\alpha) \subseteq \mathcal{S}_j$, then, by complementary slackness, we get $R_j(\beta+\alpha) \leq b_j^*$, where b_j^* is the dual variable corresponding to j in the optimal solution. Lemmas 4.8 and 4.9 imply that the expected solution cost SOL is

$$\begin{aligned} E[\text{SOL}] &\leq 1/\beta\, K^* + 1/\alpha\, E^* + \textstyle\sum_{j\in\mathcal{D}}(1/\beta\, S_j^* + 3R_j(\beta+\alpha)) \\ &\leq 2\, K^* + 2\, E^* + \textstyle\sum_{j\in\mathcal{D}}(2\, S_j^* + 3b_j^*) \\ &= 2\, K^* + 2\, E^* + 2\, S^* + 3(K^* + E^* + S*) \\ &= 5\,(K^* + E^* + S^*). \end{aligned}$$

□

4.5 A primal-dual algorithm for OWMR

The 1.8-approximation for OWMR by Levi *et al.* [91] assumes that each retailer satisfies one strict monotonicity property. Namely, they assume that each retailer is either a *J-retailer*, or a *W-retailer*. In a J-retailer, if the time that an item leaves the warehouse is delayed, so that the interval it is held in the warehouse stock is extended, then the total holding cost incurred by this item in the warehouse and in the retailer may only increase. In the opposite direction, in a W-retailer, the holding cost incurred by an item may only decrease if the delivery is delayed. These restrictions create a dependency between warehouse and retailer holding costs that is unnatural for many applications. Stauffer *et al.* [124] considered a slightly more relaxed holding cost structure, but also with dependent warehouse and retailer costs. In this section, it is considered a more general holding cost structure, that separates the holding cost incurred by warehouse and retailers. We present an approximation algorithm based on the primal-dual approach, that was not known before.

Problem's definition. In the OWMR problem there is one warehouse, indexed by number zero, and N retailer locations, that are indexed by integers $1, \ldots, N$. Each retailer i faces a demand of d_{it} units over a finite planning horizon with T time steps, and such demand may be satisfied only with items that are currently in the retailer inventory. For a pair (i, t), let r, s be such that $r \leq s \leq t$. If one unit of item is ordered at time r in the warehouse, transported to the retailer i at time s, and delivered at time t, then there is an incurred holding cost h_{rs}^{it}. Every time that the warehouse receives an order, there is a setup ordering cost K_0, that is independent of the number of ordered items. Similarly, every time that a retailer receives the items from the warehouse, there is a setup ordering cost K_i, that is independent of the number of items. The objective is to satisfy all demand points, minimizing the overall holding and ordering costs.

The OWMR problem admits a natural integer linear program formulation, with variables y_{is} that indicate whether there is an order to location i at time s, and variables x_{rs}^{it} that indicate whether demand point (i, t) is satisfied by an item ordered at time r at the warehouse, and transported to the retailer at time s. The relaxation of such integer program is given below. For simplicity, here it is assumed that demands are binary, that is, for each pair (i, t), $d_{it} = 0$, or $d_{it} = 1$. This is without loss of generality, since one can scale the holding cost function otherwise.

$$
\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{r=1}^{T} y_{0r} K_0 + \sum_{i=1}^{N}\sum_{s=1}^{T} y_{is} K_i + \sum_{i=1}^{N}\sum_{s=1}^{T}\sum_{r,s:r\leq s\leq t} x_{rs}^{it} h_{rs}^{it} & \\
\text{subject to} & \sum_{r,s:r\leq s\leq t} x_{rs}^{it} = 1 & i \in [N],\, t \in [T],\, d_{it} > 0 \\
& \sum_{r:r\leq s} x_{rs}^{it} \leq y_{is} & i \in [N],\, t \in [T],\, s \in [t] \\
& \sum_{s:r\leq s\leq t} x_{rs}^{it} \leq y_{0r} & i \in [N],\, t \in [T],\, r \in [t] \\
& x_{rs}^{it}, y_{ir} \geq 0 & i \in [0, N],\, r \in [T],\, s \in [r, T],\, t \in [s, T]
\end{array}
$$

The dual program is given next.

$$
\begin{array}{lll}
\text{maximize} & \displaystyle\sum_{i=1}^{N}\sum_{s=1}^{T} b^{it} & \\
\text{subject to} & b^{it} \leq h_{rs}^{it} + l_s^{it} + z_r^{it} & i \in [N],\, t \in [T],\, r \in [t],\, s \in [r, t] \\
& \sum_{t=s}^{T} l_s^{it} \leq K_i & i \in [N],\, s \in [T] \\
& \sum_{i=1}^{N}\sum_{t=r}^{T} z_r^{it} \leq K_0 & r \in [T] \\
& l_s^{it}, z_r^{it} \geq 0 & i \in [N],\, r \in [T],\, s \in [T],\, t \in [s, T]
\end{array}
$$

4.5.1 Holding cost model

Consider a demand point (i,t) (for item i at time t). We say that this demand is served by the pair of orders $\lceil r,s \rfloor$ if the items that serve such demand were first ordered at time r by the warehouse, later ordered at time s by the retailer, and finally delivered at time t to the client. Each such item has been held in the warehouse inventory during the period $[r,s)$, and in the retailer inventory during the period $[s,t]$. Let h^{it}_{rs} be the total cost of holding one unit of demand (i,t) served by the pair $\lceil r,s \rfloor$. We define the unit holding cost function as $h^{it}_{rs} = f^{it}_{rs} + g^{it}_{s}$, where f^{it}_{rs} is the cost of holding this unit at the warehouse, and g^{it}_{s} is the cost of holding this unit at the retailer. The following properties are assumed:

1. If $r = s$, then $f^{it}_{rs} = 0$. This means that the warehouse holding cost is zero if the item was not held in the warehouse inventory.

2. If $[r,s] \subseteq [r',s']$, then $f^{it}_{rs} \leq f^{it}_{r's'}$. This means that storing an item in the warehouse inventory at an earlier time r', or removing it at a later time s' cannot decrease the cost.

3. If $s < s'$, then $g^{it}_{s} \geq g^{it}_{s'}$. This means that storing an item in the retailer inventory at a later time s' cannot increase the cost.

4. Each demand point (i,t) has one of the following properties:

   - For every r,s,s', such that $r \leq s < s' \leq t$, we have $h^{it}_{rs} \leq h^{it}_{rs'}$. That is, h^{it}_{rs} is monotonically non-decreasing in s. This means that for demand point (i,t) and a fixed warehouse order r, it is always more economical to hold the items in the retailer inventory, rather than in the warehouse inventory. In this case, we say that (i,t) is a J-demand.
   - For every r,s,s', such that $r \leq s < s' \leq t$, we have $h^{it}_{rs} \geq h^{it}_{rs'}$. That is, h^{it}_{rs} is monotonically non-increasing in s. This means that for demand point (i,t) and a fixed warehouse order r, it is always more economical to hold the items in the warehouse inventory, rather than in the retailer inventory. In this case, we say that (i,t) is a W-demand.
   - For every r_1,r_2,r_3, such that $r_1 \leq r_2 \leq r_3 \leq t$, we have $f^{it}_{r_1r_3} \leq f^{it}_{r_1r_2} + f^{it}_{r_2r_3}$. That is, f^{it}_{rs} is subadditive with respect to time intervals $[r_1,r_2)$ and $[r_2,r_3)$. This means that removing an item from the warehouse inventory, and putting it back immediately after cannot decrease the holding cost. In this case, we say that (i,t) is an S-demand.

We notice that the holding cost model studied by Levi *et al.* [91] is a particular case of the holding cost defined above. In their setting, they assume that each retailer contains either only W-demands, or only J-demands. Notice that, although property 2 is not explicitly assumed in [91], this property is implied by their monotonicity assumptions combined with the monge-property for W-demands. Differently from J or W-demands, being an S-demands is a property of function f^{it} only. This allows to model more general problems in which the warehouse and retailer holding cost functions are independent.

4.5.2 Primal-dual algorithm

We propose a primal-dual algorithm for the OWMR problem. The algorithm is inspired on the primal-dual framework of Levi *et al.* [90], based on a "wave" dual ascent algorithm. Rather than increasing dual variables uniformly, they increase the dual variables using the growing rate of the retailer holding cost of each demand, starting with the latest demands, and moving back to the first. However, for the case of OWMR, items can be held in the warehouse inventory, so the wave does not preserve some essential properties in the analysis in [90]. For OWMR, we use a new dual ascent algorithm, and a more complex pruning phase. Namely, we use the following two-phase algorithm: in the first phase, we generalize the wave algorithm to deal with more complex holding cost models, and, in the second phase, we prune warehouse and retailer orders separately.

We summarize the algorithm below, and describe each step in the following subsections.

1. Run the dual ascent algorithm, and temporarily open warehouse and retailer orders;
2. Prune orders and open permanent orders:
   (a) Permanently open warehouse orders;
   (b) Permanently open retailer orders:
      i. Place the first round of retailer orders;
      ii. Fix unsatisfied demands with additional retailer orders;
3. Connect demands.

Dual ascent algorithm

We use the following dual ascent mechanism. We consider a wave starts at time T and, as time passes, moves backward in time continuously, until it reaches time 0. Initially, all dual variables start with value zero, and all demand points are unconnected (unsatisfied). We say that a retailer order (i, s) is paid if $\sum_{t=s}^{T} l_s^{it} = K_i$. Denote the current position of

the wave by τ. For each moment τ and item i, let $P_i(\tau)$ be the set of time steps s such that the retailer order (i, s) is paid at time τ. At each moment τ, we set the budget b^{it} of each unconnected demand point (i, t) to be

$$b^{it} \leftarrow \min \ \{g^{it}_\tau\} \cup \{f^{it}_{s\tau} + g^{it}_s + l^{it}_s | s \in P_i(\tau)\}. \tag{4.11}$$

Notice that g^{it}_τ is only defined for integer values of τ. If τ is not integer, we define g^{it}_τ by linearly interpolating $g^{it}_{\lfloor\tau\rfloor}$ and $g^{it}_{\lceil\tau\rceil}$. Similarly, for any integer s, such that $\tau \le s \le t$, we define $f^{it}_{\tau s}$ by linearly interpolating $f^{it}_{\lfloor\tau\rfloor s}$ and $f^{it}_{\lceil\tau\rceil s}$. Also, we define $g^{it}_0 = \infty$, and $f^{it}_{0s} = \infty$, for every demand point (i, t) and $s \le t$.

As the wave moves, we place temporary retailer and warehouse orders. The sets of temporarily open warehouse orders is denoted by R, and temporarily open retailer orders is denoted by S_i for each i. Initially, sets R and S_i contain no orders. Whenever a demand point is satisfied by a temporarily open pair of retailer and warehouse orders, we freeze its budget and connect such demand to this pair.

The algorithm starts at time $\tau = T$ that is decreased continuously until $\tau = 0$. While the time passes, some of the following conditions may be satisfied, and the corresponding events are triggered. The events are processed in the order that they happen.

Event 1 For some demand (i, t), such that l^{it}_s has not started increasing, it holds $b^{it} = g^{it}_s$:

- start increasing l^{it}_s at the same rate of b^{it}.

Event 2 For some retailer order (i, s), that is not marked as paid, it holds $\sum_{t=s}^{T} l^{it}_s = K_i$:

- mark order (i, s) as paid,
- add s do S_i,
- set $open(i, s) = \tau$,
- for each $t \ge s$, stop increasing l^{it}_s.

Event 3 For some demand (i, t), and retailer order (i, s), such that (i, s) is paid and z^{it}_r has not started increasing, it holds $b^{it} = g^{it}_s + f^{it}_{rs} + l^{it}_s$ for some $r \le s$:

- if warehouse order at time r is temporarily open,
  - connect demand point (i, t) to the pair of orders $\lceil r, s\rfloor$,
  - set $freeze(i, t) = \tau$;
- else (if warehouse order r is not temporarily open),
  - start increasing z^{it}_r at the same rate of b^{it}.

Event 4 For some warehouse order $(0, r)$ that is not temporarily open, it holds that $\sum_{i=1}^{N} \sum_{t=r}^{T} z_r^{it} = K_0$:

- open warehouse order $(0, r)$,
- add r to R,
- set $open(0, r) = \tau$,
- for each demand point (i, t) such that for some s, variable z_{rs}^{it} have already started increasing:
  - connect demand point (i, t) to the pair of orders $\lceil r, s \rfloor$,
  - set $freeze(i, t) = \tau$.

Intuitively, we give the following interpretation to this dual ascent mechanism. Initially, all pairs of orders are offered for free to each demand point, so each demand can be served by its nearest order, and thus pays zero for the holding cost. As the wave moves, we remove gradually (and fractionally) the free pair $\lceil \tau, \tau \rfloor$ of orders at position τ, from the right to the left, until no order is offered at all. This means that, if a demand point (i, t) wants to be connected to a pair of free orders at time τ, then it must increase its budget to at least g_τ^{it}. However, it might happen that some retailer order $s \geq \tau$ is already paid by some group of demand points. In this case, a demand point can also be connected to the free warehouse order at τ through retailer order s. If the demand chooses to be satisfied by this path, it has to increase it budget to at least the holding cost of pair $\lceil \tau, s \rfloor$, that is $f_{\tau s}^{it} + g_s^{it}$, plus its contribution towards opening retailer order s, that is l_s^{it}.

Pruning phase

The shadow of a given order (i, s) is the interval $[open(i, s), s]$. For any two warehouse orders in R with intersecting shadows, there is some demand point that contributes to both orders. Therefore, we want to open a subset of permanent warehouse orders R', with no intersecting shadows. This is accomplished by a *pruning* procedure. However, two joint orders with intersecting shadows may include distinct retailer orders, and thus dropping one of such orders may leave demand points unsatisfied. For the JRP, the algorithm of Levi *et al.* [90] simply moved any retailer order (i, s) from the dropped joint order to the position of the earliest permanent warehouse order that was placed in the shadow of (i, s). This strategy for fixing unsatisfied demand points does not work for W-retailers, since, contrary to the case of the JRP, W-retailer orders and warehouse orders are not necessarily synchronized, and thus it may happen that no warehouse order is placed in the retailer shadow.

In our algorithm, we open permanent warehouse orders and retailer orders in two steps. First, we permanently open only warehouse orders, and ignore the retailers, using a simple greedy algorithm to open orders with non-intersecting shadows. This guarantees that each demand point contributes to at most one warehouse order. Then we open the orders of each retailer using a two-round procedure. In the first round, we permanently open retailer orders with non-intersecting shadows, so that a demand point does not use the same budget to contribute to several orders. When this first round finishes, each demand whose budget is larger than the holding cost of some permanently open pair of orders is considered satisfied. In the second round, we make sure that any remaining demand is satisfied. This is done by greedily placing a retailer order for some unsatisfied demand. By carefully selecting the next unsatisfied demand point, we guarantee that each demand pays for at most two retailer orders opened in the second round.

(a) Opening permanent warehouse orders:

Let $R = \{r_1, \ldots, r_k\}$, with $r_1 < r_2 < \cdots < r_k$, be the set of all time steps at which a warehouse order was temporarily open. We will return a set R' of permanent warehouse orders. At each moment we keep the last time r' at which we placed a permanent warehouse order. Then we iterate over every $r \in R$, in increasing order of time, and add r to R' if the shadow of r does not intersect the shadow of any order included previously. More precisely, we run the following steps:

1. Set $r' \leftarrow 0$, and $R' \leftarrow \emptyset$.
2. For each $r \leftarrow r_1, \ldots, r_k$, if $open(0, r) > r'$, then
   - set $r' \leftarrow r$, and
   - update $R' \leftarrow R' \cup \{r\}$.

(b) i. First round of retailer orders:

For each retailer i, we open a first round of retailer orders using the same algorithm used for opening warehouse orders. That is, let $S_i = \{s_1, \ldots, s_k\}$, with $s_1 < s_2 < \cdots < s_k$, be the set of all time steps at which retailer i has placed temporary orders. We obtain a set S'_i of permanent retailer orders, by greedily iterating over $s \in S_i$, in order of time, and adding non-intersecting retailer orders to S'_i.

Before going to the second round, we introduce some notation. For any demand (i, t), let $left(i, t)$ be the minimum time σ such that $b^{it} = g^{it}_\sigma$. Notice that σ may be fractional (and g^{it} used in the dual ascent algorithm). Intuitively, $left(i, t)$ is the leftmost retailer order by which demand (i, t) may be served paying at most its budget. Also, a demand (i, t) is said to be currently satisfied if there is a permanently open pair of

order $\lceil r, r \rfloor$ such that $h^{it}_{rr} \leq b^{it}$, or if there is no open warehouse order $r \in R'$ such that $r \in [left(i,t), t]$.

