RELIABLE METHODS FOR LOCATION AND SIMPLEX-BASED PROBLEMS

Boglárka G.-Tóth

2022

dc_1926_21

Contents

Preface **iii**

1 Reliable methods for solving special location problems **1**
1.1 Maximal covering problem on a network . . . 1
1.1.1 Structural properties . . . 3
1.1.2 A global optimization approach . . . 4
1.1.3 Computational results . . . 9
1.1.4 Conclusions . . . 11
1.2 Firm expansion: an interval B&B method for a MINLP . . . 13
1.2.1 The location and design model for firm expansion . . . 14
1.2.2 An exact interval branch-and-bound algorithm . . . 19
1.2.3 Computational studies . . . 22
1.2.4 Conclusions . . . 25
1.3 Location equilibria under delivered pricing . . . 25
1.3.1 A reformulation as an optimization location game problem . . . 27
1.3.2 A branch-and-bound based approach . . . 29
1.3.3 The heuristic-based approach . . . 30
1.3.4 Computational results . . . 34
1.3.5 Conclusions and suggestions . . . 38

2 Simplicial branch-and-bound methods **39**
2.1 Covering a simplex by spheres . . . 39
2.1.1 Distances, Voronoi diagrams and simplices . . . 41
2.1.2 Covering a simplex, analysis based on the θ-point . . . 42
2.1.3 SCO has a boundary optimum . . . 44
2.1.4 The regular case . . . 46
2.1.5 Conclusions . . . 47
2.2 Simplex refinement using regular simplices . . . 47
2.2.1 Regular subdivision . . . 49
2.2.2 Vertex sharing . . . 55
2.2.3 Computational aspects . . . 57
2.2.4 Conclusions . . . 61
2.3 Gradient based simplicial bounds and monotonicity considerations . . . 61
2.3.1 Preliminaries . . . 62
2.3.2 Bounding techniques over a simplex . . . 63
2.3.3 Monotonicity test . . . 67
2.3.4 Keeping track of border facets . . . 74
2.3.5 Setting border status for facets in simplicial dimension reduction . . . 77
2.3.6 Simplicial B&B algorithm (SBB) . . . 78
2.3.7 Numerical results . . . 79
2.3.8 Conclusions . . . 82

3 Modelling costumer selection rules **83**
3.1 The probabilistic choice rule 84
3.2 The partially probabilistic choice rule 86
3.3 Multi-deterministic selection rule 89
3.4 Empirical comparison of the costumer selection rules 90
3.4.1 A case study 91
3.4.2 Random problems 96
3.5 Conclusions 97

Summary **99**

Bibliography **101**

Preface

Reliable Global Optimization is an important research topic for those applications, where rigorous results are needed for the global optimum. These include chemical, physical, and also some mechanical problems, where finding just a near optimal solution is insufficient, for instance to obtain structural consequences. It is also very useful for other optimization problems, as it is known how close the current solution is to the global optimum after a finite number of steps, and also the error produced by the finiteness of computer arithmetic. However, there are few reliable global optimization methods. and one of the most used one is the interval branch-and-bound method. It has a long history for bound constrained continuous problems, but lately it was developed further for harder problems taking into account either more objective functions, difficult constraints, or even bi-level problems.

Finding the optimal location of a facility is a real-life problem. Thus, it is necessary to describe it as close to reality as possible. The models usually take into account most of the available information, and thus are usually rather complex, neither convex nor concave. Therefore, facility location problems are usually global optimization problems, requiring global optimization techniques to cope with them.

The aim of this dissertation is twofold. Our first goal is to contribute to reliable global optimization. On the one hand, we aim to provide new methods for specialized problems, which are not easy to describe by a general Nonlinear Programming (NLP) or Mixed Integer Nonlinear Programming (MINLP) formulation, or the general methods cannot exploit the special structure of the problem. These are graph-based optimization problems which include both continuous and discrete decisions, but also simplicial based problems which occur in blending problems or even in portfolio optimization. To solve such problems efficiently, we have developed specialized methods, all of them based on the branch-and-bound (B&B) principle. Namely, we have developed a method for the maximal covering problem, where the sub-problems are sets of edgesets, with the number of facilities they contain. Three different bounding rules are designed which help the method in different states. We have expanded the interval branch-and-bound method to handle a few integer variables, and also found a location-price equilibrium by a suitable interval B&B. We have also studied several aspects of the simplicial B&B in order to make it more efficient. These include regular refinement, improved bounding rules, monotonicity test and exclusion by infeasibility spheres.

Our second goal is to study new realistic models for location problems. We consider the maximal covering problem on networks with non-uniform demand, and some planar competitive facility location problems. In particular, we construct models where the attraction of a costumer towards a facility depends not only on the distance to it, but also on the quality that the costumer perceives from the facility. As in real life problems, our aim is to maximize the profit of the company, to be understood as the income obtained from the market share captured by the facilities minus the costs incurred by running them. Besides, we consider possible adjustment for existing facilities, price equilibria, and different costumer selection rules as well. To show the applicability of the new models, our last goal is to solve problems with real data by using the reliable methods previously developed.

This dissertation describes only such works, where the main results have been achieved by me. Thus, I only note on the authorship when a result is done by my co-authors, or when a result is indivisible. None of the studies appeared in any former thesis, and no work with students are included. If not specifically mentioned, the results belong mainly to me. All my co-authors have seen the dissertation and justified this personally.

Chapter 1

Reliable methods for solving special location problems

In this chapter we deal with three challenging location problems. First, we study a maximal covering problem on a network, where demand is spread on the edges with given distribution, and we seek the optimal location of p facilities such that the demand covered by facilities within the covering distance is maximal [20]. Second, we solve a complex competitive problem, where a chain wants to locate a new facility and determine its optimal quality so as to gain maximum profit. Besides, the chain can decide not to open a new facility and also to close any of the existing facilities, or change their quality, respecting the available budget [82, 83]. Third, we seek for the location and price equilibria in a competitive setting, where two firms want to locate a new facility and decide on the price of the good such that they maximize their market share [77]. Furthermore, they need to incorporate in the price the cost of the delivery as well, what is called delivered pricing in literature.

All three problems are special. The first one cannot be given analytically due to the calculation of the covered demand over any kind of distribution function. Moreover, the problem has a discrete and a continuous part as selecting the right edges is a discrete decision, while locating inside an edge is a continuous one. The firm-expansion problem also falls in the MINLP (Mixed Integer Nonlinear Program) class due to the discrete decision whether to close or not the existing facilities and also to open or not the new facility. Finally, we are going to show that the equilibria problem can be reformulated as an optimization problem where, although all variables are continuous, our objective is non-differentiable due to the minimization of functions.

What is common in the above problems is that all of them are hard to solve location problems. However, we could design a specialized branch-and-bound method which are either the first to solve them or faster than existing methods.

1.1 Maximal covering problem on a network

Covering problems are well studied in the operations research literature under the assumption that both the set of users and the set of potential facilities are finite. In this section, the following variant, which leads to a MINLP is addressed: locations of p facilities are sought along the edges of a network so that the expected demand covered is maximized, where demand is continuously distributed along the edges. This MINLP has a combinatorial part (which edges of the network are chosen to contain facilities) and a continuous global optimization part (once the edges are chosen, which are the optimal locations within such edges).

The Maximal Covering Location Problem (MCLP), [15, 40, 41, 88] is a classical problem in locational analysis with applications in many fields, such as health care, emergency planning, ecology, statistical classification, homeland security, see e.g. [3, 24, 39, 51, 174, 196] and the references therein. Given a finite set of users A, each $a \in A$ with demand $\omega_a \geq 0$, a set of p facilities in a set F is sought in order to maximize the demand covered. A point is said to be covered by a set $F^* \subset F$ of p facilities if there is at least one $f \in F^*$ at a distance from a not greater than R, where $R > 0$ is a fixed value, called the *covering radius.*

Any MCLP can be easily expressed as an Integer Program. Indeed, defining binary variables y_f and z_a to indicate, respectively, whether a facility at f is open, and whether a is covered, MCLP amounts to

solving the following program:

$$\begin{aligned} \max \quad & \sum_{a\in A} \omega_a z_a \\ \text{s.t.} \quad & z_a \le \sum_{f\in F:\, d(a,f)\le R} y_f && \forall a \in A \\ & \sum_{f\in F} y_f = p \\ & y_f \in \{0,1\} && \forall f \in F \\ & z_a \in \{0,1\} && \forall a \in A. \end{aligned} \tag{1.1}$$

MCLP is known to be NP-hard [133], but formulated as (1.1) is, in words of [166], integer-friendly, in the sense that its continuous relaxation is often all-integer, and thus no much branching is usually needed in a branch-and-bound algorithm. See [87, 141, 165, 168] and the references therein for heuristic approaches to handle problems of larger size.

Extensions and closely related models to the MCLP abound in the Operations Research literature. First, MCLP has been studied assuming that the space is not a discrete set but a network: the set A of users is the set of nodes of a network N, and facilities are allowed to be located not only at the nodes, but anywhere on N. It is shown, however, that one only needs to consider a finite and relatively small set of candidate locations, [41, 133], and thus the problem can be written in the form of MCLP above. Nontrivial extensions include, for instance, replacing the basic yes/no covering function to more general decreasing functions in the distance separating the user and the facility, [15, 16, 17, 18]; another variant is found when the set A of users is finite, but the feasible locations are assumed to be a subset of the plane, yielding planar covering models, as reviewed in [153].

Much less literature exists on covering models with *regional demand*, [67, 131, 142], in which, by the very nature of the problem, assuming the demand to be concentrated at a finite set (e.g. centroids of neighbourhoods, towns, administrative units or census boundaries, [142]) is a crude approximation. The consequences of inaccuracies due to such discretization are well studied, [49, 137, 142], and thus demand is advocated to be modeled as following a continuous distribution on a given region. See also [33, 34, 35] for other location models with continuously distributed demand.

The following version of the classic MCLP with regional demand is addressed in this study: demand is assumed to be continuously distributed along the edges of a network and p points along the set of edges of the network are sought in order to maximize the expected covering of the demand. Hence, the model differs from the classic MCLP in two main issues: first, the set of feasible locations is not a discrete set, but (a set of) the edges of a network; moreover, demand is assumed here to be distributed along the edges of the network, making it a realistic model, for instance, for covering problems in an urban context, in which users are located along streets (the edges), or for the location of emergency services to attend accidents, which take place along the roads (edges of the transportation network).

Let us now introduce formally the problem under consideration. We are given a network $N = (V, E)$; each edge $e \in E$ has associated its length l_e, which allows us to talk about points in an edge: edge e, with endpoints u, v, is identified with the interval $[0, l_e]$, and we thus identify any $x \in [0, l_e]$ as the point in the edge e at distance x of u and distance $l_e - x$ of v. With this identification, the shortest-path distance between the nodes in V is readily extended to a metric d on the points in the edges. Moreover, each edge e has a weight $\omega_e \ge 0$ and a probability density function (pdf) f_e, which models the demand along edge e. We assume that a radius $R > 0$ is given, and a point x along an edge $e \in E$ is covered by the set of facilities at $t_1, \dots, t_p$ if

$$\min_{1\le i\le p} d(t_i, x) \le R. \tag{1.2}$$

The expected demand of edge e covered by facilities at $\mathbf{t} = (t_1, \dots, t_p)$ is given by

$$\omega_e \int_0^{l_e} \delta_e(x; \mathbf{t}) f_e(x)\, dx,$$

where $\delta_e(x; \mathbf{t})$ takes the value 1 when $x \in e$ is covered by facilities at $\mathbf{t} = (t_1, \dots, t_p)$, i.e., when (1.2) is fulfilled, and takes the value 0 otherwise.

With this, the optimization problem at hand can be written as

$$\max_{\mathbf{t}\in E^p} C(\mathbf{t}) := \sum_{e\in E} \omega_e \int_0^{l_e} \delta_e(x; \mathbf{t}) f_e(x)\, dx. \tag{1.3}$$

The above model is the result of common work directed by my coauthors. The remainder of this section is structured as follows. In Subsection 1.1.1, structural properties of the MINLP (1.3) are studied. These are written by my coauthors, thus I report them only shortly for completeness. The counterexample is a result of common work. A branch-and-bound method is designed in Subsection 1.1.2. Exploiting the structure of the problem, data structures and bounding procedures are proposed, and they are tested on a set of instances in Subsection 1.1.3, which part was made by myself with some help from my co-authors. The section ends with concluding remarks and possible extensions in Subsection 1.1.4.

1.1.1 Structural properties

Statement 1.1.1. *For any p-tuple of edges* $(e_1, \ldots, e_p) \in E^p$, *the function* $C : \mathbf{t} = (t_1, \ldots, t_p) \in [0, l_{e_1}] \times \ldots \times [0, l_{e_p}] \longrightarrow C(\mathbf{t})$ *is continuous in* $[0, l_{e_1}] \times \ldots \times [0, l_{e_p}]$.

Once the p-tuple of edges $(e_1, \ldots, e_p)$ is chosen, the function C is continuous on the compact set $[0, l_{e_1}] \times \ldots \times [0, l_{e_p}]$, and attains its maximum. Since the possible choices of p-tuple of edges is also finite, the maximum of C on E^p is attained. Finding such a maximum may be hard because, for arbitrary pdfs f_e defining the demand along the edges, the function C may not be convex, and thus Global Optimization techniques are to be used; in its full generality, C may lack important structural properties, such as Lipschitz-continuity. This is shown in the following example.

Example 1.1.2. Consider a graph $N = (V, E)$ with two nodes, v_1, v_2, connected by an edge e of length 2, so that we can identify the edge with the segment $[-1, 1]$ and the nodes with the endpoints of the segment. The density f_e of the demand is given by

$$f_e(x) = \frac{1}{4\sqrt{|x|}}, \qquad x \in [-1, 1],$$

depicted in the left of Figure 1.1. Consider the problem of locating one facility ($p = 1$) on e, and a coverage radius $R = 1/4$. Let us study the behavior of the function C on the interval $[-1, 1]$. First, the cumulative distribution F_e (see the right of Figure 1.1 is easily shown to be given by

$$F_e(x) = \begin{cases} 0, & \text{if } x < -1 \\ \frac{1-\sqrt{-x}}{2}, & \text{if } -1 \leq x < 0 \\ \frac{1+\sqrt{x}}{2}, & \text{if } 0 \leq x < 1 \\ 1, & \text{if } x \geq 1. \end{cases} \tag{1.4}$$

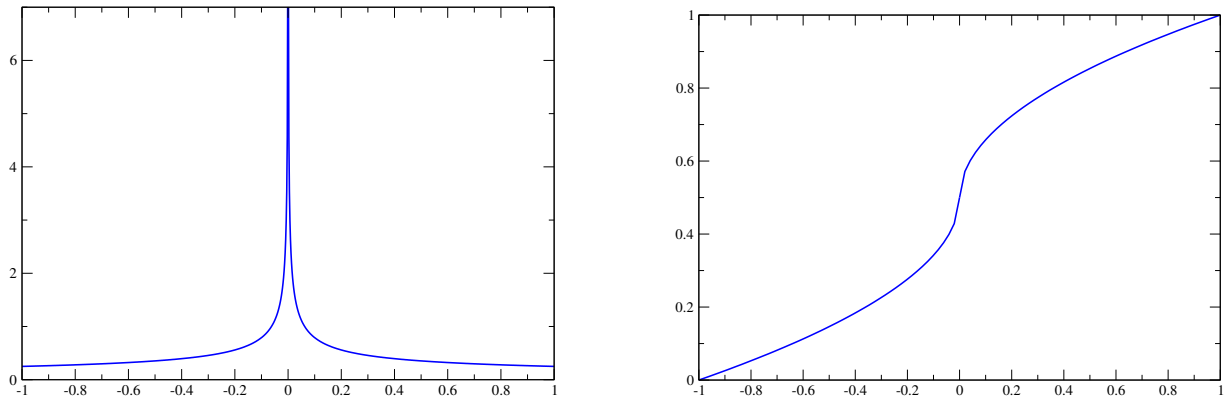


Figure 1.1: The demand density function f_e on the left, and the cumulative distribution F_e on the right for Example 1.1.2.

On the other hand, C is given by

$$C(x) = F_e\left(x + \frac{1}{4}\right) - F_e\left(x - \frac{1}{4}\right). \tag{1.5}$$

Joining (1.4) and (1.5) one obtains after some algebra the following expression of C:

$$C(x) = \begin{cases} \frac{1-\sqrt{-x-1/4}}{2}, & \text{if } -1 \leq x < -\frac{3}{4} \\ \frac{\sqrt{-x+1/4}-\sqrt{-x-1/4}}{2}, & \text{if } -\frac{3}{4} \leq x < -\frac{1}{4} \\ \frac{\sqrt{x+1/4}+\sqrt{-x+1/4}}{2}, & \text{if } -\frac{1}{4} \leq x < \frac{1}{4} \\ \frac{\sqrt{x+1/4}-\sqrt{x-1/4}}{2}, & \text{if } \frac{1}{4} \leq x < \frac{3}{4} \\ \frac{1-\sqrt{x-1/4}}{2}, & \text{if } \frac{3}{4} \leq x < 1 \end{cases},$$

which is depicted in Figure 1.2. Each part is drawn with a different color. Observe that the function C has

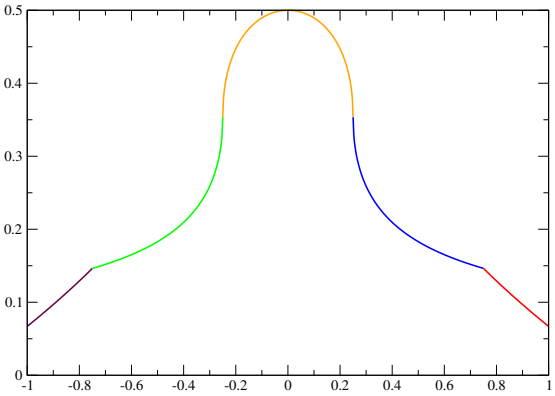


Figure 1.2: The covering function C for Example 1.1.2.

infinite directional derivatives at points $x = \pm\frac{1}{4}$, which are interior to the interval $[-1, 1]$. Hence C cannot be Lipschitz-continuous in the interval $[-1, 1]$.

Under some reasonable assumptions on the pdfs involved, the function C is Lipschitz-continuous:

Statement 1.1.3. *Suppose that, for each $e \in E$, the pdf f_e is bounded above by some constant M. Then, for any p-tuple of edges $(e_1, \ldots, e_p) \in E^p$, the function $C : \mathbf{t} = (t_1, \ldots, t_p) \in [0, l_{e_1}] \times \ldots \times [0, l_{e_p}] \longrightarrow C(\mathbf{t})$ is Lipschitz-continuous in $[0, l_{e_1}] \times \ldots \times [0, l_{e_p}]$.*

Fortunately, having a bounded pdf for the demand is a natural assumption, thus, we can design an exact method to solve our covering problem (1.3).

1.1.2 A global optimization approach

A branch-and-bound algorithm is proposed to cope with this MINLP. As in any branch-and-bound procedure, the original problem is divided into subproblems, and by finding suitable bounding procedures, one can eliminate those subproblems that cannot contain the global optimum. Thus, the two key elements are the branching and the bounding strategies, which are discussed in Subsections 1.1.2.1 and 1.1.2.2, respectively. Firstly, we define the division rules, which take advantage of the structure of the problem, by taking into account that the variables indicating the number of facilities per edge should be strongly correlated: if facilities are located at a given edge, it is unlikely that more facilities are located in neighboring edges, leaving big clusters of edges uncovered. Bounding strategies for such subdivision elements will then be built. Other important algorithmic issues of our proposal, such as the selection, elimination and termination rules, are outlined in Subsection 1.1.2.3.

1.1.2.1 Division rule

One first and naive approach is to decide first how many facilities are located within each edge, and then, once these variables are fixed, one solves, by means of a standard branch-and-bound algorithm on networks, e.g. [19, 21], the nonlinear optimization problem of deciding where to locate them. However, full inspection of all p-tuples of edges will be doable only for small networks. For this reason, our approach is to facilitate branching on the combinatorial and the continuous part at the same time.

In order to avoid the enumeration of every possible combination of p edges, we propose to construct clusters of (sub)edges. Instead of associating with each edge an integer variable indicating the number of facilities to be located in such edge, the integer variables will be associated with the clusters of (sub)edges, called hereafter *edgesets*, and the tuple of edgesets will be called *superset*.

To be precise, an edgeset is a finite collection of (sub)edges of E; a superset S is any tuple of the form $(E_1, p_1; E_2, p_2; \ldots, E_k, p_k)$, where $E_1, E_2, \ldots, E_k$ are disjoint edgesets and $p_1, \ldots, p_k$ are strictly positive integer numbers with

$$\sum_{j=1}^{k} p_j = p,$$

indicating, for each $j = 1, \ldots, k$, that exactly p_j facilities are to be located within the points in E_j.

Example 1.1.4. Consider the network depicted in Figure 1.3, with all lengths equal to 1, uniform demand on each edge, weights ω_e given by

$$\begin{array}{rcl} \omega_{12} & = & 2 \\ \omega_{14} & = & 1 \\ \omega_{23} & = & 1 \\ \omega_{34} & = & 1 \\ \omega_{45} & = & 2 \\ \omega_{46} & = & 1 \\ \omega_{56} & = & 1 \\ \omega_{67} & = & 1, \end{array} \tag{1.6}$$

and suppose $p = 3$ facilities are to be located for a covering radius $R = 1/4$. The partition of E in the three edgesets E_1, E_2, E_3,

$$\begin{array}{rcl} E_1 & = & \{(1,2), (1,4), (2,3), (3,4), (4,6)\} \\ E_2 & = & \{(6,7)\} \\ E_3 & = & \{(4,5), (5,6)\} \end{array} \tag{1.7}$$

induces, among others, the superset S

$$S = (E_1, 2; E_2, 1), \tag{1.8}$$

which corresponds to the decision of locating two facilities in the edges of E_1 and one facility in the edges of E_2.

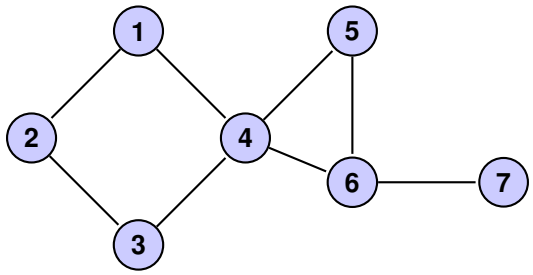


Figure 1.3: The network considered in Example 1.1.4.

Supersets will correspond to nodes in the branch-and-bound tree, as they define the subproblems. We discuss in what follows our proposal to build the starting nodes, and the way to sequentially subdivide the supersets.

Initial supersets

The root node of our branch-and-bound tree is the superset $S_0 = (E, p)$. S_0 is subdivided by using a given partition $E_1, E_2, \ldots, E_k$ of E : we add to the branch-and-bound tree list $\binom{p+k-1}{p}$ supersets of the form $(E_{i_1}, p_1; \ldots; E_{i_l}, p_l)$, where $\{i_1, ..., i_l\} \subseteq \{1, ..., k\}$ and $p_1 + \ldots + p_l = p$. Observe that, although such starting list will have a cardinality exponentially increasing in p, the difficulty of the MINLP under study only allows us to handle problems with a low value of p. Hence, the cardinality of the starting list will not grow much.

A critical issue is how the edges of the network, conforming the initial superset S_0, are split into edgesets in such a way that the so-obtained subdivision fits with the actual distribution of facilities at the optimal solution of the problem. To do this, we build from the network a discrete MCLP as follows: we consider a discrete covering problem in which we have, as possible locations, the edges of the network, we have as users also the edges e of the network, with demand ω_e, and we define the distance $d^*(e,f)$ between user e and edge f as the smallest distance between the points in e and f. Then, we consider a user e covered if $d^*(e,f) \leq R$ for some edge f. Hence, we count an edge e as fully covered (and thus, the weight ω_e is taken) as soon as some point in some f is at distance not greater than R from some point in e. Once this discrete MCLP is solved, and $f_1^*, \ldots, f_p^*$ is an optimal solution, we build the edgesets $E_1, \ldots, E_p$ so that E_j contains the edges e for which f_j^* is the closest facility.

Let us illustrate this procedure with the data of Example 1.1.4 for $p = 2$. The distance matrix d^* is then given by

	$(1,2)$	$(1,4)$	$(2,3)$	$(3,4)$	$(4,5)$	$(4,6)$	$(5,6)$	$(6,7)$
$(1,2)$	0	0	0	1	1	1	2	2
$(1,4)$	0	0	1	0	0	0	1	1
$(2,3)$	0	1	0	0	1	1	2	2
$(3,4)$	1	0	0	0	0	0	1	1
$(4,5)$	1	0	1	0	0	0	0	1
$(4,6)$	1	0	1	0	0	0	0	0
$(5,6)$	2	1	2	1	0	0	0	0
$(6,7)$	2	1	2	1	1	0	0	0

Solving the corresponding MCLP yields as an optimal solution the edges $f_1^* = (1,2)$ and $f_2^* = (4,6)$, and, starting from them, the edgesets

$$\begin{aligned} E_1 &= \{(1,2),(1,4),(2,3)\} \\ E_2 &= \{(3,4),(4,5),(4,6),(5,6),(6,7)\}, \end{aligned}$$

where, in case of ties in d^*, edges have been allocated randomly. With such definition of E_1, E_2, three supersets are obtained as split of the starting superset S_0, namely $(E_1, 2), (E_1, 1; E_2, 1), (E_2, 2)$, represented in Figure 1.4.

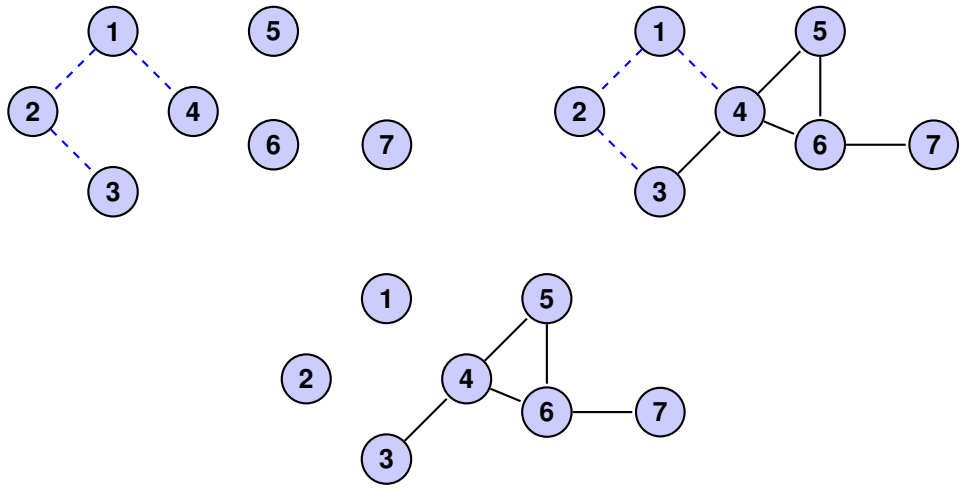


Figure 1.4: Splitting the starting superset in Example 1.1.4.

Subdivision of a superset

In order to guarantee convergence of the branch-and-bound algorithm, elements in the list should become arbitrarily small. Let us define the *diameter* $\lambda(E^*)$ of an edgeset E^* as the sum of the lengths of the (sub)edges in E^*, and define the diameter $\lambda(S)$ of a superset S as the highest length of its edgesets with assigned facilities,

$$\lambda(E_1, p_1; E_2, p_2; \ldots; E_k, p_k) = \max_j \lambda(E_j).$$

Reduction of the diameters of the supersets in the list guides our subdivision strategy. Superset $S = (E_1, p_1; E_2, p_2; \ldots; E_k, p_k)$ is subdivided as follows: first, the edgeset E_{j^*} with highest diameter is found,

$$\lambda(E_1, p_1; E_2, p_2; \ldots; E_k, p_k) = \lambda(E_{j^*}).$$

Then, the edgeset E_{j^*} is split into two subsets by identifying two "central" edges, and then clustering the edges around such edges, unless it contains only one edge, which is then bisected by its midpoint. The process in the first case, similar to the one described in Subsection 1.1.2.1 for splitting the initial set, is based on the construction of an auxiliary MCLP: a 2-facility discrete covering problem is considered, in which we have, as possible locations, the edges of the edgeset E_{j^*}, we have as users the edges e of the network, with demand ω_e, and we define the distance $d^*(e, f)$ between user e and edge f as the smallest distance between the points in e and f. Then, we consider a user e covered if $d^*(e, f) \leq R$ for some edge f. Once this discrete MCLP is solved and an optimal solution f^+, f^- is obtained, we build the sets $E^+_{j^*}$ and $E^-_{j^*}$ so that $E^+_{j^*}$ contains the edges $e \in E_{j^*}$ for which f^+ is the closest facility.

Given the splitting of E_{j^*} into $E^+_{j^*}$ and $E^-_{j^*}$, the superset S is subdivided into $p_{j^*} + 1$ supersets, by assigning, respectively, i and $p_{j^*} - i$ facilities to $E^+_{j^*}$ and $E^-_{j^*}$, $i = 0, 1, \ldots, p_{j^*}$.

By construction, one immediately has

Statement 1.1.5. *The given subdivision of the supersets is exhaustive, that is, for an infinite nested series of supersets* $\{S_q\}_{q=0}^{\infty}$, $\lambda(S_q) \to 0$ *as* $q \to \infty$.

1.1.2.2 Bounding rules

As in any branch-and-bound algorithm, procedures for giving lower and upper bounds are needed here. Lower bounds on the objective C of (1.3) are obtained by evaluating C at heuristic solutions, built as the midpoints of p (sub)edges in the superset under evaluation. Different strategies for obtaining upper bounds are described next.

Shadow bound

An easy way to obtain an upper bound for C on the superset S is to consider as covered all points in S as well as those at distance at most R of some point in S. In other words, a bound is obtained if one considers as covered the points both in S and the "shadow" of S, i.e., those points at distance R from points in S. Formally, the Shadow Bound, $\overline{C}_{SB}(S)$, for C on the superset $S = (E_1, p_1; \ldots, E_k, p_k)$ is calculated as

$$\overline{C}_{SB}(S) := \sum_{e \in E} \omega_e \int_0^{l_e} \delta_e^{SB}(x; S) f_e(x)\, dx, \tag{1.9}$$

where $\delta_e^{SB}(x; S)$ takes the value 1 when x is at distance at most R of some $y \in E_j$ and takes the value 0 otherwise.

For instance, for the data of Example 1.1.4 and the superset S in (1.8), we have

$$\begin{aligned}
\delta_e^{SB}(x; S) &= 1 \quad \forall x \in [0, 1], \forall e \in E_1 \cup E_2 \\
\delta_{(4,5)}^{SB}(x; S) &= \begin{cases} 1, & \text{if } x \in [0, 1/4] \\ 0, & \text{else} \end{cases} \\
\delta_{(5,6)}^{SB}(x; S) &= \begin{cases} 1, & \text{if } x \in [3/4, 1] \\ 0, & \text{else.} \end{cases}
\end{aligned}$$

Then, given the weights in (1.6), one obtains

$$\overline{C}_{SB}(S) = 6 + 2\frac{1}{4} + \frac{1}{4} = \frac{27}{4}.$$

By construction, the Shadow bound has the important property of monotonicity, in the sense that,

Statement 1.1.6. *If* $S = (E_1, p_1; \ldots, E_k, p_k)$ *and* $S' = (E'_1, p_1; \ldots, E'_k, p_k)$ *are supersets satisfying* $E_i \supseteq E'_i$ *for all* $i = 1 \ldots, k$, *then*

$$\overline{C}_{SB}(S) \geq \overline{C}_{SB}(S'). \tag{1.10}$$

Moreover, using the same arguments as in the proof of Property 1.1.1,

Statement 1.1.7. *If $\{(s_1^q, 1; \ldots, s_p^q, 1)\}_q$ is a sequence of supersets, where each s_j is a subedge of an edge e_j converging to some point t_j, as $q \to \infty$ then $\overline{C}_{SB}((s_1^q, 1; \ldots, s_p^q, 1)) \to C(t_1, \ldots, t_p)$.*

Hence, the bounds go arbitrarily sharp when the length of the supersets goes to zero. Consequently,

Corollary 1.1.8. *In a branch-and-bound method, if an exhaustive division rule as described in Subsection 1.1.2.1 and a convergent bounding rule, as the Shadow bound is used, the method is convergent.*

MCLP bound

The upper bound $\overline{C}_{MCLP}$ is obtained by solving a variant of a discrete MCLP as (1.1): we consider a discrete covering problem in which we have, as possible locations, the (sub)edges of the edgesets of the superset $S = (E_1, p_1; \ldots; E_k, p_k)$, we have as users the edges e of the network, with demand ω_e, and we define the distance $d^*(e, f)$ between user e and (sub)edge f as the smallest distance between the points in e and f. Then, we consider a user e covered if $d^*(e, f) \leq R$ for some (sub)edge f of some edgeset E_j.

Hence, we count an edge e as fully covered (and thus, the weight ω_e is taken) as soon as some point in some f is at distance not greater than R from some point in e. Moreover, since the number p_j of facilities within each edgeset E_j is given, we impose at most p_j different edges in E_j are to be chosen.

By construction, the optimal value of such a discrete covering problem is a valid upper bound of C on S :

$$\begin{array}{lll} \max & \sum_{e \in E} \omega_e z_e & \\ \text{s.t.} & z_e \leq \sum_{f \in \cup_j E_j} a_{ef} y_f & \forall e \in E \\ & \sum_{f \in E_i} y_f \leq p_i, & i = 1, 2, \ldots, k \\ & y_f \in \{0, 1\} & \forall f \in \cup_j E_j \\ & z_e \in \{0, 1\} & \forall e \in E, \end{array} \tag{1.11}$$

where a_{ef} is the scalar taking the value 1 if $f \in E_j$ for some j with $d^*(e, f) \leq R$, and taking the value 0 otherwise.

Contrary to what happens with the Shadow Bound $\overline{C}_{SB}$, this bound may not be sharp enough in small supersets, since, if any point of an edge is covered, then all the demand of that edge is considered as covered. For this reason, the bounding approach is not convergent.

This bound can easily be sharpened by observing that, by construction, for an edge e, if at least one point in some f in some E_j is at distance not greater than R, we are considering in (1.11) all the demand of the edge e covered, whilst a smaller amount, ω_e^*,

$$\omega_e^* = \omega_e \int_0^{l_e} \delta_e^{SB}(x, S) f_e(x)\, dx \tag{1.12}$$

can be captured. Here, $\delta_e^{SB}(x, S)$, as defined in the Shadow Bound (1.9), takes the value 1 when x is at distance at most R of some $x \in E_j$ and takes the value 0 otherwise.

In this study we call MCLP bound $\overline{C}_{MCLP}$ as the optimal value of problem (1.11) after replacing in the objective the weights ω_e by the weights ω_e^* in (1.12).

Observe that the MCLP bound is, by construction, monotonic. Moreover, when each edgeset is part of one edge, the bound obtained is exactly the Shadow Bound, and thus it will enjoy the same convergence properties as the Shadow Bound. Note also that, since an upper bound is needed, a (more crude but less expensive) upper bound is obtained if, instead of the IP (1.11), its LP relaxation is solved.

Mixed bound

The upper bounds $\overline{C}_{SB}$ and $\overline{C}_{MCLP}$ above described usually work well if the covering areas have big overlapping parts. When, on the contrary, the areas covered are almost disjoint, the problem could be split into a series of (almost) independent single-facility problems, successfully yielding sharp bounds.

More precisely, for $S = (E_1, p_1; \ldots; E_k, p_k)$, we can combine the Shadow Bound $\overline{C}_{SB}(E_j, 1)$ on E_j with any upper bound $\overline{C}_1(E_j)$ for the problem of locating one facility at some point in E_j. This way, the so-called Mixed Bound $\overline{C}_{MB}(S)$ is defined as

$$\overline{C}_{MB}(S) = \sum_{j=1}^{k} \min\left\{p_j \overline{C}_1(E_j), \overline{C}_{SB}(E_j, 1)\right\},$$

where $\overline{C}_{SB}(E_j, 1)$ is the Shadow Bound on E_j. So the problem is reduced to obtain an upper bound for the single-facility problem with the edgeset E_j as set of candidate points. If $\mathcal{F}_j$ is a collection of small subedges of the network with

$$E_j \subseteq \bigcup_{f \in \mathcal{F}_j} f,$$

then one can take as upper bound $\overline{C}_1(E_j)$ the maximum of the Shadow Bounds for locating one facility on f, when f varies in the class $\mathcal{F}_j$,

$$\overline{C}_1(E_j) = \max_{f \in \mathcal{F}_j} \overline{C}_{SB}((f, 1)),$$

yielding

$$\overline{C}_{MB}(S) = \sum_{j=1}^{k} \min\left\{p_j \max_{f \in \mathcal{F}_j} \overline{C}_{SB}((f, 1)), \overline{C}_{SB}(E_j, 1)\right\}.$$

As an illustration, consider the network in Example 1.1.4 and the superset S in (1.8). If, for each edgeset E_j, we define the split $\mathcal{F}_j$ as the edges of the network in E_j, we have:

$$\begin{array}{lcl}
\overline{C}_1(E_1) & = & \max\{\overline{C}_{SB}((1,2),1), \overline{C}_{SB}((1,4),1), \overline{C}_{SB}((2,3),1), \overline{C}_{SB}((3,4),1)\} \\
& = & \max\{10/4, 9/4, 7/4, 8/4\} = 10/4 \\
\overline{C}_{SB}(E_1, 1) & = & 7 \\
\overline{C}_1(E_2) & = & 5/4 \\
\overline{C}_{SB}(E_2, 1) & = & 3/2 \\
\overline{C}_{MB}(S) & = & 2 \cdot 10/4 + 5/4 = 25/4.
\end{array}$$

Note that, by construction, the Mixed Bound $\overline{C}_{MB}$ is monotonic. However, since it calculates separately the covering of each edgeset E_j, in case of overlapping in the areas covered, such points are counted more than once. Hence, the bound is not necessarily convergent.

1.1.2.3 Further algorithmic issues

In order to have a functional method, some other rules are necessary, although these are some of the usual rules.

Selection Rule: The next superset to be evaluated is the one with the highest upper bound on the list.

Elimination Rule: Whenever a superset S is such that $\overline{C}(S) < LB$, any possible location of the facilities in the edgesets of S would lead to a worse covering that the best solution we have so far, therefore the set S can be omitted from further consideration.

Termination Rule: When the relative error of the largest upper bound and the best found solution is less than the tolerance ε, the algorithm stops. The supersets remaining on the list contain the global optimum, and the best solution found so far is reported.

1.1.3 Computational results

Our branch-and-bound algorithm was implemented in Fortran 90 (Intel©Fortran Compiler XE 12.0), using the integration tools of the IMSL Fortran Numerical Library and calling the MIP solver of Cplex 12.5. Executions were carried out on an Intel Core i7 computer with 8 GB of RAM memory at 2.8 GHz, running Windows 7.

Two types of experiments were performed. First, a series of networks of medium size, obtained e.g. from [19, 47], were solved for a small number p of facilities: $p = 2, 3, 4$. In order to analyze the impact of p on the running times, we have tested our method on a small network, the Sioux-Falls, taken from [124].

Let us describe now the first experiment class. Problems on 7 test networks obtained are solved. The number of nodes of these networks ranges from 150 to 225, and the number of edges from 296 to 386. Demand parameters are randomly generated: the overall demand ω_e of an edge e is assumed to follow a

uniform distribution in $[0, l_e]$, and the demand along each edge is assumed to follow a beta distribution with parameters randomly generated in the interval [0.1, 5], which provides a wide range of density functions with different shapes. We stress that we have chosen the beta distribution just because the beta class is versatile enough and it requires numerical integration routines for evaluation, so the usefulness of the method is demonstrated in a difficult case. However, arbitrary densities could have been used instead.

On each network, the problem is solved for p facilities, $p = 2, 3, 4$, and three different radii R, a small, a medium and a large one with respect to the diameter of the networks. The stopping criterion is set to the relative gap of 10^{-3} for all problems.

In order to see the efficiency of the bounding rules, different settings, using the different bounding schemes proposed in the study, were compared. In all cases, the Shadow Bound $\overline{C}_{SB}$ was calculated to guarantee convergence of the branch-and-bound algorithm, and, if needed, to compute the coefficients ω_e^* in the MCLP bound $\overline{C}_{MCLP}$. The following strategies were tested:

SB: Just the Shadow Bound is calculated.

MCLP: In addition to the Shadow Bound (needed to calculate ω_e^*), the MCLP bound is also calculated.

MB: Both the Shadow Bound and the Mixed Bound are calculated.

ALL: All three bounds, namely the Shadow Bound, the MCLP and the Mixed Bound, are calculated.

Smart: Heuristic bound rule, where, for every third level in the division tree at each superset, all the bounding rules are calculated. The most efficient rule is stored for each superset, where efficiency is measured by means of a merit function which combines sharpness of the bounds and computational time: i is the most efficient bound if for any bound j it holds that $2^{R_{UB}-1}R_T > 1$, where $R_{UB} = \frac{\overline{C}_j - \tilde{f}}{\overline{C}_i - \tilde{f}}$ is the ratio of overestimations, and $R_T = \frac{T_j}{T_i}$ is the ratio of computational time for bounds j and i; otherwise the second best bound is chosen.

Once the most efficient bound is identified, only such bound is calculated for their descendants in the next two levels.

In Tables 1.1–1.3, running times in seconds of the different bounding approaches are presented for the different values of p and R. In the tables results are grouped by the radius, and average values are also shown. For the instances which did not terminate in 5 hours (18 000 sec), the achieved relative gap is reported. The best approach for each problem is highlighted.

In Table 1.1, the results for $p = 2$ are shown. One can see clearly the not surprising differences from one approach to the other with respect to the radius. Namely, while for the SB and MCLP approaches running time is decreasing as R is increasing, for MB is just the opposite. The balance of forces is already clear: although SB and MCLP are good for large radius, MB is necessary for small and medium R. Our Smart rule is shown to be the best for small and medium radii, while for large R almost always SB was the most efficient.

In Table 1.2, the running times and achieved gaps are shown for $p = 3$. For the SB and MCLP approaches, most problems with small radius are intractable, since the gap after 5 hours of running time is still over 15-25% on average. With the exponential growth of possibilities for the solution, the MB approach gets more useful. This happens because the evaluation of the Mixed Bound is expensive rather at the beginning of the algorithm, when the maximal bound for each edge is calculated, but it takes almost no time until bounds on small segments have to be evaluated. While from the pure bounding rules MB is almost always the best, the Smart approach still has a slightly better performance.

In Table 1.3, results for $p = 4$ are shown for only the MB, ALL, and Smart approaches, since SB and MCLP can solve only the PR152G problem with large R. Although the Smart approach is still the best one on average, we can see that the average time is similar for the different approaches. This is due to the fact that many problems were stopped after 5 hours, making averages similar (and high).

Let us discuss now the second experiment. In order to see how the results change as p grows, the Smart bounding rule was used for a small (24 nodes and 39 edges) network, namely, the Sioux-Falls network [124].

In Table 1.4, computational times are given for $p = 2, \ldots, 7$, and, as in the first type of experiments, three different radii. For the large radius, when $p = 6, 7$ more than 100000 supersets needed to be stored in the list; this was the maximum allowed in the program, so the reached gap was also reported in these cases.

Table 1.1: Running times ($p = 2$).

Graph	R	SB	MCLP	MB	ALL	Smart
KROA150G	small	286.4	271.2	34.8	37.2	30.7
KROA200G		413.8	373.7	36.1	38.5	33.4
KROB150G		833.4	847.3	67.3	69.5	54.2
KROB200G		789.0	770.5	53.2	56.7	45.7
PR152G		171.5	182.6	20.8	21.6	19.3
RAT195G		2021.5	2000.9	37.8	40.0	31.9
TS225G		301.4	293.3	14.6	16.9	14.1
Average		688.1	677.1	37.8	40.1	32.8
KROA150G	medium	384.9	378.5	45.7	47.1	38.6
KROA200G		269.2	258.9	92.2	98.7	86.3
KROB150G		287.0	282.0	34.2	37.8	31.2
KROB200G		538.5	544.7	190.2	202.2	181.3
PR152G		12.3	16.8	6.1	6.5	6.0
RAT195G		716.6	696.4	112.8	116.5	91.2
TS225G		242.8	178.4	29.3	32.7	25.6
Average		350.2	336.5	72.9	77.4	65.7
KROA150G	large	2.3	3.4	21.0	22.0	21.7
KROA200G		607.0	622.2	669.2	694.1	677.9
KROB150G		32.2	36.6	55.0	61.0	55.2
KROB200G		2.2	3.8	25.1	26.8	25.0
PR152G		15.7	20.3	9.8	12.1	11.0
RAT195G		2.8	6.4	22.7	26.6	24.4
TS225G		44.9	50.1	65.1	71.3	62.7
Average		101.0	106.1	124.0	130.6	125.4
Average	–	379.8	373.2	78.2	82.7	74.6

Observe that for the small and large radii, an explosion in running times happens from $p = 5$ to $p = 6$, whereas for the medium radius it is rather from $p = 4$ to $p = 5$. It is also interesting to see that the difficulty can vary from problem to problem, as for small radius and $p = 7$ facilities, it can be solved faster than the same problem with 6 facilities. This may be due to the number of local optima which are close to the global optima, or due to the flatness of the objective function near the global optimizer. Even though more extensive testing needs to be performed to fully understand the dependence of running times of the covering problems with respect to all the parameters involved, it is clear from our tests that the running times increase exponentially when p increases.

1.1.4 Conclusions

In this section we have studied a covering location problem on networks which, contrary to those already in the literature, assumes the demand distributed along the edges of the network, which is a more realistic assumption for problems with networks representing high-density regions, such as cities. The problem is a challenging MINLP, in which combinatorial decisions (which edges of the network are to be selected to contain facilities) are coupled with continuous decisions (where to locate the facilities once the edges are chosen).

A branch-and-bound algorithm has been developed for this MINLP. While some ingredients of such branch-and-bound are standard, the branching procedure is rather specific, since it successfully exploits the fact that the locational decisions are taken on a network. Different bounding rules are proposed and tested on different networks; it is shown that the so-called Smart strategy, which through a learning process, is identifying for each branch-and-bound node the most promising branching strategy, is the most promising in terms of running times.

For the resolution of the problem, we have also considered a special type of superset, where no information about the number of facilities in each edgeset is stored. For these supersets similar bounding

Table 1.2: Running times and gaps ($p = 3$).

Graph name	R	SB T(s)	SB Gap	MCLP T(s)	MCLP Gap	MB T(s)	ALL T(s)	Smart T(s)
KROA150G	small	–	0.254	–	0.218	337.8	355.0	285.0
KROA200G		–	0.149	–	0.098	243.3	252.3	181.7
KROB150G		–	0.276	–	0.206	156.1	164.1	124.2
KROB200G		–	0.209	–	0.142	453.1	446.7	363.0
PR152G		15770.3	–	16863.6	–	37.2	43.9	31.3
RAT195G		–	0.633	–	0.451	93.1	112.2	72.6
TS225G		–	0.103	–	0.086	167.5	183.4	121.7
Average		17681.5	0.232	17837.7	0.172	212.6	222.5	168.5
KROA150G	medium	12146.4	–	11096.7	–	269.8	298.9	238.5
KROA200G		4410.2	–	3992.2	–	111.8	120.4	99.0
KROB150G		–	0.001	16591.4	–	632.8	652.6	477.1
KROB200G		6332.9	–	4678.2	–	198.6	196.0	144.7
PR152G		1009.3	–	1103.3	–	25.1	27.8	24.3
RAT195G		–	0.101	–	0.071	3804.5	3794.3	3329.6
TS225G		–	0.038	–	0.005	210.6	241.5	182.6
Average		11128.4	0.021	10494.5	0.012	750.5	761.6	642.3
KROA150G	large	3072.0	–	3178.4	–	2987.0	3035.1	2978.9
KROA200G		4481.7	–	4675.3	–	3155.2	3277.2	3158.3
KROB150G		1993.0	–	1960.2	–	752.8	770.4	700.6
KROB200G		270.7	–	284.9	–	282.8	304.7	277.7
PR152G		77.2	–	106.2	–	17.0	22.0	18.2
RAT195G		150.4	–	169.3	–	181.9	201.6	182.7
TS225G		2686.5	–	1951.6	–	1872.5	1525.7	1574.3
Average		1818.8	0.001	1760.8	0.001	1321.3	1305.2	1270.1
Average	–	10209.6	0.085	10031.0	0.061	761.5	763.1	693.6

rules can be derived, although in some cases giving looser bounds. The advantage of this data structure is that it reduces the exponential growth of the number of supersets as p increases, but for the number of facilities in the experiments we have performed the results were similar. However, it may give better results for higher number of facilities, and thus this alternative approach deserves further analysis and testing.

Several extensions of the problem are possible, and in most cases the bounding strategies proposed in this work could be adapted to such extensions. To mention a few, the most straightforward extension would be the addition of capacity constraints to the covering model, as proposed e.g. in [151]. On the other hand, we have assumed the demand along each edge to follow an absolutely continuous random variable. The more general case in which the demand is expressed as a mixture of an absolutely continuous random variable and a discrete variable with finite support can be handled in the same way, by splitting the edges at the preprocessing step into subedges in such a way that the cover of points with positive mass is constant along each subedge.

A third line of extensions would consist of including congestion effects, as proposed for (standard) covering models e.g. in [38, 129]. This calls also for the re-definition of the objective, since, in this case, the potential users causing the congestion are not identified by a finite set. The fourth and most challenging extension consists of incorporating competition issues in the covering problem [50, 56, 155, 156]: in a leader-follower problem, the location of the follower is a covering problem, similar to the one described here; solving the leader problem is a much harder problem than the one addressed here, since one has to solve a bilevel problem in which the follower strategy is the one described in this study. This, as well as the other extensions, deserve further study, not only by its implications in location analysis (more realistic models for dense demand are considered) but also from the Global Optimization viewpoint, since new, challenging MINLPs are addressed with new branch-and-bound procedures.

Table 1.3: Running times and gaps ($p = 4$).

Graph name	R	MB T(s)	MB Gap	ALL T(s)	ALL Gap	Smart T(s)	Smart Gap
KROA150G	small	–	0.003	–	0.003	–	0.003
KROA200G		2612.6	–	2792.8	–	2068.5	–
KROB150G		4367.5	–	4867.2	–	3658.3	–
KROB200G		–	0.019	–	0.020	–	0.014
PR152G		225.5	–	261.5	–	171.8	–
RAT195G		1862.2	–	2105.0	–	1423.3	–
TS225G		435.7	–	535.6	–	357.2	–
Average		6500.5	0.004	6651.7	0.004	6239.9	0.003
KROA150G	medium	2428.6	–	2573.0	–	1875.6	–
KROA200G		846.8	–	881.9	–	736.5	–
KROB150G		5619.4	–	5771.9	–	4694.6	–
KROB200G		4403.2	–	4581.3	–	3406.7	–
PR152G		9976.9	–	10681.3	–	8892.0	–
RAT195G		–	0.049	–	0.049	–	0.047
TS225G		432.0	–	749.1	–	405.9	–
Average		5958.1	0.008	6176.9	0.008	5430.2	0.008
KROA150G	large	–	0.044	–	0.041	–	0.042
KROA200G		–	0.013	–	0.013	–	0.013
KROB150G		–	0.004	–	0.005	–	0.004
KROB200G		–	0.048	–	0.042	–	0.042
PR152G		16.1	–	21.5	–	17.9	–
RAT195G		–	0.049	–	0.044	–	0.045
TS225G		–	0.002	10603.9	–	10914.5	–
Average		15430.9	0.023	14375.1	0.021	14418.9	0.021
Average	–	9296.5	0.012	9067.9	0.011	8696.3	0.011

Table 1.4: Running times and gaps for the Sioux-Falls network.

p / R	2 T(s)	3 T(s)	4 T(s)	5 T(s)	6 T(s)	6 Gap	7 T(s)	7 Gap
small	3.0	5.4	20.7	91.4	10652.8	–	6487.5	–
medium	10.2	29.2	257.7	6123.1	38222.6	–	58451.4	–
large	9.8	147.7	1256.5	1493.6	49633.6	0.0012	46297.7	0.14

1.2 Firm expansion: an interval B&B method for a MINLP

In this research, a continuous *competitive* facility location problem is investigated, which means that there are other facilities in the same region offering the same product or service and a competition exists to maximize market share or profit (see [7, 59, 61, 72, 97, 152] and the references therein for a review of the topic). But when a chain is already operating in the area and wants to expand its presence, in addition to locating *new* facilities it may also invest on its *existing* facilities, increasing or decreasing their quality, or closing some of them (so as to invest the budget allocated to them to the other ones). However, to our knowledge, in none of the models in the literature the existing facilities of the locating chain are allowed to vary their quality or to be closed. And this is exactly what it is researched in this study. The most related papers with this topic are the following ones. In [172], a continuous competitive single facility location model with foresight is analysed, in which the facilities of the competitors may adjust their quality up or down; but the locating chain is new in the area (it does not own any existing facility) and an unlimited budget is assumed. A related leader-follower *discrete* model is considered in [122]. The leader is to open an unknown number of facilities, and the follower reacts by adjusting the quality of its existing facilities

and/or closing some of them and/or opening some new facilities; again, the leader is a new entrant and an unlimited budget is assumed. In both papers the customer choice rule is probabilistic, i.e., the demand at a demand point is distributed among all the facilities according to their attraction. In the *discrete* model introduced in [64], a different approach is followed: it is assumed that customers patronize a facility only if they are within its 'radius of influence' (which can be interpreted as a surrogate of the quality) and if they are in the sphere of influence of more than one facility then their demand is equally divided among them; the locating chain owns some of the existing facilities, it has a given budget to expand its presence in the area, and it may open new facilities (in a discrete set of potential locations) and/or upgrade (downgrading is not allowed) the quality (increase the radius of influence) of its existing ones. See [74] for a deeper review of those papers.

As it will be shown in Subsection 1.2.1, the continuous location model introduced in this work is a non-convex mixed-integer nonlinear programming (MINLP) problem. Many researches have been working on the topic of designing and implementing practical algorithms for coping with MINLP problems during the last decades (see [12, 23, 28, 191] and the references therein). And some software packages are available, for both convex (AlphaECP [195], BONMIN [22], Knitro [29], or MINOTAUR [127]) and non-convex MINLP problems (ANTIGONE [138], BARON [171], COUENNE [11], or SCIP [192]). However, since the available packages do not seem to be successful at solving the model introduced here, solution methods are also proposed in this study.

This work investigates up to what extent the possibilities of varying the qualities of the existing facilities and closing them affects the solution of the problem, both in profit and in location and quality of the new facility (in case it is open). The main contributions of the present work are: (1) a new *continuous* location and design model (an extension of the model in [76]) is introduced (which was a common work with José Fernández); the *expanding* chain has a *limited budget*, and it may locate a new facility anywhere in *a given region of the plane* and/or *vary (up or down) the quality of its existing facilities* and/or *close some of them*; the customer choice rule is assumed to be *probabilistic*. (2) A branch-and-bound method based on interval analysis is proposed in Subsection 1.2.2; it is suitable for small-size instances and can also be applied to many other MINLP problems. This was my main contribution done with José Fernández. The computational studies (in Subsection 1.2.3) which show the usefulness of the approaches, are done by me. The discussion ends in Subsection 1.2.4 with conclusions. The original work contains also heuristic methods for this problem, to which my contribution was rather small, so that part is left out here.

1.2.1 The location and design model for firm expansion

The new model we are going to introduce here is an extension of the model in [76], which is described in Subsection 3.1. The model in [76] is a pure-location model, in the sense that the expanding chain considers only the opening of one new facility in the area. Some existing facilities may belong to the locating chain. The location and quality of all the existing facilities is known as well as the location and demand of customers (or demand points). The utility provided by a facility to a customer is measured as quality divided by (a function of the) distance, and customers are assumed to follow the probabilistic choice rule.

In the original model, the expanding chain considers only one possibility to increase its profit: opening one new facility. However, in practice, the chain could have more options. It could also consider improving the quality of (some of) its existing facilities. Or downgrading some of them so as to allocate the budged invested in them to other facilities: if a facility has no competitors too close, then maybe a decrease in its quality, even though it provokes a decrease in its attraction, may not lead to a big loss of its market share; it can even win more profit due to the lower operational costs; and, even if it is worse for that particular facility, it can be better for the chain as a whole. In fact, it may be the case that some of the existing facilities are not profitable at all, and closing them and investing their budget in other facilities (or in the new one, if it is finally open) may be the best strategy. The following extended model takes into account all those possibilities. Note that in this model it is assumed that the expanding chain already has some facilities operating in the area.

We introduce the following notation for the model:

Index sets

- i subscript for demand points, $i = 1, \ldots, i_{\max}$.
- j subscript for the existing facilities, $j = 1, \ldots, j_{\max}$ (the first k of them ($k < j_{\max}$) are owned by the expanding chain).

Variables

f_0 coordinates of the new facility, $f_0 = (f_0^1, f_0^2)$.
α_0 quality of the new facility.
α_j quality of the j-th existing facility owned by the chain, $j = 1, \ldots, k$. At present, $\alpha_j = \tilde{\alpha}_j$.
y_j binary variable which is 0 if facility j is closed ($j = 1, \ldots, k$) or not open ($j = 0$); 1 otherwise.
ns variables of the problem, $ns = (f_0, \alpha_0, \alpha_1, \ldots, \alpha_k, y_0, \ldots, y_k)$.

Parameters

p_i location of demand point i.
w_i annual demand (or buying power) concentrated at p_i.
f_j location of existing facility j.
d_{ij} distance between p_i and f_j.
$d_i^{\min}$ radius of the circular area around p_i where the location is not allowed.
S area where the new facility can be set up.
$\tilde{\alpha}_j$ present quality of existing facility f_j, $j = 1, \ldots, j_{\max}$.
$g_i(\cdot)$ a non-decreasing (and non-negative) function.
$\tilde{u}_{ij}$ utility that p_i perceives from f_j at present, $i = 1, \ldots, i_{\max}$, $j = 1, \ldots, j_{\max}$, $\tilde{u}_{ij} = \tilde{\alpha}_j / g_i(d_{ij})$.
$\alpha_{\min}$ minimum quality level for the new facility.
$\alpha_{\max}$ maximum quality level for the new facility.
$\alpha_{\max}^j$ maximum quality level that f_j may have, $j = 1, \ldots, k$.
A_j annualized cost corresponding to the opening of the j-th facility, $j = 1, \ldots, k$.
C_j cost of closing facility f_j, $j = 1, \ldots, k$.
B annual chain's budget (for opening a new facility or varying the quality of existing facilities or closing facilities as well as for the operational costs of the open facilities).

Computed values

$d_i(f_0)$ distance between the new facility and p_i.
$u_{i0}(f_0, \alpha_0)$ utility that p_i perceives from the new facility, $u_{i0}(f_0, \alpha_0) = \alpha_0 / g_i(d_i(f_0))$.
$u_{ij}(\alpha_j)$ utility that p_i perceives from f_j, $j = 1, \ldots, k$, $u_{ij}(\alpha_j) = \alpha_j / g_i(d_{ij})$.
$V_j(\alpha_j)$ annualized cost of varying the quality of f_j to α_j, $j = 1, \ldots, k$.
$M(ns)$ annual market share captured by the chain.
$F(M(ns))$ annual expected sales obtained by the chain.
$R_j(\alpha_j)$ annual operating cost of the j-th facility, $j = 0, \ldots, k$.
$T(ns)$ annual total cost of the chain.
$\Pi(ns)$ annual profit obtained by the chain.

Since the j-th facility ($j = 1, \ldots, k$) is already established, its area can hardly be modified. Hence, most likely its quality can be upgraded from $\tilde{\alpha}_j$ up to certain level $\alpha_{\max}^j \leq \alpha_{\max}$.

In addition to the annualized cost $G(f_0, \alpha_0)$ of opening the new facility (in case it is open) at a given location f_0 with a quality α_0 (note that now G does not include the annual operating cost), some additional costs must be taken now into account, namely, $A_j, C_j, V_j(\alpha_j)$, $j = 1, \ldots, k$, and $R_j(\alpha_j)$, $j = 0, \ldots, k$, as described above.

With the previous notation, the annual cost of the expanding chain is given by

$$T(ns) = \sum_{j=1}^{k} (y_j(A_j + R_j(\alpha_j) + V_j(\alpha_j)) + (1 - y_j)C_j) + y_0(G(f_0, \alpha_0) + R_0(\alpha_0)), \tag{1.13}$$

and the market share it captures is

$$M(ns) = \sum_{i=1}^{i_{\max}} w_i \frac{y_0 u_{i0}(f_0, \alpha_0) + \sum_{j=1}^{k} y_j u_{ij}(\alpha_j)}{y_0 u_{i0}(f_0, \alpha_0) + \sum_{j=1}^{k} y_j u_{ij}(\alpha_j) + \sum_{j=k+1}^{j_{\max}} \tilde{u}_{ij}}. \tag{1.14}$$

The profit maximization problem (P) can be stated as follows:

$$\max \quad \Pi(ns) = F(M(ns)) - T(ns) \tag{1.15}$$
$$\text{s.t.} \quad f_0 \in S \subset \mathbb{R}^2 \tag{1.16}$$
$$d_i(f_0) \geq d_i^{\min}, \qquad i = 1, \ldots, i_{\max} \tag{1.17}$$
$$y_0 \alpha_{\min} \leq \alpha_0 \leq y_0 \alpha_{\max} \tag{1.18}$$
$$y_j \alpha_{\min} \leq \alpha_j \leq y_j \alpha_{\max}^j, \qquad j = 1, \ldots, k \tag{1.19}$$
$$T(ns) \leq B \tag{1.20}$$
$$y_j \in \{0, 1\}, \qquad j = 0, \ldots, k \tag{1.21}$$

Constraint (1.18) guarantees that if the new facility is opened ($y_0 = 1$) then $\alpha_0 \in [\alpha_{\min}, \alpha_{\max}]$, and if it is not opened ($y_0 = 0$) then $\alpha_0 = 0$. Something similar is done in constraints (1.19) for the quality of the existing facilities.

The cost function due to the change in the quality of facility f_j from its present value $\tilde{\alpha}_j$ to α_j, $V_j(\alpha_j)$, should be non-increasing in the interval $[\alpha_{\min}, \tilde{\alpha}_j)$ and non-decreasing in $(\tilde{\alpha}_j, \alpha_{\max}^j]$, as the bigger the difference $|\alpha_j - \tilde{\alpha}_j|$, the higher the investment required to do the modifications. Furthermore, one would expect $V_j(\tilde{\alpha}_j + \alpha_j) > V_j(\tilde{\alpha}_j - \alpha_j)$, since upgrading the quality is more expensive than downgrading it. And similarly to G, one would expect $V_j(\alpha_j)$ to be convex in $(\tilde{\alpha}_j, \alpha_{\max}]$ (also in $[\alpha_{\min}, \tilde{\alpha}_j)$), since the more quality we expect from the facility the higher the costs will be, at an increasing rate. Additionally, whenever a variation is made, a fixed cost $v_j > 0$ has to be paid (usually a variation in the quality requires a temporary closure of the facility). This fixed cost prevents too small variations; in fact, a high v_j value would prevent any change. A possible expression for $V_j(\alpha_j)$ could be the following one:

$$V_j(\alpha_j) = \begin{cases} \frac{1}{\delta_j}(G_2(2\tilde{\alpha}_j - \alpha_j) - G_2(\tilde{\alpha}_j)) + v_j & \text{if } \alpha_j < \tilde{\alpha}_j \\ 0 & \text{if } \alpha_j = \tilde{\alpha}_j \\ G_2(\alpha_j) - G_2(\tilde{\alpha}_j) + v_j & \text{if } \alpha_j > \tilde{\alpha}_j. \end{cases}$$

The parameter $\delta_j > 0$ determines how much cheaper is decreasing the quality as compared to increasing it.

As for the operating cost function $R_j(\alpha_j)$, it should be non-decreasing, although its functional form may vary depending on the particular type of facility. In this research a linear form is assumed, $R_j(\alpha_j) = o_j \alpha_j$, with $o_j > 0$ a given constant. However, note again that the particular choice of functions V_j and R_j does not affect the study of the aim of the paper.

Problem (P) is a MINLP problem. Therefore, it is a challenge from the optimization point of view, and global optimization tools are required to cope with it.

1.2.1.1 An example

The model has been applied to the case of the location of a hypermarket in an area around the city of Murcia, in south-eastern Spain [190]. The aim of this subsection is to show how the model works.

There are $i_{\max} = 21$ population centers (demand points) in the area. Figure 1.5 shows their location as a gray circle (or dot) with a radius proportional to its buying power (which in turn is considered to be proportional to its number of inhabitants). The location of the new facility is forbidden in those circles. There are five hypermarkets in the area: two from the expanding chain (marked with a green cross) and three from the competitor (marked with a red dot). In both cases, the original quality is marked on the right pane of the figure with the same symbol as the facility. The same problem has been solved with the exact method described in Subsection 1.2 for different budgets, and we can see in the pictures how the solution varies depending on the budget. The location and quality of the new facility, in case it is open, is shown inside a large blue box; 'New facility' is written on top of the box, and an interval hull for the solution boxes are shown on the location and on the quality pane as well. The new quality of the existing facilities are also shown on the quality pane by a box; a green arrow points to the new quality. When a facility is closed an orange box is drawn, and 'Closed' is written under the box. When there are several global (or near-optimal) solutions different only in the quality values, like in the figure with a budget $B = 52$, we can see that the boxes for the inclusion of the qualities become larger. In such cases, a low quality on one facility is paired with a high quality for the other facility, but for the figure we show the inclusion of all the solutions.

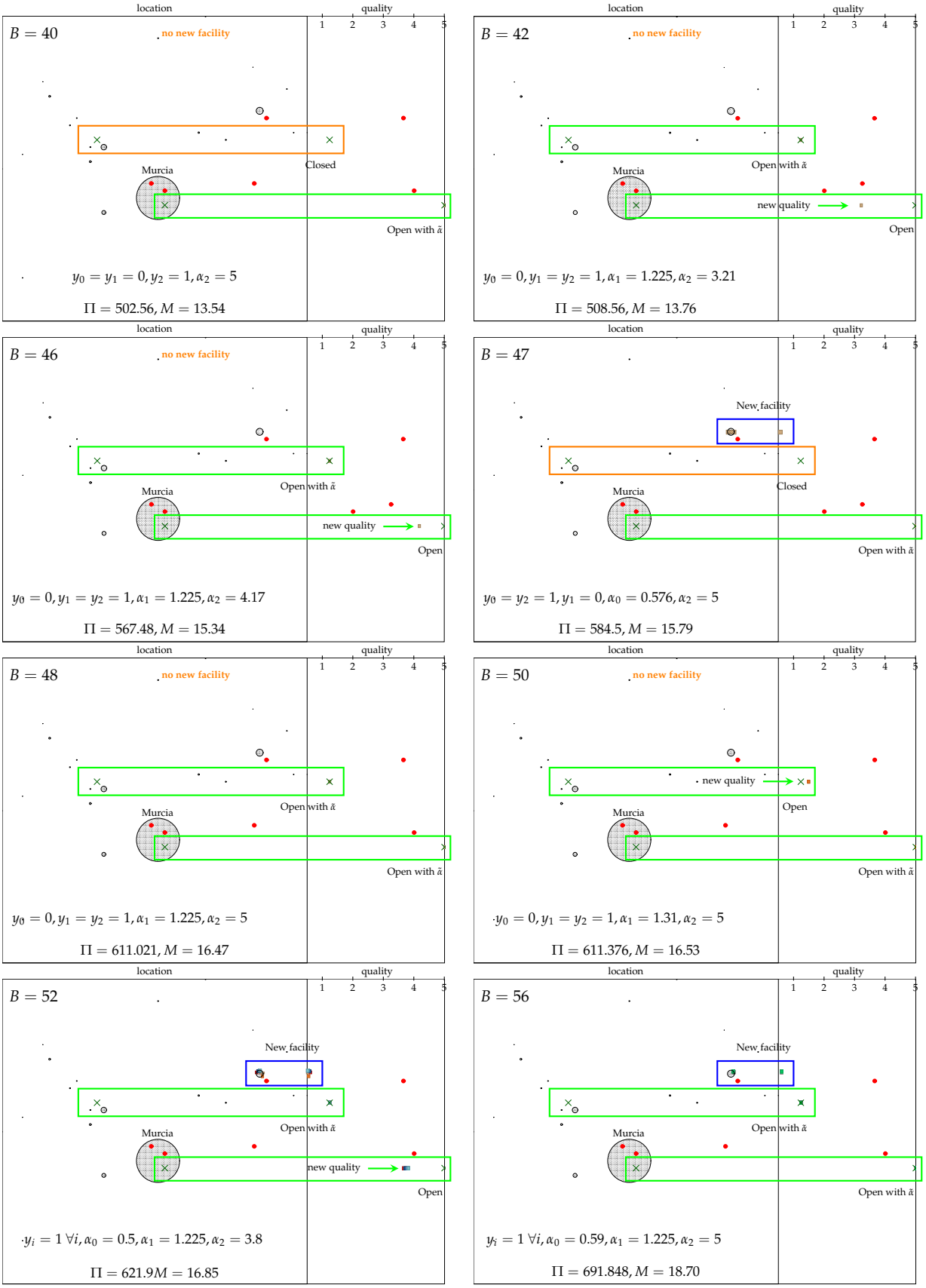


Figure 1.5: Solution of the small problem of Murcia with 21 demand points for different budgets

In the first figure, to fulfill the low budget $B = 40$, only the facility in Murcia (the most populous city) is kept open (with a high quality), the other shop has to be closed, and no new facility can be opened. The same happens for budget $B = 41$, but then for $B = 42$ both existing facilities are kept opened, although the quality of the facility in Murcia has to be reduced; no new facility is opened. From budget 42 to 46, the solution only changes in the quality of the facility of Murcia, increasing from 3.21 to 4.17. A major change happens when $B = 47$, where the new facility is opened with minimal quality, directly competing against one of the competitor's facilities; and of the two existing facilities, only the one in Murcia is kept opened, with its original quality; the other one is closed. Interestingly, this is the only budget where it is worth opening a new facility below the budget of 52. For budgets 48 and 49 the solution is to keep opened the two existing facilities with the original qualities (the new facility is not opened), while for the budgets 50 and 51, we can also increase the quality of our low quality facility. From the budget $B = 52$, it is again worth to open a new facility, and now all existing facilities remain open; however, the quality of the existing facilities are decreased: the quality of the facility in Murcia to 3.7–3.8 and the quality of the other one to its original low quality. When increasing the budget from 52 to 60, first the quality of the facility in Murcia is increased up to its original maximum quality of 5 (being the quality of the new facility minimal), and from budget 56 only the quality of the new facility is increased.

We can see how tricky the problem is, where small changes in the budget may provoke completely new configurations of the solutions.

1.2.1.2 Difficulties in solving the model

We implemented the model in AMPL [81] and tried first to solve two problems for different budgets, with all the solvers suitable for MINLP problems callable from AMPL, namely, BARON [171], BONMIN [22], COUENNE [11], SCIP [192], Knitro [29], LGO [150], and LocSol [14].

The first problem is a toy one, with just $i_{\max} = 4$ demand points, $j_{\max} = 4$ existing facilities, one of them belonging to the expanding chain ($k = 1$). Only Knitro succeeded at solving more than one of the ten instances of our first problem. But changing the conditional definition of V_j using additional binary variables BARON, SCIP, and LocSol were able to solve the problems as well. For LocSol, we had to set a reasonable time limit, because it is used as stopping criterion. Setting the time limit to 1 second for this toy problem (both BARON and our interval branch-and-bound method finished in less time) LocSol found near optimal solutions in 9 out of the 10 problems. BONMIN without any good starting point, failed at solving the toy problem regardless of the budget. SCIP could solve all the toy problems optimally but, on average, it was more than 50 times slower as compared to any of the other successful methods.

The second test problem is the quasi-real example of the previous subsection. Concerning this problem, without any starting point, BONMIN found the optimal solution only for 1 out of the 15 instances; in the rest of problems it just found a local optimum. Similarly, LGO and Knitro found the optimal solution only once of the 15 cases. SCIP found the optimal solution 11 out of the 15 problems but, on average, it spent more than 5 hours on the problems it could solve. On the other hand, BARON was able to solve all the instances optimally. Comparing the results obtained by BARON and our method, we found that for the budget settings, our method was able to find the enclosing interval of the global optima in 25 seconds on average, while BARON took 800 seconds on average. For the problems whose budget leads to the opening of the new facility and also to keep opened the existing facilities, the number of near-optimal solutions is high, due to the fact that increasing the quality of one facility can be compensated (both in profit and budget) by decreasing the quality of another facility. Our interval branch-and-bound method aims to find all global optimal points (in fact, to enclose all near-optimal points within a given accuracy). Opposed to this, BARON aims to find just one global optimum point within the given tolerance and stops immediately. Thus, in those instances BARON finds one global optimum point faster than our method finds (enclose) all the near-optimal points. For these 4 out of 15 cases our method took 3 times more CPU time on average than BARON. When using LocSol we had to set a reasonable time limit, which is not evident, as both our interval branch-and-method and BARON took times in a wide range when solving these problems. Setting the time limit to 1 second, only 2 out of 15 problems were optimally solved by LocSol. Increasing the time limit to 10 seconds, the number of problems optimally solve increased to 6. As for the last 6 problems with the highest budget, both BARON and our method took longer CPU times, we run LocSol for these problems setting a time limit of 100 and 1000 seconds. LocSol found near optimal solutions in 4 out of the 6 problems in both cases.

In order to check the reliability and performance of BARON, we then solved a larger instance, generated randomly. It has 100 demand points, 3 existing facilities, 2 belonging to the expanding chain. Using our interval method for 5 different budgets, the enclosure of the global optima is reached in 106 seconds on

average, while BARON did not find a near-optimal solution within an hour. For one problem, we let BARON to run until one day, but it could only find a good local solution, not the global optimum.

To conclude, a real-size firm expansion problem either cannot be solved or needs too much time by commercial solvers, thus making it necessary to develop a new method.

1.2.2 An exact interval branch-and-bound algorithm

To exactly solve non-convex multimodal problems when also integer variables are present, global optimization methods are needed. Spatial branch-and-bound methods (see [13]) are among the most used algorithms. They are similar to nonlinear B&B methods used for convex MINLP problems (see for instance Section 3.1 in [12]). However, the relaxed problems in convex MINLP are convex NLP problems which can be solved with local techniques; thus, branching in convex nonlinear B&B is only performed in the integer variables. On the contrary, the relaxation of a non-convex MINLP is a non-convex NLP, being themselves also hard-to-solve problems; thus, branching is also required in the continuous variables as part of the B&B process to solve them.

In this study, we design a B&B method based on interval analysis tools. Unlike real analysis, which works with real numbers, interval analysis works with compact intervals (of real numbers). One of its main advantages is that, through a clever use of its properties, it allows to compute bounds automatically, which is specially useful in the design of B&B methods (see for instance [101, 117, 159]). Interval B&B methods were successfully applied for solving many types of continuous NLP problems, including facility location problems [75, 160, 188, 189]. In this study, we design interval B&B methods to solve MINLP problems. The main idea is to work with the relaxed problem obtained when assuming that the integer variables are also continuous, and then, before rejecting some parts of the searching region, we take care of the integer variables so as not to remove parts which may contain an optimum.

We will adopt the standard notation suggested in [118] for interval analysis throughout the dissertation. Intervals will be denoted by boldface letters, and lower and upper bounds of intervals by 'underlines' and 'overlines', respectively. The width of an interval $\boldsymbol{z} = [\underline{z}, \overline{z}]$ will be denoted by $\operatorname{wid} \boldsymbol{z} = \overline{z} - \underline{z}$, whereas the width of an interval vector $\boldsymbol{z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n)^T$ (also called a *'box'*) is given by $\operatorname{wid} \boldsymbol{z} = \max\{\operatorname{wid} \boldsymbol{z}_i : i = 1, \ldots, n\}$. The midpoint of an interval $\boldsymbol{z}$ will be denoted by $\operatorname{mid} \boldsymbol{z} = (\underline{z} + \overline{z})/2$. $\mathbb{I}$ and $\mathbb{I}^n$ will denote the set of intervals and n-dimensional boxes, respectively.

The main tool used in interval B&B algorithms is that of inclusion function.

Definition 1.2.1. *For a real function, $f : \mathbb{R}^n \to \mathbb{R}$, we call an interval function $\boldsymbol{f} : \mathbb{I}^n \to \mathbb{I}$ an inclusion function, if $\{f(z) : z \in \boldsymbol{z}\} \subseteq \boldsymbol{f}(\boldsymbol{z})$ holds for all intervals $\boldsymbol{z}$ within the domain of f.*

The main benefit of an inclusion function $\boldsymbol{f}$ is that one can get bounds directly over any box in the domain of f. There are programming languages [123] which provide inclusion functions for the classical predeclared functions. And from them, and using the interval arithmetic, inclusion functions for general functions can be built automatically [189] (using, for instance, the natural interval extension [101]). Automatic differentiation [157] also allows to obtain bounds for derivatives, gradients or Hessian matrices automatically.

The introduced MINLP location problem has the additional difficulty that functions $V_j(\alpha_j)$ are defined by an 'if' expression. Still, we can build an inclusion function for V_j and its derivative as follows:

$$\boldsymbol{V}_j(\boldsymbol{\alpha}_j) = \begin{cases} \frac{1}{\delta_j}(\boldsymbol{G}_2(2\tilde{\alpha}_j - \boldsymbol{\alpha}_j) - G_2(\tilde{\alpha}_j)) + v_j & \text{if } \overline{\alpha}_j < \tilde{\alpha}_j \\ \boldsymbol{G}_2(\boldsymbol{\alpha}_j) - G_2(\tilde{\alpha}_j) + v_j & \text{if } \underline{\alpha}_j > \tilde{\alpha}_j \\ [0, \max\{\frac{1}{\delta_j}(G_2(2\tilde{\alpha}_j - \underline{\alpha}_j) - G_2(\tilde{\alpha}_j)), & \\ \qquad G_2(\overline{\alpha}_j) - G_2(\tilde{\alpha}_j)\} + v_j] & \text{if } \underline{\alpha}_j < \tilde{\alpha}_j < \overline{\alpha}_j \end{cases}$$

and

$$\boldsymbol{V}'_j(\boldsymbol{\alpha}_j) = \begin{cases} -\frac{1}{\delta_j}\boldsymbol{G}'_2(2\tilde{\alpha}_j - \boldsymbol{\alpha}_j) & \text{if } \overline{\alpha}_j < \tilde{\alpha}_j \\ \boldsymbol{G}'_2(\boldsymbol{\alpha}_j) & \text{if } \underline{\alpha}_j > \tilde{\alpha}_j \\ (-\infty, +\infty) & \text{if } \underline{\alpha}_j < \tilde{\alpha}_j < \overline{\alpha}_j, \end{cases}$$

where $\boldsymbol{G}_2$ and $\boldsymbol{G}'_2$ are inclusion functions for G_2 and G'_2, respectively (see also [117, 149]).

Next, we describe the main steps of the interval B&B method we have implemented. For a generalized notation, we will denote by z the vector of the n variables, by f the objective function to be maximized, and by $g_j(z) \leq 0, j = 1, \ldots, r$, the constraints of the problem. In our case, z is the vector ns (with dimension $n = 2k + 4$), and constraints (1.16)-(1.20) can be transformed into non-positive form constraints to obtain functions $g_j(z)$.

1.2.2.1 Selection rule

The algorithm manages a list of boxes still to be processed, $\mathcal{L}_{\mathcal{W}}$, which initially consists of a single box containing the feasible region. In a given iteration k, the box to be processed is $\boldsymbol{z}^{(k)} = \arg\max\{\overline{f}(z) : \boldsymbol{z} \in \mathcal{L}_{\mathcal{W}}\}$, i.e., the so-called best-first strategy.

1.2.2.2 Subdivision rule

Let $\boldsymbol{z}$ be the box to be subdivided. Whenever a coordinate direction i is selected to be subdivided, we perpendicularly cut it as follows:

- If the corresponding variable z_i is continuous, then the i-th component of the corresponding subboxes will be $[\underline{z}_i, \operatorname{mid} \boldsymbol{z}_i]$ and $[\operatorname{mid} \boldsymbol{z}_i, \overline{z}_i]$, respectively.
- However, if z_i is integer, then they will be $[\underline{z}_i, \lfloor \operatorname{mid} \boldsymbol{z}_i \rfloor]$ and $[\lceil \operatorname{mid} \boldsymbol{z}_i \rceil, \overline{z}_i]$; notice that if $\operatorname{mid} \boldsymbol{z}_i$ is not integer, the points in the open interval $(\lfloor \operatorname{mid} \boldsymbol{z}_i \rfloor, \lceil \operatorname{mid} \boldsymbol{z}_i \rceil)$ are not integer, and therefore, are not feasible. Hence, in this way, it is assured that the bounds of the integer variables are always integers in all boxes.

The rest of the components of the subboxes do not change when performing this cut. The selection of the coordinate directions to cut are done in two levels. First, we always select from the integer variables, and if there are no integer coordinates to be subdivided, we select from the continuous ones. In the second level, selection is done by the width of the interval, i.e. the widest component is selected to be bisected.

In our location problem all the integer variables are binary; thus, when splitting the box along a binary variable the previous process just fix the binary variable in the subboxes to 0 and 1, respectively.

Furthermore, in our location problem we also do the following procedures:

(1) In the subbox with $y_j = 0$, we also set the variable α_j equal to 0 (the j-th facility is closed or not opened), so its width is 0 and it is no longer selected to be bisected in the subdivision rule; analogously, when $y_0 = 0$ we also set $f_0^1 = 0$ and $f_0^2 = 0$.

(2) The *first time* a variable α_j, $j = 1, \ldots, k$, is selected to perform the subdivision, instead of bisecting the interval $\boldsymbol{\alpha}_j$ by its midpoint, as described above, we perform a trisection by $\tilde{\alpha}_j$, generating the three following subintervals: $[\underline{\alpha}_j, \tilde{\alpha}_j - EPS]$, $[\tilde{\alpha}_j, \tilde{\alpha}_j]$ and $[\tilde{\alpha}_j + EPS, \overline{\alpha}_j]$, where EPS is the smallest machine number the computer can handle. This has proved to be useful because of the V_j functions. Notice this is only done the first time the variable is selected; bisection is used after that.

(3) A box is always subdivided perpendicularly to two coordinate directions. Hence, we always perform a tetrasection, instead of bisection, as it is commonly used in the literature. Whenever at least two integer variables can still be subdivided, we select the first two of them. If only one integer variable is left to be subdivided, we select it, and also the widest continuous variable. If no more integer variables are to be divided, we select the two widest continuous variables for subdivision.

Because of this procedure, we have to take care of many special cases. For instance, when dividing by y_j and α_j, if α_j is selected for the first time, only 4 new subboxes are generated: for $y_j = 1$, α_j is subdivided into 3 pieces, as described above, but for $y_j = 0$ only $\alpha_j = 0$ is set.

1.2.2.3 Discarding tests

Most discarding tests have to be modified in order to deal with integer variables. Here, we will describe these tests with their modifications, if needed. As we have said, we denote by $g_j(z) \leq 0, j = 1, \ldots, r$, the constraints of the problem.

Feasibility test

We say that constraint $g_j(z) \leq 0$ is *certainly continuously satisfied* for an interval box $\boldsymbol{z}$ if $\overline{\boldsymbol{g}}_j(\boldsymbol{z}) \leq 0$, and that it is *certainly continuously unfulfilled* if $\underline{\boldsymbol{g}}_j(\boldsymbol{z}) > 0$. In case $0 \in \boldsymbol{g}_j(\boldsymbol{z})$, we cannot guarantee neither that it is certainly continuously satisfied nor it is certainly continuously unfulfilled, thus the constraint is called undetermined.

We call a box $\boldsymbol{z}$ to be *certainly continuously feasible* if all the constraints $g_j(z) \leq 0, j = 1, \ldots, r$, are certainly continuously satisfied, and *certainly continuously infeasible* when there exists at least one constraint that is certainly continuously unfulfilled. In any other case, when at least one constraint is undetermined, but none of the constraints are certainly continuously unfulfilled, the box $\boldsymbol{z}$ is called *continuously undetermined*.

Moreover, if all the continuous constraints are strictly fulfilled, i.e., $\overline{\boldsymbol{g}}_j(\boldsymbol{z}) < 0, j = 1, \ldots, r$, we call a box $\boldsymbol{z}$ *certainly continuously strictly feasible*.

We can only reject certainly continuously infeasible boxes, thus the feasibility test discards those boxes.

Note that for a *certainly continuously feasible* box $\boldsymbol{z}$, all the continuous constraints $g_j(z) \leq 0, j = 1, \ldots, r$, are satisfied for all points $z \in \boldsymbol{z}$. Therefore, all integer points in $\boldsymbol{z}$ are certainly feasible, such as the corner points of $\boldsymbol{z}$ which are integer points by the subdivision rule. This is specially useful for updating the best value found by the algorithm, used in the cut-off test.

Cut-off test

Let $\tilde{f}$ be the best objective function value obtained so far by the algorithm. Any box $\boldsymbol{z}$ is discarded by the cut-off test provided $\overline{\boldsymbol{f}}(\boldsymbol{z}) < \tilde{f}$. Note that, in order to update $\tilde{f}$, (integer) feasible points are required, and they can be easily obtained from any integer assignment in certainly continuously feasible boxes.

The problem becomes more difficult when the optimal solution is on the boundary of several constraints. The budget constraint is usually binding and, when the new facility is to be opened, usually there is also at least one binding locational constraint. In those cases, it can be hard to find a feasible point, as most boxes near to the optimal solution are continuously undetermined.

Monotonicity test

Monotonicity can help to remove boxes where the global optimum cannot lie. It can be applied to both certainly continuously feasible and undetermined boxes. To derive these rules, let $\boldsymbol{\nabla f}(\boldsymbol{z}) = (\boldsymbol{\nabla}_1\boldsymbol{f}(\boldsymbol{z}), \ldots, \boldsymbol{\nabla}_n\boldsymbol{f}(\boldsymbol{z}))^T$ be an inclusion of the gradient of the objective function f over a box $\boldsymbol{z}$. The monotonicity test can be applied when there exists a variable z_i for which the objective function is monotonous, i.e., $0 \notin \boldsymbol{\nabla}_i\boldsymbol{f}(\boldsymbol{z})$. Let us denote by $\partial\mathcal{S}$ the boundary of the searching region $\mathcal{S}$ (where the constraints $g_j(z) \leq 0$ are not taken into account). In our problem $\mathcal{S} = (S, [0, \alpha_{\max}], [0, 1], \ldots, [0, \alpha_{\max}], [0, 1])$.

In continuous unconstrained problems, the facet $\boldsymbol{z}'$ ($\boldsymbol{z}' = (\boldsymbol{z}_1, \ldots, \underline{\boldsymbol{z}}_i, \ldots, \boldsymbol{z}_n)$ or $\boldsymbol{z}' = (\boldsymbol{z}_1, \ldots, \overline{\boldsymbol{z}}_i, \ldots, \boldsymbol{z}_n)$, depending on whether the function is decreasing of increasing, respectively) of the box $\boldsymbol{z}$ contains the maximum. If $\boldsymbol{z}' \not\subset \partial\mathcal{S}$, there is another continuous box where $\boldsymbol{z}'$ is included, thus the box $\boldsymbol{z}$ can be discarded. If $\boldsymbol{z}' \subset \partial\mathcal{S}$, so $\boldsymbol{z}'$ is on the boundary, the box $\boldsymbol{z}$ is narrowed to $\boldsymbol{z}'$.

However, for constrained problems with integer variables $\boldsymbol{z}'$ has to be checked further. Having a certainly continuously feasible box $\boldsymbol{z}$ with $0 \notin \boldsymbol{\nabla}_i\boldsymbol{f}(\boldsymbol{z})$ for a variable z_i, the following holds:

1. If z_i is continuous, we know that the maximum cannot be in the interior, thus the box $\boldsymbol{z}$ can either be discarded (when it is strictly continuously feasible), or narrowed to its facet $\boldsymbol{z}'$ when $\boldsymbol{z}' \subset \partial\mathcal{S}$.

2. If z_i is integer, the box can be narrowed to the facet $\boldsymbol{z}'$.

   Notice that when the variable z_i is integer, the box cannot be removed since the gradient may change its sign in the open region $(\boldsymbol{z}_1, \ldots, (\underline{\boldsymbol{z}}_i - 1, \underline{\boldsymbol{z}}_i), \ldots, \boldsymbol{z}_n)$ or $(\boldsymbol{z}_1, \ldots, (\overline{\boldsymbol{z}}_i, \overline{\boldsymbol{z}}_i + 1), \ldots, \boldsymbol{z}_n)$, respectively. As in the division rule such regions with non-integer points are removed, we can interpret this as $\boldsymbol{z}'$ is on the boundary of the feasible set.

Now let us consider an undetermined box, $\boldsymbol{z}$. Depending on the constraints which are not certainly continuously satisfied by $\boldsymbol{z}$ the monotonicity test is different. In general, what is important in this case is whether the constraints in which the monotone variable z_i appears are certainly continuously satisfied in the facet $\boldsymbol{z}'$. We call such facet $\boldsymbol{z}'$ to be *feasible for* z_i. Now, if $0 \notin \boldsymbol{\nabla}_i\boldsymbol{f}(\boldsymbol{z})$ for a variable z_i, we can state the following:

1. If z_i is a continuous variable, and the facet $\boldsymbol{z}'$ is feasible for z_i, then the box $\boldsymbol{z}$ can be either discarded (if $\boldsymbol{z}' \not\subset \partial\mathcal{S}$) or reduced to $\boldsymbol{z}'$ (if $\boldsymbol{z}' \subset \partial\mathcal{S}$).

2. If z_i is an integer variable, and the facet z' is feasible for z_i, then the box can be narrowed to the facet z'.

Specifically for our location problem, as the variables $y_j, \alpha_j, j = 0, \ldots, k$ do not appear in the constraints delimiting the feasible area for the location of the new facility, it is enough to check whether the corresponding boxes certainly continuously satisfy the budget constraint. However, for the location variables f_0^1, f_0^2, all the constraints have to be checked as they appear in all the constraints.

In our location problem, the budget constraint is the one that provokes more continuously undetermined boxes, as it is usually binding in the optimal solution. When the budget is small, many boxes can be discarded by the feasibility test, and when it is high, it the monotonicity test is the one that becomes more efficient.

1.2.2.4 Projected one-dimensional Newton method

In [76], a one-dimensional Newton method was introduced, which can be applied here on both certainly continuously strictly feasible or undetermined boxes, on their continuous quality variables.

If the box is continuously strictly feasible, we can apply the method without any changes. The main idea is to run the interval Newton method considering only one variable at a time, fixing the other variables to their current interval value. Notice that in contrast to the classical interval Newton method, which is usually performed as a one-step (or one-iteration) method, the projected Newton method is run until it cannot remove any part of the box, or until it discards the box (as no local optimum is included in it) or until the width of the given variable is less than the tolerance in our stopping criterion.

Now let us consider an undetermined box, $\boldsymbol{z}$. We can also apply the projected one-dimensional Newton method to a variable z_i of $\boldsymbol{z}$ provided that z_i does not appear in any of the constraints which are not certainly continuously strictly satisfied by $\boldsymbol{z}$.

We have also employed the empirical rule of applying the method only to boxes whose width is less than one.

In the introduced location problem, if the box $\boldsymbol{z}$ is undetermined for some locational constraints, but strictly feasible for the budget constraint, the test can be applied to the quality variables (the objective function is concave in those variables [76]).

Projected one-dimensional non-concavity test

Similarly to the one-dimensional Newton method, one can check the non-concavity of the objective function only for a given variable, considering the other variables fixed at their current interval values. The main advantage in both projected methods is that instead of computing the whole Hessian matrix, only one second derivative is required to be computed.

Again, we can only apply the non-concavity test to a continuous variable z_i provided that z_i does not appear in any of the constraints which are not certainly continuously strictly satisfied by $\boldsymbol{z}$.

1.2.2.5 Stopping rule

We have used two stopping criteria for any of the boxes. A box $\boldsymbol{z}$ is sent to the solution list $\mathcal{L}_S$ if either its size is smaller than a given tolerance, i.e., $\text{wid}\,(\boldsymbol{z}) < \varepsilon$, or the relative width of the inclusion of its objective value is less than a given tolerance, i.e, $\text{wid}_{rel}(\boldsymbol{f}(\boldsymbol{z})) < \delta$.

1.2.3 Computational studies

1.2.3.1 Hardware and software

All the computational results have been obtained under Linux on an AMD Athlon(tm) 64 X2 with 2.2 GHz CPU and 2 GB memory. The algorithms have been implemented in C++. For the interval B&B method, we used the interval arithmetic in the PROFIL/BIAS library [119], and the automatic differentiation of the C++ Toolbox library [99]. We have used a time limit of 12 hours for each run.

1.2.3.2 Test problems

In order to check the performance of the methods and their variants, we have generated different test problems varying some parameters. Namely, we have fixed the number of demand points ($n = 100$), but varied the number of existing facilities ($m = 3, 5$) and the number of facilities belonging to the expanding

chain ($k = 1, 2$). For each setting, 10 problems were generated, by choosing the input data of the problems randomly from the following intervals using uniform distributions:

- location of demand points and existing facilities: $p_i, f_j \in ([0, 10], [0, 10])$,
- demand: $w_i \in (0, 10]$,
- quality of existing facilities: $\tilde{\alpha}_j \in [0.5, 5]$,
- income per unit of goods sold ($F(\cdot) = c \cdot M(\cdot)$): $c \in [12, 14]$,
- parameters of function Φ_i: $\phi_{i0} = 2, \phi_{i1} \in [0.5, 1.5]$,
- parameters of function G_2: $\beta_0 \in [7, 9], \beta_1 \in [5, 5.5]$,
- radius of forbidden regions: $d_i^{\min} = w_i/\rho$, where $\rho = 3\sqrt{n + 50} - 20$,
- parameter of function R_j: $o_j = 20$,
- annualized cost for keeping a facility opened: $A_j \in [8, 11]$,
- cost for closing a facility: $C_j = A_j/2$,
- parameters of function V_j: $v_j = G_2(1)$ and $\delta_j \in [3, 5]$.

After generating the instances, we found that the original qualities remain in none of them unchanged when solving the problem. This is because the qualities associated to the facilities were far from being the optimal ones, due to their randomness. Thus, in order to generate problems which are closer to reality, we performed a pre-optimization for the qualities of the existing facilities, as follows. First, we calculated the budget needed to run the facilities of the expanding chain with their current setting, $\tilde{B} = T(ns)$ (i.e., when no new facility was taken into account ($y_0 = 0$) and when the existing facilities were open with their original qualities, i.e., $y_j = 1$ and $\alpha_j = \tilde{\alpha}_j \; \forall j = 1, \ldots k$). Second, we run our B&B method for the budget $\tilde{B}$ in order to optimize the qualities of the existing facilities, by fixing $y_0 = 0$ and $y_j = 1 \; \forall j = 1, \ldots k$. Finally, we set $\tilde{\alpha}_j$ equal to the optimal value of α_j of the previous problem.

Furthermore, we also used $\tilde{B}$ to set reasonable budgets for the generated instances. In particular, we solved each problem instance for 80%, 100%, 120%, 150%, and 200% of the budget $\tilde{B}$, that is, from a scenario in which the chain does not have enough budget to keep all its facilities with their current qualities, to a scenario where the budget of the chain doubles the one required for the current setting. Hence, in all $2 \cdot 2 \cdot 10 \cdot 5 = 200$ instances were generated.

1.2.3.3 Versions of the interval branch-and-bound method

In order to analyse the efficiency of each of the discarding tests, we solved the test problems using the following versions of the interval B&B algorithm:

Basic: Only feasibility and cut-off tests are applied as discarding tests, and only the natural inclusion is evaluated for bounding.

Mono: The discarding tests of the Basic version together with the monotonicity test are applied. For bounding, the intersection of the centered form and natural inclusion [101] is used.

Newton: This is the Mono version together with the projected one-dimensional Newton method.

NonConc: This is the Newton version together with the projected one-dimensional non-concavity test.

In Table 1.5, we can compare the different versions of the interval B&B method for the different problem settings. As the main indicator of efficiency, the table gives the average computational time in seconds for the 10 problems in each setting. For those settings, where the given method could not finish before its time limit, the 12 hours were taken into account (there were 39 such cases for Basic, 4 for Mono, and only 2 for Newton and Nonconc out of the 200 scenarios).

In the first three columns of Table 1.5, the type of the problems is shown: first the number of existing facilities and the number of facilities belonging to the expanding chain is given, and then, the budget in relative terms to $\tilde{B}$, the budget needed to run the facilities of the expanding chain with their current setting. In the following columns, for each version of the interval method (Basic, Mono, Newton, and NonConc) the average computational time in seconds is reported. Moreover, we report in percentages the ratio between each version with its preceding variant, in the columns named $\frac{\text{Mono}}{\text{Basic}}$, $\frac{\text{Newt}}{\text{Mono}}$, and $\frac{\text{NonConc}}{\text{Newt}}$.

Table 1.5: Average computational time in seconds for the different versions of the interval B&B method for problems with 100 demand points

m	k	Budget	Basic	Mono	$\frac{\text{Mono}}{\text{Basic}}$	Newton	$\frac{\text{Newt}}{\text{Mono}}$	NonConc	$\frac{\text{NonConc}}{\text{Newt}}$
3	1	80%	8437	3307	39%	3293	100%	3296	100%
		100%	17732	4384	25%	4388	100%	4389	100%
		120%	17570	1928	11%	1928	100%	1922	100%
		150%	7210	177	2%	174	98%	176	101%
		200%	7117	187	3%	153	81%	159	104%
		Aver.	**11613**	**1997**	**17%**	**1987**	**100%**	**1989**	**100%**
	2	80%	7105	716	10%	714	100%	716	100%
		100%	14124	2331	17%	2314	99%	2314	100%
		120%	13804	4799	35%	4755	99%	4760	100%
		150%	14569	5925	41%	385	6%	424	110%
		200%	19022	5926	31%	307	5%	354	115%
		Aver.	**13725**	**3939**	**29%**	**1695**	**43%**	**1713**	**101%**
5	1	80%	3320	339	10%	339	100%	338	100%
		100%	8544	335	4%	333	99%	333	100%
		120%	11523	199	2%	196	98%	198	101%
		150%	10131	100	1%	92	92%	94	102%
		200%	13284	99	1%	47	48%	54	115%
		Aver.	**9360**	**214**	**2%**	**201**	**94%**	**203**	**101%**
	2	80%	11091	5230	47%	5211	100%	5230	100%
		100%	17834	9212	52%	7178	78%	7167	100%
		120%	12466	6687	54%	4162	62%	4181	100%
		150%	13877	6665	48%	3616	54%	3675	102%
		200%	15749	6758	43%	348	5%	388	112%
		Aver.	**14203**	**6911**	**49%**	**4103**	**59%**	**4128**	**101%**

Additionally, for each problem setting we report the averages for the different budget scenarios taken by the versions of the interval B&B method.

As we can see, for the Basic version of the interval method, the smallest budget always leads to easier-to-solve problems, as the feasibility test can efficiently discard large regions from the infeasible region. Increasing the budget usually increases the time, although for some (m, k) settings we can see a decrease of time after the 120% budget.

The use of the monotonicity test with the centered form always makes the interval B&B method faster. For some settings, the improvement of Mono over Basic is dramatic, being more than 10 times faster. When only one chain-owned facility exists ($k = 1$), the tendency is clear: the more budget we have, the more efficient the monotonicity test is, probably because there are more boxes which certainly satisfy the budget constraint, and to which the monotonicity test can be applied. In case two chain-owned facilities exist, there is no such tendency, but still, Mono is very efficient, two times faster than Basic.

The projected one-dimensional Newton method is also efficient in general, although not always. For the lowest budget, it cannot usually be applied, as most of the boxes are undetermined for the budget constraint; that is why for those problems it does not reduce the CPU time. The higher the budget, the more efficient the Newton method is. And it is particularly efficient when two chain-owned facilities exist (when the monotonicity test is less efficient): in those cases, when 150% and 200% of the budget $\tilde{B}$ is available, the Newton version is 20 times faster than Mono and it employs just 5% of the running time of the Mono version. Notice that in those settings there are three quality variables for which the projected one-dimensional Newton method can be applied.

On the other hand, the projected one-dimensional non-concavity test is not efficient for our location problem (only for three out of the 20 settings we can see a small improvement). For smaller budgets, it cannot usually be applied, as most of the boxes are undetermined for the budget constraint. For higher budgets, its use provoked an increase in the CPU time, mainly because both the monotonicity and the projected one-dimensional Newton are very efficient, and there is not much room for improvement.

Comparing the problem settings, the easiest problems are, in general, the ones with one chain-owned facility. The reason is clear: those problems have one variable less than the others. Interestingly, among

Table 1.6: Average computational times for each budget settings using the different versions of the interval B&B method

Budget	Basic	Mono	$\frac{\text{Mono}}{\text{Basic}}$	Newton	$\frac{\text{Newt}}{\text{Mono}}$	NonConc	$\frac{\text{NonConc}}{\text{Newt}}$
80%	7488	2398	32%	2389	100%	2395	100%
100%	14559	4065	28%	3553	87%	3551	100%
120%	13841	3403	25%	2760	81%	2765	100%
150%	11447	3217	28%	1067	33%	1092	102%
200%	13793	3242	24%	214	7%	239	112%
Aver.	**12225**	**3265**	**27%**	**1997**	**61%**	**2008**	**101%**

those problems, the ones with 5 existing facilities are easier to solve; however, it is not clear if there is a reason for that. On the contrary, when there are two chain-owned facilities, it turns out that, in general, the problems with 3 existing facilities are easier to solve.

In Table 1.6, we have reported the average results of all the problems for each budget scenario comparing the different versions of the interval B&B method. As we can see, the budget needed to run the facilities of the expanding chain with their current setting, $\tilde{B}$, gives in general the hardest problem for each variant. This is because of the way we generated the random problems: remember that we performed a pre-optimization of the qualities of the existing facilities, so it is more difficult to find a better solution. For the Basic and Mono versions, the easiest problems on average are those whose budget is more restrictive; on the contrary, the Newton and NonConc variants are more effective as the budget constraint becomes looser. On average, the Mono version needs just one third of the running time of Basic. The Newton version also reduces the computational time, specially when the budget constraint is looser. In general, compared to the Mono version, it needs 61% of its CPU time, and compared to the Basic method, only 16% on average.

1.2.4 Conclusions

When a chain wants to expand its presence in a given geographical area, it can choose among opening a new facility, varying the quality of its existing facilities up or down, closing some of its existing facilities (to invest the money devoted to those facilities to the other chain-owned facilities, or to the new one, in case it is open) or a combination of all those possibilities. In this work, the continuous competitive facility location and design model introduced in [76] has been extended to accommodate all those possibilities.

The resulting model is a hard-to-solve MINLP problem, for which the existing solvers fail: only BARON [171] and LocSol [14] seem to be able to solve some small instances, although they can fail at solving other instances, too. Thus, new algorithms are required to cope with the new problem.

An exact spatial interval branch-and-bound method is proposed to solve the problem. It is a modification of the classical interval B&B methods proposed for continuous NLP problems to handle integer variables. In particular, the subdivision rule has been modified to take into account the integer variables, as well as the monotonicity test, the projected one-dimensional Newton method and the projected one-dimensional non-concavity test. The idea for the modification of those tests is to relax the integrality of the variables, and when the moment of discarding subregions arrives, to take the integrality back into account so as not to remove promising parts.

The so-called Newton variant of the exact interval B&B algorithm can solve medium size problems (with up to 100 demand points and 5 existing facilities) without difficulty in around 30 minutes of CPU time on average. The method is guaranteed to find *all* optimal solutions.

1.3 Location equilibria for a continuous competitive facility location problem under delivered pricing

The problem of selecting the optimal or *best* location of the facilities and the setting of the *right* price of a product are two of the main drivers in a supply chain. These two interrelated logistical drivers provide a competitive advantage while contributing significantly to both the efficiency and the responsiveness of a supply chain. In other words, where to locate the facilities and what price to set are major decisions for

firms that sell the same type of product and compete for customers. The profit each firm gets is affected not only by the location of its facilities and the price that the firm sets in the market, but also by the location of its competitors and the prices they set. The maximization of profit for each competing firm can be seen as a location-price game, which has been studied since the work of Hotelling [112]. Most existing literature deals with a linear market (see [52, 86, 145]), which is in part due to the complexity of solving the associated location problems in other location spaces such as the plane or the network. The reader will find the survey papers [71, 152, 167] to be useful.

This study deals with the competitive location problem on the plane where two firms aim to locate optimally one facility each while maximizing their market share. A branch-and-bound (B&B) approach based on interval analysis is developed for small/medium size problems. A new iterative Weiszfeld-like formula is developed and used as part of an adapted version of an 'allocate-locate-allocate' scheme commonly used in solving multi-source Weber problems. A proof for validating the descent property of our approach is also provided.

Most models in this context consider a refinement of the Nash equilibrium by using a two-stage process where the choice of the location is usually prior to the decision on the price. In the first stage, firms solve the location problem and given the outcome of the first stage, they then choose the price of the product in the second stage. The corresponding two-stage solution is called a subgame perfect Nash equilibrium. It captures the idea that when firms select their location, they all anticipate the consequences of their choice on the price. The division into two stages is motivated by the fact that the location decision usually requires a massive investment. This is a strategic decision which cannot be easily altered. On the other hand, the price decision is a short term or/and operational type decision which can be adjusted from one week to the next without too much inconvenience. Note, however, that the timing for altering such prices is usually planned earlier so to provide the other activities within the supply chain enough time to accommodate the appropriate changes. This two-stage process is also adopted in this work. Note that in this context, firms are supposed to decide simultaneously the location and the prices. This is in contrast to Stackelberg competition models, where a firm (the leader) decides first the location and then the prices for its facilities, taking into account that the other firm (the follower) will react by subsequently locating and selecting prices for its own facilities, knowing the previous decision of the leader (see [96, 163]).

The existence of a price equilibrium in the *second stage* of the game depends, among other factors, on the price policy adopted. When each firm sets a fixed factory price and the buyer takes care of the carriage (f.o.b. or *mill pricing* policy) a price equilibrium rarely exists (see [89]). In this case, the associated location problem has been studied in nonlinear location spaces by taking prices as parameters (see [69, 90, 177, 198]). On the other hand, there always exists a price equilibrium when each firm charges a specific price in each market area, which includes the transport costs (*delivered pricing* policy). The existence of a price equilibrium was shown for the first time by Hoover [109]. The author analysed spatial discriminatory pricing for firms with fixed locations. It was concluded that a firm serving a particular market would be constrained in its local price by the delivery cost of the other firms serving in that market. In situations where demand elasticity is 'not too high', the equilibrium price at a given market is equal to the delivery cost of the firm with the second lowest delivery cost. This result was extended later to spatial duopoly (see [125, 126]) and to spatial oligopoly (see [57, 89]) for different types of location spaces.

In a duopoly with completely inelastic demand and constant marginal production costs, Lederer and Thisse [126] show that a location equilibrium exists and that any global minimizer of the *social cost* is an equilibrium. This is the total delivered cost if each customer was served with the lowest marginal delivered cost. In oligopoly, the same result is obtained by Dorta *et al.* [57]. Here, the authors present a model where firms take location and delivery price decisions along a network of connected but spatially separated markets. Under reasonable assumptions, they show that a location equilibrium can be found at the nodes by using global minimization of the social cost. But, to the best of our knowledge, no procedure has been proposed along this direction. Gupta [95] and Hamilton *et al.* [98] also showed that if the demand is price sensitive or the marginal production costs are not constant, the optimal locations that are based on the social cost may not coincide with an equilibrium of the location-price game.

In the present study, we assume that markets are aggregated at n demand points, where a given homogeneous price-inelastic product will be sold by the competing firms. The firms manufacture and deliver the product to the customers, who always opt to buy from the firm that offers the lowest price. This is the case for instance of firms which have to serve clients who are within a distance such that the transportation costs are similar or higher than the production cost of the product to be delivered.

In particular, we consider two firms locating a single facility each on the continuous space with the presence of constant marginal production costs. In this game, the number of available actions for each player (firm) is infinitely many (as the location of the facilities vary in (a subset of) the plane and the prices

may also be chosen continuously). The payoff function for a firm is the profit it gets, which is determined by the segment of the market served by each firm. This market depends on the prices, which in turn depend on the location of the facilities, as it will be shown. This work could also be used as a basis for the case when each firm wishes to have more than one facility.

This equilibrium location problem reduces to the multi-source Weber problem if the production costs are set to zero and the marginal transportation costs are given by the distances between the facilities and the customers. In this case, the social cost reduces to the minimisation of the transportation cost only. A review of such continuous multi-source Weber problems is given in Brimberg *et al.* [26]. Recent results in this particular area can be found in Schöbel and Scholz [175], and Brimberg and Drezner [25]. The former paper extends on the idea of the Big Square Small Square procedure originally proposed by Hansen *et al.*[102] to higher dimensions. The authors present a new approach known as The Big Cube Small Cube to optimally solve problems with $p = 2$ and 3. The latter paper presents an efficient implementation to improve on Cooper's alternate approach [46] and produces excellent results for most values of p when tested on the commonly used multi-source Weber data sets (see Brimberg *et al.*[27]). In this study, we are not limiting ourselves to this well studied location problem, but we aim to solve the problem in the general case when the social cost is not represented by the transportation cost only.

The section is organized as follows: In the next subsection we introduce some reduction schemes to transform the problem into a location game and present a reformulation of the problem. These resulted from a common work with my coauthors, and as such are indivisible. Subsection 1.3.2 discusses the interval B&B algorithm which can solve instances with any cost structure, but only of a moderate size. This was my main contribution done together with José Fernández. Subsection 1.3.3 addresses the case of larger instances where an effective alternating Weiszfeld-based algorithm is designed to cater for a general cost structure including the multi-source Weber problem as a special case. The idea came from Said Salhi worked out by myself with José Fernández, while the theoretical result is mine. Subsection 1.3.4 describes some experiments and presents our findings. In Subsection 1.2.4, our main conclusions are summarised.

1.3.1 A reformulation as an optimization location game problem

In this subsection, we first show that a price equilibrium exists, and then, using the equilibrium prices, we reduce the location-price game to a location game.

We have n customers with respective demand w_i $(i = 1, ..., n)$ that need to be served from two competing firms ($u = 1, 2$). In this study, each firm u can locate its facilities anywhere in the plane. However, to relate to recent developments in this field (see [148]), we first refer to the general case where the search space is an arbitrary location space L^u. L^u may be a discrete set, a network, or a region of the plane. Let us denote by

p^u_{xi}	the marginal delivered cost of firm u from location x to market i. This includes both transportation and production costs.
$p^u_i(X^u)$	$= \min\{p^u_{xi} : x \in X^u\}$, minimum price that firm u can offer in market i when $X^u \subset L^u$ is chosen as the set of facility locations for firm u, $u = 1, 2$.

We also assume that each firm offers a price which is either the same or above its marginal delivered cost to guarantee a profit. If the two firms offer the same price in market i, the one with the minimum marginal delivered cost can lower its price and obtains all the demand in market i. Then we consider that ties in price are broken in favour of the firm with lower marginal delivered cost. If the tied firms have the same marginal delivered cost in market i, no tie breaking rule is needed to share demand because they will obtain zero profit from market i as a result of price competition. It can be shown that a price equilibrium would be obtained as neither of the two firms, for their own competitive advantage, will attempt to change its price. This interesting results is shown in Pelegrín *et al.* [148]. In other words, once X^1 and X^2 are fixed or determined, the equilibrium prices can be defined mathematically as follows:

$$\overline{p}^1_i(X^1, X^2) = \begin{cases} p^2_i(X^2) & \text{if } p^1_i(X^1) < p^2_i(X^2) \\ p^1_i(X^1) & \text{otherwise} \end{cases}$$

$$\overline{p}^2_i(X^1, X^2) = \begin{cases} p^1_i(X^1) & \text{if } p^2_i(X^2) < p^1_i(X^1) \\ p^2_i(X^2) & \text{otherwise.} \end{cases}$$

Similarly, the market share attracted by each firm u (or the segment of the market served by firm u) is

defined as

$$\begin{aligned} M^1(X^1, X^2) &= \left\{ i \in \{1, ..., n\} : p_i^1(X^1) < p_i^2(X^2) \right\} \\ M^2(X^1, X^2) &= \left\{ i \in \{1, ..., n\} : p_i^2(X^2) < p_i^1(X^1) \right\}. \end{aligned}$$

Other markets may be served by any of the firms or by both firms, but the profit coming from these markets is 0. Their corresponding profit is computed as

$$\begin{aligned} \Pi^1(X^1, X^2) &= \sum_{i \in M^1(X^1, X^2)} \left(p_i^2(X^2) - p_i^1(X^1) \right) w_i \\ \Pi^2(X^1, X^2) &= \sum_{i \in M^2(X^1, X^2)} \left(p_i^1(X^1) - p_i^2(X^2) \right) w_i. \end{aligned}$$

In brief, the location-price game is reduced to a location game where the decisions are about the choice of the location and the payoff for player u is $\Pi^u(X^1, X^2), u = 1, 2$.

We would like to emphasize that our aim in this study is not to optimize Π^1 or Π^2 but to find a location equilibrium instead. Facilities $(\tilde{X}^1, \tilde{X}^2)$ satisfy a location equilibrium property if the following two conditions are both satisfied:

$$\Pi^1(\tilde{X}_1, \tilde{X}_2) \geq \Pi^1(X^1, \tilde{X}^2) \ \forall X^1 \ \text{ and } \ \Pi^2(\tilde{X}_1, \tilde{X}_2) \geq \Pi^2(\tilde{X}^1, X^2) \ \forall X^2.$$

As pointed out in [148], any global optimal solution of the following minimization problem is then an equilibrium:

$$\min \left\{ S(X^1, X^2) : X^1 \subset L^1, X^2 \subset L^2 \right\}, \tag{P}$$

where

$$S(X^1, X^2) = \sum_{i=1}^{n} \min \left\{ p_i^1(X^1), p_i^2(X^2) \right\} w_i$$

is known as the *social cost*. Note, however, that the converse does not hold in general. In addition, local minimizers of the social cost lead to *local* equilibria only. In other words, there exist equilibria in the same neighbourhood in which they are local minimizers.

When $L^u, u = 1, 2$, is part of a finite set of potential sites, the problem is thoroughly investigated in [148]. However, when $L^u, u = 1, 2$ is anywhere in the continuous space, problem (P) becomes relatively harder to solve than its counterpart the discrete case. In this study, we wish to address this planar competitive location problem using complex and novel global optimization techniques.

In particular, we concentrate on the case where $L^u \subseteq \mathbb{R}^2, u = 1, 2$ and $|X^u| = 1, u = 1, 2$. For simplicity we use x^u to denote the location of the facility chosen by firm u, i.e., $X^u = \{x^u\}$. As the marginal delivered cost can be written as $p_{xi}^u = c_x^u + t_{xi}^u$, with c_x^u and t_{xi}^u representing the marginal production and transportation costs, respectively, (P) can be rewritten as

$$\min \left\{ \sum_{i=1}^{n} \min \left\{ c_{x^1}^1 + t_{x^1 i}^1, c_{x^2}^2 + t_{x^2 i}^2 \right\} w_i : x^1 \in L^1, x^2 \in L^2 \right\}. \tag{P'}$$

In this work, our aim is to study Problem (P'). To highlight the difficulty in solving (P'), let us consider the particular case in which $c_x^1 = c_x^2$ for all $x \in \mathbb{R}^2$, so we can state that $p_{xi}^u = t_{xi}^u$. Furthermore, assume that the marginal transportation cost of firm u from location x^u to market i is a linear function of the distance $d_i(x^u)$ from market i to x^u, so that we can assume that $p_{xi}^u = d_i(x^u)$. Since $|X^u| = 1$ we also have $p_i^u(x^u) = p_{x^u i}^u = d_i(x^u)$. Using these assumptions, the social cost can be defined as

$$S(x^1, x^2) = \sum_{i=1}^{n} \min\{d_i(x^1), d_i(x^2)\} w_i. \tag{1.22}$$

In this case, (P') reduces to the well known uncapacitated multi-source Weber problem which is known to be NP-hard (see Megiddo and Supowit [132]).

Since the objective function of problem (P') is neither convex nor concave, global optimization techniques need to be used. An interval branch-and-bound algorithm is presented in the next section followed by our newly constructed alternating Weiszfeld-like heuristic which we will describe in Subsection 1.3.3.

1.3.2 A branch-and-bound based approach

As we are already familiar with the general interval B&B method, we only provide information on extraordinary parts, followed by the rules used in our iBB implementation. Two examples with different characteristics are also given to illustrate the method.

One of the challenges in our problem is that the objective function is given as a sum of functions which in turn are given as a minimum of functions. When a function is defined by a minimum some modifications are required to obtain the corresponding inclusion function. If $\boldsymbol{f}, \boldsymbol{g} : \mathbb{I}^n \to \mathbb{I}$ are inclusion functions for $f, g : \mathbb{R}^n \to \mathbb{R}$, respectively, we can obtain an inclusion function for $\min\{f, g\}$ as

$$\mathbf{min}(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x})) = [\min\{\underline{\boldsymbol{f}}(\boldsymbol{x}), \underline{\boldsymbol{g}}(\boldsymbol{x})\}, \min\{\overline{\boldsymbol{f}}(\boldsymbol{x}), \overline{\boldsymbol{g}}(\boldsymbol{x})\}]. \tag{1.23}$$

We can also obtain an inclusion function for its subgradient, as follows

$$\boldsymbol{\nabla}\mathbf{min}(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x})) = \begin{cases} \boldsymbol{\nabla f}(\boldsymbol{x}) & \text{if } \overline{\boldsymbol{f}}(\boldsymbol{x}) < \underline{\boldsymbol{g}}(\boldsymbol{x}) \\ \boldsymbol{\nabla g}(\boldsymbol{x}) & \text{if } \overline{\boldsymbol{g}}(\boldsymbol{x}) < \underline{\boldsymbol{f}}(\boldsymbol{x}) \\ \boldsymbol{\nabla f}(\boldsymbol{x}) \uplus \boldsymbol{\nabla g}(\boldsymbol{x}) & \text{otherwise,} \end{cases} \tag{1.24}$$

where $\uplus$ stands for the interval hull (the smallest box containing the union of the boxes). $\boldsymbol{\nabla f}(\boldsymbol{x})$ and $\boldsymbol{\nabla g}(\boldsymbol{x})$ are inclusion functions for the gradients ∇f and ∇g of f and g, respectively, which can be obtained by automatic differentiation [99, 157].

1.3.2.1 Our iBB implementation

The particular rules that we select for our iBB have shown to be promising in related work (see [149] and [76]). These include the following:

Selection Rule: We select the box $\boldsymbol{y}$ with $\underline{\boldsymbol{S}}(\boldsymbol{y}) = \min\{\underline{\boldsymbol{S}}(\boldsymbol{z}) : \boldsymbol{z} \in \mathcal{L}_{\mathcal{W}}\}$, where $\mathcal{L}_{\mathcal{W}}$ denotes the list of boxes still to be processed. This is also known as the best-first strategy.

Division Rule: Our aim is to opt for the bisection method which divides the box into two equally sized boxes. This simple but effective numerical method has also the advantage of acquiring well defined convergence properties. Here, we cut the box perpendicularly to the coordinate with the maximum width as reducing the distance between far away points can lead to tighter bounds.

Discarding Tests: We have used the classical *feasibility* and *cut-off (or mid-point) tests* [159], the *pruning test* proposed in [130], and the *monotonicity test (for strictly feasible and undetermined boxes)* recently introduced in [76]. Here, the inclusion of the subgradient $\boldsymbol{\nabla S}$ is obtained by automatic differentiation [99, 157] and using (1.23) and (1.24).

Termination Rule: If $\mathrm{w}(\boldsymbol{y}) < \varepsilon$, the box $\boldsymbol{y}$ is sent to the solution list, $\mathcal{L}_{\mathcal{S}}$.

Both the selection and division rules are the ones commonly used in interval B&B methods. No attempt has been made to investigate the suitability of other strategies. As for the discarding test, they have been included because they all proved to be effective at accelerating the algorithm when applied to the problem at hand.

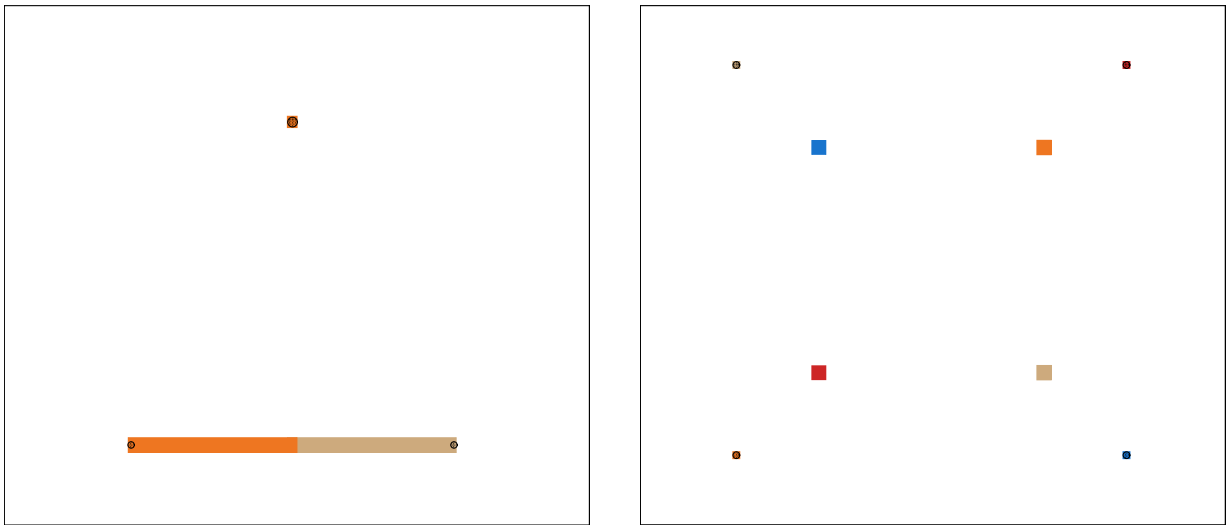
The interval B&B algorithm terminates after a finite number of iterations. A list of boxes, $\mathcal{L}_{\mathcal{S}}$, which contains *all* optimal solutions is produced. Note that any vector in the boxes from this list has a value close to the global optimum. The region of near-optimality given by $\cup_{\boldsymbol{y} \in \mathcal{L}_{\mathcal{S}}} \boldsymbol{y}$ can be useful in practice. This provides flexibility to the decision maker in choosing several solutions instead of being limited to having one single solution only. In this situation, he/she will be able to choose a final solution that incorporates other aspects that would have been difficult to explicitly quantify otherwise.

1.3.2.2 Two illustrative examples

Figure 1.6a) shows an example of the non-symmetric solutions found by the interval B&B method for a problem with three equidistant points, each with unit demand, and the simplifying assumptions that $p^u_{x^u i} = d_i(x^u)$. This simplified problem reduces to an uncapacitated multisource Weber problem. The problem has infinite optimal solutions. In orange (darker colour), we can see the optimal locations for the facility of firm 1, and in brown (lighter colour) the optimal locations for the facility of firm 2, when a lexicographical order is required. As we can infer from Figure 1.6(a), the optimal solution for firm 1 is

to locate its facility either in one of the demand points (and then firm 2 locates its facility at any point of the opposite edge of the triangle) or in an edge of the triangle (and then firm 2 locates its facility at the opposite vertex).

A second example is given in Figure 1.6(b), where a problem with four demand points located in the corners of a square is presented. The same assumptions as the ones of the previous example are considered here. The locations of firms 1 and 2 are ordered lexicographically. In this figure, four of the eight optimal locations for the facility of firm 1 can be identified. For instance, the facility can be located (i) either at one of the vertices, or (ii) as the solution of the Weber problem generated by three of the vertices. In (i) the facility of firm 2 is to be located as the solution of the corresponding Weber problem generated by the other three demand points whereas in (ii) the facility of firm 2 is located at the other vertex of the square. Each solution, which is a pair of locations, is shown in a different colour in Figure 1.6(b).



(a) (b)

Figure 1.6: Output of the interval B&B method for a problem with three demand points on the left, with four demand points on the right

The above method is suitable for small to medium size problems only as will be shown empirically in our computational results, see Subsection 1.3.4. For larger problems, a heuristic-based approach would be the best way forward. This is presented next.

1.3.3 The heuristic-based approach

We first propose an alternating Weiszfeld-like algorithm for the easier case where the problem (P') reduces to the multisource Weber problem (the minimization of (1.22)). The algorithm is based on the 'allocate-locate-allocate' methodology originally proposed by Cooper in [46] for the multi-source Weber problem. A proof to validate the descent property of such an approach is also provided. This technique is then extended to cater for a general cost structure.

1.3.3.1 The alternating Weiszfeld-like location algorithm

In this subsection, we solve the problem of minimizing (1.22) by proposing an alternating Weiszfeld location algorithm which we refer to as AWLA for short (see Algorithm 1.1). At each iteration, only the customers from which the corresponding firm gets a positive profit are considered (see steps 3 and 5). Once this allocation of the demand points to a facility is completed, the facility is relocated by solving the corresponding 1-median problem using steps 4 and 6 of Algorithm 1. These steps are implemented based on the idea of Weiszfeld [194] but using the new iterative formula which we developed, see equations (1.26) and (1.27). This process is repeated until the change in the location of the facilities is smaller than a given tolerance. This algorithm improves the objective function value from one iteration to the next until a local minimum is found and hence the search terminates. The proof for the validity of the descent property of these formula is provided at the end of the subsection.

Algorithm 1.1 AWLA

1: Define the two initial points x^1 and x^2 and set STOP=false.
2: **repeat**
3: Compute the set $M^1(x^1, x^2) = \{i \in \{1, ..., n\} : d_i(x^1) < d_i(x^2)\}$.
4: Apply the Weiszfeld like algorithm (using (1.26) and (1.27)) to solve the 1-median problem for firm 1 based on the set $M^1(x^1, x^2)$. Consider x^1 as initial point and let $\hat{x}^1$ be the new obtained location.
5: Compute the set $M^2(\hat{x}^1, x^2) = \{i \in \{1, ..., n\} : d_i(x^2) < d_i(\hat{x}^1)\}$.
6: Apply the Weiszfeld like algorithm (using (1.26) and (1.27)) to solve the 1-median problem for firm 2 based on the set $M^2(\hat{x}^1, x^2)$. Consider x^2 as initial point and let $\hat{x}^2$ be the new obtained location.
7: **if** $\max(\|\hat{x}^1 - x^1\|, \|\hat{x}^2 - x^2\|) < \varepsilon$
8: set STOP=true
9: **else**
10: set $x^1 = \hat{x}^1$ and $x^2 = \hat{x}^2$
11: **until** STOP

There are several strategies on how to select the initial locations for the facilities in Step 1. The simplest way which we implement in this work is to generate x^1 and x^2 randomly from the area of interest. However, other rules could include the following:

(i) Set x^1 equal to the median point (based on the set of all demand points) and x^2 equal to the demand point with maximum demand.

(ii) Set x^1 and x^2 as the optimal solution of the discrete 2-median problem. This can be achieved by carrying out a complete enumeration which can be performed relatively efficiently for the case of 2 facilities.

(iii) A simple guided multi-start: For instance, we can generate several pairs of locations randomly and choose a subset of the best pairs which can make up the set of our starting points from which we can identify the most appropriate x^1 and x^2.

1.3.3.2 A general cost structure

A similar strategy can be adopted for handling the general case, when $c_x^1 \neq c_x^2$ and/or $t_{xi}^1 \neq t_{xi}^2$. The only difference is that the objective function of the 1-median problems to be solved in steps 4 and 6 is not that of the classical Weber problem. In this subsection we explore this issue.

Let us denote the locations of the demand points by $a_i, i = 1, \ldots, n$. We need a mathematical formulation for c_x^u, the marginal production cost of firm u when the facility is located at x. Assuming that the operational costs are the same regardless of the location chosen, then, c_x^u depends only on the location x. This cost is usually related to the relative location of the facility with regard to the demand points through its distances. The cost c_x^u does usually increase as x approaches any demand. This is due to the operational cost of the facility in the centres of high population concentrations being higher because of the value of land and premises. In [76], suitable expressions for c_x^u are suggested including the following formulation which we use here:

$$c_x^u = \sum_{i=1}^{n} \frac{w_i}{W} \frac{1}{d^u(x, a_i))^{\phi_{i0}^u} + \phi_{i1}^u}, \tag{1.25}$$

where $W = \sum_{i=1}^{n} w_i$ is the total customers demand, and $\phi_{i0}^u, \phi_{i1}^u > 0$ are parameters whose values are given or estimated. The first one modulates the distance whereas the second allows for adjustment while avoiding the division by zero (i.e., in case $x = a_i$).

Using (1.25), the corresponding minisum problems to be solved in steps 4 and 6 of the algorithm AWLA have the following objective function

$$\min F^u(x) = \sum_{i \in M^u} w_i (c_x^u + t_{xi}^u) = Q^u c_x^u + \sum_{i \in M^u} w_i g^u(d^u(x, a_i)),$$

where M^u stands for $M^1(x^1, x^2)$ if $u = 1$, and for $M^2(\hat{x}^1, x^2)$ if $u = 2$, $Q^u = \sum_{i \in M^u} w_i$ refers to the market share supplied by firm u, and $g^u(d^u(x, a_i))$ represents the transportation cost as a function of the distance.

A Weiszfeld-like procedure is developed to solve these new minisum problems using basic calculus. By equating the first partial derivative to zero we get

$$\frac{\partial F^u}{\partial x_1} = Q^u \sum_{i=1}^{n} \frac{w_i}{W} \frac{-\phi_{i0}^u (d^u(x, a_i))^{\phi_{i0}^u - 1}}{((d^u(x, a_i)^{\phi_{i0}^u} + \phi_{i1}^u)^2} \frac{\partial d^u(x, a_i)}{\partial x_1} + \sum_{i \in M^u} w_i \frac{\mathrm{d}g}{\mathrm{d}d^u(x, a_i)} \frac{\partial d^u(x, a_i)}{\partial x_1} = 0$$

If the distance function is such that

$$\frac{\partial d^u(x,a_i)}{\partial x_1} = x_1 A^u_{i1}(x) - B^u_{i1}(x),$$

then

$$x_1 = \frac{Q^u \sum_{i=1}^{n} B^u_{i1}(x) \frac{w_i}{W} \frac{-\phi^u_{i0}(d^u(x,a_i))^{\phi^u_{i0}-1}}{((d^u(x,a_i)^{\phi^u_{i0}}+\phi^u_{i1})^2} + \sum_{i\in M^u} B^u_{i1}(x) w_i \frac{\mathrm{d}g}{\mathrm{d}d^u(x,a_i)}}{Q^u \sum_{i=1}^{n} A^u_{i1}(x) \frac{w_i}{W} \frac{-\phi^u_{i0}(d^u(x,a_i))^{\phi^u_{i0}-1}}{((d^u(x,a_i)^{\phi^u_{i0}}+\phi^u_{i1})^2} + \sum_{i\in M^u} A^u_{i1}(x) w_i \frac{\mathrm{d}g}{\mathrm{d}d^u(x,a_i)}} \tag{1.26}$$

Note that $A^u_{i1}(x)$ and $B^u_{i1}(x)$ are functions of x which depend on the chain u, the customer i and the variable x_1. A simple representation of these functions will be given here.

Also, by equating the second partial derivative to zero and provided that

$$\frac{\partial d^u(x,a_i)}{\partial x_2} = x_2 A^u_{i2}(x) - B^u_{i2}(x),$$

x_2 can be computed as

$$x_2 = \frac{Q^u \sum_{i=1}^{n} B^u_{i2}(x) \frac{w_i}{W} \frac{-\phi^u_{i0}(d^u(x,a_i))^{\phi^u_{i0}-1}}{((d^u(x,a_i)^{\phi^u_{i0}}+\phi^u_{i1})^2} + \sum_{i\in M^u} B^u_{i2}(x) w_i \frac{\mathrm{d}g}{\mathrm{d}d^u(x,a_i)}}{Q^u \sum_{i=1}^{n} A^u_{i2}(x) \frac{w_i}{W} \frac{-\phi^u_{i0}(d^u(x,a_i))^{\phi^u_{i0}-1}}{((d^u(x,a_i)^{\phi^u_{i0}}+\phi^u_{i1})^2} + \sum_{i\in M^u} A^u_{i2}(x) w_i \frac{\mathrm{d}g}{\mathrm{d}d^u(x,a_i)}}. \tag{1.27}$$

The functions $A^u_{i2}(x)$ and $B^u_{i2}(x)$ are defined in a similar way as $A^u_{i1}(x)$ and $B^u_{i1}(x)$.

Using the above equations (1.26) and (1.27), we update x_1 and x_2 in a recursive manner which leads to a Weiszfeld-like method which can easily be constructed.

Possible expressions for A^u_{i1}, B^u_{i1}, A^u_{i2}, and B^u_{i2}

A distance function satisfying the conditions

$$\frac{\partial d^u(x,a_i)}{\partial x_1} = x_1 A^u_{i1}(x) - B^u_{i1}(x), \frac{\partial d^u(x,a_i)}{\partial x_2} = x_2 A^u_{i2}(x) - B^u_{i2}(x)$$

is the ℓ_{2b}-norm, a suitable distance predicting function (see [73]), given by

$$\ell^u_{2b}(x,a_i) = \sqrt{b^u_1(x_1 - a_{i_1})^2 + b^u_2(x_2 - a_{i_2})^2}.$$

For this particular function we get the following mathematical expressions:

$$A^u_{i1} = \frac{b^u_1}{d^u(x,a_i)}, \quad B^u_{i1} = \frac{a_{i_1} b^u_1}{d^u(x,a_i)} \quad A^u_{i2} = \frac{b^u_2}{d^u(x,a_i)}, \quad B^u_{i2} = \frac{a_{i_2} b^u_2}{d^u(x,a_i)}.$$

Proposition 1.3.1. *Each step of the AWLA method is non-increasing for the multisource Weber problem.*

Proof. It is well known that the Weiszfeld algorithm is a steepest descent method with discrete step size. As AWLA applies the Weiszfeld algorithm to different 1-median problems, we need to prove that every time that the base set is changed within AWLA, the objective function of problem (1.22) decreases. Here we will only show the case when the Weiszfeld algorithm is applied to firm 1 using the set $M^1(x^1, x^2)$. A similar proof can be derived when the Weiszfeld algorithm is applied to firm 2 based on the set $M^2(\hat{x}^1, x^2)$.

We consider the following three cases: gaining a new customer, losing an existing customer and both receiving a new customer and losing an existing one. We also concentrate our proof on the case of one demand point as the reasoning can easily be extended to the case of several demand points.

Case 1: The set $M^1(x^1, x^2)$ receives a new demand point i_2.

Since demand point i_2 now belongs to $M^1(x^1, x^2)$, we have $d_{i_2}(x^1) < d_{i_2}(x^2)$. Before applying the Weiszfeld method, the social cost is

$$S_{bef}(x^1, x^2) = \sum_{i\in M^1(x^1,x^2), i\neq i_2} d_i(x^1) w_i + d_{i_2}(x^2) w_{i_2} + a,$$

where

$$a = \sum_{i \in M^2(x^1,x^2), i \neq i_2} d_i(x^2) w_i.$$

After applying the Weiszfeld method, a new point $\hat{x}^1$ is obtained and the social cost is then

$$S_{aft}(\hat{x}^1, x^2) = \sum_{i \in M^1(x^1,x^2), i \neq i_2} d_i(\hat{x}^1) w_i + d_{i_2}(\hat{x}^1) w_{i_2} + a.$$

We have that

$$S_{bef}(x^1, x^2) > \sum_{i \in M^1(x^1,x^2), i \neq i_2} d_i(x^1) w_i + d_{i_2}(x^1) w_{i_2} + a \geq S_{aft}(\hat{x}^1, x^2),$$

where the last inequality holds because of the descent property of the Weiszfeld algorithm as the demand point i_2 is now included in the base set.

Case 2: The set $M^1(x^1, x^2)$ loses one of its demand points i_1.

Since demand point i_1 does not belong to $M^1(x^1, x^2)$ any more, this leads to $d_{i_1}(x^1) > d_{i_1}(x^2)$. Before applying the Weiszfeld method, the social cost is

$$S_{bef}(x^1, x^2) = \sum_{i \in M^1(x^1,x^2), i \neq i_1} d_i(x^1) w_i + d_{i_1}(x^1) w_{i_1} + a,$$

where

$$a = \sum_{i \in M^2(x^1,x^2), i \neq i_1} d_i(x^2) w_i.$$

After applying the Weiszfeld method, a new point $\hat{x}^1$ is obtained and the social cost becomes

$$S_{aft}(\hat{x}^1, x^2) = \sum_{i \in M^1(x^1,x^2), i \neq i_1} d_i(\hat{x}^1) w_i + d_{i_1}(x^2) w_{i_1} + a.$$

We have therefore

$$S_{bef}(x^1, x^2) > \sum_{i \in M^1(x^1,x^2), i \neq i_2} d_i(x^1) w_i + d_{i_1}(x^2) w_{i_1} + a \geq S_{aft}(\hat{x}^1, x^2),$$

where the last inequality holds because of the descent property of the Weiszfeld algorithm given the demand point i_1 is no longer in the base set.

Case 3: The set $M^1(x^1, x^2)$ receives a new demand point i_2 but it also loses one of its demand points i_1.

Now $d_{i_2}(x^1) < d_{i_2}(x^2)$ and $d_{i_1}(x^1) > d_{i_1}(x^2)$. Before applying the Weiszfeld method,

$$S_{bef}(x^1, x^2) = \sum_{i \in M^1(x^1,x^2), i \neq i_1, i_2} d_i(x^1) w_i + d_{i_1}(x^1) w_{i_1} + d_{i_2}(x^2) w_{i_2} + a,$$

where

$$a = \sum_{i \in M^2(x^1,x^2), i \neq i_1, i_2} d_i(x^2) w_i.$$

After applying the Weiszfeld method, a new point $\hat{x}^1$ is obtained and the social cost is then

$$S_{aft}(\hat{x}^1, x^2) = \sum_{i \in M^1(x^1,x^2), i \neq i_1, i_2} d_i(\hat{x}^1) w_i + d_{i_1}(x^2) w_{i_1} + d_{i_2}(\hat{x}^1) w_{i_2} + a.$$

We have therefore

$$\begin{aligned} S_{bef}(x^1, x^2) &> \sum_{\substack{i \in M^1(x^1,x^2) \\ i \neq i_1, i_2}} d_i(x^1) w_i + d_{i_1}(x^2) w_{i_1} + d_{i_2}(x^1) w_{i_2} + a \\ &\geq S_{aft}(\hat{x}^1, x^2), \end{aligned}$$

where the last inequality holds because of the descent property of the Weiszfeld algorithm as the demand point i_2 is included in the base set, whereas i_1 is not.

The above cases cover all possibilities, thus we have shown what is stated.

□

1.3.3.3 Implementation issues

We would like to highlight two computational enhancements to speed up and guide the search.

(i) To avoid the new facilities to be located on demand points, infeasible regions around the customer sites are introduced. These are defined by small forbidden circles, each centred at a customer site with a small radius which is proportional to the customer demand and inversely proportional to the square root of the number of customers. These forbidden circles also help to avoid the problem of dividing by zero in equations (1.26) and (1.27) while guaranteeing both the monotonicity and the pruning tests to remain well defined. Note that a small constant value for all radii could also be used for simplicity if necessary. Note that the classical Weiszfeld algorithm when applied to the 1-median problem is not guaranteed to converge when an iterate happens to lie in one of the (non-optimal) demand points.

(ii) In addition, if from one iteration to the next, the new point, say point A, is moved to point B which happens to lie inside a forbidden circle, the two intersection points of the line passing through A and B and the circle are evaluated and the one yielding the least cost solution is then selected as the new location.

1.3.4 Computational results

Our algorithms are implemented in C++ under Linux on a Pentium IV computer with 3.2 GHz CPU and 3 GB memory. For the interval B&B method, the interval arithmetic implementation of the PROFIL/BIAS library [119] and the automatic differentiation of the C++ Toolbox library [99] are used.

The performance of the proposed algorithms for solving (P') is conducted using three scenarios. Here, we use marginal production costs as defined by (1.25). In the first scenario, small to medium size randomly generated problems with $n = 100$ to 500 with a step size of 100 are used. In the second scenario, larger instances varying in size from $n = 1000$ to $n = 10000$ with a step size of 1000, which are also randomly generated, are tested. In both scenarios, the instances were generated by randomly choosing the locations of the demand points within the square $([0, 10], [0, 10])$ and the demands w_i in the interval $(0, 10]$. The marginal transportation costs were given by the Euclidean distance, $g^u(d^u(x, a_i)) = \ell_2(x, a_i), u = 1, 2$. The parameters ϕ^u_{i0} and ϕ^u_{i1} in (1.25) were drawn from $[0.5, 2]$. To gather basic and reliable descriptive statistics, 10 instances for each value of n are considered. The third scenario, which is performed for completeness, refers to the special case given by (1.22), namely, the multi-source Weber problem for $p = 2$. In this scenario, our algorithms are tested on the four commonly used data sets from the literature (i.e., n=50, 287, 654, and 1060), see Brimberg *et al.* [27] for more details.

1.3.4.1 Scenario 1 (Small to medium size instances)

As mentioned briefly earlier, 5 data sets each with 10 instances totalling 50 cases altogether are used in our testing. The smallest being with $n = 100$ and the largest with $n = 500$.

(a) Results of the Interval B&B method

In Table 1.7, the results for the interval B&B are given. The implementation includes the discarding tests as described in Subsection 1.3.2 and uses an accuracy measure of $\varepsilon = 0.01$ as the stopping criterion. These results will be used for benchmarking against our heuristic AWLA to assess its performance. *iBBObj* shows the guaranteed upper bound on the minimum value of the objective function, MaxL records the maximum number of boxes in the working list at any time during the execution and Time iBB represents the running time in CPU seconds. The average results for each problem size ($n = 100, 200, 300, 400, 500$) is recorded and the largest CPU time is underlined.

In general, the method obtains the optimal solutions within 1 hour for those instances with $n \leq 300$ but most of the others required more time with many needing over 2 hours. MaxL, which gives an idea of the memory requirements of the algorithm, seems to increase linearly with the size of the problem with the exception of some of the instances with $n = 500$ demand points where up to 74 000 boxes were needed. Such a drawback explains some of the reasons why the interval B&B method may not be suitable for solving large instances.

(b) Results of the AWLA

In Table 1.7, the best results of the AWLA are also reported. Here, the method is repeated for 100 times starting from random points, using an accuracy of 0.01 in the application of the Weiszfeld-like algorithm. In the columns, Difference Obj gives the percentage deviation with regard to the guaranteed upper bound

on the minimum value of the objective function value as obtained by the interval B&B, which is computed as follows:

$$((Obj(AWLA) - iBBObj)/iBBObj) \times 100,$$

where $Obj(AWLA)$ refers to the best objective function value found by AWLA over the 100 runs. Difference Loc shows the difference in location measured as the Euclidean distance between the two locations found by the two methods. The number of times the optimal solution is found by AWLA in the 100 runs is denoted by TF. Time Weisz represents the total running time in CPU seconds for AWLA (the sum of the CPU time for the 100 runs). For each of the five problem sizes, the average over the 10 instances is computed and both the worst deviation and the largest CPU time are underlined. According to these results, the deviation, in terms of solution quality from the optimum is found to be relatively small, with less than 0.05% for all problems and 0.003% on average only. For the larger problems ($n = 500$), the deviations are even smaller. This could be due to the calculation of the relative distances. The negative values show those cases where the Weiszfeld method managed to find a slightly better result than the guaranteed upper bound provided by the interval branch-and-bound method. The computational effort for the AWLA is relatively negligible as it accounts on average for approximately less than 0.30% of the time spent by the interval B&B. However, we would like to emphasize that the interval B&B is an exact method which would be able to guarantee obtaining *all* the optimal solutions if it was let to run till the end, whereas AWLA is a simple greedy multi-start heuristic which is relatively very fast.

Table 1.7: Results for the interval B&B and Weiszfeld methods for small and medium sized instances

n	$iBBObj$	Difference		TF	MaxL	Time (secs)	
		Obj (%)	Loc			iBB	Weisz
100	1525.5	0.025	0.090	1	16982	432.62	1.38
100	1578.6	0.003	0.040	7	17167	453.16	1.33
100	1351.2	0.001	0.125	4	19745	609.13	1.13
100	1261.2	-0.002	0.022	1	18324	417.13	1.23
100	1295.8	0.043	0.231	1	16367	367.17	1.50
100	1474.2	0.001	0.015	49	17402	426.19	1.20
100	1522.2	-0.007	0.013	6	17347	357.15	1.25
100	1549.7	0.028	0.069	2	10883	191.69	1.56
100	1440.2	-0.007	0.008	1	38094	780.19	1.04
100	1541.9	0.002	0.013	1	26446	635.51	1.14
Average	**1454.0**	**0.009**	**0.063**	**7.3**	**19875**	**467.0**	**1.28**
200	2838.2	-0.000	0.008	53	30050	1280.34	3.95
200	2972.5	0.001	0.006	52	27566	1191.37	3.08
200	3153.8	0.003	0.063	7	25932	1249.20	3.14
200	3334.3	0.001	0.010	35	34444	1541.12	3.36
200	2721.8	0.002	0.093	5	49042	2035.25	2.58
200	3194.9	0.009	0.076	1	45036	1850.48	2.58
200	2915.7	-0.001	0.026	3	41926	1828.13	3.28
200	2985.5	-0.001	0.009	1	48406	2463.20	3.46
200	3020.0	0.005	0.056	8	30431	1316.83	3.71
200	3031.7	0.001	0.005	36	26053	1208.43	4.11
Average	**3016.8**	**0.002**	**0.035**	**20.1**	**35888**	**1596.4**	**3.32**
300	4526.0	-0.001	0.007	35	46956	3008.09	5.17
300	4612.2	-0.000	0.013	41	33360	2421.58	4.68
300	4454.1	0.003	0.063	30	48136	3205.28	4.63
300	4627.3	-0.000	0.008	17	36481	2595.10	5.08
300	4676.2	0.000	0.008	23	54122	3376.60	4.12
300	4426.4	0.001	0.024	5	39353	2719.11	5.45
300	4536.8	-0.001	0.031	13	40644	2706.72	4.86
300	4326.9	0.001	0.005	7	39011	2963.66	4.66
300	4483.5	0.003	0.022	9	38580	2809.68	4.74
300	4467.4	0.001	0.008	17	58788	3903.96	4.40
Average	**4513.7**	**0.001**	**0.019**	**19.7**	**43543**	**2971.0**	**4.78**
400	5831.5	0.000	0.010	21	74504	7205.46	6.25
400	5844.2	0.001	0.012	29	55013	5393.91	6.40
400	5715.5	0.001	0.063	37	43945	4352.18	9.88
400	6087.7	0.000	0.123	45	48411	5067.76	7.71
400	6011.5	0.002	0.053	28	34758	3125.16	8.37

Table 1.7: Results for the interval B&B and Weiszfeld methods (continued)

n	*iBBObj*	Difference		TF	MaxL	Time (secs)	
		Obj (%)	Loc			iBB	Weisz
400	5865.1	0.001	0.009	44	39071	4150.61	7.99
400	6006.5	0.000	0.012	7	59185	5475.61	7.65
400	5645.6	0.007	0.061	1	63702	5897.20	8.43
400	6642.5	0.003	0.090	4	69139	6554.46	6.46
400	6365.6	0.002	0.068	9	44716	4179.55	6.12
Average	**6001.6**	**0.002**	**0.050**	**22.5**	**53244**	**5140.2**	**7.53**
500	7563.6	0.001	0.032	40	48527	5691.62	11.77
500	7529.0	-0.000	0.009	41	65843	7673.00	8.56
500	7367.8	0.000	0.041	24	50923	6500.49	8.05
500	7683.2	-0.000	0.012	2	65395	8401.75	9.34
500	7226.2	-0.000	0.012	62	53711	6378.73	8.52
500	7373.4	0.000	0.003	15	74007	10151.39	8.42
500	7169.2	0.001	0.006	9	60734	8040.25	8.19
500	7672.8	0.001	0.037	24	63389	7994.03	11.76
500	7607.7	-0.000	0.007	6	44040	5164.02	8.59
500	7815.9	0.001	0.047	3	49832	6071.81	8.08
Average	**7500.9**	**0.000**	**0.021**	**22.6**	**57640**	**7206.7**	**9.13**

1.3.4.2 Scenario 2 (larger instances)

As in scenario 1, we used 10 instances for each value of n where $n = 1000$ to 10000 with a step size of 1000 totalling 100 instances altogether. As in scenario 1, we used 100 restarts and 0.01 accuracy in AWLA, but 30 Weiszfeld iterations are allowed at most though in most cases this restriction was found to be unnecessary. The average results with its standard deviation (St. Dev.) are summarized in Table 1.8. To provide some form of evaluation of our AWLA, we also let the interval B&B method run for one hour and record the best upper bound found which we denote by Upper Bound (UB). The percentage deviation of the objective function value from (UB) is referred to as (Difference Obj) and is computed as follows:

$$((Obj(AWLA) - UB)/UB) \times 100.$$

We have also recorded the lower bounds derived within the set execution time. However, these have been found to be loose which in our view are not that informative and could be even misleading. For clarity of presentation, these are not provided here but can be collected from the authors. The AWLA obtains solutions that outperformed the upper bounds in each of the 100 instances while requiring a tiny fraction of the amount used by the exact method. For basic descriptive statistics, we present in bold the averages and underline the worst deviation for each problem size. This demonstrates the efficiency as well as the usefulness and the simplicity of our heuristic. For example, the largest average in CPU time is just under 4 minutes accounting for approximately 7% of the total time allowed for the interval B&B. This is observed for the instances with $n = 10000$ and shown in bold in Table 1.8. For the other instances, the CPU time was found to be relatively much smaller.

Table 1.8: Average summary results for the AWLA algorithm for the larger instances

n	Upper Bound		Difference Obj		Time Weisz (secs)	
	Average	St.Dev.	Average	St.Dev.	Average	St.Dev.
1000	15138.7	366.02	-0.03	0.01	21.03	2.17
2000	30619.9	619.86	-0.04	0.02	41.57	3.57
3000	45893.4	462.73	-0.09	0.03	67.12	4.00
4000	60569.9	570.28	-0.12	0.05	92.74	5.41
5000	76164.4	918.39	-0.16	0.06	115.41	4.43
6000	91960.7	819.73	-0.12	0.07	139.65	5.42
7000	106565.0	772.34	-0.18	0.08	161.74	11.98
8000	122380.0	606.73	-0.15	0.08	189.43	9.80
9000	137261.0	962.05	-0.21	0.08	216.72	5.14
10000	152941.0	996.11	-0.25	0.13	**239.20**	13.69

Table 1.9: Results for the 2-Weber problems

	Best Known		AWLA		iBB	
			$n = 50$			
obj	135.5222		135.52222		135.52[01685051,22315334]	
x^1	2.66718	5.64436	2.6676	5.6441	2.6[46,85]	5.6[25,64]
x^2	7.23707	4.54258	7.2376	4.5408	7.2[26,56]	4.5[11,70]
			$n = 287$			
obj	14427.59		14427.60187		144[04.2532932, 27.7367554]	
x^1	21.76742	31.66553	21.768	31.665	21.[62,91]	31.[48,85]
x^2	23.99999	29.00009	24.001	29.001	2[3.9,4.1]	2[8.9,9.1]
			$n = 654$			
obj	815313.2962		815313.29615		815313.29[3511738,61504267]	
x^1	1224.3163	3766.0064	1224.3	3766.0	1224.[24,38]	376[5.8,6.2]
x^2	5633.1123	3618.9654	5633.1	3618.9	5633.[05,15]	361[8.6,9.3]
			$n = 1060$			
obj	3010448.052		3010448.05202		3010448.05[0469826,2050685]	
x^1	7548.24	4575.99	7548.2	4576.0	7548.[14,33]	457[5.9,6.1]
x^2	15981.86	4888.624	15982.0	4888.6	1598[1.7,2.1]	4888.[45,76]

1.3.4.3 Scenario 3 (the special case: the Weber-based data sets)

We also conducted an empirical experiment on the four data sets that are commonly used for the multi-source Weber problems (see Brimberg *et al.* [27]). Though this is not directly related to our problem, it is a special case when the distance is used as a proxy for the delivered cost (see Subsection 1.3.1). There are four instances with $n = 50, 287, 654$, and 1060. For n=50, 654, and 1060, all customers have a unit weight ($w_i = 1$ for all i) whereas for n =287 the value of w_i varies enormously from 1 to 1000. These instances are usually used as a platform to test new approaches for the multi-source Weber problem with varying the number of open facilities (i.e., p). Here, we first record the optimal (or best known) solution with $p = 2$ which can be found in the literature, see for instance Brimberg and Drezner [25]. As we need to use the corresponding facility configurations which are not published, we implemented a crude force approach which uses all possible pairs of nodes as a starting point in the Weiszfeld recursive equations. As the cost of the optimal solution for $n = 1060$ was not initially reported, we checked our result against the one found by the recently proposed approach of [25]. This was achieved through a private communication with Professor Zvi Drezner who kindly agreed to check it for us.

In Table 1.9 we report the 'Best Known' results and the solution of the 2-source Weber problem using AWLA and iBB algorithms. In the iBB columns, the format $A[b, c]$ is to show that the lower and upper bounds differ significantly only in the bracketed $[b, c]$ part. The results are found to be very similar in terms of the objective function value and facility coordinates. The exception is for $n = 287$, where the second facility is located on a demand point. It is interesting to see that although the interval hull of the resulting intervals in the solution space is as tight as in the other problems, the inclusion of the objective is much wider. This clearly shows how hilly the objective function may be around a good demand point. In any case, the results confirm that both AWLA and iBB work as expected, and find the optimal solution.

We have also solved the corresponding (P') problems obtained using the demand points and weights of the four 2-source Weber problems but using the general cost function (1.25). We used the 'Best Known' solutions of the 2-source Weber problems as (i) a reference point to re-evaluate the total cost used in the corresponding (P') problem and (ii) as an initial solution for AWLA. In Table 1.10 the results for the corresponding general (P') problems are summarised under the following columns. These include the solution points of the 2-source Weber problems and the function value of the objective function of the corresponding (P') at those points, the solution by AWLA starting from the solution of the 2-source Weber problem, and the best solution of the AWLA starting from 100 random points. We refer to the two AWLA implementations as AWLA(2W) and AWLA(Rand), respectively. For the two smaller cases, we can observe that the solution of (P') (as given by AWLA(Rand)), although in the neighbourhood of the solution of the 2-source Weber problem, is slightly better. It was also found that even when we start the AWLA procedure from the solution of the 2-source Weber problem, this leads to a better solution. This demonstrates that the optimal solution of the 2-source Weber problem is not necessary the optimal solution for (P'). For the two larger instances, the solution is almost the same. This can be explained by the functional form of c_x^u

Table 1.10: Results for the general problem (P')

n		2-source Weber sol.		AWLA(2W)		AWLA(Rand)	
	obj	158.82726		158.05148		158.05148	
50	x^1	2.6676	5.6441	2.52850	5.34531	2.52771	5.34556
	x^2	7.2376	4.5408	7.60611	4.50231	7.60599	4.50011
	obj	19887.30311		19475.80336		19464.73714	
287	x^1	21.768	31.665	21.07986	32.45766	21.08255	32.45162
	x^2	24.0001	29.0001	23.99993	29.00009	23.84776	29.19114
	obj	815313.30097		815313.30089		815313.30093	
654	x^1	1224.3	3766	1224.316	3765.999	1224.315	3765.980
	x^2	5633.1	3618.9	5633.114	3618.915	5633.112	3618.976
	obj	3010448.05485		3010448.05369		3010448.05369	
1060	x^1	7548.25	4575.99	7548.240	4575.993	7548.240	4575.994
	x^2	15981.87	4888.62	15981.87	4888.619	15981.86	4888.624

in (1.25): note that as the number of demand points increases, the weight of c_x^u in the objective function gets smaller. In this case, the objective function value is mostly determined by the transportation cost, as in the 2-source Weber problems. Of course, the use of other productions cost functions could have led to different solutions. In brief, we can conclude that the optimal solution of the 2-source Weber problem cannot guarantee to remain optimal for (P') as demonstrated in the two above counter-examples.

1.3.5 Conclusions and suggestions

The problem of determining a location equilibrium of a location-price game in the plane is investigated where two competing firms locate one single facility each. An interval B&B algorithm is adopted to solve the problem under a general cost function but it is found to be suitable for small size instances only. An alternating Weiszfeld-like heuristic, which is more appropriate for larger instances, is put forward based on a newly developed iterative formula for updating the coordinates of the two facilities. Interesting results are provided using extensive computational experiments. In brief, our simple multi-start heuristic produces better results while requiring a tiny fraction of the CPU time allocated to the exact method. As a special case, we also examined the relationship between the optimal solutions of the 2-source Weber problem and (P'). We conducted experiments on existing data sets where it was found that the optimal solution of the 2-source Weber problem does not necessarily remain optimal for our new problem (P').

Our investigation can be extended in several ways. These can be classified under two headings; (i) enhancements to the proposed approaches and (ii) an investigation into other related problems and open questions. In (i), additional learning schemes and guidance may need to be embedded into the implementation of the AWLA so to make the algorithm more competitive and more adaptive. Meta-heuristics could obviously be combined with our Weiszfeld-type heuristic to enhance the quality of the solutions further.

In part (ii), a study that explores the case of several competing firms that wish to locate a number of facilities each where this can be either given or considered as a decision variable is worth attempting. There are also questions that may be worth asking. For example, if the facilities have limited capacities, does this lead to a price equilibrium? Does the minimization of the social cost guarantee location equilibria? Does the complexity of the problem remains similar to the current one? In many situations, the demand may vary with the price, the marginal production cost may not even be constant, among other parameters that are unlikely to remain constant. These variations are useful and practical for end users while being challenging and deserving a thorough investigation by us as academics.

Chapter 2

Simplicial branch-and-bound methods

Branch-and-bound (B&B) algorithms in Global Optimization are used to perform an exhaustive search over the feasible area. For some special problems, the feasible area is a simplex, for instance in portfolio optimization and in blending problems, where the sum of the variables must be one. Also, for any convex or concave polyhedral set, decomposition into simplicial sets is straightforward by the algorithm described in [78]. In such cases, the natural choice is to use simplicial partition sets, resulting in a simplicial branch-and-bound method. A review of simplicial branch-and-bound can be found in [146].

In this chapter, the result of mainly three articles are presented, all of them dealing with some part of the simplicial branch-and-bound method. First, a discarding test is built on spherical exclusion regions at the vertices of the simplex. Second, the subdivision rule is revisited to study regular refinement procedures, and finally, bounding rules are designed and sharpened for simplices.

When general constraints are present, it is quite hard to show that the current simplex is feasible or not. In [37] infeasibility spheres were found from each vertex. Thus, a simplex covered by these spheres can be shown to be infeasible. Developing fast tests to check the infeasibility this way can speed up the branch-and-bound considerably. Hence, in Section 2.1 results on this Simplex Cover (SC) problem are discussed [36]. Lately, it has been proved that SC is NP-hard [199].

A natural way to define branching in simplicial branch-and-bound is bisection. The consequence of bisecting simplices is that partition sets are in general irregular. The question is how to use regular simplices in the refinement of the unit simplex. A regular simplex with fixed orientation can be represented by its center and size, facilitating storage of the search tree from a computational perspective. The problem is that a simplex defined in a space with dimension greater than three cannot be subdivided into regular subsimplices without overlapping. We study the characteristics of the refinement by regular simplices [85]. The main challenge is to find a refinement with a good convergence ratio which allows discarding simplices in an overlapped and already evaluated region. As the efficiency of the division rule in B&B algorithms is instance dependent, we focus on the worst case behaviour, i.e., none of the branches are pruned. Section 2.2 shows that for this case surprisingly an overlapping regular refinement may generate fewer simplices to be evaluated than longest edge bisection. On the other hand, the number of evaluated vertices may be larger.

The other key point in the construction of efficient B&B algorithms is how to obtain sharp and cheap bounds of the objective function over a simplex [84]. Although enclosing a simplex in a box implies an overestimation, boxes are more natural when dealing with individual coordinate bounds, and bounding ranges with Interval Arithmetic (IA) is computationally cheap. Section 2.3 introduces several linear relaxations using gradient information and Affine Arithmetic. It also studies experimentally their efficiency compared to traditional lower bounds obtained by natural and centred IA forms and their adaption to simplices. A Global Optimization B&B algorithm with monotonicity test [104] over a simplex is used to compare their efficiency over a set of low dimensional test problems with instances that either have a box constrained search region or where the feasible set is a simplex. Numerical results show that it is possible to obtain tight lower bounds over simplicial subsets.

2.1 Covering a simplex by spheres

This study is motivated by an earlier work on testing whether simplices representing the design area of blending problems may contain feasible solutions. The spheres represent areas where certainly no

feasible solution can be located. This means, that if a simplex is covered, then the simplex can be excluded from the search, because it certainly does not contain a feasible solution [37, 105]. This work studies the characteristics of the SC problem. The reasoning is illustrated with graphical examples and validated by several lemmas and a theorem.

We formulate the problem and its ingredients. A simplex is a set of convex combinations of h affine independent vertices $v_1, \ldots, v_h$,

$$S = \{x \in \mathbb{R}^n \ : \ x = \sum_{j=1}^{h} \lambda_j v_j, \ \sum_{j=1}^{h} \lambda_j = 1; \ \lambda_j \in [0,1], j = 1, \ldots, h\}. \tag{2.1}$$

Usually the vertices are defined in Euclidean space where the dimension n should at least be $h-1$. Regular simplices are special cases where all the edges have the same length. The next ingredient of the SC problem are the spheres or balls with radius r_i around the vertices,

$$B_i = \{x \in \mathbb{R}^n \ : \|x - v_i\| \leq r_i\}. \tag{2.2}$$

In our investigation we are specifically interested in the Euclidean distance norm $\|.\|_2$. Now the Simplex Cover problem can be formulated.

Simplex Cover (SC) problem. Given an instance of a simplex S and a set of spheres $B_1, \ldots, B_h$ around its vertices. Certify if

$$\forall x \in S, \ \ x \in \cup_i B_i \tag{2.3}$$

or alternatively, check whether

$$\exists x \in S, \ x \notin \cup_i B_i. \tag{2.4}$$

An equivalent representation of the SC problem can be used that focuses far more on distance from the vertices. This comes closer to the concept of Laguerre Voronoi diagrams (or Power diagrams) that is elaborated in Subsection 2.1.1. We focus on squared Euclidean distance. Define the function

$$\varphi(x) = \min_i \{\|x - v_i\|^2 - r_i^2\}, \tag{2.5}$$

where the squared distance is additively weighted [8]. Notice that function φ being a minimum of strict convex functions is not convex. Moreover, in the specific case where none of the spheres B_i is covering one of the other vertices $v_j, i \neq j$, φ has local minima of $-r_i^2$ in each vertex; $\varphi(v_i) = -r_i^2$. The SC problem is equivalent to

$$\forall x \in S \ \varphi(x) \leq 0. \tag{2.6}$$

Examples of the φ function can be seen in Figure 2.1.

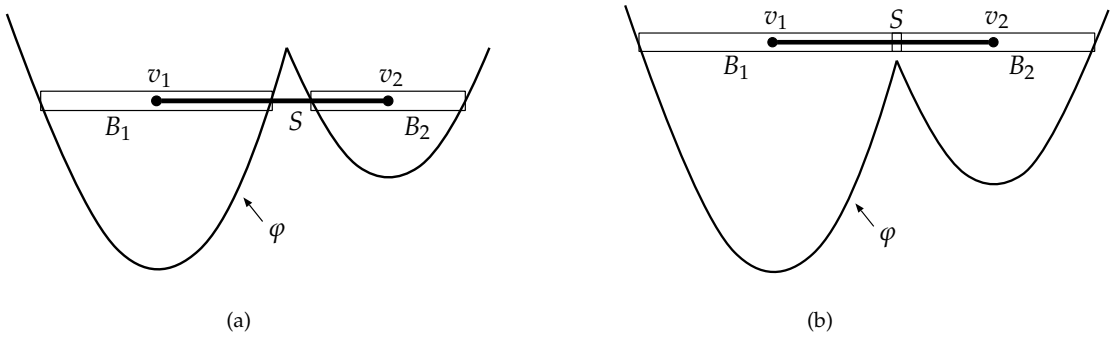


Figure 2.1: Instances of SC problem with two vertices, where S is not covered (a), and where S is covered (b).

Notice that for a regular instance of S, if one of the B_i covers another vertex, it covers them all and SC is solved.

Equivalence (2.6) shows that it is interesting to study problem SCO (Simplex Cover Optimization):

$$\Phi := \max_{x \in S} \varphi(x). \tag{2.7}$$

The SC problem has a certificate if $\Phi \leq 0$.

A lot of publications are available on so-called Power sets [68, 170] that relate to φ and mainly focus on lower dimensional analysis. Our research deals with the general dimensional case. We elaborate this in Subsection 2.1.1. Problem SCO is not necessarily easier to solve than SC (2.3) or (2.4). One of the most obvious observations on the SC problem is that the simplex S is covered if one of the spheres B_j covers it. If this is not the case, we require properties on the SC problem of covering and the SCO problem of maximizing φ over S. In Subsections 2.1.2 and 2.1.3, we derive mathematical properties of the SC and SCO problems that can be used to solve them, which we illustrate by numerical examples. Specifically, Subsection 2.1.4 analyses the case of regular simplices, which is easier to solve and useful for the blending problem. Finally, Subsection 2.1.5 summarises the results.

The problem was coming from an earlier work of my co-authors, and my main contribution were to find out about power diagrams and the theoretical results written in Subsections 2.1.2 and 2.1.3. Of course, my co-authors played an important role in how the final version of the published paper appeared.

2.1.1 Distances, Voronoi diagrams and simplices

Voronoi diagrams for a set of points p_i, divide the space into regions $D(p_i)$ where all points in $D(p_i)$ are closer to p_i than to p_j, $j \neq i$. A survey of Voronoi diagrams can be found in [8]. There are several distance functions that can be used with respect to Voronoi diagrams. The usual is to consider the euclidean distance:

$$d_e(p_1, p_2) = \|p_1 - p_2\|_2. \tag{2.8}$$

A Voronoi cell of p_i for Euclidean distance are those points closest to p_i:

$$D(p_i) = \{x \in \mathbb{R}^n | d_e(x, p_i) \leq d_e(x, p_j),\ \ j \neq i,\ j = 1, \ldots, h\}. \tag{2.9}$$

Each sphere B_i in the SC problem has a radius r_i. This radius can be considered as a weight. Two ways to weight the distance are commonly considered and could be used for what are called weighted points, p_i:

- Weight distance additively:
$$d_a(x, p_i) = d_e(x, p_i) - r_i. \tag{2.10}$$
- Weight the so-called power distance:
$$d_p(x, p_i) = d_e(x, p_i)^2 - r_i^2. \tag{2.11}$$

Using d_a in the Voronoi concept leads to diagrams with edges that are hyperbolic curves. Also so-called multiplicative Voronoi diagrams exist that have hyperbolic curves [8].

Life is a bit easier using d_p; the separating shapes are planes. One obtains a power diagram with power planes

$$\Pi_{ij} = \{x \in \mathbb{R}^n \mid d_p(x, p_i) = d_p(x, p_j)\} \tag{2.12}$$

and power cells

$$D_p(p_i) = \{x \in \mathbb{R}^n \mid d_p(x, p_i) \leq d_p(x, p_j),\ \ j \neq i,\ j = 1, \ldots, h\}. \tag{2.13}$$

For a point x outside sphere B_i, $d_p(x, p_i)$ is positive and $\sqrt{d_p(x, p_i)}$ represents the distance from the sphere with radius r_i around v_i to the point x outside the sphere along the tangent line through the point x [91]. The *power Voronoi diagram* (or *power diagram*) cells are convex polyhedra. Figure 2.2 shows the power diagram edges produced by two weighted points. The difference between a Euclidean Voronoi diagram and a Power diagram is that in the latter a cell can be empty, i.e., there is no point inside (see Figure 2.2(d)).

An important property of weighted and unweighted Voronoi diagrams is:

Statement 2.1.1. *Given h affinely independent points with $h > 2$. The Voronoi diagram contains one vertex of degree h (number of edges incident to the vertex).*

This has been shown for low dimensional cases in [8]. In Subsection 2.1.2 we will focus on the power diagram for the general dimensional case and derive a procedure to determine the corresponding so-called θ point. We will show that it plays a major role in the solution of SC and SCO problems.

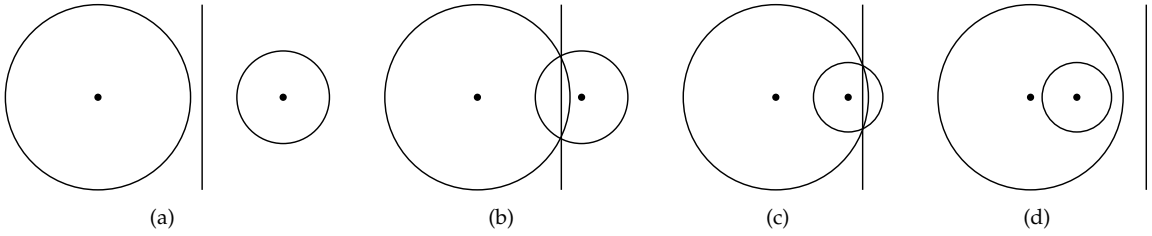


Figure 2.2: Power planes of two weighted points on four different instances

2.1.2 Covering a simplex, analysis based on the θ-point

The function

$$\varphi(x) = \min_i \{d_p(x, v_i)\}, \tag{2.14}$$

is piecewise convex in each power cell. This implies that a maximum of φ over a power cell is attained in one of its extreme points. It means that looking for the maximum value, Φ (see Equation (2.7)), we can focus on the power planes Π_{ij} that constitute the edges of power diagrams. Let us denote the interior of a set by *int*. Lemma 2.1.2 tells us that the solution of SCO should be looked for on the boundary of the power cells.

Lemma 2.1.2. *A point* $x^* \in \operatorname{argmax}_{x \in S} \varphi(x) \notin intS \cap \cup_i intD_p(v_i)$.

Proof. For an interior point x of a power cell around v_i,

$$\varphi(x) = d_p(x, v_i) < d_p(x, v_j), \quad \forall j \neq i.$$

This means that one can increase the value of φ by going in the direction $\nabla\varphi(x) = 2(x - v_i)$. Therefore, x cannot be a maximum point. □

It is easy to check whether one sphere covers the complete simplex. In this case the maximum of $\varphi(x)$ may be attained at one or several vertices v_i of S (see Figures 2.2(c) and 2.2(d)). For the case the maximum is not in a vertex, the lemma shows us that we can concentrate on the power planes Π_{ij}. Actually, it gives us the information that if the intersection of the power planes with the simplex are covered, i.e. $\varphi \leq 0$ on $S \cap \cup_{ij}\Pi_{ij}$, then all the simplex is covered. To illustrate these observations, we consider the line between the centers of the spheres of Figure 2.2 as a simplex. One can see that in Figure 2.2(a) the spheres do not intersect and the simplex is not covered in contrast to Figure 2.2(b). Notice that the intersection point of power plane and simplex corresponds to the global maximum point x^*. Figure 2.2(c) illustrates the case where one of the spheres covers the complete simplex and in (d) also the other sphere. Notice that in the last two cases, the power plane does not intersect with the simplex and the maximum point x^* is found in a vertex of S.

We now focus on the situation where the maximum is attained at an interior point of S; i.e. $x^* = V\lambda$, where matrix $V = [v_1, v_2, \ldots, v_h]$ and $\lambda_i > 0$, $\sum_i \lambda_i = 1$.

Lemma 2.1.3. *Consider problem SCO on a simplex with vertices* $v_1, v_2, \ldots, v_h$ *with corresponding* B_i*. Let* $x^* = V\lambda$ *be an interior maximum point of SCO.*

$$d_p(x^*, v_1) = d_p(x^*, v_i), \;\; i = 2, \ldots, h \tag{2.15}$$

i.e. the power function of each vertex has the same value in x^**.*

Proof. Let the index set I of active vertices be defined by $d_p(x^*, v_i) = \varphi(x^*)$ for $i \in I$. Assume that the statement is not true: x^* is interior maximum point and $\exists j, d_p(x^*, v_j) > \varphi(x^*)$ or $j \notin I$. In that case, one can construct a direction u which is the projection of $v_j - x^*$ on the linear space where the active set remains active; $d_p(x^* + u, v_i) = \varphi(x^* + u)$. Direction u is an ascent direction of φ in the point x^* and feasible, because x^* is interior. This is a contradiction with x^* being a maximum point. □

The so-called θ point satisfying equation (2.15) fulfills the necessary condition for an interior optimum. Notice that θ can also be located outside S. The recipe to compute the θ point equating planes is derived from setting the power function values equal:

$$(\theta - v_1)^T(\theta - v_1) - r_1^2 = (\theta - v_i)^T(\theta - v_i) - r_i^2, \;\; i = 2, \ldots, h. \tag{2.16}$$

Elaboration and bringing the terms with θ to the left hand side gives

$$2(v_i - v_1)^T\theta = r_1^2 - r_i^2 + v_i^T v_i - v_1^T v_1, \;\; i = 2, \ldots, h. \tag{2.17}$$

Equation (2.17) describes the plane separating v_1 and v_i on equal power function value. The edge $(v_i - v_1)$ is perpendicular to the plane and the right hand side of the equation describes a constant. Focusing on a solution θ, one can read (2.17) as linear equation $(v_i - v_1)^T\theta = constant$. As point θ has to be in all the planes (2.17) for $i = 2, \ldots, h$, we have $h - 1$ linear equalities in either $\theta = V\lambda$ or in λ which is h-dimensional:

$$2(v_i - v_1)^T V\lambda = r_1^2 - r_i^2 + v_i^T v_i - v_1^T v_1, \;\; i = 2, \ldots, h. \tag{2.18}$$

We are interested in the intersection with the plane through the points $v_1, v_2, \ldots, v_h$. It means that θ should move in that plane, such that the last linear equality is given by

$$(1, 1, \ldots, 1)^T\lambda = 1. \tag{2.19}$$

This means that finding θ can be done by solving a set of h linear equalities. Given an interior θ, we have the unique maximum point of SCO and one can simply check $d_e(\theta, v_1)^2 \leq r_1^2$ to verify the solution of SC.

Example 2.1.4. Consider the following instance of three spheres in 2-dimensional space (see Figure 2.3):

$$v_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \; v_2 = \begin{pmatrix} 5 \\ 0 \end{pmatrix}, \; v_3 = \begin{pmatrix} 3 \\ 6 \end{pmatrix}, \; r_1^2 = 4, r_2^2 = 3, r_3^2 = 1.$$

Point $\theta = \begin{pmatrix} 2.6 \\ 2.7 \end{pmatrix}$ can be determined equating the two power planes: Π_{12} between v_1 and v_2 and Π_{13} between v_1 and v_3. The corresponding solution $\lambda = (0.3, 0.25, 0.45)^T$ is in the interior of the corresponding simplex. The point is on equal power function value $\varphi(\theta) = 10.05$ for the three vertices.

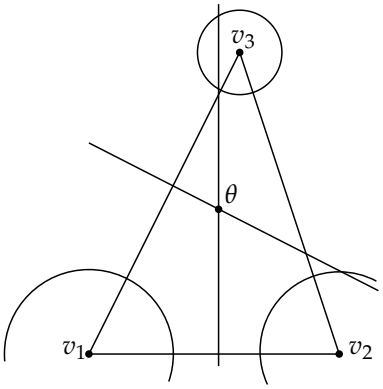


Figure 2.3: Graphical illustration of Example 2.1.4

Notice that if θ is covered by one of the spheres, it is also covered by all of them; $\theta \in B_j$ gives $\theta \in \cap B_j$. This is related to an earlier found result that says that if $\exists x \in S$ covered by all the spheres then the simplex is covered. It means that if θ is covered by one of the spheres then the simplex is fully covered. The result as proven in [37], is the following. Let *conv* denote the convex hull of a set.

Lemma 2.1.5. *Given polytope $S = \text{conv}\{v_1, \ldots, v_l\}$, with $v_1, \ldots, v_l$ the extreme points of S (vertices) with corresponding spheres B_j, $j = 1, \ldots, l$. If $\exists y \in \cap_j B_j \cap S$ then $S \subset \cup_j B_j$.*

Figure 2.4 shows illustrative examples of 2-simplices. In Figure 2.4(a) S is not covered because the θ point is not covered. Figure 2.4(b) illustrates a simplex which is covered.

The next question is what to do if the computed θ is not an interior point of S; i.e. $\exists j \; \lambda_j \leq 0$, at least one of the λ_j is not positive. It means that the maximum point x^* of SCO is found at the boundary of S. We will investigate this further in Subsection 2.1.3.

Focusing on θ, after finding θ, it may be outside S, $\theta \notin S$, but covered by all spheres, $\theta \in \cap_j B_j$. The following lemma combined with Lemma 2.1.5, shows that this is also a sufficient condition to certify the covering of S and no further analysis is required. It means that we do not need to solve problem SCO.

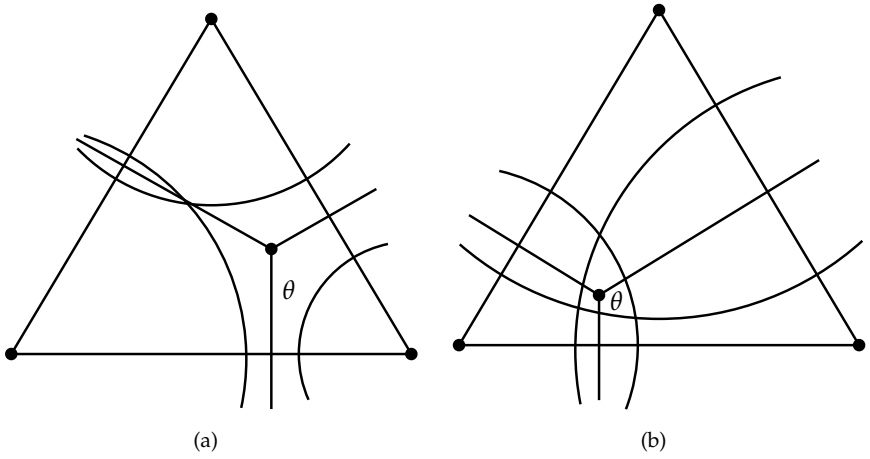


Figure 2.4: Two instances of SC with corresponding θ point.

Lemma 2.1.6. *Given a point $x = V\lambda$, $x \notin S$ and $x \in \cap_j B_j$. Then $\exists y \in \cap_j B_j \cap S$.*

Proof. Without loss of generality, let $\lambda_1, \ldots, \lambda_m \geq 0$ and $\lambda_{m+1}, \ldots, \lambda_h < 0$, so that $v_{m+1}, \ldots, v_h$ are at the other side of the facet where x is closest to. We now make an orthogonal projection of $(y - v_1) = P(x - v_1)$ on the closest facet. One can do so by constructing an $n \times (m-1)$ matrix X with elements $v_j - v_1$ for $j = 2, \ldots, m$. An important observation is that in the construction of X for the starting vector v_1, one can take any $v_i, i = 1, \ldots, m$. The projection $y = v_1 + X(X^TX)^{-1}X^T(x - v_1)$ results into $y \in S$. Notice that $\|y - v_i\|^2 = \|P(x - v_i)\|^2 \leq \|x - v_i\|^2, i = 1, \ldots, m$ and y is closer to $v_j, j = m+1, \ldots, h$. So $(y - v_j)^T(y - v_j) \leq (x - v_j)^T(x - v_j)\ \forall j$ such that $y \in \cap B_j$. □

A consequence of combining Lemmas 2.1.2-2.1.6, is the following theorem:

Theorem 2.1.7. *Let θ be a solution of (2.18) and (2.19). If $\varphi(\theta) \leq 0$, then S is covered.*

Notice that this theorem is valid even for $\theta \notin S$. However, if $\theta \notin S$ and $\varphi(\theta) > 0$, we have to investigate the boundary solutions of SCO in a more detailed way. As shown in the next section, also there the θ-point plays a role.

2.1.3 SCO has a boundary optimum

If the maximum of φ is not interior in S, it can be found on its boundary. In an extreme case, the maximum can be in a vertex of S if one big sphere covers all the simplex S. Alternatively, maximum point x^* can be attained at an edge, a triangular set or even higher dimensional face C of S. Again the θ-point plays a big role in the determination of face C. Consider the set $T = conv\{v_1, \ldots, v_h, \theta\}$. Now one can see that

$$\theta = \text{argmax}_{x \in T}\, \varphi(x) \tag{2.20}$$

because for all $x \in T$, φ increases in the direction of θ such that θ is the global maximum point of (2.20). Moreover, θ is the common vertex of the power diagram. As $S \subset T$ also $\varphi(\theta) \geq \Phi$. That reconfirms the argumentation that a negative $\varphi(\theta)$ tells us that S is covered.

If $\varphi(\theta)$ is positive, we are still interested in solving SCO. Figure 2.5 illustrates a simplex for which the θ point is located outside. The corresponding values of λ_1 and λ_2 are positive and λ_3 has a negative value. S is covered because the intersection of S with the power planes are covered.

The previous analysis also helps us to determine the binding facet C of S where maximum points x^* with $\Phi = \varphi(x^*)$ can be found. Let $\lambda_1, \ldots, \lambda_m \geq 0$, then the set C where the maximum Φ is attained is given by $C = conv\{v_1, \ldots, v_m\}$. It means that for a (local or global) maximum point $x^* = V\mu$ of SCO $\mu_{m+1}, \ldots, \mu_h = 0$. As $h - m$ elements of μ have a value of zero, SCO is now equivalent to

$$\max_C \varphi(x) = \max\{\varphi(\hat{V}\hat{\mu}) |\ \sum_1^m \hat{\mu}_j = 1, \hat{\mu} \geq 0\}, \tag{2.21}$$

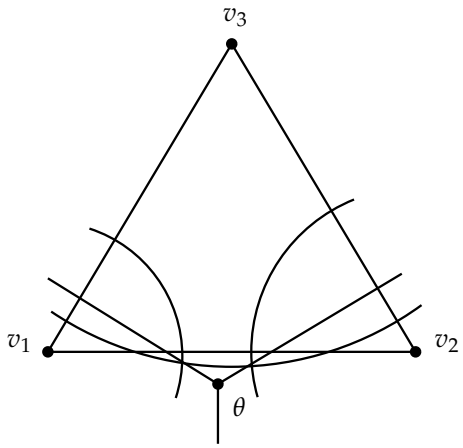


Figure 2.5: Instance of SC with corresponding θ point not covered and outside of S

where $\hat{\mu}$ and $\hat{V}$ now consist of the first m elements of μ and V, respectively.

One can consider the problem as a lower dimensional covering verification. Unfortunately, we are not dealing with an equivalent SC problem. Besides the spheres $B_1, \ldots, B_m$ we have to deal with the cut $\hat{B}_j$ of spheres $B_j, j = m+1, \ldots, h$ with the plane of C. We will describe the plane and the intersection of the spheres that correspond to points that are not vertices of face C.

The plane can be described by

$$\Gamma = \{x \in \mathbb{R}^n | x = v_1 + Xy, y \in \mathbb{R}^{m-1}\}, \tag{2.22}$$

where X is again the matrix $X = [v_2 - v_1, \ldots, v_m - v_1]$. The orthogonal projection of $v_{m+1}, \ldots, v_h$ on Γ via projection matrix $P = X(X^T X)^{-1} X^T$ gives centre points $\hat{v}_j = v_1 + P(v_j - v_1)$. Notice that due to the projection, the power function value of a point $z \in \Gamma$ to one of the vertices v_j is $\|z - v_j\|^2 = \|z - \hat{v}_j\|^2 + \|\hat{v}_j - v_j\|^2$. This means that the radii of the cuts of the spheres with plane Γ are given by $\hat{r}_j^2 = r_j^2 - \|\hat{v}_j - v_j\|^2$. A negative value means that B_j does not cut through plane Γ, so it does not have to be taken into account.

Problem (2.21) is lower dimensional than the original SC problem. Unfortunately, it is not easier as now we are dealing with a problem with m vertices and $h - m$ other centres in the plane, not necessarily in C. We know that the maximum of φ over C can be found on the vertices of the power diagram in C or at the intersection of the power diagram and the boundary of facet C. The power planes have the shape

$$2(v_i - v_j)^T \hat{V} \hat{\mu} = \hat{r}_i^2 - \hat{r}_j^2 + \hat{v}_i^T v_i - v_j^T v_j \to a_k^T \hat{\mu} = b_k, \;\; k = 1, \ldots, K, \tag{2.23}$$

where not all $K = \frac{1}{2}h(h-1)$ power planes are of interest. One can argue that the power planes in between the vertices of C are of less interest, as x^* is not situated on those power planes and at the boundary of C, as one can improve the value of φ by going to the interior.

Example 2.1.8. Consider the following instance of 5 spheres in 6-dimensional space:

$$v_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, v_2 = \begin{pmatrix} 5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, v_3 = \begin{pmatrix} 3 \\ 6 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, v_4 = \begin{pmatrix} 2 \\ 2 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}, v_5 = \begin{pmatrix} 4 \\ 3 \\ 1 \\ 0 \\ 2 \\ 1 \end{pmatrix}$$

and $r_1^2 = 4, r_2^2 = 3, r_3^2 = 1, r_4^2 = 9, r_5^2 = 15$. Now solving system (2.18) and (2.19) gives

$$\lambda = (1.342, 0.847, 1.344, -2.233, -0.3)^T.$$

The negative values in the vector mean $\theta = V\lambda = (2.6, 2.7, -0.3, -2.233, -2.833, -2.533)^T$ is situated outside simplex S. It is at the same weighted power function value $\varphi(\theta) = 29.57$ of $v_1, \ldots, v_5$, so θ is not

covered. The positive values in the λ vector indicate that we can look for the maximum of φ on the convex hull C of the first 3 vertices, as given by problem (2.21). Correspondingly, the cut of B_4 and B_5 can be found by projections $\hat{v}_4 = (2,2,0,0,0,0)^T$, $\hat{v}_5 = (4,3,0,0,0,0)^T$ and reduced radii $\hat{r}_4 = 6$ and $\hat{r}_5 = 9$. Figure 2.6

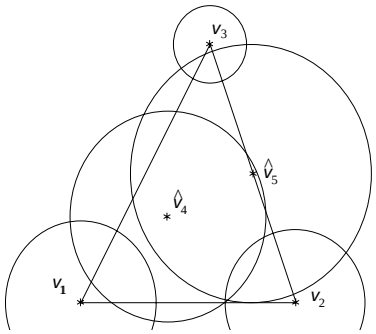


Figure 2.6: Cross-cut of example 2 with the plane through v_1, v_2, and v_3.

gives a 2D impression. The figure does not show us exactly whether C is covered. However, using a solver one can verify that the global maximum $\Phi = -0.22$ can be found in the point $x^* = (3.333, 0, 0, 0, 0, 0)^T$, and this means that S is covered.

Global optimum x^* is attained at one of the feasible basic solutions of a polyhedral set which describe the vertices of the power planes intersected with set C. To describe the polyhedral set we introduce free slack variables z_k of the power planes and define set H as

$$H = \{\hat{\mu} \in \mathbb{R}^m, z \in \mathbb{R}^K | \ \sum_1^m \hat{\mu}_j = 1, \hat{\mu} \geq 0, a_k^T \hat{\mu} - b_k = z_k, k = 1, \ldots, K\}. \tag{2.24}$$

As φ is piecewise convex, it will have local optima at the feasible basic solutions of H, but one does not know which one corresponds to the global maximum Φ. This shows that SC is a problem which is hard to solve. Mainly this will be the case for instances with a large number of vertices representing an irregular simplex where none of the spheres covers other vertices.

2.1.4 The regular case

The question is whether the SC problem is easier to solve for regular simplices. The first check in the verification is of course to see whether one of the radii is big enough to cover the complete simplex. The next step is to generate the θ point and to check whether it is covered. If the latter is not the case, one should consider problem (2.21) on the lower dimensional plane C. The regular case reveals a specific property here. Due to the equal distance of a vertex $v_j, j = m+1, \ldots, h$ to all vertices $v_i, i = 1, \ldots, m$, the projection $\hat{v}_j = Pv_j$ on C also gives an equal distance point. This means that all projected vertices $\hat{v}_j$ coincide at the centroid of C.

Example 2.1.9. Consider the following instance of 5 spheres in 5-dimensional space:

$$v_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, v_2 = \begin{pmatrix} -3 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix}, v_3 = \begin{pmatrix} -3 \\ 0 \\ 3 \\ 0 \\ 0 \end{pmatrix}, v_4 = \begin{pmatrix} -3 \\ 0 \\ 0 \\ 3 \\ 0 \end{pmatrix}, v_5 = \begin{pmatrix} -3 \\ 0 \\ 0 \\ 0 \\ 3 \end{pmatrix}$$

and $r_1^2 = 4, r_2^2 = 4, r_3^2 = 4, r_4^2 = 15, r_5^2 = 12$. The power function value between each pair of points is 18.

Now solving system (2.18) and (2.19) gives $\lambda = (.41, .41, .41, -.2, -.0333)^T$. Point $\theta = V\lambda = (-1.7667, 1.2333, 1.2333, -.6, -.1)^T$ is outside simplex S at the same weighted power function value $\varphi(\theta) = 2.533$ from $v_1, \ldots, v_5$, so θ is not covered. Projecting now via the P matrix determined by $X = [v_2 - v_1, v_3 - v_1]$ gives the centroid of $C = conv\{v_1, v_2, v_3\}$: $\hat{v}_4 = \hat{v}_5 = (-2, 1, 1, 0, 0)^T$. In the reduced problem, only $\hat{r}_4^2 = 15 - \|\hat{v}_4 - v_4\|^2 = 15 - 12 = 3$ is relevant, as $\hat{r}_5 = 0$. Due to the symmetry in this instance, the global maximum $\Phi = -1.2778$ is attained at several points on the boundary; vector $\hat{\mu}$ shows a permutation of the values $0, 0.611$ and 0.389.

2.1.5 Conclusions

The Simplex Cover problem to determine whether a given simplex in n dimensional space is covered by h spheres centered at its vertices has been investigated. It is shown that this problem is equivalent to solving a Global Optimization problem SCO. The following has been found.

- Depending on the instance, SCO may have a unique interior optimum x^* which equals the so-called vertex point θ of a power diagram. A procedure is described to find this θ point.
- If the θ point is covered by the spheres, the simplex is covered, independently of θ being located in or outside the simplex.
- If SCO has a boundary optimum it may have local non-global optima.
- In the latter case, the θ point determines the face C of the simplex where the global optimum points of SCO can be found. A global optimum point is a feasible basic solution of a polyhedral set which is determined by C and so-called power planes.
- In the latter case, the SC problem appears to be equivalent to a problem of covering C with more spheres than the ones centered at the vertices.
- For a regular instance with all equal distances between vertices, the optimum point of SCO is either unique, or SCO is equivalent to the question of covering C with the spheres at its vertices plus a sphere at its centroid.

2.2 Simplex refinement using regular simplices

This work was motivated by a mixture design problem, where one looks for mixtures composed of a maximum number n of raw materials, and the product can be represented by a vector $x \in \mathbb{R}^n$, which meets certain requirements. The set of possible mixtures is mathematically defined by the n-dimensional unit simplex

$$\Delta = \{x \in \mathbb{R}^n \mid \sum_{j=1}^{n} x_j = 1;\ 0 \leq x_j \leq 1\ \forall j\}, \tag{2.25}$$

also called a standard $(n-1)$-simplex, where the variables x_j represent the fraction of the components in a product x. Here, we use a different notation, as the vertices are unit vectors. Notice also, that in this case the dimension of the simplex and the dimension of the space is always the same, which is important. Thus, instead of differentiating the number of vertices h from the dimensions, we will use directly n for both.

In mixture design (blending) problems, the objective is to minimize the cost of the material. As discussed by [105], quadratic quality requirements, minimum dose constraints, looking for robust solutions etc. challenge the search for the best mixture design. This problem is dealt with in [105] where a branch-and-bound approach over simplices is derived using specific discarding tests introduced in [36, 37], described in Section 2.1.

One approach to find solutions for the problem is defining a regular grid with G equidistant grid points for each axis, resulting in a mesh size of $\alpha = \frac{1}{G-1}$ relative to the size $w(\Delta)$ of a simplex Δ defined by the length of its longest edge. A strategy to evaluate all grid points is not appealing, as it is not efficient. When performed on a unit box, the number of function evaluations grows exponentially with the dimension: G^n. This is not that bad on the unit simplex, as we are dealing with the mixture equality, see (2.25) and Figure 2.7. It is shown in [37] that the total number of points on the grid is given by

$$\sum_{k=1}^{n-1} \left(\begin{array}{c} G \\ k \end{array} \right) \left(\begin{array}{c} n-2 \\ k-1 \end{array} \right). \tag{2.26}$$

This means that the number of points increases rapidly. For the example in Figure 2.7, $n = 3$ and $G = 5$, we have 15 grid points and an "accuracy" of $\alpha = 0.25$ relative to the size of the unit simplex.

An appealing alternative for solving blending problems is to use a branch-and-bound strategy with the goal not to generate all the points, but to divide the area and avoid visiting those regions (subsets) which are known not to contain an optimal solution.

Over the unit simplex, the B&B method starts with Δ as the first element of a list Λ of subsets and stops when the list Λ is empty. A generated subset is not stored on Λ if it can be proved that it is infeasible

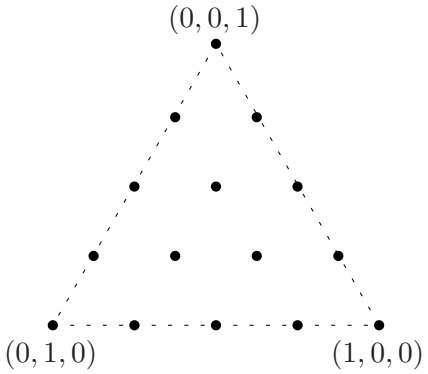


Figure 2.7: A regular grid over the 3-dimensional unit simplex (2-simplex), $G = 5$

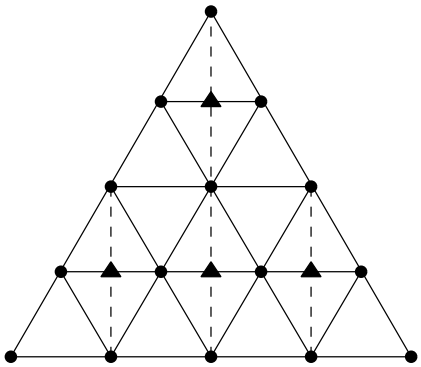


Figure 2.8: Longest edge bisection process with stopping criterion $w(S) \leq \varepsilon \cdot w(\Delta)$ for a 3D-simplex S. Dots represent generated vertices on the regular grid where $\alpha = \varepsilon = 0.25$, while the triangles represent additionally generated sample points.

and/or cannot contain a solution. The termination rule finishes the search for subsets smaller in size than ε times the size of the initial feasible area.

The branching concerns the further refinement of the division. This means that one of the subsets is selected to be split into new subsets. The use of simplicial sets in B&B and several ways of splitting them has been studied extensively in [43, 110]. Bisection of the longest edge of the selected simplex has the advantage that the sets never get a needle shape. Starting with the unit simplex, for all the generated simplices the length of the longest edge is at most twice the size of the shortest edge. Figure 2.8 sketches the idea of the bisection algorithm. It can be observed that points on a regular grid are generated, but that the bisection also generates edges (dotted lines) in at least one additional direction other than the facets of the unit simplex. The values of the coordinates of all generated points are a multiple of $\frac{1}{2^K}$, where K is an integer related to the depth of the search tree.

Simplices and their shape are relevant when dealing with simplicial mesh refinements [120]. Simplices are used in branch-and-bound on the box constrained feasible area using a face-to-face partition [100, 146]. We focus here on optimization over a simplicial shaped feasible area [54, 55]. It has been disputed in [103] and [111] that "round" simplices have a good shape for bounding purposes. The research question is whether a refinement can be done in the B&B method that uses regular simplices instead of bisection. How can such a refinement be realised and what is the effect on the number of simplices to be evaluated and the number of generated sample points (vertices)?

To investigate this question, Subsection 2.2.1 looks into the theoretical properties of a subdivision with regular simplices. Subsection 2.2.2 focuses on how to promote vertex sharing in uniform simplex covering and how to make vertices lay on a predefined grid. Subsection 2.2.3 compares the different ways of subdivision in terms of computational cost if the complete B&B tree is generated. Subsection 2.2.4 summarizes our findings.

In this work, the main idea and the majority of the results were my work, but of course helped and improved by my coauthors. The computational results were done by Leocadio G. Casado.

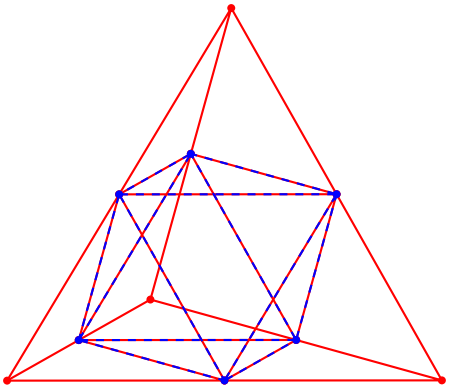


Figure 2.9: A simplex cannot be partitioned into regular simplices for $n > 3$. Cutting of regular simplices from the vertices of the 4D-simplex, an octahedron remains in the middle.

2.2.1 Regular subdivision

In general, a simplicial set is defined as in (2.1), but we can formulate it as

$$S = \{x \in \mathbb{R}^n : x = V\lambda, \lambda \in \Delta\}, \tag{2.27}$$

where Δ is the unit simplex defined in (2.25) and the vertices v_j in matrix V are affine independent. Alternatively, one can write a simplicial set as the convex hull of the vertices, $conv(V)$, as before. The viewpoint of writing a simplex as in (2.27) for a regular simplex, shows that from a subdivision perspective we can focus simply on the unit simplex. A subdivision of Δ can be translated in the partitioning of any simplicial set by multiplication with the vertex matrix V.

Our question is how one can refine Δ using regular simplices. To illustrate that for $n > 3$ a simplex cannot be partitioned into regular simplices, see Figure 2.9. Therefore, we introduce the Uniform Simplex Cover (USC) where the simplex is covered by equally sized, equally oriented overlapping subsimplices and we analyse its characteristics. To express the idea of equally sized and oriented simplices we introduce the following concepts. Each simplex has a center c and a radius r which define its vertex matrix as

$$V = c\mathbf{1}^T + rD, \tag{2.28}$$

where $\mathbf{1}$ is the all ones vector and $E = (e_1, \ldots, e_n)$ represents the identity matrix. $D = (d_1, \ldots, d_n) = E - \frac{1}{n}\mathbf{1}\mathbf{1}^T$ is a symmetric matrix with the directions from the center towards the regular simplex vertices. Notice that the center of the unit simplex Δ is $\frac{1}{n}\mathbf{1}$, whereas the radius of the unit simplex, that is the step size relative to the deviation matrix D is 1.

In order to study overlapping behaviour, our first question is how to determine that a regular simplex is enclosed (covered) by another.

Theorem 2.2.1. *Let S_1 and S_2 be simplices with vertex matrices $V = c_1\mathbf{1}^T + r_1D$ and $W = c_2\mathbf{1}^T + r_2D$. Simplex S_2 is covered (enclosed) by simplex S_1 if*

$$c_{2j} - c_{1j} + \frac{r_1 - r_2}{n} \geq 0 \quad j = 1, \ldots, n. \tag{2.29}$$

Proof. Notice that due to the convexity of S_1, if the vertices w_i of S_2 can be written as a convex combination of the vertices of S_1, then $x \in S_2 \Rightarrow x \in S_1$. Now consider λ_{ij} being an element j of vector λ_i. Vertex $w_i \in S_1$ if $\exists \lambda_i \geq 0, \sum_j \lambda_{ij} = 1$ so that

$$w_i = V\lambda_i. \tag{2.30}$$

This implies

$$\begin{aligned} w_i = c_2 + r_2d_i = (c_1\mathbf{1}^T + r_1D)\lambda_i = c_1 + r_1D\lambda_i \Rightarrow \\ r_1\lambda_i = c_2 - c_1 + r_2(e_i - \frac{1}{n}\mathbf{1}) + r_1\frac{1}{n}\mathbf{1} = c_2 - c_1 + r_2e_i + \frac{r_1 - r_2}{n}\mathbf{1}. \end{aligned} \tag{2.31}$$

Given (2.31), from condition (2.29) follows that $\forall i, j\ \lambda_{ij} \geq 0$. That means all vertices of S_2 can be written as a convex combination of the vertices of S_1 and therefore $x \in S_2 \Rightarrow x \in S_1$. □

Theorem 2.2.1 also shows that radius $r_1 \geq r_2$ is a necessary condition for S_2 to be a subset of S_1.
A refinement of the unit simplex Δ is a collection

$$R = \{S_1, \ldots, S_p\}. \tag{2.32}$$

The question of using identical subsimplices is which centers $c_1, \ldots, c_p$ and radius r to use such that Δ is covered by the refinement R:

$$\Delta \subset \bigcup_{i=1}^{p} S_i. \tag{2.33}$$

In the following we investigate a choice where m centers are distributed in a regular way along each edge. In this way, we get a subdivision R that covers Δ in a uniform way. We call this mUSC. Figure 2.10 sketches the subdivision for $m = 2$ for a 3D-simplex and a 4D-simplex. Figure 2.11 gives an impression of using $m = 4$ centers along each axis.

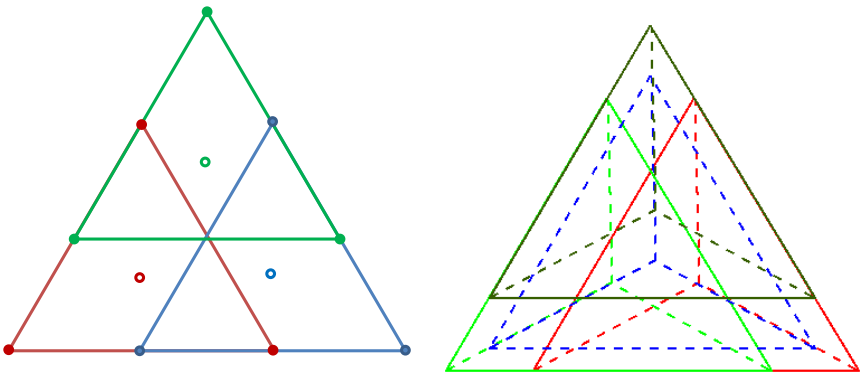


Figure 2.10: 2USC refinement of a 3D-simplex (left) and a 4D-simplex (right)

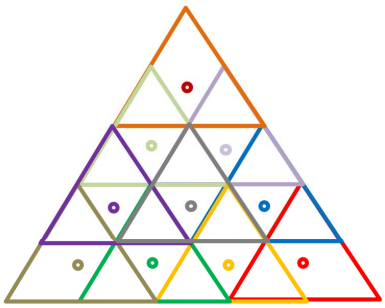


Figure 2.11: 3D-simplex refined according to 4USC into 10 subsimplices

2.2.1.1 Covering by n simplices, 2USC

We first investigate 2USC and define the centers and radius of the subsimplices. Given a simplex represented by its center c and radius r, the 2USC subdivision R consists of n subsimplices S_i, $i = 1, \ldots, n$ with a reduction in size (and radius) of $\beta \geq \frac{n-1}{n}$. Each subsimplex S_i has a radius $r_i = \beta r$ and its center is located at

$$c_i = \beta c + (1-\beta)v_i = \beta c + (1-\beta)(c + rd_i) = c + (1-\beta)rd_i. \tag{2.34}$$

For the unit simplex using the reduction $\beta = \frac{n-1}{n}$

$$c_i = \frac{n-1}{n^2}\mathbf{1} + \frac{1}{n}e_i. \tag{2.35}$$

Example 2.2.2. Consider the subsimplex S_1 of the unit simplex using the reduction $\beta = \frac{n-1}{n}$. Radius $r_1 = \frac{n-1}{n}$ while the center is $(\frac{2n-1}{n^2}, \frac{n-1}{n^2}, \ldots, \frac{n-1}{n^2})^T$. Subsimplex S_1 has the vertices $(1, 0, \ldots, 0)^T, (\frac{1}{n}, \frac{n-1}{n}, 0, \ldots, 0)^T, \ldots, (\frac{1}{n}, 0, \ldots, \frac{n-1}{n}, 0)^T, (\frac{1}{n}, 0, \ldots, 0, \frac{n-1}{n})^T$.

Theorem 2.2.3. *The 2USC refinement R of the unit simplex Δ covers Δ; if $\beta \geq \frac{n-1}{n}$ then $\forall x \in \Delta\ \exists S_i \in R$ such that $x \in S_i$.*

Proof. For each $x \in \Delta$ we have to find a simplex $S_i \in R$ and show that x can be written as a convex combination λ of its vertices. Let $x \in \Delta$ and e_i with $i \in \operatorname{argmax}_{j=1,\ldots,n} x_j$ be the vertex of Δ closest to x. Notice that $\max x_j \geq \frac{1}{n}$ as $\sum x_j = 1$.
We show that S_i defined by (2.28) and $r_i = \beta$ contains x. Due to the definition of vertices in (2.28),

$$\sum_{j=1}^{n} \lambda_j v_j = c_i + \beta \sum_{j=1}^{n} \lambda_j d_j = \frac{\beta}{n}\mathbf{1} + (1-\beta)e_i + \beta \sum_{j=1}^{n} \lambda_j (e_j - \frac{1}{n}\mathbf{1}). \tag{2.36}$$

Simplifying using $\sum \lambda_j = 1$, we have

$$\sum_{j=1}^{n} \lambda_j v_j = (1-\beta)e_i + \beta \sum_{j=1}^{n} \lambda_j e_j. \tag{2.37}$$

Take $\lambda_j = \frac{1}{\beta} x_j$ for $j \neq i$ and $\lambda_i = \frac{1}{\beta} x_i + 1 - \frac{1}{\beta}$. Substitution of the values for λ in (2.37) gives

$$\sum_{j=1}^{n} \lambda_j v_j = (1-\beta)e_i + \beta \sum_{j=1}^{n} \frac{1}{\beta} x_j e_j + \beta(1 - \frac{1}{\beta})e_i = \sum_{j=1}^{n} x_j e_j = x. \tag{2.38}$$

Clearly $\lambda_j \geq 0$ if $j \neq i$, and if $\beta \geq \frac{n-1}{n}$ also $\lambda_i \geq 0$ given that $x_i \geq \frac{1}{n}$. Additionally,

$$\sum_{j=1}^{n} \lambda_j = \frac{1}{\beta} \sum_{j=1}^{n} x_j + 1 - \frac{1}{\beta} = \frac{1}{\beta} + 1 - \frac{1}{\beta} = 1, \tag{2.39}$$

which shows that x can be written as a convex combination of the vertices of one of the subsimplices of R. □

The following proposition, illustrated in Figure 2.10 or 2.12 , gives further insight in Theorem 2.2.3.

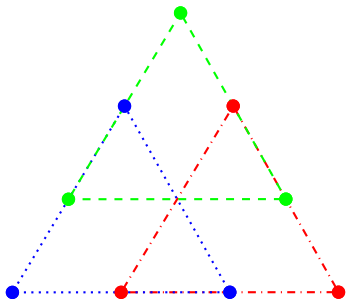


Figure 2.12: Partition of a 3D-simplex using 2USC with $\beta = \frac{n-1}{n}$. Coloured dots are the vertices of the subsimplices drawn with different line styles.

Proposition 2.2.4. *Each subsimplex S_i of R in 2USC of the unit simplex Δ contains the centroid $c = \frac{1}{n}\mathbf{1}$.*

Proof. Centroid $c = \frac{1}{n}\mathbf{1}$ is a convex combination of the vertices of S_i with $\lambda_j = \frac{1}{n-1}$ for $j \neq i$ and $\lambda_i = 0$ using the smallest reduction $\beta = \frac{n-1}{n}$.

$$\sum_{j=1}^{n} \lambda_j v_j = \sum_{j \neq i}^{n} \frac{1}{n-1}(c_i + \frac{n-1}{n} d_j) = \tag{2.40}$$

$$\sum_{j \neq i}^{n} \frac{1}{n-1} \left(\frac{n-1}{n^2}\mathbf{1} + \frac{1}{n} e_i + \frac{n-1}{n}(e_j - \frac{1}{n}\mathbf{1}) \right) = \sum_{j=1}^{n} \frac{1}{n} e_j = c. \tag{2.41}$$

□

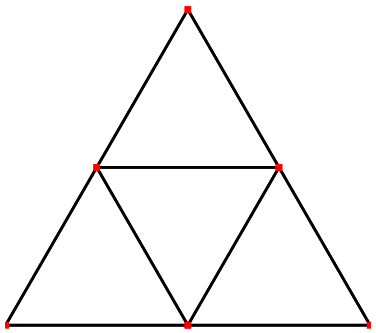


Figure 2.13: 2∇USC for a 3D-simplex with $\rho = \frac{n-2}{n-1}$.

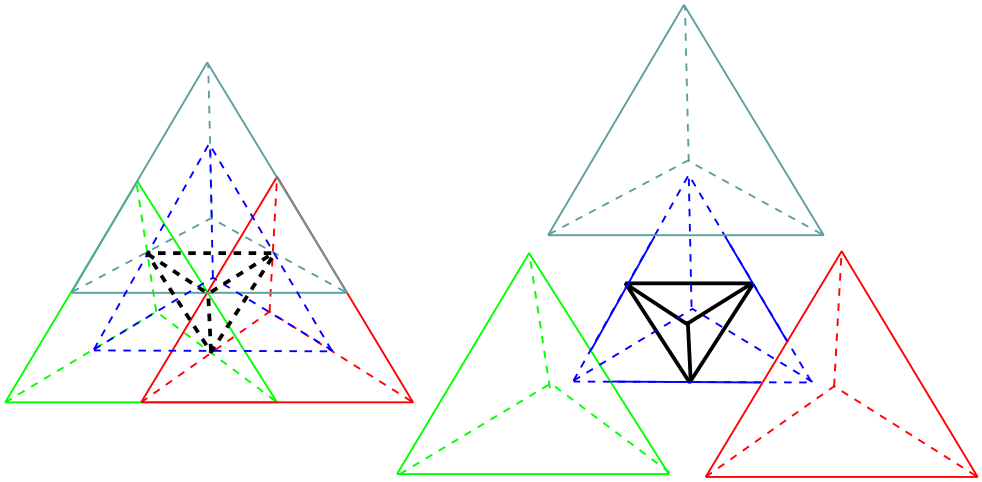


Figure 2.14: 2∇USC for a 4D-simplex with $\rho = \frac{n-2}{n-1}$. The size of the 4 overlapping subsimplices (red, blue, dark green and light green) is $\frac{n-2}{n-1} = \frac{2}{3}$ of the original simplex while the size of the upside down oriented simplex (black) is $n(1-\rho)-1 = 4(1-2/3)-1 = \frac{1}{3}$ of the original simplex.

The 2USC refinement generates a set of overlapping subsimplices that covers unit simplex Δ. How much is the overlap? To answer this question, consider the perspective of volume calculations. The $n-1$ dimensional volume of a simplex is determined by the determinant of the vertex set, projected appropriately on the plane (subspace) where the simplex is located. Without going into detail, from that perspective one should consider that the edges of S_i are β times the length of that of the unit simplex. This means in relative volume that

$$\frac{\text{Vol}(S_i)}{\text{Vol}(\Delta)} = \beta^{n-1}. \tag{2.42}$$

The overlap in terms of volume is n times this factor in relation to the volume of the unit simplex.

2.2.1.2 Refinement using $n+1$ regular simplices, 2∇USC

In our investigation we also found covers that show less overlap, but where we have to relax the property of having equally oriented and sized subsimplices. Consider subsimplices, $\check{S}_i$, $i = 1, \ldots, n$ with the same structure as S_i in 2USC having a radius $\frac{n-2}{n-1} \leq \rho < \frac{n-1}{n}$. The union of $\check{S}_i$ leaves a small "hole" $\check{S}_{n+1}$ in Δ which has an "upside down" orientation with radius $r_{n+1} = n(1-\rho)-1$. $\check{S}_{n+1}$ is defined by the center of the original simplex, $c_{n+1} = \frac{1}{n}\mathbf{1}$ having the vertices $U_{n+1} = c_{n+1}\mathbf{1}^T - r_{n+1}D$ which are the centroids of the facets of Δ if $\rho = \frac{n-2}{n-1}$ (see Figures 2.13 and 2.14). Subdivision $P = \{\check{S}_1, \ldots \check{S}_{n+1}\}$ now just covers Δ with $n+1$ simplices of smaller size than S_j.

Theorem 2.2.5. *The 2∇USC refinement P of the unit simplex Δ covers Δ if $\frac{n-2}{n-1} \leq \rho < \frac{n-1}{n}$. That is, $\forall x \in \Delta\ \exists \check{S}_i \in P$ such that $x \in \check{S}_i$.*

Proof. For each $x \in \Delta$ we have to find a simplex $\check{S}_i \in P$ and show that x can be written as a convex combination λ of its vertices. Let $x \in \Delta$ and e_i with $i \in \text{argmax}_{j=1,\ldots,n} x_j$ be a vertex of Δ closest to x. We will show that $\check{S}_i$ defined by (2.28) and $r_i = \rho$ contains x, if $x_i \geq 1-\rho$, and $x \in \check{S}_{n+1}$ if $x_i \leq 1-\rho$.

First, we show that $\check{S}_i$ contains x, if $x_i \geq 1-\rho$. One can deduce for u_j and ρ that $\lambda_j = \frac{1}{\rho}x_j$ for $j \neq i$ and $\lambda_i = \frac{1}{\rho}x_i + 1 - \frac{1}{\rho}$ using the same argument as in the proof of Theorem 2.2.3. Using these values, x is a convex combination of the vertices of $\check{S}_i$ according to equations (2.38–2.39). Clearly, $\lambda_j \geq 0$ if $j \neq i$, and $\lambda_i \geq 0$ for $x_i \geq 1-\rho$.

Secondly, we show that the upside down simplex $\check{S}_{n+1}$ contains x, if $x_i \leq 1-\rho$. Take

$$\lambda_j = \frac{1-\rho-x_j}{n(1-\rho)-1} \quad \forall j. \tag{2.43}$$

One can derive that $\lambda_j \geq 0\ \forall j$ if $x_i \leq 1-\rho$ and $\sum_{j=1}^n \lambda_j = 1$ given $\sum_{j=1}^n x_j = 1$. Moreover, for $\check{S}_{n+1}$,

$$\sum_{j=1}^n \lambda_j u_j = c_{n+1} - (n(1-\rho)-1)\sum_{j=1}^n \lambda_j d_j = \frac{1}{n}\mathbf{1} - (n(1-\rho)-1)\sum_{j=1}^n \lambda_j (e_j - \frac{1}{n}\mathbf{1}).$$

Given $\sum_{j=1}^n \lambda_j = 1$

$$\begin{aligned}\sum_{j=1}^n \lambda_j u_j &= (1-\rho)\mathbf{1} - (n(1-\rho)-1)\sum_{j=1}^n \lambda_j e_j \\ &= (1-\rho)\mathbf{1} - \sum_{j=1}^n (1-\rho-x_j)e_j = \sum_{j=1}^n x_j e_j = x.\end{aligned}$$

This shows that x can be written as a convex combination of the vertices of one of the subsimplices of P. □

In the 2∇USC subdivision, one of the subsimplices has an "upside down" orientation. To use the theoretical result of Theorem 2.2.1 to check one upside down simplex is covered by another, we should adapt the result to that situation.

Corollary 2.2.6. *Let $\check{S}_1$ and $\check{S}_2$ be "upside down" simplices defined by vertex matrices $U = c_1\mathbf{1}^T - r_1 D$ and $W = c_2\mathbf{1}^T - r_2 D$. Simplex $\check{S}_2$ is covered by simplex $\check{S}_1$ if*

$$c_{1j} - c_{2j} + \frac{r_1 - r_2}{n} \geq 0 \quad j = 1, \ldots, n. \tag{2.44}$$

Proof. Following the reasoning in the proof of Theorem 2.2.1, let equation (2.31) be modified to

$$\begin{aligned} w_i = c_2 - r_2 d_i = (c_1\mathbf{1}^T - r_1 D)\lambda_i = c_1 - r_1 D\lambda_i \Rightarrow \\ r_1\lambda_i = c_1 - c_2 + r_2(e_i - \frac{1}{n}\mathbf{1}) + r_1\frac{1}{n}\mathbf{1} = c_1 - c_2 + r_2 e_i + \frac{r_1 - r_2}{n}\mathbf{1}. \end{aligned} \tag{2.45}$$

All elements $\lambda_{ij} \geq 0$ under conditions (2.44) and $\sum_j \lambda_{ij} = 1$, which means that all vertices of $\check{S}_2$ are included in $\check{S}_1$. □

2.2.1.3 Refinement using more base points per edge, mUSC

The concept of mUSC means we can extend the cover towards more, but smaller identically oriented and sized simplices, by using more (exactly m) simplices per edge. We investigate the properties of mUSC here.

The mUSC refinement M consists of subsimplices S_i^m with a reduction of $\gamma \geq \frac{n-1}{m+n-2}$ in radius r of the simplex to be refined. Center points are characterised by taking steps from the original center toward the vertices. From the center, the new subsimplex with center $C_{t_1,\ldots,t_n}$ is described by taking in each direction d_i an integer relative step size of t_i where the total of the relative step sizes is $m-1$, $\sum_i t_i = m-1$.

$$C_{t_1,\ldots,t_n} = c + \frac{(1-\gamma)r}{m-1}\sum_{i=1}^n t_i d_i. \tag{2.46}$$

As in the multinomial distribution, there are in total

$$N(m,n) = \binom{m+n-2}{m-1} \tag{2.47}$$

outcome vectors t corresponding to the centers of the subsimplices. Notice that for the specific case $m-1=n$ the center of the subsimplex with $t=\mathbf{1}^T$ i.e. $C_{1,\dots,1}$, corresponds to the original center c.

Example 2.2.7. Consider a 4USC refinement of the 3D-simplex with $\gamma = \frac{n-1}{m+n-2} = \frac{2}{5}$. The following centers of the subsimplices as illustrated in Figures 2.11 and 2.15 are generated: $C_{300} = \frac{1}{3}\mathbf{1} + \frac{3}{5}d_1 = (\frac{11}{15}, \frac{2}{15}, \frac{2}{15})$, $C_{210} = \frac{1}{3}\mathbf{1} + \frac{2}{5}d_1 + \frac{1}{5}d_2 = (\frac{8}{15}, \frac{5}{15}, \frac{2}{15})$, $C_{111} = (\frac{5}{15}, \frac{5}{15}, \frac{5}{15})$ (the original center), $C_{120} = (\frac{5}{15}, \frac{8}{15}, \frac{2}{15})$, $C_{021} = (\frac{2}{15}, \frac{8}{15}, \frac{5}{15})$,...

In both figures there are regions (triangles) which are covered twice, or three times. In Figure 2.15, the regions with light blue are covered by only one simplex, which has its center inside. The middle blue regions are covered by the two simplices whose center enclose it, and there are three dark blue regions which are covered by three simplices. Again, covered by the ones whose centers enclose it.

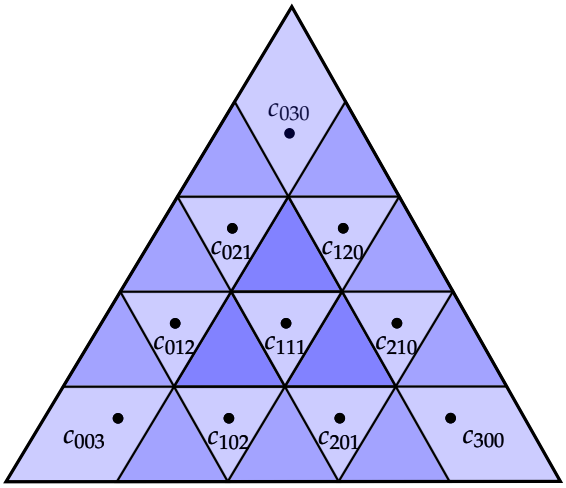


Figure 2.15: 4USC refinement of the unit 3D-simplex with $\gamma = \frac{n-1}{m+n-2} = \frac{2}{5}$

Theorem 2.2.8. *The mUSC refinement M of the unit simplex Δ covers Δ if $\gamma \geq \frac{n-1}{m+n-2}$. That is, $\forall x \in \Delta\ \exists S_i^m \in M$ such that $x \in S_i^m$.*

Proof. Knowing that a valid 2USC subdivision covers Δ, we show that the subsimplices of an mUSC subdivision can also be generated by iteratively using a 2USC subdivision. We first show that a 3USC division is equivalent to sequentially performing two 2USC subdivisions. Using reduction rate β_1 in a first 2USC division, we obtain n subsimplices with centers

$$c_{e_i} = c + (1-\beta_1)d_i \quad i = 1,\dots,n \tag{2.48}$$

and radius β_1. Notice that for 2USC, the vector t corresponds with a unit vector e_i. Now, we 2USC-subdivide all these subsimplices using reduction β_2 obtaining new subsimplices with centers

$$c + (1-\beta_1)d_i + (1-\beta_2)\beta_1 d_j \quad i,j = 1,\dots,n. \tag{2.49}$$

and radius $\beta_1\beta_2$. These centers correspond to the 3USC centers $c_{e_i+e_j}$ by carefully choosing values for reduction factors β_1, β_2 such that $\beta_1\beta_2 = \gamma$. The choice provides the centers

$$c_{e_i+e_j} = c + \frac{1-\gamma}{2}(d_i + d_j) \tag{2.50}$$

corresponding to the 3USC refinement with integer relative step sizes $t_i = 0, 1, 2$. To have (2.49) and (2.50) coincide, the values for β should fulfill $1-\beta_1 = (1-\beta_2)\beta_1 = \frac{1-\gamma}{2}$ such that $\beta_1 = \frac{1+\gamma}{2}$ and $\beta_2 = \frac{2\gamma}{1+\gamma}$.

Given a 3USC value $\gamma \geq \frac{n-1}{m+n-2} = \frac{n-1}{n+1}$ it can be verified that by this choice $\beta_1, \beta_2 \geq \frac{n-1}{n}$ fulfills the condition of Theorem 2.2.3 with respect to the 2USC covering of Δ.

Now we can generalize this to mUSC. Using the same reasoning, after $(m-1)$ 2USC-subdivisions with reduction rates $\beta_1, \ldots, \beta_{m-1}$, the $n^{(m-1)}$ subsimplices have a center of the shape

$$\begin{aligned} &c + (1-\beta_1)d_{i_1} + (1-\beta_2)\beta_1 d_{i_2} + (1-\beta_3)\beta_2\beta_1 d_{i_3} \\ &+ \ldots + (1-\beta_{m-1})\beta_{m-2}\cdots\beta_1 d_{i_{m-1}}, \end{aligned} \tag{2.51}$$

where i_j is a step direction in the j^{th} subdivision. In order to have them coincide with the $N(m,n) = \binom{m+n-2}{m-1}$ centers $C_{t_1,\ldots,t_n}$ specified by (2.46) with integer values for the relative step sizes t_i, the chosen values for $\beta_j \geq \frac{n-1}{n}$ should fulfil the following conditions:

$$\begin{aligned} \gamma &= \beta_1\beta_2\cdots\beta_{m-1} \\ \frac{1-\gamma}{m-1} &= 1-\beta_1 \\ 1-\beta_1 &= (1-\beta_2)\beta_1 \\ 1-\beta_2 &= (1-\beta_3)\beta_2 \\ &\vdots \\ 1-\beta_{m-2} &= (1-\beta_{m-1})\beta_{m-2}. \end{aligned}$$

These conditions imply for a solution $\beta_1, \ldots, \beta_{m-1}$ that

$$\beta_j = \frac{1}{2-\beta_{j+1}} \quad j = 1, \ldots, m-2. \tag{2.52}$$

Given $\gamma \geq \frac{n-1}{m+n-2}$ the elaboration of (2.52) and $\gamma = \beta_1\beta_2\cdots\beta_{m-1}$ leads to

$$\beta_j \geq \frac{m+n-2-j}{m+n-1-j} \quad j = 1, \ldots, m-1. \tag{2.53}$$

Is it easy to check that $\beta_j \geq \frac{n-1}{n}$, $j = 1, \ldots, m-1$, such that the conditions of Theorem 2.2.3 with respect to 2USC covering are valid. The $n^{(m-1)}$ subsimplices cover Δ and coincide with the $\binom{m+n-2}{m-1}$ subsimplices of the mUSC subdivision using a reduction factor $\gamma \geq \frac{n-1}{m+n-2}$. □

Although the same final simplices can be obtained by iterative 2USC-subdivision, the number of generated simplices is smaller using mUSC.

2.2.2 Vertex sharing

From the branch-and-bound perspective, it is convenient that simplices in the search tree share vertices because:

- vertices do not have to be re-evaluated reducing the number of vertex evaluations, and
- the chance that a subsimplex is covered by a larger subsimplex from another branch is greater; this will be shown later in Remark 2.2.11.

Longest edge bisection typically shares vertices. As elaborated by [37], if the user is aiming at a sampling accuracy of ε (relative to the edge size of the unit simplex) the full tree of the bisection algorithm generates sample points as vertices on a grid. The mesh size α of the grid is typically $\alpha = \frac{1}{2^k}$ and correspondingly the number of grid points per axis is $G = 2^k + 1$, with $k = \lceil -\log_2 \varepsilon \rceil$. In a branch-and-bound setting, simplices S with a small size $w(S) \leq \varepsilon \cdot w(\Delta)$ are no longer refined.

The same analysis does not apply directly for USC, as unlike bisection the refinement does not provide necessarily a partition. A subsimplex may contain sampled points apart from its vertices (vertices from other subsimplices). A refinement by 2USC up to accuracy $w(S) \leq \varepsilon \cdot w(\Delta)$ provides sample points that are not necessarily on a grid.

Consider a user goes for the rough accuracy of $\varepsilon = \frac{1}{2}$. Figure 2.16 illustrates the result of an iterative 2USC subdivision with the reduction rate $\beta = \frac{n-1}{n}$ up to the relative size of the reached subsimplices is

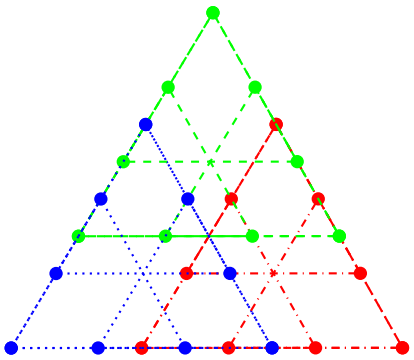


Figure 2.16: 2USC subdivision of a 3D-simplex with $\beta = \frac{n-1}{n}$ up to relative size $\varepsilon = \frac{1}{2}$.

smaller than ε. It can be seen from Figure 2.16 that most of the vertices of a subsimplex are not shared with another subsimplex.

As illustrated in the proof of Theorem 2.2.1, the reduction rates of 2USC can be adapted within certain margins to reach a target. Now, the question is how to do so, such that vertex sharing is promoted. We discuss these aspects in this section.

2.2.2.1 Using the reduction rate to promote vertex sharing

The first question is whether there exists a reduction rate β for 2USC such that after N successive refinements, at least two of the evaluated vertices at every edge of the initial simplex coincide. A condition for this is that $1 - \beta = \beta^N$. Table 2.1 shows possible reduction rates β for different values of the dimension n, such that the condition is fulfilled and $\beta \geq \frac{n-1}{n}$. The number of divisions N increases with n. Although vertex sharing is promoted by choosing a uniform reduction rate β listed in Table 2.1, vertex sharing happens only for a few vertices. Next, we investigate how vertex sharing can be promoted forcing new vertices to lay on a so-called final grid, using more than one reduction rate. We will focus on the so-called 2USC-Grid and 2∇USC-Grid.

Table 2.1: Reduction rate $\beta \geq \frac{(n-1)}{n}$ and number N of divisions leading to vertex sharing of the final subsimplices.

n	$\frac{(n-1)}{n}$	N	β
2	0.5	1	0.5
		2	0.61803
3	$\frac{2}{3}$	3	0.68232
		4	0.72449
4	0.75	5	0.75487
		6	0.77808
		7	0.79654
5	0.8	8	0.81165
		9	0.82430
6	$\frac{5}{6}$	10	0.83508

2.2.2.2 Reduction factors to promote vertices to be on a grid

Given that we have G sample points (including the unit simplex vertices) per edge, we have that grid size $\alpha = \frac{1}{G-1}$ cannot be completely freely chosen, but neither is bounded to the value of $\frac{1}{2^k}$ of the bisection refinement. The number of grid points per edge is at least $G \geq \left\lceil \frac{1}{\varepsilon} \right\rceil + 1$. Forcing points to be on a grid with G points per edge, means that they are at most $\alpha \leq \varepsilon$ apart. This is well illustrated in Figure 2.17 (right). Using iterative 2USC refinement with different reduction rates up to $\varepsilon = \frac{1}{2}$ on the 3D-simplex can generate sample points on a uniform grid that are $\alpha = \frac{1}{4}$ apart due to overlap of regular simplices.

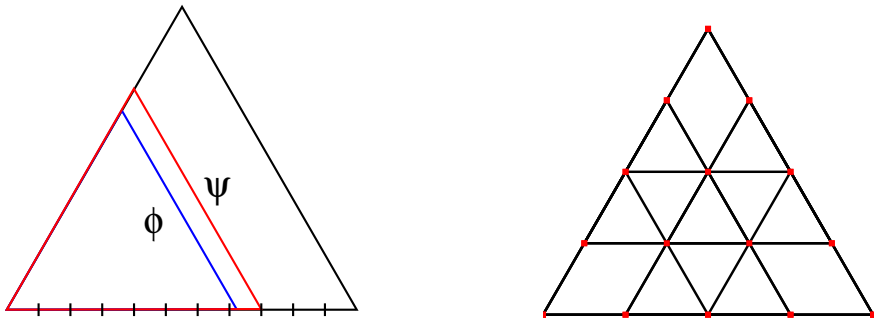


Figure 2.17: On the left-hand side: Choosing the reduction ratio to promote vertices on an $\alpha = \frac{1}{11}$ grid. On the right: Iterative refinement of a 3D-simplex using 2USC-Grid with $\phi = \frac{n-1}{n}$ and threshold value $\varepsilon = \frac{1}{2}$.

In what follows, we elaborate on 2USC-Grid and 2∇USC-Grid methods. The questions are i) what is the value of α and ii) how can we design reduction rates β, ρ of the regular covering methods such that vertices are generated on the corresponding grid?

Consider the first question on the mesh size α of the final grid depending on the regular division method on the unit simplex Δ. Let $\omega = \frac{w(S)}{w(\Delta)}$ be the relative size of a simplex S to be refined at the last but one level of the tree, i.e. the children of S have a size smaller or equal than ε. Let us assume that the size of the last simplices is exactly ε and we use the biggest possible reduction, i.e. the minimum reduction factor. For 2USC, $\varepsilon = \omega\frac{n-1}{n}$ and the desired mesh size $\alpha = \omega(1 - \frac{n-1}{n}) = \frac{\omega}{n} = \frac{\varepsilon}{n-1}$. For 2∇USC, $\varepsilon = \omega\frac{n-2}{n-1}$ and $\alpha = \omega(1 - \frac{n-2}{n-1}) = \frac{\omega}{n-1} = \frac{\varepsilon}{n-2}$. Notice that in both cases $\alpha < \varepsilon$, and the first question is answered. In practice, the size of the last simplex is not necessarily ε, but it can be shown that the final size is $(\lceil \frac{1}{\varepsilon} \rceil)^{-1}$. So for 2USC, $\alpha = \frac{(\lceil \frac{1}{\varepsilon} \rceil)^{-1}}{n-1}$ and for 2∇USC, $\alpha = \frac{(\lceil \frac{1}{\varepsilon} \rceil)^{-1}}{n-2}$.

Let ϕ denote the minimum reduction factor, i.e. $\frac{n-1}{n}$ for 2USC and $\frac{n-2}{n-1}$ for 2∇USC. One can promote the vertices of the refinement to lay on the grid for a certain mesh size α, that depends on ε, by using as reduction factor the value

$$\psi(\omega) = \frac{\left\lceil \frac{\phi\omega}{\alpha} \right\rceil \cdot \alpha}{\omega}, \tag{2.54}$$

answering the second question. The choice of the reduction factor is illustrated on the left-hand side of Figure 2.17. The right hand side of Figure 2.17 shows the final set of simplices using 2USC iteratively with choice (2.54). We call this method 2USC-Grid. Comparing Figure 2.17 with Figure 2.16 shows that one can promote vertices to coincide at grid points.

Remark 2.2.9. The reduction factor range of 2∇USC is in fact $\frac{n-2}{n-1} \leq \rho < \frac{n-1}{n}$ as discussed in Theorem 2.2.5. Formula (2.54) may give a reduction factor $\psi(\omega) \geq \frac{n-1}{n}$. In those cases, the 2∇USC refinement is equivalent to a 2USC refinement, because the upside down simplex disappears.

Remark 2.2.10. It is relatively easy to see that all generated vertices of the 2USC-Grid refinement do lay on a grid with mesh size α. For 2∇USC-Grid, this is not evident. Consider a 2∇USC refinement $\check{S}_1, \ldots, \check{S}_{n+1}$ of a simplex S, where S has vertices on an α-grid. This means that there is a value K such that $\omega = \frac{w(S)}{w(\Delta)} = K \cdot \alpha$. Equation (2.54) provides a reduction factor such that there is another integer $k \leq K$ with $\psi(\omega)\omega = k \cdot \alpha$. For the upside down sub-simplex $\frac{w(\check{S}_{n+1})}{w(\Delta)} = \omega \cdot (n(1 - \psi(\omega)) - 1) = (n-1) \cdot K \cdot \alpha - n \cdot k \cdot \alpha$, i.e. the relative distance between the vertices is also a multiple of α. As its center is the same as that of S, its vertices lay on the grid.

2.2.3 Computational aspects

This section is based on the results obtained by Leocadio G. Casado, but is presented for completeness.

Focusing on the use of partitioning by regular simplices as division rule in a branch-and-bound algorithm, the efficiency of the division rule will depend on the generated search tree. In order to make our study independent of the problem to solve, we investigate the worst case, i.e. none of the branches is pruned during the search. Algorithm 2.1 shows the pseudo code where a complete tree is generated from a division rule up to the size of the subsets smaller than $\varepsilon \cdot w(\Delta)$. Line 8 uses Theorem 2.2.1 or Corollary 2.2.6

Algorithm 2.1 Complete Search Tree

Input: Δ, ε

1: $\Lambda := \{\Delta\}$ ▷ *Set of simplices not yet split*
2: $ns := 1$ ▷ *Number of simplices*
3: **while** $\Lambda \neq \emptyset$ **do**
4: Extract a simplex S from Λ ▷ *Depth-First*
5: **if** $w(S) > \varepsilon \cdot w(\Delta)$ ▷ *Final accuracy not reached*
6: Ω:=Refine(S) ▷ Ω = *set of subsimplices*
7: **for** each $S_i \in \Omega$ **do**
8: **if** $S_i \nsubseteq \tilde{S}$, $\forall \tilde{S} \in \Lambda$
9: $ns := ns + 1$
10: Evaluate S_i
11: Store S_i in Λ

to avoid the evaluation of a simplex which is covered by one to be evaluated in the future. Depth-First search is used in line 4 in order to keep a small size of the set of simplices not yet split ($|\Lambda|$) and therefore the number of covering checks at line 8 of the algorithm is also minimal.

Remark 2.2.11. Due to the use of a Depth-First search in Algorithm 2.1, a simplex S cannot be covered by a simplex $\tilde{S}$ of the same size which has already been evaluated, because in such case simplex $\tilde{S}$ would have been removed earlier due to being covered by either S or its ancestor. Therefore, final simplices are never covered.

Since Algorithm 2.1 generates the complete search tree, the smallest number ns of generated simplices is obtained by applying mUSC just once. In the practical application of a B&B algorithm to a real problem with pruning, usually a smaller branching factor provides less simplices to be evaluated in contrast to the generation of the complete tree. Table 2.2 shows the branching factor for the studied methods.

Table 2.2: Branching factor of different refinement methods.

LEB	2USC	2USC-Grid	2∇USC	2∇USC-Grid	mUSC
2	n	n	$n+1$	n or $n+1$	$\binom{m+n-2}{m-1}$

We investigate the number of evaluated simplices and vertices for the 2USC and 2∇USC variants, because as among regular division methods they have the smallest branching factor. The benchmark is longest edge bisection (LEB) as that method has the smallest branching factor and is most commonly used as division method. Notice that the storage requirement per simplex for bisection is much larger than that of regular refinement, as the latter require only to store the center and the radius.

Table 2.3 shows the number of evaluated simplices to reach the $\frac{w(S)}{w(\Delta)} \leq \varepsilon$ criterion for the studied methods. For regular simplices $\frac{w(S)}{w(\Delta)}$ is the same as the radius of S. Therefore edge lengths calculations are not needed. This termination criterion ensures a relative nearest neighbour distance of sample points smaller than ε. For LEB with $n > 3$ more than one longest edge exists [5]. Here, we select the first one. The relative precision is set to $\varepsilon = \frac{1}{2^k}$, which is convenient for LEB. This termination criterion generates an oversampling beyond the grid for LEB (see Figure 2.8), 2USC (see Figure 2.16) and 2∇USC methods. Refinement 2∇USC shows the least number of evaluated simplices for $n = 3$ because it provides an efficient partition. Specifically for $n = 3$ 2∇USC and 2∇USC-Grid generate the same tree. In general, 2∇USC outperforms 2USC due to having a higher branching factor and less overlap. For all instances, 2USC generates more simplices among the presented methods, whereas 2∇USC-Grid is the method with the best results. 2USC-Grid and 2∇USC-Grid force vertices of simplices to lay on the α-grid using the reduction ratio (2.54).

2∇USC-Grid has two possible branching factors in Table 2.2 as explained in Remark 2.2.9. Comparing these methods, 2∇USC-Grid generates less simplices than 2USC-Grid for all evaluated cases. The reason is that on the one hand, 2∇USC-Grid has a branching factor n or n+1, which is the highest among the

Table 2.3: Number of evaluated simplices for $\varepsilon = \frac{1}{2^k}$.

n	k	LEB	2USC	2∇USC	2USC-Grid	2∇USC-Grid
3	1	11	13	5	10	5
	2	47	100	21	56	21
	3	191	663	85	283	85
4	1	63	79	26	35	19
	2	507	829	371	575	179
	3	4,027	19,407	4,863	6,825	1,660
5	1	379	621	177	126	71
	2	6,319	26,160	3,127	5,176	2,087
	3	99,563	577,509	166,061	117,676	46,987
6	1	2,189	1,265	1,509	462	266
	2	79,035	337,971	127,839	46,457	20,048
	3	2,510,297	26,473,770	5,803,618	2,723,725	878,738
7	1	12,685	13,021	2,704	1,716	1,002
	2	1,074,163	4,840,735	1,634,936	392,876	186,160
	3	71,915,489		300,537,878	49,264,562	20,617,297

presented methods, and on the other hand it provokes less overlap than 2USC. For the complete tree generation (no pruning) a higher branching factor provides less generated simplices.

Table 2.4: Ratio of the number of covered/evaluated simplices for $\varepsilon = \frac{1}{2^k}$.

n	k	2USC	2∇USC	2USC-Grid	2∇USC-Grid
3	1	0.00	0.00	0.30	0.00
	2	0.11	0.00	0.52	0.00
	3	0.19	0.00	0.73	0.00
4	1	0.01	0.00	0.74	0.32
	2	0.30	0.08	1.47	0.48
	3	0.43	0.11	1.81	0.65
5	1	0.18	0.06	1.23	0.70
	2	0.58	0.21	2.24	1.38
	3	0.83	0.29	2.47	1.73
6	1	0.16	0.13	1.73	1.11
	2	0.71	0.41	3.00	2.06
	3	1.12	0.58	3.55	2.22
7	1	0.25	0.12	2.23	1.51
	2	0.84	0.52	3.60	2.71
	3		0.79	4.26	3.30

Table 2.4 shows the ratio of the number of covered versus evaluated simplices (see Algorithm 2.1, line 8) for regular subdivision methods. A covered simplex is dropped as its region will be evaluated by the covering simplex. The ratio is much larger for the grid aware methods. Therefore, forcing simplices to have its vertices on the α-grid reduces the number of simplices to be evaluated.

Table 2.5 shows the number of generated vertices. For $n = 3$, 2∇USC and 2∇USC-Grid do not exhibit overlap and therefore generate the smallest number of vertices. For $n > 3$ the best method in terms of the number of generated vertices is LEB due to the already mentioned fact of creating vertices at coordinates multiple of $\frac{1}{2^k}$, where k is an integer related to the depth of the search tree.

Missing data in Tables 2.3 to 2.5 correspond to cases where the algorithm did not finish in 10 minutes or consumes more than 15 GB of memory.

In order to investigate how the number of evaluated simplices and vertices is affected by a chosen user accuracy ε, we run experiments for 11 values of ε in the range $[0.1, 0.5]$. The tendency has been depicted in Figures 2.18 to 2.22. Missing results mean that the algorithm did not terminate in 10 minutes.

When $n = 3$ (Figure 2.18), 2∇USC-Grid produces the smallest number of simplices and vertices. It is interesting to see that the number of evaluations is constant on some ranges for a varying value of ε.

Table 2.5: Number of generated sample points (evaluated vertices) for $\varepsilon = \frac{1}{2^k}$.

n	k	LEB	2USC	2∇USC	2USC-Grid	2∇USC-Grid
3	1	7	27	6	15	6
	2	19	201	15	45	15
	3	61	1,327	45	153	45
4	1	18	238	84	74	35
	2	92	2,488	1,190	455	165
	3	574	58,222	15,592	2,925	969
5	1	46	2,485	740	340	200
	2	438	104,641	13,140	4,845	1,820
	3	5,105	2,310,037	699,727	58,905	20,475
6	1	108	6,326	7790	1,497	942
	2	1,881	1, 689,856	664,933	53,004	20,349
	3	41,286		30,325,101	1,221,759	435,897
7	1	232	78,127	16,604	6,447	4,137
	2	7,653	29,044,411	10,119,466	584,633	230,209
	3	335,848			25,827,165	9,366,819

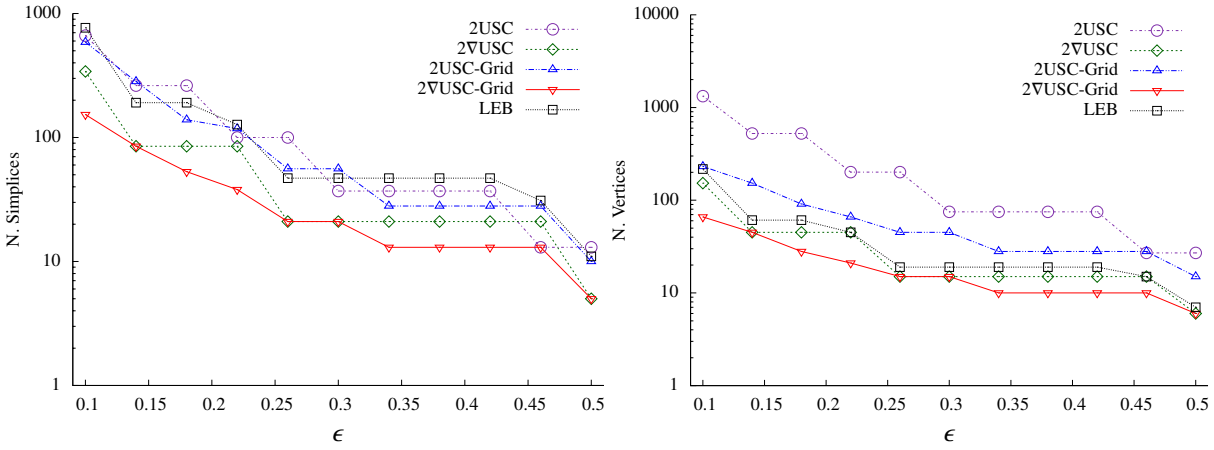


Figure 2.18: Number of evaluated simplices/vertices (left/right); $n = 3$ varying ε

Surprisingly, for many values of ε, 2USC, the worst method in Table 2.3, is generating less simplices than bisection.

For $n = 4$ (Figure 2.19), 2∇USC-Grid produces the smallest number of simplices for most of the instances but surprisingly 2USC is the best in some instances. Regarding the number of vertices, LEB is the best apart for the case where $\varepsilon = 0.1$. This shows that some values of ε are more convenient than others for each method.

For $n > 4$ (Figs. 2.20 to 2.22), 2∇USC-Grid produces the smallest number of simplices and LEB the smallest number of vertices.

Summarizing, the numerical results show that the traditional idea of partitioning the search space can sometimes be replaced by using an overlapping refinement that generates vertices on a grid. Being aware of a grid mesh, the number of evaluated simplices is reduced by several orders of magnitude. However, the number of vertices evaluated is in general worse than using longest edge bisection. When the computational cost of simplex and vertex evaluations is similar, 2∇USC-Grid is preferred.

Still, the sampling is actually too accurate in terms of a nearest neighbour distance because the reached inter-point distance is much smaller than the required user accuracy $\varepsilon \cdot w(\Delta)$. For USC methods, one could use much larger final simplices. Determining their size will reduce the number of evaluated simplices and vertices. This remains an open question.

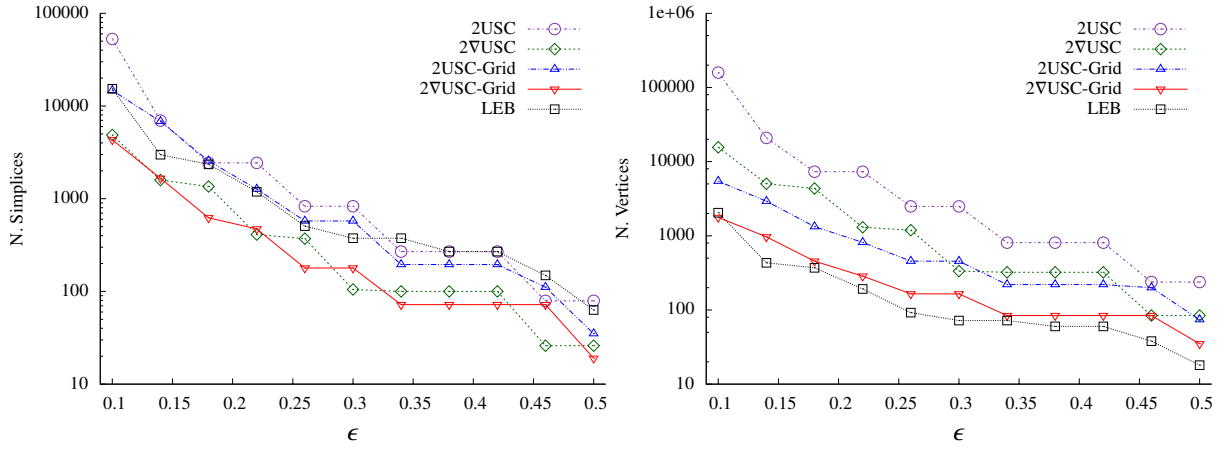


Figure 2.19: Number of evaluated simplices/vertices (left/right); $n = 4$ varying ε

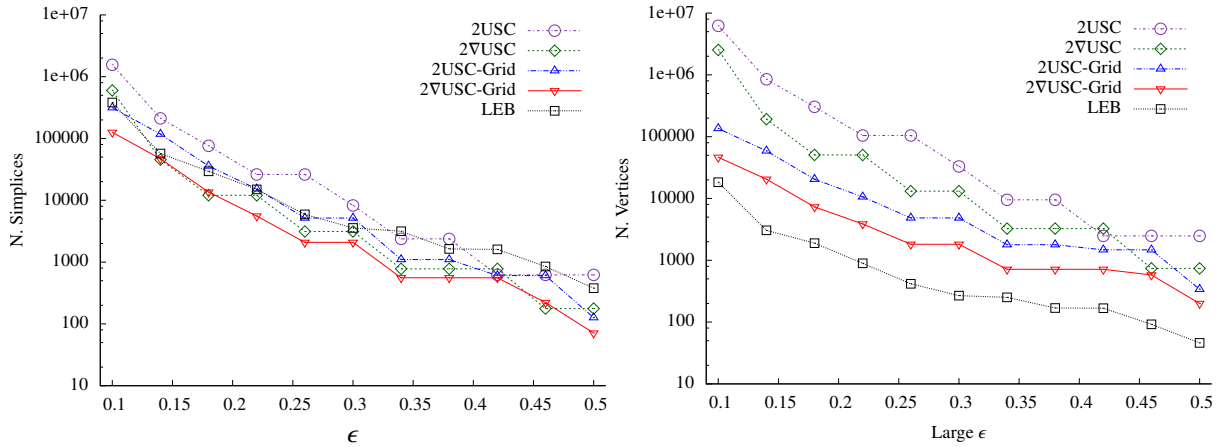


Figure 2.20: Number of evaluated simplices/vertices (left/right); $n = 5$ varying ε

2.2.4 Conclusions

Regularity has many advantages in practice. The present work investigated the research question whether simplex refinement can be done in an efficient way using regular sub-simplices. The results are counter-intuitive and surprising both theoretically and practically. Several overlapping refinement methods are described and implemented, among which 2∇USC-Grid seems most promising according to the experiments of generating a complete tree up to a user accuracy.

The most appropriate threshold value is still an open question. Moreover, it would be interesting to see how the refinement methods are going to perform on branch-and-bound instances for mixture design (blending) problems.

2.3 Gradient based simplicial bounds and monotonicity considerations

Recently, there has been a renewed interest in generating tight bounds over simplicial partition sets. Karhbet and Kearfott [116] discuss the idea of using range computation over simplices based on Interval Arithmetic. In [139], focus is on using second derivative enclosures for generating bounds. These works do not take monotonicity considerations over the simplex into account as discussed by [106]. Our research question is how information on the bounds of first derivatives can be used to derive tight bounds and to create new monotonicity tests in simplicial B&B. To investigate this question, we derive bounds based on derivative information and implement them in a B&B algorithm to compare the different techniques.

The rest of this section is organized as follows. Subsection 2.3.1 introduces the notation. Subsection 2.3.2 presents several approaches to obtain lower bounds of a function over a simplex. Subsection 2.3.3

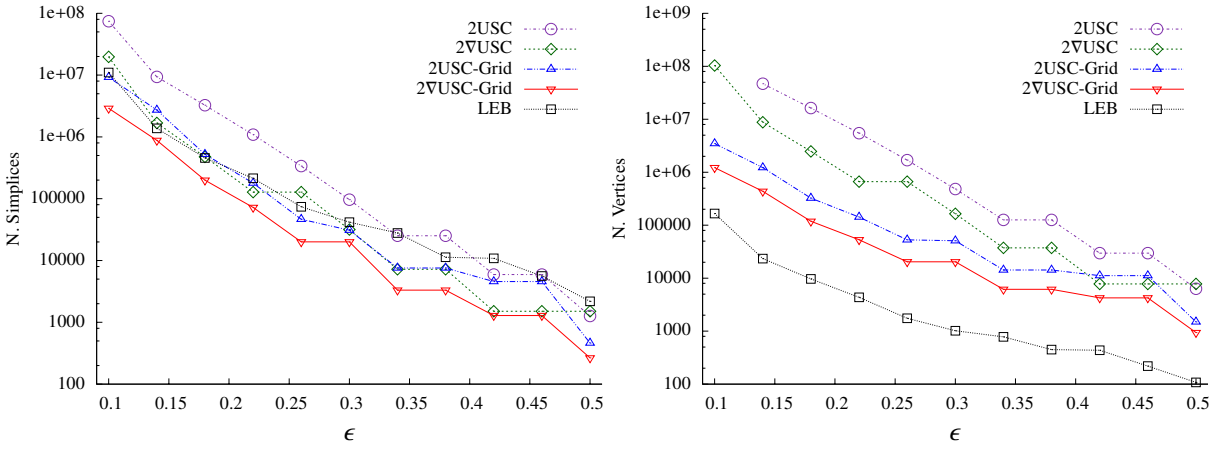


Figure 2.21: Number of evaluated simplices/vertices (left/right); $n = 6$ varying ε

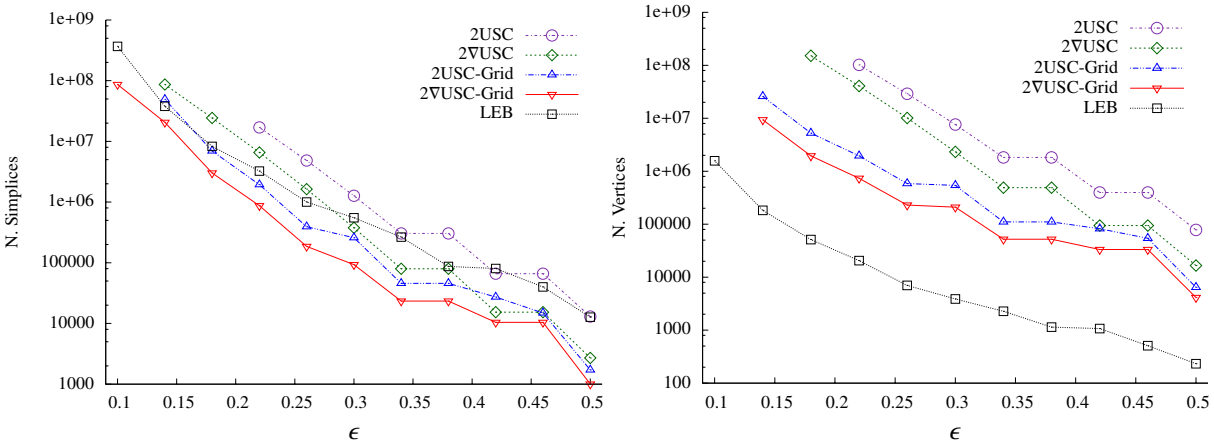


Figure 2.22: Number of evaluated simplices/vertices (left/right); $n = 7$ varying ε

deals with monotonicity over a simplex. Subsection 2.3.6 describes the Global Optimization B&B algorithm to compare lower bounding methods over a simplex. Subsection 2.3.7 compares the results of the bounding techniques numerically on a large number of instances. Finally, Subsection 2.3.8 presents our findings.

Lower bounds were the results of common work, but many of them are based on my work, for instance the theoretical result. Monotonicity results are similarly common, again, my contribution was crucial here as well. Numerical results are done by Leocadio G. Casado.

2.3.1 Preliminaries

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$ that has to be minimized over a feasible set $D \subset \mathbb{R}^n$, which is either a box or a simplex, on which f is differentiable:

$$\min_{x \in D} f(x). \tag{2.55}$$

Actually, the problem can be posed more generically, where the feasible set is not full dimensional compared to the objective function. The branch-and-bound algorithm works with a list Λ of partition sets, which as a whole include all global minimum points. Let us recall our simplicial partition sets.

Notation 2.3.1. Let $\mathcal{V} = \{v_1, \ldots, v_h\} \subset \mathbb{R}^n$ denote a set of h affinely independent vertices. For the component i of vertex j, we use the notation v_{ji}.

Notation 2.3.2. An $(h-1)$-simplex S is determined by the convex hull of $\mathcal{V}$, i.e. $S = \text{conv}(\mathcal{V})$ as formulated in (2.1).

The algorithm to be studied uses longest edge bisection, where the longest edge (v, w) of a partition set S is bisected using mid-point $x = \frac{v+w}{2}$ leading to two new simplices with vertex sets $\mathcal{V} \setminus \{v\} \cup \{x\}$ and

$\mathcal{V} \setminus \{w\} \cup \{x\}$. The algorithm also considers dimension reduction due to monotonicity considerations where the set $\mathcal{V}$ of vertices of $(h-1)$-simplex S is reduced to $\mathcal{V} \setminus \{v\}$ and S is replaced by one (or more) of its facets $F := \text{conv}(\mathcal{V} \setminus \{v\})$ for some $v \in \mathcal{V}$. Notice that F is an $(h-2)$-simplex. It may be clear that for $h = 1$, the 0-simplex $S = \text{conv}(\{v_1\})$ is an individual point and has no faces. Its dimension cannot be reduced.

The centroid of $(h-1)$-simplex $S = \text{conv}(\{v_1, \ldots, v_h\})$ is given by $c = \frac{1}{h}\sum_{j=1}^{h} v_j$ and the relative interior is defined by

$$\text{rint}(S) = \{x = \sum_j \lambda_j v_j, \lambda_j > 0, j = 1, \ldots, h, \sum_{j=1}^{h} \lambda_j = 1\}. \tag{2.56}$$

The relative boundary of a simplex S is defined by removing the relative interior from it, i.e. $\partial S = S \setminus \text{rint}(S)$. Considering a simplicial partition set S, we are interested in studying whether its (simplicial) facets F are border with respect to feasible set D.

Definition 2.3.3. *Given a feasible area D, and an $(h-1)$-simplex S with $h \leq n$ is called border with respect to D if there exists an $(h-1)$-dimensional face φ of D, such that $S \subseteq \varphi$.*

Remark 2.3.4. For a simplicial feasible area, the $(h-1)$-dimensional face φ is an $(h-1)$-simplex, while for a box, φ is an $(h-1)$-dimensional interval.

Notation 2.3.5. We recall here, that we denoted intervals by boldface letters and their lower and upper bound by 'underline' and 'overline', respectively. The radius of an interval $\boldsymbol{x} = [\underline{x}, \overline{x}]$ is denoted by $\text{rad}\,(\boldsymbol{x}) = \frac{\overline{x}-\underline{x}}{2}$ and its midpoint by $\text{mid}\,(\boldsymbol{x}) = \frac{\overline{x}+\underline{x}}{2}$. For an interval vector (also called a box) these are taken component-wise. The width of a box $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T$ is to be understood as $\text{wid}\,(\boldsymbol{x}) = 2 \max_{i=1,\ldots,n} \text{rad}\,(\boldsymbol{x}_i)$.

Notation 2.3.6. The interval hull of a simplex S is denoted by $\square S = \square\,\text{conv}(\mathcal{V})$, that is the smallest interval box enclosing the simplex S. Let $\boldsymbol{x} = \square S$, where

$$\boldsymbol{x}_i = [\underline{x}_i, \overline{x}_i] = [\min_{v_j \in \mathcal{V}} v_{ji}, \max_{v_j \in \mathcal{V}} v_{ji}] \qquad \forall i \in \{1, \ldots, n\}. \tag{2.57}$$

Remark 2.3.7. For cases where $\boldsymbol{x} = \square S \subseteq D$, f is differentiable over $\boldsymbol{x}$. Notice that if D is a box and $S \subset D$, automatically we have $\square S \subseteq D$. However, if D is a simplex, then f is not necessarily differentiable over $\boldsymbol{x} = \square S$.

2.3.2 Bounding techniques over a simplex

2.3.2.1 Extension of standard interval bounding techniques to simplices

Extensive investigation on Interval Arithmetic has lead to many ways to derive rigorous bounds, see for instance [101, 117, 140, 158].

Notation 2.3.8. Let $\boldsymbol{f}$ denote the natural interval extension [140] of an expression f with

$$\boldsymbol{f}(\boldsymbol{x}) = [\underline{\boldsymbol{f}}(\boldsymbol{x}), \overline{\boldsymbol{f}}(\boldsymbol{x})] \supseteq [\min_{x \in \boldsymbol{x}} f(x), \max_{x \in \boldsymbol{x}} f(x)],\ \forall \boldsymbol{x} \subseteq D.$$

Remark 2.3.9. $\forall x \in S \subset \boldsymbol{x} = \square S$, $f(x) \in \boldsymbol{f}(\boldsymbol{x})$.

Notation 2.3.10. Let $\boldsymbol{\nabla f}(\boldsymbol{x})$ denote an enclosure of the gradient and $\boldsymbol{\nabla f}_i(\boldsymbol{x}) = [\underline{\nabla \boldsymbol{f}_i}(\boldsymbol{x}), \overline{\nabla \boldsymbol{f}_i}(\boldsymbol{x})]$ the i-th component of the interval gradient. They can be computed using Interval Arithmetic and Automatic Differentiation [157].

Remark 2.3.11. $\forall x \in \boldsymbol{x}$, $\frac{\partial f}{\partial x_i}(x) \in \boldsymbol{\nabla f}_i(\boldsymbol{x})$. Then, $\forall x \in S \subset \boldsymbol{x} = \square S$, $\frac{\partial f}{\partial x_i}(x) \in \boldsymbol{\nabla f}_i(\boldsymbol{x})$.

Notation 2.3.12. A centered form on a box $\boldsymbol{x}$ with center c is denoted by $\boldsymbol{f}_c(\boldsymbol{x})$. It is in fact the interval extension of the first-order Taylor expansion using $\boldsymbol{\nabla f}(\boldsymbol{x})$:

$$\boldsymbol{f}_c(\boldsymbol{x}) = \boldsymbol{f}(c) + (\boldsymbol{x} - c)^T \boldsymbol{\nabla f}(\boldsymbol{x}), \text{with } c \in \boldsymbol{x}.$$

Usually c is the midpoint (or center) of box $\boldsymbol{x}$. In that case, we refer to $\boldsymbol{f}_{cb}(\boldsymbol{x})$ where $cb = \text{mid}\,(\boldsymbol{x})$. The lower bound $\underline{\boldsymbol{f}_c}(\boldsymbol{x})$ can also be written as $\underline{\boldsymbol{f}(c) + (\boldsymbol{x} - c)^T \boldsymbol{\nabla f}(\boldsymbol{x})} = \underline{\boldsymbol{f}}(c) + \underline{(\boldsymbol{x} - c)^T \boldsymbol{\nabla f}(\boldsymbol{x})}$, where underline takes the lower bound of the formula computed by IA.

Remark 2.3.13. $\forall x \in S \subset \boldsymbol{x} = \square S$, $f(x) \in \boldsymbol{f}_c(\boldsymbol{x})$. Thus, $\boldsymbol{f}_c(\boldsymbol{x})$ provides lower and upper bounds of f over S, even if $c \notin S$.

Baumann [9] proposed another base-point instead of the center cb to improve the lower and upper bounds of the centered form.

Notation 2.3.14. We denote the Baumann base-point for the optimal lower bound in the centered form on a box by bb^-. Component i is given by

$$bb_i^- = \begin{cases} \dfrac{\underline{x_i}\overline{\nabla \boldsymbol{f}_i}(\boldsymbol{x}) - \overline{x}_i \underline{\nabla \boldsymbol{f}_i}(\boldsymbol{x})}{\operatorname{wid}\left(\boldsymbol{\nabla f}_i(\boldsymbol{x})\right)} & \text{if } 0 \in \boldsymbol{\nabla f}_i(\boldsymbol{x}) \\ \underline{x}_i & \text{if } \nabla \boldsymbol{f}_i(\boldsymbol{x}) > 0 \\ \overline{x}_i & \text{if } \overline{\nabla \boldsymbol{f}}_i(\boldsymbol{x}) < 0. \end{cases}$$

Any centered form (with a base-point $y \in \boldsymbol{x}$) can be tightened based on the vertices of simplex S.

Proposition 2.3.15. *Let*

$$\underline{\boldsymbol{f}_y}(S) = \underline{\boldsymbol{f}}(y) + \min_{v \in \mathcal{V}} \{\underline{(v-y)^T \boldsymbol{\nabla f}(\boldsymbol{x})}\},\ y \in \boldsymbol{x}. \tag{2.58}$$

Then $\underline{\boldsymbol{f}_y}(S) \leq \min_{x \in S} f(x)$.

Proof. A first-order Taylor form provides a concave lower bounding function [9, 187]. A concave function takes its minimum over a convex set at its extreme points. Consequently, the lower bounding function over the simplex takes its minimum at a vertex of the simplex. Thus, instead of computing the interval enclosure over $\boldsymbol{x}$=$\square S$, taking the minimum over the simplex vertices provides a valid lower bound. □

Remark 2.3.16. We can use $y = cb$ or $y = bb^-$ in (2.58).

Now, it is interesting to see how the Baumann point bb^- can be generalized to a simplicial base-point. For bb^-, the aim is to select the best base-point for the Taylor form, such that the lower bound is as high as possible. For a simplex, instead of using the limits of enclosing box $\boldsymbol{x}$=$\square S$, we use the simplex vertices.

The highest lower bound in (2.58) over a simplex is taken at the base-point

$$\begin{aligned} &\operatorname{argmax}_{y \in \boldsymbol{x}} \min_{v \in \mathcal{V}} \left(\underline{\boldsymbol{f}}(y) + \underline{(v-y)^T \boldsymbol{\nabla f}(\boldsymbol{x})} \right) = \\ &\operatorname{argmax}_{y \in \boldsymbol{x}} \left(\underline{\boldsymbol{f}}(y) + \min_{v \in \mathcal{V}} \underline{(v-y)^T \boldsymbol{\nabla f}(\boldsymbol{x})} \right). \end{aligned} \tag{2.59}$$

Obviously, optimizing (2.59) is a nonlinear problem as it includes the optimization of $f(y)$ varying y. Therefore, it is advisable to optimize only the second part.

Definition 2.3.17. *Let us define* $bs^- = \operatorname{argmax}_{y \in \boldsymbol{x}} \min_{v \in \mathcal{V}} \underline{(v-y)^T \boldsymbol{\nabla f}(\boldsymbol{x})}$ *as the Baumann point over the simplex. This point can be found by an interval linear program:*

$$\begin{aligned} \max_{y \in \boldsymbol{x}, z \in \mathbb{R}} \quad & z \\ \text{s.t.} \quad & z \leq \underline{(v-y)^T \boldsymbