Parametric and Interdiction Variants of Center and Median Problems on Networks

Lena Leiß

Vom Fachbereich Mathematik der
Rheinland-Pfälzischen Technischen Universität Kaiserslautern-Landau
zur Verleihung des akademischen Grades
Doktor der Naturwissenschaften (Doctor rerum naturalium)
genehmigte Dissertation

Gutachter
Prof. Dr. Stefan Ruzika, RPTU in Kaiserslautern
Prof. Dr. Kathrin Klamroth, Bergische Universität Wuppertal

Datum der Disputation: 30. Januar 2025

DE-386

RPTU Rheinland-Pfälzische Technische Universität Kaiserslautern Landau

Acknowledgements

I cannot thank my supervisor Prof. Dr. Stefan Ruzika enough for the opportunities he has given me during my time in the optimization research group. His continuous support, critical feedback, and encouragement – both in mathematical aspects and challenges beyond research – have guided me to this point.

Furthermore, I would like to express my sincere gratitude to Prof. Dr. Kathrin Klamroth for her willingness and effort in serving as a co-referee for this thesis.

I would also like to thank Prof. Dr. Horst Hamacher, who hired me after my studies. Furthermore, my gratitude goes to Prof. Dr. Sven Krumke, Dr. Oliver Bachtler, and Dr. Manuel Streicher, who were always there to listen and help when I was stuck.

I am also grateful to my former and current colleagues, who made my time in the research group more than enjoyable and from whom some valuable friendships have emerged.

Finally, a special thanks goes to my family, who have always supported me throughout my life.

Abstract

Facility location planning problems are a well-studied topic in optimization – and for good reason. Their applications appear in lots of different areas of real life. In this thesis, we study location problems in the context of parametric networks, where the parameter is introduced on the weights of the edges. Furthermore, interdiction location problems are analyzed, that include an opposing force – the interdictor – with the goal to destroy the existing network by removing edges of the network to make the situation as unfavorable as possible for the locator.

In the first part of the work, we investigate location problems in the context of parametric optimization. Here, we assume that the edge weights in the considered networks are linearly dependent on a parameter λ. Parametric optimization has been studied in relation to many other classical optimization problems; however, so far, there have been no results on location problems in networks with parametric edge weights. For this purpose, we analyze the classical 1-median and 1-center problems on general graphs and also on trees. In doing so, we examine the complexity of both the solution itself and the objective function value as a function of the parameter. We emphasize that the complexity of the optimal value function can be super-polynomial on directed general graphs for both median and center problems, which motivates the development of approximation schemes for these cases. In fact, approximation methods that rely on an existing approximation scheme are introduced for both location problems on directed and undirected general graphs. The developed method is based on the fact that both median and center problems can be divided into subproblems, that get approximated individually before combining the gained information into an overall approximate solution without losing the initial approximation guarantee. Furthermore, exact algorithms are given for the two considered parametric location problems on both directed and undirected trees. These algorithms exploit the properties of trees – precisely the uniqueness of the shortest paths in said structures.

In the second part of the thesis, we introduce the p-median interdiction problem on trees. The goal of the interdictor is to remove edges of the underlying network to deteriorate the initial situation of the locator in order to worsen their objective function value. In this part, we first show that the p-median interdiction problem on trees is NP-complete by a reduction from the knapsack problem with bounded profit ratio of 2. Furthermore, we present algorithms for solving the problem on paths with unit and arbitrary lengths. Finally, an exact algorithm is presented for trees with unit edge weights and a single interdiction. Let e be the edge that is incident to the leaf that is nearest to the median location of the initial tree. We prove that removing this edge is the optimal interdiction strategy.

Zusammenfassung

Standortprobleme sind ein sehr gut untersuchtes Thema der Optimierung, denn ihre Anwendungen finden sich in vielen Bereichen des realen Lebens. In dieser Arbeit beschäftigen wir uns mit zwei Varianten von Standortproblemen auf Netzwerken. Im ersten Teil analysieren wir sie im Kontext parametrischer Optimierung, bei denen der Parameter in den Kantengewichten eingeführt wird. Weiterhin untersuchen wir Interdiction-Standortprobleme, bei denen ein gegnerischer Spieler – der Interdiktor – versucht, das Netzwerk durch das Entfernen von Kanten so zu zerstören, dass die Situation für den Standortplaner so ungünstig wie möglich ist.

Im ersten Teil untersuchen wir Standortprobleme auf Netzwerken, wo die Kantengewichte linear vom Parameter λ abhängen. Parametrische Optimierung wurde im Zusammenhang mit vielen klassischen Optimierungsproblemen untersucht; bisher sind aber keine Ergebnisse zu Standortproblemen auf Netzwerken mit parametrischen Kantengewichten bekannt. Daher analysieren wir die klassischen 1-Median und 1-Center-Probleme sowohl auf allgemeinen Graphen als auch auf Bäumen. Dabei untersuchen wir die Komplexität der Lösung selbst als auch den Zielfunktionswert als Funktion des Parameters λ. Wir zeigen, dass die Komplexität der optimalen Zielfunktion auf gerichteten allgemeinen Graphen für beide Standortprobleme superpolynomiell sein kann. Dies motiviert die Entwicklung von Approximationsmethoden für diese Fälle. Wir führen für beide Standortprobleme auf allgemeinen Graphen Approximationsmethoden ein, die auf bestehenden Approximationsschemata basieren. Die entwickelte Methode beruht darauf, dass sowohl Median als auch Center Probleme in Teilprobleme unterteilt werden können, die zunächst einzeln approximiert werden, bevor die gewonnenen Informationen zu einer approximativen Gesamtlösung kombiniert werden ohne die ursprüngliche Approximationsgüte zu verlieren. Weiterhin werden exakte Algorithmen für beide Standortprobleme auf Bäumen gegeben. Diese Algorithmen nutzen Eigenschaften von Bäumen aus – insbesondere die Eindeutigkeit der Wege. Im zweiten Teil der Arbeit führen wir das p-Median-Interdictionproblem auf Bäumen ein. Ziel des Interdiktors ist es, Kanten des zugrunde liegenden Netzwerks zu entfernen, um die Ausgangssituation des Standortplaners und somit dessen Zielfunktionswert zuverschlechtern. Wir zeigen zunächst, dass das p-Median-Interdictionproblem auf Bäumen NP-vollständig ist, indem wir eine Reduktion vom Knapsack Problem mit beschränkter profit ratio von 2 durchführen. Darüber hinaus präsentieren wir Algorithmen zur Lösung des Problems auf Pfaden. Abschließend wird ein exakter Algorithmus für Bäume mit einheitlichen Kantengewichten und einer einzelnen Interdiktion vorgestellt. Sei e die Kante, die inzident ist zu dem Blatt, welches dem ursprünglichen Median-Standort am nächsten liegt. Wir beweisen, dass das Entfernen dieser Kante die optimale Interdictionstrategie darstellt.

Contents

List of Algorithms

List of Figures

List of Tables

1 Introduction

Location planning is a field of mathematical research which crosses our daily life more often, than we might think at first sight. The root of modern location planning can be traced back to Pierre de Fermat and also Evangelista Torricelli and aims at finding the point, which minimizes the sum of the Euclidean distances of three given points to the new location – although it is not clear, who originated the problem [DH01, Chapter 1]. There is different literature on the history of location planning, for instance [LMW88, Wes93]. A popular, more applied version of this problem is the identification of a new location for a supplier of materials for further industrial processing which has been stated in [WP22] and is called Weber problem. In a more general approach, a new location is to be found which minimizes the sum of all - possibly weighted - distances from all given locations to the new one. This problem is called median location problem [LNG15]. A second problem that is well studied throughout the literature, is called center problem [LNG15]. Here, the locator aims at placing a new facility, such that the maximum distance to the other vertices is as small as possible.

Median as well as center problems have been extensively studied in many different variants and contexts. A first distinction is in the underlying structure, on which location problems can be analyzed. Although there have been approaches to generalize location problems and combine shared similarities [PK08], mainly, one distinguishes between planar location problems and network location problems. A further alteration of the main problem comes with the number of new locations to be computed. One refers to the problem of placing p new facilities as the p-median location problem (p-center location problem, respectively). Since the semial works of Hakimi [Hak64, Hak65] and Kariv and Hakimi [KH79a, KH79b] have studied the p-median problem as well as the p-center problem, several other publications can be found in the context of these problems on planes and networks. A good general overview on location planning is available in [DH01, HM03, LNG15, TFL83a, TFL83b, FMW83, RE05, Das11], for instance, and also in [BF12] with focus on dynamic versions of the location problems. There are more variants of location problems to be found in the broad literature on that topic. We mention hub-location planning, where a hub is a service center to which demand nodes are assigned. Surveys on this topic are for instance [AK08, Far+13]. Literature on facility location planning under uncertainty can be found in [Sny06]. Here, location problems are considered, that are faced with uncertainty in cost or demand, for instance. An overview on location problems with multiple criteria can be found in [FSA10].

There are several applications known and studied for location problems. Daskin and Dean [DD04] for instance deal with the subject of location planning in the context of health care

management. Here, advantages of a "good" planning cannot only be recognized from a financial point of view, but it has also a direct impact on the health of the customers of these facilities. Further application can be found when analyzing methods of waste management [AO20, Eis07]. The survey of Melo, Nickel, and Saldanha-da-Gama [MNS09] gives an extensive overview over literature for optimizing supply chain management. In [CE20] we are given an overview on literature since 2000, that has been published in the general field of service location planning. Farahani et al. [Far+19] also provides an overview on the literature on service facility location planning.

In this thesis, we combine the standard median and center location problems with parametric optimization in the first part and interdiction problems in the second. In the following, we motivate this fields, state our contributions for each part and hereby present the general structure of this work. First, **Chapter 1** gives an overview on the definitions and basic models that are used throughout the thesis. Details on the specific subjects can be found in the literature, that is listed there. We now go into more detail on what content can be found in the two following parts.

Part I: Parametric Location Problems

Part I deals with location problems on parametric networks. With the help of parametric optimization real-world problems can be modeled where data might change over time. In this thesis, the parameter is introduced on the edge weights of the network. Thus, they are linearly dependent on a real-valued parameter λ instead of being equipped with fixed values. Parametric optimization in general has been of interest in the last decades and numerous literature has been published in that field, see [Nem+25]. As of now, parametric variants of several classic network optimization problems have been analyzed, such as the shortest path problem, the assignment problem, the knapsack problem or the minimum spanning tree problem [MS00, Car83b, HK17, FSE96].

In this work, vertex restricted location planning problems are investigated on parametric networks. That is, the optimal solution(s) are sought on the vertex set of the given network. However, the introduction of a parameter on the edges might lead to networks, where optimal locations switch as λ varies within a given parameter set. Thus, the solution set of a parametric location problem consists of an optimal solution together with its critical region, that is the set of λ for which the solution is optimal. Furthermore, we are interested in the graph that maps an optimal solution to the corresponding objective function value for a fixed value of λ, the optimal value function.

These considerations lead to different questions that arise when dealing with parametric problems. First, the structure of the critical region is of interest. Also, the analysis of the optimal value function provides useful information on the problem. If we compute for every feasible solution of the parametric location problem their corresponding optimal value function, such that for every λ of the parameter set, it is known what the corresponding objective value is, the resulting functions are known to be piecewise linear and concave for minimization problems [Gus83]. Then, the solution of the parametric location problem can be interpreted as the lower envelope over these piecewise linear functions dependent on λ. The number of changes in its

slope, the breakpoints, however, and therefore also the number of linear segments on the lower envelope gives hints to the computational effort to output the solution set together with the optimal value function. First, the number of line segments can get super-polynomially large for different problems [MS00, Car83a]. Furthermore, although the optimal solution stays the same for one line segment or even consecutive segments, the optimal solutions can still alternate. Consequently, when desiring to output all solutions together with their optimal value, there are cases where this means that the output can get super-polynomially large. This has led to the idea of not only searching for exact algorithms, but to also consider approximation schemes for parametric optimization problems.

Our contribution. In the first part, we deal with 1-center and 1-median problems on both directed and undirected general graphs and directed and undirected trees.

In Chapter 4, the 1-center problem is analyzed on the four stated graph classes. For all four problems, we are able to either provide an exact algorithm in the case of trees or an approximation method for the problem on general graphs. Here, we use an existing method, see [Baz+22] that allows to approximate 1-parametric optimization problems. However, we cannot directly apply the provided algorithm to the problem. Rather, we split it into subproblems that we approximate and then build the approximate solution for the center problem of the information gained thereof. We prove that the approximation guarantee of the given approximation scheme carries over to the stated method to approximate the center problem.

Furthermore, for directed general graphs the complexity of the solution and of the optimal value function are investigated. To do so, we construct a graph based on the one introduced in [MS00] that leads to super-polynomially many breakpoints on the optimal value function for the parametric shortest path problem. For the construction we first have to prove how to bound the distances in the parametric graph from above. Then, for our construction we can use the properties of the graph in [MS00] to show, that the optimal value function of the 1-center problem can have super-polynomially many breakpoints as well. Furthermore, we show that the solution itself can switch as often.

For both directed and undirected trees the complexity of the solution and of the optimal value function are investigated as well. Here, we are able to use properties of shortest paths on trees to prove that the complexity of the 1-center problem in terms of solution and optimal value function is polynomially bounded.

Chapter 5 gives the results for the 1-median version on directed and undirected general graphs and trees, respectively. We provide exact algorithms for the parametric median problem on trees. Furthermore, for the problem on general graphs, we introduce a scheme that solves the problem approximately. We use a similar technique as already mentioned for the center problem. Subdividing the median problem in subproblems and approximating them with the method of Bazgan et al. [Baz+22] allows to combine the retrieved information to solve the median problem approximately. Again, we prove that the approximation guarantee for the approximation of the subproblems also holds true for the approximate solution of the median problem itself.

For directed general graphs we are able to prove, that the number of breakpoints of the optimal function value can be super-polynomially large. Again, for the proof we use the graph introduced in [MS00] but incorporate it into two different graphs for which we show that breakpoints of the

graph of [MS00] occur in at least one of the two graphs.
Lastly, for both tree classes we are able to show, that the complexity of the optimal value function and also the solution is polynomially bounded. Here, again, properties of trees and their shortest paths are exploited.

Part II: Interdiction Location Problems

Part II tackles the problem of interdiction location problems. Here, we introduce the interdictor that is a second player and aims to worsen the objective function value of the locator. To achieve this, the interdictor is allowed to remove one or more edges of a given network. The locator then places their location on the remaining part of the original network. In general, the interdiction can not only be done on the edges of the network, but also on its vertices. Interdiction problems in general have various applications in real life. The initial thought might be that interdiction can only be used for negative purposes, as in the context of destruction, where solving interdiction problems would help terrorists, for instance, to identify crucial targets such as road sections, supply stations or parts of critical infrastructure, for instance. On the other hand, as their opponent you could also use this analysis to retrieve information on which supply facilities or routes need special fortification [LSD11]. Furthermore, authorities could identify how to interrupt supply chains of criminals, for instance [MPS07]. Therefore, solving interdiction problems can also help to find vulnerable parts of a given system.

Interdiction problems gained a lot of interest, not only because of their various applications for real life scenarios. The combination of interdiction and network optimization problems include the shortest path problem or the maximum flow problem [IW02, Sch+20], where a lot of research is done. Other classic network optimization problems that have been studied in the context of interdiction are the matching problem [Zen10] or the knapsack problem [Fis+19], for instance. There is some literature on interdiction location problems, for instance where the interdictor can destroy multiple already existing facilities to worsen the median objective [CSM04]. However, literature on interdiction location problems, where the interdictor removes edges of a network, is limited. In this context, we mention [Frö21], where the authors prove that the p-median interdiction problem is Σ_2^p-complete in the general case. Therefore, we aim at providing more understanding of the p-median interdiction problem in other cases.

Our contribution. We investigate on the p-median location interdiction problem. In Section 6.2 we prove this problem to be NP-complete on trees. This is done by reduction from the knapsack problem with bounded profit ratio of 2 (K-$BPR2$). In order to use this reduction, we first prove the K-$BPR2$ to be NP-complete as well by a reduction from the equal partition problem. In Section 6.3, we investigate the problem on paths. For both cases of unit edge weights and arbitrary lengths, we show how to solve the problem.
In 6.4, we deal with the interdiction median problem on trees with unit edge weights. Here, we are able to provide an exact algorithm for the case where the interdictor is allowed to remove one edge of the tree. We show, that removing the edge connecting the nearest leaf to the median location in the undestructed tree leads to an optimal interdiction strategy. This part is joint work with Till Heller, Luca Schäfer and Manuel Streicher and can be found in [Lei+23].

We conclude the thesis by an outlook and ideas for further research.

2 Preliminaries

In this chapter, we give an overview over the basic definitions, notations and concepts used throughout this thesis. At the beginning of each section, we present further literature on the respective topic. Still, we assume the reader to be familiar with the basic concepts of mathematical optimization. We suggest [WN99, BT97], for instance, for an extensive overview on optimization problems.

2.1 Basic Notation

We denote by $\mathbb{N} := \{1, 2, 3, \dots\}$ the set of natural numbers without 0. By $\mathbb{N}_0 := \{0, 1, 2, \dots\}$, we denote the natural numbers together with 0. Furthermore, by $\mathbb{Z}, \mathbb{Q}$ and $\mathbb{R}$ we denote the set of integral, rational and real numbers, respectively. The empty set is denoted by $\varnothing$.

For two sets A and B, the union, intersection and the difference of the sets are denoted by $A \cup B, A \cap B$ and $A \setminus B$, respectively. Also, we write $A \subseteq B$, if A is a subset of set B.

2.2 Graph Theory

In this thesis, we analyze different optimization problems on graph structures. The following introduction on this topic is mostly based on [KN09]. For a deeper understanding, the reader is referred to this book.

Definition 2.1 — Directed Graph. A *directed graph* is a 4-tuple $G = (V, E, \alpha, \omega)$ with the following properties:

1. $V \neq \varnothing$ is a set, the set of *nodes* or *vertices* of G with $|V| = n$
2. E is a set, the set of *arcs* or *edges* of G with $|E| = m$
3. It is $V \cap E = \varnothing$
4. $\alpha \colon E \to V$ and $\omega \colon E \to V$ are maps, where $\alpha(e)$ is the *start vertex* of the arc $e \in E$ and $\omega(e)$ is the *end vertex* of e

If the direction of the edges is not important to the setup of the model, we use the concept of an undirected graph.

Definition 2.2 — Undirected Graph. An *undirected graph* is a triple $G = (V, E, \beta)$ consisting of a nonempty set V, a set E with $V \cap E = \varnothing$ and a mapping

$$\beta \colon E \to \{Y : Y \subseteq V, 1 \leq |Y| \leq 2\}.$$

Remark 2.3 For better readability, instead of $|V(G)|$, we may write $|G|$.

Definition 2.4 — Incident, adjacent. Let $v_1, v_2 \in V, e_1 \in E$ for a graph $G = (V, E)$. Then, v_1 is incident to e if it is a start or end vertex of e, thus $v \in \{\alpha(e), \omega(e)\}$. The vertices v_1 and v_2 are adjacent, if there is an edge that is incident to v_1 and v_2.

Sometimes, it is important to know the number of incident edges of a vertex.

Definition 2.5 — Degree, in-degree, out-degree. For a given vertex v in a graph $G = (V, E)$, we denote the number of incident edges, i.e. its *degree*, by $\deg(v)$. We differentiate between the *in-degree* $\deg^-(v) := |\delta^-(v)|$ with $\delta^- := \{e \in E \colon \omega(e) = v\}$ and the *out-degree* $\deg^+(v) := |\delta^+(v)|$ with $\delta^+(v) := \{e \in E \colon \alpha(e) = v\}$.

For both directed and undirected graphs, we define *paths* connecting two vertices.

Definition 2.6 — Path. A v_0-v_k-path in a directed graph $G = (V, E)$ is a sequence

$$P = (v_0, e_1, v_1, \ldots, e_k, v_k)$$

with $k \geq 0$ such that $v_0, \ldots, v_k \in V$ and $e_1, \ldots, e_k \in E$ with $\alpha(e_i) = v_{i-1}$ and $\omega(e_i) = v_i$ for all $i = 1, \ldots, k$. Let $\mathcal{P}_{v_0 v_k}$ be the set of all v_o-v_k-path P. The definition of a path in undirected graphs is analogously.

Remark 2.7 A v_0-v_k-path with $v_0 = v_k$ is called a *cycle*.

At some points, the set of edges, that belong to a path, is relevant. Therefore, we introduce the following definition.

Definition 2.8 — Edge set of a path. Let a path $P = (v_0, e_1, v_1, \ldots, e_k, v_k)$ be given. Then, we refer to the edge set of the path as $E(P) = \{e_1, e_2, \ldots, e_k\}$.

Remark 2.9 A vertex v that has degree 0 or 1 is called a *singleton* or a *leaf*, respectively.

The following definition describes all vertices, which are connected to another vertex by a path.

Definition 2.10 — Reachability. For a graph $G = (V, E)$ a vertex $w \in V$ is *reachable* from $v \in V$, if there exists a v-w-path in G. We set

$$E_G(v) = \{w \in V \colon w \text{ is reachable from } v \in V\}$$

For undirected graphs, we introduce the following definition.

Definition 2.11 — Component and connectivity in undirected graphs. Let $G = (V, E)$ be an undirected graph.

1. $E_G(v)$ is the *component* of vertex v in G
2. H is *connected*, if $E_G(v) = V(G)$ for some (all) $v \in V(G)$

For directed graphs, the corresponding definition is given below.

Definition 2.12 — Component and connectivity in directed graphs. Let $G = (V, E)$ be a directed graph.

1. v and w are *strongly connected*, if $v \in E_G(w)$ and $w \in E_G(v)$
2. $\mathrm{Comp}_G(v) = \{w \in V : w \text{ and } v \text{ are strongly connected}\}$ is the *component* of v in G
3. G is *strongly connected*, if $\mathrm{Comp}_G(v) = V$ for some (all) $v \in V$
4. G is *weakly connected*, if the underlying undirected graph is connected

A special class of graphs that is important in this thesis are trees.

Definition 2.13 — Tree. A graph $G = (V, E)$ that contains no cycles is called a *forest*. If G is also connected, it is called a *tree*.

Furthermore, we introduce subgraphs of a given graph.

Definition 2.14 — Subgraph. For a given graph $G = (V, E, \beta)$, we call $G' = (V', E', \beta')$ a *subgraph* of G, if the following properties hold:

1. $V' \subseteq V$
2. $E' \subseteq E$
3. $\beta' = \beta|E'$

Remark 2.15 If G is a tree, let some vertex $r \in V$ be the root of the breadth-first-graph of G ([Lei+01]). We denote by G_v the subtree of G, which is rooted in vertex v and contains all descendants of vertex v in the breadth-first-graph of G with root r as well as their incident edges.

Remark 2.16 If the tree is rooted and directed, by *out-tree*, we refer to the tree for which the in-degree of the root is 0.

Also, for a graph $G = (V, E)$, let a set $E' \subseteq E$ be given. Then we define $G - E' := (V, E \setminus E', \beta_{E \setminus E'})$. If $E' = \{e\}$ is a single edge, we simply write $G - e$ instead of $G - E'$.

Edges of a graph can be assigned a weight, which can be, for example, the *length* or *cost*.

Definition 2.17 — Length function on a graph; network. For a graph $G = (V, E)$, the *length function* $\ell : E \to \mathbb{R}_+$ assigns a non-negative length to all edges of the (directed or undirected) graph. A graph together with a length function (G, ℓ) is called a *network*. Depending on the underlying graph, it is either directed or undirected.

Based on this definition, we define the length of a path in a graph.

Definition 2.18 — Length of a path. Let $G = (V, E)$ be a graph and $\ell : E \to \mathbb{R}_+$ be a length function. The *length* $\ell(P)$ *of a path* $P = (v_0, e_1, v_1, \ldots, e_k, v_k)$ in G is defined as

$$\ell(P) = \sum_{i=1}^{k} \ell(e_i).$$

For characterizing the shortest path between two vertices, the following definition is needed.

Definition 2.19 — Distance of two vertices. The *distance* of two vertices in a graph G with respect to a length function $\ell\colon E \to \mathbb{R}_+$ is defined as

$$d_\ell(u, v, G) \coloneqq \inf\{\ell(P) : P \text{ is a path in } G \text{ from } u \text{ to } v\}.$$

Remark 2.20 If there is no path P connecting two vertices u and v, the distance is set to $d_\ell(u, v, G) = \infty$. For convenience reasons we write ℓ_e instead of $\ell(e)$. If we want to denote the length of the edge between two distinct vertices u and v we write $\ell(u, v)$ instead of $\ell((u, v))$. Also, if the context is clear, we write $d(u, v)$ instead of $d_\ell(u, v, G)$. On the other hand, if the graph on which the distance is measured is not clear from context we may also write $d_G(u, v)$.

Remark 2.21 In all definitions, we might omit the subscript G, if the graph is clear from the context.

2.3 Optimization Theory, Complexity and Approximation

In this section, we want to introduce the basics of optimization theory as well as complexity and approximation. We refer to Ausiello et al. [Aus+99], Garey and Johnson [GJ79], Wolsey and Nemhauser [WN99], and Williamson and Shmoys [WS11] for a detailed overview of the topic.

Definition 2.22 — Optimization Problem. A quadruple $(I, \textsc{Sol}, m, \text{goal})$ of objects given in the following characterizes an *optimizaton problem* Π:

1. I is the set of instances
2. $\textsc{Sol}$ is a function that maps an input instance $\pi \in I$ to its set of feasible solutions
3. m is the measure function, defined for pairs (π, x), $\pi \in I, x \in \textsc{Sol}(\pi)$, that provides a non-negative integer – the value of x
4. goal defines, whether Π is a minimization or a maximization problem

If Π is a minimization problem, we wish to find a solution x^* that minimizes m over the set of feasible solutions for every instance. In case of a maximization problem, we wish to maximize m.

Remark 2.23 By x^*, we denote an *optimal solution* for an instance $\pi \in I$. Furthermore, we refer to the value of an optimal solution by $\mathrm{OPT}_\Pi(\pi) \coloneqq \text{goal}\{m(\pi, x)\colon x \in \textsc{Sol}(\pi)\}$ for an instance $\pi \in I$ of the optimization problem $\Pi = (I, \textsc{Sol}, m, \text{goal})$.