(b) ii. Fixing unsatisfied demands:

For each retailer i, we open a second round of retailer orders S''_i to fix unsatisfied demands. We fix unsatisfied demands greedily. Let (i,t) be the demand point with largest $left(i,t)$. Since (i,t) is unsatisfied, we know that there is at least one permanently open warehouse order at $r \in R'$ such that $r \in [left(i,t), t]$. Let r' be the least such time. We include a retailer order r' in S''_i. In this case, we say that demand (i,t) is the *initiator* of order (i,r'). Notice that after permanently opening the retailer order (i,r'), the demand at time t, as well as any unsatisfied demand at time t' with $t' \geq r'$, will become satisfied. We repeat this process until all demand points become satisfied. More precisely, for each retailer i we execute the following steps:

1. Set $S''_i \leftarrow \emptyset$.
2. While there are unsatisfied demand points in retailer i:
   (a) let (i,t) be an unsatisfied demand point with largest $left(i,t)$,
   (b) let $r' \leftarrow \min [left(i,t), t] \cap R'$,
   (c) update $S''_i \leftarrow S''_i \cup \{r'\}$.

Connecting demand points

At the last phase of the algorithm, each demand point (i,t) is simply connected to one pair of orders $\lceil r, s \rfloor$ such that $r \in R'$, $s \in S' \cup S''$, and $r \leq s \leq t$. If there are many pairs that satisfy these properties, we choose the ordering pair with minimum h^{it}_{rs}.

4.5.3 Analysis

By the construction of the dual ascent algorithm, the set of variables (b, l, z) is a feasible solution to the dual problem. The incurred costs of the generated solution correspond to the cost of holding items for each demand and the cost of placing permanently open orders: the warehouse orders in R', the first round of retailer orders in S'_i for each i, and the second round of retailer orders in S''_i for each i. We use the following charging scheme: for each demand point (i,t) and $r' \in R'$, $z^{it}_{r'}$ is the contribution of (i,t) towards opening warehouse order r'. Similarly, for each $s' \in S'_i$, $l^{it}_{s'}$ is the contributions for retailer order s'. For retailer orders in S''_i, we use a more complex charging scheme. For a given retailer order (i,s''), with $s'' \in S''_i$, let $(i,\hat{t})$ be the corresponding initiator. Since demand $(i,\hat{t})$ was connected by the dual ascent algorithm, there must exist at least one order in S_i to

which $(i, \hat{t})$ can connect. Let $\hat{s}$ be one such order with the largest opening time, that is, an order that was first paid by the dual ascent algorithm. To pay for the order s'', we will use the same budget used to pay for retailer order $\hat{s}$. Notice that here we must use that fact that all retailer orders of item i have the same cost. To denote the order $\hat{s}$ of which the contribution will be used to pay s'', we define $N_i(s'') = \hat{s}$. The contribution of a given demand (i, t) towards s'' is thus $l^{it}_{\hat{s}}$. Finally, each demand point pays for the holding cost of its own items.

First, we give an auxiliary lemma.

Lemma 4.10. *For any demand point (i, t), there exists a paid retailer order (i, s), $s \in S_i$, such that $left(i, t) \leq open(i, s)$, and $s \leq t$.*

Proof. Let $\tau = freeze(i, t)$, and $\sigma = left(i, t)$. Since demand (i, t) has been connected, we know that there exists a retailer order $s \in S_i$ such that $\sigma \leq s$ and $\tau \leq open(i, s)$. Let $\bar{s}$ be the retailer order $s \in S_i$ such that $\sigma \leq s$ with maximum $open(i, \bar{s})$. For the sake of contradiction, suppose that $open(i, \bar{s}) < \sigma$.

Consider the state of the dual ascent algorithm at the time $\tau' = open(i, \bar{s})$, just before any event is processed, and let b be the budget of (i, t) at such moment. At this moment, we know that $\bar{s} \notin S_i$. Also, the demand (i, t) is connected to the free order τ' through some order s'. Since $\tau' < \sigma$, we obtain $g^{it}_{\tau'} > g^{it}_{\sigma} = b^{it} \geq b$. It follows that s' must be already paid, that is, $s' \in S_i$, and so $open(i, s') > \tau'$. But since $b^{it} \geq b \geq g^{it}_{s'}$, we have $\sigma \leq s'$. This is a contraction to the maximality of $\bar{s}$, and the lemma follows. □

In the following lemmas we bound the contribution of a given demand to each incurred cost of the solution. Each contribution of a demand point (i, t) is bounded by a factor of its budget b^{it}, so the total contribution is a factor of the dual solution value, and the approximation follows by the weak duality theorem.

Lemma 4.11. *The contribution of demand (i, t) towards opening warehouse orders in R' is at most b^{it}.*

Proof. Let $r \in R$ be a warehouse order to which (i, t) gives a positive contribution, that is, $z^{it}_r > 0$. Since $z^{it}_r > 0$, there is a paid retailer order s at time $\tau = freeze(i, t)$ such that $b^{it} > g^{it}_s + f^{it}_{rs} + l^{it}_s$ by Event 3 of the dual ascent algorithm. Since s is already paid at time τ, we have $s \geq open(i, s) \geq \tau$. Thus we get $s \in P_i(\tau)$, and therefore $b^{it} \leq g^{it}_s + f^{it}_{\tau s} + l^{it}_s$, so $f^{it}_{\tau s} > f^{it}_{rs}$, and, by the monotonicity of f^{it}, this implies that $r \geq \tau$.

Let $r' \in R'$ be the largest warehouse order time such that $z^{it}_{r'} > 0$. We claim that $open(i, r') \leq freeze(i, t)$. Indeed, if at time $open(i, r')$, demand (i, t) is not connected yet, then it would be connected at this point. It follows from the algorithm to open permanent warehouse orders that $R' \cap [freeze(i, t), r') \subseteq R' \cap [open(i, r'), r') = \emptyset$. Since

demand (i,t) does not contribute to warehouse orders $r'' \in R'$ with $r'' < freeze(i,t)$, it follows that (i,t) can only contribute to one warehouse order, namely r'. □

Lemma 4.12. *The contribution of demand (i,t) towards opening retailer orders in S'_i is at most b^{it}.*

Proof. Let $s_1, \ldots, s_k \in S'_i$, with $s_1 < \cdots < s_k$, be the retailer order times to which demand (i,t) gives a positive contribution, that is, for which $l^{it}_{s_j} > 0$ for each $j = 1, \ldots, k$. It follows from the algorithm to open the first round of retailer orders that the shadows of the retailer orders do not intersect, that is, for each $j = 2, \ldots, k$, we have $s_{j-1} < open(i, s_j)$.

Let τ_j be the time at which order (i, s_j) was paid, that is, $\tau_j = open(i, s_j)$, and let $b^{it}_{\tau_j}$ be the budget of demand point (i,t) at this moment. By equation (4.11), $b^{it}_{\tau_j} \leq g^{it}_{\tau_j}$. Therefore, the contribution of (i,t) to some retailer order (i, s_j) is $l^{it}_{s_j} \leq b^{it}_{\tau_j} - g^{it}_{s_j} \leq g^{it}_{\tau_j} - g^{it}_{s_j}$. For any $j \geq 2$, we have $g^{it}_{\tau_j} \leq g^{it}_{s_{j-1}}$, since $s_{j-1} < open(i, s_j) = \tau_j$, and thus $l^{it}_{s_j} \leq g^{it}_{s_{j-1}} - g^{it}_{s_j}$. For $j = 1$, trivially $l^{it}_{s_1} \leq b^{it} - g^{it}_{s_1}$. Adding up all contributions, we obtain

$$\sum_{j=1}^{k} l^{it}_{s_1} \leq b^{it} - g^{it}_{s_1} + \sum_{j=2}^{k} (g^{it}_{s_{j-1}} - g^{it}_{s_j}) \leq b^{it}. \qquad \square$$

In the next lemma, and afterwards, we use the following definition.

Definition 4.2. *We say that a demand point (i,t) is served by a twin ordering pair if there is an ordering pair $\lceil r, r \rfloor$ that is permanently open such that $left(i,t) \leq r \leq t$.*

Lemma 4.13. *The contribution of demand (i,t) towards opening retailer orders in S''_i is:*

- *at most $2b^{it}$, if (i,t) is served by a twin ordering pair, or*
- *at most b^{it}, if (i,t) is not served by a twin ordering pair.*

Proof. Let $s_1, \ldots, s_k \in S''_i$, with $s_1 < \cdots < s_k$, be the retailer order times to which demand (i,t) gives a positive contribution, that is, for which $l^{it}_{\hat{s}_j} > 0$, where $\hat{s}_j = N_i(s_j)$, for each $j = 1, \ldots, k$. Also, for each $j = 1, \ldots, k$, let (i, t_j) be the initiator of retailer order (i, s_j), and let $\tau_j = left(i, t_j)$.

Recall that, for every retailer order $s \in S''_i$, there is a warehouse order $s \in R'$. It follows from the algorithm to fix unsatisfied demands that there is no permanently open warehouse order in the interval $[\tau_j, s_j)$, for any j, that is, $[\tau_j, s_j) \cap R' = \emptyset$.

We claim that $t_{j-1} < s_j$ for any $j = 2, \ldots, k$. Let $j > 1$. We have $s_{j-1} \in R'$, but $[\tau_j, s_j) \cap R' = \emptyset$, thus $s_{j-1} < \tau_j$, since otherwise we would get $s_{j-1} \geq s_j$. It follows that $\tau_{j-1} < \tau_j$. This implies that demand point (i, t_j) was picked before demand point (i, t_{j-1}) by the algorithm. Consider the execution state of the algorithm just after (i, t_{j-1}) was picked (at step 2(a)). At this moment, we must have $s_j \in S''_i$, since (i, t_j) was picked at a

previous iteration. Therefore, at this moment the ordering pair $\lceil s_j, s_j \rfloor$ was permanently open, and thus any demand point (i, t') with $t' \geq s_j$ should be satisfied. Since (i, t_{j-1}) was picked, it was unsatisfied at this moment, and thus $t_{j-1} < s_j$.

By definition, $\hat{s}_j$ is the retailer order $s \in S_i$ such that $\tau_j \leq s \leq t_j$ with the largest $open(i, s)$. By Lemma 4.10, we know that there is at least one such s such that $\tau_j \leq open(i, s)$, and thus $\tau_j \leq open(i, \hat{s}_j)$. Let $j > 2$. Recall that $t_{j-2} < s_{j-1}$, and $s_{j-1} < \tau_j$. Hence $open(i, \hat{s}_{j-2}) \leq \hat{s}_{j-2} \leq t_{j-2} < s_{j-1} < \tau_j \leq open(i, \hat{s}_j) \leq \hat{s}_j$. We conclude that the shadows of any two retailer orders $(i, \hat{s}_{j_1})$ and $(i, \hat{s}_{j_2})$ with j_1 and j_2 odd do not intersect, that is, $[open(i, \hat{s}_{j_1}), \hat{s}_{j_1}] \cap [open(i, \hat{s}_{j_2}), \hat{s}_{j_2}] = \emptyset$. Using the same arguments of Lemma 4.12, we conclude that the total contribution of demand point (i, t) towards opening retailer orders with odd indices is at most b^{it}. Analogously, the total contribution of demand point (i, t) towards opening retailer orders with even indices is at most b^{it}.

If (i, t) is served by a twin ordering pair, then we are done. So, from now on, assume that this is not the case. Therefore, there is no open warehouse order $r \in R'$ such that $r \in [left(i, t), t]$. We claim that $k \leq 1$, and thus the lemma will follow. For the sake of contradiction, suppose that $k > 1$. Since $s_k \in R'$, and $s_k \leq t$, we obtain $s_k < left(i, t)$, otherwise we would get $[left(i, t), t] \cap R' \neq \emptyset$. But then $\hat{s}_{k-1} \leq t_{k-1} < s_k < left(i, t)$. However, since (i, t) contributes to $\hat{s}_{k-1}$, we know that $b^{it} > g^{it}_{\hat{s}_{k-1}}$, and thus $left(i, t) < \hat{s}_{k-1}$. This is a contradiction, and we are done also in this case. □

Lemma 4.14. *Suppose that demand (i, t) is not served by a twin ordering pair. Then the holding cost of demand (i, t) is at most $2b^{it}$.*

Proof. Let $\tau = freeze(i, t)$ and $\sigma = left(i, t)$.

We claim that there is an open warehouse order $r' \in R'$, such that $\tau \leq r' \leq t$. Let $r \in R$ be the warehouse order to which (i, t) was connected, such that $r \leq t$. Clearly, $open(0, r) \geq \tau$, since the warehouse order was already open when (i, t) was connected. If $r \in R'$, then the claim holds. Otherwise, r was not permanently open by the algorithm, and thus there must exist some $r' \in R' \cap [open(0, r), r]$. Then $\tau \leq open(0, r) \leq r' \leq r \leq t$, and the claim holds also in this case.

Since (i, t) is not served by a twin ordering pair, then there is no permanently open warehouse order in the interval $[\sigma, t]$, that is, $[\sigma, t] \cap R' = \emptyset$ (recall Definition 4.2). It follows that $g^{it}_{r'} > g^{it}_{\sigma} = b^{it}$, and thus $r' < \sigma$. At the freezing time τ, we know that demand point (i, t) is connected to the "free" warehouse order τ through some order s. That is, there exists $s \in S_i$, such that $s \leq t$, and $b^{it} = f^{it}_{\tau s} + g^{it}_{s} + l^{it}_{s}$. Since $b^{it} \geq g^{it}_{s}$, we know that $\sigma \leq s$,

We claim that (i, t) is not a J-demand. By the monotonicity of g^{it}, we have $h^{it}_{\tau\tau} = g^{it}_{\tau} \geq g^{it}_{r'} > g^{it}_{\sigma} = b^{it} \geq h^{it}_{\tau s}$. Since $\tau \leq r' < \sigma \leq s$, we conclude that (i, t) cannot be a J-demand. Therefore, (i, t) is either a W-demand, or an S-demand. We analyze each case separately.

First, assume that (i,t) a W-demand. By Lemma 4.10, we know that there is a paid retailer order $\bar{s} \in S_i$ such that $\sigma \leq open(i,\bar{s})$ and $\bar{s} \leq t$. Therefore, there must exist a permanently open order in the first round of retailer, $s' \in S'_i$, such that $s' \in [open(i,\bar{s}), \bar{s}]$, and thus, $\sigma \leq s' \leq t$. We will bound the cost of connecting demand point (i,t) to the ordering pair $\lceil r', s' \rfloor$, as the actual holding cost paid by (i,t) cannot be greater than this cost. We consider two subcases: $s \leq s'$, or $s > s'$. If $s \leq s'$, then $h^{it}_{r's'} \leq h^{it}_{r's} = f^{it}_{r's} + g^{it}_{s} \leq f^{it}_{\tau s} + g^{it}_{s} \leq b^{it}$, where the first inequality is true because (i,t) is W-demand. If $s > s'$, then $h^{it}_{r's'} = f^{it}_{r's'} + g^{it}_{s'} \leq f^{it}_{r's} + g^{it}_{\sigma} \leq 2b^{it}$. Therefore, we obtain $h^{it}_{r's'} \leq 2b^{it}$, and the lemma follows in this case.

From now on, we assume that (i,t) is an S-demand. Again, we know that there is an open order in the first round of retailer in $[open(i,s), s]$. Let $s' \in S'_i$ be the largest such order. If $\sigma \leq s'$, then $h^{it}_{r's'} = f^{it}_{r's'} + g^{it}_{s'} \leq f^{it}_{r's} + g^{it}_{\sigma} \leq 2b^{it}$, and we are done. So, we assume that $\sigma > s'$.

Consider the state of the dual ascent algorithm at time s', just before any event is processed, and let b be the budget of demand (i,t) at this moment. Clearly, we have $b \leq b^{it}$, because $s' \leq open(i,s) \leq \tau$. Since $s' < \sigma$, we get $b \leq b^{it} = g^{it}_{\sigma} < g^{it}_{s'}$. We conclude that at the time $\tau' = s'$, the demand point (i,t) was connected to the "free" warehouse order τ' through some paid retailer order. That is, there exists $\bar{s} \in S_i$ such that $\bar{s} \leq t$ and $b = f^{it}_{s'\bar{s}} + g^{it}_{\bar{s}} + l^{it}_{\bar{s}}$. Since $\bar{s}$ was already paid at time s', we get $open(i,\bar{s}) > s'$. Once again, there must exist an open retailer order $\bar{s}' \in S'_i \cap [open(i,\bar{s}), \bar{s}]$. Since $(s', s] \cap S'_i = \emptyset$, we conclude that $s < \bar{s}' \leq \bar{s}$.

Now we bound the holding cost of serving demand (i,t) by the ordering pair $\lceil r', \bar{s}' \rfloor$. We get $h^{it}_{r'\bar{s}'} = f^{it}_{r'\bar{s}'} + g^{it}_{\bar{s}'} \leq f^{it}_{\tau\bar{s}} + g^{it}_{s} \leq f^{it}_{s'\bar{s}} + f^{it}_{\tau s'} + g^{it}_{s} \leq f^{it}_{s'\bar{s}} + f^{it}_{\tau s} + g^{it}_{s} \leq b + b^{it} \leq 2b^{it}$, where the second inequality follows from the fact that (i,t) is an S-demand. □

Lemmas 4.13 and 4.14 imply the following corollary.

Corollary 4.2. *The contribution of demand (i,t) towards opening retailer orders in S''_i plus the holding cost of demand (i,t) is at most $3b^{it}$.*

Proof. If (i,t) is not served by a twin ordering pair, the lemma follows directly from Lemmas 4.13 and 4.14. If (i,t) is served by a twin ordering pair $\lceil r, r \rfloor$, then we know that $h^{it}_{rr} = g^{it}_{r} \leq b^{it}$, since otherwise demand (i,t) would be unsatisfied, and so the corollary follows from Lemma 4.13. □

The following theorem is now immediate.

Theorem 4.4. *The primal-dual algorithm in Subsection 4.5.2 is a 5-approximation for the OWMR with independent retailer-warehouse holding costs*

4.6 The Multilevel Joint Replenishment Problem

This section notices that the Multilevel JRP, when the supply chain is comprised of a rooted tree, can be reduced to the Multistage Assembly Problem, and thus admits a 2-approximation by Levi *et al.* [90].

Problem's definition. In the *Multilevel JRP*, one is given a tree, rooted at a special vertex p, called the *producer*. A leaf i in this tree is called a *retailer*, and an intermediate vertex is called a *warehouse*. We consider a discretized time interval $[1, T]$ and, in each time step $t \in [1, T]$, a retailer i may demand an amount d_{it} of items in stock. Each such pair (i, t) is a demand point, and can be satisfied only if the retailer i has placed an order to the parent node at some time s, with $s \leq t$. We assume that only retailer nodes can keep items in stock, so, every time a non-retailer node receives an order for a certain amount of items, it must place an order of the same amount to its parent node if it is a warehouse, or it must produce this amount of items if it is the producer. In other words, a retailer i can only place an order at time s if all vertices in the tree path $(p, \ldots, i)$ have also placed orders at time s. The placement of an order by vertex v incurs a setup cost K_v. Also, if a demand point (i, t) is satisfied by a retailer order placed at time s, then the incurred cost of keeping such d_{it} items in stock is H_{ist}. The objective is to minimize the overall ordering and holding costs.

The Multilevel JRP can be stated as an integer linear programming formulation, whose variable y_{vs} indicates whether node v places an order at time s, and variable x_{ist} indicates whether demand point (i, t) is served at time s. Next, the parent node of a non-root vertex v is denoted by $\pi(v)$, and the unique path from p to some retailer i is $P(i) = (p, \ldots, i)$. Also the set of all positive demand points is D, and the set of all nodes is V.

$$\begin{aligned} \text{minimize} \quad & \sum_{v \in V} \sum_{s \in [T]} y_{vs} K_v + \sum_{(i,t) \in D} \sum_{s \in [t]} x_{ist} H_{ist} \\ \text{subject to} \quad & \textstyle\sum_{s \in [t]} x_{ist} = 1 && (i,t) \in D \\ & x_{ist} \leq y_{vs} && (i,t) \in D, s \in [t], v \in P(i) \\ & x_{ist} \in \{0,1\} && (i,t) \in D, s \in [t] \\ & y_{vs} \in \{0,1\} && v \in V, s \in [T] \end{aligned} \tag{4.12}$$

In the *Multistage Assembly Problem with Echelon Holding Costs*, one is given a rooted tree, whose nodes correspond to items, and the root is a special item denoted by p. It is considered a discretized time interval $[1, T]$ and, at each time step $t \in [1, T]$, item p faces an external demand of d_t units. Each unit of item i is assembled using one unit of each of its predecessors (each predecessor of an item corresponds to one child node in the tree). Therefore, an order for n units of item i incurs a demand of n units of each predecessor

j, and any demand must be satisfied by the inventory currently held for j. Assembling any amount of items of any item i incurs a setup cost K_i. Any external demand of d_t units of item p is also a demand of d_t units of item i, for each i. Satisfying demand d_t using d_t units of items i assembled at time s incurs an "echelon" holding cost of H_{ist}. The objective is to minimize the overall ordering and holding costs.

The *echelon inventory level* of an item i is the total number of units of i in the system, including units that are assembled into other items and the inventory of that item. The *echelon holding cost* h_{ist} is the cost of storing one unit of item i, either in the inventory of item i, or in the inventory of other items that were assembled using this unit, from the time s that this unit was ordered to the time t it was delivered. Therefore, $H_{ist} = d_t \cdot h_{ist}$. Using echelon holding cost allows one to decompose the holding cost on a per-item basis, as used in the integer linear programming formulation due to Levi *et al.* [90].

The following formulation assumes *nested* and *zero inventory ordering* (ZIO) policies. In a nested policy, item i only places an order at a time s if all successors of i also place orders at time s. In a ZIO policy, an order is placed for an item only when its inventory is empty. These assumptions can be made without loss of generality, since it is known that there are optimal policies that satisfy both properties [42].

$$\begin{aligned}
\text{minimize} \quad & \sum_{i\in V}\sum_{s\in[T]} y_{is}K_i + \sum_{i\in V}\sum_{t\in[T]}\sum_{s\in[t]} x_{ist}H_{ist} \\
\text{subject to} \quad & \textstyle\sum_{s\in[t]} x_{ist} = 1 && i \in V, t \in [T] \\
& x_{ist} \le y_{vs} && i \in V, t \in [T], s \in [t], v \in P(i) \\
& x_{ist} \in \{0,1\} && i \in V, t \in [T], s \in [t] \\
& y_{is} \in \{0,1\} && i \in V, s \in [t]
\end{aligned} \tag{4.13}$$

Observation 4.1. *The Multilevel JRP can be reduced to the assembly problem with echelon holding costs. An α-approximation for the assembly problem implies an $(\alpha+\varepsilon)$-approximation for the Multilevel JRP, for every $\varepsilon > 0$.*

Proof. Let $l = \lceil \frac{\alpha-1}{\varepsilon} \rceil$. Consider an instance $\mathcal{I}_1$ of the Multilevel JRP. We will transform this instance into another instance of the Multilevel JRP $\mathcal{I}_2$, such that for every item i and every time step t, we have $d_{it} = 1$. Let D_1 be the set of demand points of $\mathcal{I}_1$. For each pair formed by item $i \in V$ and time $t \in [T]$ such that $(i,t) \notin D_1$, we create a (dummy) demand point (i,t), by defining $d_{it} = 1$ and $h_{ist} = 0$, for every $s = 1, \ldots, t$. Then, for every demand point $(i,t) \in D_1$, we scale the unit holding cost by d_{it}, and redefine the amount of demand to $d_{it} = 1$. Notice that these operations do not change the value of $H_{ixt} = d_{it}h_{ist}$. Now we add $l-1$ copies of each demand point (i,t), and redefine the total number of time steps to $T' = lT$: for each $i \in V$, $t \in [T]$, and $j = 1, \ldots, l-1$, we add a new demand point for item i at time $t' = jT + t$, by making $d_{it'} = 1$, with holding cost

defined by:

$$h_{ist'} = \begin{cases} 0 & (i,t) \notin D_1, \\ \infty & (i,t) \in D_1 \text{ and } s \leq jT, \\ h_{i,s-jT,t} & (i,t) \in D_1 \text{ and } s > jT. \end{cases}$$

Finally, we add an extra time step $t = 0$, and corresponding demand points $(i, 0)$ with $d_{i0} = 1$ for every i. For each demand (i, t), the cost of being served by an order at time $s = 0$ is $h_{i0t} = \infty$ if (i, t) is in D_1, or if (i, t) is a copy of a demand point in D_1, and $h_{i0t} = 0$ otherwise. This finishes the construction of $\mathcal{I}_2$. Observe that an optimal solution O_1 of $\mathcal{I}_1$ induces a solution for $\mathcal{I}_2$ with cost $l \cdot cost(O_1) + \sum_{v \in V} K_v$, and an optimal solution O_2 of $\mathcal{I}_2$ induces a solution for $\mathcal{I}_1$ with cost $(cost(O_2) - \sum_{v \in V} K_v)/l$, so we have $cost(O_2) = l \cdot cost(O_1) + \sum_{v \in V} K_v$.

Now, we use the reduction from the JRP by Levi *et al.* [90], but for the multilevel case. We create an instance $\mathcal{I}_3$ of the assembly problem from $\mathcal{I}_2$, such that every node $v \in V$ is an item, with ordering cost K_v, and node p is facing external demand $d_t = 1$ at each time step. What is missing to complete instance $\mathcal{I}_3$ is defining the echelon holding costs. For each leaf node i, we use the holding cost function of $\mathcal{I}_2$, and for other (dummy) items v, define the holding cost to be zero. Since we can restrict solutions of $\mathcal{I}_3$ to nested policies, then any solution of $\mathcal{I}_3$ induces a solution of $\mathcal{I}_2$ with the same cost, and vice-versa. The former claim is formalized in the following, the later is analogous.

We construct a solution S_2 for $\mathcal{I}_2$ from a solution S_3 for $\mathcal{I}_3$, with the same set of orders and using ZIO policies for satisfying demands. Without loss of generality, we assume that S_3 is nested and uses ZIO policies, since otherwise we could modify S_3 without increasing the cost [42]. Therefore, for every order placed at a given node v at time s, there is one order at its parent node $\pi(v)$ at the same time s, that is, if order $(v, s) \in S_3$, then order $(\pi(v), s) \in S_3$. Also, since $d_0 = 1$, for every $v \in V$, $(v, 0) \in S_3$, and so $(v, 0) \in S_2$. This means that there is a joint order in S_2 including all items at time $t = 0$, and so every demand can be satisfied. We conclude that S_2 is feasible. Now we calculate the cost of S_2. Clearly, the ordering cost of S_2 is the same as that of S_3. Also, the holding cost of S_2 is equal to the echelon holding cost of S_3 corresponding to leaf items. To see this, recall that each demand for item i at time t in the Multilevel JRP corresponds to one unit demand for item i at time t of the assembly problem, and both demands are satisfied by the same orders, namely, the latest order (i, s) with $s \leq t$. As the echelon holding cost of dummy items is zero, there are no other incurred costs in S_3. Therefore, $cost(S_3) = cost(S_2)$.

To complete the proof, it is enough to observe that given an α-approximated solution S_2 for $\mathcal{I}_2$, we obtain a solution S_1 for $\mathcal{I}_1$ such that $cost(S_1) = (cost(S_2) - \sum_{v \in V} K_v)/l \leq (\alpha \cdot cost(O_2) - \sum_{v \in V} K_v)/l = \alpha \cdot cost(O_1) + (\alpha - 1)/l \sum_{v \in V} K_v \leq (\alpha + \varepsilon) \cdot cost(O_1)$. □

Chapter remarks

Despite considering integrated decisions on distribution and inventory control may lead to significant economy, few works have considered such problems under the approximation algorithms perspective. This chapter introduced the Production and Distribution Problem (PDP), that aims at optimizing ordering, distribution and inventory holding costs in an integrated manner. A variant of the PDP has already been considered by Pochet and Wosley [116], that studied valid inequalities to strengthen integer linear programming formulations of the problem. Here, we consider a different formulation, that resembles those of the FLP and of the JRP, so that we are able to obtain an approximation algorithm, using some ideas already applied in algorithms for these problems. This leads to a 2.77-approximation for the version without retailer ordering costs, and a 5-approximation for the version with retailer ordering costs. The PDP was studied under natural assumptions that combine monotonically increasing holding costs, and metric distance functions.

Other considered supply chain problem is the OWMR with a more natural holding cost structure, for which no constant approximation was known previously. We gave a 5-approximation based on a based on a novel primal-dual technique. This contribution is two-folded since it provides a primal-dual algorithm for the OWMR, that was an open question [91], and it improves the wave mechanism introduced by Levi *et al.* [90], giving an example of a primal-dual technique that increases the dual variable using a more sophisticated processes.

Moreover, we noted that an important subclass of the Submodular JRP can be reduced to the Multistage Assembly Problem, and thus admits a constant approximation.

Chapter 5

Circle Packing Problems

This chapter presents approximation algorithms for circle packing problems. In the Circle Bin Packing Problem, the objective is to pack a set of circles into the minimum number of unit squares, and in the Circle Strip Packing Problem, the objective is to pack a set of circles into a strip of unit width and minimum height.

Problem's definition. In the *Circle Bin Packing Problem*, one is given a set of circles $\mathcal{C} = \{C_1, \ldots, C_n\}$, such that each circle $C_i \in \mathcal{C}$ has radius r_i, with $0 \leq r_i \leq 1/2$. A packing of a subset of circles $S \subseteq \mathcal{C}$ into a bin is an arrangement of S, such that the center of each circle $C_i \in S$ is associated to coordinates x_i, y_i, with $r_i \leq x_i, y_i \leq 1 - r_i$, and no two circles intersect, that is, it holds $(x_i - x_j)^2 + (y_i - y_j)^2 \geq (r_i + r_j)^2$ for every pair $C_i, C_j \in S$. The objective is to find a packing of all circles using the minimum number of bins.

In the *Circle Strip Packing Problem*, given a set of circles $\mathcal{C}$, a packing of $\mathcal{C}$ into a strip of height H is an arrangement of circles such that the center of each circle $C_i \in \mathcal{C}$ is associated to coordinates x_i, y_i, with $r_i \leq x_i \leq 1 - r_i$, $r_i \leq y_i \leq H - r_i$, and no two circles intersect. The objective is to find a packing of the circles into a strip of minimum height.

Summary of results. We give APTAS's for both the Circle Bin Packing, and the Circle Strip Packing. The bin packing problem is considered with resource augmentation in one dimension, that is, we use bins of unit width and height $1 + \gamma$, for some arbitrarily small $\gamma > 0$. These are the first approximations for these problems. As is common in the literature of packing problems, we distinguish between "large" and "small" items. However, this distinction is dynamic, so that one item may be considered small in one iteration, but large in another. Two main novel ideas differ from the approaches for rectangle packing: first, instead of packing large items using combinatorial brute force algorithms, the packing of large circles is reduced to the problem of solving a semi-algebraic system, what is done with the aid of standard quantifier elimination algorithms

from algebra; second, to pack small items, the free space of previous packings is cut in smaller bins, and the algorithm for large items is used recursively.

In Section 5.1, we review works on packing of circles and rectangles. In Section 5.2, we discuss how to decide whether a set of n circles can all be packed in a rectangular bin using algebraic quantifier elimination. In Section 5.3, an approximation algorithm for the packing of "large" circles is given. In Section 5.4, we present APTAS's for the Circle Bin Packing Problem, and for the Circle Strip Packing Problem.

5.1 Literature review

Packing problems have a large number of applications, such as packaging of boxes in containers, or cutting of material. For the special case that the items are equal circles, applications include, for example, obtaining a maximal coverage of radio towers in a geographical region [128]. In the literature of approximation algorithms, the majority of the works considers the packing of simple d-dimensional items, such as squares and cubes, into larger recipients, such as rectangular bins and strips. Most of these works are interested in the asymptotic approximation ratio. The packing problems involving circles are mostly considered through heuristics, and numerical methods.

Demaine *et al.* [47] proved that it is NP-hard to decide whether a set of circles can be packed into a unit square, or into an equilateral triangle. Therefore, the Circle Bin Packing Problem and the Circle Strip Packing Problem are also NP-hard. The problem of finding the densest packing of equal circles into a square has been largely investigated using many different optimization methods, such as continuous and nonlinear systems, and discrete methods [23]. For an extensive book on this and related problems, and corresponding used methods, see [128]. The case of non-equal circles is considered in [57], that uses heuristics, such as genetic algorithm, to pack circles in a rectangular container. The Circle Strip Packing has been considered using many approaches, such as branch-and-bound procedures, metaheuristics, etc. For a broad list of algorithms for the Circle Strip Packing, and related circle packing problems, see [69] and references therein.

For the problem of packing rectangles into rectangular bins, there is a sequence of algorithms [13, 31, 39] that leads to a $(1.525 + \varepsilon)$-approximation by Bansal *et al.* [13]. Recently, Bansal and Khan gave a 1.405-approximation [16]. For the bin packing of d-dimensional cubes, Kohayakawa *et al.* [82] gave an asymptotic approximation ratio of $2-(2/3)^d$, later improved to an APTAS by Bansal *et al.* [14]. For a survey on bin packing, see [40]. The best bound for rectangle strip packing problem is an APTAS by Kenyon and Rémila [80]. For the 3-dimensional case, the first specialized algorithm for cubes has asymptotic bound of 2.361 [110], and the best result is an APTAS due to Bansal *et al.* [15].

5.2 Packing of circles through algebraic quantifier elimination

In this section, we consider the following *circle packing decision problem*. We are given numbers $h, w \in \mathbb{Q}_+$, and a set of n circles $\mathcal{C} = \{C_1, \ldots, C_n\}$, where circle C_i has radius r_i, with $2r_i \leq \min\{w, h\}$, $1 \leq i \leq n$. The objective is to decide whether the circles can be packed in a bin of size $w \times h$ (of width w and height h). In the case of a positive answer, a realization of the packing should also be returned. More precisely, for each circle C_i, $1 \leq i \leq n$, we want to find a point $(x_i, y_i) \in \mathbb{R}^2_+$ that represents the center of C_i in a rectangle whose bottom-left and top-right corners correspond to points $(0, 0)$ and (w, h), respectively.

The circle packing decision problem can be equivalently formulated as deciding if there are real numbers $x_i, y_i \in \mathbb{R}_+$, $1 \leq i \leq n$, that satisfy the constraints

$$(x_i - x_j)^2 + (y_i - y_j)^2 \geq (r_i + r_j)^2 \quad \text{for } 1 \leq i < j \leq n, \tag{5.1}$$

$$r_i \leq x_i \leq w - r_i \quad \text{for } 1 \leq i \leq n, \text{ and} \tag{5.2}$$

$$r_i \leq y_i \leq h - r_i \quad \text{for } 1 \leq i \leq n. \tag{5.3}$$

Satisfying the set of constraints (5.1) guarantees that no two circles intersect, and satisfying the sets of constraints (5.2) and (5.3) means that each circle has to be packed entirely in the bin that expands from the origin $(0, 0)$ to the point (w, h).

We observe that the set of $2n$-dimensional real points that satisfy (5.1)-(5.3) is a semi-algebraic set in the field of the real numbers. Thus, the circle packing decision problem corresponds to deciding whether this semi-algebraic set is empty. We also can rewrite the constraints in (5.1)-(5.3) as $f_i(x_1, y_1, \ldots, x_n, y_n) \geq 0$, for $1 \leq i \leq s$, where s is the total number of constraints, and $f_i \in \mathbb{Q}[x_1, y_1, \ldots, x_n, y_n]$ is a polynomial with rational coefficients. Then, the circle packing problem is equivalent to deciding the truth of the formula

$$(\exists x_1)(\exists y_1) \ldots (\exists x_n)(\exists y_n) \bigwedge_{i=0}^{s} f_i(x_1, y_1, \ldots, x_n, y_n) \geq 0. \tag{5.4}$$

We can use any algorithm for the more general quantifier elimination problem to decide this formula. There are several algorithms for this problem, such as the algorithm of Tarski-Seidenberg Theorem [129], that is not elementary recursive, or the Cylindrical Decomposition Algorithm [41], that is doubly exponential in the number of variables. Since the formula corresponding to the circle packing problem contains only one block of variables (of existential quantifiers), we can use faster algorithms for the corresponding algebraic existential problem, such as the algorithms of Grigor'ev and Vorobjov [60], or of Basu *et al.* [19]. For an extensive book on algorithms for real algebraic geometry, see [18].

Sampling points of the solution. Any of the algorithms above receiving formula (5.4) as input will return "true" if, and only if, there is some arrangement of circles $\mathcal{C}$ in a bin of size $w \times h$. When the answer is "true", we are also interested in a realization of such packing. The algorithms in [19, 60] are based on critical points, that is, they also return a finite set of points that meets every semi-algebraic connected component of the semi-algebraic set. Therefore, a realization of the packing can be obtained by choosing one of such points (that is a point that corresponds to a connected component where all polynomials f_i, $1 \leq i \leq s$, are nonnegative).

Typically, the sample points are represented by a tuple $(f(x), g_0(x), \ldots, g_k(x))$ of $k+2$ univariate polynomials with coefficients in $\mathbb{Q}$, where k is the number of variables, and the value of the ith variable is $g_i(x)/g_0(x)$ evaluated at a real root of $f(x) = 0$. Since a point in a semi-algebraic set could potentially be irrational, we use the algorithm of Grigor'ev and Vorobjov [60], for which we have $g_0(x) = 1$, and thus an approximate rational solution of arbitrary precision can be readily obtained. In particular, the algorithm given in [60] implies the following result.

Theorem 5.1. *Let $f_1, \ldots, f_s \in \mathbb{Q}[x_1, y_1, \ldots, x_n, y_n]$ be polynomials with coefficients of bit-size at most m, and maximum degree 2. There is an algorithm that decides the truth of formula (5.4), with running time $m^{O(1)}s^{O(n^2)}$. In the case of affirmative answer, then the algorithm also returns polynomials $f, g_1, h_1, \ldots, g_n, h_n \in \mathbb{Q}[x]$ with coefficients of bit-size at most $m^{O(1)}s^{O(n)}$, and maximum degree $s^{O(n)}$, such that for a root x of $f(x) = 0$, the attribution $x_1 = g_1(x), y_1 = h_1(x), \ldots, x_n = g_n(x), y_n = h_n(x)$ is a realization of (5.4). Moreover, for any rational $\alpha > 0$, we can obtain $x'_1, y'_1, \ldots, x'_n y'_n \in \mathbb{Q}$, such that $|x'_i - x_i| \leq \alpha$ and $|y'_i - y_i| \leq \alpha$, $1 \leq i \leq n$, with running time $(\log(1/\alpha)m)^{O(1)}s^{O(n^2)}$.*

5.3 Approximate bin packing of large circles

In this section, we consider the particular case of Circle Bin Packing when the minimum radius of a circle is at least a constant. For this case, the maximum number of circles that fit in a bin is constant, so we can use the algorithm of Theorem 5.1 to decide whether a given set of circles can be packed in a bin in constant time.

Since Theorem 5.1 only gives us rational solutions that are close to real packings, we start with the next definition to deal with approximate circle bin packings. In the following, we denote by $\text{dist}(p, q)$ the Euclidean distance between two points p, q in the bi-dimensional space.

Definition 5.1. *Let $w, h \in \mathbb{Q}_+$ be positive numbers, and let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of circles, such that each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, and $2r_i \leq \min\{w, h\}$. For a given rational $\varepsilon \geq 0$, we say that a set of points $P = \{p_1, \ldots, p_n\}$, with $p_i = (x_i, y_i)$,*

$1 \leq i \leq n$, is an ε-packing of $\mathcal{C}$ into a rectangular bin of width w and height h, if the following hold:

$$\begin{aligned} \operatorname{dist}(p_i, p_j) \geq r_i + r_j - \varepsilon \geq 0 \quad & \text{for } 1 \leq i < j \leq n, \\ r_i - \varepsilon \leq x_i \leq w - r_i + \varepsilon \quad & \text{for } 1 \leq i \leq n, \text{ and} \\ r_i - \varepsilon \leq y_i \leq h - r_i + \varepsilon \quad & \text{for } 1 \leq i \leq n. \end{aligned}$$

We adopt the following strategy to fix the possible intersections of an approximate bin packing. For any two circles that intersect, the one with largest y-coordinate is lifted by an appropriate distance, as defined next.

Lemma 5.1. *Let r_1, r_2, h, ε be positive numbers such that $\varepsilon h \leq r_1 + r_2 \leq h$, and $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$ be points in the bi-dimensional space. If there hold $y_1 \geq y_2$, $\operatorname{dist}(p_1, p_2) \geq r_1 + r_2 - \varepsilon h$, and $p_1' = (x_1, y_1 + \sqrt{2\varepsilon}h)$, then $\operatorname{dist}(p_1', p_2) \geq r_1 + r_2$.*

Proof. By direct calculation,

$$\begin{aligned} \operatorname{dist}(p_1', p_2) &= \sqrt{(x_1 - x_2)^2 + (y_1 + \sqrt{2\varepsilon}h - y_2)^2} \\ &= \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + 2\sqrt{2\varepsilon}h(y_1 - y_2) + 2\varepsilon h^2} \\ &= \sqrt{\operatorname{dist}(p_1, p_2)^2 + 2\sqrt{2\varepsilon}h(y_1 - y_2) + 2\varepsilon h^2} \\ &\geq \sqrt{(r_1 + r_2 - \varepsilon h)^2 + 2\varepsilon h^2} \\ &= \sqrt{(r_1 + r_2)^2 - 2\varepsilon h(r_1 + r_2) + \varepsilon^2 h^2 + 2\varepsilon h^2} \\ &\geq r_1 + r_2, \end{aligned}$$

where the last inequality follows since $r_1 + r_2 \leq h$. □

To transform an ε-packing into a packing in a bin of augmented height, we shift all circles that intersect the left and right borders, and iteratively lift each circle, fixing possible intersections. This is formalized in the following.

Lemma 5.2. *Given a set of circles $\mathcal{C} = \{C_1, \ldots, C_n\}$, such that each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, and $2r_i \leq \min\{w, h\}$, and a corresponding (εh)-packing of $\mathcal{C}$ in a bin of width w and height h for some $\varepsilon \in (0, 1]$, we can find a packing of $\mathcal{C}$ in a bin of width w and height $(1 + n\sqrt{6\varepsilon})h$ in linear time.*

Proof. For $1 \leq i \leq n$, let $p_i = (x_i, y_i)$ be the center of circle C_i corresponding to the (εh)-packing. We start by modifying the given (εh)-packing to obtain a $(3\varepsilon h)$-packing in a bin of width w and height $h + 2\varepsilon h$, with the additional property that no circle intersects a border of such rectangular bin. For each $1 \leq i \leq n$, let $p_i' = (x_i', y_i')$ be the center of circle

C_i in the modified packing. The y-coordinate is defined as $y'_i = y_i + \varepsilon$, and the x-coordinate is defined as: $x'_i = x_i$ if C_i does not intersect the left or right border; $x'_i = r_i$ if C_i intersects the left border; and $x'_i = w - r_i$ if C_i intersected the right border (notice that C_i cannot intersect both the left and the right borders, since $2r_i \leq w$). Clearly, the definition of the centers guarantees that no circle intersects any border of the augmented bin. To see that the set of points p'_i is a $(3\varepsilon h)$-packing, just note that any two circles C_i, C_j are lifted by the same distance, so by the triangle inequality $\text{dist}(p'_i, p'_j) \geq \text{dist}(p_i, p_j) - 2\varepsilon h \geq r_i + r_j - 3\varepsilon h$.

Now, we transform the $(3\varepsilon h)$-packing into a packing in a bin of width w and height $h + 2\varepsilon h + (n-1)\sqrt{6\varepsilon}h \leq (1 + n\sqrt{6\varepsilon})h$. We can assume, without loss of generality, that circles $C_1, \ldots, C_n$ are ordered in non-decreasing order of the y-coordinate of their centers p'_i. For every circle C_i, $1 \leq i \leq n$, define its new center as $p''_i = (x'_i, y'_i + (i-1)\sqrt{6\varepsilon}h)$. Notice that the y-coordinate of the last circle is increased by $(n-1)\sqrt{6\varepsilon}h$, and thus no circle intersects any of the borders. Now, we show that no two circles C_i and C_j, with $1 \leq j < i \leq n$, intersect, then the lemma will follow:

$$\begin{aligned}
\text{dist}(p''_i, p''_j) &= \text{dist}((x'_i, y'_i + (i-1)\sqrt{6\varepsilon}h), (x'_j, y'_j + (j-1)\sqrt{6\varepsilon}h)) \\
&= \text{dist}((x'_i, y'_i + (i-j)\sqrt{6\varepsilon}h), (x'_j, y'_j)) \\
&\geq \text{dist}((x'_i, y'_i + \sqrt{6\varepsilon}h), (x'_j, y'_j)) \geq r_i + r_j,
\end{aligned}$$

where the last inequality follows from Lemma 5.1, and since we had a $(3\varepsilon h)$-packing. □

Definition 5.2. *Let $w, h \in \mathbb{Q}_+$ be positive numbers, and let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of circles, such that each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, and $2r_i \leq \min\{w, h\}$. We denote by $\text{OPT}_{w \times h}(\mathcal{C})$ the minimum number of rectangular bins of width w and height h that are necessary to pack $\mathcal{C}$.*

Now, we obtain an approximation algorithm for the bin packing of large circles. That is, assuming that the minimum radius of a circle, δ, is at least a given constant. First, Lemma 5.3 consider the special case that the number of different radii is bounded, and obtain a bin packing using at most the optimal number of bins, $\text{OPT}(\mathcal{C})$. Then, Theorem 5.2 reduces the general case to the case of bounded number of radii, and obtain a bin packing using at most an additional ε fraction of $\text{OPT}(\mathcal{C})$. The ideas of the following results are similar to those obtained for the rectangle bin packing [55]. In the following, we will denote the area of the circle of radius r by $Area(r)$.

Lemma 5.3. *Let $w, h \in \mathbb{Q}_+$ be positive numbers, and let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of circles, such that each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, and $2r_i \leq \min\{w, h\}$. Also, let $K = |\{r_1, \ldots, r_n\}|$ be the number of distinct radii, and let $\delta = \min_{1 \leq i \leq n} r_i$ be the minimum radius. For any given $\gamma \in (0, 1]$, we can obtain a packing of $\mathcal{C}$ in at most $\text{OPT}_{w \times h}(\mathcal{C})$ rectangular bins of width w and height $(1 + \gamma)h$.*

The running time is $O(n + (\log n)^{O(1)}(M^K)^{O(M^K)} + (\log(1/\gamma))^{O(1)}(M^2)^{O(M^2)}M^K)$, *for* $M = \lceil wh/Area(\delta)\rceil$.

Proof. Notice that a bin of width w and height h can contain at most $M = \lceil wh/Area(\delta)\rceil$ circles of $\mathcal{C}$. Consider an ordering of distinct radii $\bar{r}_1, \ldots, \bar{r}_K$. We say that a vector of nonnegative integers $c = (c_1, \ldots, c_K)$, with $\sum_{i=1}^{K} c_i \leq M$, is a configuration, and that c_i, $1 \leq i \leq K$, is the number of circles with radius $\bar{r}_i$ of c. A configuration c is said to be feasible if there is a packing in a bin of width w and height h containing all circles of c.

Let $\varepsilon = \gamma^2/(6M^2)$, and let $\alpha = \varepsilon h/4$. We enumerate each of the (at most M^K) configurations c, and use the algorithm of Theorem 5.1 to decide whether c is feasible. For each feasible configuration c, we also obtain an approximate packing, such that each circle of c has rational center p' at distance at most 2α of the center p in the packing realization. Therefore, for any two circles, with centers p_1 and p_2 in the packing realization, and approximate rational centers p'_1 and p'_2, we have $\text{dist}(p_1, p_2) - \text{dist}(p'_1, p'_2) \leq 4\alpha = \varepsilon h$. This means that the obtained approximate packing is an (εh)-packing of circles in c. For each feasible packing, we use Lemma 5.2, and obtain a packing of the circles of c in a rectangular bin of width w and height $(1+M\sqrt{6\varepsilon})h = (1+\gamma)h$. Let $\mathcal{X}$ be the set of feasible configurations, and let n_i, $1 \leq i \leq K$, denote the number of circles in $\mathcal{C}$ with radius $\bar{r}_i$. Solving the following integer program, we obtain a bin packing of size $\text{OPT}_{w\times h}(\mathcal{C})$ that contains x_c bins of configuration c for each $c \in \mathcal{X}$.

$$\begin{array}{lll} \text{minimize} & \sum_{c\in\mathcal{X}} x_c & \\ \text{subject to} & \sum_{c\in\mathcal{X}} c_i x_c \geq n_i, & 1 \leq i \leq K \\ & x_c \in \mathbb{Z}_+ & c \in \mathcal{X} \end{array}$$

The integer program has at most M^K variables, and bit-size $O(KM^K \log(n))$, and thus can be solved in time $O((\log n)^{O(1)}(M^K)^{O(M^K)})$ by Lenstra's algorithm [88]. The time to determine if each of the at most M^K configurations is feasible using Theorem 5.1 is $O((\log(1/\alpha))^{O(1)}(M^2)^{O(M^2)}) = O((\log(1/\gamma))^{O(1)}(M^2)^{O(M^2)})$. Summing up all terms, we obtain the claimed running time. □

Theorem 5.2. *Let* $w, h \in \mathbb{Q}_+$ *be positive numbers, and let* $\mathcal{C} = \{C_1, \ldots, C_n\}$ *be a set of circles, such that each circle* C_i, $1 \leq i \leq n$, *has radius* $r_i \in \mathbb{Q}_+$, *and* $2r_i \leq \min\{w, h\}$. *Also, let* $\delta = \min_{1\leq i\leq n} r_i$ *be the minimum radius. For any given* $\varepsilon, \gamma \in (0, 1]$, *there is an algorithm that packs* $\mathcal{C}$ *into at most* $(1+\varepsilon)\text{OPT}_{w\times h}(\mathcal{C})$ *rectangular bins of width* w *and height* $(1+\gamma)h$.

The running time of the algorithm is bounded by $O(n \log n + (\log n)^{O(1)}(M^K)^{O(M^K)} + (\log(1/\gamma))^{O(1)}(M^2)^{O(M^2)}M^K)$, *where* $M = \lceil wh/Area(\delta)\rceil$, *and* $K = \lceil 2/(\varepsilon Area(\delta))\rceil$.

Proof. If $n \leq K$, then we use the algorithm of Lemma 5.3 on instance $\mathcal{C}$, and obtain a packing of $\mathcal{C}$ in at most $\text{OPT}_{w\times h}(\mathcal{C})$ rectangular bins of width w and height $(1+\gamma)h$. If $n > K$, then we let $Q = \lfloor \varepsilon n Area(\delta)\rfloor > 1$, and execute the following steps:

1. sort the circles in non-increasing order of radius;
2. partition $\mathcal{C}$ in groups of up to Q consecutive circles greedily;
3. create an instance $\mathcal{C}'$ by changing the radius of each circle in $\mathcal{C}$ to the smallest radius of its group;
4. use the algorithm of Lemma 5.3 to find a packing of $\mathcal{C}'$.

We have obtained a packing P' of $\mathcal{C}'$ of size $\mathrm{OPT}_{w\times h}(\mathcal{C}')$ into rectangular bins of width w and height $(1+\gamma)h$. Notice that, with exception of circles in the first group, every circle in $\mathcal{C}$ can be mapped to a circle in $\mathcal{C}'$ of non-smaller radius, thus we can obtain a packing for $\mathcal{C}$ with the following steps: pack each circle in the first group in a new bin; for every other circle, pack at the position of the mapped circle in P'. Thus, we have obtained a packing of $\mathcal{C}$ that uses at most $\mathrm{OPT}_{w\times h}(\mathcal{C}')+Q \leq \mathrm{OPT}_{w\times h}(\mathcal{C})+\varepsilon n Area(\delta) \leq (1+\varepsilon)\mathrm{OPT}_{w\times h}(\mathcal{C})$ bins.

If $n \leq K$, then clearly the number of different radii in $\mathcal{C}$ is at most K, otherwise the number of different radii in C' is at most $\lceil \frac{n}{Q} \rceil = \lceil \frac{n}{\lfloor \varepsilon n Area(\delta) \rfloor} \rceil \leq \lceil \frac{2n}{\varepsilon n Area(\delta)} \rceil = K$. In either case, the running time due to algorithm of Lemma 5.3 is $O(n + (\log n)^{O(1)}(M^K)^{O(M^K)} + (\log(1/\gamma))^{O(1)}(M^2)^{O(M^2)}M^K)$. □

5.4 An asymptotic PTAS for circle packing into rectangular bins

In this section, we consider the bin packing problem of circles of any size. The main idea works as follows. First, we will use the algorithm from Section 5.3 and obtain a packing of "large" circles into bins of the original dimensions. Then, we consider bins with a small fraction of the original size, and solve the problem of packing the "small" circles in such bins recursively. To obtain a solution of the original problem, we place each obtained small bin into the free space of the packing obtained for large circles. The key idea is that, if the dimensions of the small bins are much smaller than the large circles, then the waste of space in the packing for the large circles is proportional to a fraction of the area of the large circles. Moreover, if the size of such small bin is also much larger than the small circles, then restricting the packing of small circles to small bins does not increase much the cost of a solution.

In the following, if B is a circle or rectangle, then we denote by $Area(B)$ the area of B. Also, if D is a set, then $Area(D) = \sum_{B\in D} Area(B)$. We formalize the packing steps in Algorithm 5.1. An informal description is given thereafter.

Algorithm 5.1. *Circle Bin Packing Algorithm*

Consider the parameters r and γ, such that r is a positive integer multiple of 3, and γ a number in $(0, 1]$. The algorithm receives a set of circles $\mathcal{C} = \{C_1, \ldots, C_n\}$, and numbers w, h, such that $w \leq h$, and hr/w is an integer. Moreover, each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, with $2r_i \leq w$. The algorithm returns a packing of $\mathcal{C}$ into a set of bins of width w and height $(1+\gamma)h$.

1. Let $\varepsilon = 1/r$;
2. For every integer $i \geq 0$, define $G_i = \{C_j \in \mathcal{C} : \varepsilon^{2i} w \geq 2r_j > \varepsilon^{2(i+1)} w\}$;
3. For each $0 \leq j < r$, define $H_j = \{C \in G_i : i \equiv j \pmod{r}\}$;
4. Find an integer t such that $Area(H_t) \leq \varepsilon Area(\mathcal{C})$;
5. Place each circle of H_t into its bounding box, and pack them in separate bins of width w and height $(1+\gamma)h$ using NFDH strategy [107];
6. For every integer $j \geq 0$, define $S_j = \{C \in G_i : t + (j-1)r + 1 \leq i \leq t + jr - 1\}$;
7. Define $w_0 = w$, $h_0 = h$ and $w_j = h_j = \varepsilon^{2(t+(j-1)r)+1} w$ for every $j \geq 1$;
8. Let $F_0 = \emptyset$;
9. For every $j \geq 0$:
   (a) Use the algorithm of Theorem 5.2 to obtain a packing of circles S_j into bins of width w_j and height $(1+\gamma)h_j$. Let P_j be the set of such bins;
   (b) Let A_j be a set of $\max\{|P_j| - |F_j|, 0\}$ new empty bins of width w_j and height $(1+\gamma)h_j$;
   (c) Place each bin of P_j over one distinct bin of $F_j \cup A_j$;
   (d) Set $F_{j+1} = \emptyset$, and $U_j = \emptyset$; (U_j *is used only in the analysis*)
   (e) For each bin B of $F_j \cup A_j$:
      - Let V be the set of bins corresponding to the cells of the grid with cells of width w_{j+1} and height $(1+\gamma)h_{j+1}$ over B;
      - Add to F_{j+1} all bins in V that do not intersect any circle of S_j.
      - Add to U_j all bins in V that intersect a circle of S_j.
   (f) If all circles are packed, go to step 10.
10. Place the bins $A_0, A_1, \ldots$ into the minimum number of bins of width w and height $(1+\gamma)h$.

Notice that if hr/w is not integer, then after the first iteration of the algorithm, we could round up the height h_0 of the bins in P_0 to next integer multiple of w/r. Since the obtained solution has the property that all circles S_j are completely packed in a bin of the

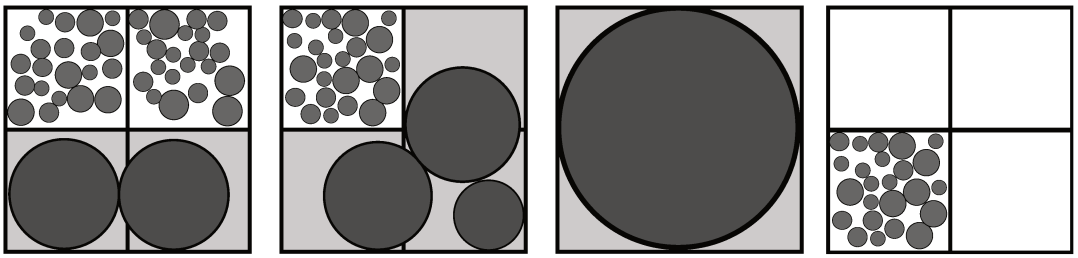
				H_t:
S_0:		G_0,	$\ldots$ G_{t-1}	G_t
S_1:	G_{t+1},	G_{t+2},	$\ldots$ G_{t+r-1}	G_{t+r}
S_2:	G_{t+r+1},	G_{t+r+2},	$\ldots$ G_{t+2r-1}	G_{t+2r}
S_j:	$G_{t+(j-1)r+1}$,	G_{t+jr+2},	$\ldots$ G_{t+jr-1}	G_{t+jr}
		$\vdots$		$\vdots$

Figure 5.1: Partitioning of circles

corresponding group P_j, we can modify such solution, by moving the rounded bins in P_1 to new bins of the original height, with additional cost of at most $1/r|P_0| \leq \varepsilon \mathrm{OPT}(\mathcal{C})$. For the sake of clarity, from now on we assume that hr/w is integer.

The algorithm partitions the set of circles into sets $H_t, S_0, S_1, \ldots$, as in Figure 5.1. For each j, we consider the subproblem of packing the circles of S_j into bins of size $w_j \times h_j$. The circles in S_j are large when compared to bins of size $w_j \times h_j$, and so we can use algorithm of Theorem 5.2. Notice that the bins considered in the next iteration have size $w_{j+1} \times h_{j+1}$, and are much smaller than the circles of S_j. Also, since the circles in H_t are considered separately, each remaining unpacked circle (of $S_{j+1}, S_{j+2}, \ldots$) fits in a bin of size $w_{j+1} \times h_{j+1}$. Figure 5.2 illustrates this process.

In each iteration $j \geq 0$, the algorithm keeps a set F_j of free bins of size $w_j \times (1+\gamma)h_j$ obtained from previous iterations. We obtain a packing of circles of S_j into a set of bins P_j. Then, we place such bins over the free space of F_j, or over additional bins A_j, if necessary. The set of sub-bins of $F_j \cup A_j$ of size $w_{j+1} \times (1+\gamma)h_{j+1}$ that intersect circles of S_j are included in the set U_j, and the remaining sub-bins are saved in F_{j+1} for the next



First, the large circles are packed using algebraic quantifier elimination algorithms. Then, the bins are divided in small sub-bins, and those that intersect any large circle are marked as used (shaded squares). Finally, the remaining circles are packed recursively, and placed over the free space of original bins, or over new bins.

Figure 5.2: Recursive packing

iteration. The algorithm finishes when all circles are considered, and the created bins $A_0, A_1, \ldots$ are combined into bins of original size.

Consider a bin B of width w_B and height h_B. Given w and h, we say that B *divides* $w \times h$ if either $w_B = w$, and $h_B = h$, or $w_B = h_B = \varepsilon^{2(t+(j-1)r)+1} w$ for some $j \geq 1$. If D is a set of bins, then we say that D divides $w \times h$ if every $B \in D$ divides $w \times h$. In what follows, we assume that we have run Algorithm 5.1, giving as input positive numbers $w, h \in \mathbb{Q}_+$, with $w \leq h$, a set of circles $\mathcal{C} = \{C_1, \ldots, C_n\}$, such that each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, and $2r_i \leq \min\{w, h\}$, and parameters $r \in \mathbb{Z}_+$, $\gamma \in (0, 1]$.

Definition 5.3. *Let $j \geq 0$. If B is a bin that divides $w \times (1+\gamma)h$, then we denote by $\mathrm{Gr}_j(B)$ the set of bins in the grid with cells of width w_j and height $(1+\gamma)h_j$ over B. Also, if D is a set of bins that divides $w \times (1+\gamma)h$, then $\mathrm{Gr}_j(D) = \cup_{B \in D} \mathrm{Gr}_j(B)$. Similarly, if B' is a bin that divides $w \times h$, then we denote by $\mathrm{Gr}'_j(B')$ the set of bins in the grid with cells of width w_j and height h_j over B', and, if D' is a set of bins that divides $w \times h$, then $\mathrm{Gr}'_j(D') = \cup_{B' \in D'} \mathrm{Gr}'_j(B')$.*

Definition 5.4. *Let B be a rectangle or circle. We denote by $\mathrm{Ar}(B) = Area(B)/((1+\gamma)wh)$ the proportional area of B over bins of width w and height $(1+\gamma)h$, and by $\mathrm{Ar}'(B) = Area(B)/(wh)$ the proportional area of A over bins of width w and height h. Also, if D is a set of rectangles or circles, then $\mathrm{Ar}(D) = \sum_{B \in D} \mathrm{Ar}(B)$, and $\mathrm{Ar}'(D) = \sum_{B \in D} \mathrm{Ar}'(B)$.*

Step (9) of Algorithm 5.1 does not generate bins of width w and height $(1+\gamma)h$. Rather, it creates a collection of sets $A_0, A_1, \ldots$ that divide $w \times (1+\gamma)h$. However, since each created bin has dimensions that are obtained by multiplying a power of $1/r$ times the original dimensions, they can easily be combined in a set of bins of width w and height $(1+\gamma)h$ using almost the same area. This is made precise in the following remark.

Remark 5.1. *If there is a packing of $\mathcal{C}$ into a set of bins D that divides $w \times (1+\gamma)h$, then there is a packing of $\mathcal{C}$ in $\lceil \mathrm{Ar}(D) \rceil$ bins of width w and height $(1+\gamma)h$. Analogously, if there is a packing of $\mathcal{C}$ into a set of bins D' that divides $w \times h$, then there is a packing of $\mathcal{C}$ in $\lceil \mathrm{Ar}'(D') \rceil$ bins of width w and height h.*

For some j, the area of U_j is not fully used, since there might be cells of U_j that partially intersect circles of S_j. This waste is bounded by the following lemmas.

Lemma 5.4. *Let $C_i \in S_j$ be a circle packed in a bin B that divides $w \times (1+\gamma)h$, and let $D \subseteq \mathrm{Gr}_{j+1}(B)$ be the subset of bins in the grid that intersect circle C_i, but are not contained in C_i, then $\mathrm{Ar}(D) \leq 16\varepsilon \mathrm{Ar}'(C_i)$.*

Proof. Each $B \in D$ has width $w_{j+1} = \varepsilon^{2(t+jr)+1} w$, and height $h_{j+1} = (1+\gamma)\varepsilon^{2(t+jr)+1} w$. Also, since $C_i \in S_j$, then $2r_i \geq \varepsilon^{2(t+jr)} w$. Consider the circles C_+ and C_-, centered at the

same position as C_i, and with radii $r_+ = r_i + w_{j+1} + h_{j+1}$, and $r_- = r_i - w_{j+1} - h_{j+1}$. Notice that every $B \in D$ is contained in $C_+ \setminus C_-$. We obtain

$$\begin{aligned} Area(D) &\leq Area(C_+) - Area(C_-) = \pi(r_+^2 - r_-^2) \\ &\leq \left((1 + 2\varepsilon + (1+\gamma)2\varepsilon)^2 - (1 - 2\varepsilon - (1+\gamma)2\varepsilon)^2\right)\pi r_i^2 \\ &\leq (1+\gamma)16\varepsilon Area(C_i). \end{aligned}$$

Therefore, $\mathrm{Ar}(D) = Area(D)/((1+\gamma)wh) \leq (16\varepsilon Area(C_i))/(wh) = 16\varepsilon \mathrm{Ar}'(C_i)$. □

Similarly, one obtain.

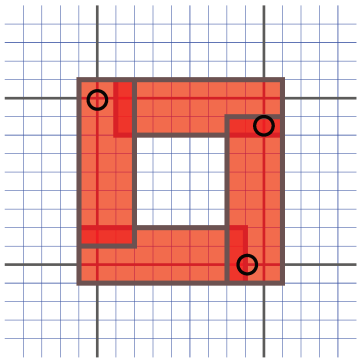
Lemma 5.5. *Let $C \in S_j$ be a circle packed in a bin B that divides $w \times h$, and let $D \subseteq \mathrm{Gr}'_{j+1}(B)$ be the subset of bins in the grid that intersect circle C, but are not contained in C. Then $\mathrm{Ar}'(D) \leq 16\varepsilon \mathrm{Ar}'(C)$.*

The following lemma shows that requiring that each set of circles S_j be packed into bins of size $w_j \times h_j$ does not increase much the solution cost. This fact is central to the algorithm, since it allows iteratively packing sets S_j's.

Lemma 5.6. *There is a packing of $\mathcal{C} \setminus H_t$ into a set of bins D that divides $w \times h$ with $\mathrm{Ar}'(D) \leq (1 + 28\varepsilon)\mathrm{OPT}_{w\times h}(\mathcal{C})$, such that for every $j \geq 0$, there is a packing of S_j into a set of bins $P'_j \subseteq \mathrm{Gr}'_j(D)$.*

Proof. We transform an optimal packing *Opt* of $\mathcal{C}$ into a packing D with the desired properties. The idea is moving circles of S_j that intersect lines of the grid of size $w_j \times h_j$ to free bins that respect the grid. The next algorithm keeps the invariant that, at the start of iteration $j \geq 1$, the set R_j contains free space to pack all such intersecting circles. Steps (3a)-(3c) move intersecting circles to bins of R_j, and steps (3d)-(3f) make sure that there are enough free bins in R_{j+1} respecting the grid of size $w_{j+1} \times h_{j+1}$.

1. Let R_1 be a set of $12\varepsilon(wh)/(w_1h_1)|Opt|$ new bins of width w_1 and height h_1;
2. Let $D_0 = Opt \cup R_1$;
3. For each $j \geq 1$:
   (a) Let $L_j = \emptyset$;
   (b) For each bin $B \in \mathrm{Gr}'_j(Opt)$:
      i. Let W be the set of circles in S_j that intersect the boundary of B;
      ii. Let V be a set of 4 new bins (2 of width $3\varepsilon w_j$ and height h_j, and 2 of width w_j and height $3\varepsilon h_j$) placed over the boundary of B, so that each circle in W is contained in one bin of V (see Figure 5.3);
      iii. For each cell $B' \in \mathrm{Gr}'_{j+1}(V)$, let $\phi(B')$ be the cell of $\mathrm{Gr}'_{j+1}(Opt)$ under B';



Each circle of W has diameter at most εw_j. To rearrange the rectangles into bins of size $w_j \times w_j$ (of R_j), we use one side of length w_j. To ensure every circle is in a rectangle, the other side has length $3\varepsilon w_j$ (see lower circle).

Figure 5.3: Circles that intersect grid lines

iv. Remove each circle of W from the packing D_j and pack it over one bin of V preserving the arrangement;

v. Add V to L_j;

(c) Make groups of $r/3$ bins (of equal sizes) of L_j forming a new bin of width w_j and height h_j, and place this bin over one distinct bin of R_j;

(d) Let $R_{j+1} = \emptyset$;

(e) Let $N_j = \emptyset$;

(f) For each bin $B \in \mathrm{Gr}_{j+1}(L_j)$, consider the cases:

i. If B does not intersect any circle of S_j, then add B to R_{j+1};

ii. If B is contained in some circle of S_j, then add $\phi(B)$ to R_{j+1};

iii. If B intersects, but is not contained in a circle of S_j, then create a new bin of width w_{j+1} and height h_{j+1} and add to R_{j+1}, and to N_j;

(g) Let $D_j = D_{j-1} \cup N_j$;

(h) If $\mathcal{C} \setminus H_t = S_0 \cup \cdots \cup S_j$, make $D = D_j$, and stop.

We will show that D is a bin packing of $\mathcal{C}$.

First, notice that $Area(L_j) = Area(R_1)$ for every $j \geq 1$. Indeed, we have $|\mathrm{Gr}'_j(Opt)| = |Opt|(wh)/(w_j h_j)$, and for each bin B in $\mathrm{Gr}'_j(Opt)$, we create a set V such that $Area(V) = 4 \cdot 3\varepsilon w_j h_j$, so $Area(L_j) = 12\varepsilon(wh)|Opt| = Area(R_1)$. Also, notice that $Area(R_{j+1}) = Area(L_j)$, since for each $B \in \mathrm{Gr}_{j+1}(L_j)$, $Area(R_{j+1})$ is increased by $Area(B)$. This is clear if steps (3(f)i) and (3(f)iii) are executed, so it enough to show that, if step (3(f)ii) is executed for bins B, B' in $\mathrm{Gr}_{j+1}(L_j)$, with $\phi(B) = \phi(B')$, then $B = B'$. By the definition

in step (3(b)iii), B and B' must intersect the same region of a circle in S_j. Since each such circle is contained in exactly one rectangle of L_j, it follows that, indeed, $B = B'$. Then $Area(R_{j+1}) = Area(L_j)$. Therefore $Area(R_j) = Area(R_1)$ for every $j \geq 1$.

Now, we show that, at the end of iteration $j \geq 0$, the following claims hold:

1. D_j is a packing of $\mathcal{C}$;
2. for each $\ell = 0, \ldots, j$, there is a packing of S_ℓ into a set $P'_\ell \subseteq \mathrm{Gr}'_\ell(D_j)$ of bins of width w_ℓ and height h_ℓ;
3. the bins in $R_{j+1} \subseteq \mathrm{Gr}_{j+1}(D_j)$ do not intersect any circle of $\mathcal{C}$.

By induction on j. For $j = 0$, the claims are clear. So let $j \geq 1$, and assume that the claims are true for $j - 1$.

Proof of claim 1: Clearly, L_j is a packing of the circles that were removed from the original packing D_{j-1}. Since r is a multiple of 3, the step (3c) is well defined, and thus we can place rectangles of L_j over bins of R_j. After step (3c), we have a bin packing of $\mathcal{C}$, since by the induction hypothesis, at the beginning of iteration j, the set R_j did not intersect any circle. This shows claim 1.

Proof of claim 2: Since, at the end of iteration, each circle of S_j that intersected a line of the grid $\mathrm{Gr}_j(D_{j-1})$ is completely contained in a bin of $R_j \subseteq \mathrm{Gr}_j(D_j)$, we obtain claim 2.

Proof of claim 3: If step (3(f)i) or step (3(f)iii) is run, then we add a free bin to R_{j+1}. Thus, we only need to argue that whenever step (3(f)ii) is run, the bin $\phi(B)$ does not intersect any circle. Let C be the circle that contains B, so that at the beginning of the iteration, $\phi(B)$ was contained in C. Since C was moved to L_j, $\phi(B)$ does not intersect any circle when step (3(f)ii) is run. This completes the induction.

Finally, we may calculate the cost of the modified solution D. Let m be the number of iterations of the algorithm. Notice that D is the disjoint union of Opt, R_1, N_1, $\ldots$, N_m. We get

$$\begin{aligned}
\mathrm{Ar}'(D) &= \mathrm{Ar}'(Opt) + \mathrm{Ar}'(R_1) + \textstyle\sum_{j=1}^{m} \mathrm{Ar}'(N_j) \\
&\leq \mathrm{Ar}'(Opt) + 12\varepsilon(wh)|Opt|/(wh) + \textstyle\sum_{j=1}^{m} 16\varepsilon \mathrm{Ar}'(S_j) \\
&\leq \mathrm{Ar}'(Opt) + 12\varepsilon \mathrm{Ar}'(Opt) + 16\varepsilon \mathrm{Ar}'(\mathcal{C}) \\
&= (1 + 28\varepsilon)\mathrm{OPT}_{w\times h}(\mathcal{C}),
\end{aligned}$$

where the first inequality comes from Lemma 5.5. □

In addition to requiring that each set of circles S_j be packed into bins of the grid with cells of width w_j and height h_j, we also require that the bins used to pack $S_{j+1}, S_{j+2}, \ldots$ do not intersect circles of $S_1, \ldots, S_{j-1}$. Again, this restriction only increases the cost of a solution by a small fraction of the optimal value, as shown by the next lemma.

Lemma 5.7. *There is a packing of $\mathcal{C} \setminus H_t$ into a set of bins D that divides $w \times h$ with $\mathrm{Ar}'(D) \leq (1+44\varepsilon)\mathrm{OPT}_{w\times h}(\mathcal{C})$, such that for every $j \geq 0$, there is a packing of S_j in a set of bins $P'_j \subseteq \mathrm{Gr}'_j(D)$. Moreover, if $B \in P'_j$, then B does not intersect any circle $C_i \in S_\ell$ for $\ell < j$.*

Proof. We start with a solution D' given by Lemma 5.6, and corresponding bin packings P'_j for each j. Without loss of generality, we assume that for each $B \in P'_j$, there is a circle $C_i \in S_j$ packed in B, since otherwise we could simply remove B from P'_j. We execute the following steps:

1. Let $P''_0 = P'_0$, and $D = D'$;
2. For each $j \geq 1$:
   (a) Let $V \subseteq \mathrm{Gr}'_j(D)$ that intersect a circle $C_i \in S_{j-1}$, but is not contained in C_i.
   (b) Create a set V' of $|V|$ new bins of width w_j and height h_j, and move the circles in V to V' preserving the arrangement.
   (c) Let $P''_j = P'_j \cup V'$, and add V' to D.
   (d) If $\mathcal{C} \setminus H_t = S_0 \cup \cdots \cup S_j$, stop.

Clearly the output of this procedure is a bin packing of $\mathcal{C}$, and a simple induction shows that at the end of iteration m, for every $j \leq m$, set $P''_j \subseteq \mathrm{Gr}'_j(D)$ is a bin packing of S_j, and for every $B \in P'_j$, B does not intersect any circle $C_i \in S_\ell$ for $\ell < j$. Using Lemma 5.5, we obtain

$$\begin{aligned}\mathrm{Ar}'(D) &\leq \mathrm{Ar}'(D') + \textstyle\sum_{j\geq 0} 16\varepsilon \mathrm{Ar}'(S_j) \\ &\leq (1+28\varepsilon)\mathrm{OPT}(\mathcal{C}) + 16\varepsilon\mathrm{OPT}(\mathcal{C}) \\ &= (1+44\varepsilon)\mathrm{OPT}(\mathcal{C}).\end{aligned}$$

□

Now we are ready to calculate the cost of the solution generated by Algorithm 5.1.

Lemma 5.8. *Algorithm 5.1 produces a packing of $\mathcal{C}$ into bins of width w and height $(1+\gamma)h$ using at most $(1+O(\varepsilon))\mathrm{OPT}_{w\times h}(\mathcal{C}) + 2$ bins.*

Proof. Let D be the packing of $\mathcal{C}$ obtained from Lemma 5.7, and let sets P'_j, $j \geq 0$, be the packings obtained for each S_j. Notice that for each $j \geq 0$, $\mathrm{OPT}_{w_j\times h_j}(S_j) \leq |P'_j|$. Recall that Algorithm 5.1 uses Theorem 5.2 to obtain a packing P_j of S_j with $|P_j| \leq (1+\varepsilon)\mathrm{OPT}_{w_j\times h_j}(S_j)$ bins. Therefore, $|P_j| \leq (1+\varepsilon)|P'_j|$, and thus $\mathrm{Ar}(P_j) \leq (1+\varepsilon)\mathrm{Ar}'(P'_j)$.

First, we show that for every $m \geq 0$, we have $\mathrm{Ar}(F_m) = \sum_{j=0}^{m-1} \mathrm{Ar}(A_j) - \sum_{j=0}^{m-1} \mathrm{Ar}(U_j)$. By induction on m. For $m = 0$, the claim is clear, so suppose that the claim is true for

some $m \geq 0$. We have,

$$\begin{aligned}\mathrm{Ar}(F_{m+1}) &= \mathrm{Ar}(F_m \cup A_m) - \mathrm{Ar}(U_m)\\ &= \mathrm{Ar}(F_m) + \mathrm{Ar}(A_m) - \mathrm{Ar}(U_m)\\ &= \textstyle\sum_{j=0}^{m-1} \mathrm{Ar}(A_j) - \sum_{j=0}^{m-1} \mathrm{Ar}(U_j) + \mathrm{Ar}(A_m) - \mathrm{Ar}(U_m)\\ &= \textstyle\sum_{j=0}^{m} \mathrm{Ar}(A_j) - \sum_{j=0}^{m} \mathrm{Ar}(U_j).\end{aligned}$$

Now, we show that for every $m \geq 0$, we have $\sum_{j=0}^{m} \mathrm{Ar}(A_j) \leq (1+\varepsilon)\mathrm{Ar}'(D) + 15\varepsilon \sum_{j=0}^{m} \mathrm{Ar}'(S_j)$. Again, by induction on m. The case $m = 0$ is clear, so suppose that this is true for $m \geq 0$. We consider two cases. If $|A_{m+1}| = 0$, then by the induction hypothesis we have

$$\begin{aligned}\textstyle\sum_{j=0}^{m+1} \mathrm{Ar}(A_j) &= \textstyle\sum_{j=0}^{m} \mathrm{Ar}(A_j)\\ &\leq (1+\varepsilon)\mathrm{Ar}'(D) + 15\varepsilon \textstyle\sum_{j=0}^{m} \mathrm{Ar}'(S_j)\\ &\leq (1+\varepsilon)\mathrm{Ar}'(D) + 15\varepsilon \textstyle\sum_{j=0}^{m+1} \mathrm{Ar}'(S_j).\end{aligned}$$

If $|A_{m+1}| > 0$, then we know that $\mathrm{Ar}(A_{m+1}) + \mathrm{Ar}(F_{m+1}) = \mathrm{Ar}(P_{m+1})$, so we get

$$\begin{aligned}\textstyle\sum_{j=0}^{m+1} \mathrm{Ar}(A_j) &= \textstyle\sum_{j=0}^{m} \mathrm{Ar}(A_j) + \mathrm{Ar}(A_{m+1})\\ &= (\textstyle\sum_{j=0}^{m} \mathrm{Ar}(U_j) + \mathrm{Ar}(F_{m+1})) + (\mathrm{Ar}(P_{m+1}) - \mathrm{Ar}(F_{m+1}))\\ &\leq (1+16\varepsilon) \textstyle\sum_{j=0}^{m} \mathrm{Ar}'(S_j) + (1+\varepsilon)\mathrm{Ar}'(P'_{m+1})\\ &= (1+\varepsilon)(\textstyle\sum_{j=0}^{m} \mathrm{Ar}'(S_j) + \mathrm{Ar}'(P'_{m+1})) + 15\varepsilon \textstyle\sum_{j=0}^{m} \mathrm{Ar}'(S_j)\\ &\leq (1+\varepsilon)\mathrm{Ar}'(D) + 15\varepsilon \textstyle\sum_{j=0}^{m+1} \mathrm{Ar}'(S_j).\end{aligned}$$

The first inequality comes from Lemma 5.4, and the second inequality comes from the fact that bins in P'_{m+1} do not intersect circles in $S_1, \ldots S_m$.

It follows that the total area of bins used for circles in $\mathcal{C} \setminus H_t$ is

$$\begin{aligned}\textstyle\sum_{j\geq 0} \mathrm{Ar}(A_j) &\leq (1+\varepsilon)\mathrm{Ar}'(D) + 15\varepsilon \textstyle\sum_{j\geq 0} \mathrm{Ar}'(S_j)\\ &\leq (1+\varepsilon)(1+44\varepsilon)\mathrm{OPT}(\mathcal{C}) + 15\varepsilon\mathrm{OPT}(\mathcal{C})\\ &\leq (1+105\varepsilon)\mathrm{OPT}(\mathcal{C}).\end{aligned}$$

For the circles in H_t, we used the NFDH algorithm to pack the bounding boxes of the circles, thus the density of the packing in each used bin of width w and height $(1+\gamma)h$, with the exception of the last, is at least $1/4$. Since each circle occupies a fraction of $\pi/4$ of its bounding box, the total number of bins used for H_t is bounded by

$$\lceil 4 \cdot 4/\pi \mathrm{Ar}(H_t) \rceil \leq \lceil 4 \cdot 4/\pi\varepsilon \mathrm{Ar}(\mathcal{C}) \rceil \leq \lceil 16/\pi\varepsilon \mathrm{OPT}(\mathcal{C}) \rceil.$$

Therefore, noticing that the sets of bins $A_0, A_1, \ldots$ divide $w \times (1+\gamma)h$, we obtain that the total number of bins of width w and height $(1+\gamma)h$ used to pack $\mathcal{C}$ is

$$\lceil (1+105\varepsilon)\mathrm{OPT}(\mathcal{C}) \rceil + \lceil 16/\pi\varepsilon \mathrm{OPT}(\mathcal{C}) \rceil \leq (1+111\varepsilon)\mathrm{OPT}(\mathcal{C}) + 2. \qquad \square$$

Next, we show that the running time of Algorithm 5.1 is polynomial.

Lemma 5.9. *Suppose $h/w = (1/\varepsilon)^{O(1/\varepsilon)}$. For any constants r and γ, the running time of Algorithm 5.1 is bounded by $O(n^2)$.*

Proof. Notice that in an implementation of Algorithm 5.1, we only run step (9) for non empty sets S_j. In each such iteration, we only need to account for the running time of step (9a), and the running time to pack the bins of P_j in free space of current bins.

We consider the following alternative (equivalent) procedure for step (9a): scale each the radius of each circle in S_j by $1/w_j$ and obtain set S'_j; run algorithm of Theorem 5.2 with the scaled S'_j and bins of width 1 and height $h_j/w_j \leq h/w$, and obtain packing P'_j; scale the obtained packing P'_j by w_j, and obtain packing P_j. Let $\delta = \varepsilon^{2r-1}$, and notice that, for every j, each circle in S'_j has radius at least $\varepsilon^{2(t+jr)}w/(2w_j) = \varepsilon^{2(t+jr)}w/(2\varepsilon^{2(t+(j-1)r)+1}w) = \delta$. Let $K = \lceil 2/(\varepsilon Area(\delta))\rceil = (1/\varepsilon)^{O(1/\varepsilon)}$, and $M = \lceil (1 \cdot h/w)/Area(\delta)\rceil = (1/\varepsilon)^{O(1/\varepsilon)}$. Therefore, from Theorem 5.2 we get that the total running time of step (9a) is $O(n(\log n)^{O(1)})$, for constants r and γ.

To pack a bin of P_j, $j \geq 0$, we need to find one element $\mathrm{Gr}_j(A_0 \cup \cdots \cup A_{j-1})$ that is not in $U_0, \ldots, U_{j-1}$, that is, finding a grid cell that does not intersect any circle of $S_0 \cup \cdots \cup S_{j-1}$. To verify whether a grid cell intersects a circle of S_ℓ, $0 \leq \ell \leq j-1$, it is enough to list the elements of U_ℓ that intersect the border of the circle. There are at most $(1/\varepsilon)^{O(1)}$ such elements per circle, so at most $n(1/\varepsilon)^{O(1)}$ elements are listed to find each free cell. Therefore, all bins can be packed in time $O(n^2(1/\varepsilon)^{O(1)})$. □

Combining Lemmas 5.8 and 5.9, we obtain the following result.

Theorem 5.3. *Let $w, h \in \mathbb{Q}_+$ be positive numbers, and let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a set of circles, such that each circle C_i, $1 \leq i \leq n$, has radius $r_i \in \mathbb{Q}_+$, and $2r_i \leq \min\{w, h\}$. For any given constants $\varepsilon, \gamma \in (0, 1]$, we can obtain a packing of $\mathcal{C}$ into at most $(1+\varepsilon)\mathrm{OPT}_{w\times h}(\mathcal{C}) + 2$ rectangular bins of width w and height $(1+\gamma)h$. The algorithm runs in time $O(n^2)$.*

Proof. If $h/w < 1/\varepsilon^2$, the theorem is immediate, so assume $h/w \geq 1/\varepsilon^2$.

Consider an optimal solution *Opt* of bins of width w and height h. We will transform this solution into a packing of bins of width w and height w/ε. First, split each bin of *Opt* in sub-bins of width w and height w/ε. Then, remove all circles that intersect consecutive sub-bins. The total area of removed circles is at most $|Opt|(w \cdot 2w)\lfloor h/(w/\varepsilon)\rfloor \leq |Opt|2wh\varepsilon$. Finally, place the removed circles into their bounding boxes and pack them into additional bins of width w and height w/ε using NFDH strategy. Since each additional bin has density of at least $\pi/16$ (with exception of the last), the number of such bins is bounded by $\lceil (16/\pi\,|Opt|2wh\varepsilon)/(w(w/\varepsilon))\rceil$. Therefore, we know that $\mathrm{OPT}_{w\times w/\varepsilon}$ is bounded by $|Opt|\lceil h/(w/\varepsilon)\rceil + \lceil (16/\pi\,|Opt|2wh\varepsilon)/(w(w/\varepsilon))\rceil \leq (1 + O(\varepsilon))|Opt|(h/w)\,\varepsilon + 2$.

Now, we use Lemma 5.8 and obtain a packing of $\mathcal{C}$ in bins of width w and height $(1+\gamma)w/\varepsilon$ of cost at most

$$(1+O(\varepsilon))\mathrm{OPT}_{w\times w/\varepsilon}+2 \leq (1+O(\varepsilon))|Opt|(h/w)\,\varepsilon+4.$$

By joining each group of $\lfloor h/(w/\varepsilon)\rfloor$ bins, we obtain a packing into bins of width w and height $(1+\gamma)h$ of cost at most

$$\left\lceil \frac{(1+O(\varepsilon))|Opt|(h/w)\,\varepsilon+4}{\lfloor h/(w/\varepsilon)\rfloor} \right\rceil \leq (1+O(\varepsilon))|Opt|+2,$$

where the inequality follows from the fact that $h/w \geq 1/\varepsilon^2$, and assuming ε sufficiently small. To complete the theorem, it is enough to notice that the running time is given by Lemma 5.9. □

It is now straightforward to extend this theorem to the Circle Strip Packing.

Theorem 5.4. *Let $\mathcal{C}=\{C_1,\ldots,C_n\}$ be a set of circles, such that each circle C_i, $1\leq i\leq n$, has radius $r_i\in\mathbb{Q}_+$, and $2r_i\leq 1$. For any given constant $\varepsilon\in(0,1]$, we can obtain a packing of $\mathcal{C}$ in a strip of unit width and height $(1+\varepsilon)\mathrm{OPTS}(\mathcal{C})+O(1/\varepsilon)$, where $\mathrm{OPTS}(\mathcal{C})$ is the height of the minimum packing of $\mathcal{C}$ in a strip of unit width. The algorithm runs in time $O(n^2)$.*

Chapter remarks

While there are many works that give approximation algorithms for problems of packing rectangular items in square bins or in a strip, the packing of circles is done mostly through heuristics and other numerical methods. One obstacle to obtain good approximation algorithms is the difficulty of bounding the unused free space between "large" circles. This chapter presents a novel iterative packing algorithm to smartly use this free space, leading to the first approximation algorithm for these natural circle packing problems. Indeed, we give APTAS's for the Circle Bin Packing and for the Circle Strip Packing.

Chapter 6

Concluding remarks

Though this work was led from the theoretical perspective, it tackled problems that rise through all the decision-making process of a supply chain. These decisions may include, for example, packing items for later transportation, coordinating to where and when to place orders for the inventory replenishment, and deciding where to install facilities of a distribution network. Obtaining approximation algorithms for these problems does not only provide solutions with guaranteed quality, but also helps unveiling ideas that find practical applications in the industry. This thesis presented approximation algorithms for a broad list of problems that appear in many areas of a supply chain. Several techniques are proposed, and for some of the considered problems, no constant-factor approximations were known before. A listing of the obtained results follows.

- A lower bound on the approximation factor of 2.04 to the SMFLP is given, and it is shown that a standard LP-rounding algorithm achieves this factor, and thus has the best possible factor for this problem.

- The upper bound factor-revealing programs (UPFRP) are introduced. They allow one to derive an approximation factor directly from a straightforwardly obtained linear program, in opposition to long proofs that depend on guessing a general dual solution for the lower bound factor-revealing programs.

- Using UPFRP's, we showed that primal-dual algorithms that achieve factors 1.861, 1.61, 1.52, and 1.575 for the MFLP [72, 104], have factors 2.87, 2.43, 2.17, and 2.04 for the SMFLP. Also, we showed that the algorithm with factor 1.861 is, in fact, a 1.816-approximation for the MFLP, and this factor is very tight.

- More general facility location problems whose distance functions are either the power of a metric, or satisfy other relaxed metrics, are studied. We give constant approximations for these generalized problems.

- The continuous version of the FLP, when facilities may be opened in any point of the space, was reduced to the corresponding discrete problem, by using the so called approximate center sets.

- Approximate center sets for the L_2^α-norm, for $\alpha \geq 1$, are given. This distance function corresponds to the Euclidean norm raised to a power α, and no approximate center sets for this function were known previously. Using these center sets, and other results from this thesis, constant approximations for ConFL with this norm for $\alpha \geq 1$ are given. For the particular case of $\alpha = 1$, we obtain a PTAS. Also using the center sets, we obtain a PTAS for the k-clustering problem with L_2^α-norm.

- A 2.77-approximation for the PDP based on a combination of diverse LP-rounding techniques is given. This is the first approximation for a problem that integrates transportation and inventory management problem with dynamic allocation of warehouses to retailers.

- A 5-approximation for the variant of the PDP with retailer ordering costs is given. This algorithm combines the shift procedure used for the OWMR [91] with an involved filtering technique.

- We considered the OWMR with independent retailer and warehouse holding costs, and gave a 5-approximation based on a novel primal-dual approach, that extends the wave mechanism used for the JRP [90]. This answers positively the question whether OWMR admitted a primal-dual approximation [91], and gives the first constant approximation for OWMR under this holding cost model.

- We notice that the Multilevel JRP may be reduced to the Multistage Assembly Problem, thus obtaining a constant approximation for an important particular case of the Submodular JRP, when the joint ordering cost is given by a submodular set-function over the participating retailers.

- We give an APTAS for the circle bin packing problem with resource augmentation in one-dimension, and an APTAS for the circle strip packing problem. These are the first approximations for these problems, and are based on several innovative techniques, such as iteratively distinguishing between large and small items, bounding the wasted free space by a fraction of items' area, and recursively packing items of given sizes.

These results were obtained thanks to novel techniques that may impact other similar problems, as well as different variants. While for a few specific studied problems there is little room for improvement, such as the Squared Metric FLP, for others, more tight

analysis and different techniques may lead to improved results. Some algorithms may be extended to tackle variants of the considered problems, or problems with similar combinatorial structures. Next, we describe possible extensions of the results, and some problems left open by this thesis.

- Although the upper factor-revealing programs introduced in Chapter 2 were used to analyze primal-dual algorithms for the SMFLP, they are described as a generic framework, and so may also be applied to other variants and problems analyzed through factor-revealing programs.
- While there is a PTAS for the Euclidean FLP by Arora *et al.* [7], there are no specialized algorithms for the Squared Euclidean FLP, and the best known factor is 2.04 for the SMFLP. We left open whether the Squared Euclidean FLP has an approximation factor better than 2.04.
- The approximate center sets used in Chapter 3 may also be used to reduce several other variants of the FLP to their corresponding discrete versions, such as a continuous version of the Soft Capacitated FLP, when each facility has a given capacity, but several copies may be opened [75].
- While Chapter 4 gives a 2.77-approximation for the PDP, the only known lower bound on the approximation factor is 1.463, implicitly obtained from the hardness of the FLP. It is left to future work either improving the lower bound, or obtaining tighter algorithms.
- Although logarithmic factors for the Submodular JRP can be readily obtained from a set covering formulation, and there is a constant approximation for the particular case of the Multilevel JRP, it is not known whether the general problem admits a constant approximation.
- The novel techniques of Chapter 5 are not specific to circles, and may be extended to other related problems. Indeed, an APTAS may also be given for the packing of more general items, such as d-dimensional L_p-norm spheres, for $p \geq 1$. Packing circles into bins of different shapes can also be tackled, such as the packing circles into circles.
- The problem of packing a set of large circles was reduced to solving a polynomial system of inequalities. Although all coefficients in such system are rational, there may be solutions with irrational values. It is worth noticing that resource augmentation is used due to the need of obtaining rational approximate solutions, and so it is left open the question whether these polynomial systems always have rational solutions.

Bibliography

[1] Marcel R. Ackermann, Johannes Blömer, and Christian Sohler. Clustering for metric and non-metric distance measures. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 799–808, 2008.

[2] Sang Ahn, Colin Cooper, Gérard Cornuéjols, and Alan Frieze. Probabilistic Analysis of a Relaxation for the k-Median Problem. *Mathematics of Operations Research*, 13(1):1–31, 1988.

[3] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, 2009.

[4] Matthew Andrews and Lisa Zhang. The Access Network Design Problem. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 40–49, 1998.

[5] Sanjeev Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, pages 2–11, 1996.

[6] Sanjeev Arora. Nearly linear time approximation schemes for Euclidean TSP and other geometric problems. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 554–563, 1997.

[7] Sanjeev Arora, Prabhakar Raghavan, and Satish Rao. Approximation schemes for Euclidean k-medians and related problems. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 106–113, 1998.

[8] David Arthur and Sergei Vassilvitskii. Worst-case and Smoothed Analysis of the ICP Algorithm, with an Application to the k-means Method. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 153–164, 2006.

[9] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[10] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pages 21–29, 2001.

[11] Mihai Bādoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via coresets. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002.

[12] M. L. Balinski. On finding integer solutions to linear programs. In *Proceedings of IBM Scientific Computing Symposium on Combinatorial Problems*, pages 225–248, 1966.

[13] Nikhil Bansal, Alberto Caprara, and Maxim Sviridenko. A New Approximation Method for Set Covering Problems, with Applications to Multidimensional Bin Packing. *SIAM Journal on Computing*, 39(4):1256–1278, 2010.

[14] Nikhil Bansal, José R. Correa, Claire Kenyon, and Maxim Sviridenko. Bin Packing in Multiple Dimensions: Inapproximability Results and Approximation Schemes. *Mathematics of Operations Research*, 31(1):31–49, 2006.

[15] Nikhil Bansal, Xin Han, Kazuo Iwama, Maxim Sviridenko, and Guochuan Zhang. A Harmonic Algorithm for the 3D Strip Packing Problem. *SIAM Journal on Computing*, 42(2):579–592, 2013.

[16] Nikhil Bansal and Arindam Khan. Improved Approximation Algorithm for Two-Dimensional Bin Packing. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 13–25, 2014.

[17] Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. *Journal of Algorithms*, 2(2):198–203, 1981.

[18] Saugata Basu, Richard Pollack, and Marie-Françoise Coste-Roy. *Algorithms in Real Algebraic Geometry (Algorithms and Computation in Mathematics)*. Springer, Secaucus, NJ, USA, 2006.

[19] Saugata Basu, Richard Pollack, and Marie-Françoise Roy. On the Combinatorial and Algebraic Complexity of Quantifier Elimination. *Journal of the ACM*, 43(6):1002–1045, 1996.

[20] Michael A. Bender and Chandra Chekuri. Performance guarantees for the TSP with a parameterized triangle inequality . *Information Processing Letters*, 73(1–2):17–21, 2000.

[21] Marcin Bienkowski, Jaroslaw Byrka, Marek Chrobak, Neil Dobbs, Tomasz Nowicki, Maxim Sviridenko, Grzegorz Swirszcz, and Neal E. Young. Approximation Algorithms for the Joint Replenishment Problem with Deadlines. In *Automata, Languages, and Programming*, pages 135–147, 2013.

[22] Marcin Bienkowski, Jaroslaw Byrka, Marek Chrobak, Łukasz Jeż, Dorian Nogneng, and Jiří Sgall. Better Approximation Bounds for the Joint Replenishment Problem. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 42–54, 2014.

[23] Ernesto G. Birgin and Jan M. Gentil. New and improved results for packing identical unitary radius circles within triangles, rectangles and strips. *Computers & Operations Research*, 37(7):1318–1327, 2010.

[24] Mourad Boudia and Christian Prins. A memetic algorithm with dynamic population management for an integrated production–distribution problem. *European Journal of Operational Research*, 195(3):703–715, 2009.

[25] Niv Buchbinder, Tracy Kimbrelt, Retsef Levi, Konstantin Makarychev, and Maxim Sviridenko. Online make-to-order joint replenishment model: primal dual competitive algorithms. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 952–961, 2008.

[26] Jaroslaw Byrka. An Optimal Bifactor Approximation Algorithm for the Metric Uncapacitated Facility Location Problem. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 29–43, 2007.

[27] Jaroslaw Byrka and Karen Aardal. The approximation gap for the metric facility location problem is not yet closed. *Operations Research Letters*, 35(3):379–384, 2007.

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

[29] Jaroslaw Byrka, MohammadReza Ghodsi, and Aravind Srinivasan. LP-rounding algorithms for facility-location problems, 2010. Available at `http://arxiv.org/abs/1007.3611`.

[30] Jaroslaw Byrka and Bartosz Rybicki. Improved LP-Rounding Approximation Algorithm for k-level Uncapacitated Facility Location. In *Automata, Languages, and Programming*, pages 157–169, 2012.

[31] Alberto Caprara. Packing 2-dimensional bins in harmony. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 490–499, 2002.

[32] Felix T. S. Chan, S. H. Chung, and Subhash Wadhwa. A hybrid genetic algorithm for production and distribution. *Omega*, 33(4):345–355, 2005.

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

[34] Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 1–10, 1999.

[35] Ke Chen. On k-Median clustering in high dimensions. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1177–1185, 2006.

[36] Fabián A. Chudak. Improved Approximation Algorithms for Uncapacitated Facility Location. In *Integer Programming and Combinatorial Optimization*, pages 180–194, 1998.

[37] Fabián A. Chudak and David B. Shmoys. Improved Approximation Algorithms for a Capacitated Facility Location Problem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 875–876, 1999.

[38] Fabián A. Chudak and David B. Shmoys. Improved Approximation Algorithms for the Uncapacitated Facility Location Problem. *SIAM Journal on Computing*, 33(1):1–25, 2004.

[39] F. Chung, M. Garey, and D. Johnson. On Packing Two-Dimensional Bins. *SIAM Journal on Algebraic Discrete Methods*, 3(1):66–76, 1982.

[40] Edward G. Coffman Jr., János Csirik, Gábor Galambos, Silvano Martello, and Daniele Vigo. Bin Packing Approximation Algorithms: Survey and Classification. In Panos M. Pardalos, Ding-Zhu Du, and Ronald L. Graham, editors, *Handbook of Combinatorial Optimization*, pages 455–531. Springer, 2013.

[41] George E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In *Automata Theory and Formal Languages*, pages 134–183, 1975.

[42] Wallace B. Crowston and Michael H. Wagner. Dynamic Lot Size Models for Multi-Stage Assembly Systems. *Management Science*, 20(1):14–21, 1973.

[43] Artur Czumaj, Christiane Lammersen, Morteza Monemizadeh, and Christian Sohler. $(1+\varepsilon)$-Approximation for Facility Location in Data Streams. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, page 1710, 2013.

[44] Sanjoy Dasgupta. The hardness of k-means clustering. Technical Report CS2008-0916, Department of Computer Science and Engineering, University of California, 2008.

[45] Mark S. Daskin, Collette R. Coullard, and Zuo-Jun Max Shen. An Inventory-Location Model: Formulation, Solution Algorithm and Computational Results. *Annals of Operations Research*, 110(1-4):83–106, 2002.

[46] Mark de Berg, Fred van Nijnatten, Rene Sitters, Gerhard J. Woeginger, and Alexander Wolff. The Traveling Salesman Problem Under Squared Euclidean Distances. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science*, pages 239–250, 2010.

[47] Erik D. Demaine, Sándor P. Fekete, and Robert J. Lang. Circle Packing for Origami Design Is Hard. In *Proceedings of the 5th International Conference on Origami in Science*, pages 609–626, 2010.

[48] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal Voronoi Tessellations: Applications and Algorithms. *SIAM Review*, 41(4):637–676, 1999.

[49] Jack Edmonds. Paths, trees, and flowers. *Journal Canadien de Mathématiques*, 17:449–467, 1965.

[50] Donald Erlenkotter. A Dual-Based Procedure for Uncapacitated Facility Location. *Operations Research*, 26(6):992–1009, 1978.

[51] Uriel Feige. A threshold of $\ln n$ for approximating set cover (preliminary version). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 314–318, 1996.

[52] Dan Feldman, Morteza Monemizadeh, and Christian Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the Twenty-Third Annual Symposium on Computational Geometry*, pages 11–18, 2007.

[53] Cristina G. Fernandes, Luis A. A. Meira, Flávio K. Miyazawa, and Lehilton L. C. Pedrosa. A Systematic Approach to Bound Factor Revealing LPs and Its Application to the Metric and Squared Metric Facility Location Problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 146–157, 2012.

[54] Wenceslas Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation Schemes for Clustering Problems. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, pages 50–58, 2003.

[55] Wenceslas Fernandez de la Vega and George S. Lueker. Bin packing can be solved within $1 + \varepsilon$ in linear time. *Combinatorica*, 1(4):349–355, 1981.

[56] Stefan Funke, Sören Laue, Rouven Naujoks, and Zvi Lotker. Power Assignment Problems in Wireless Communication: Covering Points by Disks, Reaching few Receivers Quickly, and Energy-Efficient Travelling Salesman Tours. In *Distributed Computing in Sensor Systems*, pages 282–295, 2008.

[57] John A. George, Jennifer M. George, and Bruce W. Lamar. Packing different-sized circles into a rectangular container. *European Journal of Operational Research*, 84(3):693–712, 1995.

[58] Michel X. Goemans, Andrew V. Goldberg, Serge Plotkin, David B. Shmoys, Éva Tardos, and David P. Williamson. Improved Approximation Algorithms for Network Design Problems. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 223–232, 1994.

[59] Michel X. Goemans and David P. Williamson. A General Approximation Technique for Constrained Forest Problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[60] D. Yu. Grigor'ev and Nikolaj N. Vorobjov Jr. Solving systems of polynomial inequalities in subexponential time. *Journal of Symbolic Computation*, 5(1–2):37–64, 1988.

[61] Sudipto Guha and Samir Khuller. Greedy strikes back: improved facility location algorithms. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 649–657, 1998.

[62] Sudipto Guha and Samir Khuller. Greedy strikes back: improved facility location algorithms. *Journal of Algorithms*, 31(1):228–248, 1999.

[63] Sudipto Guha, Adam Meyerson, and Kamesh Munagala. Hierarchical placement and network design problems. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 603–612, 2000.

[64] Venkatesan Guruswami and Piotr Indyk. Embeddings and non-approximability of geometric problems. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 537–538, 2003.

[65] Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. In *Proceedings of the Twenty-First Annual Symposium on Computational geometry*, pages 126–134, 2005.

[66] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing*, pages 291–300, 2004.

[67] Sariel Har-Peled and Bardia Sadri. How fast is the k-means method? In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 877–885, 2005.

[68] Susumu Hasegawa, Hiroshi Imai, Mary Inaba, Naoki Katoh, and Jun Nakano. Efficient algorithms for variance-based k-clustering. In *Proceedings of the First Pacific Conference on Computer Graphics and Applications*, pages 75–89, 1993.

[69] Mhand Hifi and Rym M'Hallah. A Literature Review on Circle and Sphere Packing Problems: Models and Methodologies. *Advances in Operations Research*, 2009:1–22, 2009.

[70] Dorit S. Hochbaum. Approximation Algorithms for the Set Covering and Vertex Cover Problems. *SIAM Journal on Computing*, 11(3):555–556, 1982.

[71] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted Voronoi diagrams and randomization to variance-based k-clustering (extended abstract). In *Proceedings of the Tenth Annual Symposium on Computational Geometry*, pages 332–339, 1994.

[72] Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *Journal of the ACM*, 50(6):795–824, 2003.

[73] 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, 2002.

[74] Kamal Jain, Ion Măndoiu, Vijay V. Vazirani, and David P. Williamson. A primal-dual schema based approximation algorithm for the element connectivity problem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 484–489, 1999.

[75] Kamal Jain and Vijay V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 2–13, 1999.

[76] Kamal Jain and Vijay V. Vazirani. Approximation algorithms for metric facility location and k-Median problems using the primal-dual schema and Lagrangian relaxation. *Journal of the ACM*, 48(2):274–296, 2001.

[77] Sugih Jamin, Cheng Jin, Yixin Jin, Danny Raz, Yuval Shavitt, and Lixia Zhang. On the placement of Internet instrumentation. In *Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 295–304, 2000.

[78] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. A local search approximation algorithm for k-means clustering. In *Proceedings of the Eighteenth Annual Symposium on Computational Geometry*, pages 10–18, 2002.

[79] O. Kariv and S. Hakimi. An Algorithmic Approach to Network Location Problems. II: The p-Medians. *SIAM Journal on Applied Mathematics*, 37(3):539–560, 1979.

[80] Claire Kenyon and Eric Rémila. A Near-Optimal Solution to a Two-Dimensional Cutting Stock Problem. *Mathematics of Operations Research*, 25(4):645–656, 2000.

[81] Moutaz Khouja and Suresh Goyal. A review of the joint replenishment problem literature: 1989–2005. *European Journal of Operational Research*, 186(1):1–16, 2008.

[82] Yoshiharu Kohayakawa, Flávio K. Miyazawa, Prabhakar Raghavan, and Yoshiko Wakabayashi. Multidimensional Cube Packing. *Algorithmica*, 40(3):173–187, 2004.

[83] Stavros G. Kolliopoulos and Satish Rao. A Nearly Linear-Time Approximation Scheme for the Euclidean k-Median Problem. *SIAM Journal on Computing*, 37(3):757–782, 2007.

[84] Madhukar R. Korupolu, C. Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–10, 1998.

[85] Alfred A. Kuehn and Michael J. Hamburger. A Heuristic Program for Locating Warehouses. *Management Science*, 9(4):643–666, 1963.

[86] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time $(1 + \varepsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 454–462, 2004.

[87] Amit Kumar, Yogish Sabharwal, and Sandeep Sen. Linear-time Approximation Schemes for Clustering Problems in Any Dimensions. *Journal of the ACM*, 57(2):5:1–5:32, 2010.

[88] H. W. Lenstra Jr. Integer Programming with a Fixed Number of Variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

[89] Retsef Levi, Andrea Lodi, and Maxim Sviridenko. Approximation Algorithms for the Multi-item Capacitated Lot-Sizing Problem Via Flow-Cover Inequalities. In *Integer Programming and Combinatorial Optimization*, pages 454–468, 2007.

[90] Retsef Levi, Robin O. Roundy, and David B. Shmoys. Primal-Dual Algorithms for Deterministic Inventory Problems. *Mathematics of Operations Research*, 31(2):267–284, 2006.

[91] Retsef Levi, Robin O. Roundy, David B. Shmoys, and Maxim Sviridenko. A Constant Approximation Algorithm for the One-Warehouse Multiretailer Problem. *Management Science*, 54(4):763–776, 2008.

[92] Bo Li, Mordecai J. Golin, Giuseppe F. Italiano, Xin Deng, and Kazem Sohraby. On the optimal placement of web proxies in the Internet. In *Proceedings of INFOCOM 1999 Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1282–1290, 1999.

[93] Shi Li. A 1.488 Approximation Algorithm for the Uncapacitated Facility Location Problem. In *Automata, Languages, and Programming*, pages 77–88, 2011.

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

[95] Shi Li and Ola Svensson. Approximating K-median via Pseudo-approximation. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, pages 901–910, 2013.

[96] Yu Li, Jia Shu, Xi Wang, Naihua Xiu, Dachuan Xu, and Jiawei Zhang. Approximation Algorithms for Integrated Distribution Network Design Problems. *INFORMS Journal on Computing*, 25(3):572–584, 2013.

[97] Jyh-Han Lin and Jeffrey Scott Vitter. ε-approximations with minimum packing constraint violation (extended abstract). In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, pages 771–782, 1992.

[98] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[99] J. B. MacQueen. Some Methods for Classification and Analysis of MultiVariate Observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[100] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The Planar k-Means Problem is NP-Hard. In *Algorithms and Computation*, pages 274–285, 2009.

[101] Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V. Vazirani. A Greedy Facility Location Algorithm Analyzed Using Dual Fitting. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 127–137, 2001.

[102] Mohammad Mahdian and Qiqi Yan. Online Bipartite Matching with Random Arrivals: An Approach Based on Strongly Factor-revealing LPs. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, pages 597–606, 2011.

[103] Mohammad Mahdian, Yinyu Ye, and Jiawei Zhang. Improved Approximation Algorithms for Metric Facility Location Problems. In *Approximation Algorithms for Combinatorial Optimization*, pages 229–242, 2002.

[104] Mohammad Mahdian, Yinyu Ye, and Jiawei Zhang. Approximation Algorithms for Metric Facility Location Problems. *SIAM Journal on Computing*, 36(2):411–432, 2006.

[105] Alan S. Manne. Plant Location under Economies-of-Scale-Decentralization and Computation. *Management Science*, 11(2):213–235, 1964.

[106] Jirí Matoušek. On Approximate Geometric k -Clustering. *Discrete & Computational Geometry*, 24(1):61–84, 2000.

[107] A. Meir and L. Moser. On packing of squares and cubes. *Journal of Combinatorial Theory*, 5(2):126–134, 1968.

[108] Luis A. A. Meira and Flávio K. Miyazawa. A Continuous Facility Location Problem and Its Application to a Clustering Problem. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, pages 1826–1831, 2008.

[109] M. Teresa Melo, Stefan Nickel, and Francisco Saldanha-da-Gama. Facility location and supply chain management – A review. *European Journal of Operational Research*, 196(2):401–412, 2009.

[110] Flávio K. Miyazawa and Yoshiko Wakabayashi. Approximation Algorithms for the Orthogonal Z-Oriented Three-Dimensional Packing Problem. *SIAM Journal on Computing*, 29(3):1008–1029, 2000.

[111] Flávio K. Miyazawa, Lehilton L. C. Pedrosa, Rafael C. S. Schouery, Maxim Sviridenko, and Yoshiko Wakabayashi. Polynomial-Time Approximation Schemes for Circle Packing Problems. In *Algorithms*, 2014. To appear.

[112] Dana Moshkovitz. The Projection Games Conjecture and the NP-Hardness of $\ln n$-Approximating Set-Cover. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 276–287, 2012.

[113] Tim Nonner and Maxim Sviridenko. An Efficient Polynomial-Time Approximation Scheme for the Joint Replenishment Problem. In *Integer Programming and Combinatorial Optimization*, pages 314–323, 2013.

[114] Rafail Ostrovsky, Yuval Rabani, Leonard J. Schulman, and Chaitanya Swamy. The Effectiveness of Lloyd-Type Methods for the k-Means Problem. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 165–176, 2006.

[115] Lehilton L. C. Pedrosa and Maxim Sviridenko. Integrated Supply Chain Management via Randomized Rounding. In *Theoretical Informatics*, pages 562–573, 2014.

[116] Yves Pochet and Laurence A. Wolsey. *Production planning by mixed integer programming*. Springer, New York, Berlin, 2006.

[117] Lili Qiu, Venkata N. Padmanabhan, and Geoffrey M. Voelker. On the placement of Web server replicas. In *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1587–1596, 2001.

[118] Sridhar Rajagopalan and Vijay V. Vazirani. On the bidirected cut relaxation for the metric Steiner tree problem. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 742–751, 1999.

[119] Ramamoorthi Ravi and David P. Williamson. An approximation algorithm for minimum-cost vertex-connectivity problems. *Algorithmica*, 18(1):21–43, 1997.

[120] Zuo-Jun Max Shen. Integrated Stochastic Supply-Chain Design Models. *Computing in Science & Engineering*, 9(2):50–59, 2007.

[121] Zuo-Jun Max Shen, Collette R. Coullard, and Mark S. Daskin. A Joint Location-Inventory Model. *Transportation Science*, 37(1):40–55, 2003.

[122] David B. Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 265–274, 1997.

[123] Jia Shu, Chung-Piaw Teo, and Zuo-Jun Max Shen. Stochastic Transportation-Inventory Network Design Problem. *Operations Research*, 53(1):48–60, 2005.

[124] Gautier Stauffer, Guillaume Massonnet, Christophe Rapine, and Jean-Philippe Gayon. A simple and fast 2-approximation algorithm for the one-warehouse multi-retailers problem. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 67–79, 2011.

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

[126] John F. Stollsteimer. A Working Model for Plant Numbers and Locations. *Journal of Farm Economics*, 45(3):631–645, 1963.

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

[128] P. G. Szabó, M. C. Markót, T. Csendes, E. Specht, L. G. Casado, and I. García. *New Approaches to Circle Packing in a Square.* Springer, New York, 2007.

[129] Alfred Tarski. *A decision method for elementary algebra and geometry.* University of California Press, 1951. 2nd ed.

[130] Chung-Piaw Teo and Jia Shu. Warehouse-Retailer Network Design Problem. *Operations Research*, 52(3):396–408, 2004.

[131] Jens Vygen. Approximation algorithms for facility location problems (Lecture Notes). Technical Report 05950-OR, Research Institute for Discrete Mathematics, University of Bonn, 2005.

[132] David P. Williamson, Michel X. Goemans, Milena Mihail, and Vijay V. Vazirani. A primal-dual approximation algorithm for generalized Steiner network problems. *Combinatorica*, 15(3):435–454, 1995.