BERGISCHE
UNIVERSITÄT
WUPPERTAL

Location Problems with k-max Functions

Modelling and Analysing Outliers in Center Problems

Dissertation

zur Erlangung des Doktorgrades (Dr. rer. nat.)
Fakultät 4 – Mathematik und Naturwissenschaften
Bergische Universität Wuppertal

vorgelegt von

Teresa Schnepper
aus Wipperfürth

Wuppertal, 31. März 2017

Die Dissertation kann wie folgt zitiert werden:

urn:nbn:de:hbz:468-20170802-142024-4
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz%3A468-20170802-142024-4]

Contents

List of Main Notation and Abbreviations

Basic Notation

(1kMG)	1-k-max problem on a graph
$A(G)$	continuum set of points on the edges of G
BN	set of all bottleneck points of G
BN_i	set of all bottleneck points of node $v_i \in V$
BN^{ab}	set of bottleneck points on edge $e_{ab} \in E$
BN_i^{ab}	set of bottleneck points of $v_i \in V$ on $e_{ab} \in E$
Cand	equal to EQ for $k \leq n-1$ and equal to V for $k = n$
Cand2	$\bigcup_{a,b \in \{1,\ldots,n\}} \left\{ \arg\min_{x=(e_{ab},t)} F_{ab}^{n-k}(x) \right\}$
Cand3	$\bigcup_{\substack{e_{ab} \in E \\ a,b \in \{1,\ldots,n\}}} \left\{ x^\alpha : x^\alpha = \left(e_{ab}, \frac{\alpha}{2l_{ab}} \right),\ \alpha = 0, 1, 2, \ldots, 2l_{ab} \right\}$
Cand4	$EQ \times (EQ \cup V)^{p-1}$
C_ℓ	cluster of $x_\ell \in X$
$C(x_\ell)$	$\{v \in V : d^w(v, x_\ell) \leq z_\ell\}$; cover of $x_\ell \in X$
$d(x', x'')$	distance of a shortest path between the points $x', x'' \in A(G)$
$d^w(v_i, x)$	weighted distance of a shortest path between $v_i \in V$ and $x \in A(G)$
$d^w(v_i, X)$	weighted distance between $v_i \in V$ and its closest new $x \in X$
$d^w(V, X)$	vector of weighted distances between the sets V and X
$E = \{e_1, \ldots, e_m\}$	set of edges of G
(e_{ab}, t)	point on edge $e_{ab} \in E$ at relative position $t \in [0, 1]$
EQ	set of equilibrium points of G
EQ_{ij}	equilibrium points of $v_i, v_j \in V$; relative boundary of EQ'_{ij}
EQ'_{ij}	$\{x \in A(G) : w_i d(v_i, x) = w_j d(v_j, x)\}$
EQ^{ab}	set of equilibrium points on edge e_{ab} of G
EQ_{ij}^{ab}	set of equilibrium points of $v_i, v_j \in V$ on $e_{ab} \in E$
f_k	k-max function
$f_{pc}(X) = f_1(X)$	p-center objective function w.r.t. X
G	finite, connected, simple, undirected graph

k	parameter defining the kth largest distance to be minimised
k-max$(d^w(V,X))$	k-max function of V and X
l_j	length of edge $e_j \in E$
m	number of edges of G
n	number of existing facilities
p	number of new facilities to locate
(pkMG)	p-k-max problem on a graph G
(pkMP)	continuous p-k-max problem
Π_n	set of all permutations of $\{1,\ldots,n\}$
σ	permutation in $\Sigma(X)$
$\Sigma(X)$	$\{\sigma \in \Pi_n : d^w(v_{\sigma(1)},X) \geq \ldots \geq d^w(v_{\sigma(n)},X)\}$
$V = \{v_1,\ldots,v_n\}$	set of nodes (existing facilities, customers) of G
$V \setminus V_{k-1}$	set of center defining nodes
V_{k-1}	set of outliers w.r.t. a given set of new facilities X
w_i	weight of facility $v_i \in V$
$X = \{x_1,\ldots,x_p\}$	set of new facilities
$\mathcal{X}_{\text{Cand4}}$	set of optimal solutions of (pkMG) determined by Algorithm 9
$\mathcal{X}_{\text{l-all}}$	set of optimal solutions of (pkMG) determined by Algorithm 13
$\mathcal{X}_{\text{l-full}}$	set of optimal solutions of (pkMG) determined by the evaluation of the finite dominating set $V(L')$
$\mathcal{X}_{\text{l-red}}$	set of optimal solutions of (pkMG) determined by the evaluation of the finite dominating set $V^{eq}(L(Y))$
$\mathcal{X}_r$	set of optimal solutions of (pkMG) determined by the recursive approach (Algorithm 10)
$[x',x''] \subseteq e_j$	subedge of $e_j \in E$ between point $x' \in A(G)$ and $x'' \in A(G)$
$y_{ij} \in EQ$	equilibrium point defined by $v_i \in V$ and $v_j \in V$

Chapter 2

$A = \{a_1,\ldots,a_n\}$	set of existing facilities in a continuous location problem
(BP)	bi-criteria optimisation model for general k-max problems
(BP$_\varepsilon$)	ε-constraint problem of (BP)
(BP$_{\varepsilon^=}$)	modified ε-constraint problem of (BP)

(BPL)	bi-criteria optimisation model for p-k-max location problems
(BPL$_\varepsilon$)	ε-constraint problem of (BPL)
(BPL$_{\varepsilon^=}$)	modified ε-constraint problem of (BPL)
f_λ	ordered median function
$\gamma_B(x)$	$\min\{r > 0 : x \in rB\}$; gauge
L^k	set of all linearity regions for fixed k
L_b^k	linearity region $\{y \in \mathbb{R}^n : y_b = k\text{-max}(y)\}$
O_σ	ordered region w.r.t. σ

Chapter 4

$e_1(S_i)$	left endpoint of line segment S_i
$e_2(S_i)$	right endpoint of line segment S_i
i_{S_i,S_j}	intersection point of line segments S_i and S_j in L_{ab}
ℓ	sweepline in L_{ab}
L_{ab}	arrangement of line segments defined by the graphs of the weighted distances functions over the edge e_{ab}
$l(S_i)$	line segment of L_{ab}; lower neighbour of S_i for current position of ℓ
$\mathcal{R}_r$	set of all subsets of $\{1, \ldots, n\}$ with cardinality r
S_i	line segment in L_{ab} corresponding to the distance function $d^w(v_i, x)$
$S_i \succeq S_j$	ordering relation; S_i is above S_j w.r.t. a fixed position of sweepline ℓ
$u(S_i)$	line segment of L_{ab}; upper neighbour of S_i for current position of ℓ

Chapter 5

$C(S_g)$	set of cells of subdivision S_g
dia(P)	diameter of path graph P
F_{ab}	upper envelope of the arrangement L_{ab}
$F_{ab}^h(x)$	h-level of the arrangement L_{ab}
$S = (V', V'')$	$\{e_{ij} \in E : v_i \in V', v_j \in V''\}$; a cut in G
S_g	subdivision of $e_g \in E$ defined by equilibrium points and nodes on e_g
X_{ab}^g	set of local minima of F_{ab}^{n-k} of L_{ab}

Chapter 6

Section 6.1 and Section 6.2

$C(q)$	$\bigcup_{\ell=1}^{q-1} C(x_\ell)$; set of nodes covered by the new facilities in X^{q-1}
$C(x_\ell, z_1)$	$\{v \in V : d^w(v, x_\ell) \leq z_1\}$; cover of x_ℓ w.r.t. radius z_1
$C(x_\ell, z_\ell)$	$\{v \in V : d^w(v, x_\ell) \leq z_\ell\}$; cover of x_ℓ w.r.t. radius z_ℓ
$EQ(q)$	$\{y \in EQ : \exists\ v_s, v_t \in V(q)$ such that $y \in EQ_{st}\}$
$\bar{n}$	$\lvert V(q)\rvert$
$\bar{p}$	$p - q + 1$
q	current iteration number
$V(q)$	$V \setminus C(q)$; customers not covered by X^{q-1}
X^{q-1}	set of facilities located in the first $q-1$ iterations
Z	$\{d^w(v_a, y) : y \in EQ_{ab}$ with $a, b \in \{1, \ldots, n\}\}$; set of possible optimal objective function values of the p-k-max problem
z_ℓ	coverage radius of equilibrium point x_ℓ

Section 6.3

$[0,1]^2$	unit square
$[0,1]^p$	unit hypercube
$A(H)$	arrangement of hyperplanes in H
$B_{ijqq}^{\alpha\beta}$	equilibrium plane over $[0,1]^p$ (resp. equilibrium line for $p=2$)
$B_{ijqr}^{\alpha\beta}$	balance plane over $[0,1]^p$ (resp. balance line for $p=2$)
B_{iiqq}^{+-}	bottleneck plane over $[0,1]^p$ (resp. bottleneck line for $p=2$)
BP	set of all balance points of G
BP_{ij}	set of balance points of nodes $v_i, v_j \in V$
$C(L)$	set of cells of a subdivision L
$d_+^w(v_i, x)$	length of a shortest path between $v_i \in V$ and $x = (e_{ab}, t)$ via $v_a \in V$
$d_-^w(v_i, x)$	length of a shortest path between $v_i \in V$ and $x = (e_{ab}, t)$ via $v_b \in V$
e_{g^*}	edge on which $x_1^* \in Y$ lies
$eq_{g^*h}(x_1^*)$	equilibrium line in $[0,1]^2$ defined by x_1^*
$eq_g(x_1^*)$	equilibrium plane in $[0,1]^p$ defined by x_1^*
$F_0^{\bar{C}}$	set of optimal vertices of a fixed cell $\bar{C} \in C(L_g(x_1^*))$
f_i^s	ith optimal s-face of the subdivision $L_g(x_1^*)$

H	set of hyperplanes in $\mathbb{R}^p$
$I(H) = (V(I), E(I))$	incidence graph of $A(H)$
L_{ijg}	subdivision of $[0,1]^p$ defined by the equilibrium planes, bottleneck planes and balance planes for fixed $v_i, v_j \in V$ and fixed indices g
L_{ijgh}	subdivision of $[0,1]^2$ defined by the equilibrium lines, bottleneck lines and balance lines for fixed $v_i, v_j \in V$ and fixed $e_g, e_h \in E$
L_g	$\bigcup_{i,j\in\{1,\dots,n\}} L_{ijg} \cup bd\,([0,1]^p)$
L_{gh}	$\bigcup_{i,j\in\{1,\dots,n\}} L_{ijgh} \cup bd([0,1]^2)$
$L_g(x_1^*)$	subdivision of $[0,1]^p$ arising from L_g by deleting bottleneck planes of type $B_{ii11}^{\alpha\beta}$ and equilibrium planes of type $B_{ij11}^{\alpha\beta}$ except for $eq_g(x_1^*)$
$L_g^{eq}(x_1^*)$	$L_g(x_1^*) \cap eq_g(x_1^*)$; subdivision $L_g(x_1^*)$ restricted to $eq_g(x_1^*)$
$L_{g^*h}(x_1^*)$	subdivision of $[0,1]^2$ defined by the balance lines, all equilibrium and all bottleneck lines corresponding to e_h and the single equilibrium line $eq_{g^*h}(x_1^*)$ corresponding to e_{g^*} on which x_1^* lies
$\bar{L}_g$	subdivision of $[0,1]^p$ defined by equilibrium planes, balance planes for $i \neq j$ and boundary faces of $[0,1]^p$
$\bar{L}_{gh}$	subdivision of $[0,1]^2$ defined by equilibrium lines, balance lines for $i \neq j$ and boundary segments of $[0,1]^2$
$\tilde{L}_g$	subdivision of $[0,1]^p$ defined by bottleneck planes, balance planes for $i = j$ and boundary faces of $[0,1]^p$
$\tilde{L}_{gh}$	subdivision of $[0,1]^2$ defined by bottleneck lines, balance lines for $i = j$ and boundary segments of $[0,1]^2$
L'_g	subdivision of $[0,1]^p$ defined by equilibrium planes, bottleneck planes and boundary faces of $[0,1]^p$
L'_{gh}	subdivision of $[0,1]^2$ defined by equilibrium lines, bottleneck lines and boundary segments of $[0,1]^2$
$V(L)$	set of vertices (0-faces) of a subdivision L
$V(L')$	$\bigcup_{g,h=1,\dots,m} V(L'_{gh})$ for $p = 2$ resp. $\bigcup_{g\subset\{1,\dots,m\}^p} V(L'_g)$ for $p \geq 3$
$V^{eq}(L(x_1^*))$	$\bigcup_{h=1,\dots,m} V^{eq}(L_{g^*h}(x_1^*))$ for $p = 2$ resp. $\bigcup_{g\subset\{1,\dots,m\}^p} V^{eq}(L_g(x_1^*))$ for $p \geq 3$
$V^{eq}(L(Y))$	$\bigcup_{\bar{x}_1\in Y} V^{eq}(L(\bar{x}_1))$ for $p = 2$ resp. $\bigcup_{\bar{x}_1\in Y} V^{eq}(L(\bar{x}_1))$ for $p \geq 3$
$V^{eq}(L_g(x_1^*))$	set of all 0-faces of $L_g(x_1^*)$ that lie in the hyperplane $eq_g(x_1^*)$
$V^{eq}(L_{g^*h}(x_1^*))$	set of intersection points of $eq_{g^*h}(x_1^*)$ with one of the other line segments in $L_{g^*h}(x_1^*)$
V_s	$\left\{v_i^s \in V(I) : f_i^s \text{ is } i\text{th optimal } s\text{-face}, i \in \{1,\dots,\lvert V_s\rvert\}\right\}$

v_i^s	node in $V(I)$ that represents f_i^s
$x_1^* \in Y$	optimal objective function value defining facility
$\mathcal{X}(x_1^*)$	set of alternative optimal solutions not in $\mathcal{X}_r$ that have the objective value defining facility $x_1^* \in Y$
Y	set of optimal objective function value defining equilibrium points

Section 6.4

B_i	$\{v \in C_i : d^w(v, x_1) > z^*\}$
C_i	cluster of $x_i \in X$
$Q(\bar{v})$	$\{\bar{x} \in A(G) : d^w(\bar{v}, \bar{x}) \leq z^*\}$
$Q(x_i)$	$\bigcap_{\bar{v} \in B_i} Q(\bar{v})$
$Q^{ab}(v_j)$	$\{\bar{x} \in e_{ab} : d^w(v_j, \bar{x}) \leq z^*\}$
$Q^{ab}(x_i)$	points of $Q(x_i)$ on edge e_{ab}

Chapter 8

G^0	graph resulting from G by setting the outlier weights to 0
$\widetilde{G} = (\widetilde{V}, \widetilde{E})$	graph resulting from G by replacing each vertex v_i by w_i copies, all having weight w_i
$G_{w^{-1}} = (V_{w^{-1}}, E)$	graph resulting from G by replacing the node weights by their reciprocal values

List of Algorithms

Chapter 1

Introduction

Success or failure of projects in both private or public sectors often depends on the decision where to locate a new facility. The overall aim of location problems is to place one or more new facilities with respect to a set of already existing facilities, also called clients or costumers, such that some kind of objective function (e.g. minimisation of cost or distance) gets optimal. Location models typically use the distances to all existing facilities as a quality criterion for the service provided by the new facilities. Common ways to aggregate these distances in one objective function are the center function, which relies on the largest distance between any customer and its closest new facility and the median function, which relies on the sum of all these distances (see, for example, Hakimi, 1964).

These classical objective functions are not necessarily useful from an economic point of view, especially when the existing facilities are distributed such that there is a (small) number of "far-away" facilities, i.e., customers that are located very distant from the majority of the other existing facilities. When such a scattered distribution of facilities is present in a location problem, the optimal solution may be influenced in an unwanted way. When locating, for example, a single facility in a median problem, the effect is that the cost of the optimal new location is disproportionally higher because the new facility is attracted by every existing facility and so also by the far-away customers.

In this thesis the focus is set on center location problems because here the effect of the distant customers on the optimal solution generally is even stronger. The new location highly depends on the location of the pair of customers with the largest distance among all pairs of existing facilities and may thus be in a suboptimal position w.r.t. the majority of customers: Many distances to the new facilities may be much larger by taking the interests of very distant facilities into account. As a consequence, the costs increase for a certain number of clients while the service quality worsens because of an unnecessarily large covering radius. On the other hand, in a covering problem, more new facilities are needed to keep the same radius. Therefore, in situations where mostly economic aspects are considered, the exclusion of so called *outliers* seems to be useful such that only a part of the clients have to be served and it makes the location problem more robust with respect to the influence of far-away facilities.

This approach is not intended to be used in situations where emergency facilities such as fire stations, hospitals or police departments have to be located. Here it is important that all existing facilities (e.g., houses with people needing help) without exclusions are reachable within a certain time limit. Buildings of companies which do not provide services fundamental for life such as delivery services supplying food or maintenance services as well

as stores of supermarket chains are in contrast good examples for a reasonable application of the mentioned approach of handling outliers. The possibility to exclude very distant costumers from the placement decision of a new delivery store can lead to a much more economic solution for the decision maker. One way to limit the influence of far-away facilities on the optimal center location is to use the kth largest distance between any customer and its corresponding closest new facility as a decision criterion where k can be varied from 1 up to the number of customers. This can be realized by a k-max function as objective instead of the center function to model the situation.

Outline of this Thesis

In this thesis the handling of outliers in center location problems is considered by using a so called k-max function. The k-max function is analysed for its mathematical properties, its relation to other well known functions used in location theory is derived, and different new modelling approaches of this problem are developed. The focus is set on the derivation of different finite dominating sets for p-k-max problems, for both $p = 1$ and $p \geq 2$, and approaches to efficiently evaluate these sets. In particular,

- the handling of outliers in center location problem is modelled by a bi-criteria optimisation problem to obtain the optimal locations of the new facilities as well as an appropriate value for the parameter k which reflects the number of outliers. Different integer programming formulations are presented for the resulting single-criteria problems. Two reformulations of the original problem are proposed to avoid possible unwanted effects on the optimal solution of a weighted problem. The relation to the ordered median problem is discussed.

- two algorithms based on finite dominating sets to solve the single-facility k-max problem in polynomial time are derived. Furthermore, two special cases of unweighted k-max problems are considered: A finite dominating set applicable to general unweighted graphs and an easy and fast solution approach for the unweighted problem on path graphs.

- a finite dominating set for the p-k-max problem is derived and used as a basis to develop a recursive algorithm that computes an optimal solution of this problem in polynomial time if the number of new facilities p is fixed. Numerical experiments for up to 50 customers confirm the practical applicability. A local analysis on the edges of the graph can be applied to find all optimal solutions of the underlying problem.

A variety of methodological tools and approaches from other fields are used, including different topics of computational geometry and polyhedral theory, a scalarisation method for multi-objective optimisation problems, graph-theoretic aspects, integer-programming approaches and structural properties of the ordered median function. The thesis is organised in nine chapters whose contents are described more detailed in the following.

Chapter 2 In this chapter the general k-max problem is introduced formally and some important properties of its objective function are proven. Afterwards, this general concept is applied to location problems to handle outliers in center problems. For this purpose, a bi-criteria optimisation model for the solution of the p-k-max location problem is proposed where the first objective function aims to minimise the kth largest distance between any customer and its closest new facility. The second objective function tries to keep the number of outliers as small as possible. Properties of the Pareto front of this problem are deduced and proven for the case of general k-max functions such that the results can also be applied to other (combinatorial) optimisation problems. Based on this, it is shown that all weakly non-dominated points of the bi-criteria problem can be obtained in polynomial time by solving a sequence of single-criteria k-max problems, each with a fixed value of k. Moreover, the efficient points of the Pareto front can be filtered out when a certain order of the subproblems is adhered.

Chapter 3 The literature review provided in this chapter gives a detailed overview on the topics closely related to the subject of the thesis. It includes sections with a short introduction to general location analysis, a closer look on center location problems and previous approaches for outlier handling in location problems. Also ordered median problems, of which the k-max problem is a special case, are reviewed and some results for combinatorial k-max problems are presented. Finally, an overview of the most common strategies for anomaly detection on big data sets in the field of statistics is given.

Chapter 4 Location problems with k-max functions on networks are introduced. p-k-max problems are shown to be NP-hard for any fixed value of k. An important factor for the development of solution approaches for the p-k-max problem is its close relation to p-center problems. It is proven that a feasible solution of the p-k-max problem is optimal if and only if it is optimal for the p-center problem on a subset of the existing facilities. In the following section, three formulations of the p-k-max problem based on (mixed-) integer programming models are presented. The first formulation is a linear integer program to evaluate a candidate set of possible optimal solutions. It is composed of a sorting problem and a location-allocation problem. The second model gets rid of the sorting process which decreases the number of variables and constraints significantly. The third model is a mixed-integer formulation which does not depend on a predefined candidate set.

Chapter 5 This chapter is dedicated to the 1-k-max problem, i.e., only one new facility has to be located. This problem has a simpler structure as the multi-facility problem. Properties of outlier nodes are analysed in the first section. These results can be used to reduce the set of customers significantly in a preprocessing phase of any solution approach as some of the nodes satisfying these properties are known to be outliers in an optimal solution. The following two sections derive two solution approaches for the k-max problem, both based on the evaluation of a finite set of candidates. The first approach proves that the set of equilibrium points of the graph is a finite dominating set. It is shown that these points define a subdivision on the edges of the graph such that the objective function is piecewise

linear and concave over each cell of this subdivision. The second approach uses h-levels of the arrangement of line segments given by the distance functions over each edge of the graph. The local minima of these h-levels are shown to be a candidate set for the k-max problem. Moreover, two special cases of k-max problems on unweighted (path) graphs are considered and adjusted solution approaches are presented.

Chapter 6 The p-k-max problem for a general $p \geq 2$ is considered in this chapter. It is shown that there exists at least one optimal solution of the problem where each new facility is located in an equilibrium point or in a node of the graph. Based on this finite dominating set, the second section provides a recursive solution approach that evaluates the candidate solutions in a much more efficient way such that in general only a small subset of all candidates has to be evaluated. The individual new facilities of a solution are located step by step and with respect to the already located facilities. The aim of the following sections is to find further optimal solutions where not all new facilities have to be located in equilibrium points or nodes. For this purpose, properties of optimal solutions are derived to identify the linearity regions of the k-max function using a local analysis on the edges of the graph. This approach yields all optimal solutions of the underlying problem. The last section shows that, starting from the optimal solutions found by the finite dominating set from the first section, the individual facilities of every solution can be shifted within a certain interval without destroying the optimality of the former solution.

Chapter 7 Most of the presented algorithms solving the p-k-max problem have been implemented in Matlab and the results of the extensive computational tests are presented in this chapter. The tests emphasise the observations made in previous chapters. Moreover, the results prove that the provided theoretical upper bounds on the number of elements in the analysed finite dominating sets are widely overestimating in practice.

Chapter 8 The impact of different levels of importance of the customers is analysed in this section. These levels are modelled by assigning a non-negative weight to every node of the graph. One result is that an optimal solution of the weighted p-k-max problem may have the unwanted effect that high weighted nodes are more likely to become outliers. Two approaches to avoid these problems in modelling outliers in weighted center problems are discussed. The first procedure is a two-stage approach where in a first phase a p-k-max problem is solved for which the original node weights are replaced by their reciprocal values. The second phase determines the optimal new locations with respect to the outliers found in the first phase. In the second approach, the value of the parameter k is not specified with respect to the number of outliers but with respect to a bound on the allowed amount of total outlier weight. Assuming integer weights, the original graph is extended by adding copies of every node such that a node in the original graph is defined to be a weighted outlier only if all of its copies are outliers w.r.t an optimal solution in the transformed graph.

Chapter 9 This last chapter contains a summary of the main results of the thesis. Moreover, some ideas for further research are presented.

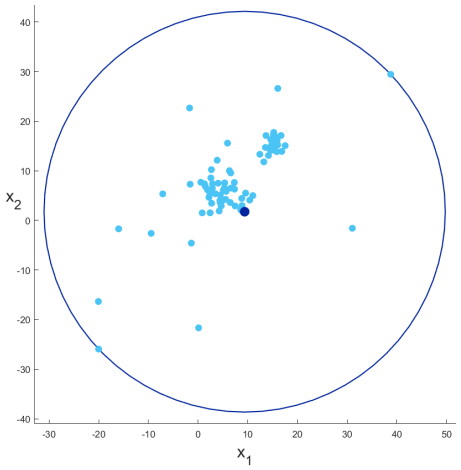
Chapter 2

Modelling Outliers in Center Location Problems

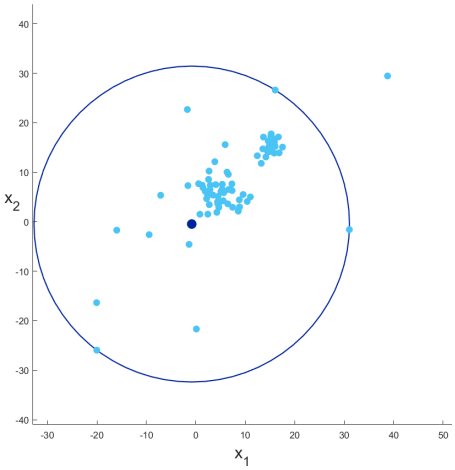
This chapter deals with the problem of modelling the detection of outliers in center location problems and finding an optimal new location with respect to these outliers. The effect of using a k-max function instead of a classical center function is illustrated at the beginning of this chapter. In the first section of this chapter, the general k-max problem is introduced formally and analysed for its mathematical properties. Afterwards, the k-max function is applied to continuous center location problems and the k-max location problem is presented to handle outliers among the existing facilities. Moreover, its connection to the well known ordered median problems is discussed. In the third section, a bi-criteria model for the solution of a general k-max problem is formulated. Some important properties are deduced and motivated by a k-max location problem but proven for general problems with k-max functions. The final section gives an overview of different possibilities to choose an appropriate value for the parameter k of the k-max function.

As explained in the introduction, the k-max function can be used to limit the influence of "far-away"-customers on the optimal solution in center location problems. The k-max function returns the kth largest value of the vector containing in each component the distance between one of the customers and its closest new facility. As a result, the $k-1$ larger distance values and the corresponding existing facilities are not considered for the location of the new facility and will therefore be neglected in the optimisation process. These neglected facilities are called *outliers* throughout this thesis whereas the not neglected facilities are called *center defining facilities*. Only the number of outliers is fixed in advance but it is not known which facilities will be the outliers, this is subject to the optimisation process. As a consequence, outliers will not influence the location of the new facilities and may have a distance to the new facilities that is larger than the optimal objective function value z^*. The outliers with a distance to the new facilities that is larger than z^* are said to be excluded from service because they can not be provided with service within the optimal radius z^*. Note that there may also be outliers that are covered within the service radius because the exclusion in the optimisation process does not automatically lead to an exclusion from service. Thus, there is an important difference between outliers and facilities without service: For a fixed number of outliers, the facilities without service are a subset of the outliers.

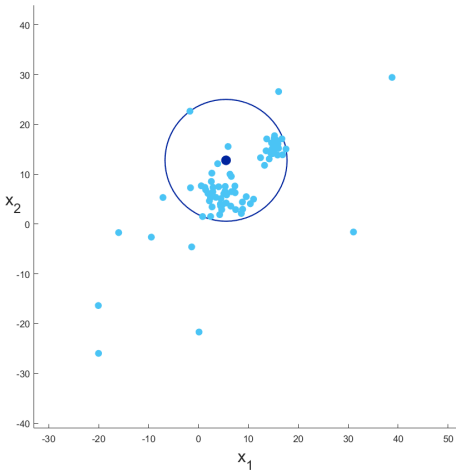
Example 2.1. *An example for a single facility center problem in the plane with $n = 75$ equally weighted existing facilities $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{R}^2$ and new facility $x \in \mathbb{R}^2$ that is*



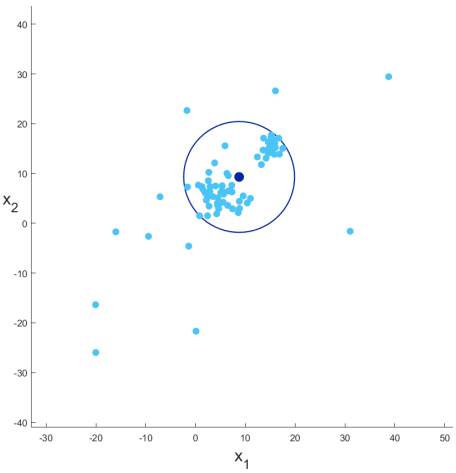
(a) Center solution - no outlier ($k = 1$)



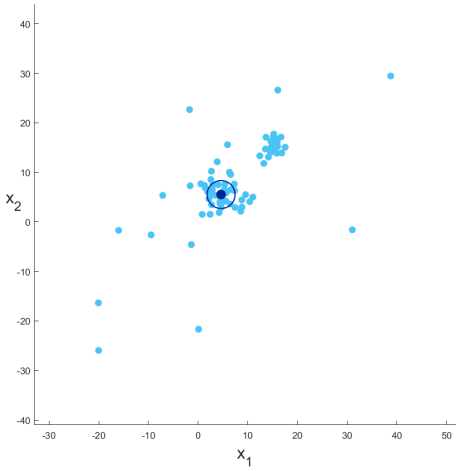
(b) One outlier ($k = 2$)



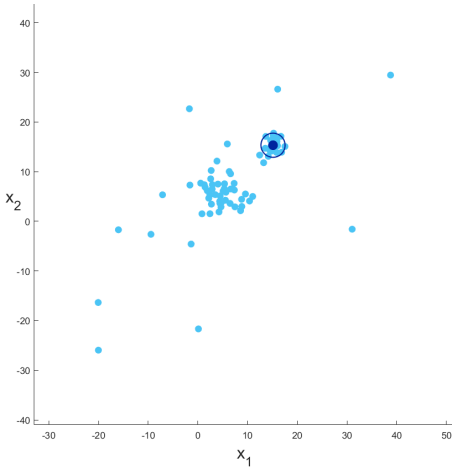
(c) Ten outliers ($k = 11$)



(d) Eleven outliers ($k = 12$)



(e) Fifty four outliers ($k = 55$)



(f) Fifty five outliers ($k = 56$)

Fig. 2.1: The effect of excluding outliers from the derivation on the coverage radius, where the customers are shown in light blue and the new facility in dark blue.

defined by

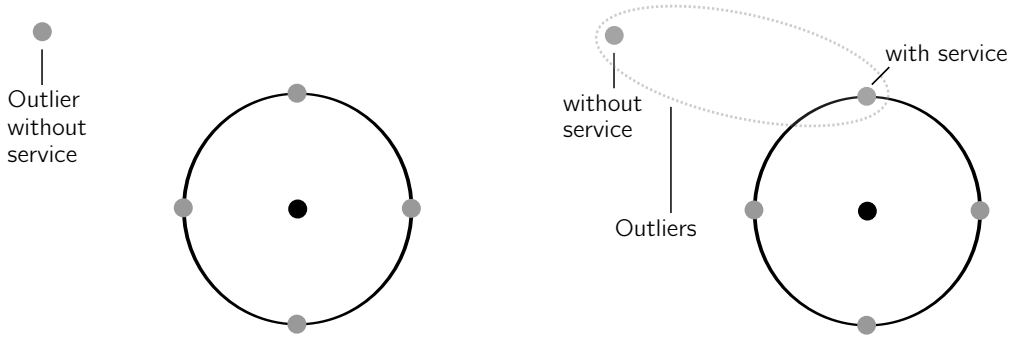
$$\min_{x \in \mathbb{R}^2} \max_{i=1,\dots,n} \sqrt{d(a_{i1}, x_1)^2 + d(a_{i2}, x_2)^2} \tag{2.1}$$

is shown in Figure 2.1(a). The effect of excluding far-away facilities from service on the covering radius can be seen easily in Figures 2.1(b)-(f): The radius of the circle equals the covering radius and therefore represents the optimal objective function value corresponding to the optimal new facility. Figure 2.1(a) shows the optimal solution of a classical center location problem, i.e. none of the existing facilities is excluded from service. It is easy to see that the radius has to be very large to cover all customers, even though the majority of the clients lies very close together in a much smaller area. The remaining subfigures show optimal solutions for a center problem where different numbers of existing facilities are treated as outliers.

*In Figure 2.1(b) the optimal solution of the same center problem is shown where it is allowed to neglect one of the existing facilities in the optimisation process as an outlier. In this case, the outlier is not provided with service because the distance to the new location is larger than z^*. The covering radius is clearly reduced compared to Figure 2.1(a). Using the k-*max *function as objective function means to set $k = 2$, i.e., not the largest distance is minimised but the second largest. The distance between the outlier and the new facility can therefore get arbitrarily large. This effect gets stronger by increasing the number of outliers as can be seen in the following figures. By increasing the number of neglected outliers, the service radius for the covered facilities of course gets smaller or stays at least equal. In this example, the best rate of improvement induced by the neglect of one additional outlier is obtained with the transitions from zero to one and from ten to eleven outliers (see Figures 2.1(a)-(d)). Note that the radius of the circle does not shrink at a constant rate with respect to the number of neglected facilities, but it depends on the distribution of the existing facilities. When allowing more outliers, the covering circle adjusts evermore to the customers that lie close together. Needless to say that it has to be examined how many outliers are reasonable since, for an economic success, the amount of not considered customers should not be overly increased.*

Another important aspect is that facilities which are excluded from service in an optimal solution with a fixed number of outliers may be covered in an optimal solution for a larger amount of outliers. Figures 2.1(e)-(f) show a case where a subset of the outliers in the optimal solution for 54 outliers are facilities covered with service in the optimal solution with 55 outliers, the covering circles of the two solutions do not intersect as they do in Figures 2.1(a),(b) and (c),(d). Hence, the status of a customer of being provided with service or not can change with the number of outliers.

In Example 2.1, the set of outliers equals the set of facilities excluded from service for all values of k. As mentioned before, this is not always the case in practice. The status of being an outlier does not automatically decide on being provided with service or not but only on the influence of this facility on the position of the new location. Figure 2.2 illustrates the difference between outliers and facilities excluded from service in a center problem like (2.1) with five existing facilities. In Figure 2.2(a) one outlier is neglected and this outlier is not provided with service in the optimal solution. In Figure 2.2(b) two outliers are neglected



(a) Optimal solution with one outlier, the outlier is excluded from service

(b) Optimal solution with two outliers, one outlier is covered with service, one is without service

Fig. 2.2: Difference between outliers and facilities excluded from service

which leads to the same optimal solution as before but now only one of the outliers is not provided with service, the other one lies on the covering circle and is therefore served. Note that the set of outliers is not unique in this case, the outlier on the boundary of the circle could be interchanged by any other facility lying also on the boundary.

In the following, a generalisation of the above described problem will be considered in the sense that not only one but p new facilities are to be located. The following sections formulate the center problem with automatic outlier detection formally as a mathematical optimisation problem.

2.1 General k-max Problems

In this section optimisation problems with k-max functions are introduced. After some notations and definitions are given to define the k-max problem, some important properties of this problem are analysed.

The k-max function (see Gorski and Ruzika (2009) and Gorski et al. (2012)) is a generalisation of the well known bottleneck function f_1 defined as

$$f_1(y) = \max_{1 \leq i \leq n} y_i,$$

where $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$ with $n \in \mathbb{N}$ is a n-dimensional vector. The bottleneck function is, besides the sum-objective, one of the most common objectives in combinatorial optimisation and returns the maximum element of y (see, for example, Gross (1959) and other sources in the literature review in Chapter 3). Obviously, with y defining a vector of (weighted or unweighted) distances between every customer and its nearest new facility, the bottleneck function equals a center function (see Definition 2.12).

The impact of the number of outliers on the objective function value described in Example 2.1 can also be observed for other applications, not only in location theory. More precisely, the objective function value of a min-max objective function generally decreases

with the number of outliers. For many combinatorial optimisation problems, this can be modelled by substituting the bottleneck function by a minimisation of a k-max function of the same argument. In the bottleneck path problem (see, for example, Punnen, 1991), a path connecting two vertices has to be found in an edge weighted graph such that the maximum edge weight of a single edge in this path is minimised. The bottleneck function is also used for bottleneck spanning tree problems (for example Camerini, 1978). The bottleneck edge is the highest weighted edge of a graph and hence the aim is to find a spanning tree of the underlying graph in which the bottleneck edge has the smallest possible weight. Generalising these two problems to the k-max concept leads to the minimisation of the kth largest edge weight of a path or of a spanning tree, respectively. A further example are assignment problems where with the k-max function the kth largest costs of the assignment have to be minimised in contrast to the bottleneck assignment problem (see Garfinkel, 1971). In this way, combinatorial optimisation problems using a bottleneck function can be generalised to the minimisation of the kth largest element using a k-max function.

In the following, let Π_n be the set of all permutations of the first n natural numbers, i.e.,

$$\Pi_n := \{\pi : \pi \text{ is a permutation of } \{1, \ldots, n\}\}.$$

The k-max function can be defined as follows.

Definition 2.2 ▶ k-max function

Let $y \in \mathbb{R}^n$ and $k \in \{1, \ldots, n\}$ be arbitrary but fixed. Then the *k-max function* is given by

$$f_k(y) = \underset{1 \le i \le n}{k\text{-max}} (y_i). \tag{kMF}$$

Note that, if the components of y are sorted by a permutation $\sigma \in \Pi_n$ with $y \mapsto y_\sigma$ such that the components of $y_\sigma = (y_{\sigma(1)}, \ldots, y_{\sigma(n)})$ are ordered in non-increasing order, i.e.,

$$y_{\sigma(1)} \ge y_{\sigma(2)} \ge \ldots \ge y_{\sigma(n)}, \tag{2.2}$$

then it holds that

$$\underset{1 \le i \le n}{k\text{-max}} (y_i) = y_{\sigma(k)},$$

i.e., the objective function value is given by the $\sigma(k)$th component of y. The set of all permutations of the components of y satisfying property (2.2) is denoted by $\Sigma(y)$, i.e.

$$\Sigma(y) := \{\sigma \in \Pi_n : y_{\sigma(1)} \ge y_{\sigma(2)} \ge \ldots \ge y_{\sigma(n)}\}$$

for any $y \in \mathbb{R}^n$. Note that the permutations $\sigma \in \Sigma(y)$ always depend on the vector $y \in \mathbb{R}^n$ that is to be sorted and may change whenever y changes. Furthermore, σ does not have to be unique for a fixed y. Since several components of y may be equal, i.e., $y_i = y_j$ for some $i \neq j,\ i, j \in \{1, \ldots, n\}$, it holds for the cardinality of $\Sigma(y)$ that $1 \le |\Sigma(y)| \le n!$. If not specified otherwise, it will be assumed in the following that y_σ with $\sigma \in \Sigma(y)$ is an arbitrary

but fixed ordering of the components of y. The aim of the k-max problem is to find a vector y^* that minimises the k-max function over a predefined feasible set.

Definition 2.3 ▶ k-max problem

Let $k \in \{1, \dots, n\}$, $k \in \mathbb{N}$, be arbitrary but fixed and let $\mathcal{Y} \subseteq \mathbb{R}^n$ be the feasible set of the problem. Then the *k-max problem* is defined as

$$\min_{y \in \mathcal{Y}} k\text{-}\max_{1 \le i \le n} (y_i) = \min_{y \in \mathcal{Y}} y_{\sigma(k)}, \tag{kMP}$$

where $\sigma \in \Sigma(y)$ sorts the elements of y in non-increasing order as given in (2.2).

Whenever the dimension of the argument vector y is clear in the following, for the sake of readability it will be written $k\text{-max}(y)$ instead of $k\text{-max}_{1 \le i \le n}(y_i)$.

A major difficulty when solving a k-max problem is that the permutation σ is not known in general and that it changes depending on y. It is clear that the k-max function is non-linear in general due to the permutation of the elements of y. However, it is possible to show that the k-max function is linear over linearity domains similar to the ordered regions of ordered median problems (see Nickel and Puerto, 2005).

Definition 2.4 ▶ Ordered region

Given a permutation $\sigma \in \Pi_n$, the set

$$O_\sigma = \left\{ y \in \mathbb{R}^n \ : y_{\sigma(1)} \ge y_{\sigma(2)} \ge y_{\sigma(n)} \right\} \subset \mathbb{R}^n$$

is called ordered region (with respect to σ).

O_σ is a region of points in which the sorting of the elements of y (in non-increasing order) does not change. Considering all possible permutations of the n elements of y, it is clear that there are $n!$ ordered regions. Note that the ordered regions are defined by $\binom{n}{2}$ hyperplanes in $\mathbb{R}^n$ of the form $x_i - y_i = 0$ for all $1 \le ij \le n$. This special form of an arrangement of hyperplanes is called a *braid arrangement* (see, for example, Stanley, 1996).

If $y \in \mathbb{R}^2$, two ordered regions are obtained that are the half spaces above and below the bisecting line $g = \{y \in \mathbb{R}^2 : y_1 = y_2\}$. Figure 2.3(c) shows the conic ordered regions in $\mathbb{R}^3$ determined by the three bisecting planes $H_1 = \{y \in \mathbb{R}^3 : y_1 = y_2\}$, $H_2 = \{y \in \mathbb{R}^3 : y_2 = y_3\}$ and $H_3 = \{y \in \mathbb{R}^3 : y_1 = y_3\}$. An important consequence is that the k-max function is linear in each of these regions. However, the ordered regions are in general not the largest possible linearity regions of $f_k(y)$.

Definition 2.5 ▶ Linearity region

Let $k, b \in \{1, \dots, n\}$ be fixed. Then the *linearity region* $L_b^k \subseteq \mathbb{R}^n$ is given by

$$\begin{aligned} L_b^k &= \{y \in \mathbb{R}^n : y_b = k\text{-max}(y)\} \\ &= \{y \in \mathbb{R}^n : \exists\, \sigma \in \Sigma(y) \text{ such that } \sigma(k) = b\} \end{aligned}$$

$$= \bigcup_{\substack{\pi \in \Pi_n \\ \pi(k)=b}} O_\pi,$$

The set of all linearity regions for fixed $k \in \{1, \dots, n\}$ is denoted by $L^k := \left\{ L_b^k : b = 1, \dots, n \right\}$.

In other words, the linearity region L_b^k consists of all $y \in \mathbb{R}^n$ in which the kth largest component of y is at position b of the sorted vector y_σ. The three different expressions for the linearity region given above follow directly from the definition of f_k to be the $\sigma(k)$th component of y for an appropriate permutation σ and the fact that just the kth largest element of y has to keep its position in the sorted vector y_σ. Hence, all ordered regions with the same kth component can be united to a linearity region. The regions $L_b^k \in L^k$ define a subdivision of $\mathbb{R}^n$ where in each of its cells the function f_k is linear because the kth element of the sorted vector y does not change and the k-max objective value is given by exactly this element.

Example 2.6. *Some examples of linearity regions are shown in Figure 2.3. For $y \in \mathbb{R}^2$ and $k = 1, 2$, the k-max function is given in Figure 2.3(a) and (b). As it is easy to see with the contour lines, $f_1(y)$ has two linearity domains separated by the angle bisector of the x_1x_2-plane. In the figure, L_1^1 is the region shown to the right of the angle bisector, L_2^1 lies to its left. $f_2(y)$ defines the same subdivision with $L_1^2 = L_2^1$ and $L_2^2 = L_1^1$. For $y \in \mathbb{R}^3$, $f_k(y)$ has three linearity regions because the k-max function is linear in each ordered region O_π, $\pi \in \Pi_n$, as stated above. Merging all ordered regions with $\pi(k) = b$ for fixed k leads to three linearity regions in total, one for each value of b. For example, for $k = 2$ (see Figure 2.3(e)) it holds that*

$$L_1^2 = O_{(3,1,2)} \cup O_{(2,1,3)}$$

since the second largest element in both ordered regions is y_1. Analogously,

$$L_2^2 = O_{(1,2,3)} \cup O_{(3,2,1)}$$

with y_2 as the second largest element and

$$L_3^2 = O_{(1,3,2)} \cup O_{(2,3,1)}$$

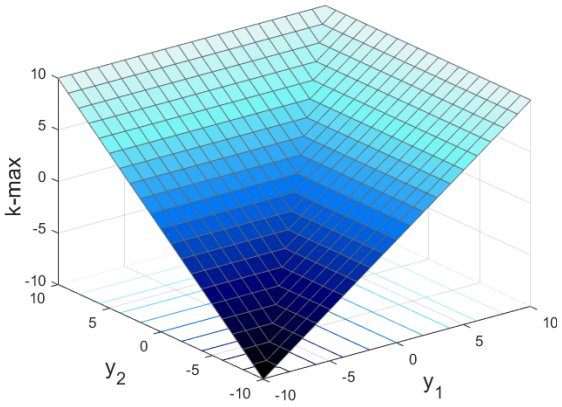
where y_3 is on the second position. The linearity regions for $k = 1$ and $k = 3$ are determined likewise and are shown in Figures 2.3(d) and (f).

When merging several ordered regions to one linearity region it is clear that there are less linearity regions than ordered regions.

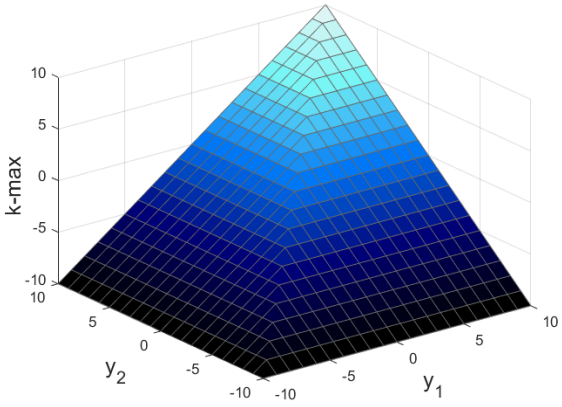
Corollary 2.7 ▶ Number of linearity regions

For $k \in \{1, \dots, n\}$ fixed, the number of linearity regions in $\mathbb{R}^n$ of the k-max function $f_k(y)$ is $|L^k| = n$.

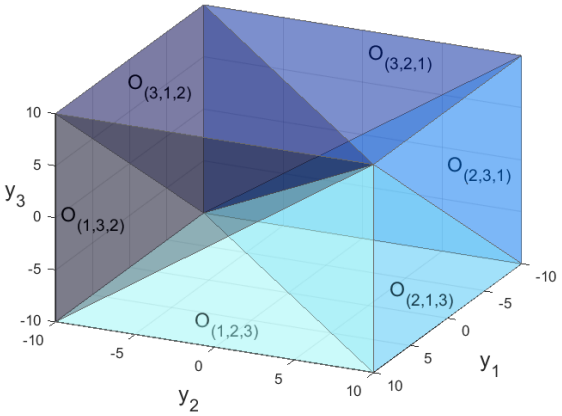
Proof. Follows directly from Definition 2.5. ∎



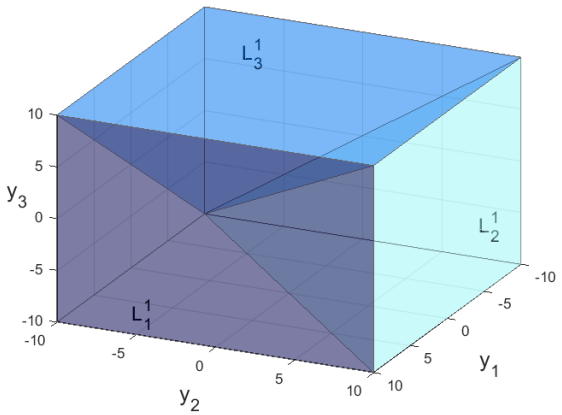
(a) 1-max function in $\mathbb{R}^2$



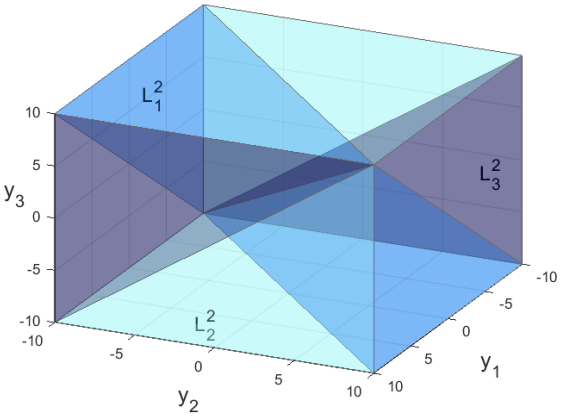
(b) 2-max function in $\mathbb{R}^2$



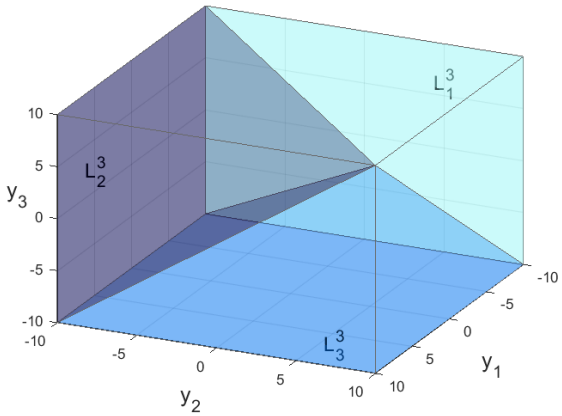
(c) Ordered regions for $y \in \mathbb{R}^3$



(d) Linearity regions for $k = 1$



(e) Linearity regions for $k = 2$



(f) Linearity regions for $k = 3$

Fig. 2.3: k-max function and linearity regions depending on the value of k

The number n of linearity domains can also be determined by the $n!$ different permutations $\pi \in \Pi_n$ where $n-1$ elements of π can take an arbitrary value of the set $\{1,\dots,n\} \setminus \{b\}$ (without repetitions) and just the kth position is fixed to b. Thus, there are $n!/(n-1)! = n$ permutations where $b \in \{1,\dots,n\}$ is on the kth position.

The linearity regions L_b^k for k, b fixed are always connected because all ordered regions intersect in the subspace where $y_1 = y_2 = \dots = y_n$ and so all linearity regions do also contain this subspace that connects the ordered regions O_π with $\pi(k) = b$ to one linearity region. In Figures 2.3(d)-(f) it can be seen that for $k = 1$ the ordered regions that are united to a linearity region have a common facet, i.e., they intersect not only in the subspace where $y_1 = y_2 = y_3$. Moreover, they are obviously convex. For $k = 3$, the same statement holds. For $k = 2$ this is not the case and the ordered regions are not convex. Thus, L_b^k is in general a non-convex set.

Lemma 2.8 ▶ Convexity of the linearity regions

Let $k, b \in \{1,\dots,n\}$ be fixed. Then it holds: The linearity region L_b^k is convex if and only if $k = 1$ or $k = n$.

Proof. The first direction is shown via contraposition, i.e., it is proven that for $k \neq 1$ and $k \neq n$ the linearity region L_b^k is not convex. Let $k \in \{2,\dots,n-1\}$ and let w.l.o.g. $b = n$. Moreover, let

$$\bar{y} = \sum_{i=1}^{k-1} e_i \quad \text{and} \quad \bar{\bar{y}} = \sum_{i=1}^{k-2} e_i + e_k,$$

where e_i is the ith unit vector. Then $\bar{y}, \bar{\bar{y}} \in L_b^k$ since

$$\bar{\sigma} = (1,\dots,k-1,n,k,k+1,\dots,n-1) \in \Sigma(\bar{y})$$

and

$$\bar{\bar{\sigma}} = (1,\dots,k-2,k,n,k-1,k+1,\dots,n-1) \in \Sigma(\bar{\bar{y}}).$$

However,

$$\frac{1}{2}\bar{y} + \frac{1}{2}\bar{\bar{y}} = \sum_{i=1}^{k-2} e_i + \frac{1}{2}e_{k-1} + \frac{1}{2}e_k \notin L_b^k$$

and hence L_b^k is not convex.

Let now $k = 1$, $b \in \{1,\dots,n\}$ fixed and $\bar{y}, \bar{\bar{y}} \in L_b^1$, i.e. $\bar{y}_{\bar{\sigma}(1)} = y_b$ with $\bar{\sigma} \in \Sigma(\bar{y})$ and $\bar{\bar{y}}_{\bar{\bar{\sigma}}(1)} = y_b$ with $\bar{\bar{\sigma}} \in \Sigma(\bar{\bar{y}})$. For $\lambda \in [0,1]$ it holds

$$\lambda\bar{y} + (1-\lambda)\bar{\bar{y}} = \lambda \begin{pmatrix} \bar{y}_1 \\ \vdots \\ \bar{y}_{\bar{\sigma}(1)} \\ \vdots \\ \bar{y}_n \end{pmatrix} + (1-\lambda) \begin{pmatrix} \bar{\bar{y}}_1 \\ \vdots \\ \bar{\bar{y}}_{\bar{\bar{\sigma}}(1)} \\ \vdots \\ \bar{\bar{y}}_n \end{pmatrix} = \begin{pmatrix} y'_1 \\ \vdots \\ y'_b \\ \vdots \\ y'_n \end{pmatrix},$$

where $\bar{\sigma}(1) = \bar{\bar{\sigma}}(1) = b$ implies that

$$\begin{aligned} y'_b &= \lambda \bar{y}_{\bar{\sigma}(1)} + (1-\lambda)\bar{\bar{y}}_{\bar{\bar{\sigma}}(1)} \\ &= \lambda \cdot \max_{1\le i\le n} \bar{y}_i + (1-\lambda) \cdot \max_{1\le i\le n} \bar{\bar{y}}_i \\ &= \max_{1\le i\le n} \{\lambda \bar{y}_i + (1-\lambda)\bar{\bar{y}}_i\} \\ &= \max_{1\le i\le n} y'_i =: y'_{\sigma'(1)}, \end{aligned}$$

with $\sigma' \in \Sigma(y')$. Therefore, $y' = \lambda\bar{y} + (1-\lambda)\bar{\bar{y}} \in L_b^1$ for all $\lambda \in [0,1]$ and for all $b \in \{1,\dots,n\}$ and thus L_b^1 is convex.

Let $k = n$, $b \in \{1,\dots,n\}$ fixed and $\bar{y}, \bar{\bar{y}} \in L_b^n$, i.e. $\bar{y}_{\bar{\sigma}(n)} = y_b$ for $\bar{\sigma} \in \Sigma(\bar{y})$ and $\bar{\bar{y}}_{\bar{\bar{\sigma}}(n)} = y_b$ for $\bar{\bar{\sigma}} \in \Sigma(\bar{\bar{y}})$. For $\lambda \in [0,1]$ it holds analogous to the case $k = 1$

$$\begin{aligned} y'_b &= \lambda \bar{y}_{\bar{\sigma}(n)} + (1-\lambda)\bar{\bar{y}}_{\bar{\bar{\sigma}}(n)} \\ &= \lambda \cdot \min_{1\le i\le n} \bar{y}_i + (1-\lambda) \cdot \min_{1\le i\le n} \bar{\bar{y}}_i \\ &= \min_{1\le i\le n} \{\lambda \bar{y}_i + (1-\lambda)\bar{\bar{y}}_i\} \\ &= \min_{1\le i\le n} y'_i =: y'_{\sigma'(n)}, \end{aligned}$$

with $\sigma' \in \Sigma(y')$. Therefore, $y' = \lambda\bar{y} + (1-\lambda)\bar{\bar{y}} \in L_b^n$ for all $\lambda \in [0,1]$ and for all $b \in \{1,\dots,n\}$ and thus L_b^n is convex. ∎

The following results provide further important properties of the k-max function $f_k(y)$.

Lemma 2.9 ▶ Continuity of (kMF)
The k-max function $f_k(y)$ is continuous on $\mathbb{R}^n$.

Proof. Let $\sigma \in \Sigma(y)$ be an arbitrary but fixed permutation and

$$O_\sigma = \{y \in \mathbb{R}^n : y_{\sigma(1)} \ge \dots \ge y_{\sigma(n)}\}$$

be an ordered region, i.e., the permutation σ of the elements in y is constant on O_σ. Therefore, the k-max function simplifies here to

$$f_k(y) = \underset{1\le i\le n}{k\text{-max}}\,(y) = y_{\sigma(k)} \quad \text{for all} \quad y \in O_\sigma.$$

On the ordered region for a fixed permutation σ, the objective function value is given by the $\sigma(k)$th element of y because the permutation is the same in every point $y \in O_\sigma$. Thus, the k-max function is continuous on the interior of every ordered region. Now the continuity on the boundary of O_σ has to be analysed.

Let $\partial(O_\sigma)$ be the boundary of O_σ and suppose that $\bar{y} \in \partial(O_\sigma)$ is a point on the boundary of O_σ, i.e.,

$$\Sigma(\bar{y}) = \{\sigma_1 = \sigma, \sigma_2, \ldots, \sigma_l\} \quad \text{with} \quad |\Sigma(\bar{y})| = l \quad \text{and} \quad l \geq 2.$$

By the definition of ordered regions, this implies that $\bar{y}_{\sigma_u(i)} = \bar{y}_{\sigma_v(i)}$ for all $u, v \in \{1, \ldots, l\}$ and for all $i \in \{1, \ldots, n\}$. In particular,

$$\bar{y}_{\sigma_u(k)} = \bar{y}_{\sigma_v(k)} = k\text{-max}(\bar{y}) \quad \text{for all} \quad u, v \in \{1, \ldots, l\}.$$

Consider a sequence

$$(y_b) \in \mathbb{R}^n \quad \text{with} \quad \lim_{b\to\infty} y_b = \bar{y}.$$

Let (y_{b_u}), $u \in \{1, \ldots, l\}$, be an infinite subsequence of (y_b) consisting of all iterates of y_b that are in O_{σ_u}. Note that theses subsequences are not necessarily disjoint, and not for all $u \in \{1, \ldots, l\}$ an infinite subsequence of (y_b) must exist. It holds

$$\lim_{b_u\to\infty} y_{b_u} = \bar{y},$$

and in particular

$$\lim_{b_u\to\infty} k\text{-max}(y_{b_u}) = \lim_{b_u\to\infty} (y_{b_u})_{\sigma_u(k)} = \bar{y}_{\sigma_u(k)} = k\text{-max}(\bar{y}).$$

Using the same argument, every infinite subsequence of (y_b) has at least one infinite subsequence $(y_{b_{\bar{u}}})$ (that is a subsequence of (y_{b_u}) for some $u \in \{1, \ldots, l\}$) for which

$$\lim_{b_{\bar{u}}\to\infty} k\text{-max}(y_{b_{\bar{u}}}) = k\text{-max}(\bar{y}).$$

This implies that

$$\lim_{b\to\infty} k\text{-max}(y_b) = k\text{-max}(\bar{y}),$$

since every subsequence k-max(y_{b_u}) has a further subsequence k-max$(y_{b_{\bar{u}}})$ which converges to k-max$(\bar{y})$ (see, for example, Proposition A.6 in Osborne (2014)).

As a consequence, there are no discontinuities of the objective function neither on the boundary nor in the interior of the ordered regions and therefore the k-max function is continuous for all $y \in \mathbb{R}^n$. ∎

Lemma 2.10

Let $\sigma \in \Sigma(y)$ be a fixed permutation. Then the k-max function satisfies $f_k(y) = f_k(y_\sigma)$ for every $y \in \mathbb{R}^n$.

Proof. Let $y \in \mathbb{R}^n$. Moreover, let the permutation σ map y to y_σ such that

$$y_{\sigma(1)} \geq y_{\sigma(2)} \geq \ldots \geq y_{\sigma(n)}.$$

From Definition 2.3 it follows immediately that

$$f_k(y) = k\text{-max}(y) = y_{\sigma(k)} = (y_\sigma)_k = k\text{-max}(y_\sigma) = f_k(y_\sigma),$$

which proves the statement. ∎

Lemma 2.11 ▶ Convexity of (kMF)
The k-max function $f_k(y)$ is convex in $\mathbb{R}^n$ if and only if $k = 1$.

Proof. The first direction is shown via contraposition, i.e., it is proven that the k-max function is non-convex for all $k \in \{2, \dots, n\}$. Let $k \in \{2, \dots, n\}$ be fixed, and consider the two points

$$y_1 = \sum_{i=1}^{k-1} e_i \quad \text{and} \quad y_2 = \sum_{i=1}^{k-2} e_i + e_k,$$

where $e_i \in \mathbb{R}^n$ is the ith unit vector. Clearly, $f_k(y_1) = f_k(y_2) = 0$. Now consider the point

$$y := \frac{1}{2}y_1 + \frac{1}{2}y_2 = \sum_{i=1}^{k-2} e_i + \frac{1}{2}e_{k-1} + \frac{1}{2}e_k.$$

Then $f_k(y) = \frac{1}{2}$ and hence

$$0 = \frac{1}{2}f_k(y_1) + \frac{1}{2}f_k(y_2) < f_k\left(\frac{1}{2}y_1 + \frac{1}{2}y_2\right) = \frac{1}{2}.$$

That means that $f_k(y)$ is not convex if $k \neq 1$ and therefore the statement holds.

For $k = 1$, the classical max function is obtained and it holds $1\text{-max}(y) = \max(y)$. As it is already known (see, e.g., Rockafellar, 1970), it clearly satisfies

$$\max\{\lambda y_1 + (1-\lambda)y_2\} \leq \lambda \max y_1 + (1-\lambda)\max y_2$$

for all $y_1, y_2 \in \mathbb{R}^n, \lambda \in [0, 1]$. ∎

Note that all the properties derived in this section can also be shown as an implication of the corresponding properties of the ordered median function (e.g., Nickel and Puerto, 2005).

2.2 Continuous p-k-max Location Problems

In this section, the general k-max function as introduced in the last section will be adapted to continuous location problems. A short overview with some notations and definitions is given to introduce the k-max location problem formally. Afterwards, the relation between a p-k-max problem and an ordered median problem is analysed. At first, the closely related p-center problem is presented shortly.

Let $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{R}^2$ be a finite set of n existing facilities (customers) and let $X = \{x_1, \ldots, x_p\} \subseteq \mathbb{R}^2$ be a finite set of p new facilities to locate, i.e., the new facilities shall be placed with respect to the customers in the Euclidean plane. With $w : A \mapsto \mathbb{R}_+$ a positive weight $w_i > 0$ is assigned to every existing facility a_i, $i = 1, \ldots, n$, i.e., the demand or the importance of a customer is expressed by different weights. Moreover, $\mathcal{X} \subseteq \mathbb{R}^2$ is the feasible area for the new facilities.

The location of the new facilities is based on the weighted distances between the individual customers and their corresponding closest new facility. A distance measure is in general given by a so called gauge γ_B. A gauge is a Minkowski functional (see, for example, Thompson, 1996) defined on a compact, convex set B that contains the origin in its interior and is defined (following, for example, Nickel and Puerto, 2005) as

$$\gamma_B(x) = \min\{r > 0 : x \in rB\}. \tag{2.3}$$

Then, the weighted distance between the set of existing facilities and the set of new facilities is given by the vector

$$d^w(A, X) := (d^w(a_1, X), \ldots, d^w(a_n, X)) = (w_1 d(a_1, X), \ldots, w_n d(a_n, X))$$

with

$$d(a_i, X) := \min_{x_j \in X} \gamma_B(a_i - x_j),$$

i.e., $d(a_i, X)$ is the shortest distance between an existing facility $a_i \in A$ and its closest new facility $x_j \in X$. A gauge thus defines a not necessarily symmetric distance function. If γ_B is symmetric, then γ_B is a norm (see Rockafellar, 1970). The set B defines the unit ball of the gauge γ_B and it holds that $\gamma_B(x) = 1$ if and only if x lies on the boundary of B. If $x \notin B$, then the value of $\gamma_B(x)$ is given by the factor by which the unit ball B has to be inflated or shrunk until the point x is on its boundary.

Applying the k-max function to location problems leads to a generalisation of the well known center location problem. The center function aims to locate one or more new facilities with respect to some existing customers such that the largest distance between every customer and its nearest new facility is minimised.

Definition 2.12 ▶ p-Center Problem

Let $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{R}^2$ be the set of existing facilities, let $\mathcal{X} \subseteq \mathbb{R}^2$ be the feasible area for new facilities and let $X = \{x_1, \ldots, x_p\} \subseteq \mathcal{X}$ with $p \in \mathbb{N}$ denote the unknown set of new facilities to locate. Then, the *p-center problem* is given by

$$\min_{X \subseteq \mathcal{X}} f_1(X) = \min_{X \subseteq \mathcal{X}} \max_{i=1,\ldots,n} d^w(a_i, X).$$

The problem was introduced by Sylvester (1857) and has been investigated extensively since then. For a more detailed overview of the topic see Chapter 3. To reduce the influence of far-away customers on the optimal solution of the problem as described in Section 2.1,

the center function can be replaced by a k-max function f_k. Thus, instead of the largest distance, the kth largest distance between any customer and its closest new facility is used as the decision criterion. Roughly spoken, the k-max function applied to location problems equals a center function with automatic outlier detection.

The parameter $k \in \{1, \ldots, n\}$ specifies the kth largest distance to be minimised and the parameter $p \in \mathbb{N}$ gives the number of new facilities to locate. With these components the p-k-max location problem can be defined as follows.

Definition 2.13 ▶ p-k-max location problem

Let $A = \{a_1, \ldots, a_n\} \subseteq \mathbb{R}^2$ be the set of existing facilities, let $\mathcal{X} \subseteq \mathbb{R}^2$ be the feasible area for new facilities and let $X = \{x_1, \ldots, x_p\} \subseteq \mathcal{X}$ with $p \in \mathbb{N}$ denote the unknown set of new facilities to locate. For $k \in \{1, \ldots, n\}$, the *p-k-max problem* is given by

$$\min_{X \subseteq \mathcal{X}} k\text{-max}(d^w(A, X)) = \min_{X \subseteq \mathcal{X}} w_{\sigma(k)} d(a_{\sigma(k)}, X), \tag{pkMP}$$

where $\sigma \in \Sigma(X)$ is a permutation of the existing facilities depending on X, i.e., it satisfies

$$w_{\sigma(1)} d(a_{\sigma(1)}, X) \geq w_{\sigma(2)} d(a_{\sigma(2)}, X) \geq \ldots \geq w_{\sigma(n)} d(a_{\sigma(n)}, X).$$

Note that the permutation σ always depends on the current solution X and may change whenever X changes. Hence, in a feasible solution X with optimal objective value z, a set of outliers

$$A_{k-1} := \left\{a_{\sigma(1)}, \ldots, a_{\sigma(k-1)}\right\}$$

is a subset of the existing facilities A of cardinality $k-1$ such that all facilities of this set have a weighted distance larger then or equal to z to X. Note that the set A_{k-1} of a feasible solution X does not have to be unique as the permutation $\sigma \in \Sigma(X)$ does not have to be unique. Facilities in A_{k-1} have no influence on the position of the new facility since they are neglected in the optimisation process. The center defining facilities of a feasible solution are given by the set $A \setminus A_{k-1}$.

Remark 2.14. *It is also possible to include zero weighted existing facilities into the problem, i.e., customers $a_i \in A$ with $w_i = 0$. From a practical point of view, this is often not useful since a zero weighted facility corresponds to an existing facility that has no demand and should therefore not have any influence on the location of the new facility. However, it is sometimes useful for theoretical reasons to include this case. Note that existing points $a_i \in A$ with weights w_i equal to zero do not contribute to the objective function as it holds that $w_i d(a_i, x) = 0$. Thus, the zero-weighted facilities will usually not be outliers, as long as the parameter k is smaller than the number of positively weighted customers minus p. Note that it is reasonable to assume that not all customers are weighted by* 0.

If $p \geq n$ the optimal objective value of the p-k-max problem is always equal to zero since at least one new facility can be opened in every existing facility a_i, $i = 1, \ldots, n$, and the remaining new facilities can be placed in arbitrary points in the plane. The single facility

problem is a special case of the above problem for $p = 1$. In this case the problem will be referred to as the k-max problem instead of 1-k-max problem.

As shown for the general k-max function in the previous section, the objective function is in general *non-linear*. This holds of course also for the k-max location objective function. For at least piecewise linear gauges (for example block norms), it can be shown that the k-max function is piecewise linear. However, the linearity regions of a k-max location function are more complicated than those of the general k-max function because the linearity of $k\text{-max}(d^w(A, X))$ does not only depend on the permutation σ but also on the domains on which the underlying norm or gauge is linear. Thus, also the ordered regions are not necessarily convex, as it can be seen considering the example of two points in the plane equipped with rectilinear distances (see Nickel and Puerto, 2005). The linearity regions are derived in Section 6.3 for p-k-max problems on networks, but the approach can similarly be applied to the continuous case.

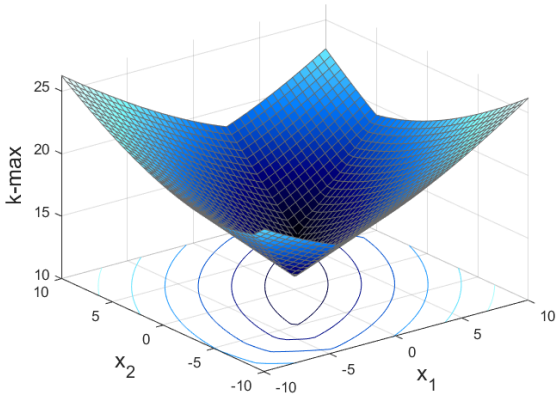
The continuity-result given in Lemma 2.9 also holds for k-max functions in the context of location problems and when general gauges are used for the measurement of distances. More precisely, the gauge distance between an existing facility $a_i \in A$ and a new facility $x \in \mathcal{X} \subseteq \mathbb{R}^2$ is continuous in x (see Durier and Michelot, 1985). If several new facilities are to be located, then $d(v_i, X)$ is defined as the minimal distance to a location in X. Therefore, as a composition of continuous functions, the k-max function for $0 < p < \infty$ new facilities is *continuous*.

Also Lemma 2.11 can be transferred to the k-max location function, i.e., the objective function of the p-k-max location problem is *convex* if and only if $k = 1$. This holds as the gauge γ_B is convex since it is positively homogeneous and satisfies the triangle inequality. Thus, the convexity of the overall function depends on the convexity of the general k-max function which is only given for $k = 1$. The above listed properties are illustrated in Figure 2.4 and the following example.

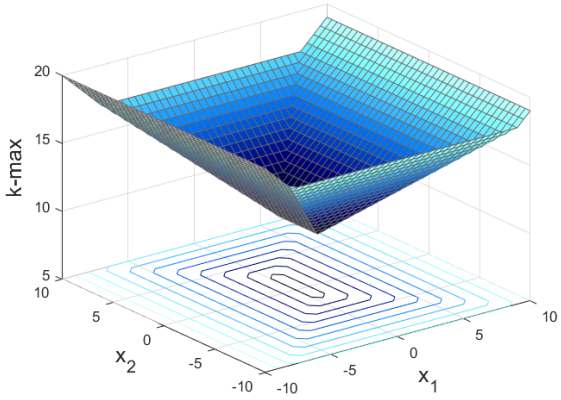
Example 2.15. *Figure 2.4 shows the* k-max *function of an unweighted* k-max *location problem, i.e.,* $w_i = 1$ *for all* $i = 1, \ldots, n$ *and* $p = 1$*, with* $n = 10$ *existing facilities for different values of* k *and with Euclidean distances on the left resp. the same example with distances measured by the infinity norm on the right. Subfigures (a) and (b) illustrate the convexity of the objective function for* $k = 1$*, whereas the functions for* $k > 1$ *are obviously non-convex (see Subfigures (c) and (d)). In case of the infinity norm, the linearity regions can be seen, whereas there are of course no linearity regions for the Euclidean norm.*

A further property of the k-max *function can be seen in subfigures (e) and (f): For* $k = n = 10$*, the minima of the objective function are attained in the existing facilities since the* n*th largest distance can be brought to* 0 *by locating the new facility in one of the existing facilities.*

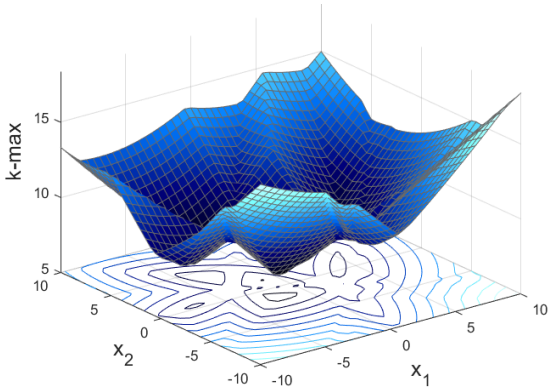
The p-k-max problem has to be distinguished from a maximum covering problem, which does in general also not enforce the coverage of all existing facilities but seeks a new location such that the maximum number of customers is covered with service within a fixed radius (see, for example, Church and Revelle, 1974). In the here stated outlier problem, a minimal



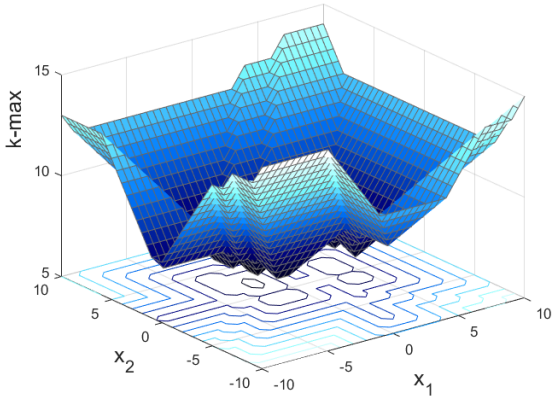
(a) $k = 1$, Euclidean norm



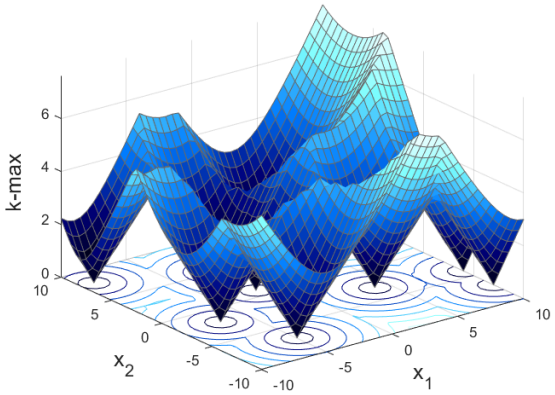
(b) $k = 1$, infinity norm



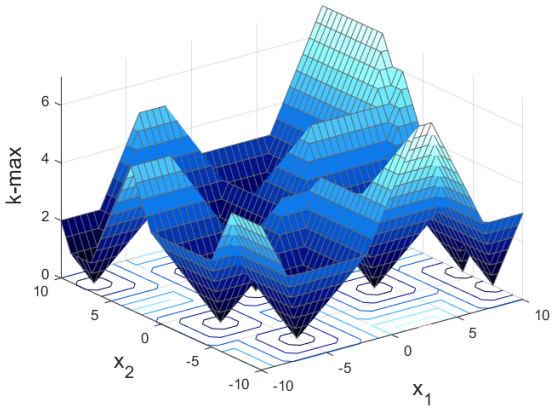
(c) $k = 5$, Euclidean norm



(d) $k = 5$, infinity norm



(e) $k = 10$, Euclidean norm



(f) $k = 10$, infinity norm

Fig. 2.4: k-max function with $n = 10$, $p = 1$ and $w_1 = \ldots = w_{10} = 1$

number of customers to cover is fixed and the value of the covering radius is obtained by the optimal objective function value as a result of the optimisation. Roughly spoken, constraints and objective functions are kind of reversed in these two problems. Moreover, the k-max function is closely related to the well known ordered median function (see, e.g., Nickel and Puerto, 2005).

Definition 2.16 ▶ Ordered median function

An *ordered median function* $f_\lambda : \mathbb{R}^n \to \mathbb{R}$ is defined by the scalar product

$$f_\lambda(d^w(A, X)) = \langle \lambda, d^w_\tau(A, X) \rangle$$

for $\lambda = (\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^n$, where $\tau \in \Pi_n$ such that $d^w(A, X) \mapsto d^w_\tau(A, X)$ with

$$d^w_\tau(A, X) = \left(d^w(a_{\tau(1)}, X), \ldots, d^w(a_{\tau(n)}, X)\right)$$

and

$$d^w(a_{\tau(1)}, X) \leq d^w(a_{\tau(2)}, X) \leq \ldots \leq d^w(a_{\tau(n)}, X).$$

The set of all ordered median functions is denoted by $\mathrm{OMF}(n)$.

The ordered median function is a weighted average of ordered elements. The allocation of weights λ to the sorted argument vector allows to compensate very small resp. large elements compared to the other elements. It provides a common theory for many location problems because they can be interpreted as special cases of the ordered median problem for a suitable fixed vector λ. Different well known instances of the ordered median problem are

- $\lambda = (0, \ldots, 0, 1)$: center problem (see Chapter 3)
- $\lambda = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)$: median problem (see Hakimi, 1964)
- $\lambda = (\alpha, \ldots, \alpha, 1)$: α-centdian problem (see Halpern, 1978)
- $\lambda = (0, \ldots, 0, 1, \overset{k}{\ldots}, 1)$: k-centrum problem (see Slater, 1982)
- $\lambda = (0, \ldots, 0, 1, \ldots, 1, 0, \ldots, 0)$: trimmed mean problem (see Rosenberger and Gasko, 1983)

The field of ordered median problems is well investigated. For more information see, for example, Nickel and Puerto (2005) or Laporte et al. (2015) as well as the references in Chapter 3. As it can be easily seen, the k-max function is an ordered median function with a special structure of the vector of weights λ:

$$
\begin{aligned}
k\text{-max}\left(d^w(A,X)\right) &= d^w(a_{\sigma(k)},X)\\
&= \sum_{i=1}^{n}\lambda_i\cdot d^w(a_{\sigma(i)},X) \quad \text{with } \lambda_k=1,\ \lambda_i=0 \text{ for } i\in\{1\ldots,n\}\setminus\{k\}\\
&= \langle\lambda, d^w_\sigma(A,X)\rangle \qquad\qquad \text{with } \lambda_k=1,\ \lambda_i=0 \text{ for } i\in\{1\ldots,n\}\setminus\{k\}\\
&= \left\langle\bar{\lambda}, d^w_\tau(A,X)\right\rangle, \qquad\quad \bar{\lambda}_{n-k+1}=1,\ \bar{\lambda}_i=0\ i\in\{1\ldots,n\}\setminus\{n-k+1\}\\
&= f_{\bar{\lambda}}(d^w(A,X)),
\end{aligned}
$$

where $\sigma\in\Sigma(X)$ and $\tau\in\Pi_n$ with $d^w(a_{\tau(1)},X)\leq\ldots\leq d^w(a_{\tau(n)},X)$. Note that τ in Definition 2.16 sorts the elements of y in the reverse order as σ for the k-max problems. Otherwise the indices for the kth largest element would be $\sigma(n-k+1)$ which would be inconvenient in the following.

The ordered median function is non-linear in general and non-convex except for λ with $0\leq\lambda_1\leq\lambda_2\leq\ldots\leq\lambda_n$. Therefore, it is difficult to solve the k-max problem with known approaches for ordered median problems since most of them deal with the convex case. Similar to the approach in Section 6.3, linearity regions of the ordered median function with underlying polyhedral gauge can be identified, see Rodríguez-Chía et al. (2000). These regions are called *ordered elementary convex sets* and provide a subdivision of the $\mathbb{R}^2$ such that on each non-empty subset of this subdivision the permutation is constant and the gauge is linear. It can be shown that the entire set of optimal solutions of the ordered median problem coincides with some ordered elementary convex sets, see Puerto and Fernández (2000). The topology of the subdivision depends on the underlying gauge and the vector λ.

It should be mentioned that most procedures to solve ordered median problems are not intended to compete against approaches developed for more special problems that are special cases of the ordered median model. Such methods of course take advantage of the structure of these particular problems. The aim of the ordered median analysis is rather to obtain a good performance for more general types of problems. Specific solution approaches designed to solve the p-k-max problem are analysed in the following chapters of this thesis.

2.3 A Bi-criteria Model

As it could be seen in the introductory Example 2.1 of a k-max location problem, the radius of the covering circle and therefore the k-max objective function value of the location problem is non-increasing when the parameter k of the k-max function increases. This effect can be used to analyse the quality of the solution of the underlying location problem. The profitability of excluding outliers from the optimisation process highly depends on the value of the parameter k. The decision maker has to choose an appropriate value for k and with this he decides on the number $k-1$ of outliers and therefore on the maximum number of clients that is excluded from service (or conversely on the minimum number of $n-k+1$

customers that have to be covered within the optimal service radius). Of course, the choice of an appropriate value of the parameter k is an important issue concerning the success of the project which is influenced by the location of the new facility.

As already mentioned, the general concept of k-max problems can also be applied to many other (combinatorial) optimisation problems since it is often useful to minimise the kth largest element instead of the largest element when outliers are present. The described relation between the parameter k and the k-max objective function does not depend on the underlying type of optimisation problem (location, shortest path, assignment, spanning tree, etc.) but is valid for all fields of application. Therefore, the k-max problem will be analysed further in its general form (kMP) as given in Definition 2.3 and the k-max location problem (pkMP) is used to explain the general results descriptively.

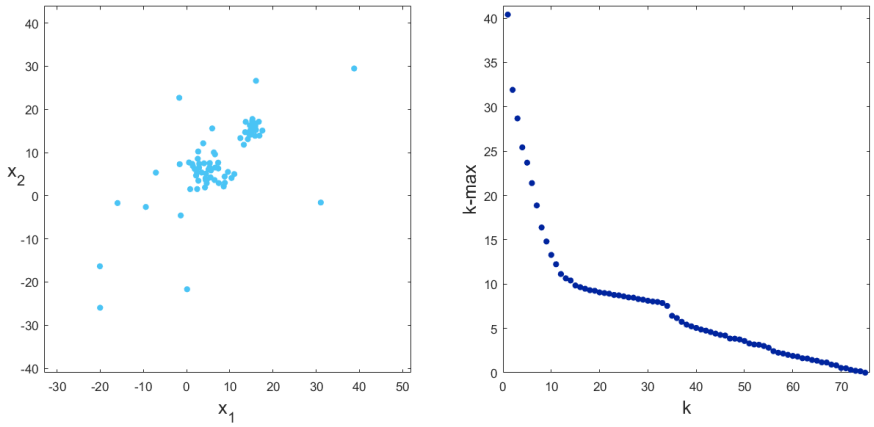


Fig. 2.5: Left: Example of a distribution of equally weighted existing facilities in the plane $\mathbb{R}^2$ (c.f. Example 2.1); right: Trade-off between the number of outliers $(k-1)$ and the optimal k-max objective function value for the distribution on the left

For the existing facilities shown on the left, Figure 2.5 illustrates the relation between the number of outliers given by the value of k from the k-max function and the corresponding k-max objective value for a continuous 1-center problem with euclidean distances. In this example, the curve is rather steep for $k \in \{1, \ldots, 11\}$, so that these first values for k lead to about 75% of the possible savings in total. For $k \in \{12, \ldots, 75\}$, the curve is rather flat and the advantage of excluding clients from service weakens. Obviously, this behaviour depends on the distribution of the underlying points and can not be generalised to other sets of existing facilities. However, the curve showing the relation between the parameter k and the objective function value of a general k-max problem is always non-increasing, no matter of what type the underlying problem is. The monotonicity-property already illustrated in Example 2.1 is shown formally with the following lemma.

Lemma 2.17 ▶ Monotonicity of the k-max value

Let y_k be the optimal solution for a general k-max problem (kMP) with fixed k. Then, for $k = 1, \ldots, n$, the k-max value $f_k(y_k)$ is non-increasing with k.

Proof. Let $y_{\bar{k}} \in \mathcal{Y}$ be an optimal solution of the $\bar{k}$-max problem (kMP) with optimal objective function value $\bar{z} = \bar{k}\text{-max}(y_{\bar{k}})$.

Assume that there exists a solution $y_{\tilde{k}} \in \mathcal{Y}$ of the $\tilde{k}$-max problem with $\tilde{k} < \bar{k}$ and optimal objective function value $\tilde{z} = \tilde{k}\text{-max}(y_{\tilde{k}})$ such that $\tilde{k}\text{-max}(y_{\tilde{k}}) < \bar{k}\text{-max}(y_{\tilde{k}})$. Since $y_{\tilde{k}}$ is also feasible for the $\bar{k}$-max problem, it holds

$$\bar{k}\text{-max}(y_{\tilde{k}}) < \bar{k}\text{-max}(y_{\bar{k}}) = \bar{z}.$$

This leads directly to a contradiction to the optimality of $y_{\bar{k}}$ for the $\bar{k}$-max problem. ∎

In general, the equality $f_{\tilde{k}}(y_{\tilde{k}}) = f_{\bar{k}}(y_{\bar{k}})$ of the optimal objective values for directly consecutive parameters $\bar{k} := \tilde{k} + 1$ can only occur if

$$y_{\bar{k}_{\bar{\sigma}(\bar{k})}} = y_{\tilde{k}_{\tilde{\sigma}(\tilde{k})}} \quad \text{for} \quad \bar{\sigma} \in \Sigma(y_{\bar{k}}) \quad \text{and} \quad \tilde{\sigma} \in \Sigma(y_{\tilde{k}}).$$

For k-max location problems this leads to the equality $f_{\tilde{k}}(X_{\tilde{k}}) = f_{\bar{k}}(X_{\bar{k}})$ where $X_{\bar{k}}$ and $X_{\tilde{k}}$ are optimal solutions of the $\bar{k}$-max resp. $\tilde{k}$-max problem on a fixed set of existing facilities. Hence, the $\bar{k}$th and $\tilde{k}$th element of the sorted distance vectors are equal. This is only possible if $X_{\bar{k}}$ also covers some of the outliers with service and therefore more than the needed $n - \bar{k} + 1$ customers such that the covering circles are equal for at least one optimal solution for $\bar{k}$ and $\tilde{k}$.

Figure 2.6 shows an example for $p = 1$ with 10 equally weighted existing facilities where the optimal solution for $\tilde{k} = 3$ and $\bar{k} = 4$ is given. The aim for $\bar{k} = 4$ is to cover seven facilities but the minimal covering circle for seven points covers even eight points. Thus, the coverage circle and therefore also the optimal solution for these two values of k are equal. This equality can also be seen in Figure 2.7 showing the curve that illustrates the relation between k and the corresponding objective value of (kMP) for the existing facilities in Figure 2.6. The curve is not strictly decreasing for $k = 3, 4$.

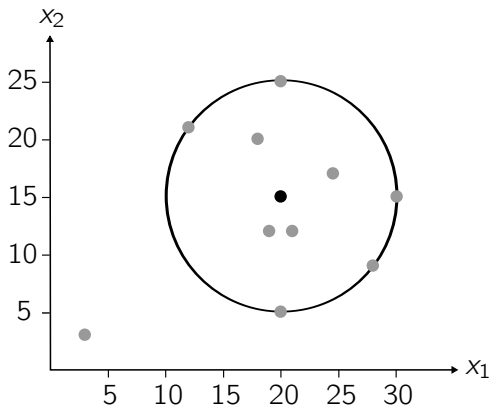


Fig. 2.6: Example with 10 customers (grey dots) and optimal solution (black dot) for $k = 3$ and $k = 4$

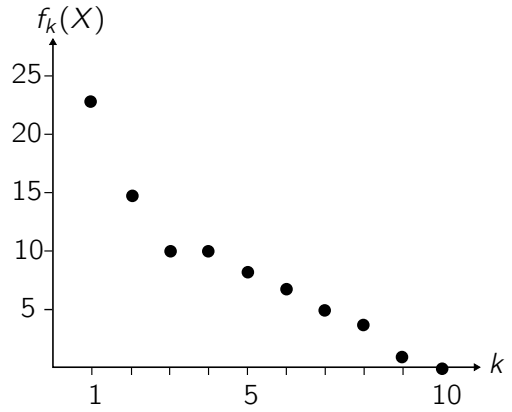


Fig. 2.7: Relation between k and the optimal k-max value for the existing facilities shown on the left.

It can be concluded that the parameter of the k-max location function influences the

solution in the following way: A small value of k ensures on the one hand that many clients are served by the new facility within the computed coverage radius, but on the other hand it may lead to large distances between the majority of customers and the new location. Conversely, a large value of k can be expected to lead to a small service radius and therefore a solution with a smaller maximum distance, but it usually also results in a smaller subset of clients served by this new facility. Thus, it is important to find a satisfying compromise between the two conflicting goals to maximise the number of customers served within a reasonable coverage radius and to minimise the maximum distance to the new facility for these customers. These conflicting goals can be combined in a bi-criteria optimisation model of the form

$$\text{(BPL)} \begin{cases} \min & k\text{-max}(d^w(A, X)) \\ \min & k \\ \text{s.t.} & X \subseteq \mathcal{X} \\ & k \in \{1, \dots, n\} \end{cases}.$$

Problem (BPL) combines the minimisation of the kth largest distance between the existing facilities $A = \{a_1, \dots, a_n\}$ and the new locations X and the minimisation of the value of the parameter k with the aim to keep the number of outliers as small as possible. Of course, the general k-max problem can be modelled in the same way as a bi-criteria problem. This problem is denoted by (BP) in the following:

$$\text{(BP)} \begin{cases} \min & k\text{-max}(y) \\ \min & k \\ \text{s.t.} & y \in \mathcal{Y} \\ & k \in \{1, \dots, n\} \end{cases}.$$

In contrast to single-criteria problems, the objective function of (BP) resp. (BPL) is a vector and not a scalar. In the field of multi-criteria optimisation there exist different concepts of optimality which can be found in a more general formulation for example in Ehrgott (2005). In the following, let $f^1(y, k) = k\text{-max}(y)$ and $f^2(y, k) = k$.

Definition 2.18 ▶ Efficiency

A solution $(\bar{y}, \bar{k})$ of (BP) is called *efficient* if there is no other feasible solution (y, k) such that

$$f^i(y, k) \leq f^i(\bar{y}, \bar{k}) \quad \text{for} \quad i = 1, 2 \quad \text{and} \quad f^j(y, k) < f^j(\bar{y}, \bar{k})$$

for at least one $j \in \{1, 2\} \setminus \{i\}$. If $(\bar{y}, \bar{k})$ is efficient, $f(\bar{y}, \bar{k}) = (f^1(\bar{y}, \bar{k}), f^2(\bar{y}, \bar{k}))$ is called *non-dominated*.

Roughly spoken, this means that in an efficient solution it is not possible to improve in one criterion without deteriorating in the other. Figure 2.8 shows the images $f(X, k)$ of the optimal solutions of the k-max problem for all $k = 1, \dots, n$ for the set of existing facilities given in Figure 2.6. This set of points in the image space is called the *trade-off front* of

(BP). The trade-off front is always non-increasing w.r.t. k due to Lemma 2.17 and coincides with the curve describing the relation between k and the corresponding optimal objective function value. Definition 2.18 is illustrated in Figure 2.8: The image $f(\bar{X}, \bar{k})$ is considered and analysed for feasible outcome vectors to the left and below. In other words, $(\bar{X}, \bar{k})$ is efficient if the set

$$f(\bar{x}, \bar{k}) - \mathbb{R}^2_{\geq} = \{z \in \mathbb{R}^2 : z = f(\bar{x}, \bar{k}) - \varepsilon \text{ with } \varepsilon_1, \varepsilon_2 \geq 0 \text{ and } (\varepsilon_1, \varepsilon_2) \neq (0, 0)\}$$

(shaded in grey) does not contain any other image of a feasible solution.

Definition 2.19 ▶ Weak efficiency

A solution $(\bar{y}, \bar{k})$ of (BP) is called *weakly efficient* if there is no other feasible solution (y, k) such that

$$f^i(y, k) < f^i(\bar{y}, \bar{k}) \quad \text{for} \quad i = 1, 2.$$

If $(\bar{y}, \bar{k})$ is weakly efficient, $f(\bar{y}, \bar{k}) = (f^1(\bar{y}, \bar{k}), f^2(\bar{y}, \bar{k}))$ is called *weakly non-dominated*.

Different from the concept of efficiency, for weak efficiency it is allowed that two solutions have the same objective value in one criterion, i.e. for a weakly non-dominated point $f(\bar{\bar{X}}, \bar{\bar{k}})$, the set $f(\bar{\bar{x}}, \bar{\bar{k}}) - \mathbb{R}^2_{\geq}$ (the interior of which is illustrated in Figure 2.8) can contain images of other feasible solutions on its boundary.

Note that the trade-off front as defined above is a superset of the non-dominated points and a subset of the weakly non-dominated points (see Figure 2.8 and Figure 2.11)

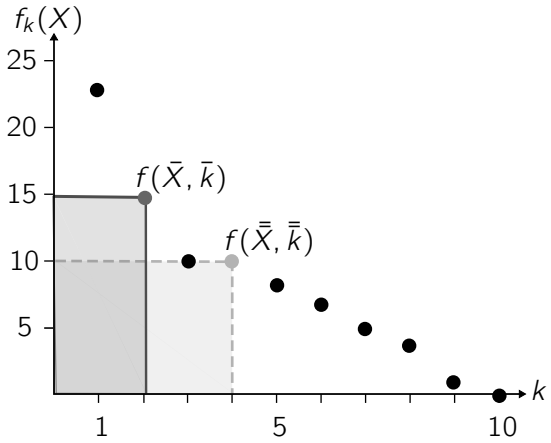


Fig. 2.8: Trade-off front of (BPL) with non-dominated point $f(\bar{X}, \bar{k})$ and weakly non-dominated point $f(\bar{\bar{X}}, \bar{\bar{k}})$

To solve the problem (BP) resp. (BPL), i.e., to find all efficient points, a well known scalarisation technique, the ε-constraint method, can be used. The idea is to generate a series of scalarised problems which minimise only one of the original objective functions while the other objective function is transformed into a constraint by setting an upper bound ε on the objective function value (see, for example, Ehrgott, 2005). The ε-constraint formulation

of (BP) resp. (BPL) is

$$(\mathrm{BP}_\varepsilon)\begin{cases}\min & k\text{-max}(y)\\ \text{s.t.} & k \le \varepsilon\\ & y \in \mathcal{Y}\\ & k \in \{1,\dots,n\}\end{cases} \quad \text{resp.} \quad (\mathrm{BPL}_\varepsilon)\begin{cases}\min & k\text{-max}(d^w(A,X))\\ \text{s.t.} & k \le \varepsilon\\ & X \subseteq \mathcal{X}\\ & k \in \{1,\dots,n\}\end{cases},$$

where the value of k is now bounded by $\varepsilon \in \mathbb{R}$. Figure 2.9 illustrates problem (BPL_4) for $\mathcal{X} = \mathbb{R}^2$. Setting an upper bound on k means to restrict the feasible set to the values of $k \in \{1,\dots,\rho\}$ with $\rho := \max\{\bar{\rho} \in \mathbb{N} : \bar{\rho} < \varepsilon\}$ (illustrated by the gray lines for $\rho = 4$). Minimising the k-max location function on this feasible set leads to two alternative optimal outcome vectors shown in grey since the k-max values for $k = 4$ and $k = 3$ are equal, as seen before. Note that the constraint of $(\mathrm{BP}_\varepsilon)$ is always active in an optimal solution with the largest feasible value of k because of the monotonicity property of Lemma 2.17.

The following theorem provides a sufficient condition for weak efficiency.

Theorem 2.20 ▶ Weak Efficiency, see Ehrgott, 2005
Let y^* be an optimal solution of $(\mathrm{BP}_\varepsilon)$. Then y^* is weakly efficient for (BP).

Due to Theorem 2.20, applying the ε-constraint method to (BP) yields weakly efficient solutions. Note that this approach provides a superset of the efficient set but the weakly efficient points can be filtered out easily as will be described later.

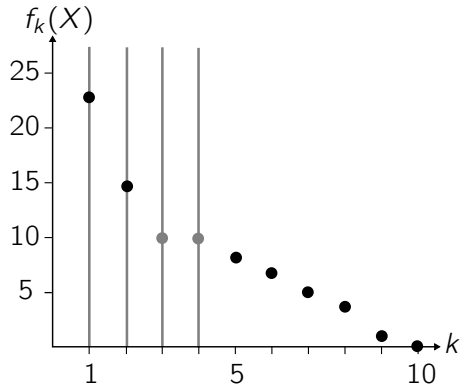


Fig. 2.9: Constraints (indicated by the gray lines) and optimal outcome vectors (gray dots) of the scalarisation (BPL_4)

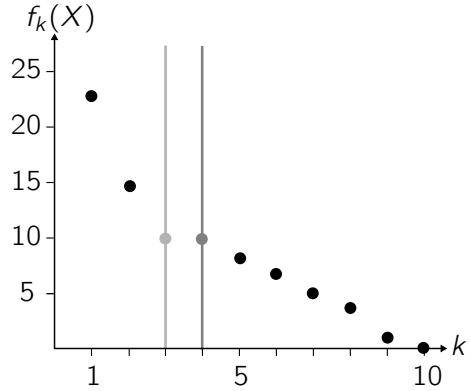


Fig. 2.10: Constraints (gray lines) and optimal outcome vectors (gray dots) for $(\mathrm{BPL}_{\varepsilon^=})$ for $\varepsilon = 3$ (light gray) and $\varepsilon = 4$ (dark gray)

Conversely, every weakly efficient solution of (BP) can be generated by a scalar problem $(\mathrm{BP}_\varepsilon)$ with a suitable upper bound ε (see, e.g., Ehrgott, 2005). For general multi-criteria problems, the choice of an appropriate ε is difficult. Due to the discrete, finite number for the values of k, this choice is obvious for $(\mathrm{BP}_\varepsilon)$: It is sufficient to solve the problem $(\mathrm{BP}_\varepsilon)$ for all $\varepsilon = 1,\dots,n$ in order to generate all weakly efficient points. As a result of the known

values for ε and the monotonicity of the k-max value, (BP_ε) and (BPL_ε) can be transformed to a modified ε-constraint problem

$$(\mathrm{BP}_{\varepsilon^=})\begin{cases}\min & k\text{-max}(y)\\ \text{s.t.} & k=\varepsilon\\ & y\in\mathcal{Y}\end{cases}\quad\text{resp.}\quad(\mathrm{BPL}_{\varepsilon^=})\begin{cases}\min & k\text{-max}(d^w(A,X))\\ \text{s.t.} & k=\varepsilon\\ & X\subseteq\mathcal{X}\end{cases},$$

for $\varepsilon\in\{1,\dots,n\}$ where the inequality-constraint is replaced by an equality-constraint such that the feasible set is now restricted to the single value $k=\varepsilon$. When setting the parameter ε and therefore k to a fixed value, ($\mathrm{BP}_{\varepsilon^=}$) is a k-max problem as stated in Definition 2.3 and ($\mathrm{BPL}_{\varepsilon^=}$) becomes a k-max location problem of type (pkMP).

Lemma 2.21 ▶ Weakly efficient solutions of ($BP_{\varepsilon^=}$)
Every optimal solution of ($\mathrm{BP}_{\varepsilon^=}$) is weakly efficient for (BP).

Proof. Let $y^*\in\mathcal{Y}$ be optimal for (BP_{k^*}) with $k^*\text{-max}(y^*)=z^*$.

Assume that y^* is not weakly efficient for (BP), i.e. there exists a $\bar{y}\in\mathcal{Y}$ such that $\bar{k}\text{-max}(\bar{y})<z^*$ with $\bar{k}<k^*$. Thus, $\bar{y}$ is also feasible for (BP_{k^*}) and it holds

$$k^*\text{-max}(\bar{y})\le\bar{k}\text{-max}(\bar{y})<z^*,$$

where the first inequality is valid due to Lemma 2.17. This leads to a contradiction to the assumption of y^* being optimal for (BP_{k^*}). ■

Because of the discrete, finite set of values for k, the solution of the modified ε-constraint problem for all $\varepsilon=1,\dots,n$ returns the set of all weakly efficient points of (BP). Hence, instead of a series of problems of type (BP_ε), a sequence of problems of type ($\mathrm{BP}_{\varepsilon^=}$) can be used to solve the initial bi-criteria problem. For the generation of all weakly efficient points, the overall number of subproblems to solve is not enlarged by the substitution of (BP_ε) by ($\mathrm{BP}_{\varepsilon^=}$). This situation is illustrated in Figures 2.9 and 2.10 where $k'=4$ and $\bar{k}=3$.

An important property of the underlying bi-criteria problem is that an upper bound on the number of non-dominated points is known in advance.

Theorem 2.22 ▶ Number of non-dominated points
The problem (BP) has at most n non-dominated points.

Proof. Let $\bar{k}\in\{1,\dots,n\}$. Obviously, only the lowermost image point $f(\bar{y},\bar{k})$ of an optimal solution $(\bar{y},\bar{k})$ can be non-dominated. All other points $f(\tilde{y},\bar{k})$, for which $(\tilde{y},\bar{k})\neq(\bar{y},\bar{k})$ is feasible but not optimal, can at most be weakly non-dominated as $f(\bar{y},\bar{k})$ lies on the boundary of the set $f(\bar{\bar{x}},\bar{\bar{k}})-\mathbb{R}^2_{\geq}$ for all points $f(\tilde{y},\bar{k})$ with $\tilde{y}\neq\bar{y}$. Since $f(\bar{y},\bar{k})$ can also be dominated by another point in the objective space (see Figure 2.11), there can in total be at most n non-dominated points for the problem (BP). ■

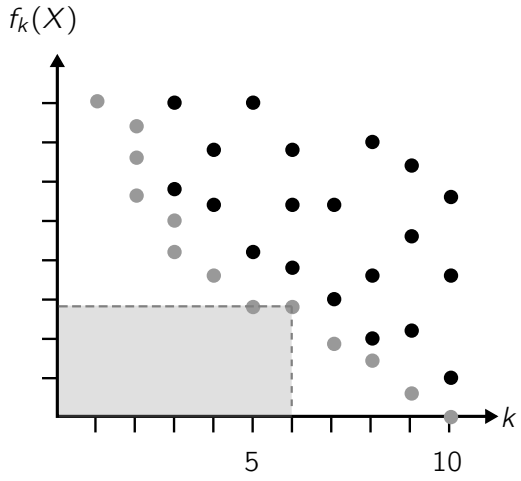


Fig. 2.11: Example of an image space of (BP) with outcome vectors of feasible solutions (black and grey points). The weakly non-dominated points are marked in grey.

Summarising the discussion above, the weakly efficient points of the bi-criteria problem (BP) can be generated by solving n single-objective k-max problems for $k = 1, \ldots, n$. Not all of these points are of interest for a decision maker, therefore, the points that are not efficient shall be filtered out. The monotone shape of the trade-off front can be used to implement this filtering process very efficiently. Instead of solving the single-objective k-max problems in an increasing order w.r.t. k, the computation is done in a reverse order starting with $k = n$ and going down to $k = 1$. In every step, the optimal objective function value $z^*_{\tilde{k}}$ of the current problem ($\text{BP}_{\tilde{k}^=}$) has to be compared with the objective function value $z^*_{\tilde{k}+1}$ of the previous problem ($\text{BP}_{(\tilde{k}+1)^=}$). If the $\tilde{k}$-max value $z^*_{\tilde{k}}$ and the $(\tilde{k}+1)$-max value $z^*_{\tilde{k}+1}$ are equal, the solution of ($\text{BP}_{(\tilde{k}+1)^=}$) is only weakly efficient and can be eliminated. The set of the non-dominated points is called the *Pareto front* of (BP).

Algorithm 1 summarises the procedure to solve the bi-criteria problem (BP).

Algorithm 1 General bi-criteria Problem (BP)

Input: Parameter $k \in \{1, \ldots, n\}$, feasible set $\mathcal{Y} \subseteq \mathbb{R}^n$.

1: Set $\mathcal{Y}_e := \emptyset$
2: **for** $k = n, \ldots, 1$ **do**
3: Determine the set of optimal solutions Y_k of the k-max problem
4: Set $z_k := k\text{-max}(y)$ with $y \in Y_k$
5: **if** $k \neq n \wedge z_k \neq z_{k+1}$ **then** ▷ Filtering out weakly efficient solutions
6: $\mathcal{Y}_e := \mathcal{Y}_e \cup Y_{k+1}$

Output: Set $\mathcal{Y}_e$ of efficient solutions of (BP).

Algorithms to solve the single-objective k-max subproblems are needed for the solution of the bi-criteria problem. They should be developed depending on the underlying type of optimisation problem to use its special structures because the overall complexity of the bi-criteria approach is highly influenced by the complexity of the ε-constraint problems.

Applied to location problems, the single-objective p-k-max problem has to be solved repeatedly to determine all weakly efficient solutions. These single-objective k-max location problems with p new facilities will be analysed extensively in the following chapters to find efficient solution methods for them. It will be shown in Chapter 5 and Chapter 6 that the single-objective p-k-max location problems can be solved by using different types of finite dominating sets. The single facility problems can be solved in polynomial time (see Section 5.2). As a consequence, the complete set of weakly non-dominated points of (BP) can be computed in polynomial time for $p = 1$.

Contrary to this, the problem of locating $p \geq 2$ facilities is NP-hard in general (see Theorem 4.14) and has therefore a much higher complexity. The complexity of Algorithm 1 is $\mathcal{O}(nT)$, where $\mathcal{O}(T)$ is the computational effort to compute the optimal solution of a single-criteria p-k-max problem. For a single facility problem this results in $\mathcal{O}(nm(n+s)\log(n))$ with $s \leq \binom{n}{2}$. Note that the solutions of the k-max problem can be computed in parallel for all possible values of the parameter k. This advantage can be used to speed up Algorithm 1.

2.4 Selecting an Appropriate Solution

The solution process of the bi-criteria model (BP) resp. (BPL) does not yield a single solution, but a set of efficient solutions. These solutions correspond to different numbers of outliers and the corresponding objective function values. Even though the existence of several alternative solutions can be very helpful in understanding the underlying problem and guarantees a greater flexibility, the decision maker is often interested in finding one or several solutions that are in a sense most sensible for a particular problem. There are different options to choose an appropriate value for the parameter k and therefore a most preferred solution for the respective situation.

One approach to select a particular solution of a Pareto front of a general multi-criteria optimisation problem is to pick a point where the front has its biggest convex bulge, also called the “knee" of the Pareto front (see, for example, Das, 1999). These points represent a good compromise between the conflicting objectives. Often, these solutions lie somewhat “in the middle” of the Pareto front and are “furthest away” from the extreme points of the curve.

There is no exact mathematical definition of the knee-solution even though this topic was often discussed in the last decades. Das (1999), among others, gives a possible characterisation based on a normal-boundary intersection procedure. Solving the normal-boundary intersection problem for certain weights provides the bulges lying in the middle of the Pareto front without determining all efficient points. Maximising the distance from these bulges to the convex hull of individual minima of the objectives leads to the wanted knee-solution. This approach is not very advantageous for the problem (BP) because the complete Pareto

front can be determined with the same effort as just one efficient point. Branke et al. (2004) also work on finding knee solutions without the need to compute the whole Pareto front. An evolutionary algorithm with different measures (one based on the angle to neighbours, another based on marginal utility) is used to focus on the knee regions.

Handl and Knowles (2007) work with a Pareto front similar to the one obtained for (BP) and give an approach to select a good solution based on the whole known front. They introduce a multi-objective clustering approach with automatic k-determination where k is the number of clusters. The algorithm consists of two phases, an initial clustering phase and a second model-selection phase to determine the number of clusters. The first phase uses an evolutionary approach to optimise two complementary clustering objectives, minimising the overall deviation of a cluster, but also minimising the connectedness (i.e., the degree to which items are placed in the same cluster as their nearest neighbours in the data space). The output is a set of approximated non-dominated clustering solutions that corresponds to different trade-offs between the two objective functions and also to different numbers of clusters. Of course, the deviation decreases and connectivity increases with an increasing number of clusters. This property results in the same structure of the Pareto front as it is observed for the (BP) problem: The solutions are ordered from left to right by the number of clusters they contain, i.e., k gradually increases as it does for the k-max problem. The Pareto front is monotone, consists of discrete points and the number of weakly efficient points is n, which is the maximum number of clusters. To find a knee-solution, an auxiliary clustering-run on random control data (i.e., data not containing clusters but covering the same data space) is performed to compare the control Pareto front with the original data. For each solution of the original problem, the Euclidean distance between this solution and the closest point on the reference curve is determined. Promising solutions are the ones with the largest distances to the control data because they are likely to form the knee.

This approach can also be applied to the problem (BP) because of the similar structure of the Pareto front. A disadvantage is that the control front is based on random data and can therefore have a negative impact on the selected solutions. Moreover, this algorithm has a high complexity. There are also some easier and more intuitive ways to pick a good solution. One is to use the trade-off among the two criteria as a basis for decision making.

Definition 2.23 ▶ Trade-off

Let $y_{\tilde{k}}, y_{\bar{k}} \in \mathcal{Y}$ be two efficient solutions of (BP) with $\tilde{k} < \bar{k},\ \tilde{k}, \bar{k} \in \{1, \ldots, n\}$. The *trade-off* $\Delta_{\tilde{k},\bar{k}}$ between $y_{\tilde{k}}$ and $y_{\bar{k}}$ is defined as

$$\Delta_{\tilde{k},\bar{k}} = \frac{\tilde{k}\text{-max}(y_{\tilde{k}}) - \bar{k}\text{-max}(y_{\bar{k}})}{\bar{k} - \tilde{k}}.$$

The trade-off is given by the decrease in the k-max function value in relation to the increase of the parameter k. Because of the discrete values for $k \in \mathbb{N}$, the trade-off $\Delta_{k,k+1}$ between two efficient points for consecutive values of k is then given by the difference in the objective function values on the ordinate of these two solutions. A strategy to choose one

of the efficient solutions is then to pick

$$x_{k^*} \quad \text{where} \quad k^* = \underset{k \in \{1,\ldots,n\}}{\arg\max} \{\Delta_{k-1,k}\},$$

i.e., the rightmost point of two consecutive efficient points generating the largest trade-off among all these pairs. In this solution of a location problem, the reduction of the covering radius in relation to the number of outliers is the best possible.

Chapter 3

Literature Review

In the following, a detailed review of the existing literature on problems related to this thesis is given. Important fields are location theory in general with a focus on center problems and the handling of outliers in location problems. A further section is dedicated to ordered median problems in its different fields. Moreover, the field of k-max optimisation and the closely related bottleneck problems are considered. Finally, related topics regarding outlier detection techniques are reviewed. The lists are not intended to be exhaustive since the focus is set on results that are closely related to this work. Whenever possible, the notation used in the articles is adapted to the notation of this thesis.

3.1 Basic Concepts in Location Analysis

In location analysis, one or more new facilities shall be located in a predefined space in relation to a set of already existing facilities (the customers) such that some kind of objective function gets optimal. The meaning of "optimal" depends on the underlying constraints and the considered optimality criterion. Facility location decisions are important subproblems in many strategic planning processes for a wide range of private and public companies.

The long history of location science can be traced back to the early 17th century when Pierre de Fermat formulated the geometric problem of finding a point in the Euclidean plane that minimises the sum of its distances to three given points. The first proposed solution based on a geometrical construction for this problem was given by Evangelista Torricelli around 1645. In 1909, **Weber (1909)** generalised this problem by assigning different weights to the given points to represent customers having a demand (the weights) and to locate a new production site that minimises the total transportation cost between the customers and the production site. Thereby he transformed the purely geometric problem into an industrial location problem for the first time and the problem is known as the continuous Weber problem since then. In the 1930s, **Weiszfeld (1937)** proposed a solution approach for the Weber problem. The iterative procedure allows the solution of the problem for an arbitrary number of known points and is one of the most used approaches until today. In the mid-1960s, **Hakimi (1964)** published his outstanding work on the absolute median problem on networks (i.e., the new facility is allowed to lie somewhere on the edges or in the nodes of the graph) where he showed that an optimal solution of the problem can always be found in the nodes of the graph. He also extended the problem to find p new facilities on the graph.

Another interesting question was proposed in the 1850s by **Sylvester (1857)**: *"It is required to find the least circle which shall contain a given system of points in a plane."* Just

a few years later, Sylvester himself answered the question. This problem is today known as the one-center problem in the plane and is one of the most studied problems in location science.

In the following years, location science became a new scientific area and a very active research field. Significant advances have been made in different fields of location science and a large number of papers were published motivated by a broad variety of practical applications. An overview of important results in all fields of location theory is given, for example, in **Love et al. (1988)**, **Francis et al. (1992)**, **Drezner (1995)**, **Drezner and Hamacher (2002)**, **Eiselt and Marianov (2011)**, **Daskin (2013)** and **Laporte et al. (2015)**.

Location problems are often categorised according to the solution space (continuous, network or discrete), the number of facilities to locate (single- or multi-facility), the distance measure (gauge, norms, graph-distances), the weights (weighted or unweighted existing facilities), the feasible region (barriers or forbidden regions), and the type of objective function (for example median, center or covering). In discrete problems, the locations of the new facilities have to be chosen from a predefined set of candidate locations, thus often binary variables are used to model the problem. In contrast to that, continuous location problems have continuous variables associated with them which define the coordinates of the facilities that are to be located. In network location problems, the existing facilities are the nodes of a graph and the new facilities are allowed to be located on the edges or in the nodes of this graph.

3.2 Center Location Problems

As this thesis will focus on generalised center problems, this field of location science is reviewed in more detail in the following section. Thereby, the three fields of continuous, discrete and network location problems are considered. Center location problems often arise in emergency service location, for example, to locate a hospital, a fire or a police station. Normally, the aim to save human life and therefore to minimise the maximum time of travelling is much more important here than the overall transportation cost. As defined in Section 2.2, the p-center solution is a set of p points that minimises the largest weighted distance between each customer and its closest new facility.

Center Location Problems on Networks

Hakimi (1964) formulated the today known 1-center problem on a graph for the first time. The proposed idea is to compute a locally optimal center point on each edge of the graph assuming that the optimal solution is restricted to be on this edge. By comparing the objective function values of all these local center points, the overall optimal center point of the graph can be found. For a detailed description of the approach see Section 5.3.

Minieka (1970) introduces the first algorithm using a set covering approach to solve an unweighted p-center problem on a graph based on the idea of Hakimi (1964). It is observed that the finite dominating set proposed for the 1-center problem is also a finite

dominating set for the multi-facility version of the problem, i.e., each of the new facilities has to lie in a local center point of an edge. The problem can be transformed into a sequence of set covering problems. Each set covering problem aims to minimise the number of facilities needed to cover all customers with service within a given coverage radius z. Let z_i with $i = 1, \ldots, L$, $L \leq nb$, where b is the number of possible new locations, be the non-decreasingly sorted distinct distances between all pairs of nodes. These values define the sequence of set covering problems with radii z_i. If the optimal number of new facilities resulting from the set covering problem is smaller than p, the optimal objective value of the p-center problem is at most z_i. Hence, the algorithm terminates when the optimal objective value of the current set covering problem is larger than p. This approach converges to the optimal solution in a finite number of iterations.

Goldman (1972) focuses on the structure of the underlying network to solve the single-facility center problem. For this purpose, an edge of the graph that is not contained in any simple cycle (i.e., its removal results in two disconnected components) is defined to be an *isthmus*. An isthmus is important as every path connecting the two components of the graph must contain the isthmus. An isthmus can be evaluated in the following way: Considering the longest shortest path between a pair of points not belonging to the same component of a graph, this path also has to pass through the isthmus. If the midpoint of the path lies on the isthmus, this edge has to be optimal. Otherwise, if the midpoint is in one of the respective components of the graph, the search for the optimal location can be restricted to that component. This approach can be extended to weighted problems. In tree graphs, every edge is an isthmus and thus the algorithm can be applied efficiently in this case.

Garfinkel et al. (1977) improve the set covering based approach by Minieka (1970). The solution space is reduced here by first finding a heuristic solution of the problem and deleting all points from the finite dominating set that generate a larger covering radius than that of the heuristic solution. In this way, the number of variables for the set covering problem can be reduced. Moreover, a bisection search strategy for the selection of the next radius value to determine an order of the set covering problems is proposed. Further improvements of the set covering approach are proposed for, example, in **Christofides and Viola (1971)**.

Kariv and Hakimi (1979) study the weighted and the unweighted single as well as the multi-facility problem. The solution approaches for the 1-center problems are based on the finite dominating set introduced by Hakimi (1964) but use more efficient ways to compute this set. In this way, the weighted case is solved in $O(mn\log(n))$ time and the unweighted case in $O(mn)$ time. This is the best known bound for the absolute 1-center problem until now. Moreover, it is proven that the absolute p-center problem is NP-hard even for the most simple structures of general unweighted planar networks. For the weighted multi-facility problem an enumeration algorithm is presented that enumerates in $\mathcal{O}(m^p n^{2p}\log(n))$ time all subsets of size p of the finite dominating set for the 1-center problem. It relies on the fact that in an optimal solution each new facility is the 1-center of an appropriate subgraph. A similar idea is used in this thesis to derive a basic solution approach for the p-k-max problem based on a slightly modified finite dominating set of the 1-k-max problem (see Algorithm 9).

Mladenović et al. (2003) propose a meta-heuristic algorithm to solve the p-center prob-

lem. The first approach uses a vertex substitution search, i.e., one facility of the current solution is replaced by another facility not belonging to this solution. The second approach is a tabu search heuristic that extends the interchange of one facility of the first approach with the interchange of more than one facility. The third algorithm is based on a variable neighbourhood search that perturbs the current solution by a k-interchange neighbourhood, and vertex substitution is used afterwards to improve the solution. Exhaustive tests show that the last approach performs best on average.

In **Bhattacharya and Shi (2014)** it is shown that the p-center problem on a graph can be transformed to the well-known Klee's measure problem (see, for example, Overmars and Yap, 1991), which is the problem to compute the measure of the union of a set of d-boxes, where a d-box is the Cartesian product of d intervals in d-dimensional space. The overall approach is based on the set covering strategy proposed by Minieka (1970) and various improvements of this algorithm. The idea is to transform the local feasibility test for the coverage radius needed for the current set covering problem to a p-dimensional Klee's measure problem. The transformation is based on the fact that there is at most one continuous region on each edge that contains all points with weighted distance larger than the current coverage radius to that node. For d fixed edges, these regions are used to construct a d-box, i.e., a rectangular area in the d-dimensional space. As solutions in this d-box do not cover all demand points within the coverage radius, the d-box is called a forbidden region. Thus, the feasibility test is transformed into the question if the union of all these forbidden regions for all nodes form the whole d-box formed by all points on the d edges. If yes, the coverage radius is infeasible, if no, it is feasible. With this procedure, the complexity of the feasibility test can be reduced significantly.

Discrete Center Location Problems

At first it should be noted that all variants of the p-center problem on networks that are based on a finite dominating set can also be transformed to discrete p-center problems since in the discrete case a candidate set is known in advance.

The discrete p-center problem is shown to be NP-hard by **Kariv and Hakimi (1979)**.

Hooker (1986) introduces a general approach for solving the non-linear single-facility center problem, i.e., the cost function is a convex function of the distances. The analysis of the problem is based on a decomposition of the network into "tree-like" segments. The objective function is defined by the maximum of convex functions of distances and is therefore convex on any "tree-like" segment. The optimal solutions of the center problem can then be found by solving a convex programming problem on each of these segments. Moreover, it is shown that not all segments need to be examined. The algorithm is particularly effective if the cost function is non-decreasing and semi-separable.

Daskin (2013) gives a mixed integer programming formulation for the discrete p-center problem. Two binary decision variables are defined: $y_i \in \{0, 1\}$ with $y_i = 1$ if a facility is placed at candidate site v_i, $i \in I$ and zero otherwise, and $x_{ij} \in \{0, 1\}$ with $x_{ij} = 1$ if customer $j \in J$ is assigned to the new facility placed at $i \in I$. Moreover, a set covering based algorithm is proposed where lower and upper bounds are defined to choose the coverage radius of the

set covering problem such that the interval to choose the radius from is bisected in every iteration.

Elloumi et al. (2004) present an integer programming formulation of the discrete p-center problem with a polynomial number of variables and constraints that is based on its relationship to the set covering problem. The idea is based on the fact that the optimal objective function value is restricted to a finite set of values $r_1, \ldots, r_L$ given by the L candidate locations. Binary variables are introduced as $z^\ell \in \{0, 1\}$, $\ell = 2, \ldots, L$, with $z^\ell = 0$ if all customers are covered by p facilities within the radius $r_{\ell-1}$ and $z^\ell = 1$ otherwise. The best known lower bound for the p-center problem is obtained in this work by applying a semi-relaxation of this formulation, i.e., the integrality constraints on a subset of the variables are kept. Moreover, a two-phase algorithm to solve the p-center problem exactly is given which is based on the work of Minieka (1970).

Mihelič and Robič (2005) use a polynomial time heuristic algorithm based on the solution of a finite series of minimum dominating set problems. The minimum dominating set problem wants to find a subset D of the nodes with minimum cardinality such that any other node is adjacent to at least one of the nodes in D. The heuristic uses the so called scoring-technique to rank the nodes as potential centers. A requirement for the algorithm is that the considered network is complete and that the distances satisfy the triangle inequality.

Pullan (2008) combines a population based meta-heuristic with a local search algorithm to solve the vertex-restricted p-center problem. The starting points for the procedure are generated by phenotype crossover and directed mutation tools of the genetic algorithm. Afterwards, the starting points are improved by local search. The algorithm is also applicable to large instances of the problem as it can easily be executed in parallel and yields moreover excellent robustness results.

Calik and Tansel (2013) propose a further integer programming formulation of the p-center problem with additional binary variables u_ℓ that are equal to 1 if the corresponding radius r_ℓ of candidate location $\ell \in \{1, \ldots, L\}$ is selected as the optimal objective function value. This formulation is tightened by using the relation to the formulation of Elloumi et al. (2004). Moreover, a polynomial time algorithm is introduced to compute a lower bound by solving a finite series of linear programming problems of polynomial size. Thus, this approach differs from the often used set covering approach. Different selection strategies to choose a restricted set of radius values, including a method called double-bound method, are analysed.

Continuous Center Location Problems

Using Euclidean distances in the plane, the single-facility center problem is equivalent to finding the center of the smallest enclosing circle of all existing facilities. If another metric is used to measure the distances, the task is to find a minimum shape isomorphic to the scaled unit ball of the metric that covers all existing points in the plane. The p-center problem is proven to be NP-hard by **Megiddo and Supowit (1984)**.

Chrystal (1885) solved the unweighted single-facility center problem with Euclidean distances. The aim is to find the smallest enclosing circle of the existing points in the plane since the midpoint of this circle is the optimal location for the center point. It is observed

that all points of the demand set not lying on the convex hull can be eliminated from further consideration as they will not influence the optimal solution. Further observations on the relation of the circle and a special triangle are made: If three demand points form the vertices of an acute triangle, the smallest enclosing circle of these three points passes through all these three points. If the triangle is obtuse, the center of the smallest enclosing circle of the three points is located at the midpoint of the side opposite to the obtuse angle. Thus, the center of the smallest enclosing circle is either the midpoint of the most distant pair of points or of the circle passing through three points forming an acute triangle. The algorithm starts with a circle of a very large radius that contains all demand points and then the radius is reduced iteratively using the above observations.

Elzinga and Hearn (1972) consider the unweighted 1-center problem with both Euclidean and rectilinear distances. Contrary to Chrystal (1885), the procedure for Euclidean distances starts with a small circle defined by two points. In each iteration, a point not covered by the current covering circle is used to find a larger circle that covers a new subset of the points, including the point found which was not covered before. Based on the results of Chrystal (1885), the current covering circle is either given by two points defining the diameter of the circle or by three points defining an acute triangle such that the circle is the perimeter of this triangle. The optimal solution is found when the current circle covers all existing points. The convergence of the algorithm is guaranteed as the radius of the covering circle increases in each iteration. For rectilinear distances, the p-center problem is to find the smallest enclosing diamond of the n existing facilities $(a_i, b_i) \in \mathbb{R}^2$ with $i = 1, \ldots, n$. This diamond can be found in $\mathcal{O}(n)$ time by computing the four values $\min_i\{a_i + b_i\}$, $\max_i\{a_i + b_i\}$, $\min_i\{a_i - b_i\}$ and $\max_i\{a_i - b_i\}$ from which four lines inclined by 45° to the system of coordinates can be derived. The midpoint of this diamond is the optimal solution of the center problem.

Shamos and Hoey (1975) give an $\mathcal{O}(n \log(n))$ time algorithm for the 1-center problem by also constructing the smallest enclosing circle of the existing facilities. The basis of the solution approach are Voronoi diagrams (see, for example, Okabe et al., 2009). For the center problem, the farthest-point Voronoi diagram has to be analysed. In the farthest point Voronoi diagram, the point defining a Voronoi region is the farthest neighbour of all points in this region. The points on a Voronoi edge have the same distance to two customers and the Voronoi vertices are points on the Voronoi diagram that are equidistant to at least three customers. If the farthest point Voronoi diagram is given, the two farthest points of the given point set can be found easily by considering each Voronoi edge and computing the distances of the two points defining this edge. Thus, if the smallest enclosing circle of the customers is defined by two points, the optimal solution is found. If otherwise the smallest enclosing circle is defined by three points, it is shown with the observations made by Chrystal (1885) that the vertices of the Voronoi diagram are candidates for the center location. This is due to the fact that the midpoint of a circle passing through three customers forming an acute triangle is a vertex of the farthest-point Voronoi diagram.

Drezner and Wesolowsky (1980) present an iterative solution procedure similar to Elzinga and Hearn (1972) to solve weighted 1-center problems with Euclidean, rectilinear and general ℓ_p-norms. It is shown that it is enough to evaluate all triples of existing points also for ℓ_p-norms with general p. Thus, the algorithm evaluates all these triples in an order

such that the objective function value is decreasing until all points are covered. As every triple is evaluated exactly once, this procedure finds an optimal solution in finitely many iterations.

Drezner (1984) introduces a heuristic and an exact method to solve the weighted p-center problem with Euclidean distances. The heuristic is based on a location-allocation idea: p starting points are chosen as the initial set of locations for the center points and all demand points are allocated to these starting points with respect to the respectively defined Voronoi regions. The starting points are improved by solving 1-center problems within the Voronoi regions. In a second step, one demand point at a time is rearranged to one of the other found centers of the first phase to improve the objective function value. The exact algorithm works iteratively and every iteration starts with a feasible solution of the p-center problem with objective function value z provided by some heuristic. Then, a set covering problem is solved to find a better solution or to prove that the current solution is already optimal. For fixed p, the exact algorithm is polynomial in the number of demand points.

Chen and Chen (2009) present some new variants of relaxation methods to solve the p-center problem. In this context, relaxation is a method to solve a p-center problem by a sequence of smaller p-center-like problems on a subset of the demand points since the optimal solution of the p-center problem on a subset of the customers provides a lower bound on the optimal solution of the full p-center problem. The resulting iterative approach updates at each step the bounds on the optimal solution until the optimum is reached. The new described variants of this approach reduce either the number of iterations, the sizes of the sub-problems or the values of the coverage-distances (or a combination of all). Therefore, larger problems can be solved. One new relaxation technique is the reverse relaxation, which starts with a lower bound of 0 on the optimal objective value and updates this bound in every iteration until the optimal value is reached. This is in contrast to other approaches that start with an infinite bound which is reduced in every step. A second variant is the binary relaxation which performs a binary search on the optimal objective function value, i.e., it updates either an upper or a lower bound on the optimal solution until the optimal value is reached.

3.3 Outlier Location Problems

Location problems usually assume that all the clients have to be provided with service. The disadvantage of this problem formulation is that a few very distant clients, called outliers, have a disproportionately strong influence on the final solution because an outlier may attract a center to be placed in its vicinity. Thus, for outlier location problems, the new aim is to serve only a specified fraction of customers. Many of the approaches described in this section are closely related or even equivalent to the p-k-max location problems considered in this thesis.

Charikar et al. (2001) considers discrete facility location, p-center- and p-median problems with outliers. Two variations of these models are formulated such that the presence of outliers can be handled: The robust facility location model and the facility location model

with penalties. In the robust model, a parameter $b \in \mathbb{N}$ with $b < n$ is inserted such that the objective is to minimise the costs to serve any subset of at least b clients. With $b = n$ no outliers are excluded. In the facility location model with penalties, every not-assigned facility is associated with a penalty cost. By setting the penalties to ∞, the standard problem without outliers is obtained. Different known approximation algorithms are generalised to these new models, among others a 3-approximation algorithm for the robust p-center problem and various primal-dual approximation algorithms based on LP-relaxations of the respective facility location and p-median models. Note that the robust model applied to a center problem is equivalent to a k-max problem for $b = n-k+1$. In contrast to the problem on networks in this thesis, the problems considered in this paper are discrete. Moreover, the presented solution techniques yield an approximated solution, while the algorithms presented in this thesis are exact solution methods.

Agarwal and Phillips (2008) consider unweighted 2-center problems with Euclidean distances. The aim for the outlier 2-center problem is to find a pair of congruent disks of minimal radius that cover $n - k + 1$ points, where n is the number of existing facilities and $k - 1$ is the maximum number of outliers. Thus, the stated problem corresponds exactly to a 2-k-max problem with Euclidean distances in the plane. Based on a partition of the point set using unit disks and separator lines, the geometric properties of the problem are used to develop a randomised algorithm. For this purpose, two cases have to be considered: The case that the centers of the optimal disks are further apart than the optimal radius, and the converse case where the centers of the optimal disks are closer to each other than their radius. The Euclidean 2-center problems can be solved in $\mathcal{O}(nk^7 \log^3(n))$ time. Moreover, it is stated that the 4- resp. 5-center problems with l_∞-distances can be solved in $k^{\mathcal{O}(1)} n \log(n)$ resp. $k^{\mathcal{O}(1)} n \log^5(n)$ time. The aim for the l_∞-norm is analogously to cover all but $k - 1$ points by $p = 4$ resp. $p = 5$ congruent axis-aligned squares of minimal side length. The solution approaches stated in this article can equivalently be used to solve the 2-k-max problem with Euclidean distances and the 4-k-max resp. the 5-k-max problem with l_∞-distances in the plane.

Atanassov et al. (2009) do not deal with location problems in particular, but their approaches can be applied to location problems as well. The analysed problem is to remove $k - 1$ points from a set V of n points in $\mathbb{R}^2$ such that the remaining point set leads to a minimal objective function value $f(V \setminus V')$ with objective function $f : 2^{\mathbb{R}^2} \mapsto \mathbb{R}$. This class of problems is called the f-based $k - 1$ outlier removal problem. Depending on the considered objective function, different types of problems are obtained. The diameter-based and the (bounding box) area/perimeter-based problems are solved by a technique that finds a problem kernel of a certain size whose solution gives a solution to the original problem. A third type is the (convex hull) area/perimeter problem and is solved with an algorithm similar to the bounded search tree method. Invariants to different types of affine transformations of the input are analysed, for example, rotation and scaling. For fixed values of k, all algorithms run in $O(n \log(n))$ time which is shown to be optimal. By interpreting the points of the set V as customers, especially the approach to minimise the diameter of the set $V \setminus V'$ can be applied to solve a 1-k-max location problem in the plane. The diameter of a set of points equals twice the radius of the smallest covering circle. Thus, the problem is equivalent to a

1-k-max problem with unweighted customer locations.

In the work of **Zarrabi-Zadeh and Mukhopadhyay (2009)** high dimensional unweighted 1-center problems with outliers in $\mathbb{R}^d$, $d \geq 2$, are considered, i.e., all but a fixed number of $k-1$ points have to be enclosed by a ball of minimum diameter. This is a generalisation of the problem considered in Agarwal and Phillips (2008) and thus also a generalisation of the continuous 1-k-max problem to higher dimensions. The decision to cover a point is made online, i.e., this information gets available during the optimisation process and no reliable information about the points arriving in the future is assumed to be available. This practical setting is in particular suitable for real world applications that involve very large data sets. It is difficult to decide which point is an outlier and which is not only based on the current information. Therefore, a buffer is introduced in which new points are added to postpone the decision. The size and a strategy of extracting points from the buffer are explained. A streaming-algorithm for the unweighted problem gives a 2-approximation. Thus, the optimal solution of the algorithm provides a radius that is at most twice as large as the radius in the exact optimal solution.

Xu and Xu (2009) analyse special median problems on bipartite graphs, called uncapacitated facility location problems with penalties, where each customer is either assigned to an opened facility or rejected by paying a penalty. This problem relates to the corresponding location model with penalties in Charikar et al. (2001). A two-phase 1.8526-approximation algorithm based on an extended primal-dual algorithm is developed in which outliers can be recognised more efficiently compared with Charikar et al. (2001). The primal-dual algorithm is run on a modified instance of the problem with scaled opening costs to get a solution with an improved quality. Afterwards, the obtained solutions are improved by a greedy local search heuristic. The main idea is to trade connection cost with opening cost by iteratively opening more facilities. If there are several facilities in one iteration which would improve the quality of the solution, the order for adding the new facilities is computed with a greedy strategy.

Ahn et al. (2011) concentrate on unweighted p-center problems with l_∞-distances and up to $k-1$ outliers that do not have to be served, i.e., p pairwise disjoint squares resp. rectangles are sought that together contain at least $n-k+1$ points such that the area of the largest box is minimised. Motivated by the classical cluster analysis, the so called box-covering method is used to identify groups of points that lie close together. Outliers that enlarge the radius of a cluster are filtered out and excluded. Both cases are shown to be NP-hard for general p even if k is fixed. However, an exact algorithm of divide & conquer type is developed to solve the problems based on the property for two and three new facilities that there always exists an axis-parallel line that separates the box defining the first cluster from the box defining the second cluster (resp. from the two boxes defining the second and third clusters for three new facilities), provided that the boxes are disjoint. For more than three new facilities, this approach is generalised by recursively finding the separating line and solving the induced subproblems. When using squares for the covering, this leads to an $\mathcal{O}(n + k\log(k))$ algorithm. A similar approach with rectangles instead of squares has a complexity of $\mathcal{O}(n + k^3)$. Both variants of the problem are extension of the rectilinear p-center problem in the plane that can exclude a predefined number of outliers. Thus, these

problems are equivalent to p-k-max problems with rectilinear distances in the plane.

In **Ott et al. (2014)** the problem of identifying outliers is considered for the field of statistics and data analysis. The joint clustering and outlier detection problem is formulated as an extension of a general facility location problem with binary variables to decide whether a point is assigned to a created cluster or declared to be an outlier. With this formulation outliers are implicitly defined as those points whose presence in the dataset has the strongest negative effect on the overall solution. By identifying each cluster with a new facility, the approach can also be applied to outlier location problems. The problem is described by an integer programming model. To solve this problem, a Lagrangian relaxation is applied by relaxing the constraint that assigns each point to a cluster or declares it as an outlier. The single relaxations are solved by a heuristic that generates valid assignments, starting by choosing the points with the largest value of the Lagrange multiplier λ as outliers. The solution of the current relaxed problem is then used to compute a new subgradient to update the Lagrange multipliers.

Wang et al. (2014) combine a robust facility location problem and the location problem with penalties from Charikar et al. (2001). The resulting model is called a robust facility location problem with penalties and is formulated as a binary linear program. Its LP-relaxation and its dual are considered, and based on these a primal-dual 3-approximation algorithm is proposed. It is a two-step method where in the first step a dual feasible solution is obtained and in the second step a primal feasible integer solution is constructed aiming to reduce the opening costs. With a greedy augmentation procedure and the insertion of dummy-facilities, this result is improved to an approximation ratio of 2.

Chakrabarty et al. (2016) introduce the non-uniform p-center problem. For a finite metric space (X, d) and a collection of balls with radii $r_1 \geq \ldots \geq r_p$, the aim of this problem is to find a location of the centers of the balls in the metric space and the minimum dilation α, such that the union of balls of radius αr_i around the ith center covers all the points in X. In practice, this problem is used to model a vehicle routing problem with fleets of different speeds, which are represented by the different radii of the balls. The non-uniform p-center problem is a generalisation of the p-center problem with outliers as presented in Charikar et al. (2001) if there are p balls with radius 1 and $k-1$ balls with radius 0, where $k-1$ is the number of outliers. A 2-approximation algorithm to solve the k-center problem with outliers is introduced. The main technique is based on the connection between the LP-relaxation of the non-uniform p-center problem and the so-called fire-fighter problem on trees (see, e.g., Finbow and MacGillivray, 2009).

Hatami and Zarrabi-Zadeh (2017) consider 2-center problems with outliers in high-dimensional data streams. For a given set of n d-dimensional points and a bound $k-1$ on the number of outliers, the aim of this problem is to find two congruent balls of minimum radius to cover at least $n-k+1$ points of the set. This problem is equivalent to the continuous unweighted 2-k-max problem in $\mathbb{R}^d$. The solution approach is based on different methods as the far/close ball separation and the center point theorem (see Danzer et al., 1963). Two cases are distinguished for the solution of the problem: If the midpoints of the two optimal balls have a distance smaller than αr^*, where $\alpha > 0$ is predefined and r^* is the optimal radius, the optimal solution of the 2-center problem can be approximated using the

optimal solution of the 1-center solution. If conversely the midpoints of the two balls have a larger distance than αr^*, the points can be separated into two areas and a so-called buffer zone to find an approximated solution. The result is a streaming algorithm that yields an approximation factor of $1.8 + \varepsilon$ for any $\epsilon > 0$. This approximation factor matches that of the best streaming algorithm for the 2-center problem with no outliers.

An approach closely related to statistics is given in **Schöbel (1999)** who analyses location problems where (instead of points) lines or hyperplanes have to be located. As it has a close and interesting connection to k-max problems, this work is analysed in more detail in the following. Given a set $A \subseteq \mathbb{R}^2$, the line location problem is to find a line L minimising the distances to the n points in A. This is equivalent to a linear regression analysis in the field of statistics where it is used to predict future values of the data set. The located lines are called "estimators" in statistics. Many estimators are sensitive to data outliers, i.e., points that do not follow the pattern of the majority of the point. A robust estimator is, for example, the "least median of squares" (LMS) estimator (see **Rousseeuw, 1984**), where the median of the distances between the line and the data points has to be minimised. Interpreting the linear regression as a location problem, different estimators can also be obtained as optimal solutions of a line location problem with a corresponding choice of an objective function.

The LMS estimator, for example, is the optimal solution of a line location problem with an ordered median objective function where $\lambda = (0, \ldots, 0, 1, 0, \ldots, 0) \in \mathbb{R}^n$ is the vector of weights and the distances are given by the squared vertical distances between the line and the data points. Hence, for the LMS estimator with $\lambda(k) = 1$ and $\lambda(l) = 0$ for all $l \in \{1, \ldots, n\}, l \neq k$, the kth largest distance to the optimal line has to be minimised. Therefore, this problem is equivalent to a k-max problem where lines instead of points have to be located. As a consequence, many properties of the k-max problems analysed in this work can be transferred to line location problems and vice versa. For example, the reduction to center problems (see Theorem 4.8 below) leads to the property that, if the optimal set of outliers would be known, the optimal line could also be found by an ordered line location problem with $\lambda = (1, 0 \ldots, 0) \in \mathbb{R}^n$ on A without the optimal set of outliers.

For many line location problems, discretisation results are known (see, for example, **Laporte et al. (2015)**, Chapter 7). To the best of our knowledge, until now there are no finite dominating sets for the special case of ordered line location problems with vector of weights $\lambda = (0, \ldots, 0, 1, 0, \ldots, 0)$. It is an open problem for further research in k-max optimisation for line location. However, the discretisation results for the general line location problem with $\lambda \in \mathbb{R}^n$ and distance d in Laporte et al. (2015) can also be applied to the k-max function since this is a special case of the ordered median function.

3.4 Ordered Median Problems

Ordered median problems are very general location problems that unify location problems with respect to the chosen objective functions. Most solution approaches in location theory are designed only for convex objective functions. For continuous ordered median problems, solution methods are often based on discretisation results and finite dominating sets. Of

course, most of these procedures are not intended to compete against approaches that are especially designed to solve specific instances of the ordered median problem as the latter can use the special structure of the particular problem. The aim of ordered median analysis is rather to obtain a good performance for more general types of problems. Note that the approaches for non-convex ordered median problems with general vector of weights λ can equivalently be applied to solve k-max problems as the k-max function is a special case of a (non-convex) ordered median function.

The idea of associating a weight coefficient λ_i with the ith-largest cost element of a feasible solution is introduced by **Yager (1988)** for the first time in a very general way. The aim is to introduce a new operator in multi-criteria decision making called the ordered weighted averaging operator in order to aggregate different criteria functions to form an overall objective function.

Ordered Median Problems on Networks

In location theory, **Nickel and Puerto (1999)** introduce the ordered median problem on networks by adapting the ordered weighted averaging operator to network location problems. They show that this function is a generalisation of the most popular objective functions in location theory. They give a finite dominating set for the weighted single facility case that consists of the equilibrium points and nodes of the graph. Moreover, they give an example showing that this result does not hold for the multi facility case. In this case it is shown that the set of nodes is a finite dominating set for the problem if the objective function is convex.

Kalcsics et al. (2002) identify finite dominating sets to solve the single-facility weighted ordered median problem with n existing facilities both on a network and in the plane. It is shown that, in a strongly connected, directed network with non-negative node weights, one optimal solution can always be found in the set of the nodes of the graph. The problem can then be solved in $\mathcal{O}(mn\log(n))$ time. For undirected graphs with general node weights, the finite dominating set consists of nodes, bottleneck points and so called equilibrium points. The evaluation of this set takes $\mathcal{O}(mn^2\log(n))$ time. It is shown that the discrete single-facility ordered median problem has the same finite dominating set. The continuous rectilinear ordered median problem in $\mathbb{R}^d$, d fixed, can be solved in $\mathcal{O}(n^{2d+1}\log(n))$ time by constructing a partition of $\mathbb{R}^d$ consisting of the pairwise bisectors of the points and the hyperplanes parallel to the axes. At least one vertex of the partition is an ordered median. If the rectilinear problem is convex (i.e., $0 \leq \lambda_1 \leq \ldots \leq \lambda_n$), it can be solved with the algorithm of Cohen and Megiddo (1993) in $\mathcal{O}(n\log^{2d}(n))$ time.

In **Kalcsics and Nickel (2003)** the convex multi-facility ordered median problem on undirected networks with non-negative weights of the form $\lambda = (a, \ldots, a, b, \ldots, b)$ with $a < b$ is considered. It is shown that the set of so called pseudo-equilibrium points is a finite dominating set for this special structure of λ. The size of this finite dominating set is $\mathcal{O}(nm(f+m^2))$ where f is the number of equilibrium points. This finite dominating set is used to develop the first polynomial time algorithm for p-facility ordered median problems on tree networks. The result can be combined with an approximation algorithm for the p-median problem to

obtain an approximated optimal solution for ordered p-median problems on general graphs in polynomial time.

A finite set of candidates to be optimal for the 2-facility ordered median problem on networks with no additional assumptions for λ is proposed in **Rodríguez-Chía et al. (2005)**. The derived finite dominating set of size $\mathcal{O}(m^3n^6)$ has a structure that is different from previous finite dominating sets because it consists of pairs of points. Thus, there is no need to choose elements of this set by pairs to get the candidate solutions. The presented result can not be generalised to problems with more than two new facilities.

Tang et al. (2009) develop (based on the finite dominating set result of Kalcsics and Nickel (2003)) a finite dominating set for the convex unweighted multi-facility ordered median problem on networks where λ can have at least two different values. It is pointed out that the set of all pseudo-equilibrium points is a polynomial size finite dominating set for the ordered p-median problem with this special choice of λ. It is also shown that there always exists an optimal solution in which one of the new facilities is a node or an equilibrium point. Besides, a three-phase polynomial time algorithm similar to Kalcsics and Nickel (2003) for the problem with at most three different values in λ on tree networks is given.

Tang et al. (2011) consider the multi-facility ordered median problem with non-negative λ-weights on a strongly connected and directed graph. It is proven that the problem has a finite dominating set in the node set of the underlying graph, i.e. the result of Kalcsics et al. (2002) is extended from the single to the multi-facility case. An exact $\mathcal{O}(pn^{p+1}\log(n))$-time algorithm is developed by evaluating the objective function in all p-tuples of nodes. Of course, this comes along with a high complexity and thus, the ordered median problem can be solved efficiently by the finite dominating set only if the number of facilities is finite and small. Therefore, also a $6\frac{2}{3}$-approximation algorithm for a special instance of the unweighted ordered p-median problem with $\lambda = (1, \ldots, 1)$, the weighted p-median problem, is presented.

Moreover, **Puerto and Rodriguez-Chia (2005)** prove that there is no finite dominating set of polynomial size for the general p-facility ordered median problem on networks with general weights even on path graphs.

Discrete Ordered Median Problems

The discrete ordered p-median problem, that is shown to be NP-hard in Nickel and Puerto (2005), is considered especially in **Nickel (2001)**. Due to the fact that every permutation of an arbitrary set or vector can be represented by an assignment problem, a non-linear integer programming formulation is developed based on a combination of a special matching subproblem with some additional constraints to ensure the correct sorting of the solution elements, with a classical discrete p-median location model in which the vector of weights is $\lambda = (1, \ldots, 1)$. The result is a model that has neither a linear objective function nor only linear constraints. A linearisation for which the number of variables and constraints is proportional to the number of existing sites, equivalent to the linearisation of the quadratic assignment problem, is also proposed.

Domínguez-Marín (2003) introduces a mixed-integer programming formulation of the discrete ordered p-median problem that combines an assignment and a location-allocation

problem. As this model contains a quadratic objective function and some non-linear constraints, several linearisations of the model are presented (see also Section 4.3 below). A branch-and-bound algorithm is given that allows to solve larger instances of the discrete ordered median problem than the methods before.

Heuristic approaches (also based on Domínguez-Marín, 2003) to solve the discrete ordered p-median problem are given by **Domínguez-Marín et al. (2005)**. One approach is based on a genetic algorithm with evolution strategies to avoid that the new population is worse than the old one. The second is a variable neighbourhood search metaheuristic. The neighbourhoods to be investigated are ranked increasingly by their distances to the current solution. To avoid local minima, a shaking process is included where the movement to a neighbourhood further from the current solution goes along with a harder shake. A major difficulty in the application of the encoding is the computation of the adjustment in the objective function value when two facilities are interchanged. Therefore, the complexity is relatively high. However, with both procedures, heuristic solutions for up to 900 existing locations can be obtained.

Boland et al. (2006) propose two alternative linear integer program formulations for the ordered p-median problem with $\mathcal{O}(n^2)$ constraints in both cases and $\mathcal{O}(n^3)$ respectively $\mathcal{O}(n^2)$ variables. Different properties of optimal solutions are shown and used to strengthen the formulation using either additional constraints or a preprocessing that fixes the values of some variables or by relaxing integrality requirements on some variables. Moreover, a special tailored branch and bound procedure based on decisions of whether or not a site is selected for facility location is given. The approach uses combinatorial lower bounds and takes advantage of the special structure of the model. The lower bounds are archived by the minima of the rows of the cost matrix calculated over the columns that correspond to opened or undecided facilities.

Stanimirović et al. (2007) present two heuristic approaches to solve the discrete ordered p-median problem based on a hybridization of genetic algorithms and a generalization of the Fast Interchange heuristic. New crossover and mutation operators that keep the feasibility of individuals are introduced. Efficient encodings of the solution for a better evaluation of the objective function are given.

Marín et al. (2009) introduce the first formulation for the discrete ordered p-median problem with non-negative weights that allows a solution of problem instances with up to 100 possible existing locations in reasonable time. The model is based on a covering formulation, where two different sets of binary variables are used for the measuring of distances and the sorting of these distances. In a preprocessing phase, the initial model is strengthened by a number of variable fixing strategies such that the number of binary variables is reduced to n. Moreover, a group of valid inequalities is added which reduces the number of binary variables extremely. This gives rise to a specialised branch & cut algorithm for solving the problem.

A more compact reformulation of the model in Marín et al. (2009) is provided in **Marín et al. (2010)** for λ vectors containing many zeros. For the reformulation the second set of binary variables, responsible for the sorting of the distances, is reduced. The solution times can be further reduced by an adjustment of the solution procedure to the new model. Afterwards, the reduced version of the model is extended by some constraints so that negative weights can be handled as well. The idea for the solution approach is the same as in Marín

et al. (2009). The number of binary variables is reduced to n and the variable fixing strategies and the branch & cut algorithm can be applied with some adjustments.

Puerto et al. (2014) introduce another heuristic for the discrete ordered p-median problem based on the approach of Domínguez-Marín et al. (2005). New neighbourhood structures that favour faster local improvements of the objective function in the local search phase are used. Improvements are obtained because there is an upper bound set on the allocation costs that are allowed in the considered neighbourhood. A special data structure is computed in a preprocessing phase and handles just the really needed information to update and evaluate solutions such that an efficient encoding of the solution approach is possible. In particular, no sorting is needed for the evaluation of the objective function in each solution. Therefore, each evaluation can be performed in linear time.

Labbé et al. (2017) introduce several new formulations of the discrete ordered median problem based on its relation to a scheduling problem. As previous formulations of the discrete ordered median problem yield rather large integrality gaps, the aim of the new formulations is to reduce this gap by analysing the formal relationships between the lower bounds of the linear relaxations of the new formulations. It is shown that the bounds obtained by the new formulations are rather tight compared to previous known bounds. Moreover, some of the formulations consist of a significant smaller number of constraints than previous formulations. The models are solved with different variants of brand and bound as well as branch and cut algorithms.

Continuous Ordered Median Problems

Puerto and Fernández (2000) introduced the ordered median problem for the first time to location theory. They consider single-facility ordered median problems in the plane with a norm to measure the distances. No solution approaches are given in this work but the set of optimal solutions of this problem is studied and a characterisation of this set is given. So called ordered regions are defined as the set of points for which the permutation of the distance vector does not change. The intersection of these regions with the elementary convex sets introduced by Durier and Michelot (1994) form the so called ordered elementary convex sets on which the ordered median function is linear as the permutation does not change and the underlying norm is linear. The important result is that there always exists an optimal solution of the ordered median problem in an extreme point of the ordered elementary convex sets. In the convex case, the whole set of optimal solutions coincides with exactly one of the ordered elementary convex sets.

Rodríguez-Chía et al. (2000) deal with convex ordered median problems in the plane with underlying polyhedral gauges. Some important geometrical properties of the problem are described, for example the ordered regions on which the permutation in the objective function is constant. Based on this, a polynomial time descent algorithm is presented where for each ordered region a linear program is solved, i.e., either the locally best solution is found in this ordered region or it is detected that this region does not contain the global optimum. As the objective function is convex, an optimal solution found in the interior of a region is globally optimal. For an optimal solution on the boundary of a region a local search on the

neighboured regions has to be performed. Moreover, two possibilities for efficient algorithms for the general (non-convex) problem are stated. The approach is also adapted to the multi facility case. This procedure can only be applied efficiently for convex objective functions since in the non-convex case all ordered regions have to be enumerated to not stick in a local minimum. Moreover, a finite dominating set is constructed for the case that forbidden regions are added to the problem. The candidate set consists of all intersection points of the boundaries of the forbidden region with all fundamental directions and all bisectors.

Nickel et al. (2005) present a model for the convex multi-criteria ordered median problem with polyhedral gauges in the plane. At first the bi-criteria problem is analysed. It is shown that the set of Pareto solutions consists of complete cells, complete faces, and vertices of the so called ordered elementary convex sets in which the ordered median function is linear. The solution procedure starts in a lexicographic optimum of one objective and proceeds, using the connectedness of the set of Pareto optimal solutions, to another Pareto optimal intersection point of the structure induced by the ordered elementary convex sets until the other lexicographic optimum is reached. This result is generalised to three-criteria and later to Q-criteria problems, where the Q-criteria problems are solved by a reduction to a series of bi-criteria problems. Some ideas for the extension to the non-convex case are also given.

Another approach to multi-criteria ordered median problems can be found in **Ohsawa et al. (2007)**. For the measurement of distances the squared Euclidean distances are used because an advantage can be taken out of the fact that in this case the level curves of the objective function are circles. An algorithm to determine the Pareto optimal solutions of a problem with more than two convex ordered median objectives with the help of structures from computational geometry such as Voronoi diagrams and arrangements of curves and lines is introduced. In two objectives this approach is applicable to any type of ordered median objectives and any polygonally bounded feasible region. For more than two criteria, the objective functions and the feasible region have to be convex.

One of the important global optimisation approaches for the non-convex single facility ordered median problem in the plane is considered by **Drezner and Nickel (2009b)**. They give a geometric branch & bound method based on triangulations, employing the Big Triangle Small Triangle approach (see Drezner and Suzuki (2004)). The idea is to triangulate the feasible area by the Delaunay triangulation (see Lee and Schachter (1980)) with the demand points as vertices. In each iteration, each triangle is split into four smaller triangles. The values of the objective function in the center points of the triangles are used to derive a lower bound, and all triangles with a larger bound are discarded. Different (rigorous and heuristic) lower bounds are given that are based on the shortest distance between a node and any point in the triangle or the longest possible distance to one of the three vertices of the triangle.

Espejo et al. (2009) present an approach to solve the convex ordered median problem for the special case of distances measured with ℓ_p -norms with $1 < p < \infty$. Since the objective function is not differentiable at the demand points and the bisector lines, the objective function is approximated by an hyperbolic approximation. The proposed solution approach is based on a modified version of the gradient descent method that produces a descending sequence of objective values. For this purpose, it is distinguished between the two cases that

an iterate point lies in the interior or on the boundary of an approximated ordered region. The convergence of this sequence to the optimal objective function value is shown.

A second global optimisation approach is considered in the paper of **Drezner and Nickel (2009a)**. It shows that the continuous ordered median problem belongs to the class of d.c. optimisation problems, i.e. the ordered median function can be expressed as a difference of two convex functions, even though it is in general not a C^2-function. One way to transform a standard ordered median problem into a d.c. problem is stated. The resulting problem is then solved with procedures borrowed from d.c. optimisation. Here, this is used to construct efficient lower bounds. The geometrical branch & bound techniques as the Big Triangle Small Triangle method proposed in Drezner and Nickel (2009b) are modified using these new bounds to solve the planar single facility ordered median problem.

Krebs and Nickel (2010) analyse different properties of the continuous ordered median problem in the plane. Among others, a sufficient criterion for the separability of unweighted ordered median problem with an underlying norm is given, i.e., it is stated under which conditions a decomposition of the ordered median problem into several partial problems is possible. This decomposition of the problem can help to accelerate many existing solution approaches for the problem. Moreover, a descent algorithm acting on the finite dominating set consisting of the vertices of the ordered elementary convex sets proposed by Rodríguez-Chía et al. (2000) is introduced. The advantage of the presented approach is that, even for the non-convex case, not all elements of the finite dominating set have to be evaluated. The iterative algorithm is based on a graph structure which is implemented on the finite dominating set such that in each iteration only the evaluation of the nodes adjacent to the actual iteration point is required. Besides this, properties of the ordered median problems with attractive and repulsive locations are stated.

Blanco et al. (2013) analyse the problem of minimizing the ordered median (or ordered weighted average) function of finitely many rational functions over compact semi-algebraic sets. The problem is transformed into a higher dimensional space where it can be modelled as a polynomial optimisation problem. This problem can be solved by a series of semidefinite programming relaxations that converges to the optimal solution of the initial problem. Each of the auxiliary problems can be solved in polynomial time. This approach is also applied to a general family of location problems. By using a reformulation of these location problems, a series of relaxed problems similar to that of the general ordered weighted average problems can be obtained. Convex as well as non-convex location problems with ℓ_p-norms in finite dimensional spaces can be solved with this approach. Moreover, the sizes of the semidefinite programming relaxations are reduced further such that even larger instances of the problems can be solved in reasonable time. **Blanco et al. (2016)** use a similar approach to solve the continuous multi-facility ordered median location problem with ℓ_p-norms in any dimension.

Grzybowski et al. (2015a) use the fact that every continuous piecewise linear function has a max-min representation in terms of its linear functions. Since the ordered median function is piecewise linear, this leads to a min-max representation of the ordered median function using a combinatorial approach. Fans of ordered cones in $\mathbb{R}^d$, different orderings and some other theoretical properties of the ordered median function are introduced. Using this representation, Grzybowski et al. (2015b) give more theoretical properties of the

ordered median function and describe the ascent resp. descent cones to derive a topological classification of this function.

Of course, the ordered median objective function can also be applied to other optimisation problems, not only in location theory. An interesting field of application are, for example, combinatorial optimisation problems (see, e.g., **Fernández et al. (2014)** and **Fernández et al. (2016)**).

For an extensive review and discussion of the field of ordered median location problems see **Nickel and Puerto (2005)**. For an overview on global optimisation approaches for general planar location problems see, for example, **Drezner (2013)**.

3.5 k-max Optimisation

In a k-max problem, an optimal solution shall be determined such that the kth largest element of the argument vector is minimal among all feasible solutions. The field of k-max optimisation is a relatively new area of research and not much investigated until very recently. k-max problems have mostly been investigated in the context of discrete optimisation.

The notion of the kth largest element of a general set S can first be found in **Punnen and Aneja (2004)**. The minimisation of the maximal difference of the cost coefficients of a feasible solution to its kth largest cost coefficient is analysed for general combinatorial optimisation problems. This type of problem is called a lexicographic balanced optimisation problem.

k-max optimisation problems are discussed in **Gorski and Ruzika (2009)** as combinatorial problems with k-max objective function. The main aspect is the development of an easy and efficient solution approach based on the formulation of a series of substitute problems with a 0-1-objective function and using a bisection on the objective function value. This approach is a generalisation of the threshold algorithm introduced by Edmonds and Fulkerson (1970) and adopted by many other authors. The new version has a complexity of $\mathcal{O}(T \log(n))$, where n is the number of elements in the ground set and $\mathcal{O}(T)$ is the complexity of the substitute problems. This implies analogous solution approaches for discrete location problems with outliers, which can be deduced from the observations made.

A generalisation of the approach in Gorski and Ruzika (2009) to multi-cirteria combinatorial optimisation problems is given by **Gorski et al. (2012)**. The first objective function is of arbitrary type and the other objectives are either bottleneck or k-max objective functions. An efficient solution approach based on the iterative solution of ε-constraint scalarisations is proposed. Thereby, the scalarisation is independent of the specific combinatorial problem considered. Polynomial time algorithms for several important problem classes like spanning tree problems with bottleneck objectives, which are NP-hard in the general multiple objective case, are derived.

Turner (2011) applies this approach to k-max shortest path problems, where the costs of the kth largest edge shall be minimised. In each iteration it is tested by using bisection if there exists a path from s to t whose kth largest cost edge has costs smaller than the costs

c for a given edge. This is done iteratively by solving a binary sum shortest path problem where the costs are defined as zero or one.

Rong et al. (2013) consider the multi-criteria 0-1-knapsack problem where the first objective is a classical sum objective function and the other objectives are of a k-min type, i.e., the aim is to maximise the kth smallest objective coefficient in any feasible knapsack solution with at least k items in the knapsack. The whole non-dominated set is determined by using a series of single-criteria multidimensional 0-1-knapsack problems which are established by a variant of the ε-constraint method. These subproblems are solved by a hybrid two stage solution method composed of several techniques such as linear programming relaxation, dynamic programming, bounding techniques, state dominance relations, core concept and the greedy principle.

Obviously, the k-max problem is a generalisation of the bottleneck problem which takes the largest element of a feasible solution into account. Thus, for $k = 1$ the k-max problem equals a bottleneck problem. Many interesting generalisations of the bottleneck problem exist and they can not all be mentioned here. Only some of them shall be mentioned in the following, where the papers are chosen to give a good overview on the different types of bottleneck problems and different solution approaches.

Gabow and Tarjan (1988) propose solution methods for two bottleneck optimisation problems: The bottleneck spanning tree problem on a directed graph with n vertices and m edges and the bottleneck maximum cardinality matching problem in an undirected graph. For the spanning tree problem, the single-source shortest path problem of Dijkstra is modified slightly to obtain an $\mathcal{O}(\min\{n\log(n) + m, m\log(n)\})$-time algorithm. For the bottleneck maximum cardinality matching problem an $\mathcal{O}((n\log(n))^{\frac{1}{2}}m)$-time algorithm is developed using a binary search to find a bottleneck matching that does in general not have maximum cardinality but comes close to it.

Berman et al. (1990) considers two-criteria problems on networks, where one objective is any general cost function and the other objective is a bottleneck function. Three problems are analysed: The first problem minimises the bottleneck function subject to a bound on the cost function, the second problem deals conversely with the minimisation of the cost function subject to a constraint on the bottleneck function. In the last problem both criteria are considered simultaneously to find the Pareto optimal solutions of the problem.

Lexicographic bottleneck problems are discussed, for example, in **Burkhard and Rendl (1991)**. In addition to minimising the largest element of a feasible solution in the first place, also the second largest element of the optimal solutions of the bottleneck problem shall be optimised in the second place. Afterwards, under the solution alternatives, the solutions with the third and fourth minimal largest elements are sought and so on. This work considers only problems where every feasible solution has the same number of elements, e.g., travelling salesman problems and assignment problems. The basic idea is to model the problem as a sum optimisation problem and to redefine the costs in an adequate way. One approach consists then in a scaling algorithm on the cost coefficients where one sum optimisation problem is solved with redefined integer costs. A second procedure is an iterative approach where at first one bottleneck optimisation problem is solved to find the largest weight,

afterwards a series of sum problems with new weights is solved.

Punnen et al. (1995) consider a problem that generalises the combinatorial bottleneck problem and the combinatorial minsum problem in one. A ground set $\mathcal{E}$ is partitioned into α subsets $\mathcal{E}_\beta$ for $\beta = 1, \ldots, \alpha$ with the aim to minimise the sum of costs of those feasible solutions S that have maximum costs in the intersection $S \cap \mathcal{E}_\beta$ for all β. Note that for $\alpha = 1$ this problem is equal to a combinatorial bottleneck problem and for $\alpha = |\mathcal{E}|$ it equals a combinatorial minsum problem. Applications of this model to known problems are shown to be solvable in polynomial time provided that the value of α is fixed and that the associated feasibility problem can be solved in polynomial time.

Sokkalingam and Aneja (1998) analyse combinatorial lexicographic bottleneck problems, e.g., path, assignment and general matching problems, where the corresponding sum optimisation problem can be solved as a linear program. The algorithm works with a similar iterative approach as Burkhard and Rendl (1991), i.e., a series of bottleneck and 0-1-sum optimisation problems is solved such that the size of the problems is reduced in every iteration. Thereby, an upper bound on the number of iterations needed is known. Instead of applying a cost scaling procedure, only feasible solutions of the sum optimisation problem solved in a previous stage are used.

In **Bornstein et al. (2012)** multi-criteria combinatorial optimisation problems with one cost and several bottleneck objective functions are addressed, where the cost function can be of MinSum, MaxSum, MinProd or MaxProd type. A solution approach based on **Pinto and Pascoal (2010)** is presented that generates the minimal complete set of Pareto-optimal solutions as long as at least one optimal solution of the problem considering only the cost objective function is known. Then, the algorithm runs in polynomial time. The algorithm consists of two phases: The first phase computes the Pareto optimal candidates by fixing bounds on the bottleneck functions. The second phase compares the obtained candidates and deletes the dominated ones. Moreover, a reoptimisation procedure can be used to accelerate the solution of the single criteria problems in the first phase because the subproblems do not have to be started from scratch.

3.6 Other Anomaly Detection Techniques

This section deals with general techniques for detecting anomalies (or often also called *outliers*) in diverse research areas and application domains. Anomaly detection is the problem to find patterns in data sets that do (however) not show the expected behaviour that most of the data points show. This topic is used in various applications as, for example, fraud detection for credit cards, electronic commerce, health care, intrusion detection for cybersecurity and military surveillance for enemy activities. Outlier detection is an important action since anomalies often correspond to significant, often even critical information on which there is a need to react in some way. As applied in many fields of science, there is no single formal definition of an outlier. Hawkins (1980) defined them as follows: "*An outlier is an observation that deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism.*" Detecting outliers origins in the field of statistics

in the 19th century but is today applied in many research areas. A multitude of outlier tests has been developed. Some key models and tools for outlier analysis are extreme values, statistical models, clustering models, distance-based models and density based models. In the following, these models are described shortly to get an general overview of existing methods. More information on different anomaly detection methods can, for example, be found in **Hawkins (1980)**, **Chandola et al. (2009)** and **Aggarwal (2013)**.