The *alphabet* is a finite, non-empty set Σ, while Σ^* is the set of all finite strings over the alphabet Σ. We can now define the *encoding scheme* to map objects to strings in Σ^*.

Definition 2.24 — Encoding scheme and input size. An encoding scheme is used to describe the instance of a problem (or other objects) in terms of strings $x \in \Sigma^*$ over an alphabet Σ. The *input size* $|x|$ is the number of characters needed to encode the instance.

Remark 2.25 As standard, we use the binary encoding scheme over $\Sigma = \{0, 1\}$ in the following. An integer n, for example, can be encoded via the binary encoding scheme in $\lceil \log_2(n) + 1 \rceil + 1$ bits. We omit the base 2 of the logarithm in the following.

Definition 2.26 — Decision problem. A *decision problem* is given by a set of instances that are either YES-instances or NO-instances. We want to verify for all instances, whether they are part of the YES-instances.

Remark 2.27 For an instance $\pi \in I$, and a threshold $K \geq 0$ the *decision problem of an optimization problem* Π asks whether there exists a feasible solution $x \in \Pi$, such that $m(\pi, x) \leq K$ (for minimization problems) or $m(\pi, x) \geq K$ (for maximization problems) holds true. The decision problem for an optimization problem is also called the *underlying language*.
We say that an algorithm *solves* the decision problem if it halts and can verify whether a given instance is a YES-instance or a NO-instance.

Definition 2.28 — Complexity class NP. The class NP consists of all decision problems for which there is a non-deterministic Turing machine M and a polynomial pol, such that M solves the decision problem after at most $\text{pol}(|x|)$ steps for every input $x \in \Sigma^*$.

In this thesis, we want to analyze the running time of algorithms. To do so, we use the $\mathcal{O}$-notation that is defined as follows.

Definition 2.29 — $\mathcal{O}$-Notation. Let $f, g \colon \mathbb{N} \to \mathbb{R}$ be two functions. Then, we use the following notations.

- $f \in \mathcal{O}(g(n))$ if there exist constants $c > 0, n_0$ and $a \in \mathbb{N}$, such that $f(n) \leq cg(n) + a$ for all $n \geq n_0$
- $f \in \Omega(g(n))$ if there exist constants $c > 0, n_0$ and $a \in \mathbb{N}$, such that $f(n) \geq cg(n) + a$ for all $n \geq n_0$
- $f \in \Theta(g(n))$ if $f \in \mathcal{O}(g(n))$ and $f \in \Omega(g(n))$

Now, we can define the running time of an algorithm as follows.

Definition 2.30 — Asymptotic running time and complexity of an algorithm. Let A be an algorithm and $T'_A(\pi)$ be the *running time* of A on an instance π. The number of steps that algorithm A takes specifies its running time. Usually, we are interested in the *worst-case-running time* T_A of an algorithm A on instances of size n:

$$T_A(n) := \max\{T'_A(\pi) \colon |\pi| \leq n\}.$$

Algorithm A has *complexity*

- $\mathcal{O}(g(n))$ if $T_A(n) \in \mathcal{O}(g(n))$
- $\Omega(g(n))$ if $T_A(n) \in \Omega(g(n))$
- $\Theta(g(n))$ if $T_A(n) \in \Theta(g(n))$

Remark 2.31 An algorithm A runs in polynomial time, if there exists a polynomial pol such that $T_A \in \mathcal{O}(\mathrm{pol(n)})$.

Definition 2.32 — Complexity class P. The class P consists of all decision problems for which there is a deterministic Turing machine M and a polynomial pol, such that M solves the decision problem after at most $\mathrm{pol}(|x|)$ steps for every input $x \in \Sigma^*$.

Definition 2.33 — Polynomial time reduction. Let Π and Π' be two decision problems. Let $f\colon \Pi \to \Pi'$ be a function, that maps instances of Π to instances of Π' such that $x \in \Pi$ if and only if $f(x) \in \Pi'$. It is called *polynomial time reduction*, if it is computable in polynomial time.

Definition 2.34 — NP-completeness. A decision problem P is called NP-*complete*, if it is in NP and for every decision problem Π' in NP, there is a polynomial time reduction from Π' to P.

In this thesis, we want to approximate different optimization problems. We introduce general definitions in the following.

Definition 2.35 — α-approximation. Let π be an instance of an optimization problem Π. For a minimization problem, a feasible solution $x \in \mathrm{Sol}(\pi)$ is an α-*approximation*, if $f(x) \leq \alpha \cdot \mathrm{OPT}_\Pi(\pi)$ for $\alpha \geq 1$.

Definition 2.36 — PTAS, FPTAS. A class of approximation algorithms $(A_\varepsilon)_{\varepsilon>0}$ for a problem Π is called a *polynomial-time approximation scheme (PTAS)*, if for every $\varepsilon > 0$, A_ε is an $(1+\varepsilon)$-approximation for every instance π of Π and runs in polynomial time in the encoding length of π. If additionally, the running time is polynomial in $1/\varepsilon$, we call $(A_\varepsilon)_{\varepsilon>0}$ a *fully polynomial-time approximation scheme (FPTAS)*.

2.4 Location Theory

This section introduces the basics of location theory. For an overview on location planning, we refer to [Ham95, LNG15, DH01].

While details may vary, most location problems can be described by five main types of information. Hamacher, Nickel, and Schneider [HNS98] discuss this in their paper and provide a classification scheme to shortly describe a location problem. It consists of five positions $P_1/P_2/P_3/P_4/P_5$, which stand for the following:

- P1 Number and type of facilities to be planned
- P2 The type of location problem (i.e. continuous, discrete, network)
- P3 Specialties, i.e. restrictions, forbidden spaces
- P4 Contains relation of existing to new facilities (i.e. distance function, costs)
- P5 Describes the objective

Remark 2.37 In this thesis, we only discuss location problems on networks. Therefore, we drop P2. Also, we might omit P3 for better readability, if no specialties need to be stated.

Classically, we differentiate between two objectives, the *center problem* and the *median problem*. The first one searches for minimum over all maximal distances from either one or more centers

to every other vertex. The latter one seeks to minimize the sum of the distances from either one or more new facilities to every other vertex. Depending on where we are allowed to place new facilities, different problems arise.

Definition 2.38 — Vertex restricted and absolute problems. Let $G = (V, E)$ be a graph. Then, we can either limit the possible locations for the new facilities to the vertices (*vertex restricted problem*) or allow them to also be placed on the edges connecting the vertices (*absolute location problem*).

Throughout this thesis we only consider cases, where the set of existing locations is the vertex set. These existing facilities might be equipped with weights $w_i > 0$ for all $i = 1, \dots, n$. Also, we denote the set of *new facilities* by

$$X = \{x_1, \dots, x_p\} \subseteq V.$$

The objective function of the center problem is then defined as follows.

Definition 2.39 — Center objective function. Let $G = (V, E)$ be a graph and $w_i > 0$. Then, the objective function for the center problem for a chosen set X of locations is

$$f(X) := \max_{i,X} w_i d(X, v_i) \quad \text{with } d(X, v) := \min_{x \in X} d(x, v).$$

Then, the p-center problem is stated as follows.

p-center location problem $(p, \max, G)$

Instance:	Undirected graph $G = (V, E)$, edge lengths $\ell \colon E \to \mathbb{Z}_+$, vertex weights $w_i > 0$ and number of locations $p \in \mathbb{Z}_+$.
Task:	Find a set $X \subseteq V$ of p new locations such that the objective function of the p-center location problem is minimal, i.e. minimize

$$\max_{i,X} w_i d(X, v_i)$$

In the following, we state the objective function of the median problem.

Definition 2.40 — Median objective function. Let $G = (V, E)$ be a graph and $w_i > 0$. Then, the objective function for the median problem for a chosen set X of locations is

$$f(X) := \sum_i w_i d(X, v_i) \quad \text{with } d(X, v) := \min_{x \in X} d(x, v).$$

Based on this objective, we formulate the p-median location problem as follows.

p-median location problem $(p, \sum, G)$

INSTANCE: Undirected graph $G = (V, E)$, edge lengths $\ell\colon E \to \mathbb{Z}_+$, vertex weights $w_i > 0$ and number of locations $p \in \mathbb{Z}_+$.

TASK: Find a set $X \subseteq V$ of p new locations such that the objective function of the p-median location problem is minimal, i.e. minimize

$$\sum_{v \in V} w_i d(X, v_i).$$

Remark 2.41 Optimal solutions for both cases of problems are denoted by X^*, while the optimal objective function value is denoted by $\mathrm{OPT}(p, \max, G)$ and $\mathrm{OPT}(p, \sum, G)$, respectively.

2.5 Interdiction Problems

This subsection gives a brief overview over interdiction problems. Network interdiction problems involve an additional opposing force, called the interdictor. Said interdictor wishes to worsen the objective function value of the optimization problem. The set of feasible interdiction strategies is defined as follows.

Definition 2.42 — Interdiction strategy. Let $B \in \mathbb{Z}_+$ be the interdiction budget. Furthermore, let $b(e) \in \mathbb{Z}_+$, i.e., $b\colon E \to \mathbb{Z}_+$ be the interdiction cost for each edge $e \in E$. Then, the *set of all feasible interdiction strategies*, denoted by Γ, can be expressed as follows:

$$\Gamma := \left\{ \gamma = (\gamma_e)_{e \in E} \in \{0,1\}^m \mid \sum_{e \in E} b(e) \cdot \gamma_e \leq B \right\},$$

where γ_e equals one, if edge e is interdicted or zero, if not.

The set of optimal interdiction strategies is denoted by $\Gamma^* = \{\gamma^* \in \Gamma \mid \gamma^* \text{ optimal}\}$. In what follows, each interdiction strategy $\gamma \in \Gamma$ induces an undirected graph $G(\gamma) := (V', E')$ with $V' = V$ and $E' = E \setminus E(\gamma)$, where $E(\gamma) := \{e \in E \mid \gamma_e = 1\}$. In this thesis, the locator places their facility on $G(\gamma)$, i.e. after the interdiction step.

Based on this, we define the decision version of the p-median location interdiction problem as follows.

Decision version of the p-median location interdiction problem

INSTANCE: Undirected graph $G = (V, E)$, edge lengths $\ell\colon E \to \mathbb{Z}_+$, interdiction costs $b\colon E \to \mathbb{Z}_+$, interdiction budget $B \in \mathbb{Z}_+$, number of locations $p \in \mathbb{Z}_+$, and decision parameter $K \in \mathbb{Z}_+$.

QUESTION: Does there exist an interdiction strategy $\gamma \in \Gamma$ such that

$$\min_{X \subseteq V, |X| = p} \sum_{v \in V} d_{G(\gamma)}(v, X) \geq K \ ?$$

In the optimization version of the stated problem, we aim to find the maximum K for which the decision version is a YES-instance.

2.6 Parametric Optimization

In this section we want to introduce the reader to linear parametric optimization problems. For a more extended introduction, we refer to the book of Gal [Gal94].
As well known, a standard linear program is given by a set of feasible solutions X and a linear objective function $f\colon X \to \mathbb{R}, f(x) \coloneqq c^\top x$ for $c \in \mathbb{R}^n, n \in \mathbb{N}$, which we wish to minimize while staying feasible, thus can be stated as

$$\min_{x \in X} f(x)$$

In the parametric version of this problem, we introduce a parameter interval $\Lambda \subseteq \mathbb{R}$ as well as a parameter $\lambda \in \Lambda$, and reformulate the objective function in the following manner: let two functions $a, b \coloneqq X \to \mathbb{R}$ be given. The objective function of the linear parametric minimization problem is then defined as

$$f_\lambda\colon X \to \mathbb{R}, f_\lambda(x) \coloneqq a(x) + \lambda \cdot b(x).$$

Linear parametric minimization problem Π^P

INSTANCE: Non-empty set X of feasible solutions, two functions $a, b : X \to \mathbb{R}$, parameter interval $\Lambda \subseteq \mathbb{R}$

TASK: For every $\lambda \in \Lambda$, find an optimal solution $x^*_\lambda \in X$ such that the objective function is minimal, i.e. minimize

$$f_\lambda(x) = a(x) + \lambda \cdot b(x)$$

By fixing the parameter to λ, we obtain the so called non-parametric version $\Pi^P(\lambda)$ of the parametric problem Π^P.

Remark 2.43 If an optimal solution of $\Pi^P(\lambda)$ exists for every $\lambda \in \Lambda$, the optimal solution of the linear parametric minimization problem is a division of the parameter set Λ into finitely many intervals $(-\infty, \lambda_1], [\lambda_i, \lambda_{i+1}]_{i=1,\cdots,I-1}, [\lambda_I, +\infty)$ together with the optimal solution x for the corresponding interval [Hel+22].

Definition 2.44 — Critical region. For a solution x, its *critical region* $\Lambda(x) \subseteq \Lambda$ is the interval of all parameters for which x is optimal for $\Pi(\lambda)$.

Definition 2.45 — Optimal value function. The *optimal value function*

$$F\colon \Lambda \to \mathbb{R}, F(\lambda) \coloneqq \min_{x \in X} f_\lambda(x)$$

maps λ to the optimal objective function value of $\Pi^P(\lambda)$ for every $\lambda \in \Lambda$.

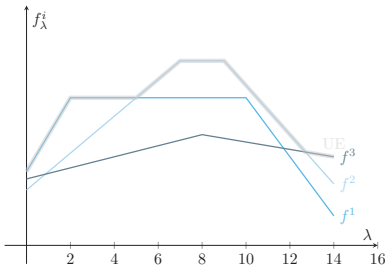


Figure 2.1: Visualization of the upper envelope.

Remark 2.46 For minimization problems, the optimal value function is piecewise linear and concave [Gus83].

Definition 2.47 — Breakpoints. The changes in the slope of the optimal value function are called *breakpoints*.

Remark 2.48 If for the intervals $(-\infty, \lambda_1], [\lambda_i, \lambda_{i+1}]_{\lambda=1,\cdots,I-1}, [\lambda_I, +\infty)$ I is minimal, breakpoints correspond to the λ_i.

Definition 2.49 — Complexity of the optimal value function. The number of breakpoints of the optimal value function indicates its *complexity*.

Remark 2.50 We refer to the number of changes of the optimal solution x^*_λ as the *complexity of the solution*.

Throughout the section on parametric optimization, upper and lower envelopes of functions as well as envelopes of linear line segments are needed. The relevant definitions are given below.

Definition 2.51 — Line segment. A *line segment* is a linear function s, whose domain is an open interval $[s_l, s_r]$, thus $s\colon (s_l, s_r) \to \mathbb{R}, x \mapsto s(x)$.

Definition 2.52 — Upper envelope. Let $S = \{s_1, \ldots, s_n\}$ be a set of n line segments. Then, the *upper envelope* of the segments is a function

$$\mathrm{UE}(S, x)\colon \mathbb{R} \to \mathbb{R}, x \mapsto \max_x \{-\infty, \{s_i(x)\colon s_i \in S, x \in (s_l)_i < x < (s_r)_i\}\}.$$

Definition 2.53 — Lower envelope. Let $S = \{s_1, \ldots, s_n\}$ be a set of n line segments. Then, the *lower envelope* of the segments is a function

$$\mathrm{LE}(S, x)\colon \mathbb{R} \to \mathbb{R}, x \mapsto \min_i \{\infty, \{s_i(x)\colon s_i \in S, (s_l)_i < x < (s_r)_i\}\}.$$

As a visualization, in Figure 2.1 the upper envelope of three parametric functions $f^1_\lambda, f^2_\lambda, f^3_\lambda$ is shown.
The following results concerning the running time to compute upper and lower envelopes and the complexity thereof are used multiple times throughout the thesis.

Theorem 2.54 [Her89] The upper as well as the lower envelope of n line segments can be computed in $\mathcal{O}(n \log n)$.

As for the complexity, in [Her89], we find the following.

Theorem 2.55 [Her89, WS88] The complexity of the upper envelope of n line segments is in $\mathcal{O}(n\alpha(n))$.

If the segments are ordered, the following holds.

Theorem 2.56 [CW02] If the n line segments are sorted in slope and sorted in the endpoints of their x-coordinate, the upper envelope can be calculated in $\mathcal{O}(n \log \log n)$.

Remark 2.57 The three theorems stated before hold also true for the computation of lower envelopes.

In this thesis, we introduce parameters in network setups. Here, we equip the edges of the underlying network with lengths that are dependent of λ, yielding edge lengths of the form $\ell_\lambda(e) = a(e) + \lambda b(e)$, for $a, b = E \to \mathbb{R}$ being two functions.

Remark 2.58 The distances in the parametric network are then referred to as $d_\lambda(v_i, v_j)$ for two vertices v_i, v_j.

Parametric Location Problems

3 Introduction and the Graph of Mulmuley and Shah

3.1 Introduction

As seen in Chapter 2, the goal of standard linear optimization problems is to maximize or minimize a linear function $f(x) = c^\top x$, dependent on a variable x, while staying feasible, thus $x \in X$ with X being the set of feasible solutions. When speaking of linear parametric optimization problems, we refer to a group of problems where the objective function is of the form

$$f_\lambda(x) = a(x) + \lambda \cdot b(x), \tag{3.1}$$

with $a(x), b(x) \colon X \to \mathbb{R}$ being two functions and $\lambda \in \Lambda \subseteq \mathbb{R}$. Thus, for every solution x, the objective function value is a function in the parameter λ. For parametric optimization problems Π^P the goal is to find an optimal solution for every possible value of λ. If the non-parametric version $\Pi^P(\lambda)$ has an optimal solution for every $\lambda \in \Lambda$, an optimal solution set for the parametric problem consists of a division of the interval Λ into intervals $(-\infty, \lambda_1], [\lambda_i, \lambda_{i+1}]_{\lambda=1,\cdots,I-1}, [\lambda_I, +\infty)$ and for every interval, a solution x that is optimal for every $\Pi(\lambda)$ the corresponding interval [Hel+22].

There are several real-world scenarios where input data may vary. Here, the introduction of a parameter that reflects these changes, is a suitable model to tackle different real-world-problems. Thus, the field of parametric optimization has been of interest for the last decades. There is various literature on the applications of parametric optimization, such as the planning of air transportation fleets [Jen87], planning of heat and power systems [DAP16] or application in process systems engineering [Obe+16, Pis+02]. Furthermore, it is interconnected to other mathematical fields, such as robust optimization [AP20], sensitivity analysis [GG12], multi-objective optimization [HRT24] or multi-level programming [PDR03, AP19], just to name a few.

Numerous optimization problems have been studied to this day. This includes for example, but not exclusively, the parametric variants of the shortest path problem [MS00, GR19, Car83b, KO81], the assignment problem [Car83b, GK10], the minimum cost flow problem [Car83a, Ruh88, HPR07], the knapsack problem [Giu+17, HK17] or the minimum spanning tree problem [FSE96]. For an extensive overview of parametric optimization problems and their real-world applications, we refer to the recent survey by Nemesch et al. [Nem+25].

In Section 2.6 we already mentioned that fixing the parameter λ in a parametric optimization problem leaves us with a standard linear optimization problem – the non-parmetric version of the parametric problem. When comparing the former with the non-parametric version, it is clear that

the parametric problem is at least as hard to solve as the latter. Since we are initially interested in a solution x for every λ, we quickly discover that the computation of an optimal solution set might be excessive in terms of computational effort. When analyzing the objective value function of the underlying parametric optimization problem as a function of λ, we find, that for several problems, this optimal value function can have super-polynomially many breakpoints, which are changes in its slope and mark the limits of each interval introduced in the beginning. Each of these breakpoints indicates a change in the optimal solution, thus making the computation of an optimal solution set impossible to do in polynomial time.

Still, the parametric and the non-parametric version of a problem may remain tractable, as it is the case for the minimum spanning tree problem [FSE96], for example. Conversely, this is not the case for the assignment problem [Car83b], the minimum cost flow problem [Car83a] and the shortest path problem, for instance. Different authors have shown that the optimal value function of the parametric shortest path problem, which maps the parameter λ to its optimal objective value, can have super-polynomially many breakpoints. This has been studied for general directed graphs in [MS00] and [Car83b], and for planar graphs in [GR19]. As a result, an exact algorithm might have to compute super-polynomially many solutions, which justifies the need of approximation schemes. At this point, we mention the paper of Bazgan et al. [Baz+22], that introduces a general approximation method for parametric optimization problems based on the Eisner-Severance method of [ES76]. Since we use their results in this thesis, we briefly introduce the reader to the topic. First, we recall mild assumptions that are needed to make sure that for all $x \in X$ and $\lambda \in \Lambda$ the values of $f_\lambda(x)$ are non-negative and also to ensure computational bounds.

- The parametric problem Π^P is given as in (3.1) with $\Lambda = [\lambda_{\min}, \infty], \lambda_{\min} \in \mathbb{R}$
- The optimal cost curve is well defined
- $a(x)$ and $b(x)$ are polynomial-time computable
- The positive rational upper and lower bounds UB and LB for $f_\lambda(x)$ are polynomial-time computable such that $f_{\lambda_{\min}}(x), b(x) \in 0 \cup [\mathrm{LB}, \mathrm{UB}]$ for all $x \in X$

Furthermore, for $\beta \geq 1$, a polynomial-time β-approximation algorithm ALG_β must be available for the non-parametric version of the problem that is used as a subroutine in the approximation algorithm. Then, in time polynomial in $1/\varepsilon$ and the input size of the problem, a $(1+\varepsilon)$-approximation can be computed if an exact algorithm is given for the non-parametric version $\Pi^P(\lambda)$ for every $\lambda \in \Lambda$. If a β-approximation exists for $\Pi^P(\lambda)$, a $(1+\varepsilon)\cdot\beta$-approximation can be computed. More precisely, their approximation ensures a running time of $\mathcal{O}(T_{\mathrm{LB/UB}} + T_{\mathrm{ALG}_\beta} \cdot (1/\varepsilon \cdot \log 1/\varepsilon + 1/\varepsilon \cdot \log \mathrm{UB}/\mathrm{LB} + 1/\varepsilon \cdot \log \beta))$, where $T_{\mathrm{LB/UB}}$ is the time to compute upper and lower bounds and T_{ALG_β} is the running time of the subroutine used. For discrete problems, this running time reduces to

$$\mathcal{O}\left(T_{\mathrm{LB/UB}} + T_{\mathrm{ALG}_\beta} \cdot \left(\frac{1}{\varepsilon} \cdot \log \frac{\mathrm{UB}}{\mathrm{LB}} + \frac{1}{\varepsilon} \cdot \log \beta\right)\right). \tag{3.2}$$

We briefly describe how the algorithm works. The algorithm subsequently analyzes intervals $[\lambda_l, \lambda_r]$. For the β-approximate solutions x_l for $\Pi^P(\lambda_l)$ and x_r for $\Pi^P(\lambda_r)$, respectively, it checks whether either x_l or x_r is $(1+\varepsilon)\cdot\beta$-approximate for the whole interval $[\lambda_I, \lambda_r]$. This is done by first evaluating, if $f_{\lambda_r}(x_l) \leq f_{\lambda_r}(x_r)$ or $f_{\lambda_l}(x_r) \leq f_{\lambda_l}(x_l)$. If neither of the cases holds true,

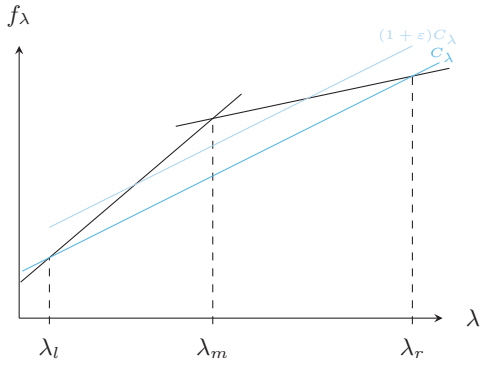


Figure 3.1: Illustration of the approximation scheme provided by Bazgan et al. [Baz+22]; visualization of one interval where neither x_l or x_r are approximate for the whole interval and thus the interval gets splitted.

consider the point where $f_\lambda(x_l)$ and $f_\lambda(x_r)$ intersect. Let $\lambda_m := \mu(x_l, x_r) := {}^{a(x_r)-a(x_l)}/_{b(x_l)-b(x_r)}$ be the corresponding λ. Then, it is checked whether the line connecting the objective function values $f_{\lambda_r}(x_r)$ and $f_{\lambda_l}(x_l)$ is within a factor of $(1+\varepsilon)$ away of the optimal value function at λ_m. If this is note the case, this is, x_l or x_r are not $(1+\varepsilon)\cdot\beta$-approximate for the whole interval $[\lambda_I, \lambda_r]$, it gets partitioned into two new intervals $[\lambda_l, \lambda_m]$ and $[\lambda_m, \lambda_r]$, which get queued for a later analysis. If the line is within a factor of $(1+\varepsilon)$, x_l or x_r are $(1+\varepsilon)\cdot\beta$-approximations for the whole interval. Details on how to compute the starting interval are given in [Baz+22]. Figure 3.1 shows a visualization for one interval. The blue line indicates the connecting line C_λ of $f_{\lambda_r}(x_r)$ and $f_{\lambda_l}(x_l)$, while the light blue line shows $(1+\varepsilon)\cdot C_\lambda$.

At this point, we mention three things on where we use a slight modification of the provided algorithm. Algorithm 3.1 provides the reader with the algorithm, where the deviations are already integrated.

First, the algorithm as stated in [Baz+22] only returns the $(1+\varepsilon)\cdot\beta$-approximate solutions. However, we also need the corresponding intervals of λ and the corresponding line segments. Nevertheless, at no additional computational cost we can store these information as well, since both the line segment and the interval are already being used to determine whether the current x_l or x_r is $(1+\varepsilon)\cdot\beta$-approximate or not. In fact, it is sufficient to output x_i together with the corresponding interval to encode all information needed. Hence, the modified output set contains triplets of a $(1+\varepsilon)\cdot\beta$-approximate solution x_i together with the corresponding interval $[\lambda_{l_i}, \lambda_{r_i}]$.

Second, when the algorithm detects that the intersection point is within a factor of $(1+\varepsilon)$ away from $C(\lambda)$, it adds both $(1+\varepsilon)\cdot\beta$-approximate solutions x_l and x_r to the output set. At this point, we divide the corresponding interval $[\lambda_l, \lambda_r]$ at λ_m which we calculated before and store x_l as the approximation for $[\lambda_l, \lambda_m]$ and x_r for $[\lambda_m, \lambda_r]$.

Third, when generating the intervals that get examined in next iterations of the approximation algorithm, we use a stack to store them, as suggested in their paper by Bazgan et al. [Baz+22, Remark 4]. That is, whenever an interval $[\lambda_l, \lambda_r]$ is divided into intervals $[\lambda_l, \lambda_m]$ and $[\lambda_m, \lambda_r]$, we stack the latter before the former. This way we ensure that the leftmost interval in terms of its λ-values that is not yet added to the solution set is the next interval to get examined. As a result, the solution set is ordered in terms of the interval bounds λ_i as well as the slopes $b(x)$ for $\beta = 1$.

These modifications do not affect the validity of the algorithm, but provide the solution data in the described way, which is needed in algorithms developed later on. We provide the reader with the modified algorithm of Bazgan et al. [Baz+22] in Algorithm 3.1. Whenever we refer to the approximation algorithm, we mean the following.

Algorithm 3.1 — Approximation of parametric optimization problems [Baz+22].

Input: An instance of a parametric optimization problem Π with the assumptions stated above, $\varepsilon > 0$, β-approximation algorithm ALG_β for the non-parametric version of Π

Output: A set of $(1+\varepsilon)\cdot\beta$-approximations x_i for the given instance of Π together with their corresponding critical regions $[\lambda_{l_i}, \lambda_{r_i}]$

Compute LB and UB
$\lambda_* \leftarrow \lambda_{\min} + {}^1\!/_\beta \cdot \varepsilon \cdot {}^{\mathrm{LB}}\!/_{\mathrm{UB}}$; $\lambda^* \leftarrow \lambda_{\min} + \beta \cdot {}^1\!/_\varepsilon \cdot {}^{\mathrm{UB}}\!/_{\mathrm{LB}}$
$x_* \leftarrow \mathrm{ALG}_\beta(\lambda_*)$; $x^* \leftarrow \mathrm{ALG}_\beta(\lambda^*)$
`stack` $\leftarrow \varnothing$
`stack`.push($([\lambda_*, \lambda^*], x_*, x^*)$) ▷ stack of intervals still to be considered
$S \leftarrow \varnothing$ ▷ solution set
while `stack` $\neq \varnothing$ **do**
    $([\lambda_l, \lambda_r], x_l, x_r) \leftarrow$ `stack`.pop()
    if $f_{\lambda_r}(x_l) \leq f_{\lambda_r}(x_r)$ **then**
        $S \leftarrow S \cup \{(x_l, [\lambda_l, \lambda_r])\}$
    else if $f_{\lambda_l}(x_r) \leq f_{\lambda_l}(x_l)$ **then**
        $S \leftarrow S \cup \{(x_r, [\lambda_l, \lambda_r])\}$
    else
        $\lambda_m \leftarrow \mu(x_l, x_r)$
        if $f_{\lambda_m}(x_l) \leq (1+\varepsilon)\cdot\left(\frac{\lambda_r-\lambda_m}{\lambda_r-\lambda_l}\cdot f_{\lambda_l}(x_l) + \frac{\lambda_m-\lambda_l}{\lambda_r-\lambda_l}\cdot f_{\lambda_r}(x_r)\right)$ **then**
            $S \leftarrow S \cup \{(x_l, [\lambda_l, \lambda_m]), (x_r, [\lambda_m, \lambda_r])\}$
        else
            $x_m \leftarrow \mathrm{ALG}_\beta(\lambda_m)$
            `stack`.push($([\lambda_m, \lambda_r], x_m, x_r)$)
            `stack`.push($([\lambda_l, \lambda_m], x_l, x_m)$)
        end if
    end if
end while
return S

Note, that Bazgan et al. [Baz+22, Remark 1] suggest values for λ_*, λ^*, that are calculated independently of ε for discrete problems, which leads to the running time stated in (3.2).

Summarizing the previous information, when analyzing parametric optimization problems, different questions arise depending on which aspect of the problem we look at. Probably the first question that comes to our mind is how the problem can be solved, thus if there is an algorithm that outputs an (or more) exact or approximate solution(s). Also, the complexity of the solution itself and of the objective function value corresponding to the solution(s) is of interest.

The standard location problems on graphs differentiate between directed and undirected general graphs and quite often directed and undirected trees. On all structures, we analyze parametric variants of the center and the median location problem (for a definition recall Section 2.4). To the best of our knowledge, no results are known for parametric location problems. So in the remainder of this part of the thesis, we want to answer the two stated questions about the complexity and analyze exact or approximate algorithms for the different parametric location problems (PLPs). The results are summarized in the following three tables. The first table leads the reader to the exact algorithms and approximation schemes, respectively.

	Center Problem		Median Problem	
	Directed	Undirected	Directed	Undirected
General Graphs	Algorithm 4.7	Algorithm 4.7, Remark 4.10	Algorithm 5.2	Algorithm 5.2, Remark 5.6
Trees	Remark 4.12	Algorithm 4.14	Remark 5.8	Algorithm 5.10 Algorithm 5.17

Table 3.1: Algorithms for the PLPs.

The next table shows, for which problems the complexity of the optimal value function has been analyzed.

	Center Problem		Median Problem	
	Directed	Undirected	Directed	Undirected
General Graphs	Theorem 4.1	–	Theorem 5.1	–
Trees	Proposition 4.13	Theorem 4.17	Proposition 5.9	Theorem 5.14 Theorem 5.19

Table 3.2: Analysis of the complexity of the objective function for the different PLPs.

The last table gives an overview of where to find the results on the complexity of the solutions of the different problems.

	Center Problem		Median Problem	
	Directed	Undirected	Directed	Undirected
General Graphs	Theorem 4.6	–	–	–
Trees	Proposition 4.11	Theorem 4.18	Proposition 5.7	Remark 5.12 Theorem 5.20

Table 3.3: Analysis of the complexity of the solution for the different PLPs.

3.2 Construction of Mulmuley and Shah

One of the results that we use in the remainder of this chapter is the analysis of the parametric shortest path problem from Mulmuley and Shah in [MS00]. Since we use their graph multiple times throughout, we introduce it up to a certain level of detail. Its construction is very technical

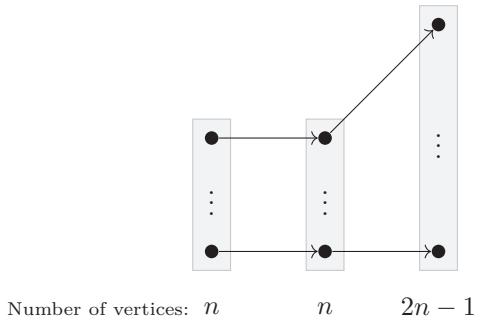


Figure 3.2: Core $G_{1,n}$ of the graph of Mulmuley and Shah; layers are indicated by boxes.

and listing its exact details will generate no added value in terms of understanding this section. However, we will present all information that is necessary for the proofs of our results with fewer details than in the original paper. While the construction of the graph is described first in [MS00], we recommend [GR19] for a detailed insight on the construction process, since it also uses the original graph, but provides a clearer description of it.

It is known that the objective function of the parametric shortest path problem can have super-polynomially many breakpoints (see [MS00, Car83b, Nik+06]). In the following, we show how Mulmuley and Shah [MS00] construct a directed graph $G = (V, E)$ with edge weights that are linear in a parameter $\lambda \in \Lambda$, for which this holds. Using this graph, we can similarly demonstrate that the objective function of the parametric center problem may have a super-polynomial number of breakpoints. This characteristic may render the computation of the objective function values for each center as a function of $\lambda \in \Lambda$ inefficient.

Core Structure of the Graph. We begin by describing the general structure of the graph G. Let $m, n \in \mathbb{N}$. The final core $G_{m,n}$ of G is a layered structure together with a dedicated starting vertex $s \in V$ and an ending vertex $t \in V$.

> **Remark 3.2** Note, that by $G_{m,n}$ we only refer to the layered „inner“ graph without the vertices s and t.

Each graph $G_{k,n}$ is constructed inductively based on its predecessor $G_{k-1,n}$ for $k \in \{1, \dots, m\}$. The parameter m indicates the number of inductive construction steps, while n denotes the number of vertices in the first layer of the base graph $G_{1,n}$, referred to as the *core* in the paper.

The vertices of $G_{m,n}$ are indexed according to their layer and row numbers: the vertex in the ith row and jth layer is denoted as $v_{i,j}$. Note that the row index starts at 0. If the context is clear, the layer index may be omitted for brevity.

The core consists of three layers, containing n, n, and $2n-1$ vertices, respectively. See Figure 3.2 for a graphical representation. The edges connecting these layers are indicated and will be formally introduced next.
The layers are connected as follows. Each vertex $v_{i,1}$ in the first layer has exactly one successor $v_{i,2}$ in the second layer for $i \in \{0, \dots, n-1\}$. The connections from the second to the third layer follow a different structure: each vertex $v_{i,2}$ has n successors, specifically $v_{i+j,3}$ for $j \in \{0, \dots, n-1\}$.

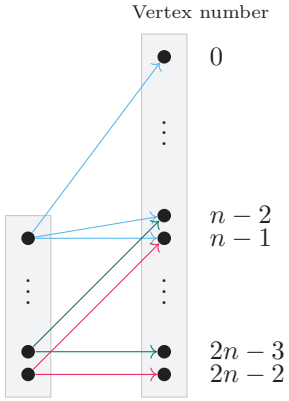


Figure 3.3: Connection of the second and third layer of $G_{1,n}$; not all edges are shown.

This means that the first vertex in layer two connects to the first n vertices of layer three (indexed 0 to $n-1$), the second vertex connects to the next n vertices (indexed 1 to n), and so forth. For a visual representation, refer to Figure 3.3, where not all edges are displayed for clarity.

Building the graph. While the graph $G_{1,n}$ consists of three layers and their connections as described above, every subsequent graph is constructed differently. Graph $G_{m,n}$ comprises three copies of graphs built in previous steps as follows. One copy of $G_{m-1,n}$ forms the left part, a reversed copy of $G_{m-1,n}$ constitutes the middle part, and a copy of $G_{m-1,2n-1}$ serves as the right part, referred to as G^L, G^M, and G^R, respectively.

Remark 3.3 Note that by a „reversed“ copy, we mean that not only is the order of the layers in $G_{m-1,n}$ reversed, but also the edges connecting the layers are inverted while maintaining the general edge structure and the weight functions on the edges. See Figure 3.4 for an illustration of $G_{1,n}$ and its reversed counterpart.

Example 3.4 Let $m = 1$. The core $G_{1,n}$ and its reversed copy are shown in Figure 3.4. As we can see, the arrangement of the layers is reversed, while the orientation of the connecting edges is also inverted.

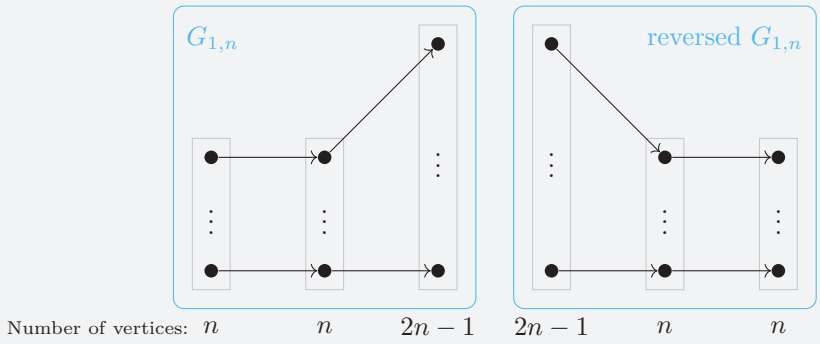


Figure 3.4: Core $G_{1,n}$ and its reversed copy.

We will now elaborate on the interconnection of G^L, G^M, and G^R.

The vertices of the last layer of G^L are connected one-to-one with the vertices of the first layer of the reversed G^M. Notably, both layers contain the same number of vertices. The last layer of G^M is connected to the first layer of G^R in the same manner as the second and third layers in the core graph. Specifically, the first vertex of the third layer in G^M is connected to the first n vertices (numbered 0 to $n-1$) of the first layer of G^R, the second vertex of the last layer is connected to the next n vertices (numbered 0 to $n-1$), and so forth. With each iteration , the graph $G_{m,n}$ expands, ultimately reaching at most $2^m(2n-1)$ rows and 3^m layers. Figure 3.5 illustrates the complete construction of $G_{2,n}$ as an extension of the previous Example 3.4.

Example 3.5 Let $m = 1$. For constructing $G_{2,n}$ we need $G_{1,n}$, its reversed copy and $G_{1,2n-1}$ as G^L, G^M and G^R, respectively. We have already shown how G^L and G^M are built in Figure 3.3 of Example 3.4. In Figure 3.5, we see the completed graph $G_{2,n}$ and the connecting edges inbetween the three parts of the graph.

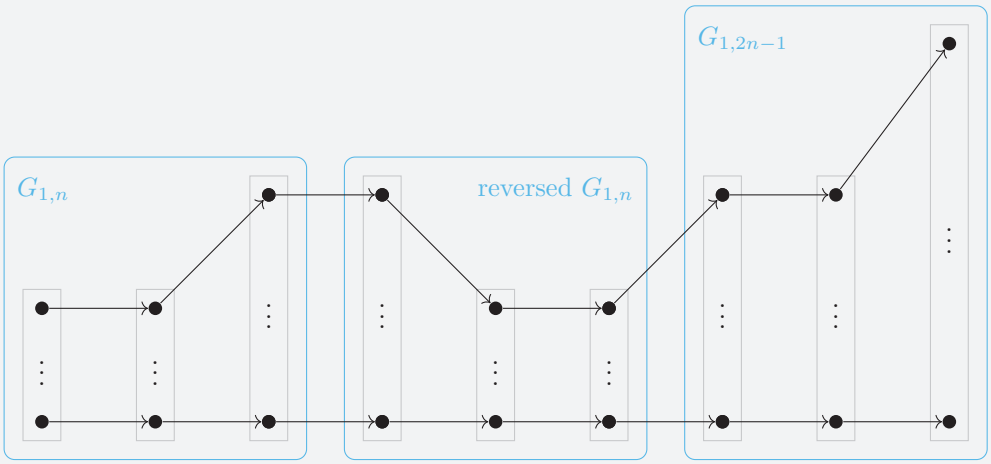


Figure 3.5: Graph $G_{2,n}$.

Weights and the curve of the objective function. After building the structure of the graph $G_{m,n}$, we now recall weights and the curve of the objective function. In the beginning we already mentioned that the final graph has a dedicated starting vertex $s \in V$ and an end vertex $t \in V$. Vertex s is connected to all n vertices in the first layer, while all vertices in the last layer are connected to t resulting in the final graph. A simplified illustration is given in Figure 3.6.

Further details on the construction of the weights can be found in the corresponding paper. However, two key aspects are worth mentioning. The starting vertex s is connected to its successors with weights denoted as $ws_\lambda(i)$ for $i = \{0, \ldots, n-1\}$. The vertices in the last layer are connected to t with weights 0. The construction process leads to disjoint intervals, within each of which n pairwise vertex-disjoint shortest s-t-path exist. To achieve this, in each iteration, the previous intervals are stretched by a factor $N > n$ and then split into n segments to generate n new shortest paths for each „old“ shortest path.

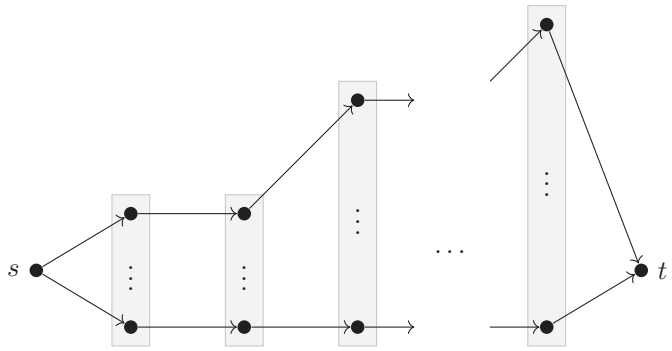


Figure 3.6: Simplified structure of the graph of Mulmuley and Shah; layers are indicated by boxes

Remark 3.6 The first of the n shortest paths passes through the first vertex of the first layer $v_{0,1}$ of $G_{m,n}$, the second of the shortest paths through the second vertex $v_{1,1}$ and so forth. The next set of shortest paths again starts at vertex $v_{1,0}$ ensuring, that the vertex succeeding s always rotates from top to bottom of the first layer.

Through the construction of the intervals and the increasing number of shortest paths Mulmuley and Shah [MS00] receive with every induction step, the authors can show, that the optimal value function of a parametric shortest path problem may have super-polynomially many breakpoints, as the authors state in Theorem 1.3:

Theorem 3.7 [MS00] There is an explicit family of graphs G_n on n vertices, with edge weights that are linear functions in a parameter λ, such that the optimal cost graph of the weight of the shortest path between s and t has $2^{\Omega(\log^2 n)}$ breakpoints.

Since we need this graph for the remainder of this section, we want to give a short summary of the main results, that we need in further proofs.

- A vertex of one specific layer only has successors in its following layer.
- Vertices of one layer are not connected to each other.
- s is connected to all vertices in the first layer of $G_{m,n}$ with weights $ws_\lambda(i)$, while all vertices of the last layer are connected to t via weightless edges.
- For increasing λ, the consecutive super-polynomially many shortest $s-t$-paths pass vertices $v_{i,1}, i = \{1, \ldots, n-1\}$ one by one, always starting at the top $v_{0,1}$, ending at $v_{n-1,1}$.

4 Parametric Center Problems

In this chapter, we analyze parametric 1-center problems on networks. Recall from Section 2.4, that the (non-parametric) center location problem is defined as minimizing $f(x)$ with $f(x) = \max_{i,x} w_i d(v_i, x)$. Also, as stated before, the edge lengths ℓ_λ of all graphs analyzed are linearly dependent on the parameter λ. Thus, the parametric center problem can be stated as follows:

Parametric center location problem $(1, G, \max)_\lambda$

INSTANCE: Graph $G = (V, E)$, edge weights $\ell_\lambda = a(e) + \lambda b(e)$, such that $a(e), b(e) \colon E \to \mathbb{R}$, a parameter set $\lambda \in \Lambda$ for $\Lambda \subseteq \mathbb{R}$, vertex weights $w > 0$

TASK: For every $\lambda \in \Lambda$, find an optimal location $x^*_\lambda \subseteq V$ such that the objective function of the center location problem is minimal, i.e. minimize

$$\max_i w_i d_\lambda(v_i, x)$$

The non-parametric 1-center problem is studied thoroughly. The literature on this problem traces back to the work of Hakimi [Hak64], where he motivates the idea of center problems by the need of a location for a police station, that minimizes the maximal distance to their operational areas. In his work, he indicates, that the vertex restricted center problem on both vertex-weighted and vertex-unweighted graphs can be solved by computing the all-pair-shortest paths on the given network and evaluate over the resulting distances. He also proposes an algorithm to solve the absolute center problem, where points on edges serve as feasible solutions as well. Later, Kariv and Hakimi [KH79a] optimize this algorithm and provide one with a running time of $\mathcal{O}(mn + n^2 \log n)$ for the absolute center without vertex weights. The weighted case is considered in this work as well, where they state an algorithm that runs in $\mathcal{O}(mn \log n)$. On tree structures, an algorithm for the computation of an absolute center is proposed by Megiddo [Meg83], that has a running time of $\mathcal{O}(n)$. Although not directly part of this work, we mention the p-center problem, where p new locations are sought that minimize the distances to their nearest existing facilities. Kariv and Hakimi [KH79a] prove this problem to be NP-hard on general graphs while giving an exact algorithm for the case of a tree. Several heuristics are known for the problem on general graphs. A thorough overview can for instance be found in [LNG15].

The next Section 4.1 deals with the center problem on directed general graphs, while Section 4.2 shows the results for the analysis of the problem on trees.

4.1 Directed General Graphs

For the remainder of this section, the underlying structure of the center problem are directed general graphs with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w > 0$.

First, we analyze the objective function of the parametric center problem. We will use the construction introduced in Section 3.2 to prove that the objective function can have super-polynomially many breakpoints, which is stated in the following theorem.

Theorem 4.1 The optimal value function of the parametric center problem can have super-polynomial many breakpoints.

For the proof of this theorem, we will need an upper bound M, that is bigger than the distance between a possible center $v \in V$ and every other vertex $u \in V$ in graph G. Note, that we can only bound $d_\lambda(v_i, v_j)$, if a v_i-v_j-path exists, since otherwise it is $d_\lambda(v_i, v_j) = \infty$. However, the feasible set $X \subseteq V$ of vertices for the location problem consists only of those vertices, from which every other vertex in V is reachable. In the non-parametric setup where λ is fixed, the distances are independent of the parameter. Assuming all v_i-v_j-path exist, it is easy to see, that there exists an M fulfilling the property. Let $d(v_s, v_t) := \max_{i,j} d(v_i, v_j)$. Then $M := d(v_s, v_t) + 1$ fulfills the desired property. In the parametric version, where all weights on the edges are dependent on λ, we need another method to find M.

Lemma 4.2 Let $G = (V, E)$ be a graph, where the weights on the edges are given by $\ell_\lambda(e) = a(e) + \lambda \cdot b(e), \lambda \in \Lambda = [\lambda_0, \infty)$. Let P_1 be a shortest v_i-v_j-path for the interval $[\lambda_0, \lambda_1]$ with edge set $E(P_1)$, for $v_i, v_j \in V$. Let $r = \sum_{e \in E(P_1)} b(e)$ be the corresponding gradient of the optimal value function $d_\lambda(P_1) = \sum_{e \in E(P_1)} a(e) + r \cdot \lambda$. Then, an upper bound on the shortest v_i-v_j-path for all λ can be found as follows:

$$M_\lambda(v_i, v_j) = \sum_{e \in E(P_1)} a(e) + r \cdot \lambda, \quad \varepsilon > 0.$$

Proof. Let P_l be the shortest v_i-v_j-path for the interval $[\lambda_{l-1}, \lambda_l], l > 1$ with edge set $E(P_l)$. Also, let $r_l := \sum_{e \in E(P_l)} b(e)$. We know, that the optimal value function of the shortest path problem with weights linearly dependent on λ, is piecewise linear and concave [Car83b]. The objective function values $d_\lambda(P_l)$ are of the form $d_\lambda(P_l) = \sum_{e \in E(P_l)} a(e) + r_l \cdot \lambda$. Due to concavity, it is $r = r_1 > r_l$ for all $l > 1$.

Also due to concavity, it is clear, that for every λ, it is $d_\lambda(P_1) \geq d_\lambda(P_l)$ for all $l > 1$. Therefore, $d_\lambda(P_1)$ serves as an upper bound for the objective function values of the shortest v_i-v_j-path. For the first interval, this is a tight bound. Figure 4.1 illustrates the lemma. ■

Remark 4.3 Note, that the path P_1 corresponds to the path, that is found when searching for the shortest v_i-v_j-path with fixed $\lambda = \lambda_0$.

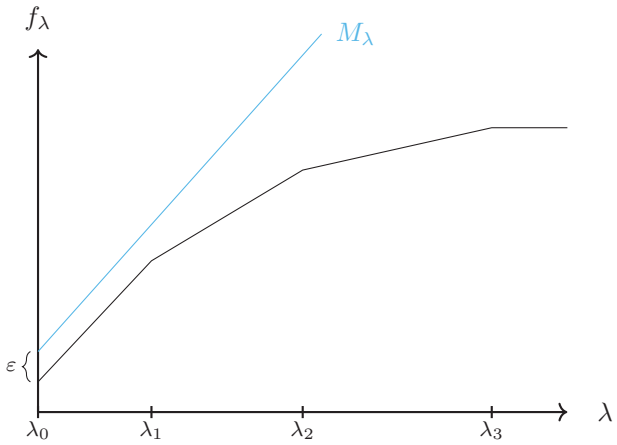


Figure 4.1: Illustration of Lemma 4.2.

For the proof of Theorem 4.1 and also the subsequent Theorem 4.6, we need a bound M_λ that is not only larger than the shortest v_i-v_j-path, but bigger than the shortest paths between a feasible center and all other vertices in the graph. This can be found using Lemma 4.4.

Lemma 4.4 Let $G = (V, E)$ be a graph, where the weights on the edges are given by $\ell_\lambda(e) = a(e)+\lambda \cdot b(e), \lambda \in \Lambda = [\lambda_0, \infty)$. Let $V_X \subseteq V$ be the subset, that contains all vertices, from which every other vertex in V is reachable. Then, for $\lambda = \lambda_0$, run an SSSP-algorithm for all $v_i \in V_X$ on graph G to obtain shortest v_i-v_j-paths P_{ij}, each of them containing edges $E(P_{ij})$. Let the corresponding objective function values be $d_\lambda(P_{ij}) = t_{ij} + r_l \cdot \lambda_0$, with $r_{ij} := \sum_{e \in E(P_{ij})} b(e)$ and $t_l := \sum_{e \in E(P_{ij})} a(e)$. Let $r := \max_{i,j} r_{ij}$. Furthermore, let $t := \max_{i,j} t_{ij}$. Then, an upper bound on the objective function values of the parametric shortest paths between every pair of vertices can be found as follows:

$$M_\lambda = t + r \cdot \lambda.$$

Proof. From Lemma 4.2 we know, that each v_i-v_j-path P_{ij} itself can be bounded by $M_\lambda(P_{ij})$ as defined there. In Remark 4.3 we already stated, that for finding these bounds, the shortest paths for $\lambda = \lambda_0$ need to be calculated. To find an upper bound on all linear functions $M_\lambda(P_{ij})$, let r and t be as stated. Since r is the biggest gradient of the functions and t is the biggest intersection with the ordinate axis found over all functions, $t + r \cdot \lambda$ is at least a tight bound on the functions $M_\lambda(P_{ij})$. ■

Remark 4.5 To bound weighted distances $w_j d_\lambda(v_i, v_j)$, $w > 0$, calculate M'_λ as described in Lemma 4.4. Then, let $w_{\max} := \max_i w_i$. An upper bound on the weighted distances is $M_\lambda := w_{\max} \cdot M'_\lambda$.

With the help of the lemmata above, we can finally prove Theorem 4.1.

Proof of Theorem 4.1. Consider the graph Mulmuley and Shah used, for which we know that the minimum value function for the shortest s-t-path has $2^{\Omega(\log^2 n)}$ many breakpoints, see Theo-

rem 3.7. We construct the following graph, see Figure 4.2, by adding a vertex t' and connecting it to t with weight M_λ, which can be found as stated in Lemma 4.4. Recall from Section 3.1,

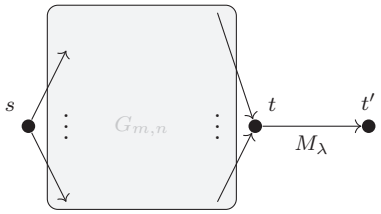


Figure 4.2: Graph G, using the construction of [MS00].

that vertices in one layer only have successors in following layers. This means, that vertex s is the only vertex reaching all other vertices in $G_{m,n}$ and also t'. Thus, vertex s is the only possible (and optimal) center for every $\lambda \in \Lambda$. Recall that M_λ is bigger than the distance between s and any other vertex in graph $G_{m,n}$. That is, for $v_i \in V(G_{m,n})$, there is no s-v_i-path whose distance $d_\lambda(s, v_i)$ is bigger than $d_\lambda(s, t') = d_\lambda(s, t) + M_\lambda$. Thus, the latter is relevant for the optimal value function of the center problem for every $\lambda \in \Lambda$. We know that $d_\lambda(s, t)$ has super-polynomial many breakpoints due to the construction of graph $G_{m,n}$. Therefore, also the optimal value function of the center problem has super-polynomially many breakpoints. ■

Now that it is clear, that the optimal value function value can have super-polynomially many breakpoints, the question arises, whether the center itself may change its position super-polynomially often in a given graph for varying λ. Indeed, the next Theorem 4.6 shows, that this might be the case.

Theorem 4.6 The position of the center of the parametric center problem can change super-polynomially often.

For the proof of this theorem, we again use an alteration of the graph of Mulmuley and Shah (see [MS00]) introduced above. Remember, that we want to minimize the maximum distance from the new location to every other vertex in the graph. We want to alter the graph of Mulmuley and Shah in such a way, that the shortest s-t-path is decisive for the placement of the center. Then, we will take advantage of the fact, that the second vertex of the s-t-path in the first layer of $G_{m,n}$ switches as described in Section 3.2. In the altered graph, we want to achieve, that these switches correspond to the (changing) placement of the center for increasing λ.

For a better understanding of the proof, we want to present its idea for fixed lambda first and deal with the parametric edge weights afterwards.

Idea of the proof of Theorem 4.6 with fixed λ. We structure the proof in two parts: first, we introduce the new built graph and then afterwards we evaluate the possible placements for the center. In the latter part, we need to calculate the shortest paths from every possible center in the graph to all other vertices to find the optimal placement.
Let λ be fixed and $G_{m,n}$ as described above for some m, n. Furthermore, let M be an upper bound on the longest distance between any pair of vertices $v_i, v_j \in V$. We choose $M := \ell_{\max} + 1$ for $\ell_{\max}$ being the biggest edge weight in the graph $G_{m,n}$, so $\ell_{\max} = \max_i \ell_i$. Note, that with

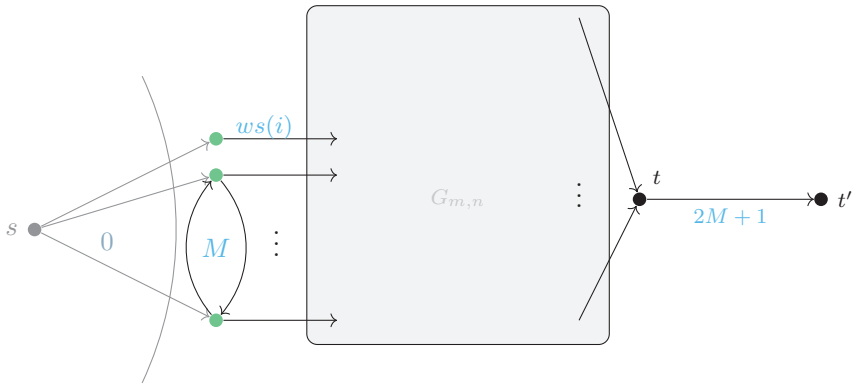


Figure 4.3: Alteration of the graph of Mulmuley and Shah, new vertices b_i depicted in green; weights are written in blue; not all edges are shown.

fixed λ, this bound can be easily calculated. Hence, we also omit it in the description of weights and distances.

Building the graph. Firstly, we introduce n new vertices $b_0, \ldots, b_{n-1}$ between vertex s and the n vertices of the first layer of the initial graph $G_{m,n}$. The new vertices are connected to s via weightless edges and to the n existing vertices with the initial weights $ws(i)$. Also, for any pair b_k, b_l, with $k, l \in \{0, \ldots, n-1\}, k \neq l$ of the newly introduced vertices, we introduce edges in both directions connecting them. Furthermore, we introduce a vertex t', that is connected to t. The weight of the edge connecting t and t' is chosen by $2M+1$. The resulting graph is shown in Figure 4.3. In the figure, it is already indicated, that we erase vertex s and its n outgoing (now) weightless edges for the proof. Note, that only two of the edges between the new vertices are depicted in the figure to avoid a confusing illustration.

Evaluation of the possible new center. Through connecting all new vertices b_i among each other, we ensure, that every one of them is a potential location for the center, because all other vertices in the graph are connected to at least one of them due to the original construction of $G_{m,n}$. More than that, none of the other vertices can be center of the graph, since none of the vertices from one specific layer are connected back to their previous layer(s) and therefore all these cannot be reached by vertices in layers further back.

Now, what remains open is the evaluation of the b_i in terms of their objective function values, say we need to calculate $f(b_i)$ for all b_i. Therefore, we first need to calculate the distances from every b_i to all other vertices in the new graph. Then, we need to extract the maximal distances among these and place the center such that the maximum distance gets minimized. The vertices to which we need to know the distances, can be parted in the vertices b_i themselves, vertices that lay in $G_{m,n}$, vertex t and vertex t'.

Before we evaluate the distances in detail, let us note the following. Since λ is fixed, we can calculate the vertex b_{i^*}, for which the distance $d(b_i, t)$ is minimal, thus $b_{i^*} = \arg\min_{i=1,\ldots,n-1} d(b_i, t)$. Let $x = d(b_{i^*}, t) = \min_{i=1,\ldots,n-1} d(b_i, t)$ be the corresponding objective function value. We find the following.

Paths to b_j: We defined the weights of the edges connecting the newly introduced vertices as

M and therefore

$$d(b_i, b_j) = M \quad \forall i, j = 1, \ldots, n-1, i \neq j$$

Paths to vertices in $G_{m,n}$: Let v_k be a vertex in $G_{m,n}$. An upper bound for the length of the shortest path from any of the vertices b_i to v_i can be evaluated as follows. There exists a vertex b_j for which $d(b_j, v_i) < M$ and therefore

$$d(b_i, v_i) \leq d(b_i, b_j) + d(b_j, v_i) < 2M.$$

Paths to t: Due to the definition of b_{i^*}, and again the fact, that there exists a vertex b_j for which $d(b_j, t) < M$ we evaluate the shortest paths to t as follows:

$$\begin{aligned} d(b_{i^*}, t) &= x \leq M \\ d(b_i, t) &\leq x + M \quad \forall i \neq i^* \\ &\leq 2M \end{aligned}$$

Paths to t': For any of the b_i, we have that

$$d(b_i, t') = d(b_i, t) + 2M + 1 \tag{4.1}$$

From the calculations above, we can see that the shortest path that is decisive for the placement of the center is the one to t', see (4.1), because it is always bigger than the distances to any other vertex in the graph. The distance to t' is minimal for the vertex b_{i^*}, as per definition, meaning that the center will be placed there. ■

Above, we fixed λ to give a general idea of the final proof of Theorem 4.6. In the parametric version, the main structure of the proof stays the same. However, an important difference is the choice of M. Before, with fixed weights, we could easily calculate M. Now, we need to use the upper bound M_λ for the pairwise shortest paths between any two vertices in the graph, that is dependent of λ that we introduced in Lemma 4.4.

Proof of Theorem 4.6. For this proof, we use the same graph as above, see Figure 4.3. Note that now all the weights of edges are dependent on λ. This holds for edges in $G_{m,n}$ (here we use the original weights) as well as the weights that formerly contained M (we now use M_λ) and the weigths $ws_\lambda(i)$ connecting the newly introduced vertices $b_i, i = \{1, \ldots, n-1\}$ with the vertices of the first layer of $G_{m,n}$ that again are the ones from the original graph of Mulmuley and Shah. We refer to the n vertices of the first layer of $G_{m,n}$ as $h_i, i = \{0, \ldots, n-1\}$, such that h_i is the successor of b_i.

With the same arguments as above, only the vertices b_i are valid candidates for the placement of the center. Remember, that since the weights on the edges are now dependent of λ, the evaluation of the distances of any b_i to all other vertices in the graph is also dependent of λ. Therefore, we review the arguments from above. Let again b_{i^*} be the vertex, for which the distance $d_\lambda(b_i, t)$ is minimal, thus $b_{i^*} = \arg\min_{i=1,\ldots,n-1} d_\lambda(b_i, t)$ for a given λ. Let $x = d_\lambda(b_{i^*}, t) = \min_{i=1,\ldots,n-1} d_\lambda(b_i, t)$ be the corresponding objective function value. Note, that b_{i^*}

is also the predecessor of that vertex in the first layer of $G_{m,n}$, that lies on the shortest path to t for that given λ, namely h_{i^*} due to the construction of the new graph. This fact is important for the evaluation of the placement of the center.

As above and with the same arguments, we evaluate the distances from b_i to b_j, to vertices in $G_{m,n}$ and to the vertices t and t':

Paths to b_j: Due to the construction it is

$$d_\lambda(b_i, b_j) = M_\lambda \quad \forall i, j = 1, \ldots, n-1, i \neq j$$

Paths to vertices in $G_{m,n}$: Let v_k be a vertex in $G_{m,n}$. Let b_j be the vertex for which $d_\lambda(b_j, v_i) < M_\lambda$ and therefore

$$d_\lambda(b_i, v_i) \leq d_\lambda(b_i, b_j) + d_\lambda(b_j, v_i) < 2M_\lambda.$$

Paths to t: Due to the definition of b_{i^*} and the fact that there is a vertex b_j for which $d_\lambda(b_j, t) < M$, we get:

$$\begin{aligned} d_\lambda(b_{i^*}, t) &= x \leq M_\lambda \\ d_\lambda(b_i, t) &\leq x + M_\lambda \quad \forall i \neq i^* \\ &\leq 2M \end{aligned}$$

Paths to t': For any of the b_i, we have that

$$d_\lambda(b_i, t') = d_\lambda(b_i, t) + 2M_\lambda + 1 \tag{4.2}$$

Again, the shortest path to vertex t' is the decisive factor for the objective function value of the corresponding b_i for all i and all λ. Therefore, to find a solution for the center problem, we need to minimize that distance given in 4.2. Clearly, $d_\lambda(b_i, t')$ is minimal, when $d_\lambda(b_i, t)$ is minimal. Since b_i is connected to h_i via the original weights $ws_\lambda(i)$, we can conclude, that $d_\lambda(b_i, t)$ is minimal for a given λ, when the path $P_\lambda(h_i, t)$ connecting h_i and t is part of the shortest s-t-path of the original graph of Mulmuley and Shah. We know, that this path changes super-polynomially often. Also the vertex h_i, that is a successor of b_i^* on the corresponding shortest path, changes whenever the shortest path changes which was introduced in Section 3.2. That in consequence means that the corresponding b_i^* that minimizes $d_\lambda(b_i, t)$, and therefore the center, changes super-polynomially often itself with increasing λ. ■

Considering the previous results, one can see the need for an approximation algorithm for the parametric center problem: an exact algorithm might have to output super-polynomially many solutions. In the following, we provide such an algorithm.

For the non-parametric center problem, the approach that evaluates all feasible solutions in terms of their objective function values and chooses the best among them, can in fact be done in $\mathcal{O}(n^3)$. This is, because computing the all-pair-shortest-paths using Floyd-Warshall, for instance, can

be done in this running time [Lei+01]. For the parametric version, however, calculating the all-pair-shortest-paths might be excessive in terms of computing, as seen before. Therefore, we use an existing approximation scheme to solve the center problem approximate. More precisely, we use an $(1+\varepsilon)$-approximation provided by Bazgan et al. [Baz+22]. As described in Section 3.1, the parametric problem, for which an approximation is sought, needs to be of the form $f_\lambda(x) = a(x) + \lambda \cdot b(x)$. Consequently, we cannot directly approximate the center problem by using a non-parametric algorithm as a subroutine. Instead, let a vertex v_i be given. We approximate the weighted single-source-shortest-paths $w_j d_\lambda(v_i, v_j)$ to all other vertices of the network. This is done by using a non-parametric SSSP as a subroutine and add the multiplicative factor w_j to every distance. Then, the upper envelope over these approximations is calculated. This leaves us with an approximation of the optimal value function of the center problem for the chosen vertex v_i. Doing this for all vertices and then calculating the lower envelope over all upper envelopes, approximates the parametric center problem.

In their work, Bazgan et al. [Baz+22] already formulate that the approximation method can be used for the parametric shortest path problem, when the functions $a(e)$ and $b(e)$ map to non-negative integers $\mathbb{N}_0$. Together with $\lambda \geq 0$, this restriction ensures, that the lengths on the edges and thus the distances $d_\lambda(v_i, v_j)$ between two vertices, that gets approximated, is non-negative, which is usually required in the context of approximation [WS11].
The method is described in the following.

Algorithm 4.7 — Approximation for $(1, \max, G)_\lambda$**.**

Input: $G = (V, E)$ with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$, vertex weights $w_i > 0$
Output: $(1+\varepsilon)$-approximation of $(1, \max, G)_\lambda$

1: **for all** $v_i \in V$ **do**
2: **for all** $v_j \in V$ **do**
3: calculate $(1+\varepsilon)$-approximation of $w_j d_\lambda(v_i, v_j)$
4: **end for**
5: calculate upper envelope of the piecewise linear functions found in line 3
6: **end for**
7: calculate lower envelope of all upper envelopes found in line 5
8: **return** the lower envelope found in line 7 and the vertices yielding its line segments

Theorem 4.8 Algorithm 4.7 results in an $(1+\varepsilon)$-approximation for the parametric center problem on directed general graphs.

Proof. The main idea of the algorithm is to evaluate all vertices that are feasible and choose the best alternative for every $\lambda \in \Lambda$. In line 3, we use the general approximation method described in [Baz+22]. In their work, they already show, how this can be done to approximate shortest paths. Therefore, what remains to be shown is that the approximation quality of the calculation also holds for the objective function value of the center problem.

Let $x \in V$ be arbitrary, but fixed. Let $f_1(\lambda), \ldots, f_n(\lambda)$ be the objective functions with $f_i(\lambda) :=$

$d_\lambda(x, v_i)$ for $v_i \in V$ and $i = \{1, \dots, n\}$. By the results of Bazgan et al., we can calculate functions $f'_1(\lambda), \dots, f'_n(\lambda)$ that are $(1+\varepsilon)$-approximate for their corresponding exact shortest path functions, that is

$$f'_i(\lambda) \leq (1+\varepsilon) f_i(\lambda). \tag{4.3}$$

Then, with $f'_{i^*}(\lambda) = \max_i\{f'_i(\lambda)\}$, the following holds:

$$\begin{aligned} \max_i\{f'_i(\lambda)\} &= f'_{i^*}(\lambda) \\ &\overset{4.3}{\leq} (1+\varepsilon) f_{i^*}(\lambda) \\ &\leq (1+\varepsilon) \max_i\{f_i(\lambda)\}, \end{aligned}$$

resulting in

$$\max_i\{f'_i(\lambda)\} \leq (1+\varepsilon) \max_i\{f_i(\lambda)\}.$$

Also, with $f_{i^*}(\lambda) = \min_i\{f_i(\lambda)\}$, it holds:

$$\begin{aligned} \min_i\{f'_i(\lambda)\} &\leq f'_{i^*}(\lambda) \\ &\overset{4.3}{\leq} (1+\varepsilon) f_{i^*}(\lambda) \\ &= (1+\varepsilon) \min_i\{f_i(\lambda)\}, \end{aligned}$$

such that

$$\min_i\{f'_i(\lambda)\} \leq (1+\varepsilon) \min_i\{f_i(\lambda)\}.$$

The first result justifies calculating the (exact) upper envelope over the approximate shortest path functions in line 5 without losing the approximation quality in this step. Also, this upper envelope consists of linear functions for which the approximation quality, and therefore (4.3) holds. Then, with the second result, calculating the lower envelope in line 7 also keeps the approximation quality. ■

In fact, Algorithm 4.7 yields an FPTAS, as the following Lemma 4.9 shows.

Lemma 4.9 Let $G = (V, E)$ be a graph with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w_i > 0$. Let C be the maximum among all values $a(e), b(e)$. Algorithm 4.7 yields an FPTAS for the parametric center problem.

Proof. Algorithm 4.7 firstly applies the approximation algorithm $n-1$ times for every one of the n vertices in the graph in order to find the shortest paths from each vertex to every other vertices. Then, for each vertex, we need to find the upper envelope of the $n-1$ shortest path approximations. Then, over all n upper envelopes, we need to calculate the lower envelope. Let T_{APP} be the running time of the approximation scheme, T_{UE} the running time for calculating the upper envelope and T_{LE} the one for the lower envelope, respectively. Then, the running time T_{AC} of Algorithm 4.7 is

$$T_{\text{AC}} = n \cdot ((n-1) \cdot T_{\text{APP}} + T_{\text{UE}}) + T_{\text{LE}}.$$

We determine the running times of all parts successively and assume that $\varepsilon < 1$.

Approximation of the shortest paths: The subroutine consists of calculating the shortest v_i-v_j-path which can be done in $\mathcal{O}(m+n\log n)$ [AMO93] and multiplying with the according weight w_j in $\mathcal{O}(1)$. For a fixed vertex v_i, this is done for all other vertices v_j.Thus, with (3.2) and the analysis in [Baz+22, Part 5], it is

$$T_{\text{APP}} \subseteq \mathcal{O}\left(\frac{1}{\varepsilon}\cdot(m+n\log n+n)\cdot\log nC\right) \subseteq \mathcal{O}\left(\frac{1}{\varepsilon}\cdot(m+n\log n)\cdot\log nC\right).$$

Complexity of the approximation of the optimal value function of the shortest paths: The complexity of the approximation is in $\mathcal{O}(\frac{1}{\varepsilon}\cdot\log nC)$ [Baz+22]. For the purpose of better readability, let $s := \frac{1}{\varepsilon}\cdot\log nC$, making it $\mathcal{O}(s)$.

Calculating the upper envelope: The upper envelope of n intersecting line segments can be computed in $\mathcal{O}(n\cdot\log n)$ time [Her89]. We know that the number of line segments, that may be generated through the approximation of the shortest paths, is in $\mathcal{O}(s)$. Hence, in our case, we want to compute the upper envelope of $n-1$ many piecewise linear functions, each consisting of at most $\mathcal{O}(s)$ line segments, meaning

$$T_{\text{UE}} \subseteq \mathcal{O}(ns\cdot\log(ns)).$$

Complexity of the upper envelope: The complexity of the upper envelope of n line segments is in $\mathcal{O}(n\cdot\alpha(n))$, where $\alpha(n)$ is the functional inverse of the Ackermann function [HS86, WS88]. Therefore, the complexity of one upper envelope in our case is $\mathcal{O}(ns\cdot\alpha(ns))$.

Calculating the lower envelope: The lower envelope of n line segments can be computed in the same time as the upper envelope, so $\mathcal{O}(n\cdot\log n)$ (see [AS00]). Since we compute the lower envelope over the n upper envelopes, each with complexity $\mathcal{O}(ns\cdot\alpha(ns))$, we end up with a computational time for the lower envelope of

$$T_{\text{LE}} \subseteq \mathcal{O}((n^2s\cdot\alpha(ns))\cdot\log(n^2s\cdot\alpha(ns))).$$

Putting the information together, we get a running time for Algorithm 4.7 of

$$\begin{aligned} T_{\text{C}} = \quad & n\cdot((n-1)\cdot T_{\text{APP}} \quad + T_{\text{UE}}) \quad + T_{\text{LE}} \\ \subseteq\ & \mathcal{O}(n\cdot(ns\cdot(m+n\log n)+ns\cdot\log(ns)) + (n^2s\cdot\alpha(ns))\cdot\log(n^2s\cdot\alpha(ns))) \\ =\ & \mathcal{O}(n^2s\cdot(m+n\log n+\log(ns)+\alpha(ns)\cdot\log(n^2s\cdot\alpha(ns)))) \end{aligned}$$

Before resubstituting s, we want to simplify the term above. We use basic rules for logarithms

and also $\alpha(ns) \leq ns$, $\alpha(ns) \leq \log(ns)$ as well as $m \leq n^2$.

$$
\begin{aligned}
&\mathcal{O}(n^2 s \cdot (m + n \log n + \log(ns) + \alpha(ns) \cdot \log(n^2 s \cdot \alpha(ns))))\\
&= \mathcal{O}(n^2 s \cdot (m + n \log n + \log(ns) + \alpha(ns) \cdot \log n + \alpha(ns) \cdot \log s + \alpha(ns) \cdot \log(\alpha(ns)))\\
&\subseteq \mathcal{O}(n^2 s \cdot (m + n \log n + \log(ns) + \alpha(ns) \cdot \log n + \alpha(ns) \cdot \log s + \alpha(ns) \cdot \log n + \alpha(ns) \cdot \log s)\\
&= \mathcal{O}(n^2 s \cdot (m + n \log n + \log(ns) + \alpha(ns) \cdot \log n + \alpha(ns) \cdot \log s)\\
&\subseteq \mathcal{O}(n^2 s \cdot (n^2 + \log(ns) + \alpha(ns) \cdot \log(ns)))\\
&= \mathcal{O}(n^2 s \cdot (n^2 + \alpha(ns) \cdot \log(ns)))\\
&\subseteq \mathcal{O}(n^2 s \cdot (n^2 + \log^2 s))
\end{aligned}
$$

where the last line is due to the fact that

$$\mathcal{O}(\alpha(ns) \log(ns)) \in \mathcal{O}(\log^2(ns)) = \mathcal{O}(\log^2 n + \log n \log s + \log^2 s)$$

and the fact that $(\log n \log s)$ is either at most $\log^2 n$ or $\log^2 s$. Resubstituting $s = \frac{1}{\varepsilon} \cdot \log nC$ gives us a running time of

$$\mathcal{O}\left(n^2 \cdot \frac{1}{\varepsilon} \cdot \log nC \cdot \left(n^2 + \log^2\left(\frac{1}{\varepsilon} \cdot \log nC\right)\right)\right)$$

which proves our claim. ■

Remark 4.10 As of now, no results are known about the complexity of the optimal value function or the solution of the parametric center problem on undirected graphs. However, we can still use Algorithm 4.7 to approximate this problem. This is, because the algorithm used for the subroutine can be used on undirected graphs in the same running time.

4.2 Trees

In the remainder of this section, we discuss the parametric center problem on tree graphs $T = (V, E)$ (see Section 2.2). Therefore, we again distinguish between directed and undirected trees.

4.2.1 Directed Trees

If we recall the center problem, we find that only those vertices are feasible, from which all other vertices can be reached. For a directed tree, this means that there is only one feasible solution, if we are faced with a rooted out-tree.

Proposition 4.11 The parametric center problem on a directed tree $T = (V, E)$ is optimally solvable, if the tree is a rooted out-tree with single optimal solution $r \in V$ being the root for all λ.

Proof. Let the tree be rooted in $r \in V$. Per definition, the root is the vertex, from which every other vertex $v_i \in V$ can be reached. This root is unique. Furthermore, if it only has descendants

– meaning the tree is a rooted out-tree – it is the only vertex fulfilling feasibility for the center problem and thus it is the single optimal solution. ■

Remark 4.12 We can find out, if a tree is a rooted out-tree by checking whether there is a single vertex, that has an in-degree of 0. If the representation of the tree is given with an adjacency list, we can check the in-degrees in $\mathcal{O}(m+n)$. Note, that connectivity is given through the definition of a tree.

As to analyze the complexity of the objective function, we use the fact, that every shortest r-v-path-path, for $r \in V$ being the root and $v \neq r \in V$ being any other vertex, can enter the optimal value function only once.

Proposition 4.13 If $T = (V, E)$ is a rooted out-tree, the objective function of the parametric center problem has at most $n - 2$ breakpoints.

Proof. From Proposition 4.11 it is known, that the root $r \in V$ is the only feasible, and thus optimal solution for all $\lambda \in \Lambda$. Recall that the objective function of the parametric center problem is $\max_{v_i \in V} w_i d_\lambda(r, v_i)$ for all λ. The distances $d_\lambda(r, v)$ are linear functions and for every λ, there is a path defining the objective function. Let v_i be the vertex, such that $w_i d_\lambda(r, v_i)$ is maximal for an interval $[\lambda_r, \lambda_s]$. Also, let v_j be the vertex for which $w_j d_\lambda(r, v_j)$ is maximal for an interval starting at λ_s. That means, there is an intersection of $w_i d_\lambda(r, v_i)$ and $w_j d_\lambda(r, v_j)$ at λ_s. Then for every $\lambda > \lambda_s$, it is $w_i d_\lambda(r, v_i) < w_j d_\lambda(r, v_j)$ due to the linearity of $d_\lambda(r, v)$. Thus, every path occurs in the optimal value function of the center problem only once. Since there is a total of $n - 1$ paths from r to every other vertex, the number of breakpoints in the objective function value is bounded by $n - 2$. ■

The observations above lead to the conclusion, that parametric center problems on directed trees are not of particular interest, since they are only feasible for a very special case.

4.2.2 Undirected Trees

This section deals with the parametric center problem on undirected trees $T = (V, E)$. Analogously as for the directed case, the fact that paths are unique in trees, simplifies the parametric center problem in comparison to the problem on general graphs and therefore the answer to the three questions stated in Section 3.1. However, unlike in the case of directed trees, in general there is more than one feasible solution in the undirected case.

Still, an exact algorithm that has polynomial running time can be formulated. For every vertex v_i, the shortest paths $d_\lambda(v_i, v_j)$ to all other vertices v_j are computed and multiplied with the corresponding vertex weight w_j. The distances $d_\lambda(v_i, v_j)$ can be found by fixing λ and solving the non-parametric problem to generate the edges of the path. Then, for every vertex, the upper envelope is calculated over the objective functions of the corresponding shortest paths. Finally, the lower envelope over all upper envelopes solves the parametric center problem. Algorithm 4.14 states the procedure.

Algorithm 4.14 — Solution of $(1, \max, T)_\lambda$.

Input: $T = (V, E)$ with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w_i > 0$

Output: Set of optimal vertices for $(1, \max, T)_\lambda$

1: **for all** $v_i \in V$ **do**
2: **for all** $v_j \in V$ **do**
3: calculate $w_j d_\lambda(v_i, v_j)$
4: **end for**
5: compute upper envelope of the linear functions found in line 3
6: **end for**
7: calculate lower envelope of all upper envelopes found in Line 5
8: **return** the lower envelope found in Line 7 and the vertices yielding the line segments

The stated algorithm indeed solves the parametric center problem, as shown in the following Theorem 4.15.

Theorem 4.15 Algorithm 4.14 solves the parametric center problem on undirected trees.

Proof. Since the algorithm evaluates all vertices in terms of their corresponding objective function values and chooses the best among them for all $\lambda \in \Lambda$, it outputs a set of optimal vertices for the parametric center problem dependent on the current λ. ■

As mentioned in the introduction, the stated algorithm solves the problem in polynomial time, as the next Lemma 4.16 shows.

Lemma 4.16 Algorithm 4.14 solves the parametric center problem on undirected trees in $\mathcal{O}(n^2 \log n)$.

Proof. The overall running time T of Algorithm 4.14 depends on the computation of the parametric single source shortest paths (T_{SP}) together with the weight-multiplication, calculating the upper envelope over them for each vertex individually (T_{UE}), and the running time for the lower envelope over all upper envelopes (T_{LE}), so

$$T = n \cdot (T_{\text{SP}} + T_{\text{UE}}) + T_{\text{LE}}.$$

Since the parametric paths are unique, they can be found with breadth-first search for fixed λ in $\mathcal{O}(n)$ for trees. Multiplication is done in $\mathcal{O}(1)$, so

$$T_{\text{SP}} \subseteq \mathcal{O}(n).$$

As a result, there are $n - 1$ linear functions $d_\lambda(v_i, v_j)$, over which the upper envelope can be calculated in

$$T_{\text{UE}} \subseteq \mathcal{O}(n \log n).$$

Each of the upper envelope consists of at most $\mathcal{O}(n)$ line segments. This is because of the following. Let a linear function $d_\lambda(v_i, v_{j_1})$ be maximal among all distances for an interval $[\lambda \in \Lambda_r, \lambda_s]$. Let $d_\lambda(v_i, v_{j_2})$ be the linear function it intersects with at λ_s, thus $d_\lambda(v_i, v_{j_2}) > d_\lambda(v_i, v_{j_1})$ for $\lambda > \lambda_s$. Due to linearity, the function $d_\lambda(v_i, v_{j_1})$ is not part of the upper envelope for all $\lambda > \lambda_s$. Considering the fact, that the lower envelope is computed over all n upper envelopes, thus over $\mathcal{O}(n^2)$ line segments, the running time for this step is

$$T_{\text{LE}} \subseteq \mathcal{O}(n^2 \log n^2),$$

see [AS00]. The overall running time then is

$$\begin{aligned} T &\subseteq \mathcal{O}(n \cdot (n + n \log n) + n^2 \log n^2) \\ &\subseteq \mathcal{O}(n^2 \cdot (\log n + \log n^2)) \\ &\subseteq \mathcal{O}(n^2 \log n). \end{aligned}$$

■

The number of breakpoints of the objective function as well as the number of switches of the optimal location of the center is bounded by the number of total paths in the tree, as it is shown in the next two lemmata.

Lemma 4.17 The number of breakpoints in the objective function of the parametric center problem on undirected trees is bounded by $n(n-1)/2$.

Proof. Let $T = (V, E)$ be an undirected tree with $|V| = n$. Then, there are no more than $n(n-1)/2$ different shortest paths P in the tree since between any two vertices v_i, v_j, the corresponding shortest path is unique. Also, the distances $d_\lambda(v_i, v_j)$ are linear functions. Therefore, once a shortest path is no longer minimal for some λ, it will also never again become optimal, hence no objective function value of a shortest path contributes to the lower envelope more than once. ■

Lemma 4.18 The number of switches of the center location is bounded by $n(n-1)/2$.

Proof. Clearly, for every $\lambda \in \Lambda$, the center of the tree is to be found on the path P_{max} for which it is $P_{max} = \arg\max_l w_j d_\lambda(P_l)$. Since there are only $n(n-1)/2$ different shortest paths in the tree, the number of switches in the location of the center is also bounded by $n(n-1)/2$. ■

5 Parametric Median Problems

Having dealt with parametric center problems in the previous chapter, this chapter analyzes the parametric median problem. From Section 2.4 we recall, that the (non-parametric) median location problem is defined as minimizing $f(x)$ with $f(x) = \sum_{i,x} w_i d(v_i, x)$. With parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w_i > 0$, the parametric median problem can be formulated as follows:

Parametric median location problem $(1, G, \sum)_\lambda$

INSTANCE: Graph $G = (V, E)$, edge weights $\ell_\lambda = a(e) + \lambda b(e)$, such that $a(e), b(e) \colon E \to \mathbb{R}$, a parameter set $\lambda \in \Lambda$ for $\Lambda \subseteq \mathbb{R}$, vertex weights $w_i > 0$

TASK: For every $\lambda \in \Lambda$, find an optimal location $x^*_\lambda \subseteq V$ such that the objective function of the median location problem is minimal, i.e. minimize

$$\sum_i w_i d_\lambda(x, v_i)$$

The 1-median problem was motivated by Hakimi [Hak64] as a location for a "switching center" in a communication network. In this work, he provided foundation for this problem. First he points out, that the vertex restricted case on general graphs with vertex weights can be solved exactly by computing the all-pair-shortest-paths and summing the retrieved weighted distances for each vertex. Evaluating over the minimum of the summed distances solves the problem. The more notable result, however, is that an absolute median can always be found on the vertices of the graph. On tree structures, the computation of the 1-median can be done by an algorithm of Goldman [Gol71] in $\mathcal{O}(n)$. Surprisingly, this method never takes the edge weights into account but rather computes the median by only comparing vertex weights. We introduce this algorithm later in Section 4.2.2, as it is used for the corresponding parametric problem as well. An introduction to the median problem can for instance be found in [LNG15]. An overview over the p-median problem is given in Section 6.1, where we analyze this problem in the context of interdiction problems.

As we did for the center problem, we also divide this chapter in two parts. The first Section 5.1 is concerned with the problem on directed general graphs while Section 5.2 deals with trees.

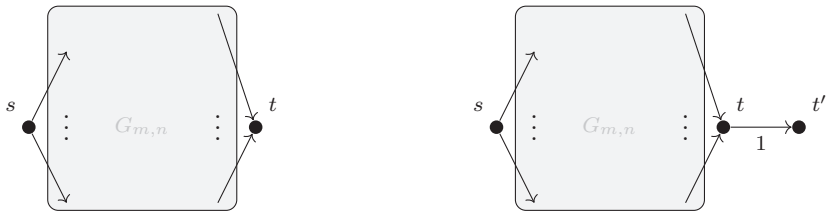
5.1 Directed General Graphs

In this section, let $G = (V, E)$ be a directed graph with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w_i > 0$. While we cannot give a result for the complexity of the

solution of the problem as of now, we can still answer the question about the complexity of the objective function values. Again, we want to use the graph of Mulmuley and Shah, introduced in Section 3.2.

Theorem 5.1 The optimal value function of the parametric median problem can have super-polynomial many breakpoints.

Proof. Consider the graph G with core $G_{m,n}$ of Mulmuley and Shah and it's alteration G' shown in Figure 5.1. Note that Subfigure 5.1a shows exactly the graph that is constructed in [MS00] and introduced in Section 3.2, while the graph G' in Subfigure 5.1b is a slight variant. Here, we introduce a vertex t' that is a successor of vertex t with constant edge weight. Vertex s is



(a) Graph G of [MS00]. (b) Graph G'; alteration of graph G of [MS00].

Figure 5.1: Graphs used in the proof of Theorem 5.1.

the only vertex that is feasible for the median location in both graphs G and G'. This is due to the construction of $G_{m,n}$ and the fact, that every vertex of a layer only has successors in following layers, never in previous ones. Thus, s is the optimal location for every λ. Recall that the objective function value on G is of the form

$$f_{\lambda,G}(s) = \sum_{i,s} d_\lambda(s, v_i) + d_\lambda(s, t)$$

for v_i being the vertices of the core of $G_{m,n}$. On G' it is

$$f_{\lambda,G'}(s) = \sum_{i,s} d_\lambda(s, v_i) + d_\lambda(s, t) + d_\lambda(s, t') = f_{\lambda,G}(s) + d_\lambda(s, t').$$

In particular, for every λ, the s-t-path having super-polynomial many breakpoints, is part of the sum once for G or twice for G', respectively.

Let λ' be a breakpoint of $d_\lambda(s, t)$ in graph G. Assume, that λ' is not a breakpoint in the objective function $f_{\lambda,G}(s)$ for graph G, meaning at λ' the objective function is linear. Then, λ' is a breakpoint of $f_{\lambda,G'}(s)$ in graph G'. This is due to the fact, that λ' is a breakpoint of $d_\lambda(s, t')$. Since $f_{\lambda,G'}(s) = f_{\lambda,G}(s) + d_\lambda(s, t')$, adding $d_\lambda(s, t')$ results in a breakpoint at λ'. We can conclude, that either λ' is a breakpoint of the objective function value in G or G'. This fact holds for every of the $2^{\Omega(\log^2 n)}$ breakpoints of $d_\lambda(s, t)$. Consequently, not less than half of the breakpoints occur in either G or G'. Therefore, on one of the graphs, the objective function has at least $2^{\Omega(\log^2 n)-1}$ breakpoints, which proves our claim. ■

Since the computation of the objective function values might be excessive, we introduce an

approximation scheme similar to the one presented for parametric center problems on directed graphs in Section 4.1. We keep the idea of dividing the problem into smaller subproblems, that we approximate. Out of these, we then build the approximate solution for the initial problem in a second step. Again, the idea is to evaluate all feasible vertices but use the $(1+\varepsilon)$-approximation introduced by Bazgan et al. [Baz+22]. As opposed to the algorithm for the center problem, we do not only approximate the shortest path between two vertices. Instead, for every vertex $v_i \in V$, we run the (non-parametric) SSSP-algorithm as a subroutine for the approximation, multiply with weights and sum all found paths. That is, we calculate the median objective for a fixed vertex and a fixed λ as a subroutine. This is possible, since the median objective fulfills the assumptions for the problem to be approximated (recall Section 3.1). Thus, we gain an approximation on the objective function value for the parametric median problem for every vertex v. Then, calculating the exact lower envelope over these n piecewise functions and evaluating the vertices yielding the line segments solves the problem approximately. As before, we need to ensure that the objective function values are non-negative for the approximation part of the described routine. Therefore, we restrict the problem in the same manner as for the center problem in Section 4.1. Let $a(e)$ and $b(e)$ functions that map to non-negative integers $\mathbb{N}_0$. Then, for all $\lambda \geq 0$, the distances $d_\lambda(v_i, v_j)$ between two vertices are non-negative and therefore also the median objective value, which is the sum of distances d_λ. Algorithm 5.2 shows the method.

Algorithm 5.2 — Approximation for $(1, \sum, G)_\lambda$.

Input: $G = (V, E)$ with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w_i > 0$

Output: $(1+\varepsilon)$-approximation of $(1, G, \sum)_\lambda$

1: **for all** $v_i \in V$ **do**
2: Compute $(1+\varepsilon)$-approximation of $\sum_j w_j d_\lambda(v_i, v_j)$
3: **end for**
4: Calculate lower envelope of all sums found in Line 2
5: **return** the lower envelope found in Line 4 and the vertices yielding its line segments

Theorem 5.3 Algorithm 5.2 results in an $(1+\varepsilon)$-approximation for the parametric median problem on directed general graphs.

Proof. Let $x \in V$ be arbitrary, but fixed. In the considered problem, we want to approximate the median objective function at x. Recall, that the edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ are linearly dependent of λ. Therefore, as mentioned before, the distances $d_\lambda(x, u)$ of the corresponding paths P_{xu} for $u \in V$ are linear functions as well and of the form $d_\lambda(x, u) = \sum_{e \in E(P_{xu})}(a(e) + \lambda b(e))$. For computing the median objective value, $\sum_{u \in V} w_u d_\lambda(x, u)$ needs to be calculated. As it is the sum of linear functions, it is a linear function as well. Therefore, the parametric median problem that we want to approximate, satisfies the required assumptions. Consequently, we can approximate the objective function of the median problem for a fixed vertex with the approximation scheme

stated in [Baz+22]. Let $f_i'(\lambda)$ be the approximation of $f_i(\lambda)$. Then, for these functions it is

$$f_i'(\lambda) \leq (1+\varepsilon) f_i(\lambda). \tag{5.1}$$

The approximate median objective function is computed for all vertices and through the evaluation of the lower envelope the best among them is chosen for every λ. It remains to be shown that the approximation quality is still valid after the computation of the exact lower envelope over the approximated functions. Let $f_{i^*}(\lambda) := \min_i\{f_i(\lambda)\}$. Then, it holds:

$$\begin{aligned}\min_i\{f_i'(\lambda)\} &\leq f_{i^*}'(\lambda) \\ &\overset{5.1}{\leq} (1+\varepsilon) f_{i^*}(\lambda) \\ &= (1+\varepsilon) \min_i\{f_i(\lambda)\},\end{aligned}$$

such that

$$\min_i\{f_i'(\lambda)\} \leq (1+\varepsilon) \min_i\{f_i(\lambda)\}.$$

Thus, calculating the lower envelope in line 4 keeps the approximation quality. ■

Theorem 5.4 Let $G = (V, E)$ be a graph with parametric edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$ and vertex weights $w_i > 0$. Let C be the maximum among all values $a(e), b(e)$. Algorithm 4.7 yields an FPTAS for the parametric median problem.

Proof. To analyze the running time T_{AM} for Algorithm 5.2, we have to analyze the running time T_{APP} for approximating the median objective values separately for all $v \in V$ and the running time T_{LE} for computing the lower envelope. Thus, the running time of Algorithm 5.2 is

$$T_{\text{AM}} = n \cdot T_{\text{APP}} + T_{\text{LE}}.$$

We determine the running times of all parts successively, for which we assume that $\varepsilon < 1$.

Approximation of the median objective values: From (3.2) we know that the running time for the approximation is in

$$\mathcal{O}\left(T_{\text{UB/LB}} + T_{\text{ALG}}\left(\frac{1}{\varepsilon}\log\frac{\text{UB}}{\text{LB}}\right)\right),$$

where $T_{\text{UB/LB}}$ is the time to compute upper and lower bounds on the objective function and T_{ALG} is the time to run the subroutine for fixed λ.

An upper bound can be found in polynomial time with calculating $2(n-1)^2C$ in $\mathcal{O}(m)$. This is, because the distance between two vertices can be no more than $2(n-1)C$, thus the sum over all distances to all other vertices is bounded by $2(n-1)^2C$ which is in $\mathcal{O}(n^2C)$. The lower bound can be set to 1.

As for T_{APP}, in our case, where we calculate the median objective function for a fixed vertex at a fixed λ, this is calculating the SSSP, multiplying with the according weights and

summing up the n received paths. Calculating the SSSP is in $\mathcal{O}(m + n \log n)$, multiplying and calculating the sum is in $\mathcal{O}(n)$.

Therefore, the approximation part of Algorithm 5.2 can be done in

$$\begin{aligned} T_{\text{APP}} &\subseteq \mathcal{O}\left(m + \frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log n^2 C\right) \\ &\subseteq \mathcal{O}\left(\frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log nC\right) \end{aligned}$$

Complexity of the approximation of the optimal value function: The complexity of one approximation is in $\mathcal{O}\left(\frac{1}{\varepsilon} \cdot (\log(\text{UB}))\right)$ (see [Baz+22]). For the considered case, this is $\mathcal{O}(1/\varepsilon \cdot \log nC)$. For better readability, let again $s := 1/\varepsilon \cdot \log nC$, thus the complexity of the approximation of the median objective for one vertex is in $\mathcal{O}(s)$.

Calculating the lower envelope: The lower envelope of n line segments can be computed in $\mathcal{O}(n \log n)$ time (see [Her89]). Thus, the lower envelope of the n approximate median objectives can be computed in

$$T_{\text{LE}} = \mathcal{O}(ns \log ns).$$

Combining the information and using rather rough estimations, we can prove our claim:

$$\begin{aligned} T_{\text{AM}} &= n \cdot T_{\text{APP}} + T_{\text{LE}} \\ &= \mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log nC + ns \log ns\right) \\ &\subseteq \mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot (n^2) \cdot \log nC + n^2 s^2\right) \\ &= \mathcal{O}\left(n^3 \cdot \frac{1}{\varepsilon} \log nC + n^2 s^2\right) \\ &= \mathcal{O}\left(n^3 \cdot \frac{1}{\varepsilon} \log nC + n^2 \left(\left(\frac{1}{\varepsilon}\right)^2 \log^2 nC\right)\right) \end{aligned}$$

■

Observation 5.1 Instead of dividing the problem in smaller subproblems, the approximation algorithm of Bazgan et al. [Baz+22] can also approximate the parametric median problem directly by using a non-parametric algorithm as a subroutine that solves the 1-median problem on the given network. That is because the objective function is of the required form, as seen before, and we make the same assumptions for $a(e), b(e)$ and λ. Furthermore, there is an exact algorithm available that solves the 1-median problem on a network in $\mathcal{O}(n(m + n \log n))$ by enumeration.

As a reminder, in Algorithm 5.2 we approximated the n optimal value functions for all vertices separately and afterwards computed the lower envelope to minimize over all possible solutions. Integrating the minimization step directly into the subroutine by solving the median problem over all vertices directly saves the running time needed to calculate the lower envelope. For this observation, recall the running time provided in [Baz+22], see (3.2).

For $s = 1/\varepsilon \cdot \log nC$, as before, the running time for Algorithm 5.2 is in

$$\mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log nC + ns \log ns\right)$$

where the term $(ns \log ns)$ is the time needed to calculate the lower envelope, see proof of Theorem 5.4. The running time for the algorithm that approximates the median problem directly, is in

$$\mathcal{O}\left(\frac{1}{\varepsilon} \cdot (n \cdot (m + n \log n)) \cdot \log nC\right).$$

However, we are able to utilize some handy properties of the approximation algorithm that allows to get a faster running time to compute the lower envelope. Thus, we are able to improve the overall running time of Algorithm 5.2. We state our improved result in the following.

Theorem 5.5 The running time of Algorithm 5.2 is in $\mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log nC\right)$.

Before we prove Theorem 5.5, we recall Theorem 2.56, which is used in the following analysis: if the n line segments are sorted in slope and sorted in the endpoints of their x-coordinate, the upper (lower) envelope can be calculated in $\mathcal{O}(n \log \log n)$ [CW02].

Proof. As seen before, the running time of Algorithm 5.2 is

$$T_{\mathrm{AM}} = \mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log nC\right) + T_{\mathrm{LE}}$$

where T_{LE} is the time to compute the lower envelope. Since we are using a stack in the approximation algorithm, we are able to generate the different approximations such that they are each sorted in λ. Moreover, the corresponding line segments are also sorted in slope. This is because the approximation of the optimal value function is concave and thus the slopes are sorted in decreasing order [Baz+22]. Consequently, after the approximation step, we have n sets of s ordered elements with respect to both slope and λ. To calculate the lower envelope over all of them by using the algorithm provided by Chen and Wada [CW02], these elements need to be merged in an ordered way. This can be done by merge sort techniques in $\mathcal{O}(ns \log n)$ [Lei+01]. Therefore, the running time to compute the lower envelope is the sum of the running time to sort the elements and to apply the algorithm by Chen and Wada [CW02]:

$$T_{\mathrm{LE}} = \mathcal{O}(ns \log n + ns \log \log ns).$$

Since we may assume that $n, s \geq 2$. Then it is $\log \log ns = \log(\log n + \log s) \leq \log(\log n \cdot \log s)$. Furthermore, from [Baz+22] we know that $s \in \mathcal{O}(1/\varepsilon \cdot \log nC)$. Additionally, since s is the number of segments in one approximation, s is also bounded by n^m. That is, because for each

edge we count the number of paths where the edge may occur. Therefore,

$$\begin{aligned} T_{\mathrm{LE}} &= \mathcal{O}(ns \log n + ns \log \log ns) \\ &\subseteq \mathcal{O}(ns(\log n + \log(\log n \cdot \log s))) \\ &\subseteq \mathcal{O}(ns(\log n + \log(\log n \cdot \log n^m))) \\ &\subseteq \mathcal{O}(ns(\log n + \log \log n + \log(m \log n))) \\ &\subseteq \mathcal{O}(ns \log n) = \mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot \log nC\right). \end{aligned}$$

Thus, the running time of Algorithm 5.2 is

$$T_{\mathrm{AM}} = \mathcal{O}\left(n \cdot \frac{1}{\varepsilon} \cdot (m + n \log n) \cdot \log nC\right).$$

■

We proved that the running time of both variants, the subdividing-approach as well as the direct approximation, actually have the same theoretical running time and also approximation guarantee. However, there might be advantages of the method described in Algorithm 5.2. When subdividing the problem and approximating every subproblem individually, we might be able to gain more information in the sense of a finer division of Λ together with the approximate solutions. Thus, although the theoretical approximation guarantee of $(1 + \varepsilon)$ is the same, in a computational study we might be able to gain better results by running the approximation over all vertices individually and use this information to build the solution set of the problem. Furthermore, since the n approximation steps are done independently of another, they can be run in parallel. In practice, this might yield running time improvements. Again, we are aware that in a theoretical consideration, both aspects of running time and approximation guarantee are equal for both approaches.

Remark 5.6 Algorithm 5.2 can be used to approximate the parametric median problem on undirected graphs as well. On undirected graphs, the algorithm used in the subroutine works on undirected graphs in the same manner and has the same running time.

5.2 Trees

For the remainder, let $T = (V, E)$ be a tree with parametric edge weights. We distinguish between directed and unirected trees, but will see, that the case of directed trees is similarly restricting as it is for the parametric center problem.

5.2.1 Directed Trees

For the parametric median problem, most of the arguments hold, that were already stated in Section 4.2.1. Therefore, we will only give the results and refer to the mentioned section for details.

Proposition 5.7 The parametric median problem on a directed tree $T = (V, E)$ is optimally

solvable, if the tree is a rooted out-tree with single optimal solution $r \in V$ being the root for all λ.

Remark 5.8 As already stated in Remark 4.12, we can check whether the tree is a rooted out-tree in $\mathcal{O}(m+n)$.

Proof. See Proposition 4.11 ■

As for the complexity of the objective function, we use the fact, that the objective function values are linearly dependent on λ.

Proposition 5.9 If $T = (V, E)$ is a rooted out-tree, the objective function of the parametric center problem is a linear function.

Proof. From Proposition 5.7 we know, that the root $r \in V$ is the optimal solution for all $\lambda \in \Lambda$. The objective function of the parametric median problem therefore is $\sum_{v \in V} d_\lambda(r, v)$ for all λ. Since $d_\lambda(r, v)$ are linear functions, the sum is also a linear function. ■

5.2.2 Undirected Trees

For the remainder of this section, let $T = (V, E)$ be an undirected tree where the edges are equipped with weights $\ell_\lambda(e) \geq 0$ for $\lambda \geq 0$.

For non-parametric median problems on undirected trees, there is the algorithm of Goldman [Gol71], that solves the problem in $\mathcal{O}(n)$, even when vertex weights $w(v_i) = w_i$ are given for all vertices v_i. In this section, we show, that this algorithm can also be used in a parametric setup, as long as the edge weights are nonnegative. The case of negative weights on the edges is discussed later on.

Surprisingly, the mentioned algorithm runs without taking the edge weights into account. While this seems counterintuitive, we can imagine the algorithm to balance the tree in terms of the weights of its vertices. We briefly describe the procedure for the non-parametric case in the following.

Given an undirected tree $T = (V, E)$, let $W := \sum_i w_i$ be the sum over all vertex weights w_i. Then choose an arbitrary leaf v_i that has one adjacent vertex v_j. If the weight w_i equals $W/2$, return vertex v_i together with vertex v_j as optimal solutions for the median location problem. In the case that $w_i > W/2$, only vertex v_i is optimal. If none of these cases hold for the chosen leaf, update the weight w_j to $w_j + w_i$ and delete v_i and edge $e = (v_i, v_j)$. Then, choose another arbitrary leaf and iterate the process until the algorithm terminates.

As indicated in the beginning, the lengths on the edges are not relevant for the algorithm, but only the vertex weights determine the solution of the median problem. This gives hope that the algorithm can also be used for parametric undirected trees, which is in deed the case. The parametric median problem with non-negative edge weights can be solved by applying Algorithm 5.10. Note that we allow for vertex weights in this specific case.

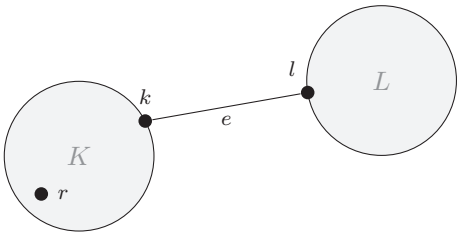


Figure 5.2: Illustration for the proof of Theorem 5.11.

Algorithm 5.10 — Solution of $(1, \sum, T)_\lambda$ for positive edge weights.

Input: $T = (V, E)$ with parametric non negative edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$, vertex weights $w(v_i) = w_i, i = \{1, \ldots, n\}$

Output: Set of optimal vertices for $(1, \sum, T)_\lambda$

$W \leftarrow \sum_i w_i$
while $\|V\| \geq 2$ **do**
  choose leaf v_i
  identify unique edge (v_i, v_j)
  if $w_i = {}^W/_2$ **then**
    identify unique edge (v_i, v_j)
    return $\{v_i, v_j\}$
  end if
  if $w_i > {}^W/_2$ **then**
    return $\{v_i\}$
  end if
  $w_j \leftarrow w_j + w_i$
  $T = (V, E) \leftarrow T \setminus \{v_i\}$
end while
return $V = \{v\}$

Theorem 5.11 Algorithm 5.10 solves the parametric median problem on undirected trees with positive edge weights.

Parametric version of the proof of 5.11 (based on [Gol71]). For $S \subset V$, let $w(S) := \sum_{v \in S} w(v)$. Also, let $f_\lambda(x, S) := \sum_{v \in S} w(x) d_\lambda(x, v)$. Let an edge $e = (k, l)$ be given. Eliminating e from the edge set E results in two subtrees with node sets K and L, where $k \in K$ and $l \in L$, respectively. See Figure 5.2 for an illustration.
Then, two things must be proven:

1. If $w(L) \geq w(K)$ then there is at least one optimal location in L

2. If $w(L) \leq w(K)$ then the problem of finding an optimal location in V is equivalent to finding an optimal location in K with redefined vertex weights $w'(k) := w(k) + w(L)$

Claim 1 Let $r \in K$ be given. We show, that $f_\lambda(r) \geq f_\lambda(l)$.

$$\begin{aligned} f_\lambda(r) &= f_\lambda(r, L) + f_\lambda(r, K) \\ &= w(L)d_\lambda(r, l) + f_\lambda(l, L) + f_\lambda(r, K) \\ &\geq w(K)d_\lambda(r, l) + f_\lambda(l, L) + f_\lambda(r, K) \\ &= f_\lambda(l, L) + \sum_{v \in K} w(v)(d_\lambda(r, l) + d_\lambda(r, v)) \\ &\geq f_\lambda(l, L) + \sum_{v \in K} w(v)d_\lambda(v, l) \\ &= f_\lambda(l, L) + f_\lambda(l, K) = f_\lambda(l) \end{aligned}$$

Claim 2 Let again $r \in K$ be given.

$$\begin{aligned} f_\lambda(r) &= f_\lambda(r, L) + w(k)d_\lambda(r, k) + f_\lambda(r, K \setminus \{k\}) \\ &= w(L)d_\lambda(r, k) + f_\lambda(k, L) + w(k)d_\lambda(r, k) + f_\lambda(r, K \setminus \{k\}) \\ &= d_\lambda(r, k)(w(L) + w(k)) + f_\lambda(k, L) + f_\lambda(r, K \setminus \{k\}) \\ &= f'_\lambda(r) + f_\lambda(k, L) \end{aligned}$$

Since $f_\lambda(k, L)$ is constant, the claim is proven.

The first statement ensures, that searching for an optimal location in the subtree with bigger summed vertex weights is valid. The second statement shows, how this is done by adjusting the vertex weights. ■

Remark 5.12 Equivalent to the non-parametric version of this problem, either one or two vertices are optimal for the stated problem.

Theorem 5.13 Algorithm 5.10 has a running time of $\mathcal{O}(n)$.

Proof. Equivalent to the non-parametric version, every vertex gets examined at most once. ■

Theorem 5.14 The objective function of the parametric median problem on undirected trees with non negative edge weights is a linear function.

Proof. Follows directly from the fact, that paths in trees are unique, and that the lengths of these parametric paths are sums of linear functions. Then, the sums of theses lengths are linear functions as well. Together with Remark 5.12 the claim holds. ■

As described above, the algorithm of Goldman works entirely without taking the weights on the edges into account. While this might be unintuitive, conversely, it means that we can use it to solve the parametric median problem on undirected graphs like the non-parametrical problem.

Note, that Algorithm 5.10 only works, as long as the weights on the edges are positive. For illustration, see Example 5.15.

Example 5.15 Let an undirected tree $T = (V, E)$ be given as in Figure 5.3.

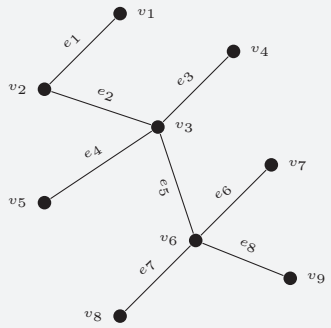


Figure 5.3: Exemplary Graph.

Let

$$\ell(e_1) = 3 + 0.7 \cdot \lambda, \quad w\ell(e_2) = 5 + 0.5 \cdot \lambda, \quad \ell(e_3) = 7 + 0.1 \cdot \lambda, \quad w\ell(e_4) = 6 + 0.4 \cdot \lambda,$$
$$\ell(e_5) = 4.5 + 0.2 \cdot \lambda, \quad \ell(e_6) = 3, \quad w\ell(e_7) = 1.5 + 0.5 \cdot \lambda, \quad \ell(e_8) = 4 + 0.5 \cdot \lambda.$$

For $\lambda \geq 0$, it is clearly $\ell(e_i) > 0$ for all i. For unit weights on the vertices, we get vertex v_3 as the center location for the parametric 1-median problem. Indeed, the optimal function values depicted in Figure 5.4 show, that v_3 is optimal for all $\lambda \geq 0$.

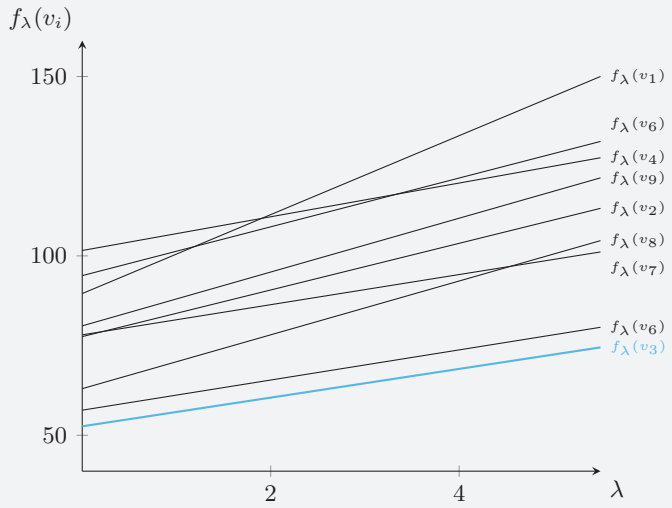


Figure 5.4: Graphs of the objective function values for vertices v_1 to v_9 for positive edge weights.

The algorithm does not work, if we allow for negative edge weights, which is shown in the next

example, where we changed the first three edge weights for the same graph in Figure 5.3:

$$\ell(e_1) = 3 - 0.7 \cdot \lambda, \quad \ell(e_2) = 5 - 0.5 \cdot \lambda, \quad \ell(e_3) = 7 - 0.1 \cdot \lambda, \quad \ell(e_4) = 6 + 0.4 \cdot \lambda,$$
$$\ell(e_5) = 4.5 + 0.2 \cdot \lambda, \quad \ell(e_6) = 3, \quad \ell(e_7) = 1.5 + 0.5 \cdot \lambda, \quad \ell(e_8) = 4 + 0.5 \cdot \lambda.$$

The objective function values are depicted in Figure 5.5. Analyzing the lower envelope shows, that there is a switch in the median location from vertex v_3 to v_1, when there corresponding objective functions intersect.

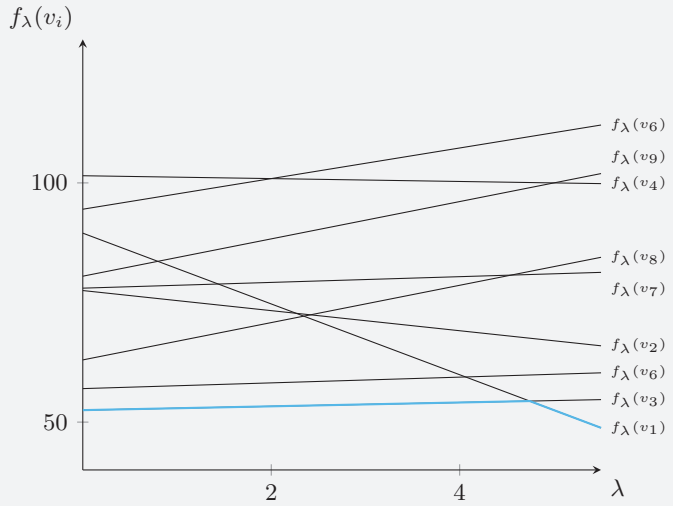


Figure 5.5: Graphs of the objective function values for vertices v_1 to v_9.

Remark 5.16 As we have stated above, the value of all parametric shortest paths in a tree are linear functions, as they are sums of linear edge weights and furthermore, they are unique. Therefore, if the parametric weights on the edges might be negative, we are still able to solve the 1-median problem exactly by calculating the APSP and evaluating the lower envelope similarly the the center problem case in Section 4.2.

Algorithm 5.17 — Solution of $(1, \sum, T)_\lambda$.

Input: $T = (V, E)$ with parametric, not necessarily non negative edge weights $\ell_\lambda(e) = a(e) + \lambda b(e)$, vertex weights $w(v_i) = w_i$

Output: Set of optimal vertices for $(1, \sum, T)_\lambda$

1: **for all** $v_i \in V$ **do**
2: **for all** $v_j \in V$ **do**
3: calculate $d_\lambda(v_i, v_j)$
4: **end for**
5: $\sum_j d_\lambda(v_i, v_j)$ for all i
6: **end for**
7: calculate lower envelope of linear functions found in line 5

8: **return** for every λ, the corresponding vertex that minimizes the lower envelope found in line 7

Theorem 5.18 The running time of 5.17 is $\mathcal{O}(n^2)$.

Proof. The running time of Algorithm 5.17 is

$$T = n \cdot (T_{SP} + T_{\text{Sum}}) + T_{LE},$$

where T_{SP} is the time needed to compute the single source shortest paths, T_{Sum} for calculating the sum in line 5 and the running time for the lower envelope T_{LE} over all objective function values for every vertex. Again, with breadth-first search for fixed λ, the single source shortest paths can be done in $\mathcal{O}(n)$ for trees, so $T_{SP} = \mathcal{O}(n)$. Summing up the resulting $n-1$ linear functions is in $\mathcal{O}(n)$, so $T_{\text{Sum}} = \mathcal{O}(n)$. Since this is done for every vertex, the lower envelope over n linear functions is computed in the last step, which is why it is $T_{LE} = \mathcal{O}(n \log n)$. The running time then is

$$\begin{aligned} T &\subseteq \mathcal{O}(n \cdot (n+n) + n \log n) \\ &\subseteq \mathcal{O}(n^2). \end{aligned}$$

■

Furthermore, we are able to give a bound for the complexity of the optimal value function and the solution itself.

Theorem 5.19 The number of breakpoints in the objective function of the parametric median problem on undirected trees with negative edge weights is in $\mathcal{O}(n)$.

Proof. For every of the n vertices $v_i \in V$, the objective function value $\sum_j d_\lambda(v_i, v_j)$ is a linear function in λ, since it is the sum of distances, that are linear functions dependent of λ as well. For these n linear functions, the number of breakpoints of their lower envelope is in $\mathcal{O}(n)$, since none of the linear functions enters the lower envelope more than once. ■

Theorem 5.20 The number of switches of the median location on parametric undirected trees is bounded by n.

Proof. As seen in Theorem 5.19, the number of breakpoints in the objective is bounded by n. With the same argument – no linear function representing the objective function value of one vertex – contributes to the lower envelope more than once, meaning that each of the n vertices is the location of the median at least once. ■

II Interdiction Location Problems

6 p-median Location Interdiction

6.1 Introducion

Interdiction problems pursue the question of how a system can be interrupted in the worst possible way in terms of the original objective function. The interruption itself can be caused by different acts, such as modification of the edge lengths or deletion of entire edges as well as the vertices. In this chapter, we analyze the p-median location interdiction problem.

The decision version of the p-median location problem is known to be NP-complete for variable p on general networks, which can be shown by a reduction from the *dominating set problem* [KH79b]. For this case, there are several heuristics known. A good overview on this topic can for example be found in [Mla+07]. For a general graph with unit length values on the edges and variable p, the problem still remains NP-complete [GJ79]. In contrast, for fixed p, the decision problem is solvable in polynomial time on general graphs by enumeration [GJ79]. Due to the hardness of the general case, research focused on particular graph structures. For G being a tree, the authors in [KH79b] present an $\mathcal{O}(n^2p^2)$-algorithm to solve the problem. A dynamic programming approach based on this result was later proposed in [Tam96], which runs in $\mathcal{O}(pn^2)$ and got improved by [BB05] to an algorithm which runs in $\mathcal{O}(n \log^{p+2} n)$. The complexity for path graphs is shown to be $\mathcal{O}(pn)$ in [HT91]. A linear time algorithm can be applied for the 1-median location problem on trees [Gol71]. An overview of the solution methods for the p-median location problem in particular can be found in [Ree06].

Interdiction problems have gained increasingly more attention lately [SS20]. There are several different applications, that motivate research in this field. The interdiction of a network can either have a desirable outcome, such as in narrowing the spread of a disease [Ass87], interdicting smuggling routes [MPS07] or – as for example done in [Woo93] – the aim of (armed) forces to reduce the amount of drugs and chemicals transported illegally via road or waterways – possibly with limited resources. Also – on the contrary – attacks on networks can be interpreted via interdiction steps. In this case, the analysis of these problems might allow the determination of valuable edges or locations of the original network.

Two main optimization problems, which have been studied in the context of interdiction, are the shortest path problem as well as the maximum flow problem. Notable research on the first topic has been done in [BGV89], [BKS98], [CD82] or [IW02], for instance. Results on the latter problem might for example be found in [GMT71], [RSL75], [Sch+20], [Wol64] or [Woo93]. In [SS20], one can find a recent overview of the literature on interdiction problems.

Research concerning the combination of location and interdiction problems on the other hand is quite scarce. In this context, we mention the r-interdiction median problem, which is defined as follows. Given a supply-system with p existing locations, the interdictor wishes to find the subset of r locations, which, when removed, yields the highest weighted distance with respect to the median location function [CSM04]. There are a few extensions to this problem, such as the possibility to fortify a fixed number of the existing facilities, which in consequence cannot be interdicted by the attacker anymore [LSD11]. In [APA10], the authors alter the concept of fortifying a specified number of locations, but rather introduce a restricted budget for fortification. A further topic, gaining more interest, is the p-hub interdiction problem, which is for example considered in [URS20]. Still, most of these approaches have in common, that the interdictor intervenes the existing locations and not the underlying network. An exception to this concept can be found in [FR21]. Here, the authors combine the covering problem with an edge interdiction problem. In [FR22], the authors present a polynomial time algorithm for the interdiction problem on trees, where an upfront chosen set of facilities is given and the interdictor wishes to worsen the reachability within the tree. In [Frö21], the authors combine the median location problem with edge interdiction. To the best of our knowledge, this is the only work on the p-median interdiction problem with edge interdiction. There, they consider the problem for different orders of action of the locator and the interdictor. For the case, that the interdictor acts before the locator, they prove this problem to be Σ_2^p-complete in the general case. In the same work, a bilevel mixed-integer formulation is presented for the problem. This result motivates the analysis of the problem for restricted cases.

In the following, we define the p-median location interdiction problem.

Decision version of the p-median location interdiction problem

INSTANCE: Undirected graph $G = (V, E)$, edge lengths $\ell\colon E \to \mathbb{Z}_+$, interdiction costs $b\colon E \to \mathbb{Z}_+$, interdiction budget $B \in \mathbb{Z}_+$, number of locations $p \in \mathbb{Z}_+$, and decision parameter $K \in \mathbb{Z}_+$.

QUESTION: Does there exist an interdiction strategy $\gamma \in \Gamma$ such that

$$\min_{X \subseteq V, |X|=p} \sum_{v \in V} d_{G(\gamma)}(X, v) \geq K \ ?$$

In the optimization version of the stated problem, we aim to find the maximum K for which the decision version is a YES-instance.

6.2 Complexity results

It is well known that the p-median location problem is NP-complete, cf. [KH79b]. Therefore it would be surprising for the corresponding interdiction problem to be polynomial time solvable. In fact Fröhlich [Frö21] proved that the decision version of the p-median location interdiction problem is Σ_2^p-complete. However, there are several restrictions of the p-median location problem which are proven to lead to polynomial time solvability, as mentioned in the introduction. Among the restricted versions, that are polynomial time solvable is the p-median location problem on

trees, as shown by Tamir [Tam96]. In this section we prove that adding the interdiction layer to the problem makes the problem significantly harder: The p-median location interdiction problem is NP-complete even on trees. For this, we consider the *knapsack problem with bounded profit ratio of 2 (K-BPR2)*. An instance is given by a set M of m items with associated weights $w_i \geq 0$ and profits $p_i \geq 0$ for all $i \in \{1, \dots, m\}$. For the profits it holds $\frac{p_i}{p_j} \leq 2$ for all p_i, p_j with $i \neq j$. We now show that this problem is NP-complete.

Lemma 6.1 The knapsack problem with bounded profit ratio of 2 is NP-complete.

Proof. We reduce the *equal partition problem* to *K-BPR2*. Let an instance of the *equal partition problem*, which is known to be NP-complete [GJ79], be given with a set $I = \{\tilde{w}_1, \dots, \tilde{w}_n\}$ such that $\sum_{i=1}^{n} \tilde{w}_i = B \in 2\,\mathbb{Z}, \tilde{w} \geq 0$. For the *equal partition problem* we ask for a partitioning of the elements of I into two subsets I_1, I_2 such that $\sum_{i \in I_1} \tilde{w}_i = \sum_{i \in I_2} \tilde{w}_i = \frac{B}{2}$ and $|I_1| = |I_2| = \frac{n}{2}$. Define $p_i = w_i := \tilde{w}_i + B + 1$ for all i. According to this definition, the smallest p_i is at least $B+1$ and the biggest p_i is not greater than $2B+1$ such that the ratio $\frac{p_i}{p_j}$ of all pairs $p_i, p_j, i \neq j$ is bounded by 2.

Suppose we are given a solution I_1 to the *equal partition problem*, we ask for a solution to the *knapsack problem* with profit

$$P \geq \frac{B}{2} + \frac{n}{2}(B+1)$$

and total weight

$$W \leq \frac{B}{2} + \frac{n}{2}(B+1).$$

We state, that the selection I_1 is such a solution. For the profit, it is $P = \sum_{i \in I_1} p_i = \sum_{i \in I_1} (\tilde{w}_i + B + 1) = \frac{B}{2} + \frac{n}{2}(B+1)$. Also, $W = \sum_{i \in W_1} w_i = \sum_{i \in I_1} (\tilde{w}_i + B + 1) = \frac{B}{2} + \frac{n}{2}(B+1)$.

On the other hand, given a solution $J \in I$ to the *knapsack problem* for which $P \geq \frac{B}{2} + \frac{n}{2}(B+1)$ and $W \leq \frac{B}{2} + \frac{n}{2}(B+1)$ we show that J together with $I \setminus J$ is a solution to the *equal partition problem*. To prove this statement, we need to show that $|J| = \frac{n}{2}$.
Suppose that $|J| < \frac{n}{2}$. Then for the profit, it is $\sum_{j \in J} p_j = \sum_{j \in J} (\tilde{w}_j + B + 1) \leq B + (\frac{n}{2} - 1)(B + 1) = \frac{n}{2}(B+1) - 1 < P$ which is a contradiction to the solution of the *knapsack problem*.
Analogously, suppose that $|J| > \frac{n}{2}$. Then for the weight it is $\sum_{j \in J} w_j = \sum_{j \in J} (\tilde{w}_j + B + 1) \geq B + (\frac{n}{2} + 1)(B+1) = 2B + 1 + \frac{n}{2}(B+1) > W$. Therefore, we have $|J| = \frac{n}{2}$. Finally, calculate

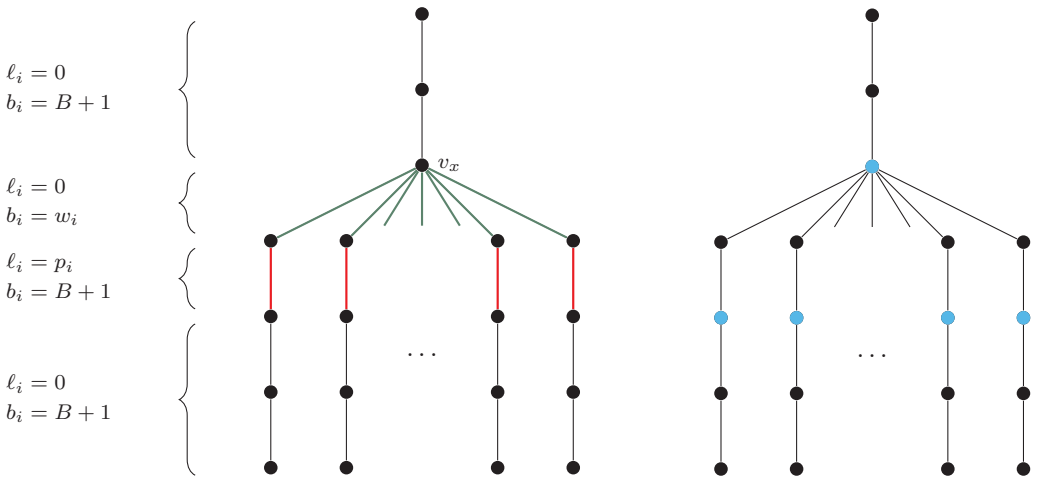
$$\sum_{j \in J} p_j = \sum_{j \in J} \tilde{w}_j + \frac{n}{2}(B+1) \geq \frac{B}{2} + \frac{n}{2}(B+1)$$

and also

$$\sum_{j \in J} w_i = \sum_{j \in J} \tilde{w}_j + \frac{n}{2}(B+1) \leq \frac{B}{2} + \frac{n}{2}(B+1).$$

Therefore, we get that $\sum_{j \in J} \tilde{w}_j = \frac{B}{2}$.

Clearly, given a solution to the *K-BPR2*, we can verify this solution in polynomial time. Thus, the *K-BPR2* is NP-complete. ■



(a) Red color depicts the edges with lengths $\ell_i = p_i \geq 0$; green color depicts the edges with individual interdiction costs $b_i = w_i$.

(b) One optimal locator's solution where the placed centers are depicted in blue.

Figure 6.1: The tree graphs used in the proof of Theorem 6.2.

With this result, we can prove our initial claim that the p-median location interdiction problem on trees is NP-complete.

Theorem 6.2 The p-median location interdiction problem, where the underlying graph is given by a tree, is NP-complete.

Proof. We show this by reducing *K-BPR2* to the location interdiction problem. Let an instance of *K-BPR2* be given by a set M of m items with associated weights w_i and profits p_i for all $i \in \{1, \ldots, m\}$. For the profit it holds for all p_i, p_j with $i \neq j$ that $\frac{p_i}{p_j} \leq 2$. Furthermore, let $W, P \in \mathbb{Z}_+$ be two integers. Consider a tree with lengths ℓ_i and interdiction costs b_i for every edge e_i as depicted in Figure 6.1a. Note, that the construction provides m paths of four vertices emerging from vertex v_x and one path of two vertices. Let $B = W$ be the interdiction budget. For every selection of interdicted edges, one optimal strategy to choose $m + 1$ centers is depicted in Figure 6.1b.

To prove this fact, we need to show that every single path emerging from vertex v_x must have one center, and furthermore, that the center in the paths with 4 vertices must be below the edges where $\ell_i = p_i$.

In case that every outgoing edge of v_x which is part of the m paths of 4 vertices (depicted in green in Figure 6.1a) is interdicted, the stated solution is optimal. Now consider the case where w.l.o.g. the leftmost interdictable edge is interdicted and let the second outgoing edge of v_x be non interdicted. The notation used can be found in Figure 6.2.

Every path with interdicted starting edge (the edge outgoing of v_x) needs at least one center for feasibility, since otherwise not all vertices are reached. Also, this center has to be placed below the edge with length greater than zero – in the considered case u_1 or a vertex below. Now, the only possibility for the locator to change the objective function value is to spare the center of

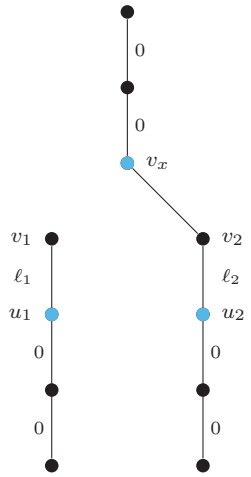


Figure 6.2: Excerpt of the tree from Figure 6.1.

the non interdicted paths – u_2 in the example – and instead place it at v_1. Then, the length ℓ_1 does not appear in the calculation of the objective function value. But instead, all vertices in the second path then need to be covered via center v_x. That means, the edge with length ℓ_2 needs to be crossed 3 times. Even if we assume that the lengths have the maximal possible factor $\ell_1 = 2 \cdot \ell_2$, it would still be better for the locator to place the center at vertex u_2.
The same explanation holds for the shifting of the center in v_x to a vertex in an interdicted path. In this case, the vertices above v_x need to be covered by another center below which is – by the same estimation as before – worse than the provided solution of Figure 6.1b.
Note that this solution is not unique. In fact – as stated before – the center in the m paths of 4 vertices can be placed at any vertex below the edges where $\ell_i = p_i$. Also, the center at v_x can be shifted up to two vertices up.

Assume we are given a solution of the *knapsack problem* such that the selection $I \in M$ of items fulfills $\sum_{i \in I} w_i \leq W$ and $\sum_{i \in I} p_i \geq P$. Now consider the optimal solution of the locator as stated above after interdiction of the edges $e_i, i \in I$. For the interdiction costs, it is $\sum_{i \in I} b_i = \sum_{i \in I} w_i \leq W$. Furthermore, the corresponding objective function value calculates as $\sum_{i \in I} \ell_i = \sum_{i \in I} p_i \geq P$.

On the other hand, if we are given a selection of interdicted edges $J \in E_M$ such that $\sum_{j \in J} b_j \leq B$ and $\sum_{j \in J} \ell_j \geq P$, we show that J is a solution to *K-BPR2*. Firstly, $\sum_{j \in J} w_j = \sum_{j \in J} b_j \leq B = W$. Also, the profit calculates as $\sum_{j \in J} p_j = \sum_{j \in J} \ell_j \geq P$.

Given an interdiction strategy γ, the corresponding p-median problem on $G(\gamma)$ can be solved in polynomial time as $G(\gamma)$ still does not contain cycles [KH79b]. Thus, the p-median interdiction problem on trees is contained in NP and therefore NP-complete. ■

We proved that the p-median location interdiction problem is NP-complete, even on trees in general. In the remainder of the article, we further restrict the problem in order to get a better impression on what makes the problem hard.

A direct consequence of the p-median location interdiction problem on trees being contained in

NP, is that regarding the interdiction budget as a constant makes the problem polynomial time solvable. This holds true, as having a constant interdiction budget makes the number of possible interdiction strategies polynomial in the instance size. Thus, the procedure of solving a p-median problem for all interdiction strategies and thereby finding the best strategy runs in polynomial time.

Observation 6.1 The p-median interdiction problem on trees is polynomial time solvable if the interdiction budget is considered a constant.

Further possible restriction are to simplify the graph structure even more, making the edges have unit interdiction costs, making the edges have constant or even unit lengths, or restricting the number p of locations to be placed. Any combinations of these restrictions is also interesting. Therefore we focus on the subcase of unit interdiction cost and regard different further restrictions. In the next section, we consider the p-median location interdiction problem on paths with unit interdiction costs and afterwards move back to the same problem on trees.

6.3 Interdicting a path

In this section, let $G = (V, E)$ be a graph with $V = \{v_1, \dots, v_n\}$ and $E = \{e_i = \{v_i, v_{i+1}\} \colon i = \{1, \dots, n-2\}$. The resulting graph has the structure of a path consisting of n vertices and $n-1$ edges. For the remainder, we refer to these types of graphs as paths.

In this section we tackle the p-median location interdiction problem on paths, where we additionally assume unit interdiction costs and $p = B + 1$, i.e. every component emerging from the interdiction can be equipped with exactly one location. Note that for unit interdiction costs, we may assume $B \leq n - 1$, as a path only contains $n - 1$ edges. We first elaborate on paths with unit lengths. In this case the interdictor can use a simple method to worsen the situation for the locator. This procedure is initially analyzed for one interdiction step ($B = 1$) and is then generalized to arbitrary interdiction budgets $B > 1$. After that, we show, how paths with arbitrary lengths can be handled for $B = 1$.

Recall the idea of Goldman's algorithm [Gol71] that is needed in the remainder of this section. For a tree T, we start at an arbitrary leaf and compare the weight of that leaf to the total summarized weight of all vertices. As long as the weight of the leaf is less than half of the total weight, we delete the leaf and update the weight of the adjacent vertex by adding the weight of the deleted leaf. In that manner, we iterate over the leaves until we find one with a weight greater or equal to the half of the total weight. This vertex is the 1-median of the original graph. With this method, we are also able to state the 1-median location(s) on a path, which is dependent on the number of the vertices. The optimal solution(s) on a path is to place the new location(s) at vertex $v_{n/2-1}$ or $v_{n/2}$ for n even or at vertex $v_{\lceil n/2 \rceil}$ for n odd.

6.3.1 Paths with unit edge weigths

Let the graph be a path $P = (v_1, e_1, v_2, \dots, e_{n-1}, v_n)$ with $\ell \equiv 1$ and $b \equiv 1$ as described above. We first examine the case for $B = 1$.

Lemma 6.3 Let P be a path with $\ell \equiv 1$ and $b \equiv 1$. The optimal interdiction strategies for the p-median location interdiction problem are to interdict e_1 or e_{n-1}, yielding in an isolated vertex and a new path of length $n-2$, i.e.

$$\Gamma^* = \{\gamma_1^* = (1, 0, \ldots, 0), \gamma_2^* = (0, \ldots, 0, 1)\} \subset \mathbb{R}^{n-1},$$

where the order of $\gamma_i^*, i = 1, 2$ is induced by the order of the edges.

Proof. As described above, the optimal solution(s) for the 1-median problem are at vertex $v_{n/2-1}$ or $v_{n/2}$ for n even or at vertex $v_{\lceil n/2 \rceil}$ for n odd. The resulting objective function value $\mathrm{OPT}(1, \sum, P)$ can then be computed via:

$$\mathrm{OPT}(1, \sum, P) = \begin{cases} \frac{n^2}{4} & n \text{ even} \\ \frac{n^2-1}{4} & \text{else} \end{cases}$$

Every interdiction strategy with $B = 1$ results in two components of the original path. Since $p = 2$, i.e. one new location in each component, we compute the optimal objective function value for the 2-median problem under the given setting by adding up both objective function values computed separately for every component. Therefore, let e_t be the interdicted edge for some $t \in \{1, \ldots, n-1\}$ yielding a separation of the original path P into $P_t = (v_1, e_1, v_2, \ldots, e_{t-1}, v_t)$ and $P_{n-t} = (v_{t+1}, e_{t+1}, \ldots, e_{n-1}, v_n)$. Then, the objective function for the overall problem of placing one new location in either part is

$$z^* = \mathrm{OPT}(1, \sum, P_t) + \mathrm{OPT}(1, \sum, P_{n-t}) = \frac{1}{4}(2t^2 + n^2 - 2nt - a) \tag{6.1}$$

with $a \in \{0, 1, 2\}$. The goal of the interdictor is to choose t such that z^* is maximal. For a given case, i.e. where n and a are fixed, we can reduce equation 6.1 to the following expression

$$t^2 - nt = \left(t - \frac{n}{2}\right)^2 - \frac{n^2}{4}$$

for the computation of the maximum. Since we aim to maximize the latter expression, it can again be reduced to

$$\left(t - \frac{n}{2}\right)^2.$$

For $t \in \{1, \ldots, n-1\}$, the maximum is found at $t_1 = 1$ or $t_2 = n-1$, which proves the claim. ■

This result allows to expand the considerations to $B \leq n-1$.

Lemma 6.4 Let a path $P = (v_1, e_1, v_2, \ldots, e_{n-1}, v_n)$ be given with $\ell \equiv 1$, $b \equiv 1$. The optimal interdiction strategy under an interdiction budget $B \leq n-1$ is to successively interdict the edges incident to leaves, thus resulting in B single vertices and one path of length $n-1-B$.

Proof. Consider an optimal interdiction strategy γ', where at least one of the interdicted edges does not cut off a leaf as depicted in Figure 6.3. Let e_r be the described edge. Given there exists an edge with the stated properties, there also exist sets of vertices $V_q = \{v_q \in V : q < r, \deg(v_q) = 1\}$

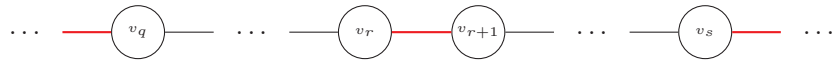


Figure 6.3: Detail of interdicted path P; interdicted edges are depicted in red.

and $V_s = \{v_s \in V : r < s, \deg(v_s) = 1\}$. Let $v_q \in V_q$ be the vertex with the biggest index and $v_s \in V_s$ the vertex with the smallest index. This requirement ensures that the component $P_{qs} = (v_q, e_q, \ldots, e_{s-1}, v_s)$ of the original path is only interdicted once at e_r.

Now consider P_{qs}, which is interdicted at e_r with strategy γ', resulting in two new paths. Using the result of Lemma 6.3, the optimal objective function value for the interdiction of path P_{qs} does not decrease by interdicting e_q instead of e_r. Therefore, successively using this method of shifting the interdicted edges to the leftmost edge of the respective components will also not decrease the overall objective value of γ' yielding an optimal interdiction strategy γ^* as stated. ■

6.3.2 Paths with arbitrary lengths

For the remainder of the section, we assume an arbitrary length function ℓ to be given.

One obvious strategy is to interdict all edges successively. In each step at a time, we compute the optimal objective function value for the locator by solving two median location problems on the remaining paths after the current interdiction step. Evaluating over all obtained objective function values yields the best edge to interdict. We present an approach which efficiently iterates over all edges by using an interesting structure of a matrix, which helps computing the locators objective function values. Let a path $P = (v_1, e_1, v_2, \ldots, e_{n-1}, v_n)$ be given. The median location is found at vertex $v_{\lceil n/2 \rceil}$ for n odd or at vertex $v_{n/2}$ or $v_{n/2+1}$ for n even. The objective function value for the 1-median problem on the given path is determined by the total number of times, each edge is crossed to reach all vertices from the median location. Given the structure of a path, these numbers are bounded by $\lfloor n/2 \rfloor$ if n is odd and by $n/2$ if n is even. More precisely, these bounds hold for the edges $e_{\lfloor n/2 \rfloor}$ and $e_{\lceil n/2 \rceil}$ incident to the median location (n odd) or the edge $e_{n/2}$ (n even), respectively. Furthermore, this number decreases by one the closer the edges are to the leaves of the path. Example 6.5 shows the case for a path of length 7.

Example 6.5 Let $P = (v_1, e_1, \ldots, e_6, v_7)$ a path with length values $\ell_i, i = \{1, \ldots, 6\}$ as depicted. With n odd and the observations above, we get that the median is located at $v_{\lceil n/2 \rceil} = v_4$. Furthermore, we can determine the number of times s_i the edge e_i is represented in the objective function value (depicted in green).

The information of how often an edge length contributes to the objective function value can be stored in a vector $S \in \mathbb{N}^{n-1}$, where each entry represents one edge. Multiplying S with the length vector ℓ yields the optimal objective function value for the path.

We use this scheme for the construction of the matrix calculating the objective function values for different interdicted edges. Assume that some edge e_t, $t \in \{1, \ldots, n-1\}$ gets interdicted.

This results in the two paths $P_{t,1} = (v_1, e_1, v_2, \ldots, e_{t-1}, v_t)$ and $P_{t,2} = (v_{t+1}, e_{t+1}, \ldots, e_{n-1}, v_n)$, for which the objective function values can be calculated separately as stated via the vectors $S_{t,1} = (s_1, \ldots, s_{t-1}) \in$ for P_1 and $S_{t,2} = (s_{t+1}, \ldots, s_{n-1})$ for P_2. The union yields a new vector $\mathcal{S}_t = (S_{t,1}, 0, S_{t,2})$. Assuming that we only interdict once, we can proceed to build $\mathcal{S}_t$ for all edges $e_t, t = 1, \ldots, n$ sequentially. The matrix $\mathscr{S} = (\mathcal{S}_t)_t \in \mathbb{N}^{(n-1)\times(n-1)}$ can again be multiplied with ℓ. Evaluating for the biggest objective function value solves the 1−interdiction median problem. Example 6.6 shows the matrix for the path of Example 6.5.

Example 6.6 Let P be the path of Example 6.5. Furthermore, let the length vector ℓ be given. The matrix obtained via the presented method is as follows:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 \\ 1 & 0 & 1 & 2 & 2 & 1 \\ 1 & 1 & 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 0 & 1 & 1 \\ 1 & 2 & 2 & 1 & 0 & 1 \\ 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix}$$

As stated, the matrix $\mathscr{S}$ is of size $(n-1) \times (n-1)$, where each row $i \in \{1, \ldots, n-1\}$ represents the objective function value if edge e_i gets interdicted. Therefore, the procedure explained runs in time $\mathcal{O}(n^2)$ since the multiplication of matrix $\mathscr{S}$ with the given length vector ℓ is of stated time. We want to mention, that an increase of interdiction steps adds factor n to the running time for each single interdiction. This is due to the fact, that we need to consider every possible combination for the edges, that get interdicted.

6.4 Interdicting a tree

In this section, we show, how to interdict a tree $T = (V, E)$ with $\ell \equiv 1$, $b \equiv 1$, $B = 1$ and $p = B + 1$.

Theorem 6.7 For $T = (V, E)$ with $\ell \equiv 1$, $b \equiv 1$, $B = 1$ and $p = B + 1$, the optimal interdiction strategy is as follows: among all leaves of T, find one, that has the least distance to at least one optimal solution of the 1−median location problem on T. Interdict the edge incident to this leaf.

Proof. Since the proof is constructive, consider the exemplary tree in Figure 6.4 which illustrates the used structures.

Let $r \in V(T)$ be an optimal solution to the 1−median location problem on T. Also, let T be rooted in r. Let $f = (v_1, v_2) \in E(T)$ be an edge with v_2 being a leaf that fulfills

$$d(r, v_2) = \min_{\substack{r^* \in X^* \\ l \text{ leaf}}} d(r^*, l). \tag{6.2}$$

For all neighbors $w \in N(r)$, it holds that

$$|T_w| \leq |T - T_w|. \tag{6.3}$$

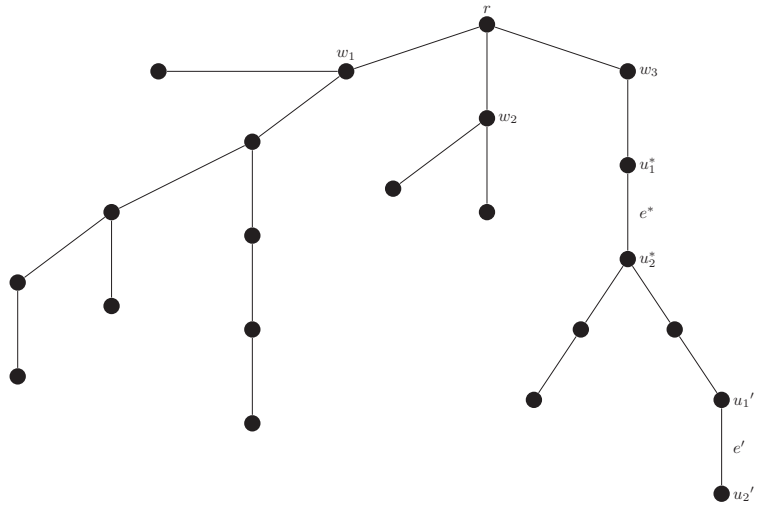


Figure 6.4: Exemplary tree for Theorem 6.7.

Assume, that $|T_w| > |T - T_w|$. Then, w would yield a better objective function value for the 1−median location problem than r, which is a contradiction to the assumption, that r is optimal for said problem. Now, for every edge $e = (u_1, u_2) \in E(T)$ with $d(r, u_1) < d(r, u_2)$, it is

$$\mathrm{OPT}(2, \textstyle\sum, T - e) \leq \mathrm{OPT}(1, \textstyle\sum, T) - (d(r, u_2) \cdot |T_{u_2}|). \tag{6.4}$$

This is, since placing a second location in the tree T_{u_2} saves at least $|T_{u_2}|$ times the distance $d(r, u_2)$ in comparison to the solution of the 1−median problem on the original tree T.

We aim at finding $e^* = (u_1^*, u_2^*)$ such that $d(r, u_2^*) \cdot \left|T_{u_2^*}\right| = \min_{e \in E(T)} d(r, u_2) \cdot |T_{u_2}|$. This leads to the biggest right hand side of inequality (6.4), thus providing an upper bound for the optimal objective function value $\mathrm{OPT}(2, \sum, T - e)$, which the interdictor maximizes. We claim, that u_2^* is a leaf.

Case 1: $d(r, u_2^*) > 1$. Suppose, u_2^* is not a leaf and let instead $e' = (u_1', u_2') \in T_{u_2^*}$ with u_2' being a leaf. Then, $\left|T_{u_2^*}\right| \geq d(r, u_2') - d(r, u_2^*) + 1$. Using this, we get

$$\begin{aligned} d(r, u_2^*) \cdot \left|T_{u_2^*}\right| &\geq d(r, u_2^*) \cdot d(r, u_2') - d^2(r, u_2^*) + d(r, u_2^*) \\ &= d(r, u_2') \left(d(r, u_2^*) - \frac{d(r, u_2^*)(d(r, u_2^*) - 1)}{d(r, u_2')} \right) \\ &> d(r, u_2') \left(d(r, u_2^*) - (d(r, u_2^*) - 1) \right) \qquad (6.5) \\ &= d(r, u_2') = d(r, u_2') \cdot \left|T_{u_2'}\right| \end{aligned}$$

where (6.5) follows from the fact that $\frac{d(r, u_2^*)}{d(r, u_2')} < 1$ under the assumptions made. This is a contradiction to $d(r, u_2^*) \cdot \left|T_{u_2^*}\right| = \min_{e \in E(T)} d(r, u_2) \cdot |T_{u_2}|$.

Case 2: $d(r, u_2^*) = 1$. In this case, (6.5) does not hold. We need to distinguish between two cases.

Case 2.1: u_2^* is a leaf. ✓

Case 2.2: Assume, that u_2^* is not a leaf. Again, we distinguish between two cases.

Case 2.2.1: Assume, that $T_{u_2^*}$ is not a path. Let $u_2' \in T_{u_2^*}$ be a leaf. There is at least one vertex $u'' \in T_{u_2^*}$ for which $d(u'') \geq 2$. Thus, it holds that

$$\begin{aligned} & d(u_2^*, u_2') \leq \left|T_{u_2^*}\right| - 2 \\ \iff & d(r, u_2') - 1 \leq \left|T_{u_2^*}\right| - 2 \\ \iff & d(r, u_2') < \left|T_{u_2^*}\right| \end{aligned}$$

Since $d(r, u_2^*) = |T_{u'}| = 1$, we get $d(r, u') \cdot |T_{u'}| < d(r, u_2^*) \cdot T_{u_2^*}$, which is a contradiction to $d(r, u_2^*) \cdot \left|T_{u_2^*}\right| = \min_{e \in E(T)} d(r, u_2) \cdot |T_{u_2}|$.

Case 2.2.2: Assume $T_{u_2^*}$ is a path. Then, we find that $\min_{e \in E(T)} d(r, u_2) \cdot |T_{u_2}| = d(r, u_2^*) \cdot \left|T_{u_2^*}\right| = d(r, u_2') \cdot \left|T_{u_2'}\right|$ with $u_2' \in T_{u_2^*}$ being a leaf.

Coming back to inequality (6.4) we conclude, that the right hand side is biggest, if for $e^* = (u_1^*, u_2^*)$, vertex u_2^* is a leaf.

We now want to use the term on the right side of inequality (6.4) as an upper bound for the objective function value of the interdiction problem. Therefore, let $f = (v_1, v_2)$ be the edge from Assumption (6.2), which gets interdicted. Then, we find the following.

If the optimal solution r to the 1−median problem on T is part of the optimal solution of the 2−median problem on $(T - f)$ – together with the leaf v_2 – then it holds that

$$\mathrm{OPT}(2, \textstyle\sum, T - f) = \mathrm{OPT}(1, \sum, T) - d(r, v_2).$$

Now suppose, that the optimal solution r to the 1−median problem on T is not part of the optimal solution of the 2−median problem on $(T - f)$ anymore. That means there exists $s \in N(r)$ in the neighborhood of r which is part of the optimal solution together with leaf v_2. Using the fact, that $d(r, s) = 1$, we get that

$$\mathrm{OPT}(2, \textstyle\sum, T - f) = \mathrm{OPT}(1, \sum, T) - d(r, v_2) - 1. \tag{6.6}$$

Note, that $\mathrm{OPT}(2, \sum, T - f)$ does not depend on the specific choice of f, meaning the objective function value is the same for all f fulfilling (6.2). In any case it holds true that

$$\mathrm{OPT}(2, \textstyle\sum, T - f) \geq \mathrm{OPT}(1, \sum, T) - d(r, v_2) - 1. \tag{6.7}$$

Now we distinguish between two cases.

Case 1. Let $f' = (v_1', v_2')$ such that v_2' is a leaf, but not fulfilling assumption (6.2). Therefore, $d(r, v_2') > d(r, v_2)$. Then we get that

$$\begin{aligned} \mathrm{OPT}(2, \textstyle\sum, T - f') &\leq \mathrm{OPT}(1, \sum, T) - d(r, v_2') && (6.8) \\ &\leq \mathrm{OPT}(1, \textstyle\sum, T) - d(r, v_2) - 1 && (6.9) \\ &\leq \mathrm{OPT}(2, \textstyle\sum, T - f) && (6.10) \end{aligned}$$

where (6.8) follows from inequality (6.4), (6.9) follows from the estimation of the distances in the current case, and (6.10) is due to inequality (6.7).

Case 2. Let $f' = (v_1', v_2')$ such that v_2' is not a leaf. Then, with $q_2 \in T_{v_2'}$ being a leaf, it is

$$\begin{aligned} \mathrm{OPT}(2, \textstyle\sum, T - f') &\leq \mathrm{OPT}(1, \textstyle\sum, T) - d(r, v_2') \cdot \left|T_{v_2'}\right| && (6.11)\\ &\leq \mathrm{OPT}(1, \textstyle\sum, T) - d(r, q_2) - 1 && (6.12)\\ &\leq \mathrm{OPT}(1, \textstyle\sum, T) - d(r, v_2) - 1 && (6.13)\\ &\leq \mathrm{OPT}(2, \textstyle\sum, T - f). && (6.14) \end{aligned}$$

The first line (6.11) again follows from inequality (6.4), (6.12) is due to the definition of q_2, (6.13) follows from condition (6.2) and the last estimation (6.14) follows from inequality (6.7).

Combining our results of cases 1 and 2, we find that choosing an edge f for the interdiction step, which fulfills condition (6.2) yields the optimal objective function value for the interdictor, namely the biggest objective function value $\mathrm{OPT}(2, \sum, T - f)$. ■

Observation 6.2 In case 2.2.2, where $d(r, u_2^*) = 1$ and $T_{u_2^*}$ is a path, we have seen that

$$\min_{e \in E(T)} d(r, u_2) \cdot |T_{u_2}| = d(r, u_2^*) \cdot \left|T_{u_2^*}\right| = d(r, u_2') \cdot \left|T_{u_2'}\right|$$

for the leaf u_2'. We stated, that we choose the edge connecting the leaf for our interdiction strategy. In fact, this is mandatory in every case, except for u_2' being the successor of u_2^*. Let $\left|T_{u_2^*}\right| > 2$. Then, if the edge e^* is interdicted, the locator can improve the objective function value of the remainder of the path, which is not connected to the main tree containing r anymore, with the placement of the new location.

We now prove the running time of the described method.

Theorem 6.8 The procedure stated in Theorem 6.7 has a running time of $\mathcal{O}(n^2)$.

Proof. The procedure described above needs to compute an optimal solution of the 1-median location problem and the distance from all leaves to this solution. The former computation can be done in time $\mathcal{O}(n)$ [Gol71], whereas the latter can be done in time $\mathcal{O}(n^2)$ by a breadth-first-search on every vertex. Thus, in total, the procedure runs in $\mathcal{O}(n^2)$. ■

Remark 6.9 It is not possible to apply the procedure to generalized problems. There are several alterations possible, where one can change the input in one parameter while the rest of the problem setup stays the same. This includes a change of the interdiction budget ($B > 1$) or an arbitrary length function on the edges. For the first problem, we immediately see, that simply choosing the B leaves closest to the median location and interdict them in one step, can lead to the following problems. First of all, it is not clear, whether B leaves exist in the original graph. But even if there are, the procedure does not necessarily give the optimal

interdiction strategy as example 6.10 shows. Also, consecutively applying the procedure B times does not yield an optimal solution, as Example 6.11 shows.

For the case of arbitrary lengths on the edges, we find, that in case 1 of the proof, the inequality $\left|T_{u_2^*}\right| \geq d(r, u_2') - d(r, u_2^*) + 1$ relies on the fact, that the distances can be calculated via the amount of vertices, which is only possible because of the unit length values of the edges. In fact, consider the following minimal example 6.12 illustrating that the proposed method does not work for arbitrary lengths.

Example 6.10 Let a tree $T = (V, E)$ be given as in Figure 6.5.

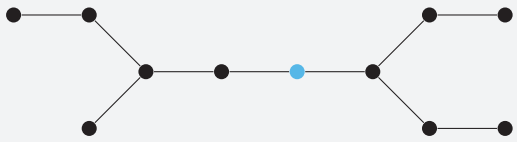
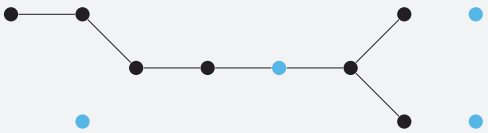
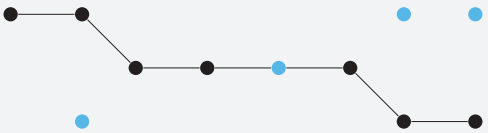


Figure 6.5: Exemplary tree, optimal median location depicted in blue.

Let the interdiction budget be given with $B = 3$. Then, if we choose the 3 leaves closest to the optimal location, we yield the graph in Figure 6.6a. The optimal objective function value of the locator is 15 in this case, but there are better interdiction strategies. One of the optimal strategies is depicted in Figure 6.6b and leaves the locator with an optimal objective function value of 16.



(a) Exemplary tree after interdiction step following the idea of Theorem 6.7, median locations depicted in blue.



(b) Exemplary tree after optimal interdiction step, median locations depicted in blue.

Figure 6.6: Exemplary tree of Figure 6.5 after different interdiction strategies.

Example 6.11 Let a tree $T = (V, E)$ be given as in Figure 6.7.

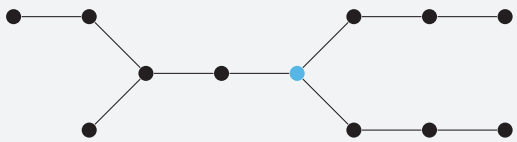
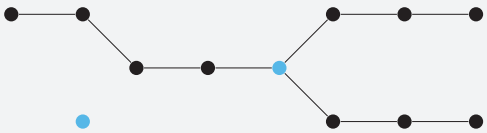


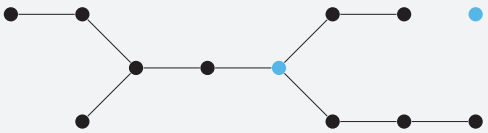
Figure 6.7: Exemplary tree, optimal median location depicted in blue.

Let the interdiction budget be given with $B = 2$. Then, applying the algorithm, we either cut the bottom left leaf or one of the leafs on the right side (see Figure 6.8). Choosing the bottom left leaf forces the interdiction of one of the leafs on the right side in the second application of the algorithm. One possibility of a resulting graph after iteratively applying the algorithm

two times is shown in Figure 6.9a. This leaves the locator with an optimal objective function value of 19 after the interdiction steps, while interdicting as shown in Figure 6.9b yields an optimal objective function value of 20, thus being a better strategy than the one found before.

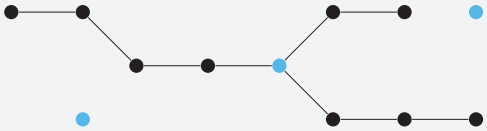


(a) Possible outcome after first interdiction step, median locations depicted in blue.

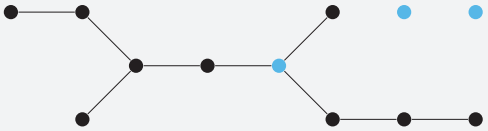


(b) Possible outcome after first interdiction step, median locations depicted in blue.

Figure 6.8: Exemplary tree of Figure 6.7 after different choices in the first call of the algorithm.



(a) Exemplary tree after iterative interdiction steps, median locations depicted in blue.



(b) Exemplary tree after optimal interdiction steps, median locations depicted in blue.

Figure 6.9: Exemplary tree of Figure 6.7 after different interdiction strategies.

Note, however, that it would have been possible to find the optimal interdiction strategy by choosing one of the leaves on the right side in the first call of the algorithm as this would have been a valid choice as well. After that, in the second call we would have chosen the leave next to the first interdicted one, thus leading to the optimal solution shown in Figure 6.9b. But since the algorithm does not take into account, what leaves are „generated“ after an interdiction, we can not guarantee an optimal outcome. It remains open, whether the optimal solution can always be found by iteratively applying the algorithm.

Example 6.12 Let a tree $T = (V, E)$ be given as in Figure 6.10 and $B = 1$.

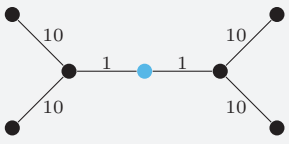
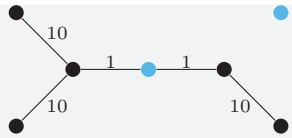
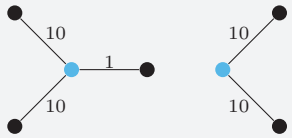


Figure 6.10: Exemplary tree, optimal median location depicted in blue.

Following the procedure proposed in Theorem 6.7, one of the four edges incident to the leaves gets interdicted resulting in the tree in Figure 6.11a. The optimal objective function value of the locator is 35, while the optimal interdiction strategy depicted in Figure 6.11b leaves the locator with an optimal objective function value of 41.



(a) Exemplary tree after interdiction step following Theorem 6.7, median locations depicted in blue.



(b) Exemplary tree after optimal interdiction step, median locations depicted in blue.

Figure 6.11: Exemplary tree of Figure 6.10 after different interdiction strategies.

7 Conclusion and Further Research

This thesis addressed two new areas in location planning.. Although location planning has been analyzed in a various different variants and in combination with different mathematical fields, the literature on location planning in the context of parametric optimization and interdiction problems is scarce.

In the first part, we investigated location planning on networks, where edge lengths are not fixed, but linearly dependent on a parameter. Using parametric optimization to tackle real world scenarios allows to model data that is variant over time, for instance. Therefore, an analysis of different parametric location problems is of interest. In this theses specifically, we studied the 1-center and 1-median problem on said networks where the underlying structure was either a directed or undirected general graph or a tree. We presented exact algorithms to solve these problems on trees and approximation methods for parametric general graphs. Furthermore, we investigated on the complexity of the solution set and also provided insights, how the optimal value function evolves depending on the parameter for the different problems.

In the second part, we studied the p-median interdiction problem. Here, the interdictor is allowed to remove a vertex from the network before the location planner selects an optimal location on the altered network. Interdiction problems are of great interest against the background of real world problems where they are relevant for numerous aspects of them. This includes the analysis of the vulnerability of systems or on the other hand information on how to disrupt undesired supply routes, for instance.
For trees, we proved this problem to be NP-complete, highlighting its complexity. For trees with unit-weighted edges, we could develop an algorithm that solves the median interdiction problem for 1 interdiction. For paths, we showed how to interdict for unit and arbitrary lengths.

The results contribute to the understanding of location planning on networks in the context of parametric optimization and interdiction, respectively.

However, for both topics, questions remain open that motivate further research in these fields.

As for parametric location planning, we already mentioned a computational study that compares both introduced approaches to approximate the median problem. Here, an investigation on whether the method of individually approximate the optimal value function for every vertex leads to a better practical running time through parallelization and/or a better theoretical approximation guarantee would be of interest.

A natural extension is to increase the number of parameters. The work of Helfrich et al. [Hel+22]

gives hope, that similar approximation approaches as introduced in this thesis could be carried over to multi-parametric problems. As they already state, the multi-parametric shortest path problem can be approximated with their algorithm. Investigating on how this can be used in a similar manner as for the single-parametric cases is of interest. Additionally, in this context an analysis on how the construction of the upper and/or lower envelope changes the running time and if it carries over the approximation guarantee remains open.

In the context of interdiction, we only provided insights on the p-median interdiction location problems. Not many interdiction location problems are investigated as of now. Since they are relevant to model real world problems, more research can be done in that field. No results were given for the p-center problem in this thesis, which is a direct continuation as a field of research. Also, analysis that is done so far leads to the conclusion that interdiction problems tend to be hard to solve exactly. Therefore, an interesting topic could be the exploration of heuristics for this class of problems.

Moreover, a more fundamental change could be made towards the understanding of the interdiction process itself. As of now, the goal is to optimally interdict such that the objective function value of the locator that acts afterwards, is worsened as much as possible. It is interesting to analyze whether the complexity of interdiction problems decreases, if we allow for "good enough" interdiction steps instead of insisting on the optimal.

Overall, this work lays the foundation for further exploration of parametric location and interdiction location problems on networks. By extending these concepts to more general settings, considering new problem variations, and developing algorithms or approximation schemes, future research can continue to advance the field of network-based facility location planning.

Bibliography

[AK08] S. A. Alumur and B. Y. Kara. "Network hub location problems: The state of the art". *European Journal of Operational Research* 190.1 (2008), pages 1–21. DOI: 10.1016/J.EJOR.2007.06.008.

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows - theory, algorithms and applications*. Prentice Hall, 1993. ISBN: 978-0-13-617549-0.

[AO20] O. J. Adeleke and D. O. Olukanni. "Facility Location Problems: Models, Techniques, and Applications in Waste Management". *Recycling* 5.2 (2020), pages 10–. DOI: 10.3390/recycling5020010.

[AP19] S. Avraamidou and E. N. Pistikopoulos. "Multi-parametric global optimization approach for tri-level mixed-integer linear optimization problems". *Journal of Global Optimization* 74 (2019), pages 443–465. DOI: 10.1007/s10898-018-0668-4.

[AP20] S. Avraamidou and E. N. Pistikopoulos. "Adjustable robust optimization through multi-parametric programming". *Optimization Letters* 14.4 (2020), pages 873–887. DOI: 10.1007/s11590-019-01438-5.

[APA10] D. Aksen, N. Piyade, and N. Aras. "The budget constrained r-interdiction median problem with capacity expansion". *Central European Journal of Operations Research* 18.3 (2010), pages 269–291. DOI: 10.1007/s10100-009-0110-6.

[AS00] P. K. Agarwal and M. Sharir. "Chapter 1 - Davenport–Schinzel Sequences and Their Geometric Applications". *Handbook of Computational Geometry*. North-Holland, 2000. ISBN: 978-0-444-82537-7.

[Ass87] N. Assimakopoulos. "A network interdiction model for hospital infection control". *Computers in Biology and Medicine* 17.6 (1987), pages 413–422. DOI: 10.1016/0010-4825(87)90060-6.

[Aus+99] G. Ausiello et al. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer-Verlag Berlin Heidelberg, 1999.

[Baz+22] C. Bazgan et al. "An approximation algorithm for a general class of parametric optimization problems". *Journal of Combinatorial Optimization* 43 (2022), pages 1–31. DOI: 10.1007/s10878-020-00646-5.

[BB05] R. Benkoczi and B. Bhattacharya. "A New Template for Solving p-Median Problems for Trees in Sub-quadratic Time". *Algorithms – ESA 2005*. Springer Berlin Heidelberg, 2005, pages 271–282. ISBN: 978-3-540-31951-1.

[BF12] A. Boloori Arabani and R. Z. Farahani. "Facility location dynamics: An overview of classifications and applications". *Computers & Industrial Engineering* 62.1 (2012), pages 408–420. DOI: 10.1016/j.cie.2011.09.018.

[BGV89] M. O. Ball, B. L. Golden, and R. V. Vohra. "Finding the most vital arcs in a network". *Operations Research Letters* 8.2 (1989), pages 73–76. DOI: 10.1016/0167-6377(89)90003-5.

[BKS98] A. Bar-Noy, S. Khuller, and B. Schieber. *The complexity of finding most vital arcs and nodes*. Technical report. UMIACS, 1998.

[BT97] D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization*. Volume 6. Athena scientific Belmont, MA, 1997.

[Car83a] P. J. Carstensen. "Complexity of some parametric integer and network programming problems". *Mathematical Programming* 26.1 (1983), pages 64–75. DOI: 10.1007/BF02591893.

[Car83b] P. J. Carstensen. "The complexity of some problems in parametric linear and combinatorial programming". PhD thesis. University of Michigan, 1983.

[CD82] H. W. Corley and Y. S. David. "Most vital links and nodes in weighted networks". *Operations Research Letters* 1.4 (1982), pages 157–160. DOI: 10.1016/0167-6377(82)90020-7.

[CE20] D. Celik Turkoglu and M. Erol Genevois. "A comparative survey of service facility location problems". *Annals of Operations Research* 292.1 (2020), pages 399–468. DOI: 10.1007/s10479-019-03385-x.

[CSM04] R. L. Church, M. P. Scaparra, and R. S. Middleton. "Identifying critical infrastructure: the median and covering facility interdiction problems". *Annals of the Association of American Geographers* 94.3 (2004), pages 491–502. DOI: 10.1111/j.1467-8306.2004.00410.x.

[CW02] W. Chen and K. Wada. "On computing the upper envelope of segments in parallel". *IEEE Transactions on Parallel and Distributed Systems* 13.1 (2002), pages 5–13. DOI: 10.1109/71.980023.

[DAP16] N. A. Diangelakis, S. Avraamidou, and E. N. Pistikopoulos. "Decentralized multi-parametric model predictive control for domestic combined heat and power systems". *Industrial & Engineering Chemistry Research* 55.12 (2016), pages 3313–3326. DOI: 10.1021/acs.iecr.5b03335.

[Das11] M. S. Daskin. *Network and discrete location: models, algorithms, and applications*. John Wiley & Sons, 2011.

[DD04] M. S. Daskin and L. K. Dean. "Location of Health Care Facilities". *Operations Research and Health Care: A Handbook of Methods and Applications*. Springer US, 2004. ISBN: 978-1-4020-8066-1.

[DH01] Z. Drezner and H. W. Hamacher. *Facility Location*. Springer Science & Business Media, 2001.

[Eis07] H. Eiselt. “Locating landfills—Optimization vs. reality”. *European Journal of Operational Research* 179.3 (2007), pages 1040–1049. DOI: 10.1016/j.ejor.2005.11.039.

[ES76] M. J. Eisner and D. G. Severance. “Mathematical techniques for efficient record segmentation in large shared databases”. *Journal of the ACM (JACM)* 23.4 (1976), pages 619–635.

[Far+13] R. Z. Farahani et al. “Hub location problems: A review of models, classification, solution techniques, and applications”. *Comput. Ind. Eng.* 64.4 (2013), pages 1096–1109. DOI: 10.1016/J.CIE.2013.01.012.

[Far+19] R. Z. Farahani et al. “OR models in urban service facility location: A critical review of applications and future developments”. *European Journal of Operational Research* 276.1 (2019), pages 1–27. DOI: 10.1016/j.ejor.2018.07.036.

[Fis+19] M. Fischetti et al. “Interdiction Games and Monotonicity, with Application to Knapsack Problems”. *INFORMS J. Comput.* 31.2 (2019), pages 390–410. DOI: 10.1287/IJOC.2018.0831.

[FMW83] R. L. Francis, L. F. McGinnis, and J. A. White. “Locational analysis”. *European Journal of Operational Research* 12.3 (1983), pages 220–252.

[FR21] N. Fröhlich and S. Ruzika. “On the hardness of covering-interdiction problems”. *Theoretical Computer Science* 871 (2021), pages 1–15. DOI: 10.1016/j.tcs.2021.04.007.

[FR22] N. Fröhlich and S. Ruzika. “Interdicting facilities in tree networks”. *TOP* 30.1 (2022), pages 95–118.

[Frö21] N. Fröhlich. *Facility Location Planning and Network Interdiction*. Dr. Hut, 2021.

[FSA10] R. Z. Farahani, M. SteadieSeifi, and N. Asgari. “Multiple criteria facility location problems: A survey”. *Applied Mathematical Modelling* 34.7 (2010), pages 1689–1709.

[FSE96] D. Fernández-Baca, G. Slutzki, and D. Eppstein. “Using sparsification for parametric minimum spanning tree problems”. *Algorithm Theory — SWAT’96*. Springer Berlin Heidelberg, 1996, pages 149–160. ISBN: 978-3-540-68529-6.

[Gal94] T. Gal. *Degeneracy, Multicriteria Decision Making, Redundancy*. De Gruyter, 1994. ISBN: 9783110871203.

[GG12] T. Gal and H. J. Greenberg. *Advances in sensitivity analysis and parametric programming*. Volume 6. Springer Science & Business Media, 2012. DOI: 10.1007/978-1-4615-6103-3.

[Giu+17] A. Giudici et al. “Approximation schemes for the parametric knapsack problem”. *Information Processing Letters* 120 (2017), pages 11–15. DOI: 10.1016/j.ipl.2016.12.003.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979. ISBN: 9780716710455.

[GK10] E. Gassner and B. Klinz. “A fast parametric assignment algorithm with applications in max-algebra”. *Networks: An International Journal* 55.2 (2010), pages 61–77.

[GMT71] P. Ghare, D. C. Montgomery, and W. Turner. “Optimal interdiction policy for a flow network”. *Naval Research Logistics Quarterly* 18.1 (1971), pages 37–45. DOI: 10.1002/nav.3800180103.

[Gol71] A. J. Goldman. “Optimal center location in simple networks”. *Transportation Science* 5.2 (1971), pages 212–221. DOI: 10.1287/trsc.5.2.212.

[GR19] K. Gajjar and J. Radhakrishnan. “Parametric shortest paths in planar graphs”. *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. 2019, pages 876–895.

[Gus83] D. Gusfield. “Parametric combinatorial computing and a problem of program module distribution”. *Journal of the ACM (JACM)* 30.3 (1983), pages 551–563.

[Hak64] S. L. Hakimi. “Optimum locations of switching centers and the absolute centers and medians of a graph”. *Operations research* 12.3 (1964), pages 450–459. DOI: 10.1287/opre.12.3.450.

[Hak65] S. L. Hakimi. “Optimum distribution of switching centers in a communication network and some related graph theoretic problems”. *Operations research* 13.3 (1965), pages 462–475. DOI: 10.1287/opre.13.3.462.

[Ham95] H. W. Hamacher. *Mathematische Lösungsverfahren für planare Standortprobleme*. Volume 246. Springer, 1995.

[Hel+22] S. Helfrich et al. “An approximation algorithm for a general class of multi-parametric optimization problems”. *Journal of Combinatorial Optimization* 44.3 (2022), pages 1459–1494.

[Her89] J. Hershberger. “Finding the upper envelope of n line segments in O(n log n) time”. *Information Processing Letters* 33.4 (1989), pages 169–174. DOI: 10.1016/0020-0190(89)90136-1.

[HK17] M. Holzhauser and S. O. Krumke. “An FPTAS for the parametric knapsack problem”. *Information Processing Letters* 126 (2017), pages 43–47. DOI: 10.1016/j.ipl.2017.06.006.

[HM03] T. S. Hale and C. R. Moberg. “Location Science Research: A Review”. *Annals of Operations Research* 123.1 (2003), pages 21–35.

[HNS98] H. W. Hamacher, S. Nickel, and A. Schneider. “Classification of location models”. *Location Science* 6.1-4 (1998), pages 229–242.

[HPR07] H. W. Hamacher, C. R. Pedersen, and S. Ruzika. “Multiple objective minimum cost flow problems: A review”. *European Journal of Operational Research* 176.3 (2007), pages 1404–1422. DOI: 10.1016/j.ejor.2005.09.033.

[HRT24] S. Helfrich, S. Ruzika, and C. Thielen. “Efficiently constructing convex approximation sets in multiobjective optimization problems”. *INFORMS Journal on Computing* (2024). DOI: 10.1287/ijoc.2023.0220.

[HS86] S. Hart and M. Sharir. “Nonlinearity of Davenport—Schinzel sequences and of generalized path compression schemes”. *Combinatorica* 6.2 (1986), pages 151–177.

[HT91] R. Hassin and A. Tamir. “Improved complexity bounds for location problems on the real line.” *Operations Research Letters* 10.7 (1991), pages 395–402. DOI: 10.1016/0167-6377(91)90041-m.

[IW02] E. Israeli and R. K. Wood. “Shortest-path network interdiction”. *Networks: An International Journal* 40.2 (2002), pages 97–111. DOI: 10.1002/net.10039.

[Jen87] L. Jenkins. “Using Parametric Integer Programming To Plan The Mix Of An Air Transport Fleet”. *INFOR: Information Systems and Operational Research* 25.2 (1987), pages 117–135. DOI: 10.1080/03155986.1987.11732033.

[KH79a] O. Kariv and S. L. Hakimi. “An Algorithmic Approach to Aetwork Aocation Aroblems. I: The p-Centers”. *SIAM Journal on Applied Mathematics* 37.3 (1979), pages 513–538.

[KH79b] O. Kariv and S. L. Hakimi. “An Algorithmic Approach to Network Location Problems. II: The p-Medians”. *SIAM Journal on Applied Mathematics* 37.3 (1979), pages 539–560.

[KN09] S. O. Krumke and H. Noltemeier. *Graphentheoretische Konzepte und Algorithmen*. Springer-Verlag, 2009.

[KO81] R. M. Karp and J. B. Orlin. “Parametric shortest path algorithms with an application to cyclic staffing”. *Discrete Applied Mathematics* 3.1 (1981), pages 37–45. DOI: 10.1016/0166-218X(81)90026-3.

[Lei+01] C. E. Leiserson et al. *Introduction to Algorithms*. Volume 2. MIT press, 2001.

[Lei+23] L. Leiß et al. “p-median location interdiction on trees”. *arXiv preprint arXiv:2301.13723* (2023). DOI: 10.48550/ARXIV.2301.13723.

[LMW88] R. Love, J. Morris, and G. Wesolowsky. *Facilities Location*. Publications in Operations Research. North-Holland, 1988. ISBN: 9780444010315.

[LNG15] G. Laporte, S. Nickel, and F. S. da Gama. *Location Science*. Volume 528. Springer, 2015.

[LSD11] F. Liberatore, M. P. Scaparra, and M. S. Daskin. “Analysis of facility protection strategies against an uncertain number of attacks: The stochastic R-interdiction median problem with fortification”. *Computers & Operations Research* 38.1 (2011), pages 357–366.

[Meg83] N. Megiddo. “Linear-Time Algorithms for Linear Programming in R^3 and Related Problems”. *SIAM J. Comput.* 12.4 (1983), pages 759–776. DOI: 10.1137/0212052.

[Mla+07] N. Mladenović et al. “The p-median problem: A survey of metaheuristic approaches”. *European Journal of Operational Research* 179.3 (2007), pages 927–939.

[MNS09] M. Melo, S. Nickel, and F. Saldanha-da-Gama. “Facility location and supply chain management—A review”. *European Journal of Operational Research* 196.2 (2009), pages 401–412. DOI: 10.1016/j.ejor.2008.05.007.

[MPS07] D. P. Morton, F. Pan, and K. J. Saeger. “Models for nuclear smuggling interdiction”. *IIE Transactions* 39.1 (2007), pages 3–14. DOI: 10.1080/07408170500488956.

[MS00] K. Mulmuley and P. Shah. “A lower bound for the shortest path problem”. *Proceedings 15th Annual IEEE Conference on Computational Complexity*. 2000, pages 14–21.

[Nem+25] L. Nemesch et al. “A Survey of Exact and Approximation Algorithms for Linear-Parametric Optimization Problems”. *arXiv preprint arXiv:2501.11544* (2025).

[Nik+06] E. Nikolova et al. “Stochastic shortest paths via quasi-convex maximization”. *Algorithms–ESA 2006: 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006. Proceedings 14*. 2006, pages 552–563.

[Obe+16] R. Oberdieck et al. “On multi-parametric programming and its applications in process systems engineering”. *Chemical Engineering Research and Design* 116 (2016), pages 61–82. DOI: https://doi.org/10.1016/j.cherd.2016.09.034.

[PDR03] E. N. Pistikopoulos, V. Dua, and J.-H. Ryu. “Global optimization of bilevel programming problems via parametric programming”. *Frontiers in global optimization*. Springer, 2003.

[Pis+02] E. N. Pistikopoulos et al. “On-line optimization via off-line parametric optimization tools”. *Computers & Chemical Engineering* 26.2 (2002), pages 175–185.

[PK08] B. Pfeiffer and K. Klamroth. “A unified model for Weber problems with continuous and network distances”. *Comput. Oper. Res.* 35.2 (2008), pages 312–326. DOI: 10.1016/J.COR.2006.03.001.

[RE05] C. S. ReVelle and H. A. Eiselt. “Location analysis: A synthesis and survey”. *European Journal of Operational Research* 165.1 (2005), pages 1–19.

[Ree06] J. Reese. “Solution methods for the p-median problem: An annotated bibliography”. *Networks: An International Journal* 48.3 (2006), pages 125–142.

[RSL75] H. D. Ratliff, G. T. Sicilia, and S. Lubore. “Finding the n most vital links in flow networks”. *Management Science* 21.5 (1975), pages 531–539.

[Ruh88] G. Ruhe. “Complexity results for multicriterial and parametric network flows using a pathological graph of Zadeh”. *Zeitschrift für Operations Research* 32 (1988), pages 9–27.

[Sch+20] L. E. Schäfer et al. “On the bicriterion maximum flow network interdiction problem”. *arXiv preprint arXiv:2010.02730* (2020).

[Sny06] L. V. Snyder. “Facility location under uncertainty: a review”. *IIE Transactions* 38.7 (2006), pages 547–564. DOI: 10.1080/07408170500216480.

[SS20] J. C. Smith and Y. Song. “A survey of network interdiction models and algorithms”. *European Journal of Operational Research* 283.3 (2020), pages 797–811. DOI: 10.1016/j.ejor.2019.06.024.

[Tam96] A. Tamir. “An $\mathcal{O}(pn^2)$ algorithm for the p-median and related problems on tree graphs”. *Operations Research Letters* 19.2 (1996), pages 59–64.

[TFL83a] B. C. Tansel, R. L. Francis, and T. J. Lowe. “State of the art—location on networks: a survey. Part I: the p-center and p-median problems”. *Management science* 29.4 (1983), pages 482–497.

[TFL83b] B. C. Tansel, R. L. Francis, and T. J. Lowe. “State of the art—location on networks: a survey. Part II: exploiting tree network structure”. *Management Science* 29.4 (1983), pages 498–511.

[URS20] T. Ullmert, S. Ruzika, and A. Schöbel. “On the p-hub interdiction problem”. *Computers & Operations Research* 124 (2020), page 105056.

[Wes93] G. O. Wesolowsky. “The Weber Problem: History and Perspectives.” *Computers & Operations Research* (1993).

[WN99] L. A. Wolsey and G. L. Nemhauser. *Integer and combinatorial optimization*. John Wiley & Sons, 1999.

[Wol64] R. Wollmer. “Removing Arcs from a Network”. *Operations Research* 12.6 (1964), pages 934–940. DOI: 10.21236/ad0601643.

[Woo93] R. K. Wood. “Deterministic network interdiction”. *Mathematical and Computer Modelling* 17.2 (1993), pages 1–18. DOI: 10.1016/0895-7177(93)90236-r.

[WP22] A. Weber and G. Pick. *Ueber den Standort der Industrien*. Volume 1. Mohr, 1922.

[WS11] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge University Press, Cambridge, 2011. ISBN: 978-0-521-19527-0.

[WS88] A. Wiernik and M. Sharir. “Planar realizations of nonlinear Davenport-Schinzel sequences by segments”. *Discrete & Computational Geometry* 3 (1988), pages 15–47.

[Zen10] R. Zenklusen. “Matching interdiction”. *Discret. Appl. Math.* 158.15 (2010), pages 1676–1690. DOI: 10.1016/J.DAM.2010.06.006.

Akademischer Lebenslauf

2019	**Annahme als Doktorandin am FB Mathematik**, *TU Kaiserslautern*
2015 – heute	**Wissenschaftliche Mitarbeiterin**, *TU Kaiserslautern*
2014	**Master of Education**, *TU Kaiserslautern*
2011	**Bachelor of Education**, *TU Kaiserslautern*
2007 – 2014	**Studium Lehramt an Gymnasien**, *TU Kaiserslautern* Fächer: Mathematik und Biologie
2007	**Abitur**, *Veldenz-Gymnasium Lauterecken*

Academic curriculum vitae

2019 **Accepted as Doctoral Student in Mathematics**, *TU Kaiserslautern*

2015 – today **Research Assistant**, *TU Kaiserslautern*

Nov. 2014 **Master of Education**, *TU Kaiserslautern*

Aug. 2011 **Bachelor of Education**, *TU Kaiserslautern*

2007 – 2014 **Studies Educational Science (Lehramt an Gymnasien)**, *TU Kaiserslautern*
Subjects: Mathematics and Biology

2007 **High School Graduation (Abitur)**, *Veldenz-Gymnasium Lauterecken*