Note that all the here described techniques can in general also be applied to identify outliers among the existing facilities in (continuous) location problems by identifying the customers with points in $\mathbb{R}^d$. These outliers can be defined differently depending on the chosen detection method and do not have to equal the definition of outliers used in this thesis for the p-k-max problems.

The most basic form of outlier detection is the extreme value analysis of 1-dimensional data which deals with the extreme deviations from the median of probability distributions. Extreme values are a very specific kind of outliers since only the data points at the margin of the data set are assumed to be outliers. These anomalies correspond to the statistical tails of underlying probability distributions. The behaviour of the maxima (resp. minima) can be described by the three extreme value distributions Gumbel, Féechet and negative Weibull which are introduced by **Fisher and Tippett (1928)**. They prove that the distribution of the maximum values of data samples tends to one of these three limiting distributions when the sample size increases.

The general idea of a statistical model is to assume that the data points are generated by a mixture of b different distributions with the probability distributions $G_1, \ldots, G_b$. Support vector machines and machine learning play an important role in this context. **Eskin (2000)** use a so called *mixture model*: With a small probability λ, an element of the model is an anomaly and with probability $(1 - \lambda)$, the element is a normal element. The corresponding probability distributions are called A for the distribution of the normal data and B for the anomaly distribution. Detecting the anomalies means then to identify which data points are generated by distribution A and which by B. At the beginning all elements are assumed to be in A. Each element of A is then tested for being an outlier by computing the difference in the log likelihood value of the distribution A if this element is removed and included in distribution B. If this value exceeds a predefined threshold, the element stays in B, otherwise it is returned back to A. Therefore, in every step the method needs to recompute the probability distributions.

A clustering technique can be used to group similar data instances into clusters. Every data point does either belong to a cluster or is an outlier in this context. Several cluster-based approaches exist for anomaly detection. **Smith et al. (2002)**, for example, use different approaches as self-organising maps, k-means clustering and expectation maximisation to cluster training data and to classify test data afterwards. The approach is based on two steps: At first the data points are clustered and then the distances to the centers of their closest clusters are computed. This value is the *anomaly score* and decides if a point is an outlier or not. The disadvantage of clustering based detection techniques is that most algorithms are originally not intended to find anomalies but to find clusters which is kind of

a complementary aim.

The last two model classes discussed here are distance- and density based models. They belong to the class of nearest-neighbour-based models. The assumption here is that normal data instances arise in dense neighbourhoods whereas outliers lie far from their closest neighbours. Thus, this approach is quite similar to the location approach with k-max functions.

A simple distance-based technique is based on defining the anomaly score of a data point as the distance to its hth nearest neighbour in the given data set. Usually, this value is taken as a threshold to decide whether a test instance is an outlier or not. **Angiulli and Pizzuti (2002)** extend this basic approach by varying the definition of the anomaly score of a data point to the sum of its distances to its h nearest neighbours. The proposed algorithm uses a linearisation of the search space by a Hilbert filling curve to efficiently determine the h nearest neighbours and consists of two phases. The first phase uses an approximation approach to reduce the number of candidate points that might belong to the solution set. The second phase determines an exact solution by evaluating the remaining candidates in an efficient way.

The idea of a density-based detection technique is to compute the density of the neighbourhood of each data point. A point that belongs to a neighbourhood with low density is assumed to be an anomaly while points in neighbourhoods with high density are normal. Many density based methods perform rather poorly if the densities of the different regions are varying greatly. To handle this, several techniques to consider the density of a data point relative to the density of its neighbours are developed. **Breunig et al. (2000)**, for example, define local outliers with the help of the *local outlier factor* (LOF) which measures how isolated the data point is w.r.t. the direct neighbourhood. For a fixed data point, the factor is given as the ratio of the average local density of its h nearest neighbours and its own local density. For an outlier, its local density is then smaller than the density of its nearest neighbours and thus the LOF is higher than the score of normal instances.

Chapter 4

Network Location Problems with Outliers

In this chapter location problems with k-max functions on networks are introduced. For the sake of readability, the name "p-k-max location problem" is often abbreviated as "p-k-max problem" throughout the rest of this thesis as only location problems are considered. Moreover, the terms "network" and "graph" are used equivalently in the following.

In the first section the p-k-max problem on networks is defined formally and some basic properties of this problem are proven. An important part for the further analysis of outliers in center location problems is the relation between p-k-max problems and p-center problems which is presented in the second section of this chapter. With this relation it is possible to get more insight into important properties of the optimal solution set of the p-k-max problem. These properties play an important role throughout this thesis. Moreover, a first algorithm for solving the p-k-max problem is derived. The next section introduces three possible (mixed-) integer programming formulations of the p-k-max problem. The first formulation combines a sorting model and a location-allocation model. Two further formulations avoid the sorting process. The last section introduces the sets of the equilibrium- and the bottleneck points, which are essential for nearly all solution approaches presented in this thesis. The algorithm of Bentley and Ottmann (1979) to compute these points is reviewed in detail.

4.1 Definitions and Basic Properties

To define the p-k-max problem formally, some notations are introduced. Let $G = (V, E)$ be a finite, connected, simple and undirected graph, specified by a set of nodes V and a set of edges E. The set $V = \{v_1, \ldots, v_n\}$ represents the finite set of existing facilities (also called *customers*). With $w : V \to \mathbb{R}_+$ a positive weight $w_i > 0$ is assigned to every node $v_i,\ i = 1, \ldots, n$. The weights express different levels of importance of the customers, e.g., the customers demand.

The set $E = \{e_1, \ldots, e_m\} \subseteq V \times V$ contains the m undirected edges of the graph G. By $l : E \to \mathbb{R}_+$ every edge e_j is associated with a strictly positive length $l_j > 0,\ j = 1, \ldots, m$. If the endpoints of the edge e_j need to be emphasised, e_j is also denoted by e_{ab} or (v_a, v_b), where $v_a, v_b \in V$ are the endpoints of e_j. Similarly, the notations l_j and l_{ab} are used interchangeable to denote the length of e_j. Note that the length l_j of an edge e_j does not necessarily represent the travel distances between its two endnodes. It can also express a time or the costs needed for travelling this way, or a measurement of importance.

Distances along an edge are linear parametrised, i.e., a point x on an edge $e_{ab} \in E$ can be represented by the pair $x = (e_{ab}, t)$ with $t \in [0, 1]$. A subset of an edge e_j from the point

$x' \in e_j$ to the point $x'' \in e_j$ is denoted by the interval $[x', x''] \subseteq e_j$ and often referred to as a *subedge* of e_j. The continuum set of points on the edges of G is denoted by $A(G)$. Note that $V \subseteq A(G)$.

Let the integer $p \in \{1, \ldots, n\}$ describe the number of new facilities to locate and let $X = \{x_1, \ldots, x_p\} \subseteq A(G)$ denote a feasible set of new facilities. The distance between a single point $x = (e_{ab}, t)$ and an arbitrary node $v_i \in V$ is given by

$$d(v_i, x) = \min\{d(v_i, v_a) + t l_{ab}, d(v_i, v_b) + (1 - t) l_{ab}\}.$$

The component-wise weighted distance from the finite set of existing facilities V to the closest new facility in the set $X \subseteq A(G)$ is then defined as the vector

$$d^w(V, X) = (d^w(v_1, X), \ldots, d^w(v_n, X))^\top = (w_1 d(v_1, X), \ldots, w_n d(v_n, X))^\top$$

with

$$d(v_i, X) = \min_{x \in X} d(v_i, x),$$

where $d(v_i, x)$ is the length of a shortest path connecting v_i and x in G as defined above. The so defined distance function $d(v_i, x)$ defines a metric: For all $v_i \in V$ and $x \in A(G)$ it holds that $d(v_i, x) \geq 0$ with $d(v_i, x) = 0$ if and only if $v_i = x$ (non-negativity), $d(v_i, x) = d(x, v_i)$ (symmetry) and $d(v_i, x) \leq d(v_i, u) + d(u, x)$ for all $u \in A(G)$ (triangle inequality).

Remark 4.1. *Note that for a fixed edge e_{ab}, $a, b \in \{1, \ldots, n\}$, there is a one to one correspondence between a point $x = (e_{ab}, t)$ and the parameter $t \in [0, 1]$. As soon as an edge e_{ab} is fixed, a point $x = (e_{ab}, t)$ can simply be identified by the parameter t. Throughout this thesis, plots of the weighted distance functions $d^w(v_i, x)$ for $i \in \{1, \ldots, n\}$ over a fixed edge e_{ab} will be with respect to $t \in [0, 1]$ even though the distance functions and most of the argumentation will in general be with respect to x.*

For p-k-max problems on graphs, $k \in \{1, \ldots, n\}$ denotes the parameter of the k-max function that specifies the kth largest distance to be minimised.

Definition 4.2 ▶ p-k-max location problem on a network

Let $G = (V, E)$ be a graph as defined above with demands $w_i > 0$, $i = 1, \ldots, n$. For $k \in \{1, \ldots, n\}$ and $X = \{x_1, \ldots, x_p\}$ with $p \in \{1, \ldots, n\}$, the p-k-max *problem* is given by

$$\min_{X \subseteq A(G)} f_k(X) = \min_{X \subseteq A(G)} k\text{-max}(d^w(V, X)) = \min_{X \subseteq A(G)} w_{\sigma(k)} d(v_{\sigma(k)}, X), \qquad \text{(pkMG)}$$

where $\sigma \in \Sigma(X)$ is a permutation of the existing facilities depending on X, i.e., it satisfies

$$d^w(v_{\sigma(1)}, X) \geq d^w(v_{\sigma(2)}, X) \geq \ldots \geq d^w(v_{\sigma(n)}, X). \qquad (4.1)$$

The general p-k-max problem on a network as defined above can be seen as a p-center problem with automatic outlier detection for a given parameter $k \in \{1, \ldots, n\}$. Note that the permutation σ depends on the current solution X and may change whenever X is changed.

The single facility problem is a special case of the above problem for $p = 1$.

A set of outliers for a feasible solution X w.r.t. the corresponding permutation σ is defined as the set

$$V_{k-1} = \{v_{\sigma(1)}, \ldots, v_{\sigma(k-1)}\}.$$

Due to $|\Sigma(X)| \geq 1$, the set of outliers w.r.t. a feasible solution X does not have to be unique. When in the following the notation V_{k-1} for the set of outliers w.r.t. X is used, an arbitrary but fixed selection is assumed in the case that V_{k-1} is not unique. A set of outliers corresponding to an optimal solution is in the following often denoted by V^*_{k-1}.

Remark 4.3. *The relation $|V_{k-1}| = k-1$ holds, i.e. the number of outliers is always smaller by one than the parameter k and at least $n-k+1$ existing facilities have to be covered with service within the coverage radius $z = k\text{-max}(d^w(V, X))$. Thus, the number of outliers is determined by k.*

Note that with this definition of V_{k-1} based on the $k-1$ largest distances, it does not mean that all outliers are excluded from service or even have a weighted distance larger than the objective function value. More precisely, the number of the facilities that are not covered may be smaller, for example, if the distances $d^w(v_{\sigma(c)}, X), \ldots, d^w(v_{\sigma(k)}, X)$ are equal for some $1 \leq c < k$. Hence, the number of $k-1$ outliers is an upper bound for facilities not receiving service within the coverage radius z (see also Figure 2.2 for an illustration of the continuous case).

Definition 4.4 ▶ Center defining nodes

Let $V_{k-1} \subset V$ be a set of outliers for the current solution X. The nodes in the set $V \setminus V_{k-1}$ are called *center defining nodes*.

Remark 4.5. *As for continuous p-k-*max *problems it is also possible to include zero weighted nodes into the problem, i.e., nodes $v_i \in V$ with $w_i = 0$. From a practical point of view this is often not useful since a zero weighted node corresponds to an existing facility that has no demand and should therefore not have any influence on the location of the new facility. Nevertheless, zero weighted facilities will later be needed for technical reasons to retain the structure of the network for distance evaluations. This occurs, for example, when the demand of one or several nodes is to be ignored while the corresponding nodes are still needed as the corresponding adjacent edges can not be deleted. In such cases, the notation will be partially adapted.*

*Let $G = (V_G, E_G)$ be a graph where $V_G = \{v_1, \ldots, v_{n_G}\}$ is the finite set of existing facilities that are represented by the n_G vertices of the graph G and the set of edges E_G. The weight function $w : V_G \to \mathbb{R}_+$ assigns a non-negative weight $w_i \geq 0$ to all $v_i \in V_G,\ i = 1, \ldots, n_G$. Moreover, let $V := \{v_i \in V_G : w_i > 0,\ i \in \{1, \ldots, n_G\}\}$ be the set of facilities with strictly positive weights and $|V| = n$. The p-k-*max *problem is then defined only with respect to the facilities in V:*

$$\min_{X \subseteq A(G)} f_k(X) = \min_{X \subseteq A(G)} k\text{-max}(d^w(V, X)) = \min_{X \subseteq A(G)} w_{\sigma(k)} d^w(v_{\sigma(k)}, X),$$

with $k \in \{1, \ldots, n\}$ and a permutation σ such that

$$d^w(v_{\sigma(1)}, X) \geq d^w(v_{\sigma(2)}, X) \geq \ldots \geq d^w(v_{\sigma(n)}, X), \quad v_{\sigma(i)} \in V,\ i = 1, \ldots, n.$$

As a consequence, existing points with weights equal to zero do not contribute to the objective function as it holds that $w_i d(v_i, x) = 0$. Thus, the zero-weighted facilities will usually not be outliers, as long as the parameter k is smaller than the number of positively weighted customers minus p. It holds $V^*_{k-1} = \{v_{\sigma(1)}, \ldots, v_{\sigma(k-1)}\} \subseteq V$. Otherwise, i.e., if $n - p < k \leq n_G$, the p-k-max problem can be trivially solved with optimal objective function value of 0.

In the following it is assumed that all weights are strictly positive unless it is stated otherwise. Note that all results of this thesis remain valid for $V \subsetneq V_G$ since the condition $V = V_G$ is not used as a necessary requirement.

After having defined the p-k-max problem properly, an important result concerning the solvability of the problem is stated.

Theorem 4.6 ▶ Existence of an optimal solution
Problem (pkMG) has an optimal solution.

Proof. The feasible region for new locations is closed and bounded. Every weighted distance function $d^w(v_i, x)$, $v_i \in V$ with $i \in \{1, \ldots, n\}$, is continuous over a fixed edge $e_{ab} \in E$. As the k-max function is a sum of these continuous weighted distance functions, the k-max function is continuous on the feasible area. Moreover, the objective value is bounded from below by 0 because of the positivity of the edge lengths $\infty > l_j > 0$ for all $j = 1, \ldots, m$ and the positivity of the demands $w_i > 0$ for all $i = 1, \ldots, n$. Thus, the result is shown. ■

The following section provides further properties of p-k-max problems and the corresponding outliers.

4.2 Relation between p-k-max Problems and p-Center Problems

After describing the basic p-k-max problem and some of its properties, the focus is now on the relation between the p-k-max problem and the well known p-center problem on graphs. Using the same notation as for the p-k-max problems, the p-center problem on G is defined by minimizing the center function

$$\min_{X \subseteq A(G)} f_{pc}(X) = \min_{X \subseteq A(G)} \max_{v_i \in V} d^w(v_i, X).$$

This problem is a special case of (pkMG) for the choice of $k = 1$, i.e., p new facilities are to be located such that the largest distance between a vertex and it's closest new facility is minimal.
The p-center problem on networks is very well studied and discussed in many papers, see e.g., the seminal works by Hakimi (1964) and Kariv and Hakimi (1979) and Chapter 3 for

a brief review. Thus, it would be nice to utilize the relation between these two problems. A very useful relation can be given as follows.

Lemma 4.7 ▶ Reduction to p-center problems

If the set V_{k-1}^* of optimal outliers for an optimal solution X^* of the p-k-max problem is known, every p-center optimum $\bar{X}$ w.r.t. the existing facilities in $V \setminus V_{k-1}^*$ is optimal for the p-k-max problem with $f_k(X^*) = f_k(\bar{X}) = f_{pc}(\bar{X})$, where f_{pc} is evaluated w.r.t. $V \setminus V_{k-1}^*$.

Proof. Let V_{k-1}^* denote an optimal set of outliers for an optimal solution X^* of the p-k-max problem on $G = (V, E)$, $|V_{k-1}^*| = k - 1$. Assume that there exists a p-center optimum $\bar{X}$ w.r.t. the existing facilities in $V \setminus V_{k-1}^*$ that is not optimal for the p-k-max problem. Thus it holds $f_k(X^*) < f_k(\bar{X})$. Since $f_k(X^*)$ is the kth largest distance among the facilities of V to the nearest point in X^*, it follows that

$$f_k(X^*) = \max_{v_i \in V \setminus V_{k-1}^*} w_i d(v_i, X^*) < f_{pc}(\bar{X}).$$

This contradicts the assumption that $\bar{X}$ is an optimal p-center solution on $V \setminus V_{k-1}^*$ and the statement follows. ∎

Lemma 4.7 implies that the solution of (pkMG) can be reduced to the solution of a p-center problem on a subset of the existing facilities. However, to get this set of nodes (i.e., $V \setminus V_{k-1}^*$) for the associated p-center problem, an optimal set of outliers has to be already known and excluded from V. Of course, finding these outliers is an optimisation problem in itself and not easy in general. Lemma 4.7 can be strengthened in the sense that there is not only an implication from the optimal solution of the p-center problem w.r.t. $V \setminus V_{k-1}^*$ to the p-k-max problem w.r.t. V, but also vice versa. Moreover, both problems have the same optimal objective function value.

Theorem 4.8 ▶ Relation between p-k-max and p-center problems

Let V_{k-1}^* denote an optimal set of $k - 1$ outliers for the p-k-max problem on G. Then it holds: $X^* = \{x_1^*, \ldots, x_p^*\}$ is optimal for the p-k-max problem with set of outliers V_{k-1}^* if and only if X^* is optimal for the p-center problem w.r.t. $V \setminus V_{k-1}^*$. It holds $f_k(X^*) = f_{pc}(X^*)$, where f_k is evaluated w.r.t V and f_{pc} is evaluated w.r.t. $V \setminus V_{k-1}^*$.

Proof. Only one direction of the statement has to be shown as the other direction follows directly with Lemma 4.7.

Let $X^* = (x_1, \ldots, x_p)$ be an optimal solution of the p-k-max problem on $G = (V, E)$ with optimal objective function value $f_k(X^*)$, and let V_{k-1}^* be a (not necessarily unique) optimal set of $k - 1$ outliers for X^*. Moreover, the optimal objective function value $f_k(X^*)$ equals the optimal objective function value $f_{pc}(X^*)$ of the p-center problem for $V \setminus V_{k-1}^*$ because

$$f_k(X^*) = \max_{v_i \in V \setminus V_{k-1}^*} \min_{\ell=1,\ldots,p} d^w(v_i, x_\ell) = f_{pc}(X^*),$$

where f_k is evaluated w.r.t V and f_{pc} is evaluated w.r.t. $V \setminus V^*_{k-1}$. Assume that X^* is not optimal for the p-center problem on $V \setminus V^*_{k-1}$. Then it holds for an optimal solution $\bar{X}$ of the p-center problem for $V \setminus V^*_{k-1}$ that

$$f_k(\bar{X}) \leq f_{pc}(\bar{X}) < f_{pc}(X^*) = f_k(X^*),$$

where the first inequality holds because

$$f_k(\bar{X}) = \min_{\substack{\bar{V}_{k-1} \subseteq V, \\ |\bar{V}_{k-1}|=k-1}} \max_{v_i \in V \setminus \bar{V}_{k-1}} w_i d(v_i, \bar{X}) \leq \max_{v_i \in V \setminus V^*_{k-1}} w_i d(v_i, \bar{X}) = f_{pc}(\bar{X}).$$

It follows that $f_k(\bar{X}) < f_k(X^*)$ which is a contradiction to the assumption that X^* is optimal for the p-k-max problem. ∎

To get an intuitive idea of the set $V \setminus V^*_{k-1}$, two important terms related to the relation between p-center and p-k-max problems are introduced in the following. Moreover, the concept of "most contiguous facilities" as defined below is applied to formulate an algorithm that determines a (not necessarily unique) set $V \setminus V^*_{k-1}$.

Definition 4.9 ▶ Most contiguous facilities

Let $p \in \mathbb{N}$ be the number of new facilities to locate and let $r \in \{1, \ldots, n\}$ be fixed. Moreover, let

$$\mathcal{R}_r := \{R : R \subseteq \{1, \ldots, n\},\ |R| = r\}.$$

The existing facilities of a graph G in the set $V_{R^*} = \{v_{i_1^*}, \ldots, v_{i_r^*}\} \subseteq V$ with $|V_{R^*}| = r$ and set of indices $R^* = \{i_1^*, \ldots, i_r^*\} \in \mathcal{R}_r$ are called *more contiguous* w.r.t. p than facilities $V_{\widetilde{R}} = \{v_{\tilde{i}_1}, \ldots, v_{\tilde{i}_r}\} \subseteq V$ with $\widetilde{R} = \{\tilde{i}_1, \ldots, \tilde{i}_r\} \in \mathcal{R}_r$ if the p-center problem with existing facilities V_{R^*} realises a smaller objective function value than the p-center problem with existing facilities $V_{\widetilde{R}}$.

If V_{R^*} attains the smallest p-center function value under all possible subsets $R \in \mathcal{R}_r$, i.e., if

$$\operatorname*{arg\,min}_{R \in \mathcal{R}_r} \left\{ \min_{X \subseteq A(G)} \left\{ \max_{\substack{i_s \in R \\ s=1,\ldots,r}} w_{i_s} d(v_{i_s}, X) \right\} \right\} =: R^*,$$

the nodes in V_{R^*} are called *most contiguous* w.r.t. p.

The sets $V_R,\ R \in \mathcal{R}_r$ defining the above defined property are called "more contiguous" resp. "most contiguous facilities" in the following. Note that these properties depend on the value of p even though it is not always explicitly mentioned in the following. The expression "further facilities" is used equivalently to denote a set $V_{\widetilde{R}}$, $\widetilde{R} \in \mathcal{R}$, of facilities that realises a larger p-center value than another set V_{R^*}, $R^* \in \mathcal{R}$. Note that this definition can analogously be applied in a continuous problem setting. Example 4.10 gives an illustration of the situation.

Example 4.10. *A graph G with $|V| = 5$, $|E| = 6$ and all weights equal to one is given and a set of most continuous facilities w.r.t. $p = 1$ with $r = 4$ elements is sought. In the left*

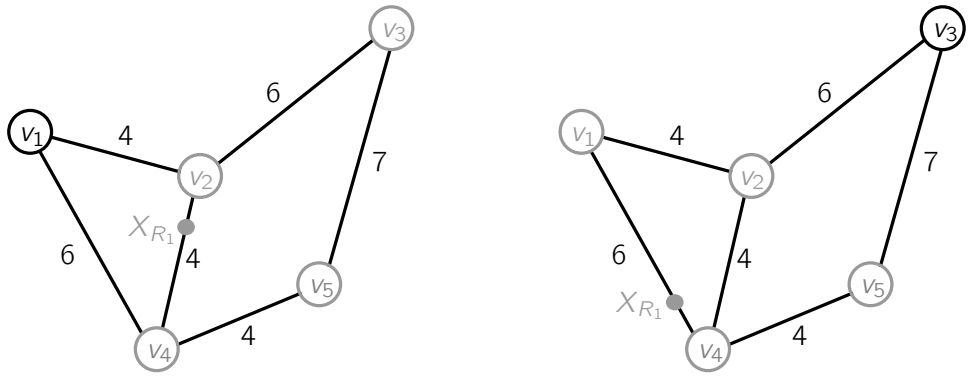


Fig. 4.1: For $r = 4$, the set $V_{R_2} = \{v_1, v_2, v_4, v_5\}$ is more contiguous w.r.t. $p = 1$ than the set $V_{R_1} = \{v_2, v_3, v_4, v_5\}$, because it provides a smaller center objective value. Moreover, V_{R_2} is the set of the most contiguous facilities w.r.t. $p = 1$ of G.

subfigure of Figure 4.1, the set $V_{R_1} = \{v_2, v_3, v_4, v_5\}$ *is selected. This subset of* V *has its optimal center location in* $X_{R_1} = \left(e_{42}, \frac{3}{4}\right)$ *with objective value* $z_{R_1} = 7$*. The alternative set* $V_{R_2} = \{v_1, v_2, v_4, v_5\}$ *in the right subfigure yields a center objective value of* $z_{R_2} = 5$ *with location at the center* $X_{R_2} = \left(e_{14}, \frac{5}{6}\right)$*. Thus,* V_{R_2} *is more contiguous than* V_{R_1} *because of the smaller center objective value.* V_{R_2} *is also the set of the most contiguous facilities because all other possible subsets* $V_R \subset V$ *with* $R \in \mathcal{R}_4$ *have a larger center value as can be easily verified. Therefore, it follows with Theorem 4.8 that* X_{R_2} *is the optimal solution of* (pkMG) *with* $k = 2$ *and* $p = 1$.

Summarizing the discussion above, a p-center problem for the (not necessarily unique) set of most contiguous facilities $V_{R^*} \subseteq V$ with cardinality $n - k + 1$ can be solved to obtain an optimal solution of the p-k-max problem on G. Also the converse is valid, i.e., an optimal solution for the p-center problem for V_{R^*} can be derived from an optimal location of pkMG.

Corollary 4.11 ▶ Relation p-k-max and p-center
$X^* = \left(x_1^*, \ldots, x_p^*\right)$ is optimal for the p-k-max problem if and only if X^* is optimal for the p-center problem on a set of most contiguous facilities $V_{R^*} \subseteq V$ with $R^* \in \mathcal{R}_{n-k+1}$.

Based on the statement of Corollary 4.11, a first algorithm for solving the p-k-max problem can be developed. The idea is to determine a set of most contiguous facilities in V and then to solve a p-center problem on this subset of existing facilities with an appropriate algorithm (see, for example, Tansel et al., 1983). The advantage of this procedure is that the p-center problem is convex. Therefore, the problem of analysing the non-convex p-k-max problem can be reduced to subproblems that are comparably easy to solve.

The easiest way to get the most contiguous facilities is by a complete enumeration as described in Algorithm 2. For that purpose, all possible subsets of V with cardinality $n-k+1$ are analysed for their p-center objective value. Every subset with the smallest value is a most contiguous subset. This approach has a complexity of $\mathcal{O}(n^{n-k+1} \cdot g(n, k, p))$ where $g(n, k, p)$ is the complexity of solving a p-center problem with $n-k+1$ customers in G and the number

of subsets to be investigated is

$$|\mathcal{R}| = \binom{n}{n-k+1} = \mathcal{O}(n^{n-k+1}).$$

The p-center problem can, for example, be solved with the algorithm of Drezner (1984) in $\mathcal{O}(n^{2p+1}\log(n))$ time (see also the other references regarding the p-center problem on networks in Chapter 3). Therefore, it follows an exponential overall complexity of $\mathcal{O}(n^{n-k+2p+2}\log(n))$ for Algorithm 2. Note that the algorithm can be parallelised easily as every set V_R can be evaluated independently from the others.

Algorithm 2 Solving p-k-max problems based on most contiguous sets

Input: Graph $G = (V, E)$ with $w_i > 0 \; \forall \; i = 1, \ldots, n$, $k \in \{1, \ldots, n-p\}$, $p \in \{1, \ldots, n\}$

1: Set $\mathcal{X} := \emptyset$, $z_{best} := \infty$
2: **for all** $R \in \mathcal{R}_{n-k+1}$ **do**
3: Determine the set of optimal solutions $\mathcal{X}_R$ of the p-center problem on $V_R \subset V$
4: Set $z_R := f_{pc}(X_R)$ with $X_R \in \mathcal{X}_R$ ▷ Optimal center-value of V_R
5: **if** $z_R < z_{best}$ **then** ▷ New most contiguous set
6: $\mathcal{X} := \mathcal{X}_R$
7: $z_{best} := z_R$
8: **else if** $z_R = z_{best}$ **then** ▷ Currently most contiguous sets
9: $\mathcal{X} := \mathcal{X} \cup \mathcal{X}_R$

Output: Set of optimal solutions $\mathcal{X}$ with $z^* := z_{best}$.

A possibility to reduce the number of candidate sets is described in Lemma 5.5 in Section 5.1 below and in the following explanations for unweighted *1-k*-max problems. A theoretical improvement in terms of the worst case complexity bound can not be shown but the practical savings depending on the structure of the graph may speed up the solution process. Moreover, more efficient concepts for solving the p-k-max problem are analysed and presented in Chapter 5 for the special case of $p = 1$ and in Chapter 6 for $p \geq 2$.

Remark 4.12. *The approach of testing all subsets of existing facilities of size $n-k+1$ can also be applied to continuous p-k-*max *problems as given in Definition 2.13 since the relation between a p-k-*max *problem and a p-center problem holds equivalently in the continuous case.*

Corollary 4.13 ▶ Uniqueness

Let X^* be an optimal solution of the p-k-max problem on $G = (V, E)$. Then it holds: X^* is unique if and only if all sets of most contiguous facilities $V_R \subseteq V$ with $R \in \mathcal{R}_{n-k+1}$ have the same unique p-center solution X^*.

The situation that the set of most contiguous facilities is not unique is not necessarily an exceptional case. It may occur quite often depending on the lengths of the edges and the weights of the nodes. All sets of most contiguous facilities have the same unique p-center solution if, for example for $p = 1$ and $w_i = 1$ for all $i = 1, \ldots, n$, the center point is in a node and all existing nodes are connected by an edge of length one just to this center point. Then it does not matter which nodes, except for the center, are the outliers. Because of that all subsets of V of size $n - k + 1$ are most contiguous facilities with the same center point.

As a further important fact concerning (pkMG), the NP-hardness of the multi-facility p-k-max problem on a general graph, can be proven using the above results.

Theorem 4.14 ▶ NP-hardness

The p-k-max problem on a graph G is NP-hard for all $k \geq 1$ arbitrary, but fixed.

Proof. Suppose w.l.o.g. that $w_i \geq 1$ for all $v_i \in V$. The NP-hardness of the p-k-max problem is shown by a reduction from the p-center problem. Consider an instance of the p-center problem on a finite, connected, simple and undirected graph $G = (V, E)$ with $|V| = n$ and $|E| = m$. This problem is known to be NP-hard which was shown by Kariv and Hakimi (1979). Let z_1 be the optimal objective value of the 1-center problem on G.

It is assumed that n is large enough such that $k < n - p + 1$ (otherwise the problem is trivially solvable). The p-center problem should now be solved using an instance of (pkMG) with $n + k - 1$ vertices defined as follows: The set of nodes V of G is extended by $k - 1$ artificial nodes v_{n+a}, $a = 1, \ldots, k - 1$. Connect the new vertices to the nodes of G by edges to node $v_1 \in V$ with length $l_{1,n+a} = 2z_1 + 1 \quad \text{for} \quad a = 1, \ldots, k - 1$. Then, the resulting graph $G' = (V', E')$ with $|V'| = n + k - 1$ and $|E'| = m + k - 1$ is connected (see Figure 4.2 for an illustration). The weights of the additional nodes are set to $w_{n+a} = 1$ for all $a = 1, \ldots, k - 1$. By this construction it is guaranteed that $d(v_i, v_s) > 2z_1$ for all $v_i \in V$ and $v_s \in V' \setminus V$, i.e. the distance from an arbitrary new vertex $v_s \in V' \setminus V$ to each vertex $v_i \in V$ and also the distances between pairs of the artificial vertices are larger than z_1.

Suppose now that $X^* = \{x_1^*, \ldots, x_p^*\} \subseteq A(G')$ is an optimal solution of the p-k-max problem w.r.t. V' with optimal objective value z^*. Then $z^* \leq z_1$, since locating one new facility (w.l.o.g. x_1) in an optimal 1-center location w.r.t. V and setting $V_{k-1} = V' \setminus V$ always yields a feasible solution of the p-k-max problem w.r.t. V' with objective value $z \leq z_1$ (and X^* is assumed to be optimal for the p-k-max problem w.r.t. V'.)

Case 1: $V_{k-1}^* = V' \setminus V$
With Corollary 4.11 it follows that the optimal solution X^* of the p-k-max problem w.r.t. V' is also an optimal solution of the p-center problem w.r.t. V.

Case 2: $v_s \notin V_{k-1}^*$ for at least one $v_s \in V' \setminus V$
Assume w.l.o.g. that $x_1^* \in X^*$ is the new facility that covers the artificial node v_s. Since $z^* \leq z_1$, x_1^* covers only v_s and no other node of G' in this solution. An alternative optimal solution $\bar{X} = \{\bar{x}_1, x_2^*, \ldots, x_p^*\}$ of the p-k-max problem can be constructed out of X^* by locating $\bar{x}_1$ in an arbitrary outlier $v_t \in V_{k-1}^* \cap V$ and setting the artificial node

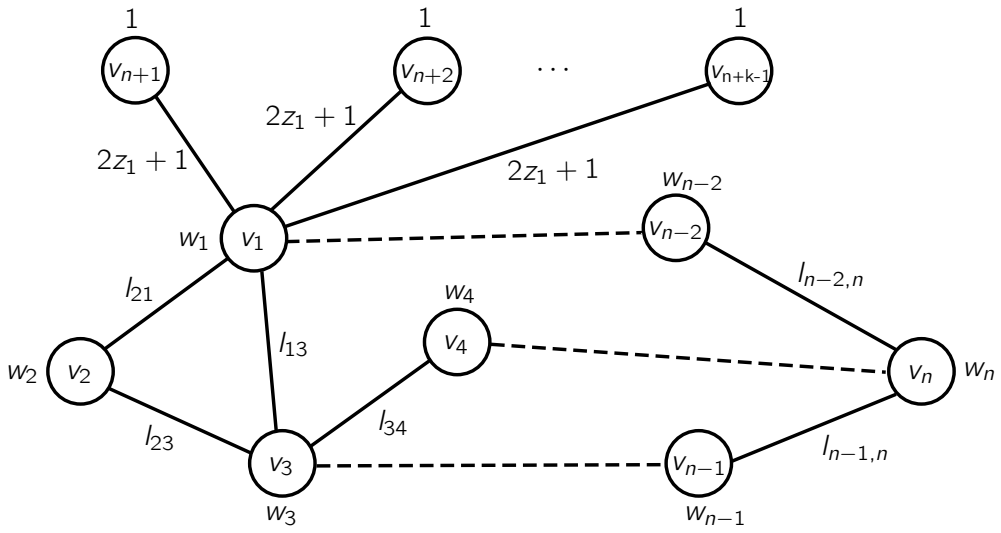


Fig. 4.2: Instance of (pkMG) on $G' = (V', E')$ with $V' = \{v_1, \ldots, v_{n+k-1}\}$ constructed for a given instance of the p-center problem on $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$

v_s to be an outlier instead. As x_1^* only covered one node, $\bar{x}_1$ does not have to cover more nodes than v_t to guarantee that $\bar{z} = z^*$.

This procedure is iterated until a new solution $\bar{\bar{X}} = \{\bar{\bar{x}}_1, \ldots, \bar{\bar{x}}_p\}$ is obtained with $\bar{\bar{x}}_1, \ldots, \bar{\bar{x}}_p \in A(G)$ such that the corresponding set of outers is $\bar{\bar{V}}_{k-1} = V' \setminus V$. Thus, the condition of Case 1 is satisfied and $\bar{\bar{X}}$ is also an optimal solution of the p-center problem w.r.t. V.

Summarizing, an optimal solution of an instance of the p-center problem can be found by solving a p-k-max problem on the extended graph G'. This transformation is done in polynomial time since the 1-center problem on G can be solved in polynomial time to obtain z_1 and at most p iterations of the shifting are needed in Case 2 to obtain a suitable solution. Therefore, the NP-hard p-center problem is polynomial time reducible to the p-k-max problem and thus (pkMG) is NP-hard itself. ∎

As a consequence, the p-k-max problem can not be solved by an algorithm in polynomial time, provided that $P \neq NP$. Note that, in the case of $p > n - k + 1$, the p-k-max problem can trivially be solved (see also Lemma 6.1 and Lemma 6.2).

4.3 Mixed Integer Formulations of the p-k-max Problem

In this section, three formulations of the p-k-max problem based on (mixed-) integer programming models are introduced. The first integer linear programming formulation is composed of two subproblems, where the first one corresponds to the sorting of the weighted

distance vector and the second one is a location-allocation problem to locate the new facilities. The second formulation leads also to an integer piecewise linear programming model, but without explicitly formulating the sorting of the distance vector. The third formulation is, moreover, not based on a finite dominating set.

4.3.1 An Integer Linear Formulation Based on Sorting

The first integer formulation of the p-k-max problem is based on the formulations of the discrete ordered median problem in Domínguez-Marín (2003). The problem there is discrete, i.e., the location of the new facilities is restricted to the nodes of the underlying graph. Thus, a candidate set of possible optimal locations is known in advance. As for the p-k-max problem the location of the new facilities is not only allowed in the nodes of G but also in every point on the edges of the graph, a finite candidate set has to be known to apply this approach. Different finite dominating sets that can be used as such a candidate set are presented in Chapter 5 and Chapter 6 of this thesis. Hence, the aim of the following model is to evaluate a candidate set (that is assumed to be given) to find the optimal solutions therein.

The discrete ordered median problem on graphs can be formulated as a combination of two subproblems: The first subproblem is an integer linear problem that corresponds to the sorting of the vector of weighted distances. The second subproblem is an integer linear program to find the locations of the new facilities and the corresponding allocation of the customers to the new facilities. Domínguez-Marín (2003) (based on Nickel, 2001) starts with a quadratic formulation and gives also several linearisations of the problem. Due to the close connection of a p-k-max problem to an ordered median problem, most of these formulations can be transferred with just some small adjustments. The first linearisation is applied to the p-k-max in the following as it gives a good insight into the structure of the problem. Let C be a finite set of possible locations for the individual new facilities and let $|C| = q$ be the number of candidate locations in the set C. The weighted distance between a node v_i, $i = 1, \ldots, n$, and a candidate location $x_j \in C$, $j = 1, \ldots, q$, is assumed to be known in advance and is denoted by d_{ij}. Moreover, define two binary variables as

$$y_j = \begin{cases} 1, & \text{if a new facility is located at site } x_j \in C \\ 0, & \text{otherwise} \end{cases}$$

for all $j = 1, \ldots, q$ and

$$x_{ij}^a = \begin{cases} 1, & \text{if customer } v_i \text{ is allocated to site } x_j \in C \text{ and this allocation} \\ & \text{corresponds to the } a\text{th largest weighted distance} \\ 0, & \text{otherwise.} \end{cases}$$

for all $i = 1, \ldots, n$, $j = 1, \ldots, q$ and $a = 1, \ldots, n$.

The integer linear programming formulation **(IP1)** of the (discrete) p-k-max problem can then be given as:

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{q} d_{ij}x_{ij}^{k} \tag{4.2}$$

$$\text{s.t.} \quad \sum_{a=1}^{n}\sum_{j=1}^{q} x_{ij}^{a} = 1 \qquad \forall\ i = 1,\ldots,n \tag{4.3}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{q} x_{ij}^{a} = 1 \qquad \forall\ a = 1,\ldots,n \tag{4.4}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{q} d_{ij}x_{ij}^{a} \geq \sum_{i=1}^{n}\sum_{j=1}^{q} d_{ij}x_{ij}^{a+1} \qquad \forall\ a = 1,\ldots,n-1 \tag{4.5}$$

$$\sum_{j=1}^{q} y_j = p \tag{4.6}$$

$$\sum_{a=1}^{n} x_{ij}^{a} \leq y_j \qquad \forall\ i = 1,\ldots,n,\ \forall\ j = 1,\ldots,q \tag{4.7}$$

$$x_{ij}^{a} \in \{0,1\},\ y_j \in \{0,1\} \qquad \forall\ a,i = 1,\ldots,n,\ \forall\ j = 1,\ldots,q \tag{4.8}$$

For a fixed allocation of the customers to a fixed set of new facilities, the variable x_{ij}^{k} is equal to one for only one combination of i, j, k and zero otherwise. Therefore, the sum in the objective function (4.2) has only one non-zero term and hence equals the minimisation of the kth largest weighted distance between the customers v_i, $i = 1,\ldots,n$, and the candidate locations x_j, $j = 1,\ldots,q$. The constraints (4.3) to (4.5) determine the permutation of the vector of weighted distances. The sorting process is modelled by an assignment problem with an additional constraint to ensure the correct sorting. The constraint (4.3) ensures that each weighted distance is placed at only one position of the sorted distance vector. Similarly, the constraint (4.4) ensures that each position of the sorted distance vector contains only one weighted distance. Moreover, these two constraints guarantee that each customer is assigned to exactly one new facility. Constraint (4.5) guarantees the non-increasing order of the sorted weighted distances. The location of the new facilities is modelled based on a linearisation of a classical location-allocation problem as proposed, for example, by Labbé et al. (1995). Constraint (4.6) sets the number of new facilities to p. A customer can only be covered with service from a candidate site, if a new facility is built there. Thus, constraint (4.7) ensures the allocation of a customer to a open new facility.

The model (IP1) has $\mathcal{O}(qn^2)$ variables and $\mathcal{O}(qn)$ constraints. For alternative formulations of the discrete ordered median problem see Domínguez-Marín (2003).

4.3.2 A Mixed Integer Formulation Based on Identifying Outliers

In this section two further formulations of the p-k-max problem are presented. The first model is an integer programming formulation which requires a finite dominating set of the

problem to be known. While the model (IP1) is based on the sorting process to determine the correct permutation of the vector of weighted distances, the following formulation does not need to sort the elements of the distance vector explicitly. The idea is to model the identification of the outliers and to skip the corresponding distances to these nodes in the objective function such that the $k-1$ largest distances are not taken into account without the need to find them by sorting. This reduces the number of variables of the model.

Again, the formulation is based on a finite candidate set which has to be derived in a preprocessing step. The finite dominating sets presented in Chapter 5 and Chapter 6 can be used here. The notation is equivalent to that of formulation (IP1), i.e., C is a finite set of possible locations for the individual new facilities with $|C| = q$. The weighted distance $d^w(v_i, x_j)$ for $i = 1, \ldots, n$ and $x_j \in C$, $j = 1, \ldots, q$, is assumed to be known in advance and is denoted by d_{ij}. Moreover, three binary variables are defined as

$$y_j = \begin{cases} 1, & \text{if a new facility is located at site } x_j \in C \\ 0, & \text{otherwise} \end{cases}$$

for all $j = 1, \ldots, q$,

$$x_{ij} = \begin{cases} 1, & \text{if customer } v_i \text{ is allocated to site } x_j \in C \\ 0, & \text{otherwise} \end{cases}$$

for all $i = 1, \ldots, n$ and $j = 1, \ldots, q$, and

$$\alpha_i = \begin{cases} 1, & \text{if node } v_i \text{ is an outlier} \\ 0, & \text{otherwise} \end{cases}$$

for all $i = 1, \ldots, n$. The integer programming formulation **(IP2)** of the p-k-max problem can then be given as:

$$\min \quad \max_{i=1,\ldots,n} \sum_{j=1}^{q} x_{ij} d_{ij} \tag{4.9}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i = k - 1 \tag{4.10}$$

$$\sum_{j=1}^{q} x_{ij} = 1 - \alpha_i \qquad \forall\ i = 1, \ldots, n \tag{4.11}$$

$$\sum_{j=1}^{q} y_j = p \tag{4.12}$$

$$x_{ij} \leq y_j \qquad \forall\ i = 1, \ldots, n,\ \forall\ j = 1, \ldots, q \tag{4.13}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\ i = 1, \ldots, n,\ \forall\ j = 1 \ldots, q \tag{4.14}$$

$$y_j \in \{0, 1\},\ \alpha_i \in \{0, 1\} \qquad \forall\ i = 1, \ldots, n \tag{4.15}$$

The first constraint (4.10) guarantees that an optimal solution has exactly $k-1$ outliers. The following constraint (4.11) ensures a link between a node being an outlier and its allocation: A customer is only allocated to a new facility, if this customer is not an outlier, i.e., outliers are not served by any of the new facilities. Non-outliers are guaranteed to be allocated to exactly one new facility. With (4.12), the number of new facilities is fixed to p. The constraint (4.13) ensures that non-outliers are only allocated to opened new facilities.

The binary variable x_{ij} is zero for all $j = 1, \ldots, q$ if node v_i is an outlier. Thus, for a fixed non-outlier v_i, $i \in \{1, \ldots, n\}$, the sum $\sum_{j=1}^{q} x_{ij} d_{ij}$ equals the weighted distance between this customer and its corresponding new facility x_j whereas this sum is zero for outlier nodes. Hence, the objective function (4.9) minimises the maximum of the distances between the center defining nodes and their corresponding new facilities. As it is assured that the $k-1$ distances to the outliers are not taken into account, no sorting of the distance vector is needed. Note that the objective function is piecewise linear.

This model consists of $\mathcal{O}(nq)$ variables and $\mathcal{O}(nq)$ constraints which makes it much smaller w.r.t. the number of variables than the model (IP1). A quite similar model is introduced in Daskin and Owen (1999).

A drawback of the above presented models (IP1) and (IP2) is that they are based on a finite dominating set for the underlying p-k-max problem. Of course, its calculation causes additional effort and thus it would be desirable to overcome this preprocessing step. It is possible to adjust the model (IP2) such that it does not only evaluate predefined candidates but that it yields optimal new facilities located somewhere on the edges or in the nodes of the graph. For this purpose, the binary variables α_i with $i = 1, \ldots, n$ are defined as above. The variable x_{ij} is redefined to

$$x_{ij} = \begin{cases} 1, & \text{if customer } v_i \text{ is allocated to site } x_j \in A(G) \\ 0, & \text{otherwise} \end{cases}$$

for all $i = 1, \ldots, n$ and $j = 1, \ldots, p$. The mixed-integer programming formulation **(IP3)** of the p-k-max problem can then be given as:

$$\min \quad \max_{i=1,\ldots,n} \sum_{j=1}^{p} x_{ij} d^w(v_i, x_j) \tag{4.16}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i = k - 1$$

$$\sum_{j=1}^{p} x_{ij} = 1 - \alpha_i \qquad \forall\ i = 1, \ldots, n \tag{4.17}$$

$$x_j \in A(G) \qquad \forall\ j = 1, \ldots, p \tag{4.18}$$

$$x_{ij} \in \{0, 1\} \qquad \forall\ i = 1, \ldots, n,\ \forall\ j = 1, \ldots, p \tag{4.19}$$

$$\alpha_i \in \{0, 1\} \qquad \forall\ i = 1, \ldots, n$$

The main difference between the formulation (IP3) and the model (IP2) is that the binary variables y_j, $j = 1, \dots, q$, are substituted by continuous variables $x_j \in A(G)$, $j = 1, \dots, p$, see constraint (4.18). Thus, x_j specifies the location of a new facility on $A(G)$ and not a decision to open a facility in a candidate or not. The number p of new facilities is fixed as the index j goes from 1 to p in the constraints (4.17), (4.18) and (4.19) and thus an extra constraint to fix this number is not needed. The constraint that ensures the correct number of outliers is unchanged whereas the constraint that allocates customers only to opened facilities is not needed here.

Note that the objective function is non linear now since the distances $d^w(v_i, x_j)$ can in general not be computed in a preprocessing step because x_j is not known. The model (IP3) consists of $\mathcal{O}(np)$ variables and $\mathcal{O}(np)$ constraints, which is, due to $p \ll q$, significantly smaller than the previous two formulations. However, it is not possible to solve the problem with this formulations directly, some adjustments have to be included to evaluate the distances in the objective function (4.16).

Remark 4.15. *The formulation (IP3) can equivalently be applied to model a continuous* p-k-max *problem. By replacing constraint* (4.18)*, for example, by the constraint* $x_j \in \mathbb{R}^2$ *for* $j = 1, \dots, p$*, a* p-k-max *problem in the plane can be modelled.*

4.4 Equilibrium- and Bottleneck Points

An often used approach to solve ordered median problems on graphs is to derive a finite dominating set and to develop an algorithm based on this set. As p-k-max problems are a special case of ordered p-median problems for a specific vector of weights λ, some of these concepts can (at least partly) be transferred to p-k-max problems. The so called equilibrium points of a graph play an important role in the theory of ordered median functions and will be essential for the p-k-max analysis as well. In this section, the calculation of these points is described. They are needed later for nearly all approaches to solve the p-k-max problems. Therefore, an algorithm to compute the equilibrium points efficiently is presented in detail. A formal definition following Nickel and Puerto (2005) is given below.

Definition 4.16 ▶ Equilibrium points

For all pairs of nodes $v_i, v_j \in V$, $i \neq j$, define the set

$$EQ'_{ij} = \left\{ x \in A(G) : w_i d(v_i, x) = w_j d(v_j, x) \right\}.$$

Moreover, let EQ_{ij} be the relative boundary of EQ'_{ij}, i.e. the set of endpoints of the closed subedges contained in EQ'_{ij}. Then,

$$EQ := \bigcup_{\substack{i,j \\ i \neq j}} EQ_{ij}$$

is the set of *equilibrium points* of the graph G.

According to this definition, equilibrium points have the same weighted distance to at least two existing facilities. EQ'_{ij} can have infinitely many elements. In this case EQ'_{ij} is not equal to the finite set EQ_{ij}. If EQ'_{ij} is finite, then both sets are equal.

Whenever the notation of the indices i, j is not sufficient to characterise the corresponding nodes precisely, the set of equilibrium points will instead be written as EQ_{v_i,v_j}. Sometimes it is important to note on which edge an equilibrium point is located. In this case the notation EQ^{ab}_{ij} for the set of equilibrium points of the nodes v_i, v_j on the edge e_{ab} is used. Note that $EQ^{ab}_{ij} = EQ^{ab}_{cd}$ is possible for i, j not equal to c, d, i.e., one equilibrium point may be generated by more than one pair of nodes. Thus, the elements in EQ do not have to be unique. Moreover, also individual equilibrium points $y \in EQ_{ij}$ are sometimes denoted by y_{ij} to emphasise the nodes v_i and v_j defining this particular equilibrium point.

Definition 4.17 ▶ Consecutive equilibrium points

Two points $x', x'' \in EQ$ are called *consecutive*, if there is no other $\tilde{x} \in EQ \setminus \{x', x''\}$ on any shortest path between x' and x''.

Examples for the sets EQ and EQ' are shown in Figure 4.3. All weights of nodes of the considered graph are equal to one. The set of equilibrium points of the graph G is $EQ = \{v_a, v_b, x_1, x_2, x_3, x_4\}$, where, for example, x_1 is an equilibrium point of the nodes v_i and v_a. The set EQ'_{ij} has infinitely many elements as it equals the complete edge $e_{ab} \in E$. Thus, only its endpoints v_a and v_b are elements of the set EQ_{ij}. The two equilibrium points $x_3 \in e_{ab}$ and $x_4 \in e_{ab}$ are consecutive.

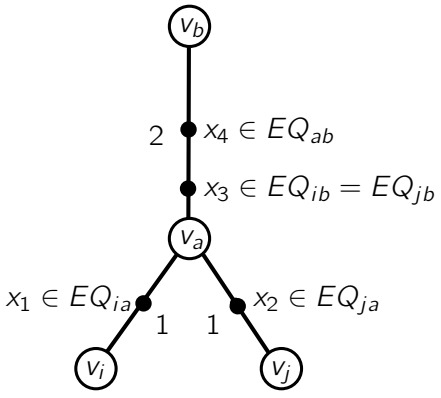


Fig. 4.3: Example problem where EQ'_{ij} is not equal to EQ_{ij} since $EQ_{ij} = e_{ab}$ has infinitely many elements

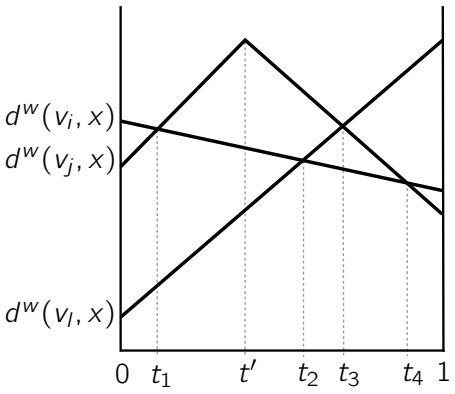


Fig. 4.4: Examples of distance functions over an edge e_{ab} with $x_i = (e_{ab}, t_i) \in EQ$ for $i = 1, \ldots, 4$

The equilibrium points of G can be determined by computing the intersection points of the graphs of the distance functions $d^w(v_i, x)$ over an edge $e_{ab} \in E$, i.e., for $x = (e_{ab}, t)$ with $t \in [0, 1]$ and for all nodes $v_i \in V$ and for all $a, b = 1, \ldots, n$. These intersections yield all equilibrium points on e_{ab} because in these points the distance of x to v_i equals the distance of x to v_j for some $v_i, v_j \in V$, $i \neq j$.

The distance function $d^w(v_i, x)$, $i \in \{1, \dots, n\}$, on e_{ab} is a piecewise linear function with at most two linear segments. The case of two linear segments arises if the maximum of $d^w(v_i, x)$ on e_{ab} is attained in the interior of e_{ab}. The breakpoint is then attained in a so called bottleneck point. The definition below is taken from Nickel and Puerto (2005).

Definition 4.18 ▶ Bottleneck point

A point $x = (e_{ab}, t)$ on an edge $e_{ab} \in E$ is called a *bottleneck point* of a node $v_i \in V$, if

$$d^w(v_i, x) = w_i(d(x, v_a) + d(v_a, v_i)) = w_i(d(x, v_b) + d(v_b, v_i)).$$

BN_i denotes the set of all bottleneck points of a node $v_i \in V$ and

$$BN := \bigcup_{i=1}^{n} BN_i$$

is the set of all bottleneck points of G.

Analogous to the notation used for equilibrium points, BN_i^{ab} denotes the set of all bottleneck points of node v_i on an edge $e_{ab} \in E$. In a bottleneck point $x = (e_{ab}, t)$ it does not matter whether a path from a node $v_i \in V$, $i \in \{1, \dots, n\} \setminus \{a, b\}$, to x contains the node v_a or v_b because both alternatives lead to the same overall distance. An example for a bottleneck point at $x = (e_{ab}, t')$ can be seen in Figure 4.4. Note that due to the assumption of $w_i > 0$, the weighted distance functions are concave over every edge (see Figure 4.4).

Remark 4.19. *For every edge e_{ab} and every pair of nodes $v_i, v_j \in V$ the inequality*

$$|EQ_{ij} \cap e_{ab}| \leq 2$$

holds. In particular, if

$$EQ_{ij} \cap e_{ab} = EQ'_{ij} \cap e_{ab}$$

for all v_i, v_j with $v_i \neq v_j$ on e_{ab}, then the graphs of every pair of weighted distance functions $d^w(v_i, x)$ and $d^w(v_j, x)$ can have at most two intersection points over the edge e_{ab}. This holds due to the fact that if, for example, the graph of the function $d^w(v_i, x)$ consists of two linear segments, these segments have slopes w_i and $-w_i$. Thus, the graphs of the functions $d^w(v_i, x)$ and $d^w(v_j, x)$ can intersect in at most two points. If

$$EQ_{ij} \cap e_{ab} \neq EQ'_{ij} \cap e_{ab}$$

then only the relative boundary of the unique intersection segment is considered in the set EQ_{ij}. The relative boundary consists of the two endpoints of this segment. As a consequence, the number of equilibrium points on G is bounded by $\mathcal{O}(mn^2)$.

In the following a sweepline technique following Bentley and Ottmann (1979) based on Shamos and Hoey (1976) is presented which computes the intersection points of an arrangement of α line segments in $\mathbb{R}^2$. This approach can later be applied to the arrangement

defined by the graphs of the weighted distance functions over a fixed edge to compute the equilibrium points of G. Let every line segment of the arrangement be denoted by S_i, $i = 1, \ldots, \alpha$. Moreover, every line segment S_i has a left and a right endpoint, denoted by $e_1(S_i)$ resp. $e_2(S_i)$ for all $i = 1, \ldots, \alpha$. As the arrangement is contained in $\mathbb{R}^2$, every point has two coordinates, an x-component and a y-component.

The basic idea of the algorithm is to sweep a vertical line ℓ through the x-components of the endpoints of the α given line segments of the arrangement. For every fixed position x of the sweepline ℓ, a total order of the line segments is defined by the y-components of the positions in which the segments intersect ℓ. For two line segments S_i and S_j, this ordering relation is denoted by $S_i \succeq S_j$ if the line segment S_i lies over the segment S_j at position x. With this ordering relation, intersection points can be detected easily: If $S_i \succeq S_j$ at position x_1 but $S_j \succeq S_i$ at position x_2 for $x_1 \leq x_2$, then the segments must intersect between x_1 and x_2.

The main loop of the algorithm performs a sweep of the vertical line ℓ from left to right through the set of segments. Whenever the position of ℓ equals the x-component of an endpoint $e_j(S)$, $j \in \{1, 2\}$, of a segment S, an event point is defined and the sweeping stops. In every event point the current ordering of segments is stored in a balanced tree R. If $e_j(S)$ is the left endpoint of S, the segment is inserted at the correct position into R. If $e_j(S)$ is the right endpoint, then it is deleted from R. After inserting S into R, S and its upper neighbour $u(S)$ with respect to the current ordering are checked for an intersection. The same is done for S and its current lower neighbour $l(S)$ because if two lines intersect, they have to become neighbours in R for some x to the left of the intersection point. With the same argument, $u(S)$ and $l(S)$ are explored for intersections when S is deleted from R. Whenever an intersection of segments S_i and S_j is detected, the intersection point itself also becomes an event point, denoted by i_{S_i,S_j}. If ℓ stops in i_{S_i,S_j}, S_i and S_j have to be swapped in R because in an intersection point the order with respect to ℓ changes from $S_i \succeq S_j$ to $S_j \succeq S_i$ (or vice versa). Therefore, it has to be assumed that not more than two lines meet in one point. After the swap, S_j and $u(S_j)$ as well as S_i and $l(S_i)$ are checked for intersections if $i_{S_i,l(S_i)}$ resp. $i_{S_j,u(S_j)} \notin R$.

The number of event points is $2\alpha + s$, assuming that $s \leq \binom{\alpha}{2}$ is the number of intersection points. Thus, the main loop of the algorithm has to be executed exactly $2\alpha + s$ times. Every operation on R can be performed in $\mathcal{O}(\log(\alpha))$ since it is implemented with a balanced tree. The set of event points should be implemented in form of a heap, so that every operation can be realised in $\mathcal{O}(\log(\alpha))$. Consequently, the total running time of the algorithm is bounded by $\mathcal{O}((\alpha + s)\log(\alpha))$. A slightly modified formulation of this algorithm in pseudocode can be found in Section 5.

In order to compute the set of equilibrium points EQ, some adjustments of the algorithm of Bentley and Ottmann (1979) have to be considered. For example, it is necessary to consider segments with endpoints that share the same x-coordinate as well as segments that have more than one intersection point (cases that were excluded in Bentley and Ottmann, 1979). This can be realised without increasing the overall complexity of the algorithm. For some $a, b \in \{1, \ldots, n\}$ with $a \neq b$ let L_{ab} be the line arrangement over an edge $e_{ab} \in E$ defined

by the graphs of the weighted distance functions $d^w(v_i, x)$ for $x = (e_{ab}, t)$ and $v_i \in V$, $i = 1, \ldots, n$. Recall that there is a one to one correspondence between a point $x = (e_{ab}, t)$ and the parameter $t \in [0, 1]$. Since the arrangement of lines is considered only over a fixed edge $e_{ab} \in E$, a point $x = (e_{ab}, t)$ can simply be identified by the parameter t.

One difference between L_{ab} and a classical line arrangement as used for the algorithm of Bentley and Ottmann (1979) is that a distance function $d^w(v_i, x)$ may consist of two linear parts such that the corresponding S_i is not a line segment. In this case, every linear part of the graph of $d^w(v_i, x)$ has to be treated as a single line segment. In particular, it is defined that

$$d^w(v_i, x) = d^w(v_i, (e_{ab}, t)) = \begin{cases} w_i(d(v_i, v_a) + t l_{ab}) := S_i^\alpha(t) & \text{for } t \in [0, t_b] \\ w_i(d(v_i, v_b) + (1-t) l_{ab}) := S_i^\beta(t) & \text{for } t \in [t_b, 1], \end{cases}$$

where $x_b = (e_{ab}, t_b)$ is the bottleneck point of $d^w(v_i, x)$ over e_{ab}. The two linear segments of the graph of $d^w(v_i, x)$ are then denoted by

$$S_i = \begin{cases} \{(t, S_i^\alpha(t)) : t \in [0, t_b]\} := S_i^\alpha & \text{on } [0, t_b] \\ \{(t, S_i^\beta(t)) : t \in [t_b, 1]\} := S_i^\beta & \text{on } [t_b, 1]. \end{cases} \tag{4.20}$$

To simplify notation, all segments are referred to as S_i in the following, even in the case that they consist of two new sub-segments. As bottleneck point never are relevant event points by themselves, no equilibrium points are lost in this way. All line segments of L_{ab} start in v_a ($t = 0$), or end in v_b ($t = 1$). Hence, to get an order of insertion and deletion, the endpoints of the segments are sorted lexicographically by their $d^w(v_i, x)$-value w.r.t. $t = 0$ in v_a and w.r.t. $t = 1$ in v_b. Thus, the endpoints of the segments are processed from the uppermost to the lowermost in the endpoints of the edge.

If two line segments intersect in infinitely many points, then just the boundary of the intersection segment (which equals EQ') is needed. In this case, it is possible to obtain two intersection points for two segments and it follows for the number s of intersection points over e_{ab} that $s \leq 2\binom{n}{2}$. Since all equilibrium points on a particular edge of G can be generated in $\mathcal{O}((n+s)\log(n))$ time with $s \leq 2\binom{n}{2}$, the computational effort for the whole network is bounded by $\mathcal{O}(m(n+s)\log(n))$. Note that for $s \gtrsim \binom{n}{2}$ the sweepline algorithm has a higher complexity than checking all $\binom{n}{2}$ pairs of distance functions on one edge for intersections, what takes $\mathcal{O}(n^2)$. Thus, the practical efficiency is problem dependent and related to the structure of the underlying network.

The overall procedure to compute the equilibrium points of one edge of a graph is summarised in Algorithm 3. Note that the algorithm can be implemented easily in parallel as the equilibrium points of one edge are independent from the equilibrium points of the other edges. To simplify the notation of the algorithm it is assumed that a point in the arrangement (e.g., the endpoints of the line segments or the intersection points of two segments) over edge e_{ab} has the two components $t \in [0, 1]$ and $y = d^w(v, (e_{ab}, t))$ for some node $v \in V$.

Algorithm 3 Equilibrium points of graph G (based on Bentley and Ottmann, 1979)

Input: Edge $e_{ab} \in E$ of G; heap Q with root $r(Q)$: set of endpoints $e(S) = (t, y)$, sorted lexicographically by t and y-value

while $Q \neq \emptyset$ **do**
  Set binary tree $R := \emptyset$ and $EQ := \emptyset$
  Set $x := r(Q)$
  if x is left endpoint of segment S_i **then** ▷ Include S_i into the ordering
    INSERTION(x, R, Q)
  if x is right endpoint of segment S_i **then** ▷ Delete S_i from the ordering
    DELETION(x, R, Q)
  if x is intersection point of $S_i \succeq S_j$ **then** ▷ Swap ordering to $S_j \succeq S_i$
    NEWEQUILIBRIUM(x, R, Q, EQ)

Output: Set EQ of equilibrium points of G.

procedure INSERTION(x, R, Q)
  if S_i intersects $u(S_i)$ **then** ▷ Check upper neighbour
    if $i_{S_i,u(S_i)} = S_i$ **then** ▷ Infinitely many intersection points
      Insert $e_1(S_i), e_2(S_i)$ into Q
      Insert i into R
    else if $i_{S_i,u(S_i)} = u(S_i)$ **then** ▷ Infinitely many intersection points
      Insert $e_1(u(S_i)), e_2(u(S_i))$ into Q
      Insert i into R
    else ▷ Unique intersection point
      Insert $i_{S_i,u(S_i)}$ into Q
      Insert i into R
  if S_i intersects $l(S_i)$ **then** ▷ Check lower neighbour
    if $i_{S_i,l(S_i)} = S_i$ **then** ▷ Infinitely many intersection points
      Insert $e_1(S_i), e_2(S_i)$ into Q
      Insert i into R
    else if $i_{S_i,l(S_i)} = l(S_i)$ **then** ▷ Infinitely many intersection points
      Insert $e_1(l(S_i)), e_2(l(S_i))$ into Q
      Insert i into R
    else ▷ Unique intersection point
      Insert $i_{S_i,l(S_i)}$ into Q
      Insert i into R
  return R, Q
end procedure

1: **procedure** DELETION(x, R, Q)
2: **if** $i_{u(S_i),l(S_i)} \notin Q \wedge u(S_i)$ intersects $l(S_i)$ **then** ▷ Check upper and lower neighbour
3: Insert $i_{u(S_i),l(S_i)}$ into Q
4: Delete i from R
5: **return** R, Q
6: **end procedure**

1: **procedure** NEWEQUILIBRIUM(x, R, Q, EQ)
2: $EQ = EQ \cup \{x\}$
3: Swap positions of i and j in R such that $S_j \succeq S_i$ ▷ Change ordering
4: **if** S_j intersects $u(S_j) \wedge i_{S_j,u(S_j)} \notin Q$ **then** ▷ Check upper neighbour
5: Insert $i_{S_j,u(S_j)}$ into Q
6: **if** S_i intersects $l(S_i) \wedge i_{S_i,l(S_i)} \notin Q$ **then** ▷ Check lower neighbour
7: Insert $i_{S_i,l(S_i)}$ into Q
8: **return** R, Q, EQ
9: **end procedure**

Example 4.20. *The sets of equilibrium points $EQ^{12}_{i,j}$ for $i, j = 1, \ldots, 5$ with $i \neq j$ of the edge e_{12} of a Graph G (see Figure 4.5) have to be computed. Figure 4.6 shows the line arrangement of the graphs of the distance functions $d^w(V, x)$ for $x \in e_{12}$. Algorithm 3 determines the equilibrium points $i_{43} = EQ^{12}_{34}$, $i_{52} = EQ^{12}_{25}$, $i_{21} = EQ^{12}_{12}$, $i_{15} = EQ^{12}_{15}$, $i_{14} = EQ^{12}_{14}$, $i_{13} = EQ^{12}_{13}$ and $i_{45} = EQ^{12}_{45}$ as intersection points of the lines. Applying the algorithm to all edges of G, the whole set*

$$
\begin{aligned}
EQ = \Big\{ & EQ^{12}_{34} = \left(e_{12}, 0\right), \quad EQ^{12}_{25} = \left(e_{12}, \frac{1}{4}\right), \quad EQ^{12}_{12} = \left(e_{12}, \frac{1}{3}\right), \quad EQ^{12}_{15} = \left(e_{12}, \frac{1}{2}\right), \\
& EQ^{12}_{14} = \left(e_{12}, \frac{3}{4}\right), \quad EQ^{12}_{13} = \left(e_{12}, \frac{9}{10}\right), \quad EQ^{12}_{45} = (e_{12}, 1), \quad EQ^{13}_{34} = (e_{13}, 0), \\
& EQ^{13}_{12} = \left(e_{13}, \frac{1}{2}\right), \quad EQ^{13}_{15} = \left(e_{13}, \frac{1}{2}\right), \quad EQ^{13}_{25} = \left(e_{13}, \frac{1}{2}\right), \quad EQ^{13}_{13} = \left(e_{13}, \frac{3}{5}\right), \\
& EQ^{13}_{14} = \left(e_{13}, \frac{3}{4}\right), \quad EQ^{13}_{35} = \left(e_{13}, \frac{3}{4}\right), \quad EQ^{13}_{23} = \left(e_{13}, \frac{3}{4}\right), \quad EQ^{13}_{25} = (e_{13}, 1), \\
& EQ^{23}_{45} = (e_{23}, 0), \quad EQ^{23}_{34} = \left(e_{23}, \frac{1}{5}\right), \quad EQ^{23}_{35} = \left(e_{23}, \frac{1}{2}\right), \quad EQ^{23}_{23} = \left(e_{23}, \frac{3}{4}\right), \\
& EQ^{24}_{45} = (e_{24}, 0), \quad EQ^{23}_{25} = (e_{23}, 1), \quad EQ^{24}_{24} = \left(e_{24}, \frac{2}{3}\right), \quad EQ^{34}_{25} = (e_{34}, 0), \\
& EQ^{34}_{24} = \left(e_{34}, \frac{1}{3}\right), \quad EQ^{34}_{45} = \left(e_{34}, \frac{1}{3}\right), \quad EQ^{34}_{34} = \left(e_{34}, \frac{2}{5}\right), \quad EQ^{34}_{23} = \left(e_{34}, \frac{1}{2}\right),
\end{aligned}
$$

$$EQ_{25}^{34} = \left(e_{34}, \frac{1}{2}\right), \quad EQ_{35}^{34} = \left(e_{34}, \frac{1}{2}\right), \quad EQ_{34}^{15} = (e_{15}, 0), \quad EQ_{15}^{15} = \left(e_{15}, \frac{1}{3}\right),$$
$$EQ_{12}^{15} = (e_{15}, 1), \quad EQ_{25}^{35} = (e_{35}, 0), \quad EQ_{13}^{35} = \left(e_{35}, \frac{1}{5}\right), \quad EQ_{14}^{35} = \left(e_{35}, \frac{1}{2}\right),$$
$$EQ_{23}^{35} = \left(e_{35}, \frac{1}{2}\right), \quad EQ_{35}^{35} = \left(e_{35}, \frac{3}{4}\right), \quad EQ_{12}^{35} = (e_{35}, 1) \Bigg\}$$

is determined and it holds that $|\mathrm{Cand}| = 39$. *All equilibrium points of* G *are shown in Figure 4.5. Note that one equilibrium point may correspond to several different sets of defining nodes, for example,* $EQ_{14}^{13} = EQ_{35}^{13} = EQ_{23}^{13} = (e_{13}, \frac{3}{4})$. *Especially equilibrium points lying in a node are found for every edge incident to that node. Filtering out these redundant points leads to a number of* 23 *equilibrium points. However, it will be seen later that in some solution approaches the redundant points are needed for the correct evaluation of the objective function.*

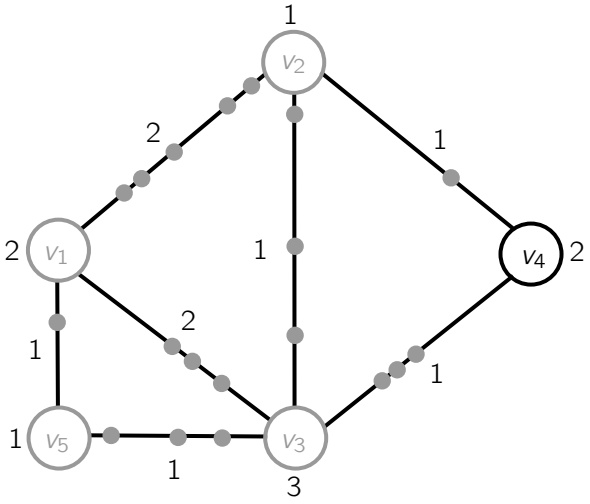


Fig. 4.5: Graph G with all of its equilibrium points (gray)

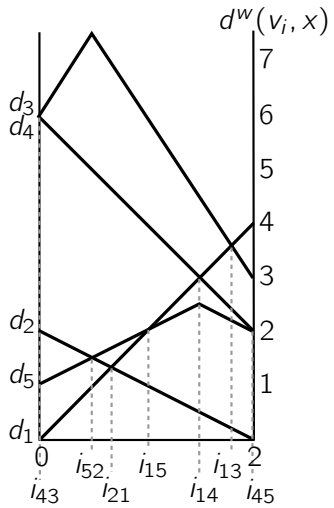


Fig. 4.6: Line arrangement with $d^w(v_i, x)$ for $v_i \in V$, $i = 1, \ldots, 5$, $x = (e_{12}, t)$.

Remark 4.21. *Tree graphs with node weights* $w_i = 1$ *for all* $i = 1, \ldots, n$ *have exactly* $n(n-1)/2$ *equilibrium points since there is a unique path connecting each pair of nodes* $v_s, v_t \in V$ *with* $v_s \neq v_t$ *whose midpoint is the unique equilibrium point of* v_s *and* v_t. *In the weighted case with* $w_i > 0$, $w_i \in \mathbb{R}$ *for all* $i = 1, \ldots, n$, *a tree graph has in general much more than* $n(n-1)/2$ *equilibrium points as each pair of nodes can have equilibrium points on more than one edge. This follows with the previous observation that the graph of a weighted distance function* $d^w(v_i, x)$ *with* $x = (e_{ab})$ *and* $t \in [0, 1]$ *over the edge* e_{ab} *has either slope* w_i *or slope* $-w_i$. *Consequently, if all nodes are equally weighted, two distance functions* $d^w(v_i, x)$ *and* $d^w(v_j, x)$ *can intersect only if they have different slopes, i.e., if the corresponding nodes* v_i *and* v_j *belong to different connected components if the edge* e_{ab} *is deleted from the tree. If the nodes have different weights, also nodes from the same*

connected component may yield an equilibrium point on the edge e_{ab}. These cases are illustrated in Figure 4.7.

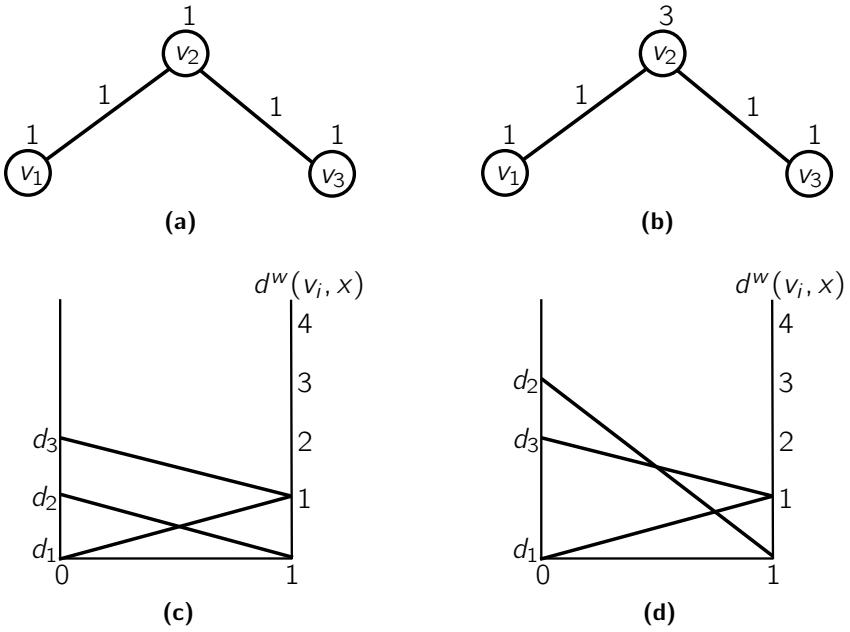


Fig. 4.7: Subfigure (c) resp. (d) illustrate the graphs of the weighted distance functions over edge e_{12} of the unweighted (Subfigure (a)) resp. the weighted (Subfigure (b)) graph. In the weighted case, the functions $d^w(v_2, x)$ and $d^w(v_3, x)$ intersect which is not possible in the unweighted case since these functions are parallel there.

Throughout this thesis there will be the need to efficiently evaluate the k-max function in candidate points. An algorithm for this purpose and its complexity are presented below.

Algorithm 4 Evaluating the p-k-max function in a candidate point

---

Input: Graph $G = (V, E)$, parameters $k \in \{1, \ldots, n-p\}$ and $p \in \{1, \ldots, n\}$, candidate solution $X = \{x_1, \ldots, x_p\}$ with $x_j = (e_{a_j b_j}, t_j)$, $j = 1, \ldots, p$.

1: **for all** $v_i \in V$ **do**

2: **for all** $j = 1, \ldots, p$ **do**

3: $z^j := \min\{w_i(d(v_{a_j}, x_j) + d(v_{a_j}, v_i)), w_i(d(v_{b_j}, x_j) + d(v_{b_j}, v_i))\}$.

4: $z_i := \min\{z_j\}$

5: Sort the vector $z = (z_1, \ldots, z_n)^\top$ with permutation $\sigma \in \Sigma(x)$ in non-increasing order.

6: Set $z^* := z_{\sigma(k)}$.

Output: Objective function value $z^* = k\text{-max}(d(V, X))$.

It is assumed that the all-pair shortest distances in G are computed in advance, for example using the algorithm of Floyd (1962) which has a complexity of $\mathcal{O}(n^3)$. Then, each distance to a fixed node can be found in constant time using the predefined shortest paths between all pairs of nodes. In order to evaluate the objective function in one candidate solution X, the vector of distances $d^w(v_i, X)$ to all existing facilities v_i is needed. As the ith element of the distance vector $d^w(V, X)$ is given by the distance between a node v_i and its nearest new facility, the distances of v_i to all p new facilities have to be computed in $\mathcal{O}(p)$ time, resulting in $\mathcal{O}(np)$ for all nodes. Afterwards, the shortest distances are sorted in non-increasing order in $\mathcal{O}(n\log(n))$. Taking the kth component needs $\mathcal{O}(1)$. This results in $\mathcal{O}(n(p+\log(n)))$ for the evaluation of one candidate point. As p is assumed to be arbitrary but fixed, it holds that $\mathcal{O}(n(p+\log(n))) = \mathcal{O}(n\log(n))$.

Chapter 5

Algorithms for 1-k-max Location Problems on Networks

The aim of this chapter is to analyse the 1-k-max problem, which is the special case of (pkMG) where just one new facility has to be located and therefore it holds that $p = 1$. This problem is analysed separately because it has a simpler structure than the multi-facility problems and solution approaches can be designed much more efficiently. The first section of this chapter investigates special properties of outliers such that the set of existing facilities may be reduced before starting a solution approach as some of the nodes are known not to be covered with service in an optimal solution. In the following two sections, two algorithms for solving 1kMG are derived, and their respective complexities are analysed. They are both based on different finite dominating sets describing candidate points for optimal solutions of the problem. The first approach is based on equilibrium points that define a subdivision of G which is helpful for solving the problem. The second idea uses h-levels to construct an even smaller finite dominating set. The third section discusses two special cases of 1-k-max problems that are easier to solve than the general case.

For simplification, the 1-k-max is denoted by k-max without explicitly mentioning the value of $p = 1$. For $k \in \{1, \dots, n\}$ the problem is given by

$$\min_{x \in A(G)} k\text{-max}(d^w(V, x)) = \min_{x \in A(G)} w_{\sigma(k)} d(v_{\sigma(k)}, x), \tag{1kMG}$$

where $x \in A(G)$ is the new location, $d^w(V, x) = (d^w(v_1, x), \dots, d^w(v_n, x))^\top$ is the vector containing the weighted length of a shortest path between v_i and x in its ith component and $\sigma \in \Sigma(x)$ is a permutation of the existing facilities such that

$$d^w(v_{\sigma(1)}, x) \geq d^w(v_{\sigma(2)}, x) \geq \dots \geq d^w(v_{\sigma(n)}, x).$$

The set of optimal solutions of (1kMG) is denoted by $\mathcal{X}$. Other notations are used as introduced at the beginning of Chapter 4.

5.1 Properties of Outliers

To get more insight into the structure of k-max problems or outliers in center problems, respectively, some properties concerning outliers are stated in the following. The aim of this section is to derive criteria to decide whether a certain node of the graph has to be

an outlier or not to use these results, for example, in preprocessing procedures for general purpose algorithms. In particular, this may reduce the computational effort of the main solution algorithm significantly since there are less decisions to make. Some of these results are used later in combination with finite dominating sets to be able to further reduce the derived candidate sets.

For general weighted graphs it is hard to establish properties for nodes that have to be fulfilled to be an outlier because the weights have a large impact and can lead to very different distributions of the outliers (see Chapter 8). Therefore, this chapter is restricted to unweighted problems.

The following theorem shows an important result for general unweighted graphs, the connectedness of the center defining nodes.

Theorem 5.1 ▶ Connectedness of center defining nodes

Let $G = (V, E)$ be an unweighted graph and let $x \in A(G)$ be a feasible solution of the k-max problem with objective function value z, and an arbitrary but fixed set of outliers V_{k-1} corresponding to x.

Then, the induced subgraph $\bar{G}(\bar{V})$ with $\bar{V} = V \setminus V_{k-1}$ of G is connected.

Proof. The feasible solution x can w.l.o.g. be assumed to be a node of G. Otherwise, an artificial node can be inserted in x without changing the problem.

Assume that $\bar{G}$ is not connected, i.e., $\bar{G}$ consists of at least two connected components $\bar{G}_1, \bar{G}_2 \in G$. Let w.l.o.g. $x \in \bar{G}_1$ and choose an arbitrary but fixed node $v \in \bar{G}_2$. Therefore, there exists no path $\bar{P}(v, x)$ in $\bar{G}$.

However, there exists a shortest path $P(v, x)$ in G because G is connected. $P(v, x)$ must contain at least one outlier $v' \in V_{k-1}$ such that

$$P(v, x) = P(v, \ldots, v', \ldots, x)$$

with $d(v', x) \geq z$ because otherwise there would be a path between v and x in $\bar{G}$. With $v \notin V_{k-1}$ it holds that $d(v, x) \leq z$. With $l_j > 0$ for all $j = 1, \ldots, m$ this leads to a contradiction and therefore $\bar{G}$ has to be connected. ■

For the special case of path graphs (see Section 5.4), the connectedness of the center defining nodes has a useful implication: If $k > 2$, an outlier node, which is not a leaf node, has to have at least one neighbour that is also an outlier. It can not only be adjacent to center defining nodes. Note that the center defining nodes in a weighted network do not have to be connected, see Chapter 8 for an example.

If one outlier of a feasible solution x is already known, the next lemma uses, whenever possible, cuts in the graph to identify a whole subset of outliers based on the known one. A cut is a partition of the node set V into two subsets V' and $V'' = V \setminus V'$. The cut $S = (V', V'') = \{e_{ij} \in E : v_i \in V', v_j \in V''\}$ consists of all edges that have one endpoint in V' and the other endpoint in V'' (see, for example, Ahuja et al., 1993). The cardinality of a cut S in an unweighted graph is the number of edges in S.

Lemma 5.2 ▶ Outlier subset by cuts of size one

Let $G = (V, E)$ be an unweighted graph and let x be a feasible solution of the k-max problem on G with $2 \leq k \leq n-1$ fixed and a set of outliers V_{k-1}corresponding to x.

If there exists a cut $S = (V', V'') = \{e_{ij}\}$ for $i, j \in \{1, \ldots, n\}$ of size $|S| = 1$ in G with $v_i \in V'$ and $v_i \in V_{k-1}$, then it holds that

$$V' \subseteq V_{k-1} \quad \vee \quad V'' \cup \{v_i\} \subseteq V_{k-1}.$$

Proof. Assume that $V' \nsubseteq V_{k-1}$ and $V'' \nsubseteq V_{k-1}$. Let $S = (V', V'') = \{e_{ij}\}$ be a cut in G and let node $v_i \in V_{k-1}$ be incident to e_{ij}. Let $\bar{G}(\bar{V})$ with $\bar{V} = V \setminus V_{k-1}$ be the induced subgraph of G. Following Theorem 5.1, $\bar{G}$ has to be connected. Due to the assumption there exists at least one $v_a \in V'$ with $v_a \notin V_{k-1}$ and at least one $v_b \in V''$ with $v_b \notin V_{k-1}$ in $\bar{G}$. However, $v_i \in V_{k-1}$ and its incident cut edge e_{ij} are not part of $\bar{G}$. Therefore, $\bar{G}$ consists of at least two connected components $\bar{G}(V')$, $\bar{G}(V'')$. This leads to a contradiction to Theorem 5.1. ■

Note that the given sets in Lemma 5.2 may not describe the whole set V_{k-1} but only a subset of nodes that are known to be outliers. There can also be more outliers but the center defining nodes always have to fulfil the property of connectedness. Assume that $\tilde{V} \subset V_{k-1}$ and $\hat{V} = V \setminus \tilde{V}$. If the connectedness of the induced graph $G(\hat{V})$ is not given, there have to be more outliers in $\hat{V}$.

Corollary 5.3 ▶ Increase of the outlier subset

Let $G = (V, E)$ be an unweighted graph and let x be a feasible solution of the k-max problem on G with $2 \leq k \leq n$ fixed and a set of outliers V_{k-1} corresponding to x. Moreover, suppose that $\tilde{V} \subset V_{k-1}$ and $\hat{V} = V \setminus \tilde{V}$. If the induced subgraph $G(\hat{V})$ decomposes into $a \geq 2$ connected components $G(\hat{V}_1), \ldots, G(\hat{V}_a)$ with $\hat{V}_1 \cup \ldots \cup \hat{V}_a = \hat{V}$, it holds that

$$\left(\bigcup_{s=1,\ldots,a} \hat{V}_s \right) \setminus \hat{V}_t \subseteq V_{k-1} \text{ for one } t \in \{1, \ldots, a\}.$$

Proof. Follows directly with Theorem 5.1 and Lemma 5.2. ■

Thus, the node set of only one connected component defines a candidate set for the center defining nodes, the nodes of all other components are outliers. As a consequence, if an outlier $v_i \in V^*_{k-1}$ of an (unknown) optimal solution is known, a $\hat{k}_s$-max problem with a reduced value $\hat{k}_s = k - (n - |\hat{V}_s|)$, $s = 1, \ldots, a$, can be solved on all connected components of $G(\hat{V})$ to find the optimal set of center defining nodes. Clearly, size and complexity of the subproblems depend on the cut set and on the initially known outlier.

Figure 5.1 gives an illustrative example of the previous results for a tree graph. Note that every edge of a tree is a cut of size one as required for Lemma 5.2. Moreover, a tree has always at least one leaf among the outliers.

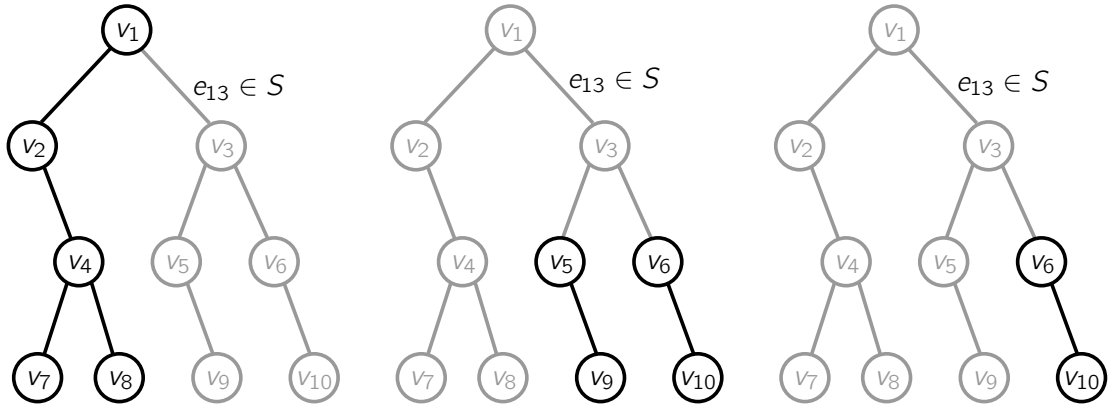


Fig. 5.1: Tree with cut edge e_{13} and known outlier v_3. Then (a) $V' = \{v_3, v_5, v_6, v_9, v_{10}\}$, or (b) $V'' \cup \{v_3\} = \{v_1, v_2, v_4, v_7, v_8, v_3\}$ are outliers. (c) If $V'' \cup \{v_3\} \subseteq V_{k-1}$, the nodes of a whole connected component of $G(V' \setminus \{v_3\})$ have to be outliers, see Theorem 5.1.

The result of Lemma 5.2 can be generalised for a cut consisting of more than one edge, provided that one of the endpoints of every edge in the cut is known to be an outlier.

Lemma 5.4 ▶ Outlier subsets by cuts of arbitrary size

Let $G = (V, E)$ be an unweighted graph and let x be a feasible solution of the k-max problem on G with $2 \leq k \leq n-1$ fixed and a set of outliers V_{k-1} corresponding to x. Moreover, let $S = (V', V'') = \{e_{ab} \in E : v_a \in V', v_b \in V''\}$ be a cut in G.
If it holds for every edge $e_{ab} \in S$ that $v_a \in V_{k-1} \ \vee \ v_b \in V_{k-1}$ then

$$V' \cup \{v_b \in V'' : v_b \in V_{k-1}\} \subseteq V_{k-1} \quad \vee \quad V'' \cup \{v_a \in V' : v_a \in V_{k-1}\} \subseteq V_{k-1}.$$

Proof. Assume that $V' \nsubseteq V_{k-1}$ and $V'' \nsubseteq V_{k-1}$. Let $S = \{e_{ab} \in E : v_a \in V', v_b \in V''\}$ be the cut set. Moreover, let $\bar{G}(\bar{V})$ with $\bar{V} = V \setminus V_{k-1}$ be the induced subgraph of G. Following Theorem 5.1, $\bar{G}$ has to be connected. Due to the assumption there exists at least one $v_i \in V'$ with $v_i \notin V_{k-1}$ and at least one $v_j \in V''$ with $v_j \notin V_{k-1}$ in $\bar{G}$ while for each edge in S either $v_a \in V_{k-1}$ or $v_b \in V_{k-1}$ and its incident cut edge are not part of $\bar{G}$. Therefore, $\bar{G}$ consists of at least two connected components $\bar{G}(V')$, $\bar{G}(V'')$. This leads to a contradiction to Theorem 5.1. ∎

Note that for a given cut it is in general not known whether an endnode of a cut edge is an outlier or not. Similarly, it can not be said which side of the cut edge only consists of outliers and which not. This can not be decided without computing an optimal new location of the corresponding k-max problem. The presented properties of outliers are rather intended to derive upper and lower bounds for a Branch-and-Bound approach to solve k-max problems on graphs. Furthermore, Lemma 5.2 above and Lemma 5.5 below can be used to decide in a preprocessing phase which nodes of the graph can not be outliers.

Lemma 5.5

Let $G = (V, E)$ be an unweighted graph and let x be feasible solution of the k-max problem on G with $2 \leq k \leq n-1$ and a set of outliers V_{k-1} corresponding to x. Moreover, $S = (V', V'') = \{e_{ij}\}$ for $i, j \in \{1, \ldots, n\}$ be a cut of size $|S| = 1$ in G. If $|V'| \geq k$ and $|V''| \geq k$ then $v_i, v_j \notin V_{k-1}$.

Proof. Assume w.l.o.g. that $v_i \in V'$ is an outlier. Following Lemma 5.2 it holds that either $V' \subseteq V_{k-1}$ or $V'' \cup \{v_i\} \subseteq V_{k-1}$. Since $|V'| \geq k$ it follows that the cardinality of the outlier set is $|V_{k-1}| \geq k$. This contradicts the definition of outliers being the nodes realising the $k-1$ largest distances to x. The case $v_j \in V_{k-1}$ can be formulated analogously. ∎

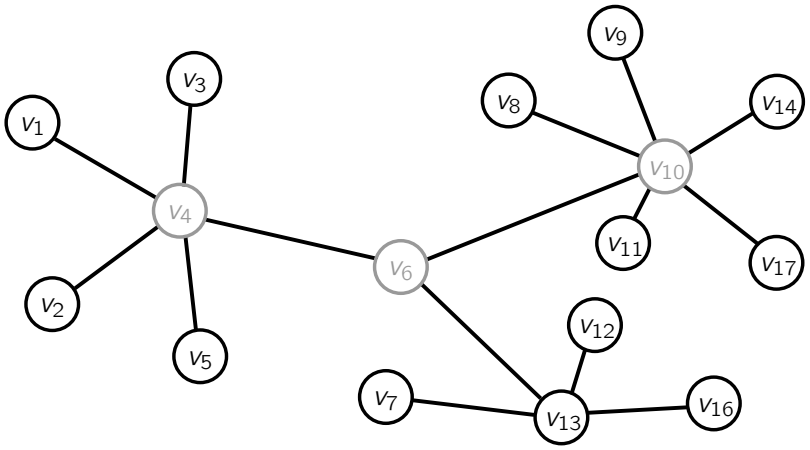


Fig. 5.2: For $k = 5$ nodes v_4, v_6, v_{10} are known to be center defining due to Lemma 5.5.

Example 5.6. *The graph shown in Figure 5.2 is given. The length l_j, $j = 1, \ldots, m$, of the edges do not matter in this example. Let $k = 5$, i.e., a solution of the k-max problem with four outliers is sought. With $S = \{e_{46}\}$ it follows that $|V'| = 5$ and $|V''| = 11$. Therefore, neither v_4 nor v_6 can be an outlier since otherwise the problem would have at least five outliers due to Lemma 5.2. With $S = \{e_{6,10}\}$ and corresponding $|V'| = 9, |V''| = 6$ also node v_{10} is center defining, i.e., it can not be an outlier. $S = \{e_{6,13}\}$ in contrast does not yield a new result since $|V'| = 12$ and $|V''| = 4$. Hence, v_{13} could be an outlier as the subset V'' is not too large with respect to k. Thus, a most contiguous set of size 12 has to contain the nodes v_4, v_6, v_{10}. This reduces the number of sets to analyse in Algorithm 2 from $\binom{16}{12} = 1820$ to $\binom{13}{9} = 715$.*

Lemma 5.5 can, for example, be used to improve Algorithm 2 which determines a set of most contiguous facilities in V and solves a center problem on this subset of the existing facilities. The most contiguous sets are testes using a complete enumeration, i.e., all possible subsets of V with cardinality $n-k+1$ are considered. Applying Lemma 5.5 in a preprocessing phase can reduce the number of subsets that need to be analysed since some nodes are known that can not be outliers due to the maximum number of outliers. This may lead to

a significant speed up of the solution process in practice. However, it does not lead to an improved bound on the theoretical complexity of Algorithm 2. The exact number of cuts in the graph as well as the number of cuts fulfilling the property of Lemma 5.5 depend on the structure of the graph. Hence, the number of identifiable center defining nodes can not be quantified. All minimum cuts of a general graph can be computed in $\mathcal{O}(n^2 \log^3(n))$ time with the randomised algorithm of Karger and Stein (1996). The overall complexity bound for Algorithm 2 is therefore not enlarged by the preprocessing phase. In practice, this approach is especially useful for tree graphs as every edge is a cut of size one and the computation of the cut sets can be skipped.

5.2 Finite Dominating Set Based on Equilibrium Points

In a general p-k-max problem (pkMG), the new locations can be placed without restrictions on the underlying graph G. Thus, infinitely many points are possible new locations for the optimal solution. Since Hakimi (1964) demonstrated for the first time that it is useful to restrict the analysis of a problem to a smaller set of candidate points, this approach is the basis for many algorithms in location theory that solve the problems by enumerating (all) candidates of the determined set. Therefore, the aim of this (as well as the next) section is to reduce the infinite set of possible points on the edges $A(G)$ to a finite dominating set (FDS). i.e., a finite set of candidates guaranteeing to contain at least one optimal solution. For more information on solution approaches based on finite dominating sets, see, for example, Hooker et al. (1991). This paper unifies and generalises results on finite dominating set on different kinds of location problems. More recent literature dealing with finite dominating sets for special instances of the ordered median problem are, for example, Pérez-Brito et al. (1997), Nickel and Puerto (1999), Kalcsics et al. (2002), Kalcsics and Nickel (2003), Rodríguez-Chía et al. (2005) and Tang et al. (2009). A short description of the papers can be found in the literature review of this thesis, see Chapter 3.

In the case of $p = 1$, the k-max problem on graphs can be solved using a finite dominating set based on the equilibrium points of the graph. To prove this result, the properties of the objective function have to be analysed further in a first step. For this purpose, let $e_g = (v_{a_g}, v_{b_g}) \in E$ be an arbitrary but fixed edge of the graph with $g \in \{1, \ldots, m\}$ and $a_g, b_g \in \{1, \ldots, n\}$ such that $v_{a_g}, v_{b_g} \in V$. Moreover, let

$$EQ^{a_g b_g} = \bigcup_{i,j \in \{1,\ldots,n\}} EQ_{ij}^{a_g b_g}$$

be the set of all equilibrium points on the edge e_g. The elements of the set $EQ^{a_g b_g}$ together with the nodes $v_{a_g}, v_{b_g} \in e_g$ define a subdivision S_g on e_g. A subset $C \subseteq e_g$ of the subdivision S_g is called a cell. Every cell is a closed and convex edge segment $[\alpha, \beta]$ with $\alpha, \beta \in EQ^{a_g b_g} \cup \{v_{a_g}, v_{b_g}\}$. Thus, the subdivision has the properties that the union of all its edge segments constitutes the whole graph and that the intersection of two edge segments consists of exactly one $x \in EQ^{a_g b_g} \cup V$. The set of all cells on e_g is denoted by $C(S_g)$. For

this subdivision the following result holds.

Lemma 5.7 ▶ Concavity and piecewise linearity of the k-max function
The objective function (kMF) of the k-max problem on graphs with positive node weights w_i, $i = 1, \ldots, n$, is piecewise linear and concave on every cell $C \in C(S_g)$, $g = 1, \ldots, m$.

Proof. Let $C = [\alpha, \beta]$ be a cell of S_g on an edge $e_g \in E$ of G, i.e., $\alpha, \beta \in EQ^{a_g b_g} \cup \{v_{v_g}, v_{b_g}\}$. Note that the sorting of distances in $d^w(V, x)$ only changes in points of EQ and that there is no $\tilde{x} \in EQ$ in the interior of C. Consequently, there exists a permutation $\sigma \in \Sigma(x)$ such that

$$d^w(v_{\sigma(1)}, x) > \ldots > d^w(v_{\sigma(n)}, x) \quad \text{for all} \quad x \in C \setminus \{\alpha, \beta\},$$

where α, β are the extreme points of the edge segment C. In α and β it holds

$$d^w(v_{\sigma(1)}, \alpha) \geq \ldots \geq d^w(v_{\sigma(n)}, \alpha) \quad \text{and} \quad d^w(v_{\sigma(1)}, \beta) \geq \ldots \geq d^w(v_{\sigma(n)}, \beta).$$

The k-max objective function can be written as

$$k\text{-max}(d^w(V, x)) = \lambda_1 w_{\sigma(1)} d(v_{\sigma(1)}, x) + \ldots + \lambda_k w_{\sigma(k)} d(v_{\sigma(k)}, x) + \ldots + \lambda_n w_{\sigma(n)} d(v_{\sigma(n)}, x)$$

with $\lambda_k = 1$ and $\lambda_i = 0$ for $i \neq k$ on every cell $C \in C(S_g)$. As the permutation σ is constant for all $x \in C$, non-linearities of the k-max function can only be caused by the non-linearities of the distance functions $d^w(v_i, x)$. As the functions $d^w(v_i, x)$ are piecewise linear and concave over every edge $e_g \in E$ and $C \subseteq e_g$ for some $g \in \{1, \ldots, m\}$, also the k-max function has to be piecewise linear and concave on every cell $C \in C(S_g)$. ∎

Even though the piecewise linearity and the concavity of the k-max function is enough to identify possible minima on the boundary of the cells, for the sake of completeness also the regions in which the objective function is linear shall be identified in the following. Let, for this purpose,

$$BN^{a_g b_g} = \bigcup_{i \in \{1, \ldots, n\}} BN_i^{a_g b_g}$$

be the set of all bottleneck points of G on $e_g \in E$, $g \in \{1, \ldots, m\}$. Then, the equilibrium points $EQ^{a_g b_g}$, the nodes $v_{a_g}, v_{b_g} \in e_g$ and the bottleneck points $BN^{a_g b_g}$ induce a subdivision S'_g on e_g. The set of cells of S'_g is denoted by $C(S'_g)$. For all $C = [\alpha, \beta] \in C(S'_g)$ it holds that

$$\alpha, \beta \in EQ^{a_g b_g} \cup BN^{a_g b_g} \cup \{v_{v_g}, v_{b_g}\}.$$

As it is shown in the proof of Lemma 5.7, the permutation $\sigma \in \Sigma(x)$ of the elements of the vector of distances $d^w(V, x)$ is constant for all $x \in C \setminus \{\alpha, \beta\}$ for $C = [\alpha, \beta] \in C(S_g)$. To define the linearity regions of the k-max function, also the linearity domains of the functions $d^w(v_i, x)$ for all $i = 1, \ldots, n$ have to be considered. Since $d^w(v_i, x)$, $i \in \{1, \ldots, n\}$, is piecewise linear and concave over the edge e_g, the linearity regions are obviously either the whole edge e_g for functions $d^w(v_i, x)$ without a breakpoint on e_g or the edge segments $[v_{a_g}, \bar{x}]$ and $[\bar{x}, v_{b_g}]$ for functions having a bottleneck point $\bar{x} \in BP^{a_g b_g}$. Thus, for a cell

$C' \in C(S'_g)$, it holds that the ordering of the elements of the distance vector is constant and that the functions $d^w(v_i, x)$, $i = 1, \ldots, n$, are linear. Hence, for the subdivision S'_g the following statement holds.

Corollary 5.8 ▶ Linearity of the k-max function

The objective function (kMF) of the k-max problem on graphs with positive node weights w_i, $i = 1, \ldots, n$, is linear on every cell $C' \in C(S'_g)$.

Although the objective function is linear on a compact cell, the set of optimal new locations does only consist of single points $x \in A(G)$. This is due to the assumption of strictly positive lengths $l_g > 0$, $g = 1, \ldots, m$, of the edges and strictly positive weights $w_i > 0$, $i = 1, \ldots, n$. If also $l_g = 0$ is allowed, the optimal set may include the union of corner points of cells and complete cells $C \in C(S'_g)$.

Remark 5.9. *For the special case of G being a tree graph, the subdivision S_g is sufficient to define the linearity regions of the k-max function. More precisely, the k-max function is linear on every cell $C \in C(S_g)$. A tree graph is connected and has no cycles. Thus, every pair of two vertices $v_i, v_j \in V$ are connected by a unique path in G. As a consequence, G does not have bottleneck points, i.e., $BP = \emptyset$ and therefore $S_g = S'_g$.*

As the objective function is piecewise linear and concave on a cell $C \in C(S_g)$, the extreme points of the cells of the subdivision S_g are candidates for optimal solutions of the k-max problem. Thus, based on Lemma 5.7, the following theorem gives a finite dominating set for the k-max problem on general graphs.

Theorem 5.10 ▶ FDS based on equilibrium points

All optimal solutions of the k-max problem with $w_i > 0$ for all $i \in \{1, \ldots, n\}$ can be found in the set

$$\text{Cand} = \begin{cases} EQ, & k \leq n-1 \\ V, & k = n. \end{cases}$$

Proof. Let $x \in A(G)$ and $\sigma \in \Sigma(x)$, i.e.,

$$d^w(v_{\sigma(1)}, x) \geq \ldots \geq d^w(v_{\sigma(k)}, x) \geq d^w(v_{\sigma(k+1)}, x) \geq \ldots \geq d^w(v_{\sigma(n)}, x). \tag{4.1}$$

Case 1: $k \leq n-1$

Let $x \in e_{ab}$ be an optimal location with optimal objective value z^* for the k-max problem (an optimal solution exists according to Theorem 4.6). Assume that $x \notin EQ$. Note that if x is also not in EQ', then $x \in e_{ab} \in E$ and can be shifted by a sufficiently small step length towards v_a and similarly towards v_b without changing the permutation σ. Thus, for the case $x \notin EQ'$, the objective function value improves by shifting x into an equilibrium point.

Now let $l \geq k$ be the smallest index such that

$$d^w(v_{\sigma(k)}, x) = d^w(v_{\sigma(k+1)}, x) = \ldots = d^w(v_{\sigma(l)}, x) > d^w(v_{\sigma(l+1)}, x).$$

Since $x \notin EQ \cup V$ (and thus, in particular, $x \in e_{ab} \in E$ is not in the relative boundary of EQ'_{ab} for all $v_a, v_b \in V$, $a \neq b$) it follows that all shortest paths $p_{\sigma(i)}$ from x to $v_{\sigma(i)}$, $i = k, \ldots, l$, either all pass through the node v_a or all pass through the node v_b (see Figure 5.3). Therefore, x can be moved a small distance ε towards the node $v_{\sigma(k)}$ with

$$0 < \varepsilon < \frac{1}{2}\left(d^w(v_{\sigma(k)}, x) - d^w(v_{\sigma(l+1)}, x)\right)$$

such that no point in $EQ \cup V$ is crossed, leading to a new location $\bar{x}$. This does not change the ordering (4.1), because all distances $d^w(v_{\sigma(k)}, x), \ldots, d^w(v_{\sigma(l)}, x)$ are reduced by ε.

Thus, $\bar{x}$ has an objective value of $d^w(v_{\sigma(k)}, \bar{x}) < d^w(v_{\sigma(k)}, x) = z^*$, contradicting the assumption.

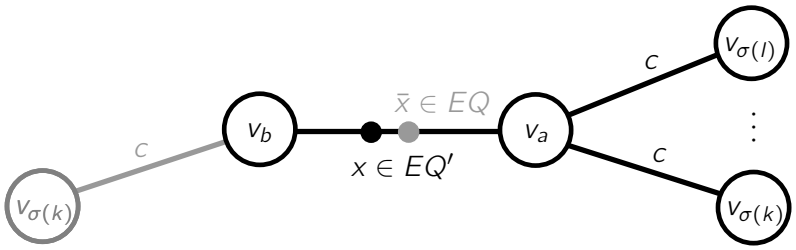


Fig. 5.3: All paths from $v_{\sigma(i)}$, $i = k, \ldots, l$, to $x \in EQ'$ contain node v_a. If otherwise (grey) there exist a path from $\bar{x} \in e_{ab}$ to $v_{\sigma(j)}$ for at least one $j \in \{k, \ldots, l\}$ that contains node v_b while all other paths from $\bar{x}$ to $v_{\sigma(i)}$ with $i \in \{k, \ldots, l\} \setminus \{j\}$ go through v_a, it follows that $\bar{x} \in EQ$.

Case 2: $k = n$

For $k = n$ the aim is to solve the n-max problem. With $\sigma \in \Sigma(x)$, minimizing the nth largest distance is equivalent to minimizing the smallest distance

$$\min_{x \in A(G)} \; n\text{-}\max_{v_i \in V}(d^w(v_i, x)) = \min_{x \in A(G)} \min_{1 \leq i \leq n} d^w(v_i, x).$$

Since the distances are bounded from below by 0, it follows

$$\min_{x \in A(G)} \min_{1 \leq i \leq n} d^w(v_i, x) \geq 0.$$

Thus, the smallest possible value that can be attained is 0. As $d^w(v_i, x) = 0$ if and only if $x = v_i$ for all $i \in \{1, \ldots, n\}$, the new facility has to be located in an arbitrary existing facility to realise this value. ∎

Note that, for the special case $k = n$, the whole candidate set $\text{Cand} = V$ is optimal and that the optimal objective function value is always $z^* = 0$. Thus, the optimal solutions are known without any computations in this case. However, more important is that in all other cases for $k < n$, the finite dominating set does not depend on k. Hence, all optimal

solutions of the k-max problem for $k = 1, \ldots, n-1$ can be computed simultaneously with one evaluation for each equilibrium point.

Note that the extreme points of the cells $C' \in C(S'_g)$ of course also form a finite dominating set of the k-max problem. However, as S' has more candidate points than S, the second set is not considered further. Theorem 5.10 provides a necessary condition for an optimal location of the k-max problem.

Corollary 5.11 ▶ Necessary optimality condition

Let $k < n$. For all optimal solutions $x^* \in \mathcal{X}$ of the k-max problem it holds that $x^* \in EQ$.

Proof. Follows directly with the proof of Theorem 5.10. ∎

Building on the result of Theorem 5.10, it is enough to evaluate the k-max function in all equilibrium points and to choose the points leading to the smallest objective function value as optimal locations. The following algorithm gives a first short overview of the procedure.

Algorithm 5 k-max problems on graphs using Cand (Outline)

Input: Graph $G = (V, E)$ with node weights $w_i > 0$ for all $i = 1, \ldots, n$, and $k \in \{1, \ldots, n\}$.

1: **if** $k < n$ **then**
2: Determine the candidate set EQ of all equilibrium points of G
3: **for all** $e_{ab} \in E$ **do**
4: Sort equilibrium points on e_{ab} such that $t_1 \leq t_2 \leq \ldots \leq t_{|EQ^{ab}|}$ with $x_i = (e_{ab}, t_i)$ for all $i = 1, \ldots, |EQ^{ab}|$
5: Evaluate the k-max function in candidate x_1
6: **for all** $i = 1, \ldots, |EQ^{ab}|$ **do**
7: Determine $\sigma_{x_i} \in \Sigma(x_i)$ from $\sigma_{x_{i-1}}$
8: Set $f_k(x_i) = d^w(v_{\sigma(k)}, x_i)$
9: **else**
10: Set Cand $:= V$ with $f_k(x) = 0$ for all $x \in$ Cand.

Output: Set of optimal solutions $\mathcal{X} = \{x^* : f_k(x^*) = \min_{x \in \text{Cand}} f_k(x)\}$ with $z^* = d^w(v_{\sigma(k)}, x^*)$.

For $k < n$, the derivation of the equilibrium points can be done, for example, by calculating in $\mathcal{O}(n^2)$ time the solutions of the equations $d^w(v_j, x) = d^w(v_i, x)$, where $v_i, v_j \in V$, $v_i \neq v_j$, over each edge of G (see Figure 4.4). Thus, the calculation of all equilibrium points takes $\mathcal{O}(mn^2)$.

The evaluation of the candidates is done on each edge e_{ab} separately: At first the candidate points on e_{ab} have to be sorted such that they are sorted increasingly along the edge which can be done in $\mathcal{O}(n^2 \log(n))$ time. The leftmost equilibrium point x_1 on e_{ab} can be evaluated in $\mathcal{O}(n \log(n))$ time by computing and sorting the weighted distances of x_1 to all existing

facilities. The permutation $\sigma_{x_i} \in \Sigma(x_i)$ of an equilibrium point x_i can be easily deduced from the permutation $\sigma_{x_{i-1}} \in \Sigma(x_{i-1})$ for consecutive points x_{i-1} and x_i. Let therefore $v_{\alpha_i}, v_{\beta_i}$ with $\alpha_i, \beta_i \in \{1, \ldots, n\}$ be the nodes defining x_i and let analogously $v_{\alpha_{i+1}}, v_{\beta_{i+1}}$ with $\alpha_{i+1}, \beta_{i+1} \in \{1, \ldots, n\}$ be the nodes defining x_{i+1}. To get $\sigma_{x_{i-1}}$ from σ_{x_i}, only the elements α_{i+1} and β_{i+1} in σ_{x_i} have to be swapped. The objective values of the remaining $\mathcal{O}(n^2)$ candidates x_i, $i = 2, \ldots, |EQ^{ab}|$, can therefore be obtained in constant time. Hence, this leads to a complexity of $\mathcal{O}(n \log(n) + n^2)$ for the evaluation of the candidate points on one fixed edge. Note that the defining nodes of all equilibrium points have to be stored for this purpose. A worst case bound of $\mathcal{O}(mn^2 \log(n))$ for Algorithm 5 follows.

It is possible to reduce the complexity of Algorithm 5 slightly to $\mathcal{O}(m(n+s)\log(n))$ by using the algorithm of Bentley and Ottmann (1979) to calculate the equilibrium points. This algorithm has a complexity of $\mathcal{O}(m(n+s)\log(n))$, where $s \leq \binom{n}{2}$ (see Algorithm 3). Even though this effort is higher than $\mathcal{O}(mn^2)$ for $s > \binom{n}{2}$, the advantage is that the needed permutation in every equilibrium point as well as the corresponding objective function value are obtained with the algorithm of Bentley and Ottmann (1979) without causing any additional effort: In each event point $x = i_{S_1,S_2}$ during the sweep, the data structure R contains the order relation among the line segments with respect to the sweepline in x. Since the segment S_i is related to the distance function $d^w(v_i, x)$, the ordering $S_i \succeq S_j$ implies $d^w(v_i, x) \geq d^w(v_j, x)$. Thus, the current R contains the needed permutation $\sigma \in \Sigma(x)$. In step 3 it is therefore not necessary to get the whole vector of distances to all existing nodes but only the component $d^w(v_{\sigma(k)}, x)$ for all $x \in EQ$. As this value is given by the distance of an optimal equilibrium point x to its defining nodes v_i, v_j, it it not necessary to compute shortest paths to $v_{\sigma(k)}$. It is given during the derivation of the equilibrium points on the ordinate axis of the arrangement of line segments above the current edge.

Example 5.12 (Continuation of Example 4.20)**.** *The distances $d^w(v_i, EQ_{ij})$ of the equilibrium points on the edge e_{12} can be taken from the ordinate axis of the arrangement of line segments shown in Figure 4.6. The Permutation $\sigma \in \Sigma(EQ^{12}_{ij})$ valid at an equilibrium point EQ^{12}_{ij}, $i, j \in \{1, \ldots, 5\}$, $i \neq j$, is given by the sorting of the line segments in each equilibrium point:*

$$
\begin{array}{llll}
EQ^{12}_{34} & \text{with} & d^w(v_3, EQ^{12}_{34}) = 6 & \text{and} \quad \sigma = \{3,4,2,5,1\} \\
EQ^{12}_{25} & \text{with} & d^w(v_2, EQ^{12}_{25}) = \frac{3}{2} & \text{and} \quad \sigma = \{3,4,5,2,1\} \\
EQ^{12}_{12} & \text{with} & d^w(v_1, EQ^{12}_{12}) = \frac{4}{3} & \text{and} \quad \sigma = \{3,4,5,1,2\} \\
EQ^{12}_{15} & \text{with} & d^w(v_1, EQ^{12}_{15}) = 2 & \text{and} \quad \sigma = \{3,4,1,5,2\} \\
EQ^{12}_{14} & \text{with} & d^w(v_1, EQ^{12}_{14}) = 3 & \text{and} \quad \sigma = \{3,1,4,5,2\} \\
EQ^{12}_{13} & \text{with} & d^w(v_1, EQ^{12}_{13}) = \frac{18}{5} & \text{and} \quad \sigma = \{1,3,4,5,2\} \\
EQ^{12}_{45} & \text{with} & d^w(v_4, EQ^{12}_{45}) = 2 & \text{and} \quad \sigma = \{1,3,5,4,2\}.
\end{array}
$$

Note that there are at least two valid permutations in every equilibrium point as at least two

Algorithm 6 Solving k-max problems on graphs using Cand

Input: Graph $G = (V, E)$ with $w_i > 0$ for all $i = 1, \ldots, n$, $k \in \{1, \ldots, n\}$; heap Q with root $r(Q)$: set of endpoints $e(S) = (t, y)$, sorted lexicographically by t and y-value

Set $z_b := \infty$ and $X_b := \emptyset$
if $k < n$ **then**
  for all $e_s \in E$ **do**
    while $Q \neq \emptyset$ **do** ▷ Determine equilibrium points
      Set $x := r(Q)$
      if x is right endpoint of segment S_i **then** ▷ Delete S_i from the ordering
        DELETION(x, R, Q)
      if x is left endpoint of segment S_i **then** ▷ Include S_i into the ordering
        INSERTION(x, R, Q)
      if x is intersection point of $S_i \succeq S_j$ **then** ▷ Swap ordering to $S_j \succeq S_i$
        NEWEQUILIBRIUMPOINTANDPERMUTATION(x, z_b, R, Q, EQ)
  Set $X_b := \bigcup_{s=1,\ldots,m} X_b^s$ ▷ Local optimal solutions of edges
else $(k = n)$ ▷ Ex. facilities are optimal
  $X_b := V$

Output: Set of optimal solutions $\mathcal{X} = \{x^* : x^* \in X_b\}$ with $z^* = d^w(v_{\sigma(k)}, x^*)$.

function NEWEQUILIBRIUMANDPERMUTATION(x, z_b, R, Q, EQ)
  $EQ = EQ \cup \{x\}$
  Swap positions of i and j in R such that $S_j \succeq S_i$ ▷ Change ordering
  Determine σ_x from current R ▷ Permutation in $x \in EQ$
  $z := w_{\sigma(k)} d(v_{\sigma(k)}, x)$ ▷ Objective function value in x
  if $z = z_b$ **then**
    $X_b^s := X_b^s \cup \{x\}$ ▷ As good as current best solutions
  else if $z < z_b$ **then**
    $X_b^s := \{x\}$ ▷ New best solution
    $z_b := z$
  if S_j intersects $u(S_j)$ **then** ▷ Check upper neighbour
    Insert $i_{S_j, u(S_j)}$ into Q
  if S_i intersects $l(S_i)$ **then** ▷ Check lower neighbour
    Insert $i_{S_i, l(S_i)}$ into Q
  return R, Q, EQ
end function

elements of the distance vector are equal. For the evaluation of the candidate points it does not matter which of the equivalent permutations of one equilibrium point is chosen.

Finding the optimal solution among the best solutions of every edge can be implemented in $\mathcal{O}(m)$ time. Thus, the overall complexity is bounded by $\mathcal{O}(m(n+s)\log(n))$ with $s \leq \binom{n}{2}$ for $k < n$, which is an improvement of a factor of n that is achieved by using the algorithm of Bentley and Ottmann. This approach is summarised in Algorithm 6, where the heap Q, the procedures INSERTION, DELETION and other notations are the same as those used for Algorithm 3. Note that the algorithm can be parallelised easily since the computation of the equilibrium points on a fixed edge can be done independently from the other edges. Moreover, also the evaluation of the single candidates can be implemented in parallel as these processes do not depend on each other. This leads to a significant improvement of the computation time.

The calculation process of the equilibrium points of a graph G provides additional important information that should be stored together with the corresponding equilibrium point $y \in EQ$. To extract this information, for each equilibrium point $y_{ij} \in EQ_{ij}$ a vector of the form

$$y_{ij} = \left(e_{ab}, t, \{v_i, v_j\}, d^w(v_i, y_{ij}), \sigma_{ij}\right),$$

is stored, where e_{ab}, $a, b \in \{1, \ldots, n\}$, is the edge on which y_{ij} is located and $t \in [0, 1]$ is the corresponding parameter to define the position on e_{ij}. $v_i, v_j \in V$ are the nodes defining y_{ij} and $d^w(v_i, y_{ij}) = d^w(v_j, y_{ij})$ is the weighted distance from y_{ij} to its defining nodes. $\sigma_{ij} \in \Sigma(y_{ij})$ gives a permutation of the distance vector in y_{ij}. Thus, in the following it can be assumed that with $y_{ij} \in EQ_{ij}$ this information is also known.

Remark 5.13. *As a consequence of the proof of Theorem 5.10, it is in practice useful to rearrange the set of equilibrium points such that its elements y_{ij} with $i, j \in \{1, \ldots, n\}$ are sorted in non-decreasing order by their distances $d^w(v_i, y_{ij})$ to their defining nodes. If the candidates are evaluated in this order, the evaluation of the candidate points can be terminated when the first feasible solution $\bar{y}_{gh}$ with $\bar{z} = d^w(v_g, \bar{y}_{gh})$ for $g, h \in \{1, \ldots, n\}$ is found. This solution is then obviously optimal as all solutions considered later have equal or larger objective function values. If all optimal solutions are needed, the equilibrium points $\tilde{y}_{pq}$ with $d^w(v_p, \tilde{y}_{pq}) = \bar{z}$ also have to be evaluated. The optimal solutions of the k-max problem are (especially for larger values of k) found much faster in general when this sorting of the equilibrium points is applied. Note that the edge-wise evaluation of the equilibrium points is not possible with this approach and that the theoretical complexity of Algorithm 5 is thus increased to $\mathcal{O}(mn^2(\log(mn) + n\log(n))$. However, the practical speed up is worth the additional effort (see Chapter 7.1). A bisection approach on the candidate set is not possible, see Remark 6.9.*

Example 5.14 (Continuation of Example 5.12)**.** *Evaluating the equilibrium points of the example graph in Figure 4.5 leads to the following optimal solutions of the k-max problem: For $k = 1$, an optimal solution is given by*

$$x^1 = \left(e_{13}, \frac{3}{4}\right) \quad \textit{resp.} \quad \bar{x}^1 = \left(e_{35}, \frac{1}{2}\right) \quad \textit{with} \quad z^1 = 3.$$

The 2-max *problem is solved optimally with*

$$x^2 = \left(e_{34}, \frac{1}{3}\right) \quad \textit{and} \quad z^2 = \frac{4}{3}.$$

For $k = 3$ *it holds that*

$$x^3 = (e_{13}, 1) \quad \textit{with} \quad z^3 = 1.$$

For $k = 4$,

$$x^4 = \left(e_{15}, \frac{1}{3}\right) \quad \textit{resp.} \quad \bar{x}^4 = \left(e_{24}, \frac{2}{3}\right) \quad \textit{with} \quad z^4 = \frac{2}{3}.$$

is optimal. The solutions are illustrated in Figure 5.4

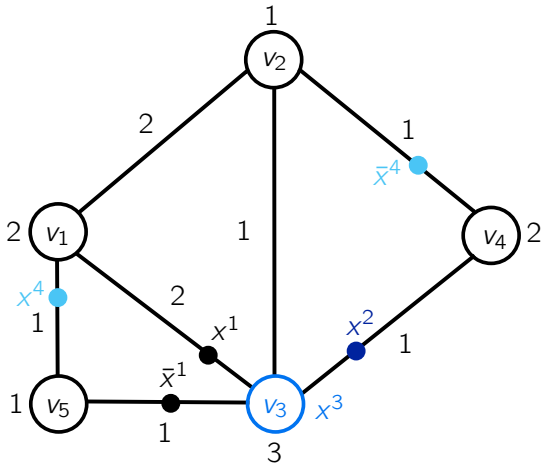


Fig. 5.4: Optimal new facility for $k = 1$ (two alternatives), $k = 2$, $k = 3$ and $k = 4$ (two alternatives)

An advantage of Algorithm 6 is that it can be adjusted easily to compute simultaneously the sets of optimal solutions $\mathcal{X}(k)$ for all possible values of the parameter k without increasing the overall complexity. An equilibrium point $y_{ij} \in EQ_{ij}$ is optimal for a value k only if

$$d^w(v_{\sigma(k)}, y_{ij}) = d^w(v_i, y_{ij}) \quad \text{and} \quad d^w(v_{\sigma(k+1)}, y_{ij}) = d^w(v_j, y_{ij})$$

resp.

$$d^w(v_{\sigma(k)}, y_{ij}) = d^w(v_j, y_{ij}) \quad \text{and} \quad d^w(v_{\sigma(k+1)}, y_{ij}) = d^w(v_i, y_{ij})$$

for a permutation $\sigma \in \Sigma(y_{ij})$, which follows from the proof of Theorem 5.10 (see also Lemma 5.21). As all equilibrium points have to be computed and their objective function values have to be evaluated (even if the value of k is fixed beforehand), it does not cause additional effort to associate every equilibrium point to the sets $\mathcal{X}(i)$ and $\mathcal{X}(j)$ for which it may be an element. In order to compare a new solution with the currently best solution in $\mathcal{X}(i)$ resp. $\mathcal{X}(j)$, only two comparisons are needed. In this way, the optimal solutions of the k-max problem for all $k = 1, \ldots, n$ can be computed simultaneously in polynomial time $\mathcal{O}(m(n+s)\log(n))$ with $s \leq \binom{n}{2}$ being the number of equilibrium points on one edge.

5.3 Finite Dominating Set Based on h-levels

Another approach for a finite dominating set of the k-max problem with $k < n$ is based on the method of Hakimi (1964) to compute a finite dominating set for the absolute center problem on a graph which is defined as

$$\min_{x \in A(G)} f_1(x) = \min_{x \in A(G)} \max_{1 \le i \le n} d^w(v_i, x).$$

The idea here is to make a local analysis, i.e., to find a local center on each edge of G, similar to the derivation of the equilibrium points. Let $e_{ab} \in E$ with $v_a, v_b \in V$ be an arbitrary but fixed edge of the graph G. The piecewise linear, concave graphs of the weighted distance functions $d^w(v_i, x)$ can be computed over the edge e_{ab} as functions of $x = (e_{ab}, t),\ t \in [0, 1]$. This results in an arrangement L_{ab} of piecewise linear functions over this edge e_{ab}. These piecewise linear functions can also be interpreted as an arrangement of linear segments, where each linear segment of a piecewise linear function is treated independently (see equations (4.20)). An example for a graph with six nodes can be seen in Figure 5.5. Recall that, in the case of a fixed edge $e_{ab} \in E$ with $a, b \in 1, \ldots, n$, there is a one to one correspondence between a point $x = (e_{ab}, t)$ and the parameter $t \in [0, 1]$. As the following line arrangements are always considered for a fixed edge, x and t can be used equivalently.

The center problem aims at finding a point $x \in A(G)$ such that the maximum weighted distance between the existing nodes and the new location is minimised. Therefore, for each value of $x \in e_{ab}$, the maximum of all the lines in L_{ab} is of interest. This largest weighted distance depends on the position of x and is given by

$$F_{ab}(x) = \max_{1 \le i \le n} d^w(v_i, x) \quad \text{with} \quad x = (e_{ab}, t),\ t \in [0, 1].$$

The function $F_{ab}(x)$ is shown in Figure 5.5(b) in dark blue and is called the *upper envelope* of the arrangement L_{ab}. It is a curve consisting of line segments and break points which are called vertices of the envelope. The local minima of F_{ab} correspond to the local centers on the current edge. The computation of the upper envelope of an arrangement of line segments in the plane is a known and well investigated problem in the field of Computational Geometry. Hershberger (1989), for example, solves the problem by using a divide and conquer approach in $\mathcal{O}(n \log(n))$.

Now, to get the optimal position for the new location restricted to e_{ab}, the minimum of F_{ab} has to be found. This can be done quite easily since F_{ab} is a piecewise linear function that has a finite number of local minima. The elements of the set of local minima X^c_{ab} of F_{ab} are candidates for the optimal location of the new facility since they locally minimise the center objective value for all possible locations on e_{ab}.

By repeating this procedure and determining the local minima of F_{ab} for all $e_{ab} \in E$ with $v_a, v_b \in V$, a finite dominating set

$$\text{Cent} := \bigcup_{a,b \in \{1,\ldots,n\}} \{X^c_{ab} : e_{ab} \in E\}$$

for the center problem is obtained. A local minimum of F_{ab} can be characterised either by being an endpoint of e_{ab} with $t = 0$ or $t = 1$ or by being an intersection point of two weighted distance functions $d^w(v_i, x)$ and $d^w(v_j, x)$ with $i, j \in \{1, \ldots, n\}$, $i \neq j$, such that their slopes at $x' = (e_{ab}, t')$, $t' \in [0, 1]$, have opposite signs. The number of points which are candidates for being a local minimum on one edge is therefore at most $(n(n-1)/2)+2$. Hence, the cardinality of the corresponding candidate set C is bounded by $|C| = \mathcal{O}(mn^2)$.

Remark 5.15. *In practice, this bound on the number of candidate points overestimates the cardinality of points to evaluate. Not only the local minima of F_{ab} fulfil the mentioned properties, but also most of the intersection points below the upper envelope do so. Bounding the number of vertices of the upper envelope is a special case of the problem to bound the complexity of a so called k-level (see Definition 5.16). This is a well known problem in Computational Geometry for which no satisfactorily tight upper bound is known until now.*

The concept of k-levels will be introduced and used in the following to generalise the idea of Hakimi (1964) to k-max problems for $k \in \{2, \ldots, n-1\}$. The parameter k in the concept of a k-level has another meaning than for the k-max problems.To avoid confusion when generally describing the concept of k-levels, the parameter k (which in this thesis is always related to the k-max objective) will be replaced by a parameter h and thus it is talked of h-levels in the following, unless it is specially referred to the value of k.

Whereas the largest distance and therefore the upper envelope of the line arrangement was of interest for a center problem, the k-max problem needs, depending on the parameter k, the kth highest chain of line segments and vertices such that at most $k-1$ other chains lie above it resp. at least $n-k+1$ lie below (follows with Corollary 4.11). An h-level has similar properties and can be easily transferred. Agarwal et al. (1998b) define it as follows.

Definition 5.16 ▶ h-level

For an arrangement L of q line segments in general position in $\mathbb{R}^2$ and a given parameter $h \in \{1, \ldots, q-1\}$, the h-level $F^h(x)$ of L, $x \in \mathbb{R}$, is defined as the closure of all points on line segments which have exactly h line segments strictly below them.

Note that in general the number of line segments in L_{ab} is greater than n because a distance function over e_{ab} is a piecewise linear function of up to two linear segments. Thus, the number of line segments in L_{ab} is bounded by $q \leq 2n = \mathcal{O}(n)$. Roughly spoken, for $h = n-k$, the chain contributing to the objective function of the k-max problem described above corresponds to the (n-k)-level $F_{ab}^{n-k}(x)$ of the current arrangement L_{ab} on the edge e_{ab}. Definition 5.16 is based on linear segments in general position, i.e., it is assumed that there are no parallel lines and not more than two segments intersect in one point. In the arrangement L_{ab}, in contrast, it is in general likely that two or more line segments (or parts of them) are identical, i.e. that they share infinitely many points. In this case, the properties of the (n-k)-level are modified in the sense that it is defined as the closure of the set of all points on line segments that have *at least* $n-k$ line segments *on or* below them. Thereby it fulfils the requirements of the k-max problem to cover *at least* $n-k+1$ facilities. An example can be seen in Figure 5.5(b).

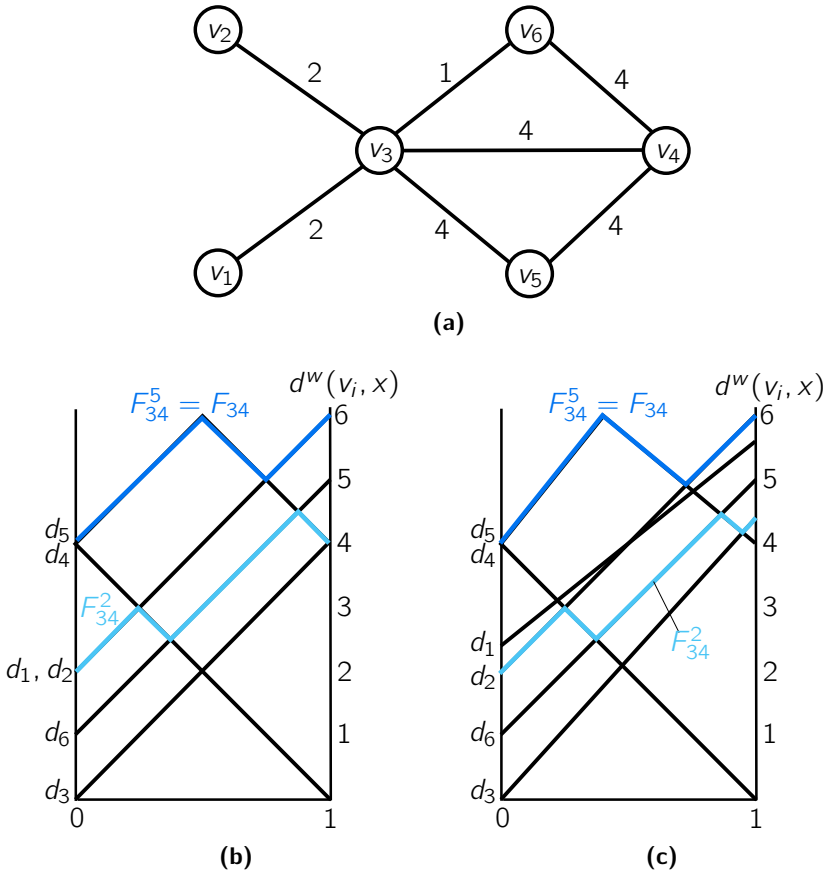


Fig. 5.5: (a) Example of a graph with $V = \{v_1, \ldots, v_6\}$ and $w_i = 1$ for all nodes. (b) The 5-level $F_{34}^5(x) = F_{34}(x)$ over e_{34} is equal to the upper envelope of the arrangement L_{34}. It's local minima are elements of the FDS for the 1-1-max problem resp. the center problem. The local minima of the 2-level $F_{34}^2(x)$ correspond to the FDS of the 1-4-max problem. (c) Arrangement L_{34} from subfigure (b) transformed to general position resulting in L_{34}^g.

The $(n\text{-}k)$-level represents the distance function from a location $x = (e_{ab}, t)$ on the edge e_{ab}, $t \in [0, 1]$, to the existing node that has the kth largest distance to x. Obviously, the kth furthest node depends on the location of x on e_{ab} and can only change in the vertices of the corresponding level. Hence, the local minima of $F_{ab}^{n-k}(x)$ are candidates for the optimal location of the new facility.

Corollary 5.17 ▶ FDS based on h-levels

Let $F_{ab}^{n-k}(x)$ be the $(n\text{-}k)$-level in the arrangement L_{ab} of line segments on $e_{ab} \in E$ with $a, b \in \{1, \ldots, n\}$. Then, all optimal solutions of the *1-k*-max problem with $1 \leq k \leq n-1$ can be found in the set

$$\text{Cand2} := \bigcup_{a,b \in \{1,\ldots,n\}} \left\{ \arg\min_{x=(e_{ab},t)} F_{ab}^{n-k}(x) \right\}.$$

Cand2 consists of the local minima of the corresponding (n-k)-level. The local minima that are intersection points of two distance functions are equilibrium points by definition but, in contrast to Cand, also nodes of G can belong to Cand2 even if $k < n$. On the other hand, the number of candidate points in Cand2 is much smaller in practice because just the intersection points that belong to the current level (and therefore to the correct value of k) are considered.

Remark 5.18. *Note that it is not sufficient to include just the global minima of $F_{ab}^{n-k}(x)$ in the finite dominating set* Cand2 *because for $k < n$, candidates for an optimal location have to lie in an intersection point of at least two distance functions, see Corollary 5.11. More precisely, the candidate location has to be an equilibrium point while the global minimum of $F_{ab}^{n-k}(x)$ could be attained in a point with $t = 0$ or $t = 1$ that is not an equilibrium point. An example is shown in Figure 5.5(c).*

For computing an h-level in the given arrangement of distance functions over e_{ab}, the linear segments are first moved into general position (see Figure 5.5(c)). This setting can always be constructed by adding a (sufficiently) small constant ε to the corresponding slopes such that the results of the transformation can still be identified with the original intersection points. Note that this can always be done in such a way that two formerly parallel lines do not intersect in the interior of e_{ab} after this transformation. In the following, it is assumed that L_{ab} is in general position.

Now, procedures for computing arrangements of q line segments can be applied, for example, the algorithm of Hershberger (1992). This algorithm iteratively computes the upper envelope of the arrangement. After each iteration, the current envelope is removed by discarding the segments or pieces of segments that appear on the envelope. In the next iteration, the next level is computed as the upper envelope of the remaining segments. This procedure is called *onion peeling* and can be implemented in $\mathcal{O}(q\alpha(q)\log^2(q))$ time, where $\alpha(q)$ is the very slow-growing inverse of the Ackermann's function introduced by Ackermann (1928). Further information and algorithms to compute an h-level of an arrangement of line segments can be found, for example, in Agarwal et al. (1998a) and Agarwal et al. (1998b). Hence, the complexity for deriving all vertices of all m (n-k)-levels is bounded by $\mathcal{O}(mn\alpha(n)\log^2(n))$.

For a general arrangement of line segments, an h-level is not necessarily connected but may have discontinuities where the level jumps from one segment to a segment directly above or below it. This happens if another segment starts or ends at a point below the current level. The complexity of an h-level is therefore defined as the number of vertices of the level plus the number of discontinuities (see Agarwal et al., 1998b).

Lemma 5.19 ▶ Continuity

The (n-k)-levels are piecewise linear continuous functions.

Proof. The discontinuities of general h-levels can not occur for (n-k)-levels based on L because the distance functions are piecewise linear and thus in every endpoint of one segment

another segment starts. By the definition of an (n-k)-level, one of this adjacent segments must belong to the (n-k)-level. ∎

As for the upper envelope, the question concerning a tight upper bound on the number of vertices (and with this on the number of the local minima of the (n-k)-level) is an open problem in combinatorial geometry. Some estimates on the complexity are known but it is conjectured that the best possible upper bound must be smaller (see, for example, Pach and Sharir, 2009). A proofed upper bound can, for example, be found in Agarwal et al. (1998b). However, the best known upper bound is $\mathcal{O}(q^{\frac{4}{3}})$, given by Dey (1997). This is the number of all vertices of the h-level. However, only the local minima are needed for the finite dominating set. The size of this set will be thus much smaller in general. The local minima can be filtered out easily by comparing their function values given by the second components of the vertices. With $q \leq 2n$ an upper bound of $\mathcal{O}(mn^{\frac{4}{3}})$ on the size of the set Cand2 can be obtained.

Algorithm 7 Solving k-max problems on graphs using Cand2

Input: Graph $G = (V, E)$ with $w_i > 0 \; \forall \; i = 1, \ldots, n,, \; k \in \{1, \ldots, n-1\}$.

1: **for all** $e_{ab} \in E$ **do**

2: Transform the arrangement L_{ab} into general position.

3: Determine the (n-k)-level $F_{ab}^{n-k}(x)$ of L_{ab}.

4: Compute the set of local minima

$$X_{ab}^{g} := \underset{x=(e_{ab},t)}{\arg\min} \; F_{ab}^{n-k}(x).$$

5: Transform the elements of X_{ab}^{g} back to the set X_{ab}.

6: Determine the finite dominating set

$$\text{Cand2} := \bigcup_{\substack{e_{ab} \in E \\ a,b \in \{1,\ldots,n\}}} X_{ab}.$$

7: Compute the set of points $\mathcal{X} := \{x^* : x_2^* = \min_{x \in \text{Cand2}} x_2\}$.

Output: Set of optimal solutions $\mathcal{X}$ with $z^* = x_2^*$.

Algorithm 7 summarizes the above ideas for solving the k-max problem on a graph based on the finite dominating set Cand2. The overall complexity using the set Cand2 to derive all optimal solutions consists of the following parts: For one edge e_{ab}, the transformation of L_{ab} to L_{ab}^{g} (in general position) can be implemented in at most $\mathcal{O}(n)$ time because only a small constant has to be added to the slope of each line segment. The computation of the (n-k)-level in L_{ab}^{g} takes $\mathcal{O}(n\alpha(n)\log^2(n))$ time as described above. Filtering out the

local minima from all vertices of the (n-k)-level needs $\mathcal{O}(n^{\frac{4}{3}})$, which corresponds to the number of vertices of the level. Only one comparison of the x_2-values has to be made for every candidate. Every candidate can be identified with the corresponding candidate point in the original arrangement by one additional list traversal on e_{ab}, which can be implemented together with the comparison of the respective objective values. Repeating this for all m edges of G leads to a complexity of $\mathcal{O}(mn^{\frac{4}{3}})$ since $\alpha(n) \ll \log(n)$ for n tending to infinity. Finally, the candidates with the smallest x_2-values have to be found; this can be implemented in $\mathcal{O}(mn^{\frac{4}{3}})$. Overall, a complexity of $\mathcal{O}(mn^{\frac{4}{3}})$ time for Algorithm 7 is obtained. Note that the approach can be implemented easily in parallel since the computation on a fixed edge does not depend on the results of the computation on the other edges. This leads to a clear acceleration of the algorithm in practice.

The finite dominating set based on h-levels has some properties depending on the parameter k. To clarify this, the notation Cand2(k) is used in the following.

Theorem 5.20 ▶ Partition of the FDS Cand2 with respect to levels
Let L_{ab}^{g} be an arrangement of weighted distance functions $d^w(v_i, x)$, $i = 1, \ldots, n$, over e_{ab} in general position. Then the finite dominating sets Cand2(k) for (1kMG) for different values of k, $1 \leq k \leq n-1$, are disjoint, i.e.

$$\text{Cand2}(r) \cap \text{Cand2}(s) = \emptyset \quad \text{for all} \quad r, s \in \{1, \ldots, n-1\},\ r \neq s.$$

Proof. Let $1 \leq k \leq n-1$ and let Cand2(k) be the finite dominating set based on the (n-k)-level $F_{ab}^{n-k}(x)$ of the arrangement L_{ab}^{g}. The (n-k)-level and the (n-k-1)-level are consecutive in a top-down view. Since L_{ab}^{g} is in general position, $F_{ab}^{n-k}(x)$ and $F_{ab}^{n-k-1}(x)$ are edge disjoint but may overlap in some of their vertices.

The following discussion is based on Dey (1997). Denote the set of vertices of L_{ab}^{g} with exactly $n-k$ lines below them as S_{n-k}. Consequently, the set of vertices in the (n-k)-level is

$$S_{n-k} \cup S_{n-k-1}.$$

The corresponding level $F_{ab}^{n-k}(x)$ makes a "left turn" in each vertex $v \in S_{n-k-1}$ and a "right turn" in $v \in S_{n-k}$, see Figure 5.6. This means that the local minima of the level are elements of S_{n-k-1} whereas S_{n-k} contains the local maxima. Since

$$S_{n-k} \cap S_{n-k-1} = \emptyset,$$

the local minima of consecutive levels are disjoint. Obviously, non consecutive levels can not have common vertices. As a consequence, the related finite dominating sets are also disjoint. ■

With the knowledge that a local minimum belongs to the finite dominating set Cand2(k) for a fixed k, this point can be excluded for other values of k. Conversely, for a given candidate point it is not immediately clear how the corresponding level can be identified. Lemma 5.21 partially answers this question.

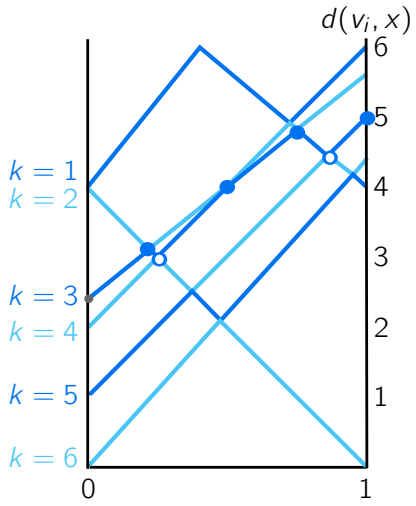


Fig. 5.6: Edge disjoint levels of L_{34}^{g} for different values of k. Vertices of $F_{34}^{3}(x)$ in S_3 are indicated by filled circles, vertices in S_2 by empty circles.

Lemma 5.21 ▶ Identification of the FDS Cand2 depending on k

Let σ be a permutation such that condition (4.1) is satisfied and let $x \in EQ$ be an equilibrium point with $d^w(v_{\sigma(s)}, x) = d^w(v_{\sigma(s+1)}, x)$ for some $s \in \{1, \ldots, n\}$. Then it holds that

$$x \in \text{Cand2}(s) \quad \dot{\vee} \quad x \in \text{Cand2}(s+1).$$

Proof. Let $x \in e_{ab}$ be an equilibrium point of G and let $\sigma \in \Sigma(x)$. i.e., it holds that

$$d^w(v_{\sigma(1)}, x) \geq \ldots \geq d^w(v_{\sigma(k)}, x) \geq d^w(v_{\sigma(k+1)}, x) \geq \ldots \geq d^w(v_{\sigma(n)}, x).$$

With $x \in EQ$ it follows by assumption that

$$d^w(v_{\sigma(s)}, x) = d^w(v_{\sigma(s+1)}, x) \quad \text{for some} \quad s \in \{1, \ldots, n\}.$$

This means that x has equal weighted distance from $v_{\sigma(s)}$ and $v_{\sigma(s+1)}$, and therefore it is a vertex of the levels $F_{ab}^{n-s}(x)$ and $F_{ab}^{n-s-1}(x)$. Consequently, either

$$x \in S_{n-s} \text{ or } x \in S_{n-s-1}.$$

However, it can not be decided whether x is a local minimum of $F_{ab}^{n-s}(x)$ or of $F_{ab}^{n-s-1}(x)$ because the roles of s and $s+1$ could also be swapped and still satisfy the equation $d^w(v_{\sigma(s)}, x) = d^w(v_{\sigma(s+1)}, x)$. Applying Corollary 5.17 implies that x belongs to the finite dominating set for $k = s$ or for $k = s+1$, but not to both because of Theorem 5.20. ■

The simultaneous computation of all optimal solutions for all values of the parameter $k = 1, \ldots, n-1$ is not possible with the finite dominating set Cand2. It consists of the vertices of the (n-k)-level over each edge of G and depends thus on the exact value of k.

5.4 Special Cases

This section discusses two special cases of the general k-max problem on graphs. These problems have an easier structure and thus it is not necessary to compute the whole candidate sets described before. In both cases the underlying graph of the k-max problem is unweighted, i.e., the weights w_i of all nodes $v_i \in V$, $i = 1, \ldots, n$, are equal and thus normally set to $w_i = 1$. The first considered problem type is based on a general unweighted graph as defined at the beginning of Chapter 4. The second approach is based on path graphs.

5.4.1 Unweighted k-max Problems on General Graphs

The special case of node weights $w_i = 1$ for all nodes $v_i \in V$, $i = 1, \ldots, n$, can of course be solved using the algorithms based on the finite dominating sets Cand and Cand2. However, there is another more simple candidate set that can be established for the unweighted k-max problem. This set indirectly uses equilibrium points without the need to compute them.

W.l.o.g. the length of the edges in G are assumed to be integer in this section, i.e. it holds that $l_j \in \mathbb{N}$ with $l_j \neq 0$ for all $j = 1, \ldots, m$. The idea is to find a set that is an easy derivable superset of EQ such that it is guaranteed that all candidate points of Cand are also elements of the new set for $k < n$. The following lemma describes such a superset.

Lemma 5.22 ▶ FDS based on step length 0.5
Let $G = (V, E)$ be a graph with $w_i = 1$ for all $i = 1, \ldots, n$ and $l_j \in \mathbb{N}$ with $l_j \neq 0$ for all $j = 1, \ldots, m$. All optimal locations of the *1-k*-max problem can then be found in the set

$$\text{Cand3} = \bigcup_{\substack{e_{ab} \in E \\ a,b \in \{1,\ldots,n\}}} \left\{ x^\alpha : x^\alpha = \left(e_{ab}, \frac{\alpha}{2l_{ab}} \right),\ \alpha = 0, 1, 2, \ldots, 2l_{ab} \right\}.$$

Proof. Let p_{gh} be any path from a node $v_g \in V$ to another node $v_h \in V$ and let $l(p_{gh})$ be the length of the path given as the sum of its edge lengths. Moreover, let $x_\beta \in EQ_{gh}$ on this path be given by the pair $x_\beta = (p_{gh}, \beta)$ with $\beta \in [0, 1]$ (analogously to the definition of a point on an edge) such that

$$d(v_g, x_\beta) = \beta l(p_{gh}) \quad \text{and} \quad d(v_h, x_\beta) = l(p_{gh}) - \beta l(p_{gh}).$$

As x_β is an equilibrium point of v_g and v_h, the distances $d(v_g, x_\beta)$ and $d(v_h, x_\beta)$ have to be equal. Thus, it follows immediately that $\beta = \frac{1}{2}$ and x_β is exactly the midpoint of p_{gh}.

W.l.o.g. it is assumed now that x_β lies on an edge $e_{ab} \in p_{gh}$ with $a, b \in \{1, \ldots, n\}$. $l(p_{gh})$ is the sum of integer values l_j, $j = 1, \ldots, m$. Hence,

$$x_\beta = \left(p_{gh}, \frac{1}{2} \right) = \left(e_{ab}, \frac{\alpha}{2l_{ab}} \right) \text{ for one } \alpha \in \{0, 1, 2, \ldots, 2l_{ab}\}.$$

Therefore, $x_\beta \in$ Cand3. Since x_β is an arbitrary equilibrium point, it follows that all equilib-

rium points of G are elements of Cand3 and thus Cand3 $\supseteq$ Cand. Since all optimal solutions of the k-max problem lie in Cand (see Theorem 5.10), Cand3 is, as a superset of Cand, also a finite dominating set for the k-max problem. ∎

To derive the candidate points for this finite dominating set, every edge e_{ab} has to be divided into steps of length $\frac{1}{2}l_{ab}$, where v_a and v_b do also belong to the set with $\alpha = 0$ resp. $\alpha = 2l_{ab}$. The number of candidates is then given by

$$|\,\text{Cand3}\,| = \sum_{\substack{e_{ab}\in E \\ a,b\in\{1,\dots,n\}}} (2l_{ab} - 1) + n,$$

because the number of candidate points on an edge e_{ab} is $2l_{ab} + 1$. In order to not count the endnodes v_a and v_b of e_{ab} several times in case that they are incident to more than one edge, only the inner candidate points on e_{ab} are counted. This results in the given sum for the inner candidate points of all m edges of G. Afterwards, the number of vertices is added because they all do belong to the finite dominating set. To derive an upper bound on the size of the candidate set Cand3, define

$$L := \max_{\substack{e_{ab}\in E \\ a,b\in\{1,\dots,n\}}} l_{ab}.$$

Then it holds for the maximum number of candidate points that $|\,\text{Cand3}\,| \leq 2Lm - m + n$ and therefore $|\,\text{Cand3}\,| = \mathcal{O}(Lm + n)$.

This bound generally overestimates the number of candidates significantly, especially if the edge lengths are very different. Of course it is much easier to compute the candidate set Cand3 than the first two sets Cand and Cand2, but the disadvantage is the dependence on the length of the edges of the graph. Thus, in general, the size of Cand3 is much larger than the sizes of the other finite dominating sets and with the need to evaluate the objective function value in every candidate, it becomes more costly than the two approaches before: For one candidate point the distances to all existing facilities have to be computed and sorted. This leads to a complexity of $\mathcal{O}(n\log(n))$. Finally, the derivation of an optimal solution for the k-max problem takes $\mathcal{O}(Lmn\log(n) + n^2\log(n))$ time, with L being the maximum of all edge length. This approach for the special case of all nodes being weighted equally should just be applied to problems with a small number of short edges, otherwise the finite dominating sets Cand and Cand2 should be preferred.

Note that a generalisation to other weight sets would in general involve largely increased candidate sets. As an example consider only one edge of length $l_{ab} = 1$ and two adjacent nodes with different weights, i.e., $w_a = 1, w_b = 2$ or $w_b = 1, w_b = 3$ and so on.

5.4.2 Unweighted k-max Problems on Path Graphs

Path graphs are graphs with a particularly simple structure. Hence, in this case, an easy procedure with low complexity to solve the problem can be given. A path graph $P = (V, E)$ is a tree that has two nodes with vertex degree one (the leaves) and $n-2$ nodes with vertex

degree two (inner nodes). Such a graph can therefore be drawn as a single straight line on which all of its vertices and edges lie such that its vertices can be listed in the order $v_1, v_2, \ldots, v_n$ and edges are of type

$$E = \{e_{i,i+1} = (v_i, v_{i+1}) : i = 1, \ldots, n-1\}.$$

In the following, it is assumed that the nodes and edges of the graph are indexed as specified above, and that the graph is unweighted, i.e., it holds $w_i = 1$ for all $i = 1, \ldots, n$.

The idea is to apply the result of Theorem 5.1 to path graphs such that only a small number of possibilities to choose the (not necessarily unique) optimal set of outliers V^*_{k-1} remains. This leads to a strong property for the center defining nodes and an easy procedure to solve the k-max problem can be derived.

Corollary 5.23 ▶ Position of outliers on path graphs

Let $P = (V, E)$ be an unweighted path graph and let $x \in A(P)$ be a feasible solution of the k-max problem on P with its corresponding set of outliers V_{k-1}. It holds: All outliers $v_i \in V_{k-1}$ are connected to one of the leaf nodes v_1 or v_n by a path

$$P(v_i, v_1) = (v_1, v_2, \ldots, v_{i-1}, v_i) \quad \text{with} \quad \{v_1, \ldots, v_i\} \subseteq V_{k-1}$$

or a path

$$P(v_i, v_n) = (v_i, v_{i+1}, \ldots, v_{n-1}, v_n) \quad \text{with} \quad \{v_{i+1}, \ldots, v_n\} \subseteq V_{k-1}.$$

Equivalently, the center defining nodes $V \setminus V_{k-1}$ form a subpath $P_j = P(v_j, v_{j+n-k})$ of P with starting node $v_j \in \{v_1, \ldots, v_k\}$.

Proof. Follows directly with Theorem 5.1 and the structure of a path graph. ∎

Following Theorem 5.1, the center defining nodes have to be connected and therefore have to form a subpath of the path graph P. As a consequence, the center point of the center defining path is the optimal location for the new facility. Hence, by evaluating all starting nodes $v_j \in \{v_1, \ldots, v_k\}$ of a possible center defining path $P(v_j, v_{j+n-k})$, all candidates for the optimal k-max solution are obtained, given by the center locations $x(P_j)$, $j = 1, \ldots, k$, of the respective paths. Since the center location of an unweighted path is unique, this leads to exactly k candidate locations. Compared to general graphs where all $\binom{n}{n-k+1}$ most contiguous sets have to be checked (see Algorithm 2), the number of these sets to be evaluated is much smaller here because only subsets $\mathcal{R}_{n-k+1}$ that form a connected subpath of P have to be considered.

Obviously, the subpath leading to the smallest center objective value defines the optimal new location. Since the center point of an unweighted path graph P_j is the midpoint of P_j, the center objective value is given by

$$z(P_j) = \frac{1}{2} \cdot \mathrm{dia}(P_j),$$

with $\mathrm{dia}(P_j)$, $j = 1, \ldots, k$, being the diameter of P_j (which, in the case of a path graph, equals the length of P). Therefore, the subpaths can easily be evaluated, i.e., it can be easily checked whether they are the optimal center defining paths by just comparing their diameters. The subpath with the smallest diameter defines the optimal set of center defining nodes.

Figure 5.7 gives an example of a k-max problem on a path graph for $k = 4$. The first subfigure shows the original path graph P. Below are the four possible center defining paths (black) and their corresponding outliers (grey). The optimal solution of the problem is $x^* = x(P_3)$ with objective function value $z^* = 1.5$ since P_3 has $\mathrm{dia}(P_3) = 3$ and is thus the shortest possible center defining path.

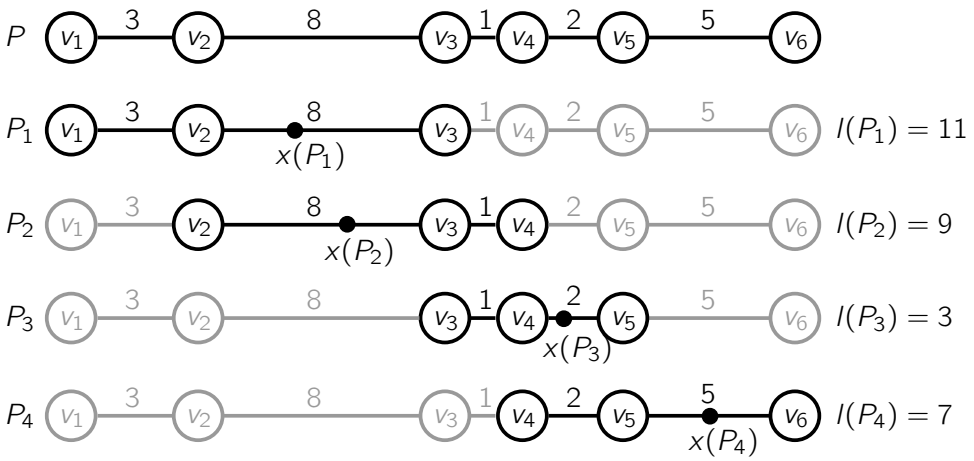


Fig. 5.7: A path graph P with $n = 6$ and $k = 4$. The subpaths P_j with $v_j \in \{v_j, v_{j+n-k}\}$ are shown in black with their corresponding center points $x(P_j)$, $j = 1, \ldots, k$.

For the evaluation of the subpaths, their diameters have to be computed. By updating the diameter $\mathrm{dia}(P_{j-1})$ of the subpath $P_{j-1} = (v_{j-1}, v_{j-1+n-k})$, $j = 2, \ldots, k$, the diameter of $P_j = (v_j, v_{j-n-k})$ can be determined easily with

$$\mathrm{dia}(P_j) = \mathrm{dia}(P_{j-1}) - l_{j-1} + l_{j+n-k-1},$$

since P_j is obtained from P_{j-1} by deleting edge e_{j-1} and node v_{j-1} and adding edge $e_{j+n-k-1}$ and node v_{j+n-k}. Algorithm 8 summarises this approach to solve the k-max problem on an unweighted path graph.

The complexity of this algorithm is $\mathcal{O}(n)$ because there are exactly k graphs to be evaluated for being the center defining subpaths. Since $k \leq n$, the number of these subpaths is bounded by n. The individual steps of Algorithm 8 can be implemented in constant time. Thus, the problem on an unweighted path graph can be solved much more efficiently than the problem on general graphs. This holds also if the solutions for all values of k have to be determined.

Since for every value of k there are k subpaths to be analysed, there are in total

$$\sum_{k=1}^{n} \frac{n(n+1)}{2} = \mathcal{O}(n^2)$$

subpaths for all values of k.

Algorithm 8 Solving the 1-k-max problem on an unweighted path graph

Input: Graph $G = (V, E)$, $k \in \{1, \ldots, n-1\}$

Set path $P_1 := (v_1, \ldots, v_{1+n-k})$
Set $\text{dia}(P_1) := \sum_{i=1}^{n-k} l_i$ and dia_min $:= \text{dia}(P_1)$
Set $x(P_1)$ as the midpoint of P_1 and $\mathcal{X} := \{x(P_1)\}$
for $j = 2, \ldots, k$ **do**
    Set path $P_j := (v_j, \ldots, v_{j+n-k})$
    Set $\text{dia}(P_j) := \text{dia}(P_{j-1}) - l_{j-1} + l_{j+n-k-1}$
    Set $x(P_j)$ as the midpoint of P_j
    if $\text{dia}(P_j) < \text{dia}(P_{j-1})$ **then** ▷ New currently best solution
        dia_min $:= \text{dia}(P_j)$
        $\mathcal{X} := \{x(P_j)\}$
    else if $\text{dia}(P_j) = \text{dia}(P_{j-1})$ **then** ▷ Additional currently best solution
        $\mathcal{X} := \mathcal{X} \cup \{x(P_j)\}$

Output: Set $\mathcal{X}$ of optimal solutions and $z^* := \frac{1}{2} \cdot$ dia_min.

Note that an approach that recursively determines outliers in $k-1$ steps does in general not lead to an optimal solution of the 1-k-max problem on path graphs. Consider, for example, the following recursive strategy: In each step the leaf node and its incident edge is deleted that shortens the diameter of the graph the most. This greedy strategy may lead to a very unsatisfying solution, i.e., after some iterations the optimal solution can not be reached any more. This is illustrated in Example 5.24.

Example 5.24. *Figure 5.8 gives an example for the situation described above. The first subfigure shows the initial graph P with five nodes. A k-max problem with $k = 3$ is considered. In the recursive approach described above, the first step would be to choose node v_5 as an outlier and to delete it because $l_4 = 2 > 1 = l_1$. In the second step, v_4 is deleted for the same reason. The result is a subgraph with diameter 11 and set of outliers $V_{k-1} = \{v_4, v_5\}$. However, the optimal solution would be to choose nodes v_1 and v_3 as outliers because then the remaining graph has a diameter of 4. Hence, after the first step in the recursive approach, the optimal solution can not be reached any more.*

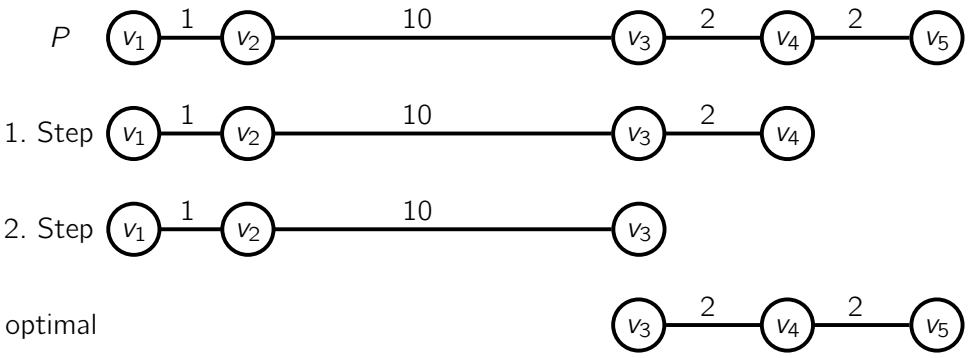


Fig. 5.8: Recursive approach that does not yield the correct optimal solution.

5.5 Conclusion

Two solution approaches for the k-max problem on graphs based on two different finite dominating sets were presented in this chapter. To conclude, a short comparison of the two finite dominating sets Cand and Cand2 for the k-max problem is made.

The first candidate set Cand is based on the set of equilibrium points of the graph which are given by all intersection points of the line arrangements over every edge of the graph. The cardinality of this finite dominating set is $\mathcal{O}(mn^2)$. The second candidate set Cand2 consists only of the intersection points of the line arrangement given by the considered level for the given value of the parameter k and hence $|\,\text{Cand2}\,| = \mathcal{O}(mn^{\frac{4}{3}})$. This difference in the theoretical size may be even larger than the worst case estimation suggests because the upper bound on the size of the number of vertices of an h-level is not tight and Cand2 has in general much fewer elements than Cand. However, the computational tests (see Chapter 7) have shown that the actual number of equilibrium points of a graph is in practice much smaller than it could be assumed from the theoretical upper bound. Hence, the difference in the size of the two finite dominating sets may in practice not be that important. Note that Cand2 is in general not a subset of Cand as Cand2 may also contain nodes that are not equilibrium points. A local minimum of an $(n\text{-}k)$-level may be attained in a node no matter if two distance functions intersect there.

The complexity of the solution approach based on Cand is bounded by $\mathcal{O}(m(n+s)\log(n))$ with $s \leq \binom{n}{2}$, whereas the algorithm to solve the problem with Cand2 needs at most $\mathcal{O}(mn^{\frac{4}{3}})$ time. Thus, it depends on the particular instance (the number of intersection points over the edges) which finite dominating set yields a better worst case bound for solving the k-max problem. However, an advantage of the set Cand is that the k-max problem can be solved for all values of $k = 1, \ldots, n-1$ simultaneously as the optimal solutions for all values of k belong to Cand. In contrast to that, the elements of Cand2 depend on a fixed value of k as the minima of the corresponding $(n\text{-}k)$-level have to be computed.

Furthermore, two special cases of the k-max problem were considered in this chapter. In both cases, the nodes are unweighted, i.e., $w_i = 1$ for all $i = 1, \ldots, n$. The first approach is for general graphs and computes a finite dominating set which contains the set Cand as

a subset. The computation of this candidate set is easy and fast. However, its cardinality depends on the number and the lengths of the edges and may therefore get very large.

The second case deals with path graphs which have an easy structure such that only a small number of "most contiguous sets" remain important to evaluate. Thus, an optimal solution of this special case can be found very easily.

Chapter 6

Algorithms for p-k-max Location Problems on Networks

In this chapter, the p-k-max location problem as defined in Definition 4.2 will be analysed, i.e., a set of $p \in \{1, \dots, n\}$ new facilities $X = \{x_1, \dots, x_p\}$ has to be located on the underlying network $G = (V, E)$ such that the kth largest distance for a $k \in \{1, \dots, n\}$ between any customer and its closest new facility is minimised:

$$\min_{X \subseteq A(G)} f_k(X) = \min_{X \subseteq A(G)} k\text{-max}(d^w(V, X)) = \min_{X \subseteq A(G)} w_{\sigma(k)} d^w(v_{\sigma(k)}, X),$$

where $\sigma \in \Sigma(X)$ is a permutation that sorts the elements of the vector $d^w(V, X) \in \mathbb{R}^n$ of weighted distances in non-increasing order. The number $p \in \{1, \dots, n\}$ of new facilities to locate is arbitrary but fixed, i.e., it does not grow with the problem size.

In the first section, a finite dominating set is derived and used as a basis for an algorithm to solve the p-k-max problem. The second section presents a recursive approach that generates the individual candidates for optimal solutions step by step. Each new facility of a candidate solution is located with respect to the already located facilities such that many possible locations can be excluded from the further analyses. These two approaches guarantee to find at least one optimal solutions of the p-k-max problem. The aim of the following section is to find alternative optimal solutions with other properties than the optimal solutions found in the two sections before. A local analysis on the edges of the graph is used to identify the linearity regions of the k-max function. Moreover, it is shown that this approach even yields all optimal solutions of the underlying problem. Another technique to generate alternative optimal solutions of the p-k-max problem that are not an element of the finite dominating set presented in the first section is given in the fourth section. The chapter is closed with a short comparison of the described approaches.

To get an intuition of the solutions of a p-k-max problem, Figure 6.1 shows an optimal solution $X^* = \{x_1^*, \dots, x_p^*\}$ of the 3-2-max problem on an unweighted graph G. The length of every edge corresponds to the Euclidean distance between its two endnodes of this edge in the planar embedding of the graph. The set of outliers w.r.t. X^* is given by V_{k-1}^*. As now p new facilities can be located, there are also p clusters, one defined by each new facility $x_1^*, \dots, x_p^*$. A cluster C_i consists of the customers that are covered with service by the corresponding new facility x_i with $i \in \{1, \dots, p\}$ and are therefore described by the set

$$C_i = \{v \in V \setminus V_{k-1} : d^w(v, x_i) \leq d^w(v, x_\ell) \; \forall \; \ell > i \; \wedge \; d^w(v, x_i) < d^w(v, x_\ell) \; \forall \; \ell < i\}$$

for $i \in \{1, \ldots, p\}$ and for all $\ell = 1, \ldots, p$ with $\ell \neq i$. Note that, if an existing facility is of equal distance from two or more new facilities, it can be assigned w.l.o.g. to the new facility with the smallest index.

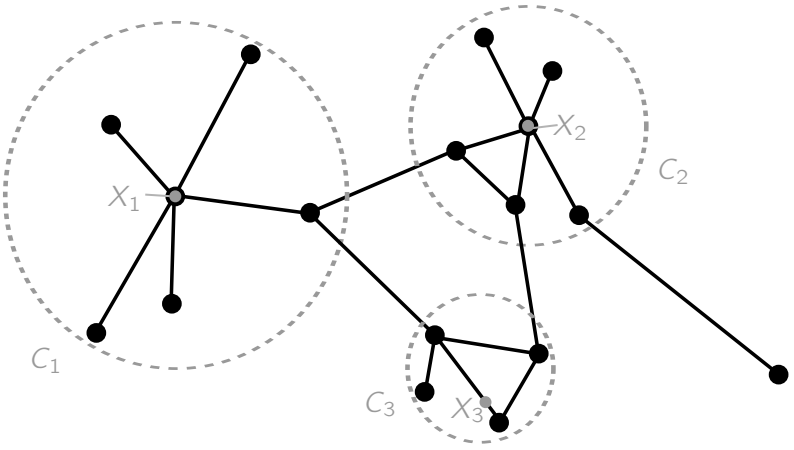


Fig. 6.1: An optimal solution of the 3-2-max problem on G with the corresponding clusters

The optimal k-max value z^* is the largest weighted distance between a customer provided with service and its closest new facility. In the following, a new facility x_{i^*}, $i^* \in \{1, \ldots, p\}$, that yields the weighted distance defining the k-max value in the corresponding solution X^*, i.e., for which

$$d^w(X^*, V \setminus V^*_{k-1}) = d^w(x_{i^*}, C_{i^*}) = z^*$$

holds, is called an *objective function value defining facility* resp. a *facility that determines the objective function value*. Note that there can in general be more than just one objective function value defining facility in a solution. In the example of Figure 6.1, the objective function value defining facility is obviously x_1. The maximum distance between a new facility and its most distant customer is also called the *radius* of the cluster. An important observation is that the k-max value gives the radius of the largest cluster defined by the solution X^*. The other clusters can at most be as large as the cluster of the objective function value defining facility, otherwise this would imply a larger objective function value. Note that not all new facilities are the center point of the nodes allocated to it.

Some combinations of values for the parameters k and p can be excluded from a further analysis because a set of optimal solutions of the p-k-max problem is known without any calculations. The following lemma gives the exact number of optimal solutions and their characterisation for $k = n - p + 1$.

Lemma 6.1 ▶ Case $k = n - p + 1$

For $k = n - p + 1$ the p-k-max problem has $\binom{n}{p}$ optimal solutions of the form

$$\mathcal{X}^* = \big\{\{x_1, \ldots, x_p\} : x_1, \ldots, x_p \in V \text{ with } x_i \neq x_j \ \forall\ i, j = 1, \ldots, p\big\}.$$

Proof. Let $p \in \{1, \ldots, n\}$ be arbitrary but fixed and let $k = n - p + 1$. With the definition of k it holds that $p = n - k + 1$, i.e., the number of facilities to locate equals the number of customers that have to be covered with service. Locating all new facilities $X = \{x_1, \ldots, x_p\}$ in p arbitrary but pairwise different nodes $v_{i_1}, \ldots, v_{i_p} \in V$ leads to

$$d^w(v_{i_j}, X) = 0 \quad \text{for all} \quad j = 1, \ldots, p.$$

Obviously, all other components $d^w(v_a, X)$ with $v_a \in V$ and $a \notin \{i_1, \ldots . i_p\}$ are strictly positive as all node weights and all edge lengths are strictly positive. Thus, the vector of distances $d^w_\sigma(V, X)$, which is the vector $d^w(V, X)$ sorted by the permutation $\sigma \in \Sigma(X)$, is of the form

$$\begin{aligned} d^w_\sigma(V, X) &= (d^w(v_{\sigma(1)}, X), \ldots, d^w(v_{\sigma(n-p),X}), d^w(v_{\sigma(n-p+1)}, X), \ldots, d^w(v_{\sigma(n)}, X))^\top \\ &= (d^w(v_{\sigma(1)}, X), \ldots, d^w(v_{\sigma(n-p),X}), 0, \ldots, 0)^\top. \end{aligned}$$

As the objective function value of this solution is $z^* = d^w(v_{\sigma(k)}, X) = d^w(v_{\sigma(n-p+1)}, X) = 0$ and the objective function value of a p-k-max problem is bounded from below by 0, X has to be an optimal solution.

As the p new facilities can be placed in p out of n nodes, there are $\binom{n}{p}$ alternative optimal solutions. Moreover, a distance of 0 can only be realised by locating each new facility in a node. Hence, there are no other optimal solutions that do not locate all new facilities in nodes. ∎

The following lemma shows that the p-k-max problem for a parameter k that is strictly larger than $n - p + 1$ has infinitely many optimal solutions and that they are easy to identify.

Lemma 6.2 ▶ Case $k \in \{n - p + 2, \ldots, n\}$

Let $p \in \{2, \ldots, n\}$. For $k \in \{n - p + 2, \ldots, n\}$ the p-k-max problem has infinitely many optimal solutions. The set of optimal solutions is of the form

$$\begin{aligned} \mathcal{X}^* = \big\{ \{x_1^*, \ldots, x_p^*\} : x_1^*, \ldots, x_{n-k+1}^* \in V \text{ with } x_i \neq x_j \ \forall\ i, j = 1, \ldots, n-k+1 \\ \text{and } x_{n-k+2}^*, \ldots, x_p^* \in A(G) \big\}. \end{aligned}$$

Proof. Let $p \in \{2, \ldots, n\}$ be arbitrary but fixed and let $k \in \{n - p + 2, \ldots, n\}$. For a parameter $s \in \{1, \ldots, p - 1\}$ it holds that $k = n - p + 1 + s \Leftrightarrow p - s = n - k + 1$. That means that s more new facilities can be located than customers have to be covered with service. Thus, locating a subset $\bar{X} = \{x_1, \ldots, x_{p-s}\}$ of the new facilities $X = \{x_1, \ldots, x_p\}$ in $p - s$ arbitrary but pairwise different nodes $v_{i_1}, \ldots, v_{i_{p-s}} \in V$ leads to

$$d^w(v_{i_j}, \bar{X}) = 0 \quad \text{for all} \quad j = 1, \ldots, p - s.$$

By locating the remaining s new facilities $x_{p-s+1}, \ldots, x_p$ arbitrarily on $A(G)$, all other components $d^w(v_a, X)$ with $v_a \in V$ and $a \notin \{i_1, \ldots . i_{p-s}\}$ are non-negative as all node weights and all edge lengths are strictly positive. Thus, the vector of distances $d^w_\sigma(V, X)$, which is

the vector $d^w(V,X)$ sorted by the permutation $\sigma \in \Sigma(X)$, is of the form

$$\begin{aligned} d^w_\sigma(V,X) &= (d^w(v_{\sigma(1)},X),\dots,d^w(v_{\sigma(n-p+s),X}),d^w(v_{\sigma(n-p+s+1)},X),\dots,d^w(v_{\sigma(n)},X))^\top \\ &= (d^w(v_{\sigma(1)},X),\dots,d^w(v_{\sigma(n-p+s),X}),0,\dots,0)^\top. \end{aligned}$$

As the objective function value of this solution is

$$z^* = d^w(v_{\sigma(k)},X) = d^w(v_{\sigma(n-p+s+1)},X) = 0$$

and the objective function value of a p-k-max problem is bounded from below by 0, X has to be an optimal solution. Since $A(G)$ contains infinitely many points to locate $x_{n-p+s},\dots,x_p$, there are infinitely many optimal solutions of the p-k-max problem on G. ■

As a consequence of Lemma 6.1 and Lemma 6.2, it is assumed in the following (unless it is stated otherwise) that $n \geq 2$ and the values of the parameters k and p satisfy the relation $k \leq n-p$. Otherwise, the optimal solutions can be obtained using the above results.

6.1 Finite Dominating Set Based on Equilibrium Points

As already stated at the beginning of Section 5.2, it is very helpful to know a finite dominating set of the underlying problem because the set of possible optimal solutions is reduced to a finite number of candidates. Often, this is a good starting point so create more efficient solution approaches than to simply enumerate all these candidates. Thus, a finite dominating set for the p-k-max problem is derived in this section. The result for the finite dominating set based on equilibrium points of the *1*-k-max problem can not directly be transferred to the p-k-max case. However, a similar statement holds. Again, the set of candidates for an optimal solution is based on equilibrium points. Theorem 6.3 shows that at least one optimal solution of the given problem lies in the below defined finite dominating set Cand4.

Theorem 6.3 ▶ FDS for the p-k-max problem

Let $p \in \{1,\dots,n\}$ and $k \in \{1,\dots,n-p\}$. At least one optimal solution $X^* = \{x_1^*,\dots,x_p^*\}$ of the p-k-max problem, where w.l.o.g x_1^* determines the optimal objective function value z^*, can be found in the set

$$\begin{aligned} \text{Cand4} &= EQ \times (EQ \cup V)^{p-1} \\ &= \big\{\{x_1,\dots,x_p\} : x_1 \in EQ,\ x_2,\dots,x_p \in EQ \cup V\big\}. \end{aligned}$$

Proof. Let $\tilde{X} = \{\tilde{x}_1,\dots,\tilde{x}_p\}$ with $\tilde{x}_1,\dots,\tilde{x}_p \in A(G)$ be optimal locations with optimal objective value $\tilde{z}$ (an optimal solution exists according to Theorem 4.6). Let

$$V_\ell := \{v_i \in V : d^w(v_i,\tilde{x}_\ell) \leq d^w(v_i,\tilde{x}_g)\ \forall\, g > \ell \wedge d^w(v_i,\tilde{x}_\ell) < d^w(v_i,\tilde{x}_g)\ \forall\, g < \ell\}$$

for all $\ell \in \{1,\dots,p\}$ define an optimal allocation of the existing facilities to the new facilities $\tilde{x}_1,\dots,\tilde{x}_p$. Note that if an existing facility is of equal distance from two or more new facilities,

it is assigned w.l.o.g. to the new facility with the smallest index. Moreover, let $\sigma \in \Sigma(\tilde{X})$, i.e.,

$$\min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(1)}, \tilde{x}_\ell) \geq \min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(2)}, \tilde{x}_\ell) \geq \dots \geq \min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(n)}, \tilde{x}_\ell).$$

Then

$$\min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(k)}, \tilde{x}_\ell) = \tilde{z} \quad \text{and} \quad \min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(i)}, \tilde{x}_\ell) \geq \tilde{z} \;\; \forall\, i < k. \tag{6.1}$$

Suppose w.l.o.g. that the minimum in (6.1) is attained in $\tilde{x}_1$, i.e., $\tilde{z} = d^w(v_{\sigma(k)}, \tilde{x}_1)$, and let

$$k_\ell := |\{v_{\sigma(i)} \in V_\ell \,:\, i < k\}| \quad \text{for all} \quad \ell \in \{1, \dots, p\}$$

be the number of outliers in V_ℓ. Clearly, $\sum_{\ell=1}^{p} k_\ell + 1 = k$. An illustration of the situation is given in Figure 6.2.

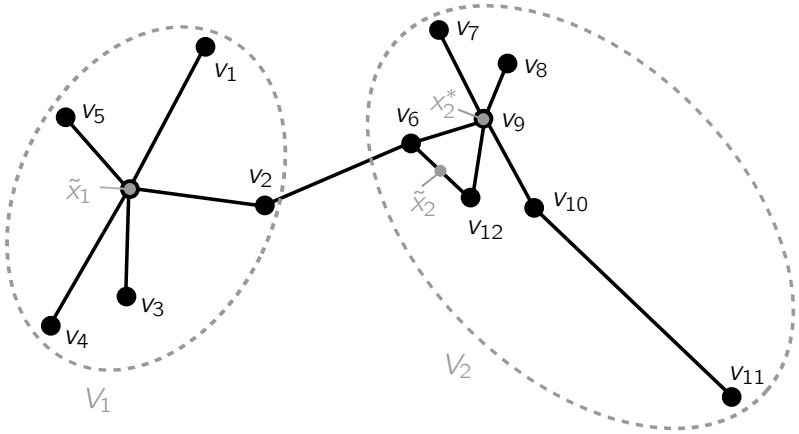


Fig. 6.2: Graph G with sets $V_1 = \{v_1, \dots, v_6\}$ and $V_2 = \{v_7, \dots, v_{12}\}$ for the optimal solution $\tilde{X} = \{\tilde{x}_1, \tilde{x}_2\}$ of the 2-2-max problem. Here, $k_1 = 0$ and $k_2 = 1$ as node v_{11} is the outlier of this solution. Solving the 1-1-max problem on V_1 leads to $\tilde{x}_1$ again, the optimal solution of the 1-2-max problem is $x_2^* \in EQ_{7,10}$.

Now an alternative optimal solution $x_1^*, \dots, x_p^* \in A(G)$ is constructed such that $x_1^* \in EQ$ and $x_2^*, \dots, x_p^* \in EQ \cup V$ as follows: Let $x_1^* \in A(G)$ be selected such that x_1^* is an optimal location for the 1-(k_1+1)-max problem with existing facilities V_1 in G (all nodes in $V \setminus V_1$ have weight 0 in this problem, see Remark 4.5). Note that the problem is solvable because of $k_1 \leq |V_1|$ due to the definition of k_1.

The 1-(k_1+1)-max problem has an optimal solution in EQ for all $1 \leq k_1 \leq |V_1| - 2$ and in V only if $k_1 = |V_1| - 1$ (see Theorem 5.10). In the latter case all nodes in V_1 except of one have to be outliers. Thus, x_1^* is located in a node $v^* \in V_1$ and yields $z_1^* = 0$ in this case. With the same argument, $\tilde{x}_1$ also had to be located in a node of V_1 and hence $\tilde{z} = 0$.

It follows that

$$\min\left\{d^w(v_{\sigma(i)}, x_1^*), d^w(v_{\sigma(i)}, \tilde{x}_2), \ldots, d^w(v_{\sigma(i)}, \tilde{x}_p)\right\} = 0 \quad \text{for all} \quad i \geq k.$$

As x_1^* is already located in a node, $\tilde{x}_2, \ldots, \tilde{x}_p$ have to bring all other $n - k \geq p$ distances to 0 which is not possible due to the positive edge and node weights. As a consequence, the case $k_1 = |V_1| - 1$ can not occur and $x_1^* \in EQ$. Since $\tilde{x}_1$ is also feasible for this problem and has objective value $\tilde{z}$, it is known that the optimal objective value z_1^* of this problem satisfies $z_1^* \leq \tilde{z}$.

Similarly, let $x_\ell^* \in EQ \cup V$, $\ell \in \{2, \ldots, p\}$, be selected such that x_ℓ^* is an optimal location for the 1-$(k_\ell+1)$-max problem with existing facilities V_ℓ in G (all nodes in $V \setminus V_\ell$ have weight 0 in this problem). Note that the problem is solvable because of $k_\ell \leq |V_\ell|$ due to the definition of k_ℓ. As $\tilde{x}_\ell$ is also feasible for this problem and has objective value of at most $\tilde{z}$, it is known that the optimal objective value z_ℓ^* of this problem satisfies $z_\ell^* \leq \tilde{z}$.

Consequently, all existing facilities in V can be allocated to $x_1^*, \ldots, x_p^*$ such that the kth largest distance occurring in this allocation is at most $\max_{\ell \in \{1,\ldots,p\}} z_\ell^* \leq \tilde{z}$. Since distances can not be larger in an optimal allocation, it can be concluded that $X^* = \{x_1^*, \ldots, x_p^*\}$ is an alternative feasible solution satisfying $k\text{-max}(d^w(v_i, X^*)) \leq \tilde{z}$. Since $\tilde{z}$ is optimal, this inequality must hold with equality, and $\{x_1^*, \ldots, x_p^*\}$ is indeed an alternative optimal solution. ∎

As stated in Remark 4.19, the total number of equilibrium points of G is bounded by $\mathcal{O}(mn^2)$. This leads to a size of $\mathcal{O}(m^p n^{2p})$ of the finite dominating set Cand4 since all sets $\{x_1, \ldots, x_p\}$ with $x_1 \in EQ$ and $x_2, \ldots, x_p \in EQ \cup V$ have to be considered as candidate solutions of the p-k-max problem. As a consequence of Theorem 6.3, an optimal solution of the p-k-max problem on graphs can be found by evaluating the objective function in all these candidate points of Cand4.

Note that the set of optimal solutions $\mathcal{X}_{\text{Cand4}}$ found in this way does in general not contain all optimal solutions of the underlying p-k-max problem. Algorithm 9 has a worst case complexity of $\mathcal{O}(m^p n^{2p+1} \log(n))$.

This can be seen by analysing the individual steps. The derivation of the equilibrium points with the algorithm of Bentley and Ottmann (1979) requires $\mathcal{O}(m(n+s)\log(n))$ time with $s \leq 2\binom{n}{2}$ (see Section 4.4). Finding and filtering all redundant candidates out of a set of size $\mathcal{O}(m^p n^{2p})$ takes $\mathcal{O}(m^p n^{2p} p \log(mn))$. Afterwards, all candidates $X = \{x_1, \ldots, x_p\} \in$ Cand4 have to be evaluated in $\mathcal{O}(m^p n^{2p+1}(p + \log(n)))$ (see Algorithm 4). Thus, the overall complexity is bounded by $\mathcal{O}(m^p n^{2p}(p \log(mn) + np + n \log(n)))$. As $m \leq n^2$ and therefore $\log(mn) \leq n$, this expression can be simplified to $\mathcal{O}(m^p n^{2p+1}(p + \log(n)))$ and as p is assumed to be arbitrary but fixed, the above mentioned result holds.

Note that the filtering of the set EQ for multiply defined equilibrium points does not improve the theoretical worst case bound of the algorithm. However, this filtering step yields in general significant improvements w.r.t. the computation time as the number of candidates to be tested may be much smaller.

Note moreover that the complexity of Algorithm 9 can not be improved by extracting the information about the permutations and the distances between the nodes and the candidates from the derivation of the equilibrium points as it can be done for the *1-k*-max problem. Since the distances to every node have to be computed from all new facilities, the permutation σ is in general different from the one given by the ordering of the line segments in one point $x \in EQ$ computed by the algorithm of Bentley and Ottmann (1979). However, the computations can be done in parallel easily: The equilibrium points on a fixed edge do not depend on the equilibrium points of the other edge and the single candidate solutions can be evaluated independently from the other candidates. This leads to a significant acceleration of the algorithm in practice.

Algorithm 9 *p-k*-max problems on graphs using Cand4

Input: Graph $G = (V, E)$ with $w_i > 0 \; \forall \; i = 1, \ldots, n$, $p \in \{2, \ldots, n\}$, $k \in \{1, \ldots, n-p\}$.

1: Determine the set EQ of all equilibrium points of G.

2: Determine the set Cand4 $:= EQ \times (EQ \cup V)^{p-1}$.

3: Filter out redundant candidates of Cand4.

4: Evaluate the k-max function in all candidates $X \in$ Cand4.

Output: Set of optimal solutions $\mathcal{X}_{\text{Cand4}} := \{X^* = \{x_1^*, \ldots, x_p^*\} : f_k(X^*) = \min_{X \in \text{Cand4}} f_k(X)\}$ with optimal objective function value $z^* := d^w(v_{\sigma(k)}, X^*)$.

Example 6.4. *Let $G = (V, E)$ be the example graph with node weights and edge length given in Figure 4.5. Evaluating all candidates of the finite dominating set* Cand4 *leads to the following four optimal solutions of the* 2-1-max *problem:*

$$
\begin{aligned}
x_1^1 &= \left(e_{34}, \frac{1}{3}\right) \quad x_2^1 = (e_{12}, 0)\,, \\
x_1^2 &= \left(e_{34}, \frac{1}{3}\right) \quad x_2^2 = \left(e_{12}, \frac{1}{4}\right), \\
x_1^3 &= \left(e_{34}, \frac{1}{3}\right) \quad x_2^3 = \left(e_{12}, \frac{1}{3}\right), \\
x_1^4 &= \left(e_{34}, \frac{1}{3}\right) \quad x_2^4 = \left(e_{15}, \frac{1}{3}\right),
\end{aligned}
$$

with optimal objective function value $z^* = \frac{4}{3}$. Note that the first facility is the same in all four solutions. The solutions are illustrated in Figure 6.3.

Remark 6.5. *The above presented approach yields all optimal solutions of the p-k-*max *problem in which all new facilities lie in the set $EQ \times (EQ \cup V)^{p-1}$. It is also possible to consider only candidate solutions in which the individual new facilities are located in pairwise different positions in $A(G)$ such that no two new facilities coincide. In this case, it is only guaranteed to find at least one, but not to find all optimal solutions of the set* Cand4 *since,*

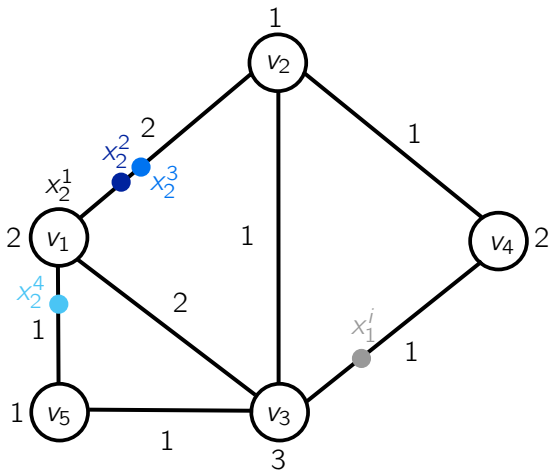


Fig. 6.3: Optimal solutions of the 2-1-max problem. The second new facility x_2^i is the same in all four solutions for $i = 1, \ldots, 4$.

particularly in the case that not all new facilities are needed to realise the optimal objective function value, some of the new facilities may be placed in arbitrary locations and thus also at the same location as some other new facilities.

Algorithm 9 can be extended easily to simultaneously compute an optimal solution for all possible values of the parameter k with $1 \leq k \leq n - p$. Once the candidate set and the corresponding sorted vectors of distances are computed for each candidate point, the evaluation of the candidates can be realised in an extra step. The elements of the distance vector of each candidate are associated to the corresponding values of k and compared to the currently best objective function value with respect to k. Since $k \leq n$, the extension does not lead to a higher worst case complexity bound but generates a lot more information about the trade-off (see Section 2.3) between the objective function value, i.e., the required coverage radius, and the value of k.

6.2 A Recursive Approach

The evaluation of the candidates in the finite dominating set Cand4 (see Algorithm 9) can be made more efficient for practical usage if not all candidates $X = \{x_1, \ldots, x_p\}$ with $x_1 \in EQ$ and $x_2, \ldots, x_p \in EQ \cup V$ are checked for their objective function value. So in the following, an algorithm will be developed which generates every candidate recursively. Thus, for each individual new facility it can be decided which positions with respect to the already located facilities can potentially lead to an optimal solution and which can not. In this way, the majority of elements of Cand4 do not have to be considered and the set of candidates that has to be evaluated is in general significantly smaller.

It is known from Theorem 6.3 that the new facility that determines the optimal objective function value z^* lies in an equilibrium point. As a consequence, the optimal objective

function value z^* of the p-k-max problem is known to be in the set

$$Z = \{d^w(v_a, y) : y \in EQ_{ab} \text{ with } a, b \in \{1, \ldots, n\}\}.$$

In the following, the elements of the set Z are called z_i and are assumed to be indexed such that $z_1 > z_2 > \ldots > z_{|Z|}$ for $i = 1, \ldots, |Z|$. Note that the cardinality of the set Z is usually smaller than the cardinality of the set EQ. With every combination $a, b \in \{1, \ldots, n\}$, $a \neq b$, a set EQ_{ab} and with it possible values of the objective function of facility x_1 are obtained. By considering all these combinations of $a, b \in \{1, \ldots, n\}$, $a \neq b$, the optimal location for x_1 and the corresponding objective function value can be found. Each candidate $X = \{x_1, \ldots, x_p\}$ can then be constructed in the following way:

The first facility x_1 is located in an arbitrary equilibrium point EQ_{ab} with $a, b \in \{1, \ldots, n\}$, $a \neq b$, and the corresponding objective function value $z_1 = d^w(v_a, x_1) \in Z$ is assumed to be optimal for the p-k-max problem. As z_1 is then known, the set

$$C(x_1) = \{v \in V : d^w(v, x_1) \leq z_1\}$$

of nodes covered by x_1 can be determined. $C(x_1)$ is called the *cover* of x_1 and z_1 its *coverage radius*. Since the number of outliers is $k-1$ and x_1 covers $|C(x_1)|$ nodes, only the remaining $n - k - |C(x_1)| + 1$ nodes have to be covered by $x_2, \ldots, x_p \in EQ \cup V$. The location of the remaining new facilities can then be derived by solving a $(p\text{-}1)$-k-max problem on the nodes $V \setminus C(x_1)$. Hence, the initial problem is reduced to a smaller set of customers and $p-1$ new facilities. This reduced problem can be solved recursively with the same approach by locating the next facility x_2 and assuming that it gives the optimal objective function value of the $(p\text{-}1)$-k-max problem and so on. If the solution of a subproblem leads to a contradiction to the optimality of z_1 for the initial problem or to the optimality of the objective function value of a previous iteration, respectively, the candidate does not have to be considered further.

Remark 6.6. *Note that not all duplicate equilibrium points can be filtered out at the beginning since they may differ with respect to the distance to their defining points.*

In the following, it will be assumed that the new facilities are located recursively and with a non-increasing coverage radius with respect to their respective defining (equilibrium) points. Note that at least one optimal solution of $\mathcal{X}_{\text{Cand4}}$ will be found in this way.

Let $X^{q-1} := \{x_1, \ldots, x_{q-1}\} \subseteq X$, $2 < q \leq p$, be the set of already located new facilities and assume that each new facility x_ℓ is located in an equilibrium point, i.e., $x_\ell \in EQ_{a_\ell, b_\ell}$ with $\ell \in \{1, \ldots, q-1\}$, $a_\ell, b_\ell \in \{1, \ldots, n\}$, $a_\ell \neq b_\ell$. Each candidate facility x_ℓ induces a value $z_\ell = d^w(v_{a_\ell}, x_\ell) \leq z_1$ and a cover

$$C(x_\ell) = \left\{v \in V : d^w(v_{a_\ell}, x_\ell) \leq z_\ell\right\} \quad \text{for} \quad \ell = 1, \ldots, q-1$$

of customers assigned to x_ℓ. Define

$$C(q) = \bigcup_{\ell=1}^{q-1} C(x_\ell)$$

as the set of all nodes already covered by at least one of the new facilities in X^{q-1} within its respective coverage radius. Thus, only the customers in the set

$$V(q) := V \setminus C(q),$$

i.e., customers which are not already covered by $x_1, \ldots, x_{q-1}$, have to be provided with service by the remaining new facilities $x_q, \ldots, x_p$. Formally, this can be realised by solving a (p-q+1)-k-max problem on the customer set $V(q)$ by setting the weights of the nodes in $V \setminus V(q)$ to 0 (see Remark 4.5).

With $\bar{p} := p - q + 1$ and $\bar{n} := n - |C(q)| = |V(q)|$, four cases have to be distinguished for the solution of the $\bar{p}$-k-max problem on the $\bar{n}$ remaining facilities in $V(q)$.

Case 1: $k \leq \bar{n} - \bar{p}$,
i.e., the kth largest distance can not be brought to 0 by just locating $x_q, \ldots, x_p$ in arbitrary, pairwise different nodes of $V(q)$. Applying Theorem 6.3 and the fact that the equilibrium points of zero-weighted facilities do not have to be considered leads to a finite dominating set

$$EQ(q) \times (EQ(q) \cup V(q))^{p-q},$$

for the $\bar{p}$-k-max problem, where

$$EQ(q) := \{y \in EQ : \exists\ v_s, v_t \in V(q) \text{ such that } y \in EQ_{st}\}$$

is the set of equilibrium points defined by two nodes v_s, v_t that both are not covered by one of the facilities in X^{q-1}. Thus, with the above described recursive approach, the next new facility $x_q \in X$ is located in an equilibrium point of $EQ_{a_q,b_q} \subseteq EQ(q)$ leading to a coverage radius $z_q = d^w(v_{a_q}, x_q)$ which is assumed to be optimal for the $\bar{p}$-k-max problem and satisfies $z_q \leq z_{q-1}$ (The case where $z_q > z_{q-1}$ can only occur when the coverage radius z_ℓ in one of the previous iterations was selected too small. This will be discussed later, see page 125). Afterwards, the recursion continues by solving the resulting smaller location problems until $k > \bar{n} - \bar{p}$. Note that if $q = p$, then all p iterations were Case 1-iterations since $\bar{n} - \bar{p}$ is monotonically decreasing with the iteration number q. In this case it holds obviously that all facilities of X are located in equilibrium points.

Case 2: $k = \bar{n} - \bar{p} + 1$,
i.e., the number $\bar{p}$ of facilities to locate equals the number $\bar{n} - k + 1$ of customers that still have to be covered with service. Following Lemma 6.1, an optimal solution of the problem is obtained by locating the remaining new facilities $x_q, \ldots, x_p$ in $\bar{p}$ arbitrary but pairwise different nodes of $V(q)$.

Case 3: $\bar{n} - \bar{p} + 2 \leq k \leq \bar{n}$,
i.e., the number $\bar{p}$ of facilities to locate is larger than the number $\bar{n} - k + 1$ of customers that still have to be covered with service but k is defined properly. Following Lemma 6.2

it holds that $x_q, \ldots, x_{q+\bar{n}-k}$ have to be located in arbitrary pairwise different nodes of $V(q)$ and $x_{q+\bar{n}-k+1}, \ldots, x_p$ can be placed arbitrarily in $EQ(q) \cup V(q)$. Note that the same objective function value could be obtained without $x_{q+\bar{n}-k+1}, \ldots, x_p$, i.e., using only $q + \bar{n} - k$ new facilities.

Case 4: $k \geq \bar{n}$,
i.e., the $\bar{p}$-k-max problem is not properly defined as the parameter k is larger than the number of the underlying facilities. This situation occurs if more than $n-k+1$ facilities of the initial p-k-max problem are covered by $x_1, \ldots, x_{q-1}$. Hence, all remaining new facilities $x_q, \ldots, x_p$ can be located arbitrarily in $EQ(q) \cup V(q)$ since they can neither improve nor worsen the overall objective value z_1, which was assumed to be optimal for the initial p-k-max problem. Note that in this case the same overall objective function value can be realised with only $q - 1$ new facilities.

The recursion continues with the next iteration only in Case 1 since in Case 2 to Case 4 all p new facilities of the current candidate are located in just one iteration. Thus, $k \geq \bar{n} - \bar{p} + 1$ is a stopping criterion for the recursion.

In Case 1 two further stopping criteria can be applied. If the objective function value z_q is larger than the coverage radius z_{q-1} of $C(x_{q-1})$, i.e.,

$$z_q > z_{q-1}, \tag{6.2}$$

X^{q-1} can not be extended to a solution of the initial p-k-max problem satisfying the condition $z_1 \geq z_2 \geq \ldots \geq z_p$. As the current ($p$-$q$+1)-$k$-max problem arises as a subproblem of the (p-(q-1)+1)-k-max problem with assumed optimal objective function value z_{q-1}, the distances from x_q to a node in $C(x_q)$ have to be smaller or equal then z_{q-1} to satisfy the assumption on the optimality of z_{q-1}. This corresponds to the fact that the optimal k-max value is the radius of the largest cluster of an optimal solution (see Figure 6.1).

In the last iteration, a candidate X can also be excluded from further investigations if

$$|V \setminus \bigcup_{\ell=1}^{p} C(x_\ell)| > k - 1. \tag{6.3}$$

In this case there are more outliers than allowed since more than $k - 1$ customers are not covered with service by the current assumption on the optimal objective function value.

In general, it is not necessary to consider all possible objective function value defining values $z \in Z$ to find an optimal solution of the p-k-max problem if they are considered in non-decreasing order. Hence, let the elements $y_{st} \in EQ(q)$ be sorted non-decreasingly by their distances $d^w(v_s, y_{st})$ to their defining node v_s for all $q \in \{1, \ldots, p\}$ and $s, t \in \{1, \ldots, n\}$. If the equilibrium points are tested in this order whether they are the objective function value defining facility, the first found feasible solution $\bar{X}$ is optimal. Thus, the algorithm can be terminated at this point if just one optimal solution is sought. To find further optimal solutions of the candidate set Cand4, the remaining equilibrium points having the same optimal distance to their defining nodes have to be considered. Note that the worst case

complexity for determining just one or all possible optimal solutions that can be found by the recursive approach is the same.

Overall, all relevant candidates can be obtained by iteratively locating x_1 in every equilibrium point and then starting the recursion from this point. In this way, all potential overall objective function values in the set Z and all possible positions for x_1 are considered such that the optimal value and an optimal position are determined. Only the solution with the currently best overall objective function value z_1 has to be stored and compared to the current candidate X.

The recursive approach is summarised in Algorithm 10. This version of the algorithm finds all optimal solutions that can be found with the recursive approach and does not terminate after the first optimal solution is found. The derivation of the equilibrium points with the algorithm of Bentley and Ottmann (1979), needs $\mathcal{O}(m(n+s)log(n))$ time with $s \leq 2\binom{n}{2}$. The sorting of the equilibrium points can be realised in $\mathcal{O}((mn^2)\log(mn))$. The function SmallerPkmaxProblem is called at this point for the first time.

In the function, at first the set $\overline{EQ}$ has to be determined by deleting the corresponding elements from EQ such that $\overline{EQ}$ remains sorted. This takes at most $\mathcal{O}(mn^2)$ time, which corresponds to the maximum number of equilibrium points. Since in the worst case all equilibrium points have to be tested for defining the optimal objective function value, the loop in line 7 takes $\mathcal{O}(mn^2)$ iterations. Finding the cover $C(x)$ has a complexity of $\mathcal{O}(n)$. The Cases 2 to 4 locate at most $p-1$ facilities in pairwise disjoint, arbitrary points of known sets, thus the complexity is $\mathcal{O}(p)$. As $p \leq n$, this can be neglected such that one call of the function needs $\mathcal{O}(mn^3)$ in the worst case. Case 1 calls the next recursion step. This can happen at most $p-1$ times such that the overall function is called recursively p times in the worst case. Summarising the discussion above, this leads to an overall complexity of $\mathcal{O}(m^p n^{3p})$ for Algorithm 10.

Even though this complexity is worse than the complexity of Algorithm 9, the recursion leads to much better results in practice as usually a much smaller number of candidates has to be tested for their objective function value (for more details see Section 7). Moreover, the algorithm can be implemented to compute the individual candidates in parallel as the construction of a certain candidate $X = \{x_1, \ldots, x_p\}$ does not depend on the other candidates. This leads to a significant improvement of the computation time of Algorithm 10.

Remark 6.7. *The optimal* k-max *value is the coverage radius of the largest cluster in an optimal solution. Thus, all objective defining equilibrium points* $x_1 \in EQ_{ij}$ *that lead to at least one optimal solution* $X = \{x_1, \ldots, x_p\} \in$ Cand4 *of the* 2-k-max *problem, are known after the application of Algorithm 10.*

Example 6.8. *The 2-*k-max *problem with* $k = 1$ *has to be solved on* G *(given in Figure 6.4). The set of equilibrium points is computed in Example 4.20 and has a cardinality of* $|EQ| = 23$. *Thus, the set* Cand4 $= EQ \times (EQ \cup V)$ *consists of* $23 \cdot (23+5) = 644$ *candidates that have to be checked for their objective function value.*

Algorithm 10 A recursive approach for p-k-max problems on graphs

Input: $G = (V, E)$, $p \in \{2, \dots, n\}$, $k \in \{1, \dots, n-p\}$

1: Global variables $V, E, k, n, p_0 := p, n_0 := n$
2: Determine the set EQ of all equilibrium points of G. ▷ Globally known
3: Sort $EQ \ni y_{gh}$ non-decreasingly w.r.t. $d^w(v_g, y_{gh})$ to their defining nodes v_g
4: $[\mathcal{X}, z_b] =$SmallerPkmaxProblem$(p, \infty, 0, \infty, \emptyset, \emptyset, \emptyset)$ ▷ Start of recursion

Output: Set of optimal solutions $\mathcal{X}_r$ with optimal objective function value z^*.

5: **function** SmallerPkmaxProblem$(p, z^*, z_1, z, C, X, \mathcal{X})$
6: $\overline{EQ} := EQ \setminus \{y_{ab} \in EQ : v_a \in C \vee v_b \in C\}$ (without changing the sorting).
7: **for all** $y_{gh} \in \overline{EQ}$ **do** ▷ All equilibrium points of not-covered nodes
8: Set new facility $x := y_{gh}$ with coverage radius $\bar{z} := d^w(v_g, x)$
9: **if** $p = p_0$ and $\bar{z} \leq z^*$ **then** ▷ Define z_1 after locating the first facility
10: $z_1 := \bar{z}$
11: **else if** $p = p_0$ and $\bar{z} > z^*$ **then** ▷ Stop; as $y \in EQ$ are sorted non-decreasingly
12: **return** $\mathcal{X}, z^*$
13: **if** $\bar{z} \leq z$ **then** ▷ z stays optimal
14: Extend current candidate to $\bar{X} := X \cup \{x\}$
15: Determine cover $C(x) = \{v \in V : d^w(v, x) \leq \bar{z}\}$
16: Set $\bar{C} := C \cup C(x)$, $\bar{n} := n_0 - |\bar{C}|$, $\bar{p} := p - 1$
17: **if** $\bar{p} > 0$ **then** ▷ Not all new facilities are already located
18: **if** $k \leq \bar{n} - \bar{p}$ **then** ▷ Case 1
19: $[\mathcal{X}, z^*] =$SmallerPkmaxProblem$(\bar{p}, z^*, z_1, \bar{z}, \bar{C}, \bar{X}, \mathcal{X})$
20: **else if** $k = \bar{n} - \bar{p} + 1$ **then** ▷ Case 2
21: Locate $x_{p_0-\bar{p}+1}, \dots, x_{p_0} \in V \setminus \bar{C}$ with $x_i \neq x_j \; \forall \, i, j = p_0 - \bar{p} + 1, \dots, p_0$
22: Complete candidate is $\bar{X} = \bar{X} \cup \{x_{p_0-\bar{p}+1}, \dots, x_{p_0}\}$
23: Set $\bar{p} := 0$, $\bar{n} := 0$
24: **else if** $\bar{n} - \bar{p} + 2 \leq k \leq \bar{n}$ **then** ▷ Case 3
25: Locate $x_{p_0-\bar{p}+1}, \dots, x_{p_0-\bar{p}+\bar{n}-k+1} \in V \setminus \bar{C}$ with $x_i \neq x_j$
$\forall \, i, j = p_0 - \bar{p} + 1, \dots, p_0 - \bar{p} + \bar{n} - k + 1, x_{p_0-\bar{p}+\bar{n}-k+2}, \dots, x_{p_0} \in EQ \cup V$
26: Complete candidate is $\bar{X} = \bar{X} \cup \{x_{p_0-\bar{p}+1}, \dots, x_{p_0}\}$
27: Set $\bar{p} := 0$, $\bar{n} := 0$
28: **else** ▷ Case 4
29: Locate pairwise different $x_{p_0-\bar{p}+1}, \dots, x_{p_0} \in EQ \cup V$
30: Complete candidate is $\bar{X} = \bar{X} \cup \{x_{p_0-\bar{p}+1}, \dots, x_{p_0}\}$
31: Set $\bar{p} := 0$, $\bar{n} := 0$
32: **if** $\bar{p} = 0$ and $\bar{n} \leq k - 1$ **then** ▷ New candidate $\bar{X}$ completed
33: **if** $z_1 = z^*$ **then**
34: Update set of optimal solutions to $\mathcal{X} := \mathcal{X} \cup \{\bar{X}\}$
35: **else**
36: Update set of optimal solutions to $\mathcal{X} := \{\bar{X}\}$, $z^* := z_1$
37: **else** ▷ Stop; as $y \in EQ$ are sorted non-decreasingly
38: **return** $\mathcal{X}, z^*$
39: **return** $\mathcal{X}, z^*$
40: **end function**

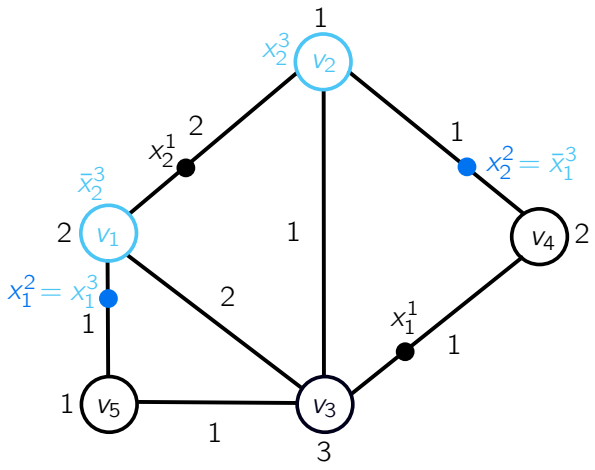


Fig. 6.4: Graph G with optimal solutions $\{x_1^k, x_2^k\}$ of the 2-k-max problem with $k \in \{1, 2, 3\}$.

In contrast to that, applying Algorithm 10 to solve the 2-1-max *problem results in less candidates that have to be evaluated. The set of sorted objective function values corresponding to the set EQ given in Example 4.20 is*

$$Z = \left\{\frac{2}{3}, \frac{3}{4}, 1, \frac{6}{5}, \frac{4}{3}, \frac{3}{2}, 2, \frac{12}{5}, 3, \frac{18}{5}, 6\right\}.$$

Starting with $z = \frac{2}{3}$*, the algorithm considers the 10 candidates*

$$\left\{EQ_{15}^{15}, EQ_{24}^{24}\right\}, \left\{EQ_{24}^{24}, EQ_{15}^{15}\right\}, \left\{EQ_{23}^{23}, EQ_{15}^{15}\right\}, \left\{EQ_{35}^{35}, EQ_{24}^{24}\right\}, \left\{EQ_{34}^{34}, EQ_{15}^{15}\right\},$$

$$\left\{EQ_{34}^{34}, EQ_{25}^{13}\right\}, \left\{EQ_{34}^{34}, EQ_{25}^{23}\right\}, \left\{EQ_{34}^{34}, EQ_{25}^{34}\right\}, \left\{EQ_{34}^{34}, EQ_{25}^{35}\right\}, \left\{EQ_{12}^{12}, EQ_{35}^{35}\right\}$$

without finding a feasible solution. A candidate is here defined as a completed solution X*, i.e., all* p *new facilities of* X *are located, but* X *does not have to be feasible, i.e., it does not have to satisfy the condition* (6.3)*. Then, for* $z = \frac{4}{3}$*, the first feasible and therefore optimal solution is found as*

$$X^1 = \left\{x_1^1 = \left(e_{34}, \frac{1}{3}\right), \; x_2^1 = \left(e_{12}, \frac{1}{3}\right)\right\} \quad \textit{with} \quad z^1 = \frac{4}{3}.$$

In this case, X *is the only optimal solution in* Cand4 *but it is also found with changed roles of* x_1^1 *and* x_2^1*. Thus, both new facilities define the optimal objective function value (see Remark 6.7). For* $k = 2$*, the optimal solution found by the recursive approach is*

$$X^2 = \left\{x_1^2 = \left(e_{15}, \frac{1}{3}\right), x_2^2 = \left(e_{24}, \frac{2}{3}\right)\right\} \quad \textit{with} \quad z^2 = \frac{2}{3}$$

and for $k = 3$ *the two alternative solutions*

$$X^3 = \left\{x_1^3 = \left(e_{15}, \frac{1}{3}\right), \; x_2^3 = (e_{12}, 1)\right\} \quad \textit{with} \quad \bar{X}^3 = \left\{\bar{x}_1^3 = \left(e_{24}, \frac{2}{3}\right), \; \bar{x}_2^3 = (e_{12}, 0)\right\}$$

are optimal with $z^3 = \frac{2}{3}$. *Note that for* $k = 2$ *and* $k = 3$ *no other candidates than the optimal solutions have to be considered with the recursive approach as the optimal objective function value corresponds to the smallest value of* Z. *All these optimal solutions* $X^k = \{x_1^k, x_2^k\}$ *are shown in Figure 6.4.*

Remark 6.9. *If the optimal objective function value is rather large compared to the other elements of* Z, *nearly all equilibrium points have to be tested for being objective function value defining due to the non-decreasing sorting of* Z. *Therefore, it might seem to be reasonable to apply a bisection strategy on the set* Z *of possible optimal objective function values to reduce the number of candidates that have to be considered. However, this approach may lead to wrong solutions in general.*

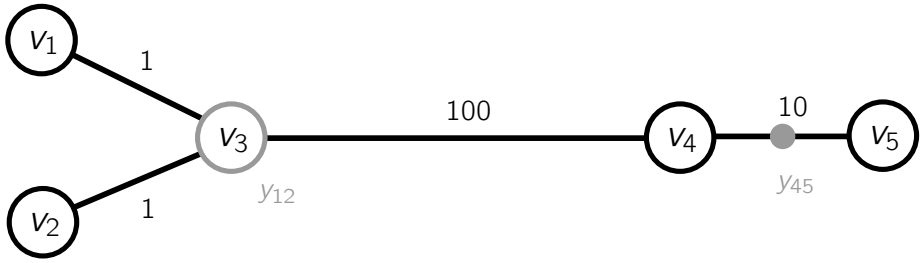


Fig. 6.5: y_{45} with objective function value $z = 5$ is no feasible solution of the 1-3-max problem, $y_{12} = v_3$ with objective function value $z = 2$ is feasible.

The problem is that it is not sufficient to test if there is an equilibrium point with exact objective function value $\bar{z}$. *If there is no solution in the set* Cand4 *with objective value* $\bar{z}$, *this does not mean that there is also no feasible solution in* Cand4 *for another* $\tilde{z} \in Z$ *with* $\tilde{z} < \bar{z}$. *An example can be seen in Figure 6.5, where the* 1-3-max *problem has to be solved on the graph* G. *For* $\bar{z} = 5$, *the equilibrium point* $y_{45} = EQ_{45}^{45}$ *is considered as* $d^w(v_4, y_{45}) = 5$. *Obviously,* y_{45} *does not lead to a feasible solution as it covers only two existing facilities whereas at least three customers have to be covered with service in a feasible solution. Moreover, there is no other equilibrium point with distance* 5 *to its defining facilities. However, for* $\tilde{z} = 2$ *the point* $y_{12} \in EQ_{12}^{13}$ *is a feasible solution of the underlying problem with objective function value* 2 *as* y_{12} *covers three existing facilities.*

Consequently, it rather has to be tested for each possible objective value $\bar{z}$ *if there is an equilibrium point with* $z \in Z$ *for* $z \leq \bar{z}$. *Therefore, all these values* $z \leq \bar{z}$ *have to be considered anyway.*

Two ways of defining a cover

For every new facility $x_\ell \in EQ_{a_\ell b_\ell}$ with $\ell \in \{2, \ldots, p\}$, the recursive approach defines all existing facilities that have a weighted distance of at most $z_\ell = d^w(v_{a_\ell}, x_\ell)$ to x_ℓ as its cover. Thus, the radii z_ℓ of the covers are getting smaller or stay at most equal such that $z_1 \geq z_2 \geq \ldots \geq z_p$.

However, it is also possible to define the cover $C(x_\ell)$, $\ell = 2, \ldots, p$, as the set of facilities that have a weighted distance to x_ℓ of at most z_1. As z_1 is assumed to be optimal for the initial p-k-max problem for the current candidate, all other new facilities are also allowed to generate weighted distances of up to z_1 to its allocated customers. By choosing z_1 as the coverage radius for every new facility, the cover $C(x_\ell)$ gets possibly larger since $z_1 \geq z_\ell$ for all $\ell = 1, \ldots, q-1$. As a consequence, $V(q)$ gets possibly smaller and the following (p-q+1)-k-max problem corresponds to a smaller set of existing facilities. In this way the number of recursion steps in Algorithm 10 can be decreased by choosing z_1 as coverage radius of $C(x_\ell)$ instead of z_ℓ. However, the definition of the cover based on z_ℓ leads to a reasonable property of the found optimal solutions.

For a clearer notation, let

$$C(x_\ell, z_\ell) = \{v \in V : d^w(v, x_\ell) \leq z_\ell\}, \quad \ell = 1, \ldots, p,$$

be the cover based on the distance z_ℓ of the current facility x_ℓ to its defining nodes and let

$$C(x_\ell, z_1) = \{v \in V : d^w(v, x_\ell) \leq z_1\}, \quad \ell = 1, \ldots, p,$$

be the corresponding cover based on the distance z_1 that is given by the weighted distance of the first facility x_1 to its defining nodes. If the covers are defined as $C(x_\ell, z_\ell)$, $\ell = 1, \ldots, p$, Algorithm 10 yields only a smaller subset $\mathcal{X}_{z_\ell}$ of the set of optimal solutions $\mathcal{X}_{z_1}$ obtained with $C(x_\ell, z_1)$. Since in general $z_\ell < z_1$ for all $\ell > 1$, a coverage radius of z_ℓ is more restrictive and thus fewer combinations of equilibrium points are qualified to cover the remaining nodes. However, the optimal solutions in $\mathcal{X}_{z_\ell}$ are particularly relevant solutions in practice because they have a special structure: As $C(x_\ell, z_\ell)$ uses the distance of the current new facility $x_\ell = y_{st} \in EQ(\ell)$ to its defining nodes v_s, v_t as coverage radius, x_ℓ is always the center point of its cover. Contrary to this, the cover $C(x_\ell, z_1)$ uses $z_1 > z_\ell$ and not the distance $d^w(v_s, x_\ell)$ as coverage radius. Thus, the covers are larger and x_ℓ may lie in a non-central position w.r.t. its cover since v_s and v_t do in general not define the diameter of the cover. As a consequence, it is only guaranteed that the distance of every covered node to its nearest new facility is smaller or equal to z_1.

Example 6.10. *Figure 6.6 and Figure 6.7 show an example of a* 2-1-max *problem on the given graph with* $n = 12$ *nodes. All nodes are equally weighted and the edge weights correspond to the Euclidean lengths of the lines representing the edges. Obviously, both solutions* $\bar{X} \in \mathcal{X}_{z_\ell}$ *and* $\tilde{X} \in \mathcal{X}_{z_1}$ *are optimal for the given problem and the first facility defines the optimal objective function value of* z_1*. However,* $\bar{X}$ *is in practice usually preferred in relation to* $\tilde{X}$ *as* $x_2 \in \bar{X}$ *yields a smaller maximum distance to the nodes of its cover.*

In contrast to the evaluation of all candidate pairs (see Algorithm 9), the recursive approach is in general not suitable to compute the solutions of the p-k-max problem for all values of the parameter k simultaneously. This holds as the sizes of the supbproblems depend on the value of k such that different candidates may be relevant for the optimal solution. In the following section, a technique to find further optimal solutions, that are not an element of Cand4, is introduced.

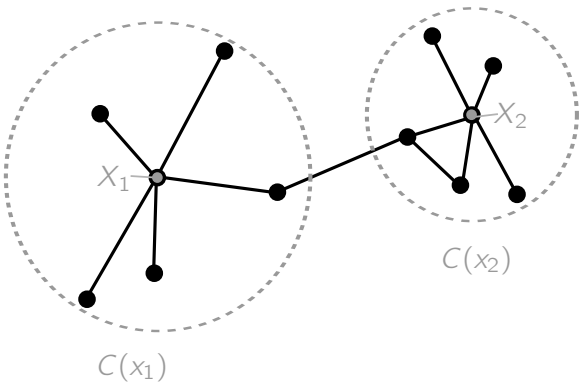


Fig. 6.6: Optimal $\bar{X} \in \mathcal{X}_{z_\ell}$ with $z^* = z_1$ based on $C(x_\ell, z_\ell)$, $\ell = 1, 2$

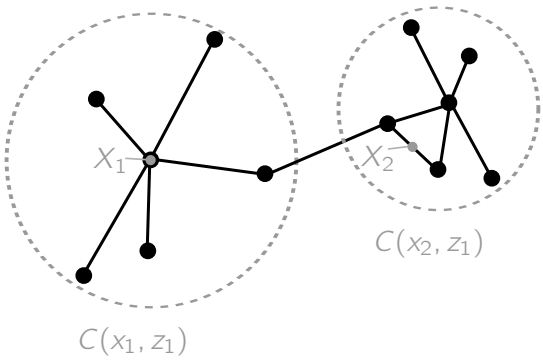


Fig. 6.7: One optimal solution $\tilde{X} \in \mathcal{X}_{z_1}$ with $z^* = z_1$ based on $C(x_\ell, z_1)$, $\ell = 1, 2$

6.3 A Local Analysis to Find All Optimal Solutions

In this section, the finite dominating set Cand4 from Theorem 6.3 is extended such that this new candidate set contains alternative optimal solutions, i.e., alternative optimal solutions that have other properties than the candidates in Cand4. The overall aim of this section is to determine, if possible, all optimal solutions of the p-k-max problem. Therefore, the idea is to apply a local analysis to generate a subdivision such that the objective function is piecewise linear and concave on every cell of this subdivision, similar to the idea used for the single facility case. The resulting finite dominating set can afterwards be reduced by using the solutions generated by the recursive approach described in Section 6.2.

Theorem 6.3 only ensures that *at least one* optimal solution of the p-k-max problem has all new locations in the finite dominating set Cand4 $= EQ \times (EQ \cup V)^{p-1}$. The following example shows that the problem may have alternative optimal solutions that do not lie in this candidate set.

Example 6.11 (Continuation of Example 6.8)**.** *Consider the graph G with $n = 5$ nodes from Example 6.8. As stated in Example 6.8, an optimal solution of the* 2-1-max *problem is*

$$X^1 = \left\{ \left(e_{34}, \frac{1}{3} \right), \left(e_{12}, \frac{1}{3} \right) \right\}$$

*with optimal objective function value $z^1 = \frac{4}{3}$. Figure 6.8 shows another feasible solution $\bar{X}^1 = \{\bar{x}_1^1, \bar{x}_2^1\}$ of the 2-k-*max *problem with $k = 1$ on G with*

$$\bar{x}_1^1 = \left(e_{34}, \frac{1}{3} \right) \in EQ_{45}^{34} \quad \textit{and} \quad \bar{x}_2^1 = \left(e_{15}, \frac{2}{3} \right) \notin EQ \cup V.$$

This solution is also optimal since the objective function value of $\bar{X}^1$ is $\bar{z}^1 = z^1 = \frac{4}{3}$. Obviously, $\bar{X}^1$ can not be found using the set Cand4 *as $\bar{x}_2^1$ is neither an equilibrium point nor a node of G.*

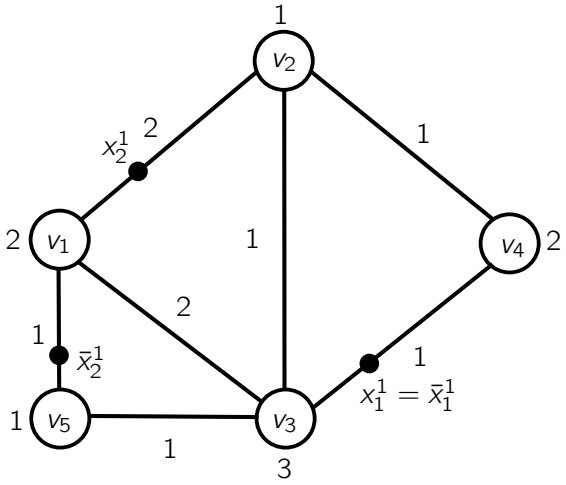


Fig. 6.8: Alternative optimal solution $\bar{X}^1 = \{\bar{x}_1^1, \bar{x}_2^1\}$ of the 2-1-max problem with $\bar{X}^1$ not being an element of Cand4

Obviously, there are other relevant points on the graph besides the equilibrium points that have to be considered for the analysis of *p-k*-max problems. To get an idea of the properties of optimal solutions not lying in the candidate set Cand4, the distance functions over the corresponding edges of the optimal new locations are analysed further. As shown in Section 4.4, the weighted distance functions $d^w(v_i, x)$ over an edge $e_{ab} \in E$ depend on individual new facilities $x = (e_g, t)$ with $t \in [0, l_{ab}]$, and are piecewise linear and concave with breakpoints only at bottleneck points.

Example 6.12 (Continuation of Example 6.11)**.** *Figure 6.9 shows the graphs of the weighted distance functions over the edges $e_{34} = (v_3, v_4)$ and $e_{15} = (v_1, v_5)$. As shown in Example 6.11, for the optimal solution $\bar{X}^1 = \{\bar{x}_1, \bar{x}_2\}$, it holds that $\bar{x}_1^1 \in e_{34}$ and $\bar{x}_2^1 \in e_{15}$. An important observation in this example is that for the new locations in $\bar{X}^1$ it follows*

$$d^w(v_4, \bar{x}_1^1) = d^w(v_1, \bar{x}_2^1) = \frac{4}{3},$$

i.e., node v_4 has the same weighted distance to the new facility $\bar{x}_1^1$ as node v_1 has to the new facility $\bar{x}_2^1$. This results in

$$d^w(V, \bar{X}^1) = \left(\frac{4}{3}, \frac{4}{3}, 1, \frac{4}{3}, \frac{1}{3}\right)^\top,$$

i.e., there are at least two equal elements in the vector of distances. Therefore, at least two different permutations $\sigma_1, \sigma_2 \in \Sigma(\bar{X}^1)$ with $\sigma_1 = (1, 2, 4, 3, 5)$ and $\sigma_2 = (1, 4, 2, 3, 5)$ exist such that

$$d^w(v_{\sigma_a(1)}, \bar{X}^1) \geq \ldots \geq d^w(v_{\sigma_a(5)}, \bar{X}^1)$$

for $a \in \{1, 2\}$. Obviously, with just a small shift of one of the new facilities, this equality of the distances is broken and at least one of the permutations does no longer lead to

*the wanted ordering. Consequently, this results in a non-linearity of the p-k-*max *objective function in* $\bar{X}^1$. *As* $\bar{x}_1^1 \in EQ_{45}$, *it also holds that* $d^w(v_5, \bar{x}_1^1) = d^w(v_1, \bar{x}_2^1) = \frac{4}{3}$.

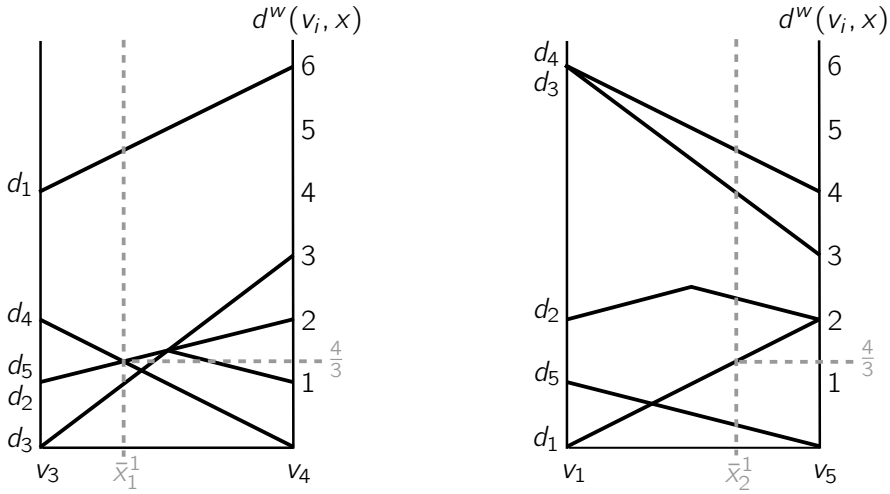


Fig. 6.9: Distance functions $d^w(v_i, x)$ over the edges e_{34} (right) and e_{15} (left)

In the following the case of only two new facilities $X = \{x_1, x_2\}$ is considered first to get an intuition of the problem and its properties. Afterwards, the described ideas will be presented in a more formal way for p new facilities .

6.3.1 The 2-facility Case

To analyse the 2-k-max problem further, some notation with respect to the distances is needed. The weighted distance from a node $v_i \in V$ to a point $x = (e_g, t)$ with $e_g = (v_{a_g}, v_{b_g})$ is

$$d^w(v_i, x) = \min\left\{d_+^w(v_i, x), d_-^w(v_i, x)\right\},$$

where

$$d_+^w(v_i, x) = w_i(d(v_i, v_{a_g}) + tl_g)$$

is the length of the shortest path between v_i and x via node v_{a_g} and

$$d_-^w(v_i, x) = w_i(d(v_i, v_{b_g}) + (1-t)l_g)$$

is the length of a respective shortest path through node v_{b_g}.

In the following, let $X = \{x_1, x_2\} \subseteq A(G)$ be a solution of the 2-k-max problem with $x_1 = (e_g, t_1)$ and $x_2 = (e_h, t_2)$ with $e_g = (v_{a_g}, v_{b_g})$, $g \in \{1, \ldots, m\}$ and $e_h = (v_{a_h}, v_{b_h})$, $h \in \{1, \ldots, m\}$ and $v_{a_g}, v_{b_g}, v_{a_h}, v_{b_h} \in V$. Note that $g = h$ is possible for now. As observed in Example 6.12, the permutation of the sorted distance vector of a solution X changes only if at least two elements of the distance vector $d^w(V, X)$ are equal, i.e., if

$$d_\alpha^w(v_i, x_q) = d_\beta^w(v_j, x_r) \quad \text{for} \quad i, j \in \{1, \ldots, n\},\ q, r \in \{1, 2\},\ \alpha, \beta \in \{+, -\}. \tag{6.4}$$

As the non-linearity of the k-max function is induced by the different sortings of the vector of distances, the objective function of the 2-k-max problem may have a breakpoint in X. Note that it is allowed that $\alpha = \beta$ as well as $i = j$ and $q = r$.

In the following, a subdivision of the unit square $[0,1]^2$ will be determined to obtain a representation of linearity domains of the 2-k-max function. Therefore, two edges $e_g, e_h \in E$ with $g, h \in \{1, \dots, m\}$ are assumed to be fixed and the two new facilities $x_1 = (e_g, t_1)$ and $x_2 = (e_h, t_2)$ are considered to be located on these edges. Although it is not completely correct, the candidate $X = \{x_1, x_2\}$ will (as always) be written as a set of points by using set notations to emphasise that the individual new facilities are interchangeable. Nevertheless, the notation $\{x_1, x_2\} \in e_g \times e_h$ will be used. To avoid duplication of candidates, it can be assumed without loss of generality that $g \leq h$.

As soon as an edge $e \in E$ is fixed, there is a one to one correspondence between a point $x = (e, t)$ on this edge and the parameter $t \in [0,1]$. Thus, a distinction will be made between the candidate $\{x_1, x_2\}$ being an element of the Cartesian product $e_g \times e_h$ of two edges and the parameters t_1, t_2 varying on the unit square $[0,1]^2$. The subdivision and the resulting linearity domains live on the unit square $[0,1]^2$ such that every point of the subdivision corresponds to a pair $\{x_1, x_2\} \in e_g \times e_h$ on the edges of G. Hence, each edge of the network is represented by the unit interval $[0,1]$ such that the Cartesian product $e_g \times e_h$ of the edges $e_g, e_h \in E$ is represented by the unit square $[0,1]^2$. With the result of the following lemma, pairs of equal edges $e_g = e_h$ do not have to be considered.

Lemma 6.13 ▶ Optimal facilities on different edges

Let $X^* = \{x_1^*, x_2^*\}$ be an optimal solution of a 2-k-max problem with $n \geq 2$ and $k \leq n-2$. Then x_1^* and x_2^* can be described by $x_1^* = (e_g, t_1)$ and $x_2^* = (e_h, t_2)$ with $g, h \in \{1, \dots, m\}$ such that $e_g \neq e_h$, i.e. the two optimal new facilities can not lie in the interior of the same edge.

Proof. Let $X^* = \{x_1^*, x_2^*\}$ be an optimal solution of the 2-k-max problem with optimal objective value z^* and set of outliers V_{k-1}^*. Assume that $x_1^* = (e_g, t_1)$ and $x_2^* = (e_g, t_2)$ for $g \in \{1, \dots, m\}$, i.e., x_1^* and x_2^* lie on the same edge $e_g = (v_{g_a}, v_{g_b}) \in E$ with $v_{g_a}, v_{g_b} \in V$ for $g_a, g_b \in \{1, \dots, n\}$. Moreover, let w.l.o.g. $t_1 \leq t_2$ and $t_1, t_2 \in (0,1)$ such that x_1^* and x_2^* lie in the relative interior of e_g and not in v_{g_a} or v_{g_b}, i.e. both points x_1^* and x_2^* can not be described using another edge of G. An optimal allocation of the customers to x_1^* and x_2^* is defined by the sets

$$V_1 = \{v \in V : d^w(v, x_1^*) \leq d^w(v, x_2^*)\}$$

and

$$V_2 = \{v \in V : d^w(v, x_2^*) < d^w(v, x_1^*)\}.$$

Note that the shortest path between a node $v \in V_1$ and x_1^* always contains the node v_{g_a} due to $t_1 \leq t_2$. Analogously, the shortest path between x_2^* and $v \in V_2$ goes through node v_{g_b}. Additionally, assume w.l.o.g. that x_1^* is the facility that defines the optimal objective function value z^*, i.e., it exists a node $v \in V_1$ such that $d^w(v, x_1^*) = z^*$.

As all shortest paths between nodes $v_i \in V_1$ and x_1^* pass through v_{g_a}, it holds that

$$d^w(v_i, x_1^*) = w_i(d(v_i, v_{g_a}) + t_1 l_g).$$

Hence, the weighted distances to all nodes $v_i \in V_1$ can be reduced by shifting x_1^* towards the endnode v_{g_a} of the edge e_g such that $t_1 l_g = 0$. The resulting new solution is $\bar{X} = \{\bar{x}_1, x_2^*\}$ with $\bar{x}_1 = v_{g_a}$. The shifting of x_1^* into v_{g_a} does not change the allocation to the new facilities as the distances of x_1^* to the nodes in V_2 can only increase. Note that the set of outliers $\bar{V}_{k-1}$ may be different from V_{k-1}^* but the distances of the nodes covered by $\bar{x}_1$ are strictly smaller than the distances of these nodes to x_1^*.

Case 1: $d^w(v_j, x_2^*) < z^*$ for all $v_j \in V_2 \setminus V_{k-1}^*$
The new solution $\bar{X}$ yields a strictly better objective function value $\bar{z} < z^*$ than X^* since the distance of x_2^* to all nodes covered by it is strictly smaller than z^*. This contradicts the assumption of X^* being optimal.

Case 2: $d^w(v_j, x_2^*) = z^*$ for at least one $v_j \in V_2 \setminus V_{k-1}^*$
The new solution $\bar{X}$ is as good as X^* as it has the same objective function value. Thus, shift x_2^* towards the endnode $v_{g_b} =: \tilde{x}_2$ of edge e_g analogously to the shifting of x_1^*. As a consequence, $d^w(v, \tilde{x}_2) < d^w(v, x_2^*)$ for all $v \in V_2$. Then, the new solution $\tilde{X} = \{\bar{x}_1, \tilde{x}_2\}$ is strictly better than x^* and this contradicts the optimality of X^*. ∎

Note that the result of Lemma 6.13 does in general not hold for the location of $p \geq 3$ new facilities. If the optimal objective function value can also be realised with $2 \leq \bar{p} < p$ new facilities, the remaining $p - \bar{p}$ new facilities can be located arbitrarily on the edges of the graph and therefore there may be an optimal solution with two or more new facilities on the same edge.

The approach and the notation described below are similar to that of Kalcsics (2011), who derived a finite dominating set for the multi-facility median problem with positive and negative weights on general graphs, i.e., for the problem

$$\min_{X=\{x_1,\ldots,x_p\}\subseteq A(G)} f(X) = \sum_{v_i \in V} w_i d(v_i, X), \tag{6.5}$$

where $w_i \in \mathbb{R}$ may be positive or negative for $i = 1, \ldots, n$. For the 2-facility problem it is observed that the equality

$$d^w(v_i, x_1^*) = d^w(v_i, x_2^*) \quad \text{for some} \quad i \in \{1, \ldots, n\},$$

holds in at least one optimal solution $X^* = \{x_1^*, x_2^*\}$ with $x_1^* = (e_g, t_1^*)$ and $x_2^* = (e_h, t_2^*)$, i.e., at least one node $v_i \in V$ has the same distance to both new locations. This results in a non-linearity of the distance function $d^w(v_i, X)$ at X^*. For a fixed pair of edges e_g, e_h, the distance functions $d^w(v_i, X)$ are piecewise linear and concave for weight $w_i > 0$, resp. convex for $w_i < 0$ in $t_1, t_2 \in [0, 1]$ over the corresponding unit square $[0, 1]^2$ as the functions $d^w(v_i, x_q)$, $q = 1, 2$, are piecewise linear and concave, resp. convex there.

Considering two fixed edges e_g and e_h to locate $x_1 = (e_g, t_1)$ and $x_2 = (e_h, t_2)$ on, the behaviour of the distance function $d^w(v_i, X)$ is analysed over the unit interval $[0,1]^2$ representing the Cartesian product $e_g \times e_h$. It can be seen that the breakpoints of $d^w(v_i, X)$ occur in intersection points of at least two distinct functions $d^w_\alpha(v_i, x_q)$ and $d^w_\beta(v_i, x_r)$ with $q, r \in \{1,2\}$ and $\alpha, \beta \in \{+,-\}$. Two cases are distinguished: For $q = r$, the set of intersection points is defined by all points $\{x_1, x_2\} \in e_g \times e_h$ that satisfy the equation

$$d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_q) \quad \text{with} \quad \alpha \neq \beta. \tag{6.6}$$

As $\alpha \neq \beta$, $x_q \in BN_i$ is a bottleneck point in this case. Hence, the set of all intersection points $\{x_1, x_2\}$ satisfying equation (6.6) define a vertical ($q = 1$) resp. horizontal ($q = 2$) line depending on $t_1, t_2 \in \mathbb{R}$ that may or may not intersect the unit square $[0,1]^2$.

For $q \neq r$, the equation

$$d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_r) \tag{6.7}$$

is satisfied by all intersection points of the functions $d^w_\alpha(v_i, x_q)$ and $d^w_\beta(v_i, x_r)$ for some $i \in \{1, \ldots, n\}$. Equation (6.7) describes a linear function depending on $t_1, t_2 \in \mathbb{R}$ with known slope. The line can, but does not have to, intersect the unit square $[0,1]^2$.

The set of all lines for all pairs of nodes that are induced by the conditions (6.6) and (6.7) and that do intersect $[0,1]^2$ introduces a subdivision of the unit square representing the edges $e_g \times e_h$. The subsets induced by the subdivision that have non-empty interior are called *cells*. As the subdivision is build up based on the set of breakpoints of the distance functions, $d^w(v_i, X)$ is linear over each cell of this subdivision. The cells of the subdivision are convex as they are formed by the intersection of linear functions. Thus, a local minimum of the 2-facility median problem with positive and negative weights has to lie in the set of intersection points of the lines defining the subdivision, including the intersection points of these lines with the bounding segments of the unit square. A global optimal solution can then be found by constructing such a subdivision for every pair of edges of the graph and comparing the local optimal solutions on them. Similar approaches are also used in Kalcsics et al. (2014) and Kalcsics et al. (2015).

Similarly, a subdivision for the 2-k-max problem can be obtained such that the k-max function is linear on every induced subset of the subdivision. Therefore, the breakpoints of the function $d^w(V, X)$, described by Equation (6.4), are characterised in the following by distinguishing four cases.

Case 1: $i \neq j$ and $q = r$,
i.e., $d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_q)$ for $i, j \in \{1, \ldots, n\}$ with $i \neq j$, $q \in \{1,2\}$ and $\alpha, \beta \in \{+,-\}$. For all combinations of α and β, this equation describes the equilibrium points $EQ_{ij}^{a_g b_g}$ resp. $EQ_{ij}^{a_h b_h}$ of the nodes v_i and v_j on the edge e_g resp. e_h. The set of breakpoints for $q = 1$ is therefore given by

$$B^{\alpha\beta}_{ij11} = \left\{ \{x_1, x_2\} \in e_g \times e_h : x_1 = EQ_{ij}^{a_g b_g}(\alpha, \beta) \right\},$$

where $EQ_{ij}^{a_g b_g}(\alpha, \beta)$ is the equilibrium point given by the equation $d^w_\alpha(v_i, x_1) = d^w_\beta(v_j, x_1)$

for a fixed combination of $\alpha, \beta \in \{+, -\}$. Analogously, it holds for $q = 2$ that

$$B_{ij22}^{\alpha\beta} = \left\{ \{x_1, x_2\} \in e_g \times e_h : x_2 \in EQ_{ij}^{a_h b_h}(\alpha, \beta) \right\}.$$

Because of $B_{ijqq}^{\alpha\beta} = B_{jiqq}^{\beta\alpha}$ for $q = 1, 2$, it can w.l.o.g. be assumed that $i < j$. Analysing all four possible combinations of α and β leads to four equations for $t_q \in [0, 1]$ describing the location of $x_q = (e_u, t_q)$, for $q \in \{1, 2\}$ and corresponding $u \in \{g, h\}$:

$$\alpha = +,\ \beta = + \quad \Rightarrow \quad t_q = \frac{w_j d(v_j, v_{a_u}) - w_i d(v_i, v_{a_u})}{w_i l_u - w_j l_u}, \quad \text{if} \quad w_i \neq w_j \tag{6.8}$$

$$\alpha = +,\ \beta = - \quad \Rightarrow \quad t_q = \frac{w_j d(v_j, v_{b_u}) + w_j l_u - w_i d(v_i, v_{a_u})}{w_i l_u + w_j l_u} \tag{6.9}$$

$$\alpha = -,\ \beta = + \quad \Rightarrow \quad t_q = \frac{w_j d(v_j, v_{a_u}) - w_i l_u - w_i d(v_i, v_{b_u})}{-w_i l_u - w_j l_u} \tag{6.10}$$

$$\alpha = -,\ \beta = - \quad \Rightarrow \quad t_q = \frac{w_j d(v_j, v_{b_u}) + w_j l_u - w_i d(v_i, v_{b_u}) - w_i l_u}{-w_i l_u + w_j l_u}, \quad \text{if} \quad w_i \neq w_j. \tag{6.11}$$

The right hand side of these equations is given by a constant in all four cases. Hence, the set $B_{ij11}^{\alpha\beta}$ defines a constant function in $t_q \in [0, 1]$ (as x_q depends on t_q) that describes a vertical line and $B_{ij22}^{\alpha\beta}$ analogously defines a constant function depending on $t_r \in [0, 1]$ that gives a horizontal line. Note that not the whole line in $t_q \in \mathbb{R}$ resp. $t_r \in \mathbb{R}$ but only the intersection of the line with the unit square $[0, 1]^2$ is of interest since t_q, t_r have to be in $[0, 1]$ to describe a point on an edge of the graph (see also Remark 6.18).

Definition 6.14 ▶ Equilibrium lines

The lines corresponding to $B_{ijqq}^{\alpha\beta}$ for $i, j \in \{1, \ldots, n\}$, $i \neq j$, $q \in \{1, 2\}$ and $\alpha, \beta \in \{+, -\}$ over the unit square $[0, 1]^2$ are called *equilibrium lines*.

Figure 6.10 gives an example of an equilibrium line for $i = 2, j = 4$ and $x_1 \in e_{34}$, $x_2 \in e_{15}$ for the known example-graph given in Figure 4.5. The maximum number of equilibrium lines for every combination v_i, v_j over the unit square for a fixed pair of edges is eight. However, not all equations (6.8)-(6.11) have to have a solution for $t_q, t_r \in [0, 1]$ such that there can also be less than eight equilibrium lines.

Note that the set of equilibrium points of v_i and v_j on $e_u \in E$, $u \in \{g, h\}$, described by the equations (6.8) and (6.11), has infinitely many elements, i.e., $|EQ'_{ij}| = \infty$. In the case of $w_i = w_j$, there are no equilibrium lines of type (6.8) and (6.11) as the denominator of the fraction is zero. However, if both boundary points of EQ_{ij} are attained in the nodes v_{a_u} and v_{b_u}, the corresponding equilibrium lines are given by the boundaries of the unit square. Otherwise, one of the boundary points of EQ_{ij} is attained in an inner point of e_u and then the equilibrium line can also be described by (6.9) or (6.10).

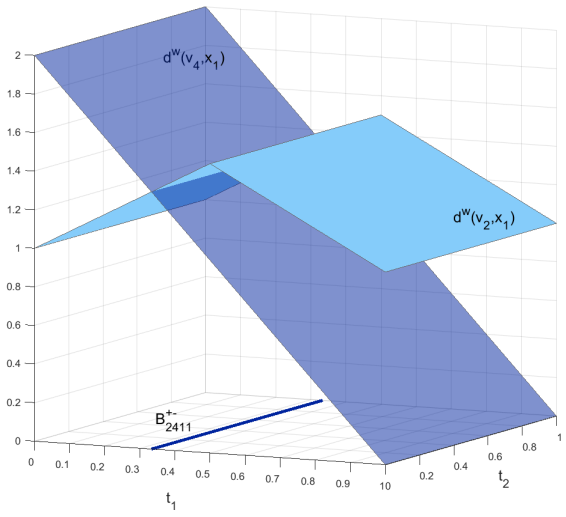


Fig. 6.10: Equilibrium line given by B^{+-}_{2411} for $i = 2$, $j = 4$, $x_1 = (e_{34}, t_1)$, $x_2 = (e_{15}, t_2)$

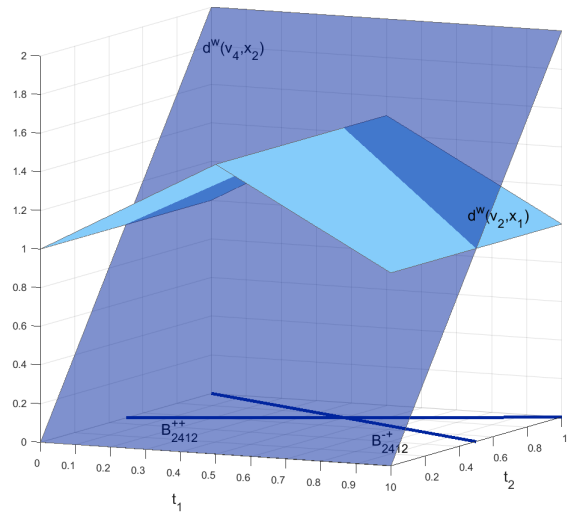


Fig. 6.11: Balance lines given by B^{++}_{2412} and B^{-+}_{2412} for $x_1 = (e_{34}, t_1)$, $x_2 = (e_{15}, t_2)$

Case 2: $i \neq j$ and $q \neq r$

In this case, the set of breakpoints is given by

$$B^{\alpha\beta}_{ijqr} = \left\{ \{x_1, x_2\} \in e_g \times e_h : d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_r) \right\},$$

for $i, j \in \{1, \ldots, n\}$ with $i \neq j$, $q, r \in \{1, 2\}$ with $q \neq r$ and $\alpha, \beta \in \{+, -\}$. This set contains pairs $\{x_1, x_2\}$ that share the same distance to respective existing facilities v_i and v_j. Because of $B^{\alpha\beta}_{ijqr} = B^{\beta\alpha}_{jirq}$, it can w.l.o.g. be assumed that $i < j$ and $q < r$, i.e., $q = 1$ and $r = 2$. Analysing again all four cases of α-β-combinations leads to the following equations in $t_1, t_2 \in [0, 1]$ to describe the locations of $x_1 = (e_g, t_1)$ and $x_2 = (e_h, t_2)$:

$$w_i l_g t_1 - w_j l_h t_2 = w_j d(v_j, v_{a_h}) - w_i d(v_i, v_{a_g}) \tag{6.12}$$

$$w_i l_g t_1 + w_j l_h t_2 = w_j d(v_j, v_{b_h}) - w_i d(v_i, v_{a_g}) + w_j l_h \tag{6.13}$$

$$-w_i l_g t_1 - w_j l_h t_2 = w_j d(v_j, v_{a_h}) - w_i d(v_i, v_{b_g}) - w_i l_g \tag{6.14}$$

$$-w_i l_g t_1 + w_j l_h t_2 = w_j d(v_j, v_{b_h}) - w_i d(v_i, v_{b_g}) - w_i l_g + w_j l_h. \tag{6.15}$$

Obviously, the set of breakpoints $B^{\alpha\beta}_{ijqr}$ defines a linear function depending on $t_1, t_2 \in [0, 1]$ for every fixed combination of $i < j$, $q < r$ and α, β over the unit square $[0, 1]^2$. Therefore, there are at most four line segments for every pair of nodes v_i, v_j and for every pair of edges e_g, e_h as it is possible that not all equations have a solution for t_1, t_2 in the interval $[0, 1]$. The lines corresponding to the sets $B^{\alpha\beta}_{ijqr}$ for $i \in \{1, \ldots, n\}$ with $i \neq j$, $q, r \in \{1, 2\}$ with $q \neq r$, and $\alpha, \beta \in \{+, -\}$ over the unit square $[0, 1]^2$ are called *balance lines* in the following. A formal definition is given later with Definition 6.17.

Note that the balance lines of type (6.12) and (6.15) are parallel as they both have slope $\frac{w_i l_g}{w_j l_h}$, whereas (6.13) and (6.14) are parallel with slope $-\frac{w_i l_g}{w_j l_h}$. Figure 6.11 shows the balance

lines defined by the sets B^{++}_{2412} of type (6.12) and B^{-+}_{2412} of type (6.14) for the edges e_{34} and e_{15} of the example graph in Figure 4.5.

Case 3: $i = j$ and $q = r$,
i.e., $d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_q)$ such that $\alpha \neq \beta$ with $\alpha, \beta \in \{+, -\}$, $i \in \{1, \ldots, n\}$, $q \in \{1, 2\}$. As α and β are not allowed to be equal it can w.l.o.g. be assumed that $\alpha = +$ and $\beta = -$. The equation describes the bottleneck point $BN_i^{a_u b_u}$ of node v_i on edge e_u, $u \in \{g, h\}$. The set of breakpoints for $q = 1$ is therefore given by

$$B^{+-}_{ii11} = \left\{ \{x_1, x_2\} \in e_g \times e_h : x_1 = BN_i^{a_g b_g} \right\},$$

and for $q = 2$ analogously by

$$B^{+-}_{ii22} = \left\{ \{x_1, x_2\} \in e_g \times e_h : x_2 = BN_i^{a_h b_h} \right\}.$$

Solving the above equation for t_q to obtain the location of $x_q = (e_u, t_q)$ leads to

$$t_q = \frac{w_i d(v_i, v_{b_u}) + w_i l_u - w_i d(v_i, v_{a_u})}{2 w_i l_u} \quad \text{for} \quad q \in 1, 2, \ u \in \{g, h\}$$

which is, similar to the equilibrium lines, a function constant in t_q. For $t_q \in [0, 1]$, the function describes a vertical (for $q = 1$) resp. a horizontal (for $q = 2$) line over the unit square $[0, 1]^2$.

Definition 6.15 ▶ Bottleneck lines
The lines corresponding to B^{+-}_{iiqq} for $i \in \{1, \ldots, n\}$, $q \in \{1, 2\}$ and $\alpha, \beta \in \{+, -\}$ over the unit square $[0, 1]^2$ are called *bottleneck lines*.

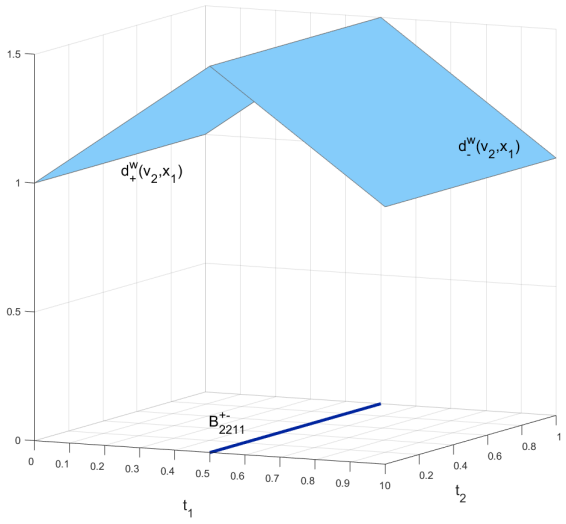


Fig. 6.12: Bottleneck line given by B^{+-}_{2211} for $i = 2$ and $x_1 = (e_{34}, t_1)$, $x_2 = (e_{15}, t_2)$

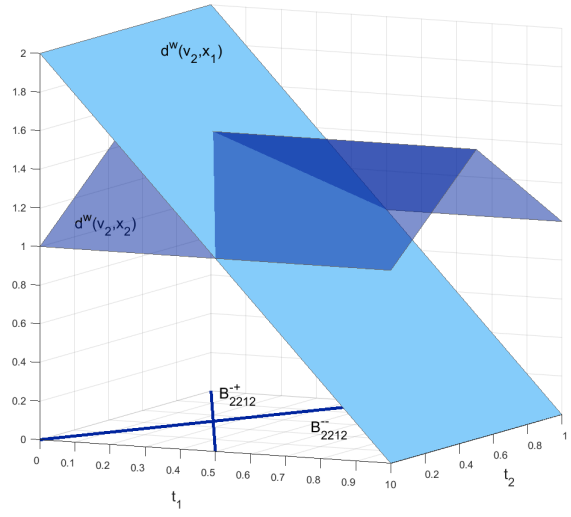


Fig. 6.13: Balance lines of B^{-+}_{2212} and B^{--}_{2212} for $i = 2$ and $x_1 = (e_{12}, t_1)$, $x_2 = (e_{34}, t_2)$

Thus, there are at most two bottleneck lines for every combination v_i, v_j over $[0,1]^2$ in this case because not all equations have to have a solution for $t_q \in [0,1]$. An example of a bottleneck line is given in Figure 6.12 by B^{+-}_{2211} for the edges e_{34} and e_{15} of the graph given in Figure 4.5.

Case 4: $i = j$ and $q \neq r$
Here, the set of breakpoints is described by the set

$$B^{\alpha\beta}_{iiqr} = \left\{ \{x_1, x_2\} \in e_g \times e_h : d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_r) \right\},$$

for $i \in \{1, \ldots, n\}$, $q, r \in \{1, 2\}$ with $q \neq r$ and $\alpha, \beta \in \{+, -\}$. This set contains pairs $\{x_1, x_2\}$ that have the same distance to the node v_i. Because of $B^{\alpha\beta}_{iiqr} = B^{\beta\alpha}_{iirq}$, it can be assumed w.l.o.g. that $i < j$ and $q < r$, i.e., $q = 1$ and $r = 2$. The corresponding equations in $t_1, t_2 \in [0,1]^2$, derived from all possible α-β-combinations to locate the facilities $x_1 = (e_g, t_1)$ and $x_2 = (e_h, t_2)$, are:

$$l_g t_1 - l_h t_2 = d(v_i, v_{a_h}) - d(v_i, v_{a_g}) \tag{6.16}$$

$$l_g t_1 + l_h t_2 = d(v_i, v_{b_h}) - d(v_i, v_{a_g}) + l_h \tag{6.17}$$

$$-l_g t_1 - l_h t_2 = d(v_i, v_{a_h}) - d(v_i, v_{b_g}) - l_g \tag{6.18}$$

$$-l_g t_1 + l_h t_2 = d(v_i, v_{b_h}) - d(v_i, v_{b_g}) - l_g + l_h. \tag{6.19}$$

Thus, the set of breakpoints $B^{\alpha\beta}_{iiqr}$ describes, for a fixed combination of $i \in \{1, \ldots, n\}$ and $\alpha, \beta \in \{+, -\}$, a linear function in $t_1, t_2 \in [0,1]$ over the unit square $[0,1]^2$. As the set of breakpoints in Case 2 equals the set of breakpoints of this case if $i = j$ is also considered for Case 2, the two sets of breakpoints are treated simultaneously for $q \neq r$ in the following where $i \neq j$ is allowed as well as $i = j$.

Definition 6.16 ▶ Balance points
For all v_i, $v_j \in V$, define

$$BP_{ij} = \left\{ x_1, x_2 \in A(G) : w_i d(v_i, x_1) = w_j d(v_j, x_2) \right\}.$$

Set $BP := \bigcup_{i,j \in \{1,\ldots,n\}} BP_{ij}$. The points in BP are called *balance points* of G.

Thus, balance points are pairs of new locations $\{x_1, x_2\}$ where the weighted distance of x_1 to a node v_i equals the weighted distance of x_2 to a node v_j. Note that there is in general an infinite number of balance points for every combination of v_i and v_j with $i, j \in \{1, \ldots, n\}$.

Definition 6.17 ▶ Balance Lines
The lines corresponding to the sets $B^{\alpha\beta}_{ijqr}$ for $i, j \in \{1, \ldots, n\}$, $q, r \in \{1, 2\}$ with $q \neq r$ and $\alpha, \beta \in \{+, -\}$ over the unit square $[0,1]^2$ are called *balance lines*.

An example is given in Figure 6.13 that shows the balance lines described by the sets B^{-+}_{2212}

and B^{--}_{2212} for $x_1 \in e_{12}$ and $x_2 \in e_{34}$ of the example graph given in Figure 4.5. Obviously, for $v_i = v_j$, the lines given by (6.12) to (6.15) equal the lines described by (6.16) to (6.19). Thus, the lines of type (6.16) and (6.19) and the lines of type (6.17) and (6.18) are parallel with a slope of $\pm\frac{w_i l_g}{w_j l_h}$.

Remark 6.18. *Note that the lines described by the equations* (6.8)-(6.19) *are actually line segments as* t_1 *and* t_2 *are restricted to the interval* $[0,1]$ *to describe a point on an edge of the graph. However, the start and endpoint of every line segment lies on a boundary segment of the unit square* $[0,1]^2$ *such that no line segment starts or ends in an arbitrary inner point of the unit square. This will be important later to derive convexity properties.*

The equilibrium, bottleneck and balance lines for fixed nodes $i, j \in \{1, \ldots, n\}$ define a subdivision L_{ijgh} of the unit square $[0,1]^2$ representing a pair of edges $e_g \times e_h$. With $\bigcup_{i,j\in\{1,\ldots,n\}} L_{ijgh}$, the subdivision induced by the lines of all pairs of nodes over $[0,1]^2$ is given. Moreover,

$$L_{gh} = \bigcup_{i,j\in\{1,\ldots,n\}} L_{ijgh} \cup bd([0,1]^2)$$

describes the subdivision induced by all pairs of nodes as well as the boundary segments of the unit square $[0,1]^2$. A maximal closed subset $C \subset [0,1]^2$ of the subdivision L_{gh} that does not intersect with an element of L_{gh} in its interior is called a cell. Every cell is closed such that $bd(C) \subset L_{gh}$ and convex since it is the intersection of half spaces. The set of all cells of the subdivision is called $C(L_{gh})$. The set of all intersection points $u \in [0,1]^2$ of the line segments in L_{gh} are denoted by $V(L_{gh})$. Note that these intersection points are the extreme points of the cells. For an illustration see Figure 6.14. The properties of the weighted distance function $d^w(V,X)$ and the permutation $\sigma \in \Sigma(X)$ on subsets of the subdivision are analysed further in the following.

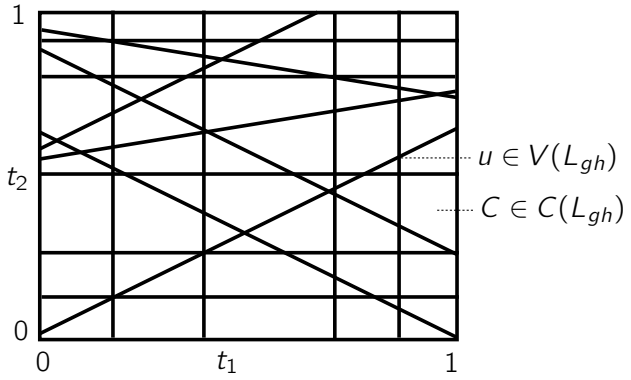


Fig. 6.14: Possible subdivision L_{gh} over $[0,1]^2$ representing $e_g \times e_h$ induced by equilibrium, bottleneck and balance lines

Theorem 6.19 ▶ Linearity of the k-max function

The objective function of the 2-k-max problem with $n \geq 2$ and $k \leq n-2$ on graphs is linear on every cell $C \in C(L_{gh})$ of the subdivision L_{gh}.

Proof. The correctness of Theorem 6.19 will follow as a special case of Theorem 6.28 below and its proof for the special case that $p = 2$. Thus, the proof is omitted at this point and a reference is made to Theorem 6.28 on page 153. ∎

As already recognised for the 1-k-max problem, the linearity of the k-max function depends on two factors: The ordering of the elements of the distance vector and the linearity of the weighted distance functions $d^w(v_i, X)$, $i = 1, \ldots, n$. The proof of Theorem 6.28 below will show that the arrangement of lines L_{gh} results as the union of two other linear arrangements: The subdivision $\bar{L}_{gh}$ that is induced by the equilibrium lines (Case 1) and the balance lines described by the set $B^{\alpha\beta}_{ijqr}$ for $i \neq j$ (Case 2) and the subdivision $\tilde{L}_{gh}$ consisting of the bottleneck lines (Case 3) and the balance lines given by $B^{\alpha\beta}_{iiqr}$ for $i = j$ (Case 4). The permutation $\sigma \in \Sigma(X)$ is constant on each cell $\bar{C} \in C(\bar{L}_{gh})$, i.e., the cells $\bar{C}$ correspond to those regions of points X, where the sorting of the elements of the weighted distance vector $d^w(V, X)$ does not change. Moreover, the cells $\tilde{C} \in C(\tilde{L}_{gh})$ are the linearity regions of the weighted distance functions $d^w(v_i, X)$ for all $i = 1, \ldots, n$.

Remark 6.20. *Let L'_{gh} be the subdivision of the unit square $[0, 1]^2$ resulting only from the boundary segments of the unit square, the equilibrium and the balance lines. In contrast to L_{gh}, the bottleneck lines are not considered (see Figure 6.15 where the deleted lines in comparison to Figure 6.14 are assumed to be the balance lines). Then it can be shown that the* k-max *function is piecewise linear and concave on each cell $C' \in C(L'_{gh})$ of the subdivision L'_{gh}. This result holds since each function $d^w(v_i, X)$, $i = 1, \ldots, n$, is piecewise linear and concave over the unit square $[0, 1]^2$ representing $e_g \times e_h$. Thus, if a distance function $d^w(v_i, X)$ has a bottleneck point on e_g resp. e_h, the corresponding bottleneck line describes the maximum of the function and is therefore not of interest for finding the minimum of the* k-max *function.*

Lemma 6.21 ▶ FDS based on L'_{gh}

At least one optimal solution of the 2-k-max problem with $n \geq 2$ and $k \leq n - 2$ can be found in the set

$$V(L') = \bigcup_{g,h=1,\ldots,m} V(L'_{gh}).$$

Proof. Following Remark 6.20, the k-max function is piecewise linear and concave on each cell C of the subdivision L'_{gh} of $[0, 1]^2$ for all $g, h \in \{1, \ldots, m\}$. Moreover, every cell is convex and compact such that at least one optimal solution has to be attained in an extreme point of L'_{gh}. Thus, the set $V(L'_{gh})$ contains a local optimal solution of the 2-k-max-problem restricted to the current pair of edges e_g and e_h. A global minimum of the k-max function can therefore be found in the set $V(L')$ as the local minimum with the smallest objective function value. ∎

Remark 6.22. *Note that the set*

$$V(L) = \bigcup_{g,h=1,\ldots,m} V(L_{gh}) \supseteq V(L')$$

*is a superset of the finite dominating set $V(L')$ and therefore a finite dominating set for the 2-k-*max *problem itself. In contrast to the set $V(L')$ it also contains intersection points determined by bottleneck lines. These intersection points can be omitted when searching for an optimum for the 2-k-*max *problem, but they are necessary to identify intervals of optimal solutions of the underlying problem. Therefore, the subdivision L_{gh} is analysed further in the following even though L'_{gh} leads to a smaller finite dominating set so far.*

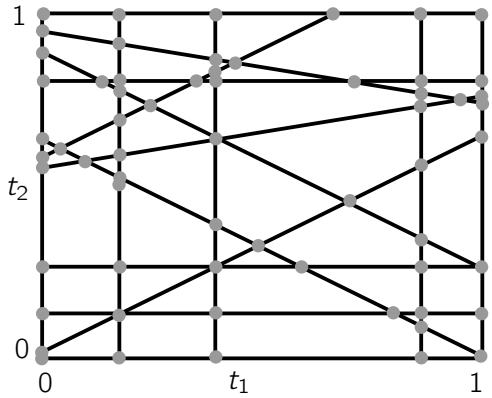


Fig. 6.15: Possible subdivision L'_{gh} over $[0,1]^2$ deduced from L_{gh} in Figure 6.14 by eliminating the bottleneck lines. The intersection points $V(L'_{gh})$ are a subset of the finite dominating set $V(L)$.

For a fixed pair $i,j \in \{1,\ldots,n\}$, $i \neq j$, there are at most eight equilibrium lines and four balance lines for a pair of edges e_g and e_h. As the number of ij-combinations with $i \neq j$ is $\frac{1}{2}n(n-1)$, there are in total at most $6n^2-6n$ line segments describing $\bar{L}_{gh}$. The subdivision $\tilde{L}_{gh}$ consists of at most two bottleneck and four balance lines per pair $i,j \in \{1,\ldots,n\}$, $i=j$. This results in at most $6n$ line segments on $\tilde{L}_{gh}$. Consequently, L_{gh} consists of at most $6n^2 = \mathcal{O}(n^2)$ line segments as it is the union of $\bar{L}_{gh}$ and $\tilde{L}_{gh}$. This subdivision (including all intersection points) can be build up, for example, with the algorithm of Edelsbrunner et al. (1986) which has a complexity of $\mathcal{O}(s^d)$, where s is the number of hyperplanes and d the dimension of the space, i.e., here $\mathcal{O}(n^4)$. A set of $6n^2$ line segments can have at most $18n^4-3n^2$ intersections points, thus the cardinality of $V(L_{gh})$ is $\mathcal{O}(n^4)$. Therefore, the size of the finite dominating set $V(L)$ is $\mathcal{O}(n^4m^2)$. Note that $V(L'_{gh})$ also has a cardinality of $\mathcal{O}(n^4m^2)$, even though it consists of slightly less intersection points.

However, solving the 2-k-max problem using one of the finite dominating sets given in Lemma 6.21 is too expensive in general since the complexity of the evaluation of the candidate points exceeds the complexity of building up all needed subdivisions. Following Algorithm 4, the evaluation of one candidate point needs $\mathcal{O}(n\log(n))$ time, Thus, analysing the whole set of candidate points has a worst case complexity bound of $\mathcal{O}(n^5m^2\log(n))$, which defines the complexity of the whole procedure to find an optimal solution. The problem is that the permutation $\sigma \in \Sigma(X)$ in the candidate points $X \in V(L_{gh})$ is in general not known. It would be helpful to reduce the size of the finite dominating set significantly to have to evaluate less candidate points and thus to reduce the computational effort.

Theorem 6.23 ▶ Objective value defining equilibrium point

Let $X^* = \{x_1^*, x_2^*\}$ be an optimal solution of the 2-k-max problem with $n \geq 2$ and $k \leq n-2$. W.l.o.g., x_1^* is a new facility that determines the optimal objective value z^*. Then it holds that $x_1^* \in EQ$.

Proof. The correctness of Theorem 6.23 follows as a special case of Theorem 6.30 below and its proof for the special case that $p = 2$. Thus, the proof is omitted at this point and a reference is made to Theorem 6.30 on page 155. ∎

Theorem 6.30 shows that the optimal objective value defining facility of the 2-k-max problem has to be an equilibrium point in all optimal solution of the problem. In contrast to that, Theorem 6.3 only proofed that it exists at least one optimal solution for which the optimal objective value defining facility is an equilibrium point.

As a consequence of Theorem 6.23, the finite dominating set $V(L)$ can be restricted to all points $\{x_1, x_2\} \in V(L)$, for which either x_1 or x_2 is an equilibrium point. However, the number of points with this property is still large as the number of intersection points of two non equilibrium lines is very small with respect to the size of $V(L)$. But also this finite dominating set can be reduced further by using the information given by the optimal solutions computed with the recursive approach (see Section 6.2).

Corollary 6.24 ▶ Objective value defining facilities

Let $\mathcal{X}_r$ be the set of optimal solutions of the 2-k-max problem with $p \geq 2$ and $k \leq n-2$ found by Algorithm 10. Moreover, let

$$Y = \left\{x_1^* \in EQ_{ij}, i, j \in \{1, \ldots, n\} : \exists\, X^* = \{x_1^*, x_2^*\} \in \ \mathcal{X}_r \ \text{ with } \ d^w(v_i, x_1^*) = z^*\right\}$$

be the set of all optimal objective function value defining equilibrium points that are an element of an optimal solution found by the recursive approach.

Then, there is no optimal solution $\tilde{X} = \{\tilde{x}_1, \tilde{x}_2\} \notin$ Cand4 of the 2-k-max problem with objective value defining facility $\tilde{x}_1$ for which $\tilde{x}_1 \notin Y$.

Proof. Follows directly as a special case for $p = 2$ of the proof of Theorem 6.32: The facility $\tilde{x}_2 \notin EQ \cup V$ of an optimal solution $\tilde{X} = \{\tilde{x}_1, \tilde{x}_2\} \notin$ Cand4 can be moved into an equilibrium point or a node by solving a center problem on $V_2 = \{v_i \in V : w_i d(v_i, \tilde{x}_2) < w_i(v_i, \tilde{x}_1)\}$ without changing the objective value of z^*. Thus, afterwards it holds that $\tilde{x}_2 \in EQ \cup V$ and the solution is still optimal and thus $\tilde{x}_1 \in Y$, following Remark 6.7. ∎

As a consequence of Theorem 6.23 and Corollary 6.24, the set Y of objective function value defining equilibrium points of all optimal solutions of the 2-k-max problem are known after the application of the recursive approach. Alternative optimal solutions X not in $\mathcal{X}_r$ can therefore only be in a set

$$\mathcal{X}(x_1^*) = \{X = \{x_1^*, x_2\} : x_2 \in A(G) \ \wedge \ d^w(v_{\sigma(k)}, X) = z^*\} \quad \text{for} \quad x_1^* \in Y$$

of optimal solutions that have the same objective function value defining facility x_1^*.

Now, let $X^* = \{x_1^*, x_2^*\} \in \mathcal{X}_r$ be an optimal solution of the 2-k-max problem where

$$x_1^* = (e_{g^*}, t_1^*) \quad \text{with} \quad e_{g^*} = (v_{a_{g^*}}, v_{b_{g^*}}) \in E,\ g^* \in \{1, \ldots, m\}.$$

Moreover, let

$$x_1^* \in EQ_{cd}^{a_{g^*} b_{g^*}} \quad \text{for fixed} \quad c, d \in \{1, \ldots, n\}$$

be the new facility defining the optimal objective function value z^*. As this solution is found by the recursive approach, it holds that $x_1^* \in Y$. Since the edge e_{g^*} on which the optimal new location x_1^* lies is known, not all $m(m-1)/2$ pairs of edges e_g, e_h with $g, h \in \{1, \ldots, m\}$ have to be considered to build up a subdivision L_{gh}. Only the combinations $e_{g^*} \times e_h$ with fixed edge e_{g^*} corresponding to an $x_1^* \in Y$ have to be considered. Let $h \in \{1, \ldots, m\}$ be arbitrary but fixed in the following.

As the location of $x_1^* \in EQ_{cd}^{a_{g^*} b_{g^*}}$ is already fixed, all alternative optimal solutions of the set $\mathcal{X}(x_1^*)$ have to lie on the equilibrium line described by x_1^* on the unit square $[0,1]^2$ representing $e_{g^*} \times e_h$. This equilibrium line is denoted by $eq_{g^*h}(x_1^*)$. Obviously, all intersection points $\{x_1, x_2\}$ of the finite dominating set $V(L)$ for which $x_1 \neq x_1^*$ can be neglected to find further optimal solutions in $\mathcal{X}(x_1^*)$. Thus, let $L_{g^*h}(x_1^*)$ be the subdivision of the unit square $[0,1]^2$ representing $e_{g^*} \times e_h$ that consists of all balance lines, all equilibrium and all bottleneck lines corresponding to edge e_h and the single equilibrium line $eq_{g^*h}(x_1^*)$ corresponding to edge e_{g^*}. Then, $C(L_{g^*h}(x_1^*))$ is the set of all cells of the subdivision $L_{g^*h}(x_1^*)$. The set $V(L_{g^*h}(x_1^*))$ describes the set of all intersection points of $L_{g^*h}(x_1^*)$ and

$$V^{eq}(L_{g^*h}(x_1^*)) \subset V(L_{g^*h}(x_1^*))$$

is the set of intersection points of $eq_{g^*h}(x_1^*)$ with one of the other line segments of the subdivision $L_{g^*h}(x_1^*)$. An example can be seen in Figure 6.16.

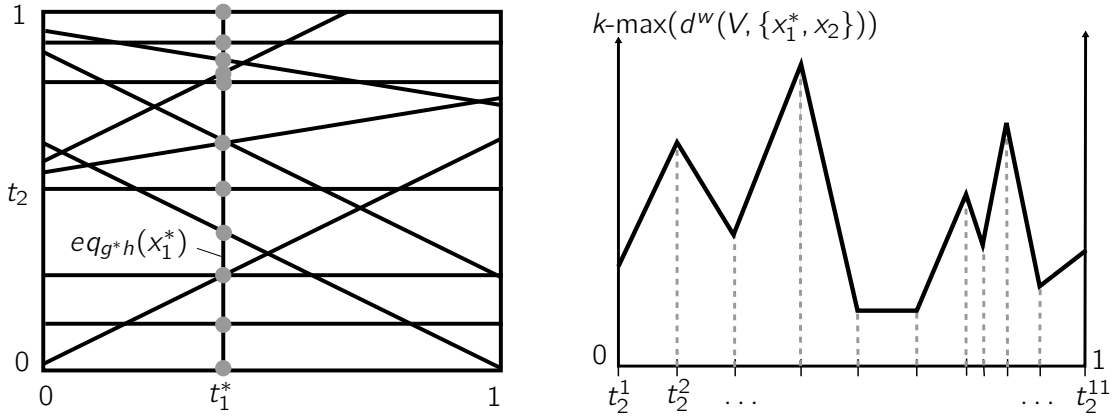


Fig. 6.16: Subdivision $L_{g^*h}(x_1^*)$ of $[0,1]^2$ for $e_{g^*} \times e_h$ with intersection points $V^{eq}(L_{g^*h}(x_1^*))$ (grey) on $eq_{g^*h}(x_1^*)$ (left). k-max($d^w(V, X)$) over eq_{g^*h} with breakpoints in $x_2^f = (e_h, t_2^f)$, $f = 1, \ldots, |V^{eq}(L_{g^*h}(x_1^*))|$ (right).

Corollary 6.25 ▶ Improved FDS based on $L_{g^*h}(x_1^*)$

Let $X^* = \{x_1^*, x_2^*\} \in \mathcal{X}_r$ with $x_1^* = (e_{g^*}, t_1^*) \in Y$, $g^* \in \{1, \ldots, m\}$, fixed. If the 2-k-max problem with $n \geq 2$ and $k \leq n-2$ has more than one optimal solution $\{x_1, x_2\} \in \mathcal{X}(x_1^*)$ with $x_1 = x_1^*$, at least one of these alternative optimal solutions can be found in the set

$$V^{eq}(L(x_1^*)) = \bigcup_{h=1,\ldots,m} V^{eq}(L_{g^*h}(x_1^*)).$$

As a direct consequence of Corollary 6.25, at least one alternative optimal solution $\{\bar{x}_1, x_2\} \in \mathcal{X}(\bar{x}_1)$ (if existent) for every $\bar{x}_1 \in Y$ with $\{\bar{x}_1, \bar{x}_2\} \in \mathcal{X}_r$ is an element of the set

$$V^{eq}(L(Y)) = \bigcup_{\bar{x}_1 \in Y} V^{eq}(L(\bar{x}_1)).$$

Note that the finite dominating set $V^{eq}(L(Y))$ yields more optimal solutions (if existent) than the set $\mathcal{X}_r$ of optimal solutions generated by the recursive approach. For example, the alternative optimal solution $\bar{\bar{X}}^1$ of the 2-1-max problem given in Example 6.11 is an element of $V^{eq}(L(Y))$ but not of $\mathcal{X}_r$. As the solutions $X \in \mathcal{X}_r$ lie in $EQ \cup (EQ \times V)$, they are intersection points of at least two equilibrium lines or an equilibrium line and a boundary line of a unit square and thus in $V^{eq}(L(Y))$. However, there may still be optimal solutions that can not be found in the given finite dominating set but they can easily be computed when all optimal solutions in $V^{eq}(L(x_1^*))$ are known.

Theorem 6.26 ▶ All optimal solutions

Let $X^* = \{x_1^*, x_2^*\} \in \mathcal{X}_r$ with $x_1^* = (e_{g^*}, t_1^*) \in Y$, $g^* \in \{1, \ldots, m\}$, fixed. Moreover, let $X', X'' \in V^{eq}(L_{g^*h}(x_1^*))$, $h \in \{1, \ldots, m\}$, be two optimal solutions of the 2-k-max problem for $n \geq 2$, $k \leq n-2$, and $X', X'' \in C$ for a cell $C \in C(L_{g^*h}(x_1^*))$. Then, all pairs

$$\{x_1^*, x_2\} \quad \text{with} \quad x_2 \in bd(C) \cap eq_{g^*h}(x_1^*)$$

are optimal solutions of the 2-k-max problem.

Proof. As the k-max function with fixed $x_1^* \in Y$ is piecewise linear over the edge e_h (see Figure 6.16), and the points X' and X'' are consecutive breakpoints of this function that both yield the optimal objective function value, the statement follows. See also Theorem 6.32 below on page 158. ∎

As a consequence of Theorem 6.26, not only extreme points of cells of a subdivision $L_{g^*h}(x_1^*)$ can be optimal but also whole line segments corresponding to a boundary of a cell. However, a complete cell of the subdivision L_{gh} can never be optimal as at least one new facility has to be located in an equilibrium point (see Theorem 6.30). This property is satisfied only on the corresponding equilibrium lines which do not lie in the interior of a cell of the subdivision $L_{g^*h}(x_1^*)$ by definition.

Now, all (maybe infinitely many) optimal solutions of the 2-k-max problem can be obtained

by building up the subdivisions $L_{g^*h}(x_1^*)$ in $[0,1]^2$ for all $h = \{1,\dots,m\}$ and all $x_1^* \in Y$, finding the intersection points of the equilibrium line $eq_{g^*h}(x_1^*)$ with the other line segments and evaluating the k-max function in all these candidate points. The intersection points leading to the smallest objective function value are then optimal for the problem. Optimal boundary lines of cells can be found as the line segments between two consecutive optimal extreme points of the same cell. Note that it is sufficient to use a version of the recursive approach that finds only one optimal solutions for every optimal objective value defining equilibrium point. Thus, the recursion can be terminated on the recursion levels 2 to p when the first feasible solution is found. The approach applying the local analysis based on the results of the recursive approach is summarised in Algorithm 11.

Algorithm 11 Local Analysis to find all optimal solutions of the 2-k-max problem

Input: Graph $G = (V, E)$ with $w_i > 0 \;\forall\; i = 1,\dots,n$, $k \in \{1,\dots,n-2\}$, optimal objective value z^*, set Y of z^*-defining equilibrium points found by the recursive approach.

Set $z_b := \infty$, $z_0 := \infty$
for all $x_1^* = (e_{g^*}, t_1^*) \in Y$ **do** ▷ All relevant optimal objective value defining facilities
  for all $h = 1,\dots,m$ with $h \neq g^*$ **do** ▷ All pairs of edges e_h and fixed e_{g^*}
    Derive the subdivision $L_{g^*h}(x_1^*)$ of $[0,1]^2$ and the set $V^{eq}(L_{g^*h}(x_1^*))$ of intersection points on $eq_{g^*h}(x_1^*)$
    Sort the pairs $X = \{x_1^*, x_2\} \in V^{eq}(L_{g^*h}(x_1^*))$ increasingly in x_2
    for all $X_i \in V^{eq}(L_{g^*h}(x_1^*))$ **do**
      Compute the objective value $z_i = k\text{-max}(d^w(V, X_i))$ of X_i
      if $z_i = z^*$ **then** ▷ X_i is optimal
        if $z_i = z_{i-1}$ **then** ▷ Edge segment $[X_i, X_{i-1}]$ is optimal
          $\mathcal{X} = \mathcal{X} \cup \{[X_{i-1}, X_i]\} \setminus X_{i-1}$
        else ▷ Edge segment is not optimal
          $\mathcal{X} = \mathcal{X} \cup \{X_i\}$

Output: Set $\mathcal{X}$ of all optimal solutions of the 2-k-max problem on G.

Algorithm 11 has a worst case complexity of $\mathcal{O}(m^2 n^5 \log(n))$. The first loop on the elements of Y needs $\mathcal{O}(mn^2)$ iterations as this is the number of equilibrium points and therefore the maximum number of elements in Y. Note that the cardinality of Y will generally be much smaller in practice. The second loop considers all edges e_h, $h = 1,\dots,m$ and $h \neq g^*$, to build up the subdivision L_{g^*h} in $\mathcal{O}(m)$ iterations. The sets $B_{ijqr}^{\alpha\beta}$ for fixed $i, j \in \{1,\dots,n\}$, $q, r \in \{1,2\}$ and $\alpha, \beta \in \{+,-\}$ and their linear representations as equilibrium, balance or bottleneck lines can be determined in constant time. Since there are at most 12 line segments for every combination of $v_i, v_j \in V$ on each unit square, the subdivision $L_{g^*h}(x_1^*)$ consists of at most $\mathcal{O}(n^2)$ line segments. Thus, $L_{g^*h}(x_1^*)$ can be constructed in $\mathcal{O}(n^2)$

time. As only the intersection points of the equilibrium line $eq_{g^*h}(x_1^*)$ with all other lines are needed, $V^{eq}(L_{g^*h}(x_1^*))$ can be computed in $\mathcal{O}(n^2)$. Moreover, $|V^{eq}(L_{g^*h}(x_1^*))| = \mathcal{O}(n^2)$. Sorting the intersection points increasingly w.r.t. their x_2-components needs $\mathcal{O}(n^2 \log(n))$ time. Finally, the evaluation of the candidates takes $\mathcal{O}(n \log(n))$ time per candidate and thus $\mathcal{O}(n^3 \log(n))$ time for all candidates of a fixed subdivision. For this purpose, it is necessary to store in each intersection point the information on the defining nodes. Thus, the stated overall complexity follows.

Of course, this complexity result only holds if the set Y is already known. Otherwise, the complexity of Algorithm 11 is $\mathcal{O}(m^2 n^6)$ which is dominated by the complexity of Algorithm 10 to determine Y. Note that the whole procedure can be parallelised easily as the construction of the subdivision for a fixed pair of edges does not depend on the other pairs of edges. This leads to a significant improvement of the computation time in practice.

Furthermore, in practice it is often not necessary to compute really *all* optimal solutions of a problem as the interest to locate a new facility could be particularly high in a certain region whereas other areas are less wanted. Then, it is of course also possible to analyse only selected optimal solutions of $\{x_1^*, x_2\} \in \mathcal{X}_r$ for further alternative optimal locations that are of particular interest for the decision maker. In this case, i.e., for the evaluation of only one $x_1 \in Y$, the worst case bound of the local analysis is $\mathcal{O}(mn^3 \log(n))$. If moreover only the neighbourhood of x_2 on a certain edge h is of interest to find alternative optimal solutions, the computation on this edge takes only $\mathcal{O}(n^3 \log(n))$.

Example 6.27 (Continuation of Example 6.11)**.** *The approach of Algorithm 11 is applied to the* 2-1-max *problem on the graph given in Example 4.5. An optimal solution resulting from the recursive approach is*

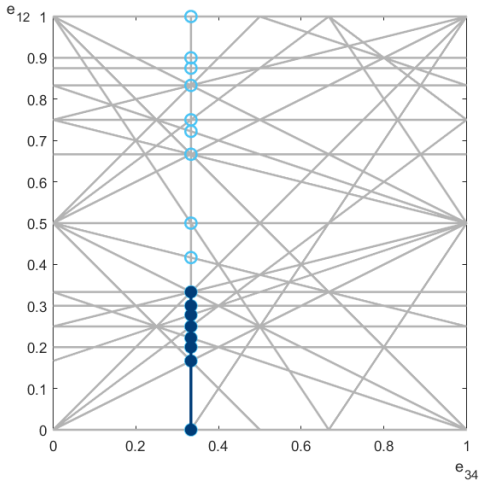
$$X^1 = \{x_1^1, x_2^1\} \quad \textit{with} \quad x_1^1 = \left(e_{34}, \frac{1}{3}\right) \quad \textit{and} \quad x_2^1 = \left(e_{12}, \frac{1}{3}\right),$$

with optimal objective function value $z^1 = \frac{4}{3}$*. As this solution is also found with changed roles of* x_1^1 *and* x_2^1*, both equilibrium points are objective function value defining and therefore*

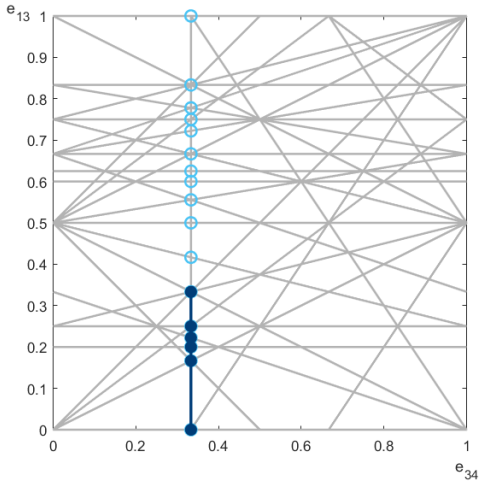
$$Y = \left\{ \left(e_{34}, \frac{1}{3}\right), \left(e_{12}, \frac{1}{3}\right) \right\}.$$

The solution approach is shown for the first objective function value defining facility $\left(e_{34}, \frac{1}{3}\right)$ *in the following.*

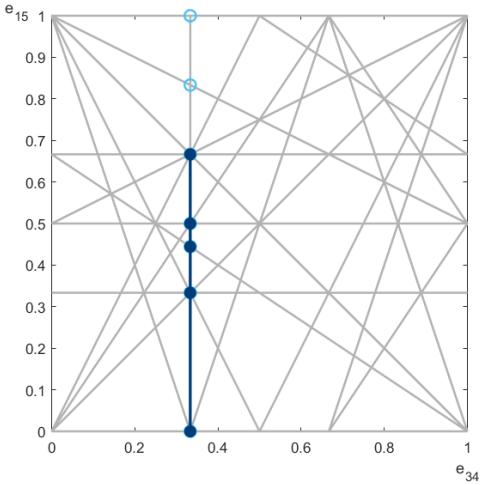
As $x_1^1 = \left(e_{34}, \frac{1}{3}\right)$*, it holds that* $g^* = 6$ *with* $e_6 = (v_3, v_4)$ *is fixed now. The subdivisions* $L_{6h}(x_1^1)$ *for* $h = 1, \ldots, 7$ *except for* $h = 6$ *(see Lemma 6.13) are shown in Figure 6.17. The analysed candidate intersection points are marked with an empty circle, the intersection points and the line segments that contain the local minima of the subdivision are marked in mid-blue (see Figure 6.17(d)-(f)). The local minima that are also globally optimal are drawn in dark blue (see Figure 6.17(a)-(c)). Therefore, the alternative optimal solutions with* x_1^1



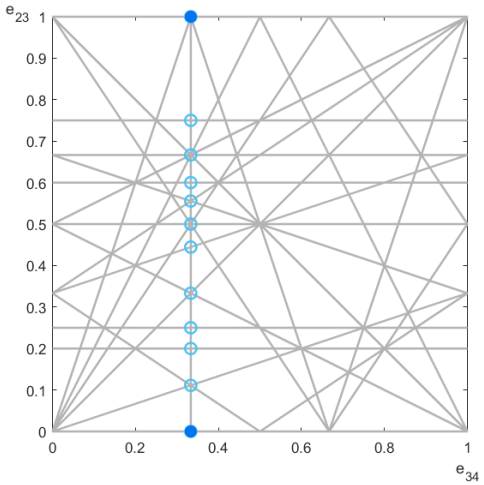
(a) Subdivision $L_{61}(x_1^*)$



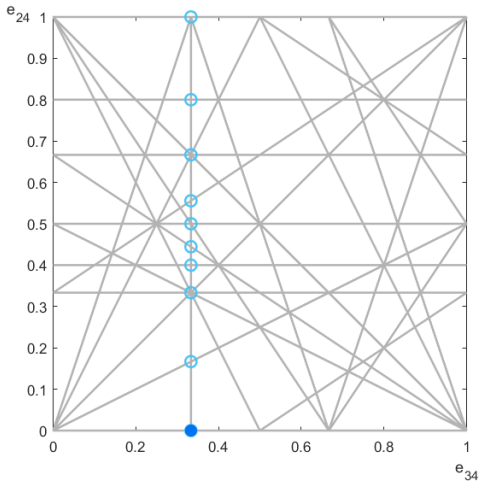
(b) Subdivision $L_{62}(x_1^*)$



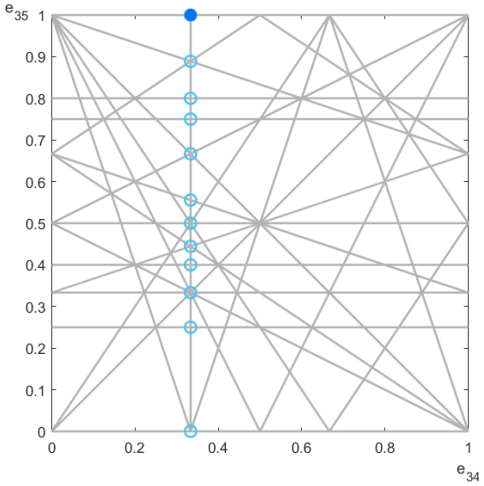
(c) Subdivision $L_{63}(x_1^*)$



(d) Subdivision $L_{64}(x_1^*)$



(e) Subdivision $L_{65}(x_1^*)$



(f) Subdivision $L_{67}(x_1^*)$

Fig. 6.17: Subdivisions $L_{g^*h}(x_1^*)$ (light gray) with local minima (light blue) and global minima (dark blue). The latter are the optimal solutions in $\mathcal{X} = \mathcal{X}(x_1^*)$ of the 2-1-max problem.

as objective function value defining facility are given by $\mathcal{X}(x_1^1) = \{x_1^1, x_2^1\}$ *with*

$$x_2^1 \in \left[(e_{12}, 0), \left(e_{12}, \frac{1}{3}\right)\right] \cup \left[(e_{13}, 0), \left(e_{13}, \frac{1}{3}\right)\right] \cup \left[(e_{15}, 0), \left(e_{15}, \frac{2}{3}\right)\right].$$

For an illustration in the graph see Figure 6.18.

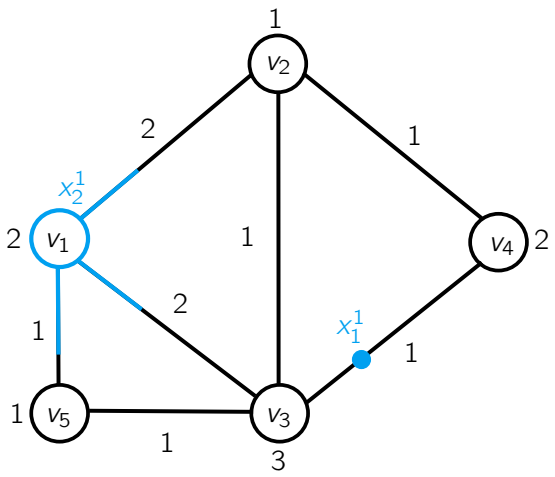


Fig. 6.18: Set $\mathcal{X}$ for $k = 1$ with $z^* = \frac{4}{3}$ for Example 6.27

Note that the solution $\bar{X}^1 = \{\bar{x}_1^1, \bar{x}_2^1\} \notin$ Cand4 *with* $\bar{x}_2^1 = \left(e_{15}, \frac{2}{3}\right)$ *given in Example 6.11 is also contained in the set* $\mathcal{X}(x_1^1)$*. Moreover, in this example it holds that* $\mathcal{X} = \mathcal{X}(x_1^1)$ *as the other objective value defining facility* $\left(e_{12}, \frac{1}{3}\right) \in Y$ *just yields the optimal solution* X^1 *that is also an element of* $\mathcal{X}(x_1^1)$*.*

6.3.2 The p-facility Case

In the following, the results of the last section for $p = 2$ new facilities will be transferred to the general case of $p \geq 2$ new facilities to locate. Basically, the same ideas as for the 2-facility case are used and most statements can be applied analogously for the more general case.

Let $X = \{x_1, \ldots, x_p\}$ be the set of p new facilities with $x_1, \ldots, x_p \in A(G)$. Moreover, let $x_q = (e_{g_q}, t_q)$ and $e_{g_q} = (v_{a_{g_q}}, v_{b_{g_q}})$ for all $q = 1, \ldots, p$ and $g_q \in \{1, \ldots, m\}$. For the weighted distance between v_i and X it holds

$$d^w(v_i, X) = \min\left\{d_+^w(v_i, x_1), d_-^w(v_i, x_1), \ldots, d_+^w(v_i, x_p), d_-^w(v_i, x_p)\right\}.$$

As for the 2-facility case, the ordering of the elements of the vector of weighted distances $d^w(V, X)$ may change whenever at least two of its elements are equal, i.e., if

$$d_\alpha^w(v_i, x_q) = d_\beta^w(v_j, x_r) \quad \text{for} \quad i, j \in \{1, \ldots, n\},\ q, r \in \{1, \ldots, p\},\ \alpha, \beta \in \{+, -\}. \quad (6.20)$$

Hence, all points satisfying this equation may correspond to a breakpoint of the objective

function of the p-k-max problem. The aim is to derive a finite dominating set for the p-k-max problem based on a subdivision of the unit hypercube $[0,1]^p$ analogously to the 2-facility case.

Instead of pairs of edges, now p-tuples of edges $e_{g_1}, \ldots, e_{g_p} \in E$, each edge identified with the unit interval $[0,1]$, are considered such that the Cartesian product $e_{g_1} \times \ldots \times e_{g_p}$ is represented by the unit hypercube $[0,1]^p$. Let

$$g = \{g_1, \ldots, g_p\} \subset \{1, \ldots, m\}^p$$

be such a fixed p-tuple of edges. Note that (contrary to the 2-facility case) it is not excluded that several edges are equal as now more than one new facility may lie on one certain edge in an optimal solution. The p new facilities $x_1 = (e_{g_1}, t_1), \ldots, x_p = (e_{g_p}, t_p)$ are considered to be located on the edges $e_{g_1}, \ldots, e_{g_p}$ in the following. For the candidate X the set notation will be used to emphasise the interchangeability of the individual facilities. Nevertheless, as in the case $p = 2$, the notation $X \in e_{g_1} \times \ldots \times e_{g_p}$ will be used. To avoid duplication of candidates, it can be assumed without loss of generality that $g_j \leq g_{j+1}$ for $j = 1, \ldots, p-1$.

Due to the one to one correspondence between a point $x = (e, t)$ on a fixed edge e and the parameter $t \in [0,1]$, feasible solutions can be represented equivalently by the candidate X being an element of the Cartesian product $e_{g_1} \times \ldots \times e_{g_p}$ and by the corresponding parameters $t_1, \ldots, t_p$ varying on the unit hypercube $[0,1]^p$. Hence, every point of the subdivision $[0,1]^p$ corresponds to the location of p new facilities on the edges of G.

To derive the subdivision of the unit hypercube $[0,1]^p$ on which cells the k-max function is linear, the possible breakpoints of the function $d^w(V, X)$ have to be analysed by considering the four cases w.r.t. the values of $i, j \in \{1, \ldots, n\}$ and $q, r \in \{1, \ldots, p\}$ of equation (6.20).

Case 1: $i \neq j$ and $q = r$,
i.e., $d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_q)$ for $i, j \in \{1, \ldots, n\}$, $i \neq j$, $q \in \{1, \ldots, p\}$ and $\alpha, \beta \in \{+, -\}$. Thus, the set of breakpoints is given by

$$B^{\alpha,\beta}_{ijqq} = \left\{ \{x_1, \ldots, x_p\} \in e_{g_1} \times \ldots, \times e_{g_p} : x_q = EQ^{a_{g_q} b_{g_q}}_{ij}(\alpha, \beta) \right\},$$

where $EQ^{a_{g_q} b_{g_q}}_{ij}(\alpha, \beta)$ denotes the equilibrium point on edge e_{g_q} given by the equation $d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_q)$ for a fixed combination of $\alpha, \beta \in \{+, -\}$. For all combinations of α and β, the set $B^{\alpha,\beta}_{ijqq}$ describes the equilibrium points $EQ^{a_{g_q} b_{g_q}}_{ij}$ of the nodes $v_i, v_j \in V$ on the edge e_{g_q}. In the following, the hyperplanes described by these equations are called *equilibrium planes*.

Case 2: $i \neq j$ and $q \neq r$,
The set of breakpoints in this case is given by

$$B^{\alpha\beta}_{ijqr} = \left\{ \{x_1, \ldots, x_p\} \in e_{g_1} \times \ldots \times e_{g_p} : d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_r) \right\},$$

for $i, j \in \{1, \ldots, n\}$ with $i \neq j$, $q, r \in \{1, \ldots, p\}$ with $q \neq r$ and $\alpha, \beta \in \{+, -\}$. The set contains all p-tuples that have two elements on the edges e_{g_q} and e_{g_h}, respectively, that

have the same weighted distance to two different nodes v_i and v_j. The hyperplanes defined by these equations are called *balance planes*.

Case 3: $i = j$ and $q = r$,
i.e., $d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_q)$ such that $\alpha \neq \beta$, $\alpha, \beta \in \{+, -\}$, $i \in \{1, \ldots, n\}$, $q \in \{1, \ldots, p\}$. W.l.o.g. it can be assumed that $\alpha = +$ and $\beta = -$ because here it clearly does not make sense to choose the same value for the parameters α and β. The set of breakpoints for a fixed $q \in \{1, \ldots, p\}$ is then given by

$$B^{+-}_{iiqq} = \left\{ \{x_1, \ldots, x_p\} \in e_{g_1} \times \ldots \times e_{g_p} : x_q = BN_i^{a_{g_q} b_{g_q}} \right\}.$$

For fixed values of $i \in \{1, \ldots, n\}$ and $q \in \{1, \ldots, p\}$, the set B^{+-}_{iiqq} describes the bottleneck point $BN_i^{a_{g_q} b_{g_q}}$ of node v_i on the edge e_{g_q}. The hyperplanes defined by these equations are called *bottleneck planes*.

Case 4: $i = j$ and $q \neq r$
The set of breakpoints in this case is described by the set

$$B^{\alpha\beta}_{iiqr} = \left\{ \{x_1, \ldots, x_p\} \in e_{g_1} \times \ldots \times e_{g_p} : d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_r) \right\},$$

for $i \in \{1, \ldots, n\}$, $q, r \in \{1, \ldots, p\}$ with $q \neq r$ and $\alpha, \beta \in \{+, -\}$. Due to the similarity to Case 2, these hyperplanes are also named *balance planes*.

For an illustration of the four cases see Figure 6.10 to Figure 6.13, where the special case $p = 2$ are shown. Due to the symmetry in i, j, in q, r, and in α, β, it can be assumed w.l.o.g. that $i \leq j$ and $q \leq r$ in all four cases. By considering all combinations of α and β in every case, the resulting equations describing the locations of the new facilities $x_1, \ldots, x_p$ are completely analogous to equations (6.8)-(6.19) for the 2-facility case. If $p > 2$, then these equations describe hyperplanes in $\mathbb{R}^p$, restricted to the unit hypercube $[0, 1]^p$. In Case 1 and Case 3, the equations only depend on t_q for a fixed value of $q \in \{1, \ldots, p\}$ and thus the described hyperplanes are parallel to the face of $[0, 1]^p$ for which $t_q = 0$. In Case 2 and Case 4, the resulting linear functions with known slope depend on two parameters $t_q, t_r \in [0, 1]$, $q, r \in \{1, \ldots, p\}$. As $t_i \in [0, 1]$ for $i \in \{1, \ldots, p\}$, not every equation leads to a non-empty intersection with the unit hypercube such that only an upper bound on the number of hyperplanes intersecting $[0, 1]^p$ is known.

Let $g = (g_1, \ldots, g_p)$ be the vector of indices of the considered edges $e_{g_1} \times \ldots \times e_{g_q}$ with $g_1 \leq g_2 \leq \ldots \leq g_p$. The equilibrium, bottleneck and balance hyperplanes for fixed nodes $i, j \in \{1, \ldots, n\}$ yield a subdivision L_{ijg} of the unit hypercube. By

$$\bigcup_{i,j \in \{1,\ldots,n\}} L_{ijg},$$

the subdivision induced by the hyperplanes for all pairs of nodes over $e_{g_1} \times \ldots \times e_{g_q}$ is described. Then,

$$L_g = \bigcup_{i,j \in \{1,\ldots,n\}} L_{ijg} \cup bd\left([0, 1]^p\right)$$

is the subdivision of $[0,1]^p$ induced by all pairs of nodes as well as the boundary segments of $[0,1]^p$. For an illustration of L_g see Figure 6.14, where the special case for $p = 2$ is shown.

For $j \in \{1,\dots,p\}$, a *j-flat* of the arrangement of these hyperplanes is the affine convex hull of $j+1$ affinely independent points. Moreover, each flat may be divided into smaller sections by the other hyperplanes that do not contain the flat. These sections are called *faces* of the arrangement. A face is a j-face if its dimension is j. In particular, a p-face is called a *cell* $C \subset [0,1]^p$ of the arrangement and the set $C(L_g)$ denotes the set of all cells of L_g. The extreme points of a cell $C \in C(L_g)$ are 0-faces, which are also called *vertices*. The set of all vertices of L_g is denoted by $V(L_g)$. The (p-1)-faces are also called *facets* of the arrangement. For more information on arrangements of hyperplanes see Edelsbrunner (1987). Now, the Theorem 6.19 can be generalised to the case $p \geq 2$.

Theorem 6.28 ▶ Linearity of the k-max function

The objective function of the p-k-max problem on graphs is linear on every cell $C \in C(L_g)$ of the subdivision L_g of $[0,1]^p$.

Proof. The hyperplanes that define the subdivision L_g of the unit hypercube are constructed from all $X = \{x_1,\dots,x_p\}$ that satisfy the equation

$$d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_r) \quad \text{for some} \quad i,j \in \{1,\dots,n\},\ q,r \in \{1,\dots,p\},\ \alpha,\beta \in \{+,-\}.$$

Let $\bar{L}_g$ be the subdivision of the unit hypercube $[0,1]^p$ representing $e_{g_1} \times \ldots \times e_{g_p}$ which is induced by the boundary segments of $[0,1]^p$, the equilibrium planes (Case 1) and the balance planes for $i \neq j$ (Case 2). Moreover, let $C(\bar{L}_g)$ be the set of all cells of this subdivision. Then, all $X = \{x_1,\dots,x_p\}$ that satisfy either that

$$d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_q) \quad \text{or that} \quad d^w_\alpha(v_i, x_q) = d^w_\beta(v_j, x_r)$$

for $i,j \in \{1,\dots,n\}$ with $i \neq j$, $q,r \in \{1,\dots,p\}$ with $q \neq r$ and $\alpha,\beta \in \{+,-\}$, lie on the boundary of a cell $\bar{C} \in C(\bar{L}_g)$. Thus, the permutation $\sigma \in \Sigma(X)$, that sorts $d^w(V,X)$ in non-increasing order, only changes by crossing a boundary face of a cell of $\bar{L}_g$ because $d^w(V,X)$ is continuous and has at least two equal elements only for a $X \in bd(\bar{C})$. Thus, the permutation $\sigma \in \Sigma(X)$ is constant on every cell $\bar{C} \in C(\bar{L}_g)$.

Now, let $\tilde{L}_g$ be the subdivision of the unit hypercube representing $e_{g_1} \times \ldots \times e_{g_p}$ which is induced by the boundary segments of $[0,1]^p$, the bottleneck planes (Case 3) and the balance planes for $i = j$ (Case 4). The set of all cells of the subdivision is called $C(\tilde{L}_g)$. The corresponding equations that describe the boundaries of the cells $\tilde{C}$ of $\tilde{L}_g$ are

$$d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_q) \quad \text{with} \quad \alpha \neq \beta \quad \text{and} \quad d^w_\alpha(v_i, x_q) = d^w_\beta(v_i, x_r)$$

for $i \in \{1,\dots,n\}$, $q,r \in \{1,\dots,p\}$ with $q \neq r$ and $\alpha,\beta \in \{+,-\}$. Moreover, they describe the breakpoints of the distance functions $d^w(v_i,X)$, $i = 1,\dots,n$ (see Kalcsics, 2011). Hence, the functions $d^w(v_i,X)$, $i = 1,\dots,n$, are linear over every cell $\tilde{C} \in C(\tilde{L}_g)$.

Every cell $C \in C(L_g)$ of the subdivision given by all equilibrium, bottleneck and balance

planes is formed as the intersection of a cell $\bar{C} \in C(\bar{L}_g)$ and a cell $\tilde{C} \in C(\tilde{L}_g)$, i.e.,

$$C = \bar{C} \cap \tilde{C} \quad \text{for} \quad \bar{C} \in C(\bar{L}_g),\ \tilde{C} \in C(\tilde{L}_g).$$

Consequently, the k-max function over $C \in C(L_g)$ can be written as

$$k\text{-max}(d^w(V,X)) = \lambda_1 d^w(v_{\sigma(1)},X) + \ldots + \lambda_k d^w(v_{\sigma(k)},X) + \ldots + \lambda_n d^w(v_{\sigma(n)},X)$$

with $\lambda_k = 1$ and $\lambda_i = 0$ for $i \neq k$ on every cell $C \in C(L_g)$, where the permutation σ is constant and the functions $d^w(v_i,X)$, $i = 1,\ldots,n$, are linear. Therefore, the k-max function is linear over every cell $C \in C(L_g)$. ∎

Note that, similar to the 2-facility case, it can be shown that the k-max function is piecewise linear and concave on each cell $C' \in C(L'_g)$, where L'_g is the subdivision of $[0,1]^p$ that is defined only by the boundary faces of $[0,1]^p$ and the equilibrium and the balance planes. This is due to the fact that bottleneck planes describe maxima of the weighted distance functions and are therefore not of interest for finding a minimum.

The finite dominating set resulting from the above considerations is given in Lemma 6.29.

Lemma 6.29 ▶ FDS based on L_g

At least one optimal solution of the p-k-max problem with $n \geq 2$ and $k \leq n - p$ can be found in the set

$$V(L') = \bigcup_{g \subset \{1,\ldots,m\}^p} V(L'_g),$$

where $g = \{g_1,\ldots,g_p\}$ with $g_j \leq g_{j+1}$ for $j = 1,\ldots,p-1$.

Proof. Following Theorem 6.28, the k-max function is linear on each cell C of the subdivision L_g of the unit hypercube for all $g = \{g_1,\ldots,g_p\}$. As the bottleneck planes describe maxima of the weighted distance functions, it follows that the k-max function is piecewise linear and concave on each cell $C' \in C(L'_g)$. Moreover, every cell is convex and compact. Thus, a local minimum of the k-max function has to be attained on the boundary of a cell $C' \in C(L'_g)$ for every $g = \{g_1,\ldots,g_p\} \subset \{1,\ldots,m\}^p$.

In particular, the set $V(L'_g)$ contains a local optimal solution of the p-k-max-problem restricted to the current p-tuple $e_{g_1} \times \ldots \times e_{g_p}$. A global minimum of the k-max function can therefore be found in the set $V(L')$ as a local minimum with the smallest objective function value. ∎

Note that the set $V(L) = \bigcup_{g \subset \{1,\ldots,m\}^p} V(L_g)$ is of course also a finite dominating set for the p-k-max problem as it is a superset of $V(L')$.

For a fixed vector $g \subset \{1,\ldots,m\}^p$ of edge indices, there exist at most $\mathcal{O}(n^2p)$ equilibrium planes, $\mathcal{O}(np)$ bottleneck planes and $\mathcal{O}(n^2p^2)$ balance plane. This adds up to at most $\mathcal{O}(n^2p^2)$ hyperplanes that constitute L_g. To determine the number of candidates resulting from the subdivision L_g, the number of 0-faces of the arrangement is needed. Following Edelsbrunner (1987), the number $f_j^p(b)$ of j-faces resulting from a non-simple arrangement

of b hyperplanes in $\mathbb{R}^p$ is bounded by

$$f_j^p(b) = \sum_{i=0}^{j} \binom{p-i}{j-i} \binom{b}{p-i}. \tag{6.21}$$

Thus, the number of 0-faces in this case of $\mathcal{O}(n^2p^2)$ hyperplanes is

$$f_0^p(n^2p^2) = \binom{p}{0} \binom{n^2p^2}{p} \in \mathcal{O}(n^{2p}p^{2p}).$$

Hence, the size of $V(L_g)$ is $\mathcal{O}(n^{2p}p^{2p})$. Since there are $\mathcal{O}(m^p)$ possible vectors

$$g = \{g_1, \ldots, g_p\} \subset \{1, \ldots, m\}^p \quad \text{with} \quad g_j \leq g_{j+1}$$

for $j = 1, \ldots, p-1$, the cardinality of the finite dominating set $V(L)$ is $\mathcal{O}(m^p n^{2p} p^{2p})$.

Every subdivision L_g of $[0,1]^p$ for a fixed g and all its j-faces for $j = 1, \ldots, p$ can be constructed in $\mathcal{O}(n^{2p}p^{2p})$ time using the algorithm of Edelsbrunner et al. (1986). The evaluation of one candidate point can be done in $\mathcal{O}(n(p+\log(n)))$ time (see Algorithm 4). Therefore, all candidates of $V(L_g)$ need $\mathcal{O}(n^{2p+1}p^{2p}(p+\log(n)))$ time to be evaluated. As $\mathcal{O}(m^p)$ subdivisions have to be considered, the complexity can be bounded by $\mathcal{O}(m^p n^{2p+1} p^{2p}(p+\log(n)))$, which is mainly determined by the evaluation of the candidate points. Since p is assumed to be fixed, this simplifies to $\mathcal{O}(m^p n^{2p+1} \log(n))$ time for solving the p-k-max problem utilizing the finite dominating set $V(L)$.

An important observation that reduces the number of candidates to evaluate significantly, is that the facility that defines the optimal objective value always lies in an equilibrium point.

Theorem 6.30 ▶ Objective value defining equilibrium point

Let $X^* = \{x_1^*, \ldots, x_p^*\}$ be an optimal solution of the p-k-max problem with $n \geq 2$ and $k \leq n-p$. W.l.o.g., x_1^* is a new facility that determines the optimal objective function value z^*. Then it holds that $x_1^* \in EQ$.

Proof. Let $X^* = \{x_1^*, \ldots, x_p^*\} \in A(G) \times \ldots \times A(G)$ be optimal locations with optimal objective value z^* (an optimal solution exists according to Theorem 4.6). Let

$$V_\ell := \left\{ v_i \in V \,:\, d^w(v_i, x_\ell^*) \leq d^w(v_i, x_g^*) \;\forall\, g > \ell \;\wedge\; d^w(v_i, x_\ell^*) < d^w(v_i, x_g^*) \;\forall\, g < \ell \right\}$$

for all $\ell \in \{1, \ldots, p\}$ define an optimal assignment of the existing facilities to the new facilities $x_1^*, \ldots, x_p^*$. Note that if an existing facility is of equal distance from two or more new facilities, it is assigned w.l.o.g. to the new facility with the smallest index. Moreover, let $\sigma \in \Sigma(X^*)$, i.e.,

$$\min_{\ell \in \{1,\ldots,p\}} d^w(v_{\sigma(1)}, x_\ell^*) \geq \min_{\ell \in \{1,\ldots,p\}} d^w(v_{\sigma(2)}, x_\ell^*) \geq \cdots \geq \min_{\ell \in \{1,\ldots,p\}} d^w(v_{\sigma(n)}, x_\ell^*).$$

Then

$$\min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(k)}, x^*_\ell) = z^* \quad \text{and} \quad \min_{\ell\in\{1,\dots,p\}} d^w(v_{\sigma(i)}, x^*_\ell) \geq z^* \quad \forall\, i < k.$$

Further, define

$$k_\ell := |\{v_{\sigma(i)} \in V_\ell \,:\, i < k\}| \quad \text{for all} \quad \ell \in \{1,\dots,p\}.$$

Clearly, $\sum_{\ell=1}^{p} k_\ell + 1 = k$. Suppose w.l.o.g. that the optimal objective function value is attained in x^*_1, i.e., $z^* = d^w(v_{\sigma(k)}, x^*_1)$ and assume that $x^*_1, \dots, x^*_p \notin EQ$.

Case 1: z^* is attained only in V_1, i.e., $w_{\sigma(i)} d(v_{\sigma(i)}, x^*_\ell) < z^*$ for all $i > k$ and $\ell = 2, \dots, p$. Now, an alternative optimal solution $\bar{X} = \{\bar{x}_1, x^*_2, \dots, x^*_p\} \in A(G) \times \dots \times A(G)$ is constructed such that $\bar{x}_1 \in EQ$ as follows:

Let $\bar{x}_1 \in A(G)$ be selected such that $\bar{x}_1$ is an optimal location for the 1-(k_1+1)-max problem with optimal objective function value $\bar{z}_1$ and existing facilities V_1 in G (all nodes in $V \setminus V_1$ have weight 0 in this problem, see Remark 4.5). Note that the problem is solvable because of $k_1 \leq |V_1|$, otherwise the allocation V_ℓ, $\ell = \{1,\dots,p\}$, would not be optimal. The 1-(k_1+1)-max problem has its optimal solution in EQ for all $1 \leq k_1 \leq |V_1| - 2$ and in V only if $k_1 = |V_1| - 1$ (see Theorem 5.10).

In the latter case, all nodes in V_1 except of one have to be outliers. Thus, $\bar{x}_1$ is located in this non-outlier node $\bar{v} \in V_1$ and yields $\bar{z}_1 = 0$. With the same argument x^*_1 also had to be located in a node of V_1 and hence $z^* = 0$. It follows that

$$\min\left\{d^w(v_{\sigma(i)}, \bar{x}_1), d^w(v_{\sigma(i)}, x^*_2), \dots, d^w(v_{\sigma(i)}, x^*_p)\right\} = 0 \quad \text{for all} \quad i \geq k.$$

As $\bar{x}_1$ is already located in a node, $x^*_2, \dots, x^*_p$ have to bring all other $n-k \geq p$ distances to 0 which is not possible due to the positive edge and node weights. As a consequence, the case $k_1 = |V_1| - 1$ can not occur and $\bar{x}_1 \in EQ$. Since x^*_1 is also feasible for this problem and has objective value z^*, it is known that the optimal objective value $\bar{z}_1$ of this problem satisfies $\bar{z}_1 \leq z^*$. As $x^*_1 \notin EQ$, a non-zero improvement can be realised by shifting from x^*_1 to $\bar{x}_1$ because, following Theorem 6.23, the optimal solution of the subproblem on V_1 has to be located in EQ.

With the assumption $w_{\sigma(i)} d(v_{\sigma(i)}, x^*_\ell) < z^*$ for all $i > k$ and $\ell = 2, \dots, p$, all existing facilities in $V = V_1 \cup \dots \cup V_p$ can be allocated to $\bar{X}$ such that

$$w_{\theta(k)} d(v_{\theta(k)}, \bar{X}) < z^*,$$

where θ is a permutation in $\Sigma(\bar{X})$, because, if $\sigma \neq \theta$, the new allocation realises even smaller distances than before. Consequently, it can be concluded that $\bar{X}$ is also a feasible solution satisfying

$$k\text{-max}(w_i d(v_i, \bar{X})) < z^*.$$

This contradicts the assumption that X^* is an optimal solution because by shifting x_1^* into EQ, a strictly better objective value can be obtained.

Case 2: z^* is attained in V_1 and in at least one other cluster V_ℓ, $\ell \in \{2, \ldots, n\}$, i.e., there exists a node $v_{\sigma(i)} \in V_\ell$, $i > k$, with

$$w_{\sigma(i)} d(v_{\sigma(i)}, x_\ell^*) = z^*.$$

Again, an alternative solution $\bar{X} = \{\bar{x}_1, \ldots, \bar{x}_p\}$, where $\bar{x}_q = x_q^*$ for all $q \in \{1, \ldots, p\}$ for which the optimal objective function value is not attained in V_q, is constructed as follows: Select $\bar{x}_1 \in EQ$ like in Case 1. It follows that $\bar{z}_1 < z^*$.

Similarly, for all clusters V_ℓ in which the optimal objective function value is attained, let $\bar{x}_\ell \in EQ \cup V$ be selected such that $\bar{x}_\ell$ is an optimal location for the single facility 1-(k_l+1)-max problem with existing facilities V_ℓ in G (all nodes in $V \setminus V_\ell$ have weight 0 in this problem). Since x_ℓ^* is feasible for this problem and has an objective value of at most z^*, it is known that the optimal objective value $\bar{z}_\ell$ of this problem satisfies $\bar{z}_\ell < z^*$, because $x_\ell^* \notin EQ \cup V$ but $\bar{x}_\ell \in EQ$ or $\bar{x}_\ell \in V$ (see Theorem 5.10).

Now, all $v_i \in V$ can be allocated to $\bar{X}$ such that $w_{\theta(k)} d(v_{\theta(k)}, \bar{X}) < z^*$, where θ is defined as in Case 1, because a reallocation just occurs if smaller distances than z^* can be realised. Finally, it can be concluded that $\bar{X}$ is also a feasible solution satisfying

$$k\text{-max}(w_i d(v_i, \bar{X})) < z^*.$$

Thus, X^* can not be an optimal solution and the result is shown. ∎

The finite dominating set $V(L)$ can be reduced further by using the information obtained from the optimal solutions computed by the recursive approach described in Algorithm 10. For this purpose, let $\mathcal{X}_r$ be the set of optimal solutions found by Algorithm 10. Then, with the set

$$Y = \left\{ x_1^* \in EQ_{ij}, i, j \in \{1, \ldots, n\} : \exists\, X^* = \{x_1^*, \ldots, x_p^*\} \in \ \mathcal{X}_r \text{ with } \ d^w(v_i, x_1^*) = z^* \right\}$$

the statement of Corollary 6.24 holds analogously for the p-facility case, i.e., all equilibrium points defining the optimal objective function value z^* in an optimal solution of the p-k-max problem are known from the output of the recursive approach. Thus, all alternative optimal solutions are described by the sets

$$\mathcal{X}(x_1^*) = \left\{ X = \{x_1^*, x_2, \ldots, x_p\} : x_2, \ldots, x_p \in A(G) \ \wedge \ d^w(v_{\sigma(k)}, X) = z^* \right\} \quad \text{for} \quad x_1^* \in Y$$

of optimal solutions that have the same objective function value defining facility x_1^*, $x_1^* \in Y$. For a fixed $x_1^* = (e_{g^*}, t_1^*) \in Y$, all these alternative optimal solutions have to lie on the equilibrium plane described by $B_{cd11}^{\alpha,\beta}$ for $x_1^* \in EQ_{cd}^{a_{g^*} b_{g^*}}$ with fixed $c, d \in \{1, \ldots, n\}$.

Analogous to the 2-facility case, let $eq_g(x_1^*)$ be the equilibrium plane which is described by x_1^* and let $L_g(x_1^*)$ for $g = \{g^*, g_2, \ldots, g_p\}$ with $g_2 \leq \ldots, \leq g_p$ be the subdivision of $[0, 1]^p$

arising from L_g by deleting all bottleneck planes of type $B^{\alpha\beta}_{ii11}$ and all equilibrium planes of type $B^{\alpha\beta}_{ij11}$ except for $eq_g(x_1^*)$ for all $i,j = 1,\dots,n$ and $\alpha,\beta \in \{+,-\}$. $C(L_g(x_1^*))$ denotes the set of cells of the subdivision. Moreover, $V^{eq}(L_g(x_1^*))$ is the set of all 0-faces of the subdivision $L_g(x_1^*)$ that lie on the hyperplane $eq_g(x_1^*)$, i.e., the set of all points on $eq_g(x_1^*)$ that are intersected by at least $p-1$ of the other hyperplanes in $L_g(x_1^*)$. For an illustration of $L_g(x_1^*)$ see Figure 6.16 (left), where the special case of $p = 2$ is shown.

Corollary 6.31 ▶ Improved FDS based on $L_g(x_1^*)$

Let $X^* = \{x_1^*,\dots,x_p^*\} \in \mathcal{X}_r$ with $x_1^* = (e_{g^*}, t_1^*) \in Y$, $g^* \in \{1,\dots,m\}$, fixed. If the p-k-max problem with $n \geq 2$ and $k \leq n-p$ has more than one optimal solution $\{x_1,\dots,x_p\} \in \mathcal{X}(x_1^*)$ with $x_1 = x_1^*$, at least one of these alternative optimal solutions can be found in the set

$$V^{eq}(L(x_1^*)) = \bigcup_{g \subset \{1,\dots,m\}^p} V^{eq}(L_g(x_1^*)),$$

where $g = \{g^*, g_2 \dots, g_p\}$ with $g_j \leq g_{j+1}$ for $j = 2,\dots,p-1$.

Moreover, at least one alternative optimal solution $\{\bar{x}_1, x_2,\dots,x_p\} \in \mathcal{X}(\bar{x}_1)$ (if existent) for every $\bar{x}_1 \in Y$ with $\{\bar{x}_1,\dots,\bar{x}_p\} \in \mathcal{X}_r$ is an element of the set

$$V^{eq}(L(Y)) = \bigcup_{\bar{x}_1 \in Y} V^{eq}(L(\bar{x}_1)).$$

For further information see the special case $p = 2$ on page 146. However, there may still be optimal solutions that can not be found in this finite dominating set, but they can be computed when the optimal solutions in $V^{eq}(L(x_1^*))$ are known. For this purpose, let $\mathrm{conv}\,(A)$ be the convex hull of the elements in A.

Theorem 6.32 ▶ All optimal solutions

Let $X^* = \{x_1^*,\dots,x_p^*\} \in \mathcal{X}_r$ with optimal objective function value z^* and $x_1^* = (e_{g^*}, t_1^*) \in Y$ with $g^* \in \{1,\dots,m\}$ be fixed. Moreover, for a fixed $g = \{g^*, g_2,\dots,g_p\} \subseteq \{1,\dots,m\}^p$ and a cell $\bar{C} \in C(L_g(x_1^*))$, let

$$F_0^{\bar{C}} = \left\{\bar{X} \in V^{eq}(L_g(x_1^*)) \;:\; \bar{X} \in \bar{C} \;\wedge\; d^w(V,\bar{X}) = z^*\right\}$$

be the set of all vertices of the cell $\bar{C}$ that are optimal for the p-k-max problem. Then, all p-tuples

$$X = \{x_1^*, x_2,\dots,x_p\} \quad \text{with} \quad X \in \mathrm{conv}\left(F_0^{\bar{C}}\right),$$

are optimal solutions of the p-k-max problem with $n \geq 2$ and $k \leq n-p$.

Proof. Consider the subdivision $L_g(x_1^*)$, $g = \{g^*, g_2,\dots,g_p\} \subset \{1,\dots,m\}^p$, of $[0,1]^p$ with respect to x_1^* and a cell $\bar{C} \in C(L_g(x_1^*))$. Let f_b be a b-face with dimension $b \leq p-1$ of the subdivision and

$$F^{eq} = \{f_b : f_b \subset eq_g(x_1^*) \;\wedge\; 0 \leq b \leq p-1\}$$

be the set of all faces of dimension smaller or equal to $p-1$ that are a subset of the hyperplane $eq_g(x_1^*)$. As $x_1^* \in Y$ is fixed, all alternative optimal solutions in $\mathcal{X}(x_1^*)$ have to lie on the equilibrium plane $eq_g(x_1^*)$ determined by x_1^*. Thus, consider the k-max function restricted to the variables $x_2, \ldots, x_p \in e_{g_2} \times \ldots \times e_{g_p}$, i.e., the function

$$k\text{-max}(d^w(V, \{x_1^*, x_2, \ldots, x_p\}))$$

over $eq_g(x_1^*)$ in the unit hypercube $[0,1]^p$. The hyperplane $eq_g(x_1^*)$ in $[0,1]^p$ of $L_g(x_1^*)$ defined by x_1^* can be described as the union of boundary faces of cells adjacent to $eq_g(x_1^*)$, i.e.,

$$eq_g(x_1^*) = \bigcup_{f \in F^{eq}} f.$$

Note that $eq_g(x_1^*)$ is in L_g and also in $L_g(x_1^*)$. As all equilibrium and bottleneck planes contributing to L_g but not to $L_g(x_1^*)$ are parallel to $eq_g(x_1^*)$, they do not lead to additional intersection points with $eq_g(x_1^*)$ in L_g. Thus, the b-faces for all $b \leq p-1$ of L_g and $L_g(x_1^*)$ on $eq_g(x_1^*)$ coincide and there exists a face $\bar{f}_b \in F^{eq}$ with $b \leq p-1$ such that

$$\bar{X}_j \in \bar{f}_b \quad \text{for all} \quad \bar{X}_j \in F_0^{\bar{C}}.$$

Since the k-max function is linear in every cell $C \in C(L_g)$ of the subdivision L_g (see Theorem 6.16), the k-max function is also linear in every face $f_b \in F^{eq}$ with $1 \leq b \leq p-1$ because every face f_b is a boundary face of a cell C of L_g.

As all $\bar{X}_j \in \bar{f}_b$ have the same optimal objective function value, the k-max function is constant over $\text{conv}\left(F_0^{\bar{C}}\right) = \bar{f}_b$. Consequently, all $X = \{x_1^*, x_2, \ldots, x_p\}$ with $X \in \text{conv}\left(F_0^{\bar{C}}\right)$ are optimal solutions of the p-k-max problem. Additionally, note that the k-max function is piecewise linear over $eq_g(x_1^*)$. An illustration is given in Figure 6.16 (right), where for $p=2$ the k-max function restricted to the edge $e_{g^*} = e_h$ over the subdivision of $[0,1]^2$ given in Figure 6.16 (left) is shown. ■

As a consequence of Theorem 6.32, b-faces up to a dimension of $b = p-1$ can be optimal for the p-k-max problem. A whole cell of the subdivision can not be optimal because otherwise the objective function value defining facility would not lie on the hyperplane $eq_g(x_1^*)$, i.e., it would not be an equilibrium point.

The procedure to determine all optimal solutions of the p-k-max problem is mostly analogous to the 2-facility case: At first, the subdivisions $L_g(x_1^*)$ of $[0,1]^p$ for all $g \subseteq \{1, \ldots, m\}$ and a fixed x_1^* have to be constructed. Afterwards, the 0-faces, i.e., the intersection points of $eq_g(x_1^*)$ with the other hyperplanes, have to be determined. Then, the k-max function is evaluated in these candidate points and the intersection points leading to the smallest objective function value z^* are optimal. The last step of finding the optimal convex hulls of optimal intersections belonging to the same cell is more complicated in the general case as it is not possible to identify adjacent 0-faces by just sorting coordinates as in the 2-facility case. To overcome this problem, the Algorithm of Edelsbrunner (1987) (for more details see also Edelsbrunner et al., 1986) and the data structure used therein are needed and presented

shortly in the following with the notation adjusted to this thesis.

Edelsbrunner (1987) describes an algorithm for constructing an arrangement of hyperplanes, i.e., an algorithm that builds up a data structure which stores all faces of the arrangement and also all incidences between pairs of faces. Let H be a set of a hyperplanes in $\mathbb{R}^p$ and let $A(H)$ be the arrangement resulting from H. A data structure called *incidence graph* $I(H) = (V(I), E(I))$ is used to represent the arrangement $A(H)$ in the following way:

Each face f of $A(H)$ is represented by a vertex $v(f) \in V(I)$ and if two faces f and f' are incident, the vertices $v(f)$ and $v(f')$ are connected by an edge. Besides the regular s-faces of dimension $0 \leq s \leq p$, two more faces are defined: The (-1)-face (that corresponds to the empty set) and the $(p+1)$-face (representing the whole arrangement $A(H)$). The (-1)-face is defined to be incident with all vertices of $I(H)$ representing a 0-face and the $(p+1)$-face is defined to be incident with all vertices representing a p-face. Figure 6.19 shows an example of an incidence graph (right) of the arrangement given on the left, where the second level of the graph represents the vertices of the subdivision, the third level represents the edges and the fourth level the cells. Note that an incidence graph of a set of a hyperplanes in $\mathbb{R}^p$ contains $\mathcal{O}(a^p)$ vertices and edges. The implementation of an incidence graph contains two lists stored in each vertex $v(f)$ of an s-face f: One list containing pointers to the vertices representing an $(s+1)$-face incident to f and the other list containing the pointers to all vertices of $(s\text{-}1)$-faces incident to f.

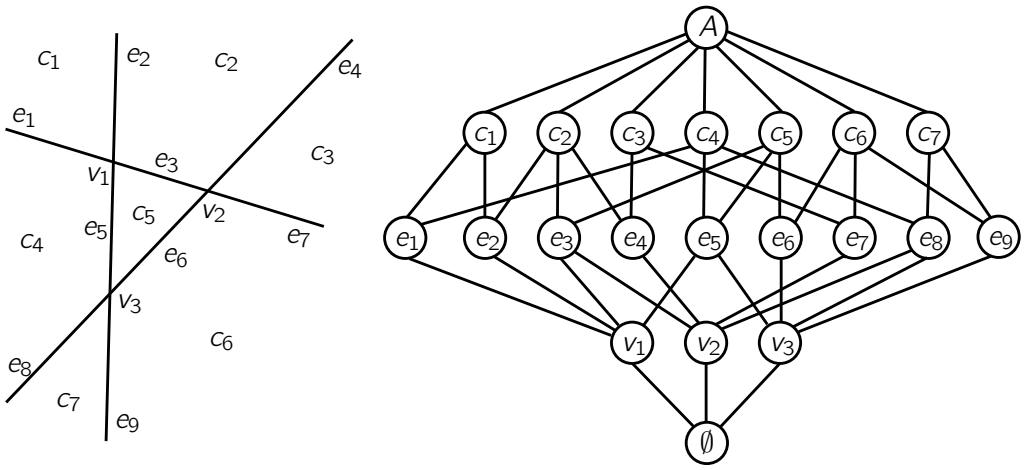


Fig. 6.19: Arrangement $A(H)$ in $\mathbb{R}^2$ and its incidence graph $I(H)$

To generate the incidence graph $I(H)$, Edelsbrunner introduces an incremental algorithm. It starts with an arrangement $A(H_p)$ of $p < a$ hyperplanes whose normal vectors are linearly independent. The other $a - p$ hyperplanes are added iteratively to the arrangement. The incidence graph is updated after each insertion of a new hyperplane. The initial set of hyperplanes H_p can be found easily by analysing the determinants of the normal vectors of subsets of hyperplanes to check if the normal vectors are linearly independent.

Now, let $A(H_i)$ be the arrangement of the $p < i < a$ hyperplanes added in the first i iterations of the algorithm and let h_{i+1} be the hyperplane that is added in the next iteration. All s-faces with $0 \leq s \leq p$ of $A(H_i)$ that are not intersected by h_{i+1} remain an s-face in

$A(H_{i+1})$ and do not lead to a change in $I(H)$. All other s-faces $A(H_i)$ that are intersected by h_{i+1} are marked in different colors depending on the position of the face relative to h_{i+1}. If an s-face is marked, it has to be split into two s-faces bounded by the two halfspaces defined by h_{i+1} and an (s-1)-face representing the intersection of the s-face and h_{i+1}. Thus, the incidence graph has to be manipulated by adding the corresponding nodes and updating the related edges. The operations needed to update $I(H)$ depend on the color of the current s-face. The complete algorithm to build up $I(H)$ has a complexity of $\mathcal{O}(a^p)$, which is best possible for an arrangement of a hyperplanes in $\mathbb{R}^p$.

The incidence graph is very useful to analyse which optimal 0-faces of $L_g(x_1^*)$ belong to the same cell such that their convex hull is optimal for the p-k-max problem. The subdivision $L_g(x_1^*)$ of $[0,1]^p$ for a fixed set of edges $g \subseteq E$ is an arrangement of at most $\mathcal{O}(n^2p^2)$ hyperplanes. As x_1^* is fixed, and with it also the equilibrium plane $eq_g(x_1^*)$, it is sufficient to analyse the subdivision

$$L_g^{eq}(x_1^*) = L_g(x_1^*) \cap eq_g(x_1^*)$$

in $\mathbb{R}^{p-1}$ which corresponds to the subdivision $L_g(x_1^*)$ restricted to $eq_g(x_1^*)$. The notation for the parts (vertices, cells etc.) of $L_g^{eq}(x_1^*)$ are used equivalently to the subdivisions before. Let $I(L^{eq}) = (V(I), E(I)))$ be the incidence graph of $L_g^{eq}(x_1^*)$. The idea is to go bottom-up through the incidence graph and to identify for each dimension $s \in \{0, \dots, p-1\}$ which s-faces are optimal. It can be used that the optimal 0-faces can be determined easily and that an s-face for $s \geq 2$ is only optimal if all of its incident (s-1)-faces are optimal. Thus, denote by

$$V_s = \{v_i^s \in V(I) : f_i^s \text{ is } i\text{th optimal } s\text{-face}, i \in \{1, \dots, |V_s|\}\},$$

$s \in \{0, \dots, p-1\}$, the set of vertices v_i^s of the incidence graph that represent an s-face f_i^s for which it holds that all $X \in f_i^s$ are optimal for the p-k-max problem. Note that V_0 equals the set of optimal extreme points in $V^{eq}(L_g(x_1^*))$ which is assumed to be known from a previous step of the solution approach. In the following, no distinction will be made between the faces f_i^s of $L_g^{eq}(x_1^*)$ and the nodes $v_i^s \in V(I)$ that represent them.

It is now assumed that the set V_{s-1} of optimal (s-1)-faces is known. Thus, in the next step the set V_s of optimal s-faces has to be computed based on the information of V_{s-1}. Therefore, let $v_i^{s-1} \in V_{s-1}$ be the ith optimal (s-1)-face, $i \in \{1, \dots, |V_{s-1}|\}$, and v_j^s be an s-face incident to v_i^{s-1} such that $(v_i^{s-1}, v_j^s) \in E(I)$. Then, v_j^s is optimal for the p-k-max problem if and only if it holds that

$$v_h^{s-1} \in V_{s-1} \quad \text{for all} \quad (v_h^{s-1}, v_j^s) \in E \quad \text{with} \quad h \in \{1, \dots, |V_{s-1}|\}, \tag{6.22}$$

i.e., if all of its incident subfaces v_h^{s-1} are optimal. Note that it is enough to analyse just one incident optimal $v_{i'}^s \neq v_i^{s-1}$ because then the other incident subfaces also have to be optimal as the k-max function is constant over f_j^s if it is constant over two of its (s-1)-dimensional boundary faces f_i^{s-1} and $f_{i'}^{s-1}$. Thus, if the condition (6.22) is satisfied, v_j^s is an element of V_s. If (6.22) is not satisfied for all superfaces f_j^s of f_i^{s-1}, then f_i^{s-1} does not contribute to an optimal face of a larger dimension and is stored in the set F_{opt} of optimal faces of $L_g^{eq}(x_1^*)$. As a consequence, only optimal faces of maximum dimension are stored and not

also all their smaller-dimensional subfaces.

As at least $s+1$ optimal (s-1)-faces are needed to construct an optimal s-face, the condition $|V_{s-1}| < s+1$ is used as a stopping criterion. Since it is easier to construct the convex hull of optimal solutions for the p-k-max problem when the extreme points of the convex hull are known, a set EP_{opt} is build up analogously to F_{opt} and it contains all sets of 0-faces whose convex hulls are optimal. The procedure is summarised in Algorithm 12.

Algorithm 12 Determine optimal extreme points belonging to the same cell

Input: Incidence Graph $I(L) = (E(I), V(I))$ of $L_g^{eq}(x_1^*)$ for a fixed $g \subseteq E(G)$, set V_{opt} of optimal 0-faces of $L_g^{eq}(x_1^*)$

1: Set $V_0 := V_{opt}$, $V_s := \emptyset$ for all $s = 1, \ldots . p-1$, $EP(v_a^0) := v_a^0$ for all $a \in \{1, 2, \ldots\}$, $EP_0 := \{\{v_i^0 \in V_0\} : i = 1, \ldots, |V_0|\}$, $A := \emptyset$
2: **while** $|V_{s-1}| \geq s+1$ **do** ▷ As long as there are enough optimal vertices on level s
3: **for all** $v_i^{s-1} \in V_{s-1}$ **do** ▷ Choose an optimal (s-1)-face
4: $\alpha = 0$
5: **for all** v_j^s with $(v_i^{s-1}, v_j^s) \in E(I)$ **do** ▷ Choose an s-faces incident with v_i^{s-1}
6: **if** $\exists\ v_{h'}^{s-1} \in V_{s-1}$ with $v_{h'}^{s-1} \neq v_i^{s-1}$ and $(v_{h'}^{s-1}, v_j^s) \in E(I)$ **then**
7: $\alpha = 1$ ▷ All predecessors of v_j^s are optimal
8: **for all** $v_h^{s-1} \in V_{s-1}$ with $(v_h^{s-1}, v_j^s) \in E(I)$ **do** ▷ (s-1)-faces incident v_j^s
9: $A = A \cup \{v_h^{s-1}\}$ ▷ Optimal predecessors of v_j^s
10: $V_s = V_s \cup \{v_j^s\}$ ▷ Optimal s-faces
11: $EP(v_j^s) = \bigcup_{v_a^{s-1} \in A} EP(v_a^{s-1})$ ▷ 0-faces bounding v_j^s
12: $EP_s = EP_s \cup \{EP(v_j^s)\}$ ▷ Sets of 0-faces of every optimal s-face
13: $A = \emptyset$
14: **if** $\alpha = 0$ **then** ▷ v_i^{s-1} is not predecessor of an optimal s-face
15: $F_{opt} = F_{opt} \cup \{v_i^{s-1}\}$ ▷ Optimal faces
16: $EP_{opt} = EP_{opt} \cup EP(v_i^{s-1})$ ▷ Sets of 0-faces bounding an optimal face
17: $A = \emptyset$
18: $s = s+1$
19: **if** $|V_{s-1}| < s+1$ **then** ▷ While-criterion not fulfilled
20: $F_{opt} = F_{opt} \cup V_{s-1}$
21: $EP_{opt} = EP_{opt} \cup EP_{s-1}$

Output: $EP_{opt} = \{M = \{v_i^0 \in V_{opt} : v_i^0 \subset \bar{C}\ \forall i\} : \bar{C} \in C(L_g^{eq}(x_1^*))\ \wedge\ M \neq \emptyset\}$, where each element M is a set that contains the optimal 0-faces of a cell $\bar{C} \in C(L_g^{eq}(x_1^*))$ having at least one optimal 0-face.

The worst case complexity of Algorithm 12 is $\mathcal{O}(n^{4p})$. The while-loop is processed at most $\mathcal{O}(p)$ times as it may go through all dimensions $s \in \{0, \dots, p-1\}$. The number of all (s-1)-faces is bounded by $\mathcal{O}((n^2p^2)^{p-1})$ (see Edelsbrunner (1987)), which is the number of hyperplanes taken to the power of the dimension of the space. Thus, $|V_{s-1}| = \mathcal{O}((n^2p^2)^{p-1})$, which is also the complexity of the first for-loop. The second for-loop needs $\mathcal{O}((n^2p^2)^{p-1})$ time as all s-faces v_j^s have to be considered, not only the optimal ones. To find all subfaces v_i^{s-1} incident to v_j^s, all predecessors of v_j^s have to be analysed. Since a face can be bounded by at most all n^2p^2 hyperplanes, the complexity needed to check the if-condition is $\mathcal{O}(n^2p^2)$. The same argument holds for the last for-loop which therefore also needs $\mathcal{O}(n^2p^2)$ time in the worst case. Summarising the discussion above, this leads to a complexity of $\mathcal{O}(n^{4p}p^{4p+1})$. As p is considered to be fixed, the above mentioned overall complexity for the computation of the sets of 0-faces follows. In practice this procedure will be more efficient as the number of optimal s-faces will be in general much smaller than the total number of s-faces.

Now that all parts of the solution approach to determine all optimal solutions of the p-k-max problem have been derived, the procedure is summarised in Algorithm 13.

The output of Algorithm 13 is the set EP_{opt} where each element is a set that contains the optimal 0-faces that belong to a cell $\bar{C} \in C(L_g^{eq}(x_1^*))$. Thus, each set $EP \in EP_{opt}$ describes the extreme points of a polytope in $\mathbb{R}^s$ for $s \in \{1, \dots, p-1\}$. All $X \in \mathbb{R}^n$ that lie inside or on the boundary of this polytope are optimal for the p-k-max problem (see Theorem 6.32). If needed, the set

$$\mathcal{X} = \mathcal{X} \cup \{X \in \mathbb{R}^p : X \in \text{conv}(EP) \text{ for } EP \in EP_{opt}\}$$

of all optimal solutions of the p-k-max problem can be obtained easily by computing the convex hull of each set $EP \in EP_{opt}$, for example with the Algorithm of Seidel (1986) or Chazelle (1993).

The worst case complexity of the below algorithm is $\mathcal{O}(m^p n^{4p+2})$. The computational effort of considering all elements of the set Y is $\mathcal{O}(mn^2)$, which is the maximum number of equilibrium points of G. The number of subdivisions that have to be considered is $\mathcal{O}(m^{p-1})$ since $g_1 = g^*$ is fixed. The maximum number of equilibrium-, bottleneck- and balance planes is $\mathcal{O}(n^2p^2)$ which was already shown in the complexity analysis of the 2-facility case. For the computation of H', all these hyperplanes have to be considered and each intersection of $h \in H$ and $eq_g(x_1^*)$ can be determined in $\mathcal{O}(p)$ time. The subdivision $L_g^{eq}(x_1^*)$ can be constructed with the algorithm of Edelsbrunner (1987) in $\mathcal{O}((n^2p^2)^{p-1})$ time as $L_g^{eq}(x_1^*)$ is a subdivision of the unit hypercube $[0,1]^{p-1}$. Following equation (6.21), the number of optimal extreme points in $V(L_g^{eq}(x_1^*))$ is bounded by $\mathcal{O}((n^2p^2)^{p-1})$. As every candidate solution needs $\mathcal{O}(n(p + \log(n)))$ time to be evaluated, the evaluation of all 0-faces in $L_g^{eq}(x_1^*)$ has a complexity of $\mathcal{O}(n^{2p-1}p^{2p-2}(p + \log(n)))$. The set EP_{opt} can be computed using Algorithm 12 which has a worst case complexity of $\mathcal{O}(n^{4p}p^{4p+1})$. Hence, an overall complexity of $\mathcal{O}(m^p n^{4p+2}p^{4p+1})$ for Algorithm 13 follows, which can be simplified to $\mathcal{O}(m^p n^{4p+2})$ since p is assumed to be fixed. Thus, the effort of this algorithm is mainly determined by the computation of the set EP_{opt} to get all optimal solutions of the problem.

Note that the algorithm can be implemented in parallel easily: The whole computation for a fixed objective function value defining facility is independent from the other z^*-defining facilities. Moreover, the construction of the subdivision for a fixed p-tuple of edges does not depend on the results of the subdivisions of another combination of edges. Also the evaluation of the found intersection points can be parallelised. This leads to a significant improvement of the computation time in practice.

Algorithm 13 Local Analysis to find all optimal solutions of the p-k-max problem

Input: Graph $G = (V, E)$ with $w_i > 0 \ \forall \ i = 1, \ldots, n$, $p \in \{2, \ldots, n\}$, $k \in \{1, \ldots, n-p\}$, optimal objective value z^*, set Y of z^*-defining equilibrium points.

Set $V_{opt} := \emptyset$, $H' := \emptyset$
for all $x_1 = (e_{g^*}, t_1^*) \in Y$ **do** ▷ All relevant optimal objective value defining facilities
  for all $(g_2, \ldots, g_p) \subseteq \{1, \ldots, m\}^{p-1}$ with $g_2 \leq \ldots \leq g_p$ **do** ▷ All p-tuples g...
    Set $g := (g^*, g_2, \ldots, g_p)$ ▷ ...with fixed edge g^*
    Derive the set H of equilibrium-, bottleneck- and balance planes for $e_{g^*} \times \ldots \times e_{g_p}$
    for all $a = 1, \ldots, |H|$ **do**
      Compute the intersection h' of $h_a \in H$ with $eq_g(x_1^*)$
      $H' = H' \cup h'$ ▷ Hyperplanes that define the subdivision restricted to $eq_g(x_1^*)$
    Derive the subdivision $L_g^{eq}(x_1^*) = L_g(x_1^*) \cap eq_g(x_1^*)$ of the set of hyperplanes H'
    for all $X_i \in V(L_g^{eq}(x_1^*))$ **do** ▷ Test all 0-faces for optimality
      Compute the objective value $z_i = k\text{-max}(d^w(V, X_i))$ of X_i
      if $z_i = z^*$ **then** ▷ X_i is optimal
        $V_{opt} = V_{opt} \cup \{X_i\}$
    Compute the set EP_{opt}, where each element is a set that contains the 0-faces of a cell $\bar{C} \in C(L_g^{eq}(x_1^*))$ ▷ See Algorithm 12

Output: $EP_{opt} = \{M = \{v_i^0 \in V_{opt} : v_i^0 \subset \bar{C} \ \forall i\} : \bar{C} \in C(L_g^{eq}(x_1^*)) \ \wedge \ M \neq \emptyset\}$, where each element M is a set that contains the optimal 0-faces of a cell $\bar{C} \in C(L_g^{eq}(x_1^*))$ having at least one optimal 0-face.

If not all optimal solutions of the p-k-max problem are needed, the overall complexity of the local analysis approach reduces to $\mathcal{O}(m^p n^{2p+1} \log(n))$ for fixed p, which is determined by the complexity of the evaluation of the 0-faces of $L_g^{eq}(x_1^*)$. Note that the set V_{opt} contains alternative optimal solutions (if existent) that can not be found by the recursive approach. Thus, the application of Algorithm 13 provides useful additional information without increasing the overall worst case complexity determined by the recursive approach.

Remark 6.33. *Note that the worst case complexity of $\mathcal{O}(m^p n^{4p+2})$ is generally a very pessimistic estimation as the number of optimal 0-faces as well as the number of all other*

relevant s-faces for $s \in \{1, \ldots, p-1\}$ is estimated using the maximum possible number of faces. However, in practice, the number of this faces is in general much smaller than it could be suggested by the theoretical upper bound (see Chapter 7).

Remark 6.34. *The local analysis approach can equivalently be applied to solve p-k-*max *problems in the plane with polyhedral gauges, e.g. the ℓ_1-norm or the ℓ_∞-norm. For this purpose, the network distances have to be substituted by the corresponding gauge distances to determine the set of breakpoints in Cases 1 to 4. The resulting equilibrium planes, bottleneck planes and balance planes are hyperplanes in $\mathbb{R}^{2p}$ and thus constitute a subdivision of the $\mathbb{R}^{2p}$. As the cells of this subdivision are convex and the k-*max *function is linear on each of these cells, the* 0*-faces define a finite dominating set of the p-k-*max *problem. Note that there may, in general, be more hyperplanes defining such a subdivision than it is the case for problems on networks since, for example, the absolute value in the ℓ_∞-norm has to be resolved.*

However, the local analysis approach is not promising for general gauges. If, for example, Euclidean distances are used for the measurement of distances, the equations describing the sets of breakpoints are not linear and do therefore not form hyperplanes. In general, the obtained cells of the subdivision are not convex and the intersection points of the subdivision do not have to define a finite dominating set.

6.4 Shifting Optimal Facilities

In this section, another approach to generate (infinitely many) alternative optimal solutions, if existent, to the optimal solutions in Cand4 is presented. In this method, the recursive approach given in Algorithm 10 resp. the evaluation of the whole candidate set described in Algorithm 9 is used as a starting point. The idea is to generate alternative optimal solutions by shifting the individual new facilities of an optimal solution in Cand4 within a certain interval without destroying the optimality of the solution started from.

Let $X = \{x_1, \ldots, x_p\} \in \mathcal{X}_r \subseteq$ Cand4 be an optimal solution of the p-k-max problem found by the recursive approach (see Algorithm 10 in Section 6.2) with optimal objective function value z^*. Moreover, it is assumed that $x_1 \in X$ is a facility that defines the optimal objective function value of X. A set of outliers corresponding to this optimal solution is given by V_{k-1}. Then it holds that every new facility $x_i = (e_j, t_i)$ with $i \in \{2, \ldots, p\}$ and $j \in \{1, \ldots, m\}$ can be moved within a certain interval to the left and to the right along the edges of the graph without loosing the optimality of X. Note that this does not exclude the case that the interval just contains one point such that the corresponding facility can not be moved. Define therefore the cluster of $x_i \in X$ as

$$C_i = \{v \in V \setminus V_{k-1} : d^w(v, x_i) \leq d^w(v, x_\ell) \; \forall \; \ell > i \; \wedge \; d^w(v, x_i) < d^w(v, x_\ell) \; \forall \; \ell < i\}$$

for $i \in \{1, \ldots, p\}$ as the set of all nodes covered with service by the new facility x_i in the optimal solution X. If an existing facility is of equal distance from two or more new facilities, it is assigned w.l.o.g. to the new facility with the smallest index.

The optimal objective function value z^* defines the largest weighted distance of a covered existing facility to its closest new facility (which is assumed to be x_1 here). As x_1 has to be an equilibrium point, z^* is the radius of a largest cluster defined by the optimal solution X (for an example see Figure 6.1 or Figure 6.6). Hence, the other new facilities $x_2, \ldots, x_p$ are also allowed to generate a weighted distance of up to z^* to their allocated nodes that are not outliers. This implies that the new facilities not defining the objective function value do not have to be at the center point of the nodes allocated to it. A new facility x_i with $i \in \{2, \ldots, p\}$ can therefore be moved to the left or to the right along the edges of the graph until it holds $d^w(\bar{v}, x_i) = z^*$ for a node $\bar{v} \in C_i$ allocated to x_i without changing the objective function value of the respective solution. By shifting x_i further than this into this direction, the weighted distance $d^w(\bar{v}, x_i)$ gets larger than z^* and may destroy the optimality of X (unless the reached point x_i is a bottleneck point of $\bar{v} \in C_i$ such that a further movement of x_i into the same direction decreases the weighted distance to its corresponding node again, or a reallocation yields an objective value of at most z^*).

However, in the case that $\bar{v}$ is also covered by x_1 (even though it is closer to x_i in the solution X), the distance $d^w(\bar{v}, x_i)$ is allowed to be larger than z^* as the location of x_1 is fixed. Thus, define the set of nodes that are allocated to x_i and that have a weighted distance larger than z^* to x_1 by

$$B_i = \{v \in C_i : d^w(v, x_1) > z^*\} \quad \text{for} \quad i = 2, \ldots, p$$

and let $\bar{v}$ be an element of B_i. Moreover, let $Q(\bar{v})$ be the set of points with a weighted distance of at most z^* to $\bar{v}$, i.e.,

$$Q(\bar{v}) = \{\bar{x} \in A(G) : d^w(\bar{v}, \bar{x}) \leq z^*\}$$

and define

$$Q(x_i) = \bigcap_{\bar{v} \in B_i} Q(\bar{v}).$$

Then, $x_i \in X$ can be replaced by any point $\bar{x}_i \in Q(x_i)$ and the new solution

$$\bar{X} = \{x_1, \ldots, x_{i-1}, \bar{x}_i, x_{i+1}, \ldots, x_p\} \quad \text{with} \quad x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_p \in X$$

is also optimal, since $\bar{x}_i$ covers all nodes within the optimal radius that were covered by x_i before and that were not also covered by x_1. Thus, the coverage radius is still z^* and hence optimal. Note that $Q(x_i) = \{x_i\}$ is possible. In this case, x_i can not be moved without destroying the optimality.

Example 6.35. *A* 2-2-max *problem with node weights* $w_i = 1$ *for all* $i = 1, \ldots, 5$ *is solved on the graph shown in Figure 6.20 using the recursive approach of Algorithm 10 to evaluate the finite dominating set* $\text{Cand4} = EQ \times (EQ \cup V)$*. The resulting optimal solution is*

$$X = \{x_1, x_2\} = \left\{ \left(e_{24}, \frac{1}{8} \right), (e_{23}, 1) \right\}$$

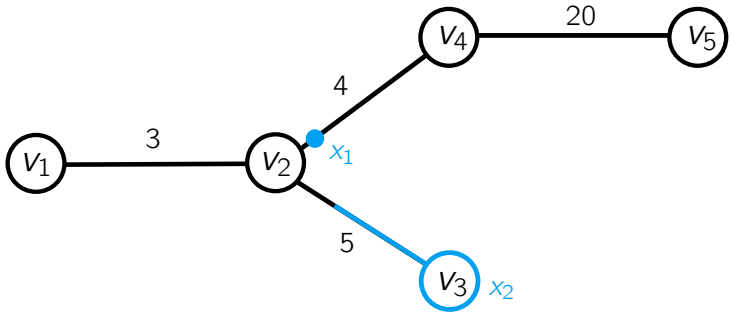


Fig. 6.20: Infinitely many alternative optimal solutions of the 2-2-max problem on G can be found by shifting $x_2 \in X \subseteq \mathcal{X}_r$ along the edges of G, starting from $x_2 = (e_{23}, 1)$.

with optimal objective function value $z^* = \frac{7}{2}$ *determined by the first new facility. The corresponding set of outliers is* $V_{k-1} = \{v_5\}$*. Consequently, the first new facility is fixed and only the second new facility can be moved. It holds that*

$$C_1 = \{v_1, v_2, v_4\} \quad \textit{and} \quad C_2 = B_2 = \{v_3\}.$$

Hence,

$$Q(v_3) = \left[\left(e_{23}, \frac{3}{10}\right), (e_{23}, 1)\right] = Q(x_2).$$

The facility $x_2 \in X$ *can therefore be shifted within the edge segment* $Q(x_2)$ *without loosing the optimality by replacing this facility in the solution* X*. The corresponding edge segment is marked in blue in Figure 6.20.*

The above described procedure can be applied to all new facilities x_i for $i = 2, \ldots, p$ from an optimal solution. There are in general infinitely many optimal solutions that can be constructed from a solution $X \in \mathcal{X}_r$.

Instead of using the set $\mathcal{X}_r$ for the above approach, also the set $\mathcal{X}_{\text{Cand4}}$ of optimal solutions resulting from Algorithm 9 can be utilised as a starting point of the shifting procedure. However, this is rather of theoretical interest. In practice, the approach in this form is not efficient as the computation of the needed optimal solutions in the finite dominating set Cand4 may take too much time unless the problem is rather small in the number of customers and new facilities (see Section 7). As the recursive approach is much faster, it can be used for practical problems and returns $\mathcal{X}_r$ in reasonable time.

Since $\mathcal{X}_r \subseteq \mathcal{X}_{\text{Cand4}}$, starting the shifting of the new facilities based on an optimal solution of the set $\mathcal{X}_{\text{Cand4}}$ yields in general further alternative optimal solutions of the p-k-max problem that can not be found when starting from an $X \in \mathcal{X}_r$. This can be illustrated by considering Example 6.36.

Example 6.36 (Continuation of Example 6.35)**.** *The* 2-2-max *problem of Example 6.35 is considered. The resulting optimal solutions found by the evaluation of all candidates in*

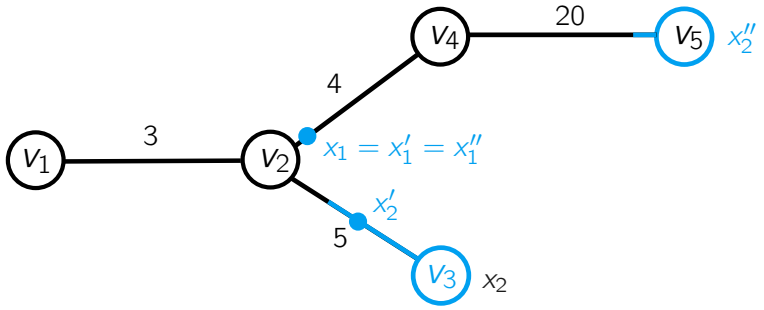


Fig. 6.21: All optimal solutions of the 2-2-max problem on G determined by shifting the second facility of all optimal solutions out of the set Cand4

Cand4 *are*

$$X = \{x_1, x_2\} = \left\{ \left(e_{24}, \frac{1}{8}\right), (e_{23}, 1) \right\}, \quad X' = \{x_1', x_2'\} = \left\{ \left(e_{24}, \frac{1}{8}\right), \left(e_{23}, \frac{1}{2}\right) \right\},$$
$$X'' = \{x_1'', x_2''\} = \left\{ \left(e_{24}, \frac{1}{8}\right), (e_{45}, 1) \right\}$$

with optimal objective function value $z^* = \frac{7}{2}$*, again determined by the same first new facility in all three solutions. The corresponding sets of outliers are* $V_{k-1} = \{v_5\} = V_{k-1}'$ *and* $V_{k-1}'' = \{v_3\}$*. As* $Q(x_2') = Q(x_2)$ *but*

$$Q(x_2) \neq Q(x_2'') = \left[\left(e_{45}, \frac{33}{40}\right), (e_{45}, 1) \right],$$

there are further optimal solutions on the edge e_{45} *that were not found using the unique solution of the recursive approach. The corresponding edge segments are marked in blue in Figure 6.21. The solutions found are all optimal solutions of the underlying problem.*

A possibility to compute the set $Q(x_i)$, $i \in \{2, \ldots, p\}$, efficiently is given in the following. For this purpose, consider an arbitrary but fixed edge $e_{ab} \in E$. The subset $Q^{ab}(x_i) \subseteq Q(x_i)$ of points on e_{ab} belonging to $Q(x_i)$ can be computed as follows: Let $v_j \in B_i$ be fixed. Moreover, let $(e_{ab}, \alpha) \in e_{ab}$ be a point on e_{ab} with $\alpha \in [0, 1]$ such that

$$w_j(d(v_j, v_a) + d(v_a, (e_{ab}, \alpha))) = z^*,$$

and analogously $(e_{ab}, \beta) \in e_{ab}$ with $\beta \in [0, 1]$ such that

$$w_j(d(v_j, v_b) + d(v_b, (e_{ab}, \beta))) = z^*.$$

Note that such a value of α, β may not exist (this case will be discussed later). Then, for $Q^{ab}(v_j) = \{\bar{x} \in e_{ab} : d^w(v_j, \bar{x}) \leq z^*\}$ it holds that

$$Q^{ab}(v_j) = [(e_{ab}, 0), (e_{ab}, \alpha)] \cup [(e_{ab}, \beta), (e_{ab}, 1)],$$

which is the subset of $Q(v_j)$ that lies on edge e_{ab}. If $d(v_j, v_a) > z^*$, there is no such point (e_{ab}, α) with $\alpha \in [0, 1]$ and hence the corresponding interval $[(e_{ab}, 0), (e_{ab}, \alpha)]$ is empty. If, on the other hand, $w_j(d(v_j, v_a) + l_{ab}) < z^*$, it holds that $Q^{ab}(v_j) = e_{ab}$. These observations hold analogously for $d(v_j, v_b) > z^*$ and $w_j(d(v_j, v_b) + l_{ab}) < z^*$. Applying this analysis for all nodes $v_j \in B_i$ and updating the bounds α resp. β in each iteration to the leftmost resp. rightmost known bound such that still all nodes $v_j \in B_i$ are covered by x_i yields the set $Q^{ab}(x_i)$. If $d(v_j, v_a) > z^*$ and $d(v_j, v_b) > z^*$ for at least one $v_j \in B_i$, then $Q^{ab}(x_i)$ is empty.

Algorithm 14 Shifting optimal facilities to find alternative optimal solutions

Input: Graph $G = (V, E)$ with $w_i > 0 \ \forall \ i = 1, \ldots, n$, $p \in \{2, \ldots, n\}$, $k \in \{1, \ldots, n-p\}$, set of optimal solutions $\bar{\mathcal{X}} \subseteq \{\mathcal{X}_{\text{Cand4}}, \mathcal{X}_r\}$, optimal objective function value z^*

Set $\mathcal{X}_s := \emptyset$
for all $X \in \bar{\mathcal{X}}$ **do**
  Determine the cluster C_1 of the z^*-defining facility x_1
  for all $x_i \in X$, $i = 2, \ldots, p$ **do**
    Determine the cluster C_i and the set B_i
    for all $e_{ab} \in E$ **do**
      Set $\bar{\alpha} := 1$ and $\bar{\beta} := 0$ ▷ Construct $Q^{ab}(x_i)$ with bounds $\bar{\alpha}$ and $\bar{\beta}$
      for all $v_j \in B_i$, $j = 1, \ldots, |B_i|$ **do**
        while $\bar{\alpha} \neq -1$ and $\bar{\beta} \neq -1$ **do**
          if $w_j(d(v_j, v_a) + d(v_a, (e_{ab}, \alpha))) = z^*$ for a value of $\alpha \in [0, 1]$ **then**
            if $\alpha < \bar{\alpha}$ **then** ▷ New bound $\bar{\alpha}$ of $Q(v_j)$ on e_{ab}
              Set $\bar{\alpha} := \alpha$
          else if $d(v_j, v_a) > z^*$ **then**
            Set $\bar{\alpha} := -1$ ▷ No bound $\bar{\alpha}$ of $Q(v_j)$ on e_{ab}
          if $w_j(d(v_j, v_b) + d(v_b, (e_{ab}, \beta))) = z^*$ for a value of $\beta \in [0, 1]$ **then**
            if $\beta > \bar{\beta}$ **then** ▷ New bound $\bar{\beta}$ of $Q(v_j)$ on e_{ab}
              Set $\bar{\beta} := \beta$
          else if $d(v_j, v_b) > z^*$ **then**
            Set $\bar{\beta} := -1$ ▷ No bound $\bar{\beta}$ of $Q(v_j)$ on e_{ab}
      Set $Q^{ab}(x_i) := [(e_{ab}, 0), (e_{ab}, \bar{\alpha})] \cup \left[(e_{ab}, \bar{\beta}), (e_{ab}, 1)\right]$ where
      $[(e_{ab}, 0), (e_{ab}, -1)] := \emptyset$ and $[(e_{ab}, -1), (e_{ab}, 1)] := \emptyset$
    Set $Q(x_i) := \bigcup_{e_{ab} \in E} Q^{ab}(x_i)$
    Set $\mathcal{X}_s = \mathcal{X}_s \cup \left\{\{x_1, \bar{x}_2, \ldots, \bar{x}_p\} : \bar{x}_j \in Q(x_j), j = 1, \ldots, p\right\}$

Output: Set $\mathcal{X}_s$ of alternative optimal solutions of the p-k-max problem.

The process is summarised in Algorithm 14. Note that it is assumed that the optimal objective function value defining facility is indexed as the first new facility in all optimal solutions. The algorithm is kept general such that both sets $\mathcal{X}_{\text{Cand4}}$ and $\mathcal{X}_r$ of optimal solutions can be used as a starting point.

Algorithm 14 has a worst case complexity of $\mathcal{O}(m^{p+1}n^{2p+1})$. The first for-loop needs at most $\mathcal{O}(m^p n^{2p})$ iterations as this this is the maximum number of elements of the set $\mathcal{X}_{\text{Cand4}}$. The second for-loop needs $\mathcal{O}(p)$ iterations since all but one of the new facilities of the solution X have to be considered. Determining C_i and B_i, $i = 2, \ldots, p$, can be done in $\mathcal{O}(n)$ time. The loop over all edges needs $\mathcal{O}(m)$ iterations. The innermost for-loop considers all nodes covered by x_i and needs therefore at most $\mathcal{O}(n)$ time. The rest of the statements has a constant complexity. Overall, this leads to the mentioned complexity as p is assumed to be fixed. Note that the algorithm can be parallelised easily since the shifting which starts from a fixed optimal solution does not depend on the shifting of the other optimal solutions in $\bar{\mathcal{X}}$.This leads to a clear acceleration of the computation in practice.

Note that it is not guaranteed to find all optimal solutions of the p-k-max problem using the described shifting-approach, even if $\bar{X} = \mathcal{X}_{\text{Cand4}}$ is chosen in Algorithm 14. If, for example, the optimal objective function value z^* of the problem can also be realised with $\bar{p} < p$ new facilities, the remaining $p - \bar{p}$ facilities can be located arbitrarily on the edges or in the nodes of G. As there may be long edges with two consecutive points $x, x' \in EQ \cup V$ with $d(x, x') \geq 2z^*$, not all points in $A(G)$ can be reached by the shifting.

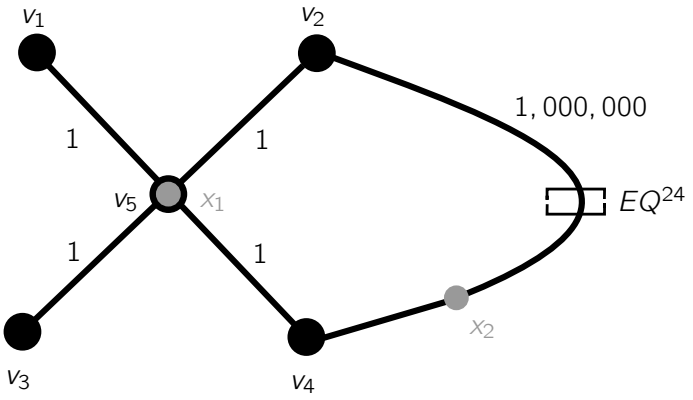


Fig. 6.22: Optimal solution $X = \{x_1, x_2\}$ which can not be found by the shifting approach.

Example 6.37. *In Figure 6.22 a graph $G = (V, E)$ with $w_i = 1$ for all $i = 1, \ldots, 5$ is given. The edge lengths are all equal to 1, except for the edge e_{24} with $l_{24} = 1,000,000$. An optimal solution of the* 2-1-max *problem is $X = \{x_1, x_2\}$ with $z^* = 1$ as shown in gray. As x_1 covers all nodes within the optimal coverage radius, x_2 can be located arbitrarily in $A(G)$ since it can neither improve nor worsen the overall objective function value. Thus, the optimal value z^* could also be reached with just one new facility.*

Note that all equilibrium points EQ^{24} on the edge e_{24} lie very close together in the highlighted interval. As $d^w(v_4, x_2) >> z^*$ and also $d(x', x_2) \gg z^*$ for any $x' \in EQ^{24}$, the

optimal solution X can not be generated from a solution $\bar{X} \in \mathcal{X}_{\text{Cand4}}$ using the shifting approach.

The advantage of this shifting-approach is that each new facility of an optimal solution X from the set $\mathcal{X}_{\text{Cand}\,4}$ resp. $\mathcal{X}_r$ can be analysed separately from the others. Thus, if one optimal new location is of particular interest for the decision maker, it is possible to analyse its neighbourhood and search for further optimal solutions without changing the locations of the other new facilities. It is also possible to examine the neighbourhood of a particular node for alternative locations of the corresponding new facility serving this customer in different optimal solutions. If only some of the new facilities are analysed for further optimal solutions, the complexity of the approach is reduced significantly.

6.5 Conclusion

In this section the results of this chapter are briefly summarised and the different presented approaches to solve a p-k-max problem are compared. The main properties (candidate set, maximum number of candidates and the complexity to compute the corresponding set of optimal solutions) of the presented approaches are summarised in Table 6.1.

The first approach (see Algorithm 9) evaluates all candidate solutions of the finite dominating set $\text{Cand4} = EQ \times (EQ \cup V)^{p-1}$. It is guaranteed to find at least one optimal solution of the p-k-max problem, and moreover all optimal solutions in the candidate set Cand4 are found. A drawback of this approach is that its computation time w.r.t. the number of nodes and the number of new facilities to locate is very high, even for rather small problems. Thus, it is not suitable for practical usage except for small instances. However, it is a straight forward approach and it can be extended easily to compute in parallel optimal solutions for all possible values of the parameter k with $k \in \{1, \ldots, n-1\}$.

The recursive approach (see Algorithm 10) is also based on the finite dominating set Cand4 and returns a set $\mathcal{X}_r$ of optimal solutions. Every candidate solution is generated recursively such that it can be decided for each individual new facility whether this candidate can lead to an optimal solution. Thus, in general the majority of candidates in Cand4 do not have to be considered. This leads to a significant acceleration such that the recursive approach is in practice much faster. However, this is not reflected in the worst case complexity which is even worse than the complexity of evaluating all candidates in Cand4. The computation time strongly depends on the value of the parameter k as it directly influences the k-max value and the possible objective function values are tested in non-decreasing order. The recursive approach guarantees to find at least one optimal solution of the p-k-max problem but in general not all optimal solutions in Cand4 are found, and thus $\mathcal{X}_r \subseteq \mathcal{X}_{\text{Cand4}}$. However, the found solutions are particularly relevant in practice because all new facilities are the center points of their corresponding covers, and are thus located "optimally" within their respective covers.

The local analysis is split in three parts: The approach based on

- the evaluation of the finite dominating set $V(L')$ (see Lemma 6.29) yielding a set of optimal solutions $\mathcal{X}_{\text{l-full}}$,
- the evaluation of the finite dominating set $V^{eq}(Y)$ (see Corollary 6.31) determining the set of optimal solutions $\mathcal{X}_{\text{l-red}}$,
- and the extension of this approach that determines all optimal solutions of the p-k-max problem given in the set $\mathcal{X}_{\text{l-all}}$ (see Theorem 6.32).

All these approaches have in common that they yield further optimal solutions that can not be found using the previous two algorithms since in addition to the equilibrium points and the nodes in V also balance points are considered as candidate locations.

The candidate set $V(L')$ is based on the intersection points of the regions in which the objective function is piecewise linear and concave. In contrast to that, the candidate set $V^{eq}(L(x_1^*))$ only consists of a small subset of these intersection points. The information to reduce the candidate set is based on the optimal solutions in $\mathcal{X}_r$. Hence, $\mathcal{X}_r$ has to be computed in a preprocessing step, but then the candidate set $V^{eq}(L(x_1^*))$ is in practice significantly smaller than $V(L')$. The advantage of the approach using $V(L')$ is that it can be used as a stand-alone approach, i.e., the sets of optimal solutions $\mathcal{X}_{\text{Cand4}}$ and $\mathcal{X}_r$ are not needed. Moreover, using the finite dominating set $V(L')$, the optimal solutions of the k-max problem for all values of k can be obtained simultaneously with just one evaluation of all its elements. Nevertheless, both approaches lead to the same optimal solutions. Thus, it holds that

$$\mathcal{X}_{\text{l-full}} = \mathcal{X}_{\text{l-red}} \supseteq \mathcal{X}_{\text{Cand4}} \supseteq \mathcal{X}_r.$$

The computational results show that the set of optimal solutions based on $V^{eq}(L(x_1^*))$ can be computed much faster (especially for rather large problems) than the one based on $V(L')$ even though both methods have the same worst case complexity. As $\mathcal{X}_{\text{l-full}}$ resp. $\mathcal{X}_{\text{l-red}}$ contain in general more optimal solutions than $\mathcal{X}_r$, the higher computation time may pay off here. Note that $\mathcal{X}_{\text{Cand4}}$, $\mathcal{X}_r$, $\mathcal{X}_{\text{l-full}}$ and $\mathcal{X}_{\text{l-red}}$ are sets with a finite number of optimal solutions and that they do in general not contain all optimal solutions of the p-k-max problem.

Set of optimal solutions	Candidate set	No. candidates	Complexity
$\mathcal{X}_{\text{Cand4}}$	$EQ \times (EQ \cup V)^{p-1}$	$\mathcal{O}(m^p n^{2p})$	$\mathcal{O}(m^p n^{2p+1} \log(n))$
$\mathcal{X}_r$	$EQ \times (EQ \cup V)^{p-1}$	$\mathcal{O}(m^p n^{2p})$	$\mathcal{O}(m^p n^{3p})$
$\mathcal{X}_{\text{l-full}}$	$V(L')$	$\mathcal{O}(m^p n^{2p})$	$\mathcal{O}(m^p n^{2p+1} \log(n))$
$\mathcal{X}_{\text{l-red}}$	$V^{eq}(L(x_1^*))$	$\mathcal{O}(m^p n^{2p-2})$	$\mathcal{O}(m^p n^{2p+1} \log(n))^{\dagger}$

†without complexity of computing $\mathcal{X}_r$

Table 6.1: Comparison of the main approaches to solve the p-k-max problem

In contrast to that, the set $\mathcal{X}_{\text{l-all}}$ as well as the set $\mathcal{X}_s$ obtained by the shifting-approach (see Algorithm 14) may contain an infinite number of optimal solutions. The local analysis as described in Algorithm 13 is the only approach that guarantees to find all optimal solutions

of the underlying p-k-max problem. However, this comes with a rather high worst case complexity of $\mathcal{O}(m^p n^{4p+2})$. In practice, it also becomes expensive to determine the optimal facets of the subdivisions, especially for large numbers of new facilities. For smaller problems (especially for $p = 2$), the solutions can be computed in reasonable time. The alternative optimal solutions are determined for all new facilities simultaneously. With the shifting-approach, in contrast, it is possible to focus on individual new facilities that are of special interest. This approach determines (infinitely many) alternative optimal solutions by shifting the individual new facilities of an optimal solution in $\mathcal{X}_{\mathsf{Cand4}}$ resp. $\mathcal{X}_r$ along the edges of the graph within a certain interval. The worst case complexity of $\mathcal{O}(m^{p+1} n^{2p+1})$ time to determine the whole set $\mathcal{X}_s$ can be reduced by concentrating on a small number of particularly interesting new facilities.

Chapter 7

Computational Results

In this chapter the results of the computational tests of the implemented algorithms for the p-k-max problem developed in the previous chapters are presented. The algorithms are compared with respect to their performance on randomly generated test instances. All numerical tests were performed on a compute server with 4 $\times$ Intel Xeon E7540 Hexacore (2,00 GHz) and 128 GB RAM. All algorithms were implemented and run in MATLAB, version R2013a.

Note that the implemented solution approaches would highly benefit from parallelisation with respect to the CPU-times as mentioned in the respective sections. However, the parallelisation would lead to more or less significant improvements depending on the algorithm. Thus, in order to obtain a reasonable comparison of the solution ideas, only one core of a single processor was used for the computation of the presented results for all algorithms. As an exemplary case, the effect of parallelisation is analysed for the evaluation of the finite dominating set Cand for $p = 1$ in the first section of this chapter.

The test instances used for the numerical tests are randomly generated Euclidean graphs which are very suitable to model the most common problems in practice. For a given number n of nodes and a density ρ of the graph, the input data is generated in the following way:

- Customers of the form $v_i = (x_1^i, x_2^i) \in \mathbb{Z}^2$ for $i = 1, \ldots, n$ are randomly placed in integer coordinates with mean 50 and standard deviation of 30 to favour a non-uniform distribution in the plane.
- The number m of edges of the graph is obtained from the density parameter ρ by $m = \lceil (\rho n(n-1))/2 \rceil$ such that $m = 0$ (empty graph) for $\rho = 0$ and $m = (n(n-1))/2$ (complete graph) for $\rho = 1$.
- The nodes of the graph are placed in the customers' locations. To guarantee the connectedness of the graph, at first a random spanning tree is constructed on these nodes. Afterwards, the missing $m - n + 1$ edges are added randomly to the spanning tree.
- Every edge e_{ab} of the graph is weighted (i.e., has a length l_{ab}) by the Euclidean distance between the corresponding customers v_a and v_b.
- A randomly generated vector $w \in \mathbb{N}^n$ with $w_i \in \{1, \ldots, 15\}$ for all $i = 1, \ldots, n$ gives the node weights. This range of weights is sufficiently large to generate an effect on the problem instances.

For every pair (n, ρ), 20 instances of the problem are tested where 15 of these instances are randomly weighted and 5 instances are unweighted. The presented results are the average of the results of these 20 instances of the same type. As an exemplary case, the test results are considered separately for weighted and unweighted graphs for $p = 1$ to analyse the dependence of the number of equilibrium points on the node weights. Moreover, for every instance further parameters as the value of k or the number of new facilities p are varied. These parameters are stated in the corresponding sections. All tests are conducted up to a maximum computation time of 18000 seconds. The following sections distinguish between p-k-max problems for $p = 1$, $p = 2$ and $p \geq 3$ new facilities.

7.1 1-k-max Problems

In this section two variants of the evaluation of the finite dominating set Cand $= EQ$ for the k-max problem ($p = 1$) with $k \in \{1, \ldots, n-1\}$ (see Algorithm 5) are tested and compared. The first variant simply evaluates all candidate points of the set EQ and chooses the solution yielding the smallest objective function value. The second variant sorts the set of equilibrium points non-decreasingly with respect to the weighted distances to their corresponding defining facilities. If the candidates are evaluated in this order, the algorithm can be stopped when the first feasible solution is found. To find the same set of optimal solutions with both approaches, all following equilibrium points yielding the same optimal objective value are also considered. Each test-graph of type (n, ρ) for fixed $n \in \{10, 20, 30, 50, 100\}$ and fixed $\rho \in \{0.1, 0.3, 0.5\}$ is tested for all values of $k \in \{1, 2, 0.25n, 0.5n, 0.75n\}$. Also problems with 200 new facilities and a density of 0.1 are considered. Note that graphs of type $(10, 0.1)$ are not connected since the density of 0.1 leads to only 5 edges which is not sufficient to connect all 10 vertices. Problems of this type are therefore not considered. In order to keep the implementations simple, the equilibrium points of the underlying graphs are not computed with the algorithm of Bentley and Ottmann (1979) here and in the following, but by computing the intersection points of each pair of weighted distance functions over an edge.

The finite dominating set Cand has at most $\mathcal{O}(n^2 m)$ elements as this is the maximum possible number of equilibrium points of a graph. Thus, the size of the candidate set depends obviously on the number of nodes and the density of the graph. The tests on the above described instances show that the theoretical upper bound of $\mathcal{O}(n^2 m)$ on the number of equilibrium points is widely overestimating in practice. The test instances have on average only 6.29% of the maximum possible number of equilibrium points which makes the finite dominating set much smaller than it could be assumed from the theoretical analysis. An interesting observation is that the tested unweighted graphs have on average 43.24% more equilibrium points than the weighted graphs. The larger the range of the single node weights is, the smaller is the number of equilibrium points in general. This may be explained by the fact that the corresponding graphs of weighted distance functions are less likely to intersect over an edge since also the range of different weighted distances is much larger in this case. This effect gets stronger when the number of nodes and edges of the graph increase. This

behaviour is reflected in Table 7.1 which gives the average number of equilibrium points for weighted and unweighted problems, respectively. It also shows the effect on the average time needed to solve the k-max problem with the unsorted variant of Algorithm 5. Note that the graphs of type $(20, 0.1)$ are tree graphs as the density of 0.1 for $n = 20$ leads to exactly 19 edges. Therefore, in the unweighted case, every graph of this type has exactly 190 equilibrium points. Moreover, $(20, 0.1)$-graphs have in general more equilibrium points in the weighted than in the unweighted case, contrary to all other types of graphs (for an explanation see Remark 4.21).

In the following analysis, the average values over all 20 test instances are taken and not distinguished for weighted and unweighted problems.

n	ρ	#wEQ	#uEQ	Δ(#) in %	wtime [s]	utime [s]	Δ(s) in %
10	0.1	-	-	-	-	-	-
	0.3	149	161	7.957	0.011	0.011	1.927
	0.5	313	329	5.374	0.020	0.020	1.502
20	0.1	362	190	-47.446	0.031	0.027	-12.523
	0.3	2 574	3 280	27.437	0.147	0.163	10.846
	0.5	4 868	6 836	40.420	0.261	0.303	16.182
30	0.1	2 572	2 796	8.701	0.180	0.186	2.986
	0.3	12 881	19 098	48.270	0.732	0.879	20.212
	0.5	23 838	36 249	52.067	1.294	1.589	22.816
50	0.1	25 051	35 429	41.429	1.621	1.936	19.456
	0.3	98 597	159 532	61.802	5.804	7.482	28.912
	0.5	200 367	283 725	41.603	10.804	13.156	21.777
100	0.1	402 011	677 279	68.473	30.166	39.523	31.018
	0.3	1 665 680	2 737 055	64.321	108.579	150.620	38.719
	0.5	3 109 222	4 705 625	51.344	196.411	253.178	28.902
200	0.1	6 791 481	12 642 004	86.145	640.122	976.462	52.543

Table 7.1: Comparison of the average number of equilibrium points for weighted (#wEQ) and unweighted (#uEQ) problems and the CPU-times of Algorithm 5 for unweighted (utime) and weighted (wtime) problems. The fifth column contains the relative deviation of #wEQ from #uEQ and the last column specifies analogously the relative deviation of wtime from utime. Number n of existing facilities and density ρ of the graph are given, $k = 1$ and $p = 1$ are fixed.

Unsorted evaluation of Cand = *EQ* (Algorithm 5) Table 7.2 shows the results of the numerical tests for the unsorted variant of Algorithm 5. As expected, the number of equi-

librium points increases with n and ρ and the corresponding CPU-times clearly increase with the number of candidates to evaluate. The computation times do not depend on the value of the parameter k as in any case all equilibrium points have to be tested.

n	ρ	$\lvert EQ \rvert$	$k = 1$	$k = 2$	$k = 0.25n$	$k = 0.5n$	$k = 0.75n$
10	0.1	-	-	-	-	-	-
	0.3	152	0.011	0.011	0.011	0.012	0.037
	0.5	317	0.020	0.020	0.020	0.020	0.020
20	0.1	319	0.030	0.030	0.030	0.030	0.030
	0.3	2 750	0.151	0.151	0.151	0.151	0.152
	0.5	5 360	0.272	0.272	0.272	0.272	0.273
30	0.1	2 628	0.182	0.182	0.182	0.182	0.183
	0.3	14 435	0.769	0.768	0.768	0.767	0.768
	0.5	26 941	1.367	1.372	1.375	1.377	1.380
50	0.1	27 645	1.700	1.701	1.701	1.705	1.708
	0.3	113 831	6.223	6.234	6.262	6.277	6.277
	0.5	221 207	11.392	11.424	11.422	11.367	11.413
100	0.1	470 828	32.506	32.262	32.452	32.620	32.817
	0.3	1 933 524	119.090	119.243	119.578	119.743	119.596
	0.5	3 508 322	210.603	210.857	210.919	210.594	210.909
200	0.1	8 254 112	724.207	724.292	724.253	726.431	732.103

Table 7.2: Evaluation of the finite dominating set Cand (see Algorithm 5): Numbers of equilibrium points and CPU-times in seconds for given number of n existing facilities and given density ρ of the graph, distinguished w.r.t. the value of k

Sorted evaluation of Cand = *EQ* (Second variant of Algorithm 5) The results of the tests for the solution of the k-max problem with the sorted evaluation of the set Cand are given in Table 7.3. Again, the CPU-times increase with the number of equilibrium points but much slower than it is the case for the unsorted approach. The amount of considered candidates in the sorted approach strongly depends on the parameter k: The larger the value of k, the smaller is the optimal objective function value of the underlying problem and thus usually also the number of candidates that have to be evaluated until the first optimal solution is found.

Comparison of both variants of Algorithm 5 A comparison of the number of considered candidates and the computation times of the two approaches are shown in Figure 7.1. It can be seen that the sorted variant of Algorithm 5 considers on average over all test problems

			$k = 1$		$k = 2$		$k = 0.25n$		$k = 0.5n$		$k = 0.75n$	
n	ρ	$\lvert EQ\rvert$	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]
	0.1	-	-	-	-	-	-	-	-	-	-	-
10	0.3	152	132	0.011	113	0.010	95	0.010	57	0.010	9	0.030
	0.5	317	245	0.019	200	0.018	158	0.017	84	0.016	10	0.015
	0.1	319	312	0.029	303	0.029	257	0.028	166	0.027	63	0.048
20	0.3	2 750	2 148	0.142	1 898	0.137	1 251	0.123	650	0.111	178	0.102
	0.5	5 360	3 871	0.249	3 289	0.237	2 232	0.215	1 021	0.190	256	0.176
	0.1	2 628	2 354	0.179	2 235	0.176	1 469	0.159	848	0.145	260	0.133
30	0.3	14 435	10 988	0.722	9 614	0.691	5 430	0.599	2 532	0.537	678	0.498
	0.5	26 941	19 076	1.257	16 769	1.205	9 839	1.053	4 565	0.938	1 030	0.862
	0.1	27 645	23 055	1.637	21 146	1.589	11 691	1.347	5 367	1.187	1 465	1.089
50	0.3	113 831	91 026	6.072	81 407	5.827	36 559	4.680	17 018	4.185	4 600	3.872
	0.5	221 207	172 686	11.185	155 682	10.749	69 015	8.542	30 742	7.569	8 143	7.001
	0.1	470 828	380 800	31.578	355 734	30.701	151 828	23.665	70 106	20.849	20 327	19.186
100	0.3	1 933 524	1 566 185	120.827	1 474 780	117.725	601 137	87.724	254 763	75.980	68 455	69.634
	0.5	3 508 322	3 000 217	230.525	2 779 655	222.791	1 192 401	167.667	481 850	143.078	126 998	130.920
200	0.1	8 254 112	6 951 438	742.594	6 537 644	718.762	2 335 848	474.856	940 131	394.400	244 515	354.608

Table 7.3: Evaluation of the finite dominating set Cand (see Algorithm 5) where the equilibrium points are sorted non-decreasingly with respect to the distances to their corresponding defining facilities: Numbers of considered solutions (#cSol.) and CPU-times in seconds for given number of n existing facilities and given density ρ of the graph, distinguished w.r.t. the value of k

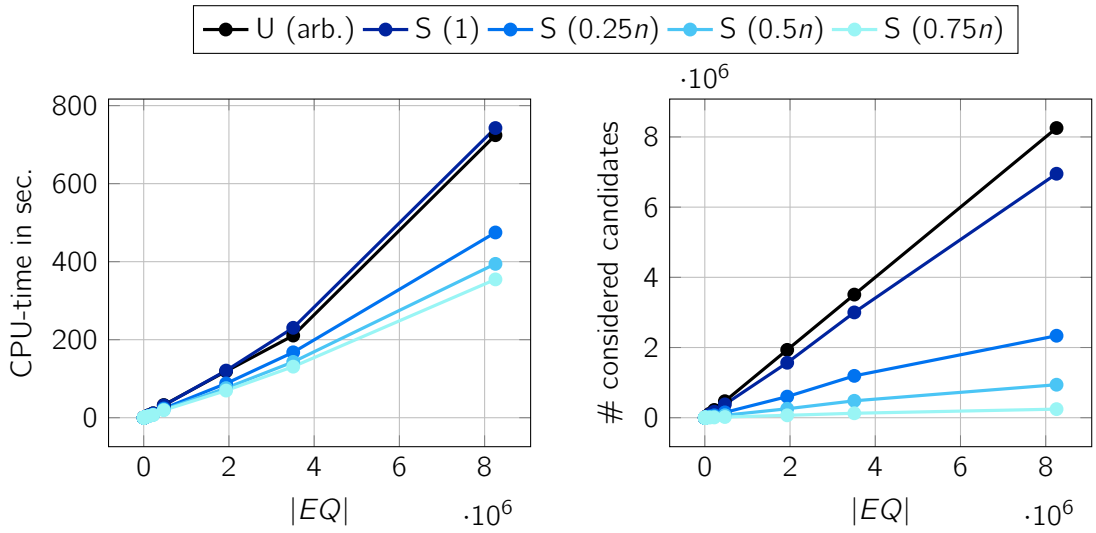


Fig. 7.1: CPU times in seconds and number of tested candidates for the evaluation of Cand in the sorted (S) and unsorted (U) variant. The unsorted evaluation of Cand is independent of k, whereas each blue line corresponds to a fixed k for the sorted evaluation.

40% of the equilibrium points for $k = 0.25n$, 20% for $k = 0.5n$ and only 5% for $k = 0.75n$. Moreover, the sorted approach needs around 78% of the CPU-time of the unsorted approach for $k = 0.25n$, 70% for $k = 0.5n$ and 63% for $k = 0.75n$. Note that the relative savings in the number of candidates and the computation time increase with the problem size. The savings in terms of the computation time are smaller than the savings with respect to the number of considered candidates as the sorting of the equilibrium points takes some extra time. Moreover, for the sorted approach, all equilibrium points have to be computed and stored before they can be evaluated. Depending on the problem size, this leads to large data sets that have to be stored. However, the sorted evaluation of the set Cand is in general more efficient than the unsorted version. Only for very small values of k (here for example $k = 1$ or $k = 2$), the sorted version is slower since nearly all candidates have to be considered anyway and the extra effort needed for the sorting does not pay off in general.

Implementation of Algorithm 5 using parallelisation As an examplary case, the effect of a parallelised implementation[1] of Algorithm 5 on the CPU-times is shown in Table 7.4. For the parallelisation, the evaluation of the different candidates is carried out simultaneously on 24 processors using 48 threads. It can be seen that the CPU-times of the parallelised version are slightly higher than the computation times of the not-parallelised implementation for the smallest problems up to type $(20, 0.1)$. These problems are too small such that the distribution to the different processors and the final merging of the results takes more time than just evaluating all candidates on one processor. As expected for all problems of larger type, the computation times highly benefit from the parallelisation and the savings in time

[1] Implemented by Pascal Colling in the context of a programming course at the University of Wuppertal in the winter term 2016/17, supervised by Dr. Michael Stiglmayr from the Working Group Optimization and Approximation and by myself.

even increase with increasing problem size. This is due to the fact that the influence of the computational overhead of the parallelisation, i.e., task starting and termination times, idle, synchronisations, data communications and the presence of serial components, becomes smaller the larger the underlying problem is. For further information on parallel programming see, for example, Grama (2003). For practical problems, the algorithms presented in this thesis should be implemented in parallel.

n	ρ	|EQ|	time [s]	Δ(s) in %
10	0.1	-	-	-
	0.3	152	0.076	691.560
	0.5	317	0.076	380.522
20	0.1	319	0.076	254.300
	0.3	2 750	0.104	68.890
	0.5	5 360	0.125	45.969
30	0.1	2 628	0.076	41.612
	0.3	14 435	0.175	22.708
	0.5	26 941	0.281	20.532
50	0.1	27 645	0.323	19.000
	0.3	113 831	1.088	17.478
	0.5	221 207	2.030	17.822
100	0.1	470 828	3.563	10.961
	0.3	1 933 524	13.316	11.181
	0.5	3 508 322	23.504	11.160
200	0.1	8 254 112	76.345	10.542
	0.3	32 783 232	282.595	11.168

Table 7.4: Effect of a parallelisation on the unsorted evaluation of Cand (Algorithm 5): Numbers of equilibrium points, CPU-times in seconds of the parallelised implementation and percentage of the CPU-time in comparison to the not-parallelised implementation of Algorithm 5 for given number of n existing facilities and given density ρ of the graph

7.2 2-k-max Problems

In this section most of the Algorithms presented in Chapter 6 to solve the p-k-max problem are tested and compared for $p = 2$ new facilities. This includes the evaluation of the finite dominating set Cand4 (Algorithm 9), the recursive approach (Algorithm 10), the evaluation

of the finite dominating set $V(L')$ given in Lemma 6.21 (here also referred to as the full local analysis) and the local analysis approach (Algorithm 11; here also referred to as the reduced local analysis). Each test-graph of type (n, ρ) for fixed $n \in \{10, 20, 30, 50\}$ and fixed $\rho \in \{0.1, 0.3, 0.5\}$ is tested for all values of $k \in \{1, 2, 0.25n, 0.5n, 0.75n\}$, provided that the time bound of 18 000 seconds is not violated. Note that graphs of type $(10, 0.1)$ are not connected and are therefore again omitted. The observations made in Section 7.1 regarding the size of the set of equilibrium points and its dependence on the node weights of course also hold for $p = 2$ as the set of equilibrium points does not depend on the value of p.

Evaluation of Cand4 = $EQ \times (EQ \cup V)$ (Algorithm 9) Table 7.5 shows the results of the numerical tests for the evaluation of the candidate set Cand4 $= EQ \times (EQ \cup V)$. The cardinality of Cand4 is directly deducible from the number of equilibrium points and thus increases with the number of nodes n, the density ρ of the graph and depends also on the node weights. In the worst case, the number of candidates grows more than quadratically with the number of equilibrium points. As a consequence, the corresponding CPU-times clearly increase with the size of the underlying graph. Note that symmetric candidates are not considered, i.e., if a candidate $\{x_1, x_2\}$ with $x_1, x_2 \in EQ$ is evaluated, the candidate $\{x_2, x_1\}$ is not tested. The computation times do not depend on the value of the parameter k as in all cases the same number of candidates has to be tested.

Recursive approach (Algorithm 10) The results of the tests w.r.t. the CPU-times and the considered candidates for the recursive approach are given in Table 7.6. The CPU-times increase with the problem size but much slower than it is the case for the evaluation of Cand4, i.e., the recursive approach performs much better than the evaluation of the complete finite dominating set on nearly all test instances. A candidate solution in the recursive approach is defined as a completed solution $X = \{x_1, x_2\}$ where both new locations of X are located but X does not have to be feasible, i.e., it does not have to satisfy condition (6.3). Only these completed candidate solutions are counted in #cSol. in Table 7.6. The number of the completed solutions strongly depends on the parameter k: The larger the value of k, the smaller is the optimal objective function value of the underlying problem and therefore also the number of candidates that have to be evaluated since the possible objective values are considered in non-decreasing order. Thus, the 2-k-max problem for $k \geq 2$ is in general faster solvable with the recursive approach than the 2-center problem. However, there are solution procedures especially designed for p-center problems which yield a better performance as it can be seen, for example, in Chen and Chen (2009) or Calik and Tansel (2013). Moreover, the results of the tests show that the sizes of the two covers corresponding to the two optimal facilities do depend on the underlying instance, i.e., the cluster belonging to an optimal facility x_1 may be much larger than the cover of the corresponding optimal facility x_2 for one instance while the two covers may be nearly equally large for another instance.

Comparison of the recursive approach and the evaluation of Cand4 A comparison of the number of considered candidates and the computation times of the two approaches is shown in Figure 7.2. It can be seen that the recursive approach considers, on average over

n	ρ	$\lvert EQ\rvert$	#Cand	$k=1$	$k=2$	$k=0.25n$	$k=0.5n$	$k=0.75n$
10	0.1	-	-	-	-	-	-	-
	0.3	152	24 655	1.096	1.094	1.096	1.095	1.116
	0.5	317	103 530	4.400	4.406	4.412	4.401	4.421
20	0.1	319	107 911	4.527	4.517	4.512	4.520	4.534
	0.3	2 750	7 618 604	286.659	286.500	286.860	286.498	286.604
	0.5	5 360	28 838 948	1 107.097	1 107.162	1 106.440	1 103.246	1 100.345
30	0.1	2 628	6 984 960	294.390	294.242	294.651	294.642	295.109
	0.3	14 435	208 803 720	8 320.175	8 319.095	8 291.395	8 287.959	8 315,435
	0.5	26 941	726 625 711	> 18 000	> 18 000	> 18 000	> 18 000	> 18 000
50	0.1	27 645	765 628 275	> 18 000	> 18 000	> 18 000	> 18 000	> 18 000

Table 7.5: Evaluation of the finite dominating set Cand4 (see Algorithm 9): Numbers of candidates and CPU-times in seconds for given number of n existing facilities and given density ρ of the graph, distinguished w.r.t. the value of k

		$k = 1$		$k = 2$		$k = 0.25n$		$k = 0.5n$		$k = 0.75n$	
n	ρ	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]
10	0.1	-	-	-	-	-	-	-	-	-	-
	0.3	591	0.168	421	0.132	301	0.105	55	0.051	1	0.042
	0.5	1 867	0.523	1 185	0.378	664	0.267	133	0.121	1	0.031
20	0.1	6 882	1.156	6 349	1.057	4 707	0.805	1 431	0.345	86	0.092
	0.3	165 580	39.862	118 471	31.396	59 034	19.282	13 832	7.429	573	1.242
	0.5	538 476	137.101	359 937	102.868	155 242	56.749	36 364	21.064	854	2.748
30	0.1	245 096	59.176	209 628	51.508	96 472	28.107	31 554	12.605	1 865	2.310
	0.3	4 718 514	1 390.232	3 511 194	1 122.578	1 061 135	489.775	256 512	203.543	9 769	28.174
	0.5	14 181 114	4 364.916	11 060 342	3 647.557	3 411 724	1 615.885	738 091	609.108	16 067	66.392
50	0.1	22 984 006	7 074.623	19 269 903	6 156.654	5 527 881	2 474.902	1 191 697	937.685	94 311	202.334
	0.3	-	> 18 000	-	> 18 000	-	> 18 000	-	> 18 000	728 439	2 099.356
	0.5	-	> 18 000	-	> 18 000	-	> 18 000	-	> 18 000	2 177 602	7 921.215

Table 7.6: Evaluation of the recursive approach (Algorithm 10): Numbers of completely constructed solutions (#cSol) and CPU-times in seconds for given number of n existing facilities and given density ρ of the graph, both distinguished w.r.t. the value of k

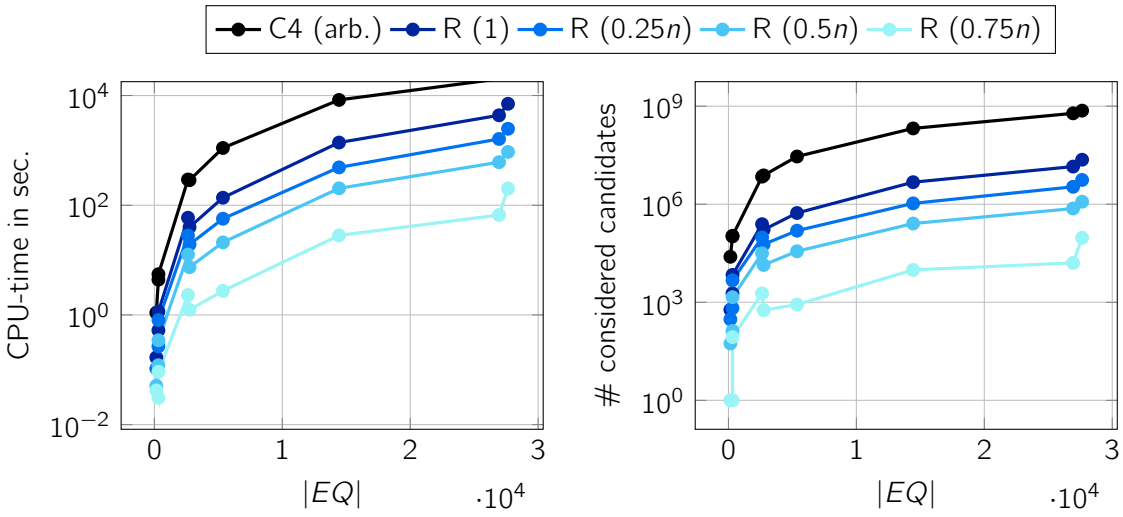


Fig. 7.2: CPU times in seconds and number of tested candidates for the evaluation of Cand4 (C4) and the recursive approach (R). The evaluation of Cand4 is independent of the value of k. Each blue line corresponds to a specific value of k in the recursive approach.

all test problems, 2.81% of the candidates in Cand4 for $k = 1$, 1.18% for $k = 0.25n$, 0.31% for $k = 0.5n$ and only 0.02% for $k = 0.75n$. However, note that also a lot of not-completed solutions are considered in the recursive approach which are not counted for #cSol. The dependence of the results on k is reflected even better in the computation time: The recursive approach needs around 16.55% of the CPU-time of the evaluation of Cand4 for $k = 1$, 8.67% for $k = 0.25n$, 3.75% for $k = 0.5n$ and 1.19% for $k = 0.75n$.

Full local analysis (Lemma 6.21) Table 7.7 summarises the size of the finite dominating set $V(L')$ for the tested instances and the corresponding computation times for its evaluation. Again, the size of the candidate set (and therefore also the CPU-times) increases with the size of the underlying problem, i.e., it depends on the number of customers and the density of the graph. As expected, the number of candidates and the computation times do not depend on the value of k as the constructed subdivisions are independent of k. However, a similar dependence of the size of $V(L')$ on the node weights as for the set EQ can be observed: The tested unweighted graphs have on average 44.81% more candidate points than the weighted problems. Again, the larger the differences of the node weights are, the smaller is the number of candidate points since the corresponding graphs of the weighted distance functions are less likely to intersect over a given edge if the ranges of objective values are larger. Moreover, the results for the tested instances show that the theoretical maximum number of $(18n^4 - 3n^2)m^2$ intersection points of the subdivisions is widely overestimating in practice. For the considered test instances, the set $V(L')$ contains on average only 6.38% of the maximum possible number of candidate points. Nevertheless, the finite dominating set $V(L')$ is of rather theoretical interest as the number of candidates gets usually very large. The cardinality of $V(L')$ is significantly larger than the cardinality of Cand4: Cand4 contains on average 1.86% of the candidates in comparison to $V(L')$. If only one optimal solution

is needed, the recursive approach performs much better. In the extreme case ($k = 1$) it needs only around 0.86% of the CPU-time needed by the full local analysis. However, the evaluation of the finite dominating set $V(L')$ contains in general more alternative optimal solutions of the problem. Note that the given results for problems of type $(30, 0.1)$ are the average over the 15 unweighted instances. The weighted instances could not be solved within the time bound of 18000 seconds, but needed around 22 000 seconds on average.

n	ρ	#Cand	$k = 1$	$k = 2$	$k = 0.25n$	$k = 0.5n$	$k = 0.75n$
10	0.1	-	-	-	-	-	-
	0.3	1 043 016	11.473	11.466	11.487	11.484	11.486
	0.5	4 667 700	50.939	50.981	51.016	50.986	50.992
20	0.1	11 783 328	200.338	200.192	200.590	200.254	200.281
	0.3	453 873 031	8 029.164	8 026.415	8 032.070	8 012.247	8 069.942
	0.5	> 900 000 000	> 18 000	> 18 000	> 18 000	> 18 000	> 18 000
30	0.1	291 740 500	7 883.214	7 896.173	7 904.613	7 899.809	7 903.576
	0.3	> 700 000 000	> 18 000	> 18 000	> 18 000	> 18 000	> 18 000

Table 7.7: Evaluation of the finite dominating set $V(L')$ (see Lemma 6.29): Numbers of candidates and CPU-times in seconds for given number of n existing facilities and given density ρ of the graph, distinguished w.r.t. the value of k

Reduced local analysis (Algorithm 11) The results of the tests for Algorithm 11 are given in Table 7.8. Note that the discrete version of the algorithm is tested, i.e., not all (infinitely many) optimal solutions are computed but only the solutions belonging to the finite dominating set $V^{eq}(L(x_1^*))$. From these, the complete optimal set can be easily deduced. The number of candidates gives the number of intersection points for all subdivisions, it does not include the completed candidates used in the recursive approach which has to be called as a preprocessing. The CPU-time, in contrast, is measured for the overall procedure, including the recursive approach. As for the finite dominating set $V(L')$, the number of intersection points of the subdivisions increase with the problem size. The number of intersection points for a fixed subdivision does not depend on k. However, as the cardinality of the set Y of objective function value defining facilities may vary w.r.t. the value of k, also the total amount of candidates might differ with varying values of k. The computation times, in contrast, decrease with an increasing value of the parameter k since the recursive approach depends on k. As $V^{eq}(v_1^*) \subseteq V(L')$ (see Remark 6.22), a similar dependence of the size of the set $V^{eq}(x_1^*)$ on the weights can be observed as for the set $V(L')$. It should be mentioned that the given CPU-times are mainly determined by the recursive approach. The local analysis itself takes only 14.38% on average of the overall computation times. Hence, the local analysis computes in general many further optimal solutions which are not contained in the set $EQ \cup V$ with only small additional effort.

		$k = 1$		$k = 2$		$k = 0.25n$		$k = 0.5n$		$k = 0.75n$	
n	ρ	#Cand.	time [s]	#Cand.	time [s]	#Cand.	time [s]	#Cand.	time [s]	#Cand.	time [s]
10	0.1	-	-	-	-	-	-	-	-	-	-
	0.3	1 443	0.205	1 447	0.164	1 495	0.136	1 499	0.075	1 467	0.069
	0.5	3 200	0.622	3 242	0.461	3 315	0.338	3 315	0.174	3 173	0.073
20	0.1	5 122	1.371	5 079	1.260	5 035	0.978	5 044	0.465	5 033	0.180
	0.3	30 056	44.915	30 351	35.598	32 559	22.119	31 904	8.813	31 799	1.815
	0.5	59 202	153.727	58 902	115.934	59 796	64.449	60 737	24.503	60 283	3.858
30	0.1	33 344	60.932	33 389	53.101	33 227	29.416	33 273	13.567	33 581	2.924
	0.3	164 714	1 411.676	172 522	1 140.710	175 160	503.864	174 677	211.478	175 079	31.636
	0.5	305 911	4 502.882	306 436	3 762.006	31 4682	1 680.585	314 009	639.868	302 514	73.939
50	0.1	363 792	7 135.507	355 796	6 196.555	349 921	2 495.658	355 217	949.539	362 752	212.869
	0.3	-	$> 18\,000$	-	$> 18\,000$	-	$> 18\,000$	-	$> 18\,000$	1 425 648	2 220.202
	0.5	-	$> 18\,000$	-	$> 18\,000$	-	$> 18\,000$	-	$> 18\,000$	2 569 633	7 684.503

Table 7.8: Evaluation of the reduced local analysis (Algorithm 11): Numbers of candidates and CPU-times in seconds for given number of n existing facilities and given density ρ of the graph, both distinguished w.r.t. the value of k

Comparison of the full and the reduced local analysis Obviously, the reduced local analysis yields much better results in terms of the CPU-time for all test instances than the full local analysis. The reduced analysis needs on average over all test problems just 0.054% of the intersection points which does not vary significantly for different values of k. Moreover, the reduced local analysis needs only 1% of the CPU-times as compared to the full local analysis for $k = 1$, 0.60% for $k = 0.25n$, 0.30% for $k = 0.5n$ and 0.18% for $k = 0.75n$. This decreasing behaviour of the CPU-times with respect to k holds as the computation times for the recursive approach (and therefore also of the reduced local analysis) depend on k in this way.

7.3 p-k-max Problems

In this section the evaluation of the finite dominating set Cand4 (see Algorithm 9) and the recursive approach (see Algorithm 10) for the solution of the p-k-max problem with $p \geq 3$ are compared. In order to analyse the dependence of the two solution approaches on the number p of new facilities to locate, the value of the parameter k is fixed to $k = 2$ and the density of the underlying graphs is fixed to $\rho = 0.1$ in this section. Every instance of $n \in \{7, 10, 15, 20, 25, 30, 35\}$ is tested for all values of $p \in \{3, 5, 7, 10\}$, provided the time bound of 18 000 seconds per instance is not exceeded. The average computation times and the average numbers of considered candidates are stated in Table 7.9 for the evaluation of Cand4 and in Table 7.10 for the recursive approach.

	$p = 3$		$p = 5$		$p = 7$	
n	#Cand.	time [s]	#Cand.	time [s]	#Cand.	time [s]
7	$3.4 \cdot 10^4$	0.660	$4.2 \cdot 10^7$	39.609	-	-
10	$3.5 \cdot 10^5$	6.010	$1.9 \cdot 10^9$	1792.140	$1.0 \cdot 10^{13}$	$> 18\,000$
15	$5.1 \cdot 10^6$	88.655	$1.6 \cdot 10^{11}$	$> 18\,000$	$5.1 \cdot 10^{15}$	$> 18\,000$
20	$3.2 \cdot 10^7$	534.886	$3.4 \cdot 10^{12}$	$> 18\,000$	$3.5 \cdot 10^{17}$	$> 18\,000$
25	$1.2 \cdot 10^9$	$> 18\,000$	$1.5 \cdot 10^{15}$	$> 18\,000$	$1.8 \cdot 10^{21}$	$> 18\,000$

Table 7.9: Evaluation of the finite dominating set Cand4 (Algorithm 9): Numbers of candidates and CPU-times in seconds, both distinguished w.r.t. the number p of new facilities and w.r.t. a given number of n existing facilities. Density $\rho = 0.1$ and $k = 2$ are fixed.

Evaluation of Cand4 = $EQ \times (EQ \cup V)^{p-1}$ (Algorithm 9) The size of the finite dominating set Cand4 is

$$|\,\text{Cand4}\,| = |EQ| \cdot (|EQ| + n)^{p-1}$$

which grows exponentially in p and leads thus to a huge number of possible candidate solutions even for relatively small values of p. Hence, the time bound of 18 000 seconds is

reached quickly for the evaluation of Cand4. Clearly, the same dependence of the size of Cand4 on the node weights as for $p = 2$ can be observed. However, it is possible to solve problems with a small number of customers and up to five new facilities. Note that (similar to the case of $p = 2$) symmetric solutions are not considered for the evaluation of the possible candidate solutions, i.e., a fixed set of p new facilities is considered only once. Therefore, the number of evaluated candidates is much smaller than $|\,\text{Cand4}\,|$. As the number of not considered candidates increases with the value of p, it is possible to solve, for example, the problems for $n = 10$ and $p = 5$ faster than the problems for $n = 25$ and $p = 3$ even though the candidate set Cand4 is larger in this case (see Table 7.9). Again, the computation time and the number of candidates do not depend on the value of the parameter k.

Recursive Approach (Algorithm 10) Even though the computation time also increases with the problem size, the recursive approach performs much better for the tested instances than the evaluation of the candidate set Cand4. It is defined analogously to the definition stated in Section 7.2 for $p = 2$: A completed solution $X = \{x_1, \ldots, x_p\}$ is a solution where all p new facilities of X are located, but X does not have to be feasible, i.e., the new facilities do not have to satisfy condition (6.3). The number of completed solutions does not only depend on p but rather on the underlying graph and the concrete problem instance. The dependence on k is the same as observed for $p = 2$: The number of considered solutions and the computation time increase with decreasing value of k as the possible objective function values are considered in non-decreasing order. Moreover, the results of the tests show that the sizes of the covers corresponding to the p optimal facilities do strongly depend on the distribution of the existing facilities, i.e., the covers associated to an optimal solution of one problem might be nearly equally large while the covers for another problem might be very different in size.

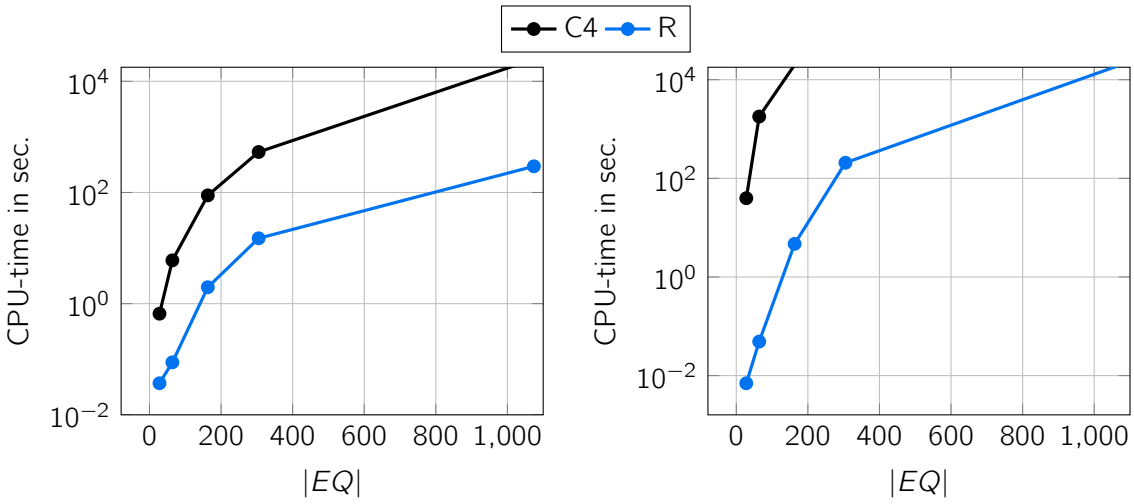


Fig. 7.3: Comparison of the CPU times in seconds for the evaluation of Cand4 (C4) and the recursive approach (R) for $p = 3$ (left) and $p = 5$ (right) and fixed $k = 2$, $\rho = 0.1$.

	$p = 3$		$p = 5$		$p = 7$		$p = 10$	
n	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]	#cSol.	time [s]
7	1	0.037	1	0.007	-	-	-	-
10	64	0.088	5	0.049	2	0.013	-	-
15	3 072	1.975	478	4.692	6	1.742	3	0.114
20	30 398	14.990	59 397	207.082	4 114	202.169	24	29.091
25	544 475	295.664	-	> 18 000	-	> 18 000	-	> 18 000
30	6 491 516	4 239.441	-	> 18 000	-	> 18 000	-	> 18 000
35	-	> 18 000	-	> 18 000	-	> 18 000	-	> 18 000

Table 7.10: Evaluation of the recursive approach (Algorithm 10): Numbers of completely constructed solutions (#cSol) and CPU-times in seconds, both distinguished w.r.t. the number p of new facilities and w.r.t. a given number of n existing facilities. Density $\rho = 0.1$ and $k = 2$ are fixed.

Comparison of the evaluation of Cand4 and the recursive approach The relation of the computation times of the two approaches is shown in Figure 7.3. The recursive approach considers in the worst case of the tested instances 0.044% of the possible solutions in the set Cand4 and needs 3.04% of the CPU-time. Thus, the recursive approach performs much better on all tested instances. However, note that there are in general many possible solutions tested by the recursive approach which are not completed (and therefore not counted in #cSol.) because one of the later located facilities may have a larger weighted distance to its defining facilities. Clearly, there is room for improvement for higher values of p also for the recursive approach.

7.4 Conclusion

Nearly all algorithms introduced in this thesis were implemented in Matlab and the results of the exhaustive computational tests were summarised in this chapter. More precisely, two variants of the evaluation of the finite dominating set Cand (Algorithm 5), the evaluation of the finite dominating set Cand4 (Algorithm 9), the recursive approach (Algorithm 10), the evaluation of the finite dominating set $V(L')$ given in Lemma 6.21 and the local analysis approach (Algorithm 11) were run on different randomly generated instances to compare the solution approaches.

For $p = 1$, instances with up to 200 existing facilities were tested. The tests show that the sorted evaluation of the candidate set Cand yields in general much better CPU-times than the unsorted evaluation.

For $p = 2$, instances with up to 50 existing facilities could be solved in reasonable time. As expected, the recursive approach (Algorithm 10) yields the best CPU-time to solve the 2-k-max problem. However, with the reduced local analysis (Algorithm 11), many further optimal solutions can be found with only small additional effort.

For multi-facility problems, instances with up to 20 existing facilities and 3 new facilities could be solved. For $p = 5$ instances of up to 10 existing facilities were considered. Clearly, the recursive approach yields also for the general case of p new facilities much better results than the simple evaluation of all candidates of the finite dominating set Cand4.

An important observation is that all finite dominating sets have in practice much less elements than it could be suggested from their theoretical upper bounds.

Moreover, a clear difference in the number of equilibrium points and thus also in the CPU-times for weighted and unweighted instances could be shown: Unweighted problems have in general much less equilibrium points than weighted problems. The effect on the computation times were stated exemplary for the case of $p = 1$.

The effect of a parallelised implementation on the CPU-times was illustrated for k-max problems. The parallelised implementation of Algorithm 5 needed for the largest instances on average only 11% of the computation time as compared to the non-parallelised implementation. Clearly, also the computation times of the remaining solution approaches would highly benefit from parallelisation.

Chapter 8

The Impact of Different Demands

This chapter deals with a problem that occurs in modelling a p-k-max problem on a weighted graph. High weights in p-k-max problems with $k \geq 2$ can have an unwanted effect because they are likely to become outliers even though this is not intended by labelling a node with a high weight. As a high node weight usually indicates a high level of importance of the customer, it would be desirable to locate a new facility as close as possible to the corresponding high weighted node.

To every node $v_i \in V$ of a graph G a positive weight $w_i > 0$, $i = 1, \ldots, n$, is assigned. These weights can be used for expressing the demand or different levels of importance of the existing facilities. When a node does not represent a single customer but a cluster of clients (like a district of a town), the weight can also represent the number of customers in the district. Regardless of the interpretation of the weights, it holds: The higher the value w_i is, the more important is the customer at v_i. The effect of a highly weighted node in a center problem is that the optimal new location is attracted more by this facility than by a facility with a lower weight because the weighted distance $w_i d(v_i, x)$ increases with higher weights. Thus, to compensate this and to not worsen the objective function value too much, the optimisation process will try to place the new facility nearer to the highly weighted customers.

In contrast to center location problems, the usage of node weights in location problems with k-max functions may lead to unwanted results as the following example shows.

Example 8.1. *Figure 8.1 shows the solutions x^1, x^2 and x^3 of a k-*max *problem on a weighted path graph G for the parameters $k = 1$, $k = 2$ resp. $k = 3$. The optimal new locations lead to the distance vectors $d(V, x^k)$ which are*

$$d(V, x^1) = \left(\frac{28}{9}, \frac{38}{9}, \frac{80}{9}, \frac{1}{9}, \frac{80}{9}\right)^\top$$

$$d(V, x^2) = \left(\frac{16}{9}, \frac{14}{9}, \frac{16}{9}, \frac{11}{9}, \frac{200}{9}\right)^\top$$

$$d(V, x^3) = \left(\frac{3}{2}, 1, 4, \frac{3}{2}, 25\right)^\top.$$

The optimal new facility x^1 of the center problem behaves as intended with the high weights of nodes v_3 and v_5: The optimal solution is attracted the most by these two nodes and lies therefore exactly in their weighted midpoint, i.e., slightly closer to v_5 than to v_3 because

$w_5 > w_3$.

Contrary to this, for $k = 2$ the weights do not have the desired effect. For the optimal new location x^2 the corresponding outlier is node v_5 (marked by the dotted circle in the same color) because its weighted distance to x^2 is the largest among the existing facilities. As a consequence, the node with the highest importance is an outlier in this optimal solution. This usually contradicts the intention of the high weight to bring the new facility as close as possible to this customer. Here, node v_3 with the second highest weight attracts the solution the most.

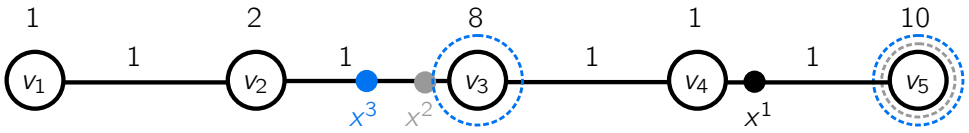


Fig. 8.1: Optimal location x^1, x^2 resp. x^3 of a k-max problem on the graph for $k = 1$, $k = 2$ resp. $k = 3$ and its corresponding outliers

This unwanted effect is even stronger for $k = 3$. The optimal solution x^3 leads to the outliers v_3 and v_5, i.e., the two nodes with the highest weights were not taken into account to find the optimal new location. The location of x^3 is determined by the nodes v_1 and v_4, that means the closeness to the high weighted node v_3 is incidental here.

An effect as described in the example appears often, mainly if the edge weights l_j for $j = 1, \ldots, m$ are much smaller than some node weights w_i, $i \in \{1, \ldots, n\}$. Then, a high weight w_i has a large impact on the weighted distance $w_i d(v_i, x)$ from the node v_i to a new location x such that it is likely to become one of the $k-1$ largest distances, even though the unweighted distance may be rather small. As a consequence, high weighted customers are often outliers and do not influence the location of the new facility, contrary to the intention of the weights.

The above example also illustrates two further difficulties when using weights. The outliers can lie "in the middle" of the graph, i.e., on a shortest path between center defining points. Intuitively, this contradicts the natural understanding of outliers to be very distant customers. Despite this, it does not make sense in practice to label facilities as outliers that lie very close, or even on the shortest path to a covered facility because the serving of the outlier would not increase the maximum weighted travelling time or costs. Moreover, the center defining nodes do not have to be connected as it is the case in unweighted problems (see Theorem 5.1). The mentioned problems in modelling with weighted nodes also occur for $p \geq 1$ new facilities.

Of course it is harder to solve weighted problems since theoretical properties of the outliers (as for example in Section 5.1) can not be applied to weighted graphs and every existing facility could be an outlier. However, the real difficulties of using node weights do not lie in the mathematical solvability of the weighted problem but in the applicability to real world problems.

In the following two sections, two possibilities to avoid the above mentioned problems in modelling outliers in weighted center problems are discussed. The first procedure is a two-stage approach based on reciprocal node weights that does not have a higher complexity than a solution approach for a k-max problem of type pkMG. The second algorithm determines so called weighted outliers by inserting multiple copies of every node in G. The complexity is higher than in the first approach but a suitable new location does not have to be computed in a separate phase. Both approaches can be applied for $p \geq 1$ new facilities.

8.1 A Two-stage Approach based on Reciprocal Node Weights

One possibility to favour high weighted nodes in a graph $G_w = (V_w, E)$ in the sense that they are less likely to become outliers and can attract the new facility is a two-stage approach based on reciprocal weight values. In a first step, the original weights w_i are replaced by their reciprocal values $\frac{1}{w_i}$, $i = 1, \ldots, n$, to determine suitable outliers. Taking the reciprocal is always possible because of $w_i > 0$ for all $i = 1, \ldots, n$. In a second step, these outliers are taken out of the optimisation process and a center problem is solved such that the outliers do not influence the location of the center point. A more detailed description of the approach is given below.

In the first phase, the graph G_w is transformed to a graph $G_{w^{-1}} = (V_{w^{-1}}, E)$, in which all node weights w_i, $i = 1, \ldots, n$, are replaced by their reciprocal values (see Figure 8.2). Since the levels of importance are reversed in $G_{w^{-1}}$, low weighted nodes in the original graph have the highest weights now. Therefore, an optimal solution $x_{w^{-1}} \in A(G_{w^{-1}})$ of the k-max problem on $G_{w^{-1}}$ favours originally low weighted nodes to become outliers because they are likely to lead to high weighted distances in $G_{w^{-1}}$. Since the transformed weights do not give the right level of importance to the nodes, an optimal location $x_{w^{-1}}$ is in general not expected to be a good location for the original problem on G_w. Note that $x_{w^{-1}}$ with $V_{k-1}^{w^{-1}}$ is in general not even a reasonable combination for the k-max problem on G since the distances $d^w(v, x_{w^{-1}})$ with $v \in V_{k-1}^{w^{-1}}$, do not have to be the $k-1$ largest distances. Hence, only the outliers $V_{k-1}^{w^{-1}}$ corresponding to $x_{w^{-1}}$ are stored, not the new location itself.

A second phase is applied to find a new location $x \in A(G_w)$ which is optimal for the already determined set of outliers $V_{k-1}^{w^{-1}}$. Therefore, let $\bar{w} : V \mapsto \mathbb{R}_+$ be a new weight function on the nodes of V that assigns a non-negative weight to all existing facilities such that

$$\bar{w}_i = 0 \quad \text{for all} \quad v_i \in V_{k-1}^{w^{-1}} \quad \text{and} \quad \bar{w}_i = w_i \quad \text{for all} \quad v_i \notin V_{k-1}^{w^{-1}}$$

and let G^0 be the graph resulting from G by substituting w by $\bar{w}$. Setting the node weights of the outliers to zero is equivalent to omitting the customers in the set V_{k-1}^{w} from the location objective without destroying the structure of the network (see Remark 4.5). As the k-max problem on the whole node set is equivalent to a center problem on the node set without the outliers (see Theorem 4.8), the new optimal location x for the set of outliers $V_{k-1}^{w^{-1}}$ is a center point of G^0. Note that only those candidates of a finite dominating set need to be considered that are defined by nodes with strictly positive weights, since other

candidates can not be an optimal solution of the problem.

x is a suitable solution for the location problem with outliers on G, since it favours low weighted nodes to be outliers and minimises the maximum distance to the covered nodes under this condition. Note that x is a feasible solution of the k-max problem on G, but in general not optimal.

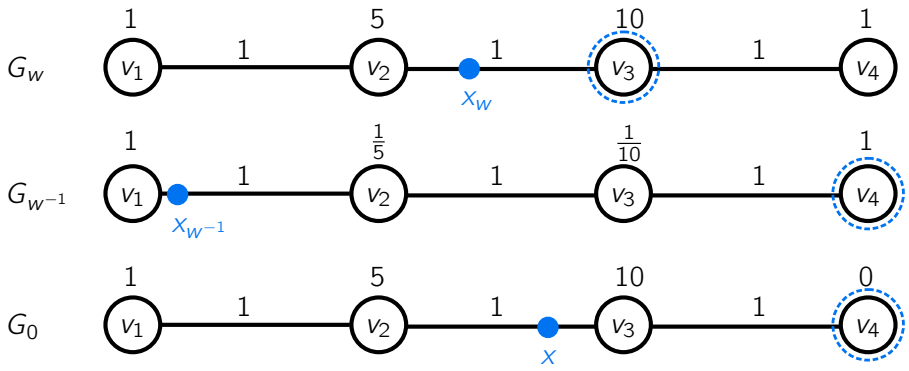


Fig. 8.2: The node with highest weight is outlier in the original graph G_w for $k = 2$. The solution of the k-max problem with $k = 2$ on the graph $G_{\frac{1}{w}}$ with reciprocal weights leads to outlier v_4. The solution x of the center problem on G_0 leads to a more satisfying result.

Example 8.2. *The optimal solution $x_w \in A(G_w)$ of the* 1-2-max *problem on the line graph G_w in Figure 8.2 illustrates the unwanted situation of having a high weighted node as an outlier: v_3 with a weight much higher than the other nodes does not influence the location of the new facility. To the contrary, the optimal solution of the* 1-2-max *problem on $G_{w^{-1}}$ returns v_4 as outlier. This is a much more satisfactory result since v_4 is one of the lowest weighted nodes in the original graph. In many applications, $x_{w^{-1}}$ is not an appropriate location for the original problem because it lies even more distant from the high weighted nodes than x_w.*

The optimal solution $x = (e_{23}, \frac{9}{11})$ of the center problem on G^0 is depicted in the third graph where $\bar{w}_4 = 0$ because v_4 is the outlier determined in Phase 1. Since the high weighted nodes v_2 and v_3 are now both center defining, the new facility x is located in their weighted midpoint and is therefore much closer to v_3 than x_w. Thus, x is a practically more satisfactory solution for the location problem with one outlier on G_w.

Note that the derived outliers may also be covered by the new facility x, i.e., they may lie within the coverage radius of the center point x. This is the case in Example 8.1 since the distance vector with respect to x is $d^w(V, x) = (\frac{20}{11}, \frac{100}{11}, \frac{20}{11}, \frac{13}{11})$ and the coverage radius equals the optimal objective function value $z^* = \frac{20}{11}$.

It can not be proven that the described two-stage approach yields better solutions (in the sense that low weighted nodes are more likely to become outliers) than the direct solution of a k-max problem. If high weighted nodes become outliers or not depends on many factors as the relation between the edge and the node weights, the differences between the node

weights and the design of the graph itself. But the two-stage approach gives an alternative procedure that can be applied if the direct k-max concept does not yield solutions suitable for the real world problem. The approach is summarised in Algorithm 15.

Algorithm 15 Two-stage approach with reciprocal node weights

Input: Graph $G_w = (V_w, E)$ with $w_i > 0 \; \forall \; i = 1, \ldots, n$, $k \in \{1, \ldots, n\}$

1: Transform G_w to $G_{w^{-1}}$ by setting $w_i := \frac{1}{w_i}$ for all $i = 1, \ldots, n$ ▷ Phase 1

2: Solve the k-max problem on $G_{w^{-1}}$ to determine $V_{k-1}^{w^{-1}}$

3: Get a center x of G^0 with $\bar{w}_i = 0 \; \forall v_i \in V_{k-1}^{w^{-1}}$, $i = 1, \ldots, n$, and else $\bar{w}_i = w_i$ ▷ Phase 2

Output: New location $x \in A(G_w)$ and set of outliers $V_{k-1}^{w^{-1}}$

The complexity of the Algorithm equals the complexity $\mathcal{O}(T)$ to solve the k-max problem on $G_{w^{-1}}$. The other two steps do not have impact on the overall complexity. The transformation of the graph needs at most $\mathcal{O}(n)$ and the solution of a center problem with n nodes in the last step can be done in $\mathcal{O}(mn\log(n))$ by applying the algorithm of Kariv and Hakimi (1979). Thus, using the finite dominating set consisting of the equilibrium points of G, the overall complexity of Algorithm 15 is $\mathcal{O}(m(n+s)\log(n))$ with $s \leq 2\binom{n}{2}$. Therefore, the advantage of this approach is that the complexity does not increase with respect to a k-max problem of type (1kMG).

Note that the described unwanted effects in modelling with weighted nodes also exist for problems where $p \geq 2$ new facilities have to be located because the effect of high node weights on the weighted distances is independent of the number of new facilities. Thus, the solution of a p-k-max problem on the transformed graph yields a set of outliers that is likely to consist of originally high weighted nodes. The solution of a p-center problem gives a suitable solution for the location problem on G as Theorem 4.8 also holds for the relation between p-center and p-k-max problems. Therefore, the introduced two-stage approach can also be applied for p new locations by replacing the *1*-k-max problem in Phase 1 by a p-k-max problem on $G_{w^{-1}}$ and the center problem in Phase 2 by a p-center problem. The complexity in this case equals the complexity to solve a p-k-max problem on a weighted graph. Moreover, the two-stage approach can analogously be applied to p-k-max location problems in the plane as defined in Section 2.2.

8.2 Weighted Outliers by Multiple Copies of Nodes

Another possibility to favour high weighted nodes to be covered is to specify the value of k not with respect to the number of acceptable outlier locations, but rather as a bound on the allowed amount of total outlier weight. Then either a small number of nodes with high node weights or many nodes with low weights can be outliers in the resulting solution. High weighted nodes can thus not be easily excluded as outliers as a node $v_i \in V$ with a high

weight w_i can only be an outlier if $k \geq w_i + 1$. To account for this, k is not longer chosen between 1 and n, but rather between 1 and the total sum of all demands $\beta = \sum_{i=1}^{n} w_i$.

This approach can be interpreted intuitively by viewing w_i as the number of customers living in the corresponding local area represented by the node v_i. Hence, a high node weight corresponds to many potential customers, and the decision to neglect this node as an outlier should be related to the number of customers of this node. As shown in Example 8.1, solving a k-max problem on G, where k determines the exact number of outliers, does in general not yield the wanted result.

To find a solution that is suitable for the location problem with outliers on $G = (V, E)$, the k-max problem is solved on a transformed graph instead of the original graph. It is assumed that all demands are integer in the following, i.e., $w_i \in \mathbb{N} \setminus \{0\}$, $i = 1, \ldots, n$. The transformation of the graph G is based on replacing all vertices v_i, $i = 1, \ldots, n$, by w_i copies $v_i^1, \ldots, v_i^{w_i}$, all having weight w_i. The transformed graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ is defined as follows:

$$\widetilde{V} = \left\{v_1^1, \ldots, v_1^{w_1}, v_2^1, \ldots, v_2^{w_2}, \ldots, v_n^1, \ldots, v_n^{w_n}\right\} \quad \text{with} \quad |\widetilde{V}| = \beta$$
$$\widetilde{E} = \left\{(v_i^1, v_j^1) : (v_i, v_j) \in E\right\} \cup \left\{(v_i^1, v_i^s) : s \in \{2, \ldots, w_i\}\right\} \quad \text{with} \quad |\widetilde{E}| = m + \beta - n$$
$$\tilde{w}_i^s = w_i \quad \forall\ i = 1, \ldots, n,\ s = 1, \ldots, w_i$$
$$l_{v_i^1, v_j^1} = l_{v_i, v_j} \quad \forall\ v_i \neq v_j \quad \text{with} \quad (v_i^1, v_j^1) \in \widetilde{E}$$
$$l_{v_i^1, v_i^s} = 0 \quad \forall\ i = 1, \ldots, n,\ s = 2, \ldots, w_i.$$

The set of nodes is $\widetilde{V}$ where $\{v_i^1, \ldots, v_i^{w_i}\} \subseteq \widetilde{V}$ denote the w_i copies of $v_i \in V$. Two nodes v_i^1 and v_j^1 are connected by an edge if v_i and v_j in G share an edge. Every vertex v_i^s, $s = 2, \ldots, w_i$, is linked to v_i^1 by an edge with length $l_{v_i^1, v_i^s} = 0$, such that $\widetilde{G}$ has $m + \beta - n$ edges and is connected. The weights of all nodes are set to $w_i^s = w_i$ for all $s = 1, \ldots, w_i$ and $i = 1, \ldots, n$.

Remark 8.3. *Note that for an edge e_{ij} with $l_{ij} = 0$, every point $x = (e_{ij}, t)$ satisfies $t = 0$. Thus, since $l_{v_i^1, v_i^s} = 0$ for all $i = 1, \ldots, n$ and $s = 2, \ldots, w_i$, every point $\tilde{x} \in (v_i^1, v_i^s) \in \tilde{E}$ can be equally shifted into $v_i^1 \in \tilde{V}$, i.e., $v_i^1 = ((v_i^1, v_i^s), 0)$ for all $s = 2, \ldots, w_i$.*

By the definition of the transformation between G and $\widetilde{G}$, there exists a point

$$x' = ((v_a^1, v_b^1), \lambda) \in A(\widetilde{G}) \quad \textit{with} \quad (v_a^1, v_b^1) \in \widetilde{E}$$

corresponding to

$$x = ((v_a, v_b), \lambda) \in A(G) \quad \textit{with} \quad (v_a, v_b) \in E$$

and $\lambda \in [0, 1]$ such that

$$d^w(v_r, x) = d^{\tilde{w}}(v_r^1, x') \quad \textit{for all} \quad r = 1, \ldots, n. \tag{8.1}$$

As each node $v_r^s \in \widetilde{V}$ with $s \in \{2, \ldots, w_r\}$ is connected to $v_r^1 \in \widetilde{V}$ by an edge of length

zero, it holds that

$$d^w(v_r, x) = d^{\tilde{w}}(v_r^1, x') + l_{v_r^1, v_r^s} = d^{\tilde{w}}(v_r^s, x') \quad \textit{for all} \quad s = 1, \dots, w_r \textit{ and } r = 1, \dots, n, \tag{8.2}$$

i.e., the weighted distance between a point x in G and a node $v_i \in V$ equals the weighted distance between the corresponding point $\widetilde{x}$ in $\widetilde{G}$ and an arbitrary copy of v_i.

The two graphs have a strong connection, described by the following lemma, which also gives a reason to assign the original weight to every copy of v_i.

Lemma 8.4 ▶ Center Point of G and $\widetilde{G}$

For $v_i, v_j \in V$ and $v_i^1, v_j^1 \in \widetilde{V}$ let $\bar{x} = ((v_i, v_j), t) \in A(G)$ and $\tilde{x} = ((v_i^1, v_j^1), t) \in A(\widetilde{G})$ for $t \in [0, 1]$. Then it holds: $\bar{x}$ is a center point of G if and only if $\tilde{x}$ is a center point of $\widetilde{G}$.

Proof. Let $\tilde{x} = ((v_a^1, v_b^1), \lambda) \in A(\widetilde{G})$ with $(v_a^1, v_b^1) \in \widetilde{E}$ and $\bar{x} = ((v_a, v_b), \lambda) \in A(G)$ with $(v_a, v_b) \in E$ be the corresponding point in G with $\lambda \in [0, 1]$. Moreover, let $\bar{z}$ be the optimal center objective function value in G and let $\tilde{z}$ be the optimal center objective function value in $\widetilde{G}$. With (8.1) and (8.2) it follows directly that $\bar{z} = \tilde{z}$.

Assume that $\bar{x}$ is the center point of G, i.e., it realises an optimal objective function value of $\bar{z}$. As $d^w(v_r, \bar{x}) \leq \bar{z}$ for all $v_r \in V$, it holds due to (8.2) that $d^{\tilde{w}}(v_r^s, \tilde{x}) \leq \bar{z}$. Since $\tilde{z} = \bar{z}$, $\tilde{x}$ must be the center point in $\widetilde{G}$.

Assume now that $\tilde{x}$ is the center point of $\widetilde{G}$, i.e., it realises an optimal objective function value of $\tilde{z}$. As $d^w(v_r, \tilde{x}) \leq \tilde{z}$ for all $v_r^s \in \widetilde{V}$, it holds due to (8.2) that $d^w(v_r, \bar{x}) \leq \tilde{z}$. Since $\tilde{z} = \bar{z}$, $\bar{x}$ must be the center point in G. ∎

Remark 8.5. *Note that the multiplicity w_i of copies of a node $v_i \in V$ does not change the center location $\bar{x}$. Hence, Lemma 8.4 can also be applied if there are $\alpha \in \{1, \dots, w_i\}$ copies of v_i in $\widetilde{G}$.*

As a next step, a k-max problem with $k \in \{1, \dots, \beta\}$ is solved on $\widetilde{G}$. Note that $l_{v_i^1, v_i^s} = 0$ does not effect the solvability of the underlying problem since all finite dominating sets described in this work can handle edge lengths of zero. Note that edges with length zero are not considered to build up a subdivision for the local analysis (see Algorithm 13) as new facilities will never be located on such an edge.

Example 8.6. *Figure 8.3 illustrates the transformation of G to $\widetilde{G}$ for a path graph with $n = 5$ nodes. Vertex v_2 has weight $w_2 = 2$ and therefore has to be doubled such that $\widetilde{V}$ contains two nodes v_2^1 and v_2^2. Both get the original weight $w_2^1 = w_2^2 = 2$. Vertex v_5 with $w_5 = 3$ is tripled analogously. All other nodes have weight* 1.

*The optimal solutions $\bar{x}^k$ of the k-*max *problem with $k = 1, 2, 3$ in G are shown in the first subfigure. In all these solutions high weighted nodes are outliers. The second subfigure shows the optimal solutions $\tilde{x}^k$ of the k-*max *problem in $\widetilde{G}$. $k = 2$, for example, yields the solution*

$$\tilde{x}^2 = \left(e_{34}, \frac{4}{5}\right) \quad \textit{with} \quad d^w(V, \tilde{x}^2) = \left(\frac{14}{5}, \frac{18}{5}, \frac{18}{5}, \frac{4}{5}, \frac{1}{5}, \frac{18}{5}, \frac{18}{5}, \frac{18}{5}\right).$$

Thus, the optimal objective function value is $\tilde{z}^2 = \frac{18}{5}$ and the corresponding outlier is either one of the copies of v_5 or a copy of v_2 (marked by the black circles).

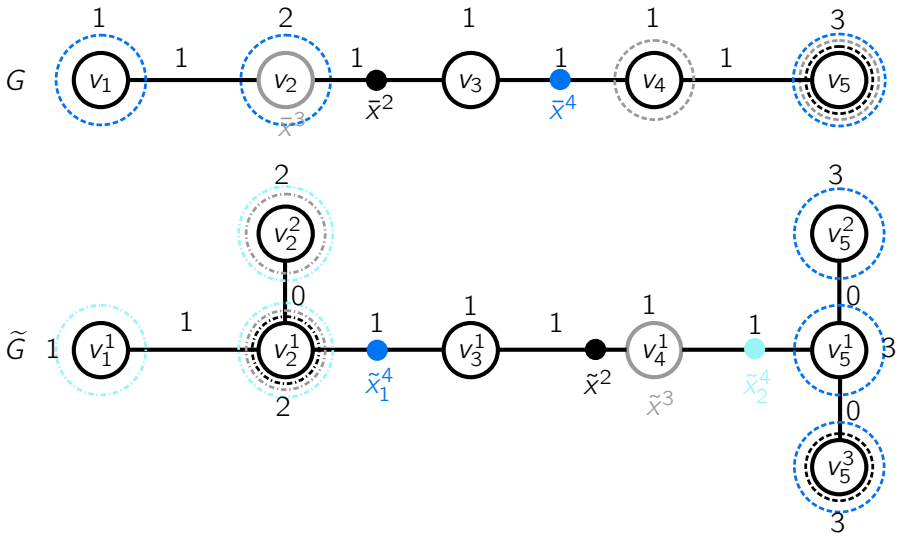


Fig. 8.3: The highest weighted node v_3 is an outlier in all optimal solutions $\bar{x}^k$ of the k-max problem with $k = 1, 2, 3$ in G (see first subfigure). The optimal solutions $\tilde{x}^k$ of the corresponding k-max problems in the transformed graph $\widetilde{G}$ are shown in the second subfigure.

The aim now is to transform the optimal solution of the k-max problem on $\widetilde{G}$ and its outliers back to G. In a first step, the outliers are transformed back to G, i.e., based on $\widetilde{V}_{k-1}$, nodes in G will be identified that should not influence the location of the new facility.

Definition 8.7 ▶ Weighted Outlier

A node $v_i \in V$ of G is called *weighted outlier* if all of its copies $v_i^1, \ldots, v_i^{w_i}$ in $\widetilde{G}$ are outliers in the solution of the k-max problem on $\widetilde{G}$. The corresponding set of weighted outliers in G is denoted by V_{k-1}^w.

Note that these weighted outliers do in general not correspond to outliers in the sense of an associated $\tilde{k}$-max problem on G. i.e., the weighted outliers do not necessarily lead to the $\tilde{k}-1$ largest elements of the vector of weighted distances between the existing facilities and their corresponding new facilities in G. Let $\tilde{x}$ be an optimal solution of the k-max problem on $\widetilde{G}$ with corresponding set of outliers $\widetilde{V}_{k-1}$ which is w.l.o.g. of the form

$$\widetilde{V}_{k-1} = \left\{ v_{i_1}^1, \ldots, v_{i_1}^{s_1}, \ldots, v_{i_q}^1, \ldots, v_{i_q}^{s_q} \right\}$$
$$\text{where} \quad v_{i_j}^s \in \widetilde{V} \quad \text{for all} \quad i_j \in \{i_1, \ldots, i_q\},\ s \in \{1, \ldots, s_j\}$$
$$\text{with} \quad |\widetilde{V}| = k - 1 = \sum_{j=1}^{q} s_j.$$

Due to (8.1) and (8.2), the weighted distances $d^{\tilde{w}}(v_i^s, \tilde{x})$ from the new facility $\tilde{x}$ to the artificial nodes equal the weighted distance $d^w(v_i, \tilde{x})$ to the original node $v_i \in V$. Let $v_{i_a}^b$ be an arbitrary but fixed outlier in $\widetilde{V}_{k-1}$ with $a \in \{1, \ldots, q\}$ and $b \in \{1, \ldots, w_{i_a}\}$. Two cases have to be considered for the transformation of $\widetilde{V}$ to the original graph G:

$$\text{Case 1}: \quad v_{i_a}^c \in \widetilde{V}_{k-1} \quad \text{for all} \quad c \in \{1, \ldots, w_{i_a}\} \setminus \{b\} \qquad \Rightarrow \quad v_{i_a} \in V_{k-1}^w, \tag{8.3}$$

$$\text{Case 2}: \quad \exists\, v_{i_a}^c \notin \widetilde{V}_{k-1} \quad \text{with} \quad c \in \{1, \ldots, w_{i_a}\} \setminus \{b\} \qquad \Rightarrow \quad v_{i_a} \notin V_{k-1}^w. \tag{8.4}$$

Definition 8.8 ▶ Fully neglected node

The nodes in V that satisfy the property (8.3) are called *fully neglected nodes*.

Fully neglected nodes correspond to a weighted outlier in G. It holds that $|V_{k-1}^w| \leq k-1$ since there could be $k-1$ weighted outliers with weight 1 each. i.e.,

$$\widetilde{V}_{k-1} = \left\{ v_{i_1}^1, \ldots, v_{i_{k-1}}^1 \right\} \quad \text{with} \quad \tilde{w}_{i_1} = \ldots = \tilde{w}_{i_{k-1}} = 1$$

or one outlier with weight $k-1$, i.e.,

$$\widetilde{V}_{k-1} = \left\{ v_{i_j}^1, \ldots, v_{i_j}^{k-1} \right\} \quad \text{with} \quad \tilde{w}_i = k-1 \quad \text{and} \quad i_j \in \{i_1, \ldots, i_q\},$$

as well as every combination in between. As a consequence, $k-1$ equals the maximum number of weighted outliers here while this is the exact number of outliers in a k-max problem. In a real world example that could mean that all customers in a district represented by $v_i \in V_{k-1}^w$ are labelled as outliers.

Definition 8.9 ▶ Partially neglected node

The nodes in V that satisfy the property (8.4) are called *partially neglected nodes*.

Partially neglected nodes do not correspond to a weighted outlier in G. Note in particular that, if at least one of the copies of $v_{i_a} \in V$ is not an outlier in $\widetilde{V}$ for the solution $\tilde{x}$ in $\widetilde{G}$, then the node v_{i_a} is not a weighted outlier in G. In a real world problem this corresponds to the situation of neglecting just a subset of the customers in a district. This is of course not useful because once the supplier paid the costs or time to get into this district, the costs of also supplying the other facilities will not increase the overall costs w.r.t. the k-max objective. Thus, an entire district should either be provided with service or be neglected completely. As a consequence, it is not useful to choose v_{i_a} as an outlier for the current value of k if not all nodes in the set $\{v_{i_s}^1, \ldots, v_{i_s}^{w_{i_s}}\}$ are selected as outliers in $\tilde{G}$.

Now, a new location in G has to be found which is optimal for the already determined set of weighted outliers V_{k-1}^w, i.e., a location that is not influenced by the weighted outliers. Different from the two-stage approach discussed in the previous section, there is no need for a second phase here because the new locations are determined by the solution of the k-max problem in $\widetilde{G}$.

Let $\bar{w} : V \to \mathbb{R}_+$ be a new weight function on the nodes of V that assigns a non-negative

weight to all existing facilities such that

$$\bar{w}_i = 0 \quad \text{for all} \quad v_i \in V_{k-1}^w \quad \text{and} \quad \bar{w}_i = w_i \quad \text{for all} \quad v_i \notin V_{k-1}^w$$

and let G^0 be the graph resulting from G by substituting w by $\bar{w}$. Setting the node weights of the outliers to zero is equivalent to omitting the customers in the set V_{k-1}^w from the location objective without destroying the structure of the network (see Remark 4.5).

Lemma 8.10 ▶ Optimal solution in G

Let V_{k-1}^w be the set of weighted outliers w.r.t. an optimal solution $\tilde{x} = ((v_i^1, v_j^1), t) \in A(\widetilde{G})$ of the k-max problem on $\widetilde{G}$. Then, the corresponding point $\bar{x} = ((v_i, v_j), t) \in A(G)$ is optimal for the center problem on G^0.

Proof. Let $\tilde{x}$ be the optimal solution of the k-max problem on $\widetilde{G}$ with set of outliers $\widetilde{V}_{k-1}$. Moreover, let $V_{k-1}^w = \{v_{i_1}, \dots, v_{i_q}\}$ be the set of weighted outliers in G. Define $\widetilde{G}^0$ with $\bar{\bar{w}}$ analogously to G^0, i.e., the node weights of all outliers in $\widetilde{V}_{k-1}$ are set to zero. As every weighted outlier in G is fully neglected, it holds that the nodes

$$v_{i_1}^1, \dots, v_{i_1}^{w_{i_1}}, \dots, v_{i_q}^1, \dots, v_{i_q}^{w_{i_q}} \in \widetilde{V}$$

are weighted by zero and therefore omitted from a location objective on $\widetilde{G}^0$. A partially neglected node $v_j \in V$ has at least one copy $v_j^s \in \widetilde{V}$, $s \in \{1, \dots, w_j\}$ that is not an outlier and is therefore not weighted by zero such that the customers v_j^s are not omitted in $\widetilde{G}^0$. Following Theorem 4.8, the solution of a k-max problem on $\widetilde{G}$ is then equivalent to the solution of a center problem $\widetilde{G}^0$, and $\tilde{x}$ is an optimal center point of $\widetilde{G}^0$. With Remark 8.5, Lemma 8.4 can be applied to G^0 and $\widetilde{G}^0$ such that $\bar{x} = ((v_i, v_j), t) \in A(G)$ is an optimal center location for G^0. ∎

Thus, the important point is that G and $\widetilde{G}$ still have the same center point if a node in G and all of its copies in $\widetilde{G}$ are neglected for the center problem.

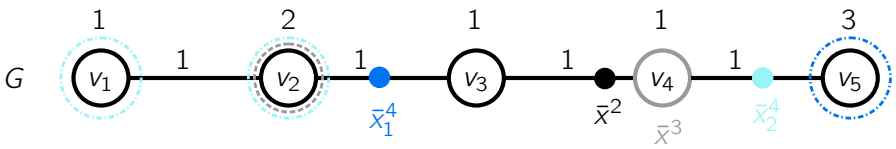


Fig. 8.4: Solutions $\bar{x}^k = \tilde{x}^k$, $k = 1, 2, 3$, of the location problem in G with sets of weighted outliers V_{k-1}^w

Example 8.11 (Continuation of Example 8.6)**.** *Figure 8.4 illustrates the transformation of the set of outliers $\widetilde{V}_{k-1}$ to the set of weighted outliers V_{k-1}^w.*

Since copy v_5^3 of $v_5 \in G$ is an outlier in $\widetilde{G}$ for $k = 2$, v_5^3 is partially neglected and thus G has no weighted outliers for $k = 2$. In this case, the decision maker should reconsider his

choice of the parameter k since $\bar{x}_2$ is the center point of G that is also optimal for $k = 1$ (see Remark 8.14). For $k = 3$, both copies of v_2 are outliers in $\widetilde{G}$. Thus, v_2 is fully neglected and hence a weighted outlier in G. For $k = 4$ there exist two different optimal solutions with different numbers of weighted outliers. $\tilde{x}_1^4$ corresponds to $\widetilde{V}_{k-1} = \{v_5^1, v_5^2, v_5^3\}$. In this case, v_5 is fully neglected and hence the only weighted outlier. $\tilde{x}_2^4$ corresponds to the two weighted outliers v_1 and v_2 since all of their copies are outliers in $\widetilde{G}$. In all cases the optimal new location $\bar{x}^k$ on G equals the optimal location $\tilde{x}^k$ on $\widetilde{G}$.

Remark 8.12. *It is also possible to choose a new parameter $\tilde{k} := k + \beta - n + 1$ to solve the $\tilde{k}$-*max *problem on $\widetilde{G}$. The resulting outliers may be more convenient in some cases for the definition of the weighted outliers of G as it leads to fully neglected outliers more often. In Example 8.6, node v_1^1 would be the outlier for $k = 2$ (as $\tilde{k} = 6$) and thus v_1 would be a fully neglected outlier in G. As it does not change the optimal objective function value in relation to Example 8.11, this is an equally good solution.*

Example 8.11 shows that different optimal solutions of the k-max problem in $\widetilde{G}$ can lead to different numbers of partially neglected nodes. In particular, it always exists an optimal solution that has at most one partially neglected node, as the next lemma states. Therefore, the number of weighted outliers is not too much affected by the partially neglected nodes.

Lemma 8.13 ▶ Partially neglected outliers in $\widetilde{G}$

There always is an optimal solution $x \in A(G)$ of the location problem on G that has at most one partially neglected node.

Proof. Let $x \in A(G)$ with V_{k-1}^w be an optimal solution of the location problem on G that has more than one partially neglected node. W.l.o.g., let $v_i \in V$, $i \in \{1, \dots, n\}$, and $v_j \in V$, $j \in \{1, \dots, n\} \setminus \{i\}$, be two arbitrary but fixed partially neglected nodes with respect to x. Therefore, w.l.o.g. it holds for their copies $\{v_i^1, \dots, v_i^{w_i}\}$ and $\left\{v_j^1, \dots, v_j^{w_j}\right\}$ in $\widetilde{V}$ with respect to the optimal solution $\tilde{x}$ of the k-max problem on $\widetilde{G}$ that

$$\{v_i^1, \dots, v_i^{s_i}\} =: \widetilde{V}_{k-1}^i \subseteq \widetilde{V}_{k-1} \quad \text{and} \quad \left\{v_i^{s_i+1}, \dots, v_i^{w_i}\right\} \notin \widetilde{V}_{k-1}$$

and respectively

$$\left\{v_j^1, \dots, v_j^{s_j}\right\} =: \widetilde{V}_{k-1}^j \subseteq \widetilde{V}_{k-1} \quad \text{and} \quad \left\{v_j^{s_j+1}, \dots, v_j^{w_j}\right\} \notin \widetilde{V}_{k-1}.$$

Then, all copies of all partially neglected nodes have the same distance to the new facility: Assume to the contrary that w.l.o.g. $d^{\tilde{w}}(v_i^a, \tilde{x}) < d^{\tilde{w}}(v_j^b, \tilde{x})$ for some $a \in \{1, \dots, w_i\}$ and $b \in \{1, \dots, w_j\}$. As it holds that

$$d^{\tilde{w}}(v_i^s, \tilde{x}) = d^{\tilde{w}}(v_i^a, \tilde{x}) \quad \text{for all} \quad s = 1, \dots, w_i$$

and

$$d^{\tilde{w}}(v_j^r, \tilde{x}) = d^{\tilde{w}}(v_j^b, \tilde{x}) \quad \text{for all} \quad r = 1, \dots, w_j,$$

it follows that $d^{\tilde{w}}(v_i^s, \tilde{x}) < d^{\tilde{w}}(v_j^r, \tilde{x})$ for all r, s. This contradicts the property of v_j being partially neglected and thus, it holds that

$$d^{\tilde{w}}(v_i^s, \tilde{x}) = d^{\tilde{w}}(v_j^r, \tilde{x}) \quad \text{for all} \quad s \in \{1, \dots, w_i\},\ r \in \{1, \dots, w_j\}.$$

Thus, $\min\{s_i, w_j - s_j\}$ elements of $\widetilde{V}_{k-1}^i$ can be deleted from $\widetilde{V}_{k-1}$ and be replaced by $\min\{s_i, w_j - s_j\}$ nodes of the set $\left\{v_j^{s_j+1}, \dots, v_j^{w_j}\right\}$ without changing the optimal objective function value of the k-max problem.

If $\min\{s_i, w_j - s_j\} = s_i$, all s_i outlier nodes $\left\{v_i^1, \dots, v_i^{s_i}\right\}$ can be deleted such that

$$\widetilde{V}_{k-1}^i = \emptyset \quad \text{and} \quad \widetilde{V}_{k-1}^j = \left\{v_j^1, \dots, v_j^{s_j}, v_j^{s_j+1}, v_j^{s_j+s_i}\right\} \text{ with } s_j + s_i \leq w_j,$$

afterwards. In this case, $v_i \in V$ is not partially neglected any more since none of its copies is an outlier in $\widetilde{G}$. Note that $v_j \in V$ may still be partially neglected.

If $\min\{s_i, w_j - s_j\} = w_j - s_j$, not all nodes in $\widetilde{V}_{k-1}^i$ can be deleted, such that $v_i \in V$ stays partially neglected. But, in this case, $w_j - s_j$ from $\widetilde{V}_{k-1}^i$ can be replaced by all nodes in $\left\{v_j^{s_j+1}, \dots, v_j^{w_j}\right\} \notin \widetilde{V}_{k-1}$ such that

$$\widetilde{V}_{k-1}^i = \left\{v_i^1, \dots, v_i^{s_i-(w_j-s_j)}\right\} \quad \text{and} \quad \widetilde{V}_{k-1}^j = \left\{v_j^1, \dots, v_j^{w_j}\right\}.$$

Hence, all copies of v_j are now outliers in $\widetilde{G}$ such that v_j is now fully neglected and therefore an outlier in G. Again, this operation does not change the objective function value.

In the two cases, one of the partially neglected outliers becomes either fully neglected or not neglected at all. By iterating this procedure, the number of partially neglected nodes can be reduced to 1. Thus, a new set of outliers with just one partially neglected node is obtained without changing the optimal solution x and the optimal objective function value of the location problem on G. ∎

Summarising the discussion above, nodes with small weights are favoured to become outliers in this approach. Moreover, the solutions are more flexible since the size of the set of outliers is not fixed to $k-1$ as it is the case for classical k-max problems. Note that choosing the parameter k from the interval $\{1, ..., \beta\}$ allows for weighted outlier sets with cardinalities from 0 to $n-1$. In particular, a high weighted node with $w_i > k-1$ can never be a weighted outlier since not all copies of v_i can be outliers in the transformed graph.

Remark 8.14. *High weighted nodes can become outliers only for an equally high value of k. From a bi-criteria perspective, the two conflicting objective functions (minimising the optimal k-*max *objective function value and minimising the number of outliers, see Section 2.3), imply that there is a trade-off between neglecting a district with many customers and a high value of k. Moreover, optimal solutions of the k-*max *problem on $\widetilde{G}$ that lead to partially neglected nodes in G are weakly efficient solutions of the bi-criteria problem (BP) on $\widetilde{G}$. In this case, the value of k can either be decreased until the partially neglected node in G is not neglected at all without changing the optimal objective function value. Alternatively, the value of k can*

be increased until the partially neglected node in G is fully neglected such that the objective function value can be reduced.

Algorithm 16 Determining weighted outliers and the corresponding center point

Input: Graph $G = (V, E)$ with $w_i \in \mathbb{N} \setminus \{0\}\ \forall\ i = 1, \dots, n$, and $k \in \{1, \dots, \beta - 1\}$.

1: Compute a finite dominating set FDS of G.
2: Construct $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ by introducing copies $\left\{v_i^1 \dots, v_i^{w_i}\right\}$ of every node $v_i \in V$.
3: Determine $\tilde{x} = \arg\min_{x \in \text{FDS}} k\text{-max}(d^{\tilde{w}}(\widetilde{V}, x))$ with set of outliers $\widetilde{V}_{k-1}$.
4: **for all** $v_i \in V$ **do**
5: **if** $v_i^s \in \widetilde{V}_{k-1}$ for all $s = 1, \dots, w_i$ **then** ▷ All copies are outliers
6: $v_i \in V_{k-1}^w$ ▷ v_i is weighted outlier

Output: Location $\tilde{x} \in A(G)$ and set of weighted outliers V_{k-1}^w with $z^* = k\text{-max}(d^{\tilde{w}}(\widetilde{V}, \tilde{x}))$.

The described approach is summarised in Algorithm 16. For the solution of the k-max problem on $\widetilde{G}$ one of the finite dominating sets from Chapter 4 can be applied. By choosing the set of equilibrium points as finite dominating set, it is sufficient to compute the equilibrium points of G to solve the k-max problem on $\widetilde{G}$, even though $\widetilde{G}$ has more nodes and edges than G. Since the distances $d^{\tilde{w}}(v_i^s, x)$, $s = 1, \dots, w_i$, of a point $x \in A(\widetilde{G})$ to a copy of $v_i \in V$ all equal the original distance $d^w(v_i, \tilde{x})$, the added nodes do not lead to new equilibrium points. Moreover, due to the edge lengths $l_{v_i^1, v_i^s} = 0$ for $i \in \{1, \dots, n\}$, $s \in \{1, \dots, w_i\}$ of an edge $(v_i^1, v_i^s) \in \widetilde{E} \setminus E$ between a node in the original graph and one of its copies in the transformed graph, equilibrium points can not be located on this edge (v_i^1, v_i^s). Therefore, the set of equilibrium points of $\widetilde{G}$ equals the set of equilibrium points of G. Note that the finite dominating set based on h-levels can also be applied to G instead of $\widetilde{G}$ with the same arguments.

Computing the finite dominating set consisting of the equilibrium points of G takes $\mathcal{O}(m(n + s\log(n)))$ with $s \leq 2\binom{n}{2}$, see Algorithm 3. In an implementation it is not necessary to really transform the graph since all later operations can be implemented using suitably manipulated vectors of distances in which all components are copied w_i times. Thus, this step takes at most $\mathcal{O}(\beta)$ time. Evaluating the k-max function on $\widetilde{G}$ can be done in $\mathcal{O}(mn^2\log(n))$ (as shown for Algorithm 5) since the copies of nodes in G do not have to be considered for the evaluation, the corresponding distances can just be copied to obtain the vector of distances. Finding the candidate point with the minimal objective function value needs $\mathcal{O}(mn^2)$. The loops over the nodes of G and all their copies take $\mathcal{O}(n\beta)$. Thus, the overall complexity of Algorithm 16 is $\mathcal{O}(mn^2\log(n) + n\beta)$.

The complexity of this approach using weighted outliers depends, contrary to the complexity of the two-stage approach described above, on the node weights of the graph. This is a disadvantage of this approach as the node weights can be arbitrarily large (even though it does in practice not make sense to choose the weights too large as otherwise the distances

have no influence on the solution of the problem). On the other hand, the approach gives more flexibility because the number of outliers is not fixed to $k-1$.

Note that the described approach and all related results can be generalised to $p \geq 2$ new facilities by just computing the p-k-max solution on $\widetilde{G}$ instead of the k-max point for $p = 1$. Once the set of outliers $\widetilde{V}_{k-1}$ depending on p is computed, the set of weighted outliers in G can be derived in the same way as for one new facility, since the status of being fully or partially neglected is not influenced by the number of new facilities.

Moreover, the approach using weighted outliers can also be applied analogously to continuous k-max location problems as introduced in Section 2.2 since the network-structure is not needed for the stated proofs. Instead of inserting artificial nodes and edges of length 0, the transformation of the problem is realised by inserting copies of the existing facilities which have the same coordinates as the respective points of the initial problem.

The two presented approaches are not implemented and tested in the section of computational results as the two-stage approach as well as the weighted outliers are a reformulation of the initial k-max problem that can be performed as a preprocessing step. The obtained problems can then be solved using the algorithms presented in the previous sections of this thesis, which are tested exhaustively in Chapter 7.

Chapter 9

Conclusion and Outlook

In this thesis a new approach for handling outliers in center location problems was introduced. If far-away customers are present in a location problem, the optimal center solution may be influenced in an unwanted way. Therefore, the exclusion of outlier customers may often be economically useful. To implement this exclusion of outliers, the k-max objective function was introduced in the context of center location problems. This objective uses the kth largest distance between any customer and its corresponding closest new facility as the decision criterion for the location of the new facilities. In this way, the influence of outliers on the optimal center location can be significantly reduced.

The focus in this thesis was set on three topics: The discussion of advantages and disadvantages of different **modelling approaches**, and the analysis of **single facility k-max problems** on networks as well as the more complex **multi-facility p-k-max problems** on networks. For the solution of these problems, several finite dominating sets with different properties were derived and approaches for their efficient evaluation were introduced.

Modelling Approaches A bi-criteria optimisation model for handling outliers in center location problems was introduced. The ε-constraint scalarisation method was applied to generate the weakly efficient solutions of the underlying problem. This procedure is very efficient since all reasonable values for the upper bounds ε on the number of outliers are known in advance. Each ε-constraint problem can be transformed into a single criteria k-max problem and thus all weakly non-dominated points of the bi-criteria problem can be obtained by solving a finite series of single-criteria k-max problems. Moreover, it was shown that the efficient solutions can be filtered out easily. Using the finite dominating sets derived in this thesis, all efficient points of the bi-criteria model can be computed in polynomial time for the single-facility case.

Furthermore, it was illustrated that the assignment of different positive weights to the customers may lead to unwanted effects on the optimal solutions of the problem: High weighted clients are more likely to become outliers than lower weighted clients. Two approaches to reduce this effect were introduced. A two-stage approach replaces the original node weights by their reciprocal values. Solving the p-k-max problem on this graph favours low weighted nodes to be outliers. A second phase is applied to find the center location with respect to the set of outliers found in the first phase. This approach yields a reasonable solution for the initial problem in polynomial time. Another approach extends the original graph by inserting w_i copies of each node v_i to the graph. It was shown that the optimal solution on the transformed graph also yields a suitable solution for the original graph, with an adapted definition of nodes being weighted outliers. The complexity of this approach

depends on the node weights of the graph but it also leads to much more flexible solutions since the number of outliers is not fixed to $k-1$.

Three (mixed-) integer programming formulations for the single-criteria p-k-max problem were presented. The first integer linear formulation is composed of a sorting problem and a location-allocation problem. In the second formulation, the number of variables and constraints was reduced further by identifying the outliers of the current candidate solution and minimising the largest weighted distance to the remaining facilities. The third model is a mixed integer formulation which does not assume that the candidate locations are known in advance.

1-k-max Problems The focus in designing solution methods for the k-max problem was set on identifying finite dominating sets for the problem. The first finite dominating set is twofold. For $k \leq n-1$ it consists of the equilibrium points of the underlying graph. It was shown that the equilibrium points define a subdivision of G such that the objective function is piecewise linear and concave in every cell of this subdivision. In the case of $k = n$ the complete set V of nodes is optimal and the problem can be trivially solved with objective value 0. The (already polynomial) complexity of the evaluation of this finite dominating set is further reduced by applying the algorithm of Bentley and Ottmann (1979) for the computation of the equilibrium points. Exhaustive computational results confirmed that the finite dominating set is in practice much smaller than the theoretical upper bound suggests. The tests showed that large problem instances of up to 200 customers can be solved in satisfactory time and that a parallelised implementation of the algorithm allows even larger instances.

The second candidate set consists of the local minima of (n-k)-levels. Each (n-k)-level is constructed as the kth highest chain of line segments and vertices in the arrangement of line segments given by the graphs of the weighted distances functions over an edge of the underlying graph. The local minima of the (n-k)-levels thus correspond to selected equilibrium points or nodes of G. The number of candidate points in this set is much smaller than the number of candidates in the first finite dominating set since only a small subset of all equilibrium points has to be considered.

Moreover, finite dominating sets and corresponding solution approaches for two special cases of unweighted (path) graphs were deduced.

p-k-max Problems Different finite dominating sets and efficient algorithms to evaluate these sets were deduced for p-k-max problems with $p \geq 2$. A first finite dominating set is given by all p-tuples consisting of all combinations of the equilibrium points and the nodes of the underlying graph. Even though the size of this set is in practice much smaller than the theoretical upper bound suggests, it grows exponentially in the number of new facilities. In order to reduce the number of candidates that have to be evaluated, a recursive approach was introduced. It was proven that an objective value defining facility always lies in an equilibrium point. The individual new facilities of each solution are located iteratively. Hence, for each new facility it can be checked which locations may potentially lead to an optimal solution based on whether this location contradicts the optimality of the objective value

defined by the previously located facilities. The computational tests showed that in general only a small subset of the candidates in the finite dominating set have to be considered. Larger problem instances of up to 50 customers can be solved within a time bound of five hours with the recursive approach.

In order to find alternative optimal solutions not being an element of the initial finite dominating set, a local analysis on each edge of the graph was performed. By deriving an optimality condition, an arrangement of hyperplanes based on equilibrium planes and balance planes was obtained which defines a subdivision of the unit hypercube. The k-max function was shown to be piecewise linear and concave over each cell of this subdivision. Thus, the 0-faces of the arrangement of hyperplanes are a finite dominating set for the p-k-max problem. This candidate set was further reduced using the information given by the set of optimal objective value defining facilities provided by the recursive approach. As a consequence, only the 0-faces lying in a specific hyperplane of the subdivisions were needed to constitute a finite dominating set and the number of candidates for an optimal solution was reduced enormously. For a fixed value of p, the finite dominating set is of polynomial size. Computational tests underlined this improvement in terms of much smaller CPU-times. Moreover, all (infinitely many) optimal solutions of the p-k-max problem were obtained by constructing the convex hull of every set of optimal 0-faces that belong to the same cell of a subdivision.

As an alternative approach, the individual facilities of an optimal solution of the first candidate set can be shifted within a certain interval along the edges of the graph without destroying the optimality of the solution. Thus each new facility of an optimal solution can be analysed separately and the neighbourhood of particularly interesting locations can be analysed for further optimal locations.

Topics for future research The results of this thesis are mainly dedicated to location problems on networks. An interesting field for future research is therefore the question if and how the here presented approaches can be adapted to other location problems, and in particular to continuous p-k-max problems. Some procedures as Algorithm 15 and Algorithm 16 as well as the relation to p-center problems (see Theorem 4.8) can equivalently be applied to p-k-max problems in the plane.

A p-k-median problem as a generalisation of a p-median problem could be defined as the problem to minimise the sum of the $n - k + 1$ smallest distances between the customers and the new facilities. This problem is already known as the anti-h-centrum problem with $h = n - k + 1$, see Nickel and Puerto (2005). If the outliers of the problem are known, it should also hold that the p-k-median problem reduces to a p-median problem on the remaining facilities which are covered with service.

The identification of properties of outliers in p-k-max problems in the plane (similar to these derived in Section 5.1 on networks) would be helpful. The property of the connectedness of the center defining nodes in unweighted k-max problems on graphs should be replaceable by the statement that outliers of an unweighted k-max problem in the plane have to lie outside the convex hull of the center defining nodes.

Especially the relation to the p-center problem can be used to derive further solution

approaches (for both problems in the plane and on networks) based on the known approaches for p-center problems. With this approach it should also be possible to include a number of environmental factors of practical problems into the model, for example, forbidden regions or barriers to travel. In the case of barriers or forbidden regions it could be analysed how the presented finite dominating sets have to be adjusted, i.e., which candidates can no longer be optimal and which new points (for example on the border of a forbidden region) have to be added.

p-k-max problems have in general (infinitely) many optimal solutions such that some of these obtained locations are more sensible in a practical setting than others. Thus, it could be useful to analyse further properties of optimal solutions which lead to (in practice) particularly relevant locations. These properties can of course also be chosen with regard to naturally arising restrictions or requirements as mentioned above.

It could be promising to analyse heuristic techniques for the solution of the p-k-max problem. This would omit the need to evaluate finite dominating sets which may get very large, especially for large sets of existing customers and a large number of new facilities to locate.

A further approach could be based on the fact that the k-max function is a d.c. function composed of two convex ordered median functions $f_{\lambda^{k+1}}$ and f_{λ^k} with

$$\lambda_1^{k+1}, \ldots, \lambda_{n-k-1}^{k+1} = 0 \text{ and } \lambda_{n-k}^{k+1}, \ldots, \lambda_n^{k+1} = 1$$

resp.

$$\lambda_1^{k+1}, \ldots, \lambda_{n-k}^{k+1} = 0 \text{ and } \lambda_{n-k+1}^{k+1}, \ldots, \lambda_n^{k+1} = 1,$$

i.e., it holds that $f_k(y) = f_{\lambda^{k+1}}(y) - f_{\lambda^k}(y)$. Many solution techniques for this class of functions exist which could be adjusted for the p-k-max problem.

Bibliography

Ackermann, W. (1928). Zum Hilbertschen Aufbau der reellen Zahlen. *Mathematische Annalen*, 99(118-133).

Agarwal, P., De Berg, M., Matousek, J., and Schwarzkopf, O. (1998a). Constructing levels in arrangements and higher order Voronoi diagrams. *SIAM Journal on Computing*, 27(3):654–667.

Agarwal, P. and Phillips, J. (2008). An efficient algorithm for 2D euclidean 2-center with outliers. *16th Annual European Symposium on Algorithms (ESA)*, pages 64–75.

Agarwal, P. K., Aronov, B., and Sharir, M. (1998b). On levels in arrangements of lines, segments, planes, and triangles. *Discrete & Computational Geometry*, 19(3):315–331.

Aggarwal, C. (2013). *Outlier analysis*. Springer-Verlag New York, first edition.

Ahn, H.-K., Bae, S., Demaine, E., Demaine, M., Kim, S.-S., Korman, M., Reinbacher, I., and Son, W. (2011). Covering points by disjoint boxes with outliers. *Computational Geometry*, 44(3):178–190.

Ahuja, R., Magnanti, T., and Orlin, J. (1993). *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall.

Angiulli, F. and Pizzuti, C. (2002). Fast outlier detection in high dimensional spaces. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 15–27. Springer-Verlag London.

Atanassov, R., Bose, P., Couture, M., Maheshwari, A., Morin, P., Paquette, M., Smid, M., and Wuhrer, S. (2009). Algorithms for optimal outlier removal. *Journal of Discrete Algorithms*, 7(2):239–248.

Bentley, J. and Ottmann, T. (1979). Algorithms for reporting and counting geometric intersections. *IEEE Transportation on Computers*, 28:643–647.

Berman, O., Einav, D., and Handler, G. (1990). The constrained bottleneck problem in networks. *Operations Research*, 38(1):178–181.

Bhattacharya, B. and Shi, Q. (2014). Improved algorithms to network p-center location problems. *Computational Geometry*, 47(2):307–315.

Blanco, V., Ben-Ali, S., and Puerto, J. (2013). Minimizing ordered weighted averaging of rational functions with applications to continuous location. *Computers & Operations Research*, 40(5):1448–1460.

Blanco, V., Puerto, J., and Ben-Ali, S. (2016). Continuous multifacility ordered median location problems. *European Journal of Operational Research*, 250(1):56–64.

Boland, N., Domínguez-Marín, P., Nickel, S., and Puerto, J. (2006). Exact procedures for solving the discrete ordered median problem. *Computers & Operations Research*, 33(11):3270–3300.

Bornstein, C., Maculan, N., Pascoal, M., and Pinto, L. (2012). Multiobjective combinatorial optimization problems with a cost and several bottleneck objective functions: an algorithm with reoptimization. *Computers & Operations Research*, 39(9):1969–1976.

Branke, J., Deb, K., Dierolf, H., and Osswald, M. (2004). Finding knees in multi-objective optimization. In *Parallel Problem Solving from Nature-PPSN VIII*, pages 722–731. Springer-Verlag Berlin Heidelberg.

Breunig, M., Kriegel, H.-P., Ng, R., and Sander, J. (2000). Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM.

Burkhard, R. and Rendl, F. (1991). Lexicographic bottleneck problems. *Operations Research Letters*, 10(5):303–308.

Calik, H. and Tansel, B. (2013). Double bound method for solving the p-center location problem. *Computers & Operations Research*, 40(12):2991–2999.

Camerini, P. (1978). The min-max spanning tree problem and some extentions. *Information Processing Letters*, 7(1):10–14.

Chakrabarty, D., Goyal, P., and Krishnaswamy, R. (2016). The non-uniform k-center problem. *arXiv preprint arXiv:1605.03692*.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15.

Charikar, M., Khuller, S., Mount, D., and Narasimhan, G. (2001). Algorithms for facility location problems with outliers. *Proc. Symposium on Discrete Algorithms*, pages 642–651.

Chazelle, B. (1993). An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10(4):377–409.

Chen, D. and Chen, R. (2009). New relaxation-based algorithms for the optimal solution of the continuous and discrete p-center problems. *Computers & Operations Research*, 36(5):1646–1655.

Christofides, N. and Viola, P. (1971). The optimum location of multi-centres on a graph. *Journal of the Operational Research Society*, 22(2):145–154.

Chrystal, G. (1885). On the problem to construct the minimum circle enclosing n given points in the plane. *Proceedings of the Edinburgh Mathematical Society*, 3:30–33.

Church, R. and Revelle, C. (1974). The maximal covering location problem. *Papers of Regional Science Association*, 32(1):101–118.

Cohen, E. and Megiddo, N. (1993). Maximizing concave functions in fixed dimension. *Complexity in Numerical Optimization*, pages 74–87.

Danzer, L., Grünbaum, B., and Klee, V. (1963). Helly's theorem and its relatives.

Das, I. (1999). On characterizing the "knee" of the pareto curve based on normal-boundary intersection. *Structural Optimization*, 18:107–115.

Daskin, M. (2013). *Network and Discrete Location: Models, Algorithms, and Applications*. John Wiley & Sons, second edition.

Daskin, M. and Owen, S. (1999). Two New Location Covering Problems: The Partial Covering P-Center Problem and the Partial Set Covering Problem. *Geographical Analysis*, 31.

Dey, T. (1997). Improved bounds on planar k-sets and k-levels. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 156–161. IEEE.

Domínguez-Marín, P. (2003). *The Discrete Ordered Median Problem: Models and Solution Methods*. PhD thesis, Universität Kaiserslautern.

Domínguez-Marín, P., Nickel, S., Hansen, P., and Mladenović, N. (2005). Heuristic procedures for solving the discrete ordered median problem. *Annals of Operations Research04*, 136(1):145–173.

Drezner, Z. (1984). The p-center problem - heuristic and optimal algorithms. *The Journal of the Operational Research Society*, 35(8):741–748.

Drezner, Z. (1995). *Facility Location: A Survey of Applications and Methods*. Springer-Verlag New York.

Drezner, Z. (2013). Solving planar location problems by global optimization. *Logistics Research*, 6(1):17–23.

Drezner, Z. and Hamacher, H., editors (2002). *Facility Location: Applications and Theory*. Springer-Verlag Berlin Heidelberg.

Drezner, Z. and Nickel, S. (2009a). Constructing a DC decomposition for ordered median problems. *Journal of Global Optimization*, 45(2):187–201.

Drezner, Z. and Nickel, S. (2009b). Solving the ordered one-median problem in the plane. *European Journal of Operational Research*, 195(1):46–61.

Drezner, Z. and Suzuki, A. (2004). The big triangle small triangle method for the solution of nonconvex facility location problems. *Operations Research*, 52(1):128–135.

Drezner, Z. and Wesolowsky, G. O. (1980). Single facility l_p-distance minimax location. *SIAM Journal on Algebraic Discrete Methods*, 1(3):315–321.

Durier, R. and Michelot, C. (1985). Geometrical properties of the fermat-weber problem. *European Journal of Operational Research*, 20(3):332–343.

Durier, R. and Michelot, C. (1994). On the set of optimal points to the weber problem: further results. *Transportation Science*, 28(2):141–149.

Edelsbrunner, H. (1987). *Algorithms in Combinatorial Geometry*. Springer-Verlag Berlin Heidelberg, first edition.

Edelsbrunner, H., O'Rourke, J., and Seidel, R. (1986). Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15:341–363.

Edmonds, J. and Fulkerson, D. (1970). Bottleneck extrema. *Journal of Combinatorial Theory*, 8(3):299–306.

Ehrgott, M. (2005). *Multicriteria optimisation*. Springer-Verlag Berlin Heidelberg, second edition.

Eiselt, H. and Marianov, V. (2011). *Foundations of location analysis*, volume 155. Springer US.

Elloumi, S., Labbé, M., and Pochet, Y. (2004). A new formulation and resolution method for the p-center problem. *INFORMS Journal on Computing*, 16(1):84–94.

Elzinga, J. and Hearn, D. W. (1972). Geometrical solutions for some minimax location problems. *Transportation Science*, 6(4):379–394.

Eskin, E. (2000). Anomaly detection over noisy data using learned probability distributions. In *In Proceedings of the International Conference on Machine Learning*. Citeseer.

Espejo, I., Rodríguez-Chía, A., and Valero, C. (2009). Convex ordered median problem with lp-norms. *Computers & Operations Research*, 36(7):2250–2262.

Fernández, E., Pozo, M., and Puerto, J. (2014). Ordered weighted average combinatorial optimization: Formulations and their properties. *Discrete Applied Mathematics*, 169:97–118.

Fernández, E., Pozo, M. A., Puerto, J., and Scozzari, A. (2016). Ordered weighted average optimization in multiobjective spanning tree problem. *European Journal of Operational Research*.

Finbow, S. and MacGillivray, G. (2009). The firefighter problem: a survey of results, directions and questions. *Australasian Journal of Combinatorics*, 43(57-77):6.

Fisher, R. A. and Tippett, L. H. C. (1928). Limiting forms of the frequency distribution of the largest or smallest member of a sample. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 24, pages 180–190. Cambridge Univ Press.

Floyd, R. (1962). Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345.

Francis, R., McGinnis, L., and White, J. (1992). *Facility Layout and Location: An Analytical Approach*. Prentice Hall, second edition.

Gabow, H. and Tarjan, R. (1988). Algorithms for two bottleneck optimization problems. *Journal of Algorithms*, 9(3):411–417.

Garfinkel, R. (1971). An improved algorithm for the bottleneck assignment problem. *Operations Research*, 19(7).

Garfinkel, R., Neebe, A., and Rao, M. (1977). The m-center problem: Minimax facility location. *Management Science*, 23(10):1133–1142.

Goldman, A. (1972). Minimax location of a facility in a network. *Transportation Science*, 6(4):407–418.

Gorski, J., Klamroth, K., and Ruzika, S. (2012). Generalized Multiple Objective Bottleneck Problems. *Operations Research Letters*, 40(4):276–281.

Gorski, J. and Ruzika, S. (2009). On k-max-optimization. *Operations Research Letters*, 37(1):23–26.

Grama, A. (2003). *Introduction to parallel computing*. Pearson Education.

Gross, O. (1959). The Bottleneck Assignment Problem. *The RAND Corporation*.

Grzybowski, J., Kalcsics, J., Nickel, S., Pallaschke, D., and Urbański, R. (2015a). On max-min representations of ordered median functions. *Optimization: A Journal of Mathematical Programming and Operations Research*, 64(2):339–348.

Grzybowski, J., Kalcsics, J., Nickel, S., Pallaschke, D., and Urbański, R. (2015b). On topological types of ordered median functions. *Optimization: A Journal of Mathematical Programming and Operations Research*, 64(1):149–160.

Hakimi, S. (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research*, 12(3):450–459.

Halpern, J. (1978). Finding minimal center-median convex combination (cent-dian) of a graph. *Management Science*, 24(5):535–544.

Handl, J. and Knowles, J. (2007). An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation*, 11(1):56–76.

Hatami, B. and Zarrabi-Zadeh, H. (2017). A streaming algorithm for 2-center with outliers in high dimensions. *Computational Geometry*, 60:26–36.

Hawkins, D. (1980). *Identification of outliers*, volume 11. Springer Netherlands.

Hershberger, J. (1989). Finding the upper envelope of n line segments in $O(nlog(n))$ time. *Information Processing Letters*, 33:169–174.

Hershberger, J. (1992). Upper envelope onion peeling. *Computational Geometry: Theory and Applications*, 2(2):93–110.

Hooker, J. (1986). Solving nonlinear single-facility network location problems. *Operations Research*, 34(5):732–743.

Hooker, J. N., Garfinkel, R. S., and Chen, C. K. (1991). Finite Dominating Sets for Network Location Problems. *Operations Research*, 39(1):100–118.

Kalcsics, J. (2011). The multi-facility median problem with Pos/Neg weights on general graphs. *Computers & Operations Research*, 38(3):674–682.

Kalcsics, J. and Nickel, S. (2003). Multifacility ordered median problems on networks: A further analysis. *Networks*, 41(1):1–12.

Kalcsics, J., Nickel, S., Pozo, M., Puerto, J., and Rodríguez-Chía, A. (2014). The multi-criteria p-facility median location problem on networks. *European Journal of Operational Research*, 235(3):484–493.

Kalcsics, J., Nickel, S., Puerto, J., and Rodríguez-Chía, A. (2015). Several 2-facility location problems on networks with equity objectives. *Networks*, 65(1):1–9.

Kalcsics, J., Nickel, S., Puerto, J., and Tamir, A. (2002). Algorithmic results for ordered median problems. *Operations Research Letters*, 30(3):149–158.

Karger, D. and Stein, C. (1996). A New Approach to the Minimum Cut Problem. *Journal of the ACM (JACM)*, 43(4):601–640.

Kariv, O. and Hakimi, S. (1979). An algorithmic approach to network location problems. I: The p-centers. *SIAM Journal on Applied Mathematics*, 37(3):513–538.

Krebs, J. and Nickel, S. (2010). Extensions to the continuous ordered median problem. *Mathematical Methods of Operations Research*, 71(2):283–306.

Labbé, M., Peeters, D., and Thisse, J.-F. (1995). Location on Networks. *Handbooks in operations research and management science*, 8:551–624.

Labbé, M., Ponce, D., and Puerto, J. (2017). A comparative study of formulations and solution methods for the discrete ordered p-median problem. *Computers & Operations Research*, 78:230–242.

Laporte, G., Nickel, S., and da Gama, F. S. (2015). *Location Science*. Springer International Publishing.

Lee, D. and Schachter, B. (1980). Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer & Information Sciences*, 9(3):219–242.

Love, R., Morris, J., and Wesolowsky, G. (1988). *Facilities Location: Models & Methods*. North-Holland.

Marín, A., Nickel, S., Puerto, J., and Velten, S. (2009). A flexible model and efficient solution strategies for discrete location problems. *Discrete Applied Mathematics*, 157(5):1128–1145.

Marín, A., Nickel, S., and Velten, S. (2010). An extended covering model for flexible discrete and equity location problems. *Mathematical Methods of Operations Research*, 71(1):125–163.

Megiddo, N. and Supowit, K. (1984). On the complexity of some common geometric location problems. *SIAM journal on computing*, 13(1):182–196.

Mihelič, J. and Robič, B. (2005). Solving the k-center problem efficiently with a dominating set algorithm. *CIT. Journal of computing and information technology*, 13(3):225–234.

Minieka, E. (1970). The m-center problem. *Siam Review*, 12(1):138–139.

Mladenović, N., Labbé, M., and Hansen, P. (2003). Solving the p-center problem with tabu search and variable neighborhood search. *Networks*, 42(1):48–64.

Nickel, S. (2001). Discrete ordered Weber problems. In *Operations research proceedings*, pages 71–76. Springer Berlin Heidelberg.

Nickel, S. and Puerto, J. (1999). A unified approach to network location problems. *Networks*, 34(4):283–290.

Nickel, S. and Puerto, J. (2005). *Location Theory: A Unified Approach*. Springer-Verlag Berlin Heidelberg.

Nickel, S., Puerto, J., Rodríguez-Chía, A., and Weissler, A. (2005). Multicriteria planar ordered median problems. *Journal of Optimization Theory and Applications*, 126(3):657–683.

Ohsawa, Y., Ozaki, N., Plastria, F., and Tamura, K. (2007). Quadratic ordered median location problems. *Journal of the Operations Research Society of Japan*, 50(4):540–562.

Okabe, A., Boots, B., Sugihara, K., and Chiu, S. N. (2009). *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons.

Osborne, M. (2014). *Locally Convex Spaces*. 269. Springer International Publishing, first edition.

Ott, L., Pang, L., Ramos, F., and Chawla, S. (2014). On integrated clustering and outlier detection. *Advances in Neural Information Processing Systems*, pages 1359–1367.

Overmars, M. and Yap, C.-K. (1991). New upper bounds in klee's measure problem. *SIAM Journal on Computing*, 20(6):1034–1045.

Pach, J. and Sharir, M. (2009). *Combinatorial geometry and its algorithmic applications*. American Mathematical Society.

Pérez-Brito, D., Moreno-Pérez, J., and Rodríguez-Martín, I. (1997). Finite dominating set for the p-facility cent-dian network location problem. *Studies in Locational Analysis*, 11:27–40.

Pinto, L. and Pascoal, M. (2010). On algorithms for the tricriteria shortest path problem with two bottleneck objective functions. *Computers & Operations Research*, 37(10):1774–1779.

Puerto, J. and Fernández, F. (2000). Geometrical properties of the symmetrical single facility location problem. *Journal of Nonlinear and Convex Analysis*, 1(3):321–342.

Puerto, J., Pérez-Brito, D., and García-González, C. (2014). A modified variable neighborhood search for the discrete ordered median problem. *European Journal of Operational Research*, 234(1):61–76.

Puerto, J. and Rodriguez-Chia, A. (2005). On the exponential cardinality of the FDS for the ordered p-median problem. *Mathematics of Operations Research*, 28(4):693–715.

Pullan, W. (2008). A memetic genetic algorithm for the vertex p-center problem. *Evolutionary computation*, 16(3):417–436.

Punnen, A. (1991). A linear time algorithm for the maximum capacity path problem. *European Journal of Operational Research*, 53:402–404.

Punnen, A. and Aneja, Y. (2004). Lexicographic balanced optimization problems. *Operations Research Letters*, 32(1):27–30.

Punnen, A., Nair, K., and Aneja, Y. (1995). Generalized bottleneck problems. *Optimization: A Journal of Mathematical Programming and Operations Research*, 35(2):159–169.

Rockafellar, R. (1970). *Convex analysis*. Princeton university press.

Rodríguez-Chía, A., Nickel, S., Puerto, J., and Fernández, F. (2000). A flexible approach to location problems. *Mathematical Methods of Operations Research*, 51(1):69–89.

Rodríguez-Chía, A., Puerto, J., Pérez-Brito, D., and Moreno, J. (2005). The p-facility ordered median problem on networks. *Top*, 13(1):105–126.

Rong, A., Klamroth, K., and Figueira, J. (2013). Multicriteria 0-1 knapsack problems with k-min objectives. *Computers & Operations Research*, 40(5):1481–1496.

Rosenberger, J. and Gasko, M. (1983). Comparing location estimators: Trimmed means, medians, and trimean. *Understanding robust and exploratory data analysis*, pages 297–336.

Rousseeuw, P. (1984). Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880.

Schöbel, A. (1999). *Locating Lines and Hyperplanes - Theory and Algorithms*. Applied Optimization. Springer US, first edition.

Seidel, R. (1986). Constructing Higher-dimensional Convex Hulls at Logarithmic Cost Per Face. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 404–413, New York, NY, USA. ACM.

Shamos, M. and Hoey, D. (1975). Closest-point problems. In *Foundations of Computer Science, 1975., 16th Annual Symposium on*, pages 151–162. IEEE.

Shamos, M. I. and Hoey, D. (1976). Geometric intersection problems. *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, pages 208–215.

Slater, P. (1982). Locating central paths in a graph. *Transportation Science*, 16(1):1–18.

Smith, R., Bivens, A., Embrechts, M., Palagiri, C., and Szymanski, B. (2002). Clustering approaches for anomaly-based intrusion detection. In *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks*, pages 579–584. ASME Press.

Sokkalingam, P. and Aneja, Y. (1998). Lexicographic bottleneck combinatorial problems. *Operations Research Letters*, 23:27–33.

Stanimirović, Z., Kratica, J., and Dugošija, D. (2007). Genetic algorithms for solving the discrete ordered median problem. *European Journal of Operational Research*, 182(3):983–1001.

Stanley, R. (1996). Hyperplane arrangements, interval orders, and trees. *Proceedings of the National Academy of Sciences*, 93(6):2620–2625.

Sylvester, J. (1857). A question in the geometry of situation. *Quarterly Journal of Pure and Applied Mathematics*.

Tang, H., Cheng, T., and Ng, C. (2009). Finite dominating sets for the multi-facility ordered median problem in networks and algorithmic applications. *Computers & Industrial Engineering*, 57(3):707–712.

Tang, H., Cheng, T., and Ng, C. (2011). Multi-facility ordered median problems in directed networks. *Journal of Systems Science and Complexity*, 24(1):61–67.

Tansel, B. C., Francis, R. L., and Lowe, T. J. (1983). Location on networks: A survey. Part I: The p-center and p-median problems. Technical Report 4, Management Science.

Thompson, A. (1996). *Minkowski geometry*. Cambridge University Press.

Turner, L. (2011). Variants of the shortest path problem. *Algorithmic Operations Research*, 6(2).

Wang, F., Xu, D., and Wu, C. (2014). Combinatorial approximation algorithms for the robust facility location problem with penalties. *Journal of Global Optimizaton*.

Weber, A. (1909). *Über den Standort der Industrien*, volume 2. Verlag von J.C.B Mohr (Paul Siebeck).

Weiszfeld, E. (1937). Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386.

Xu, G. and Xu, J. (2009). An improved approximation algorithm for uncapacitated facility location problem with penalties. *Journal of Combinatorial Optimization*, 17(4):424–436.

Yager, R. (1988). On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems, Man and Cybernetics*, 18(1):183–190.

Zarrabi-Zadeh, H. and Mukhopadhyay, A. (2009). Streaming 1-center with outliers in high dimensions. *CCCG*, pages 83–86.

Acknowledgements

During the studies for my thesis, many people supported me in some way. I would like to mention some of them here.

First of all, my special thanks go to my supervisor Prof. Dr. Kathrin Klamroth for giving me the opportunity to work in this interesting field, for her support whenever needed, and for never loosing her cheerful mood.

I am grateful to Prof. Dr. Justo Puerto for being the second reviewer of this thesis. In 2014 I had the opportunity to spend several month at the Institute of Mathematics (IMUS) at the University of Seville, Spain, to work with him. Our meetings and discussions were great motivation and a big help for the development of some fundamental ideas for this thesis. I also want to thank all the (temporary and permanent) members of the IMUS for hosting me in their group and for the great time I had in Seville.

A big "thank you" goes to Dr. Michael Stiglmayr for a lot of helpful discussions that really brought me forward and the patient answering of many questions. Many thanks also to Dr. Andrea Wagner for spending her time on proofreading my thesis and her motivating words.

I spend some great years working in the Group of Optimization and Approximation at the University of Wuppertal together with my former and recent colleagues Katharina Baumann, Prof. Dr. Peter Beisel, Dr. Kerstin Dächert, Prof. Dr. Margareta Heilmann, Dr. Markus Kaiser, Dr. Renaud Lacour, Marco Milano, Dr. Britta Schulze, Dr. Martin Wagner and Kirsten Wilshaus. Thank you for the close cooperation, for many (on- and off-topic) conversations and the familiar atmosphere in all these years.

During the first three years of my PhD studies I was funded by the Foundation of German Business (sdw). I am grateful for the financial support and the broad offer of interesting seminars on different topics.

I also want to thank my friends from Wuppertal and from Wipperfürth for being by my side, some of them for much more than half of my life. I am sure you know that you are meant.

Last, but not least, I thank my dear parents Ursula and Josef W. Schnepper and my "sister-heart" Anne for their support in all fields of my life and for encouraging me to go on. And Marcel Ließ, my partner in life, thank you for your love, your support and your patience, especially during the last month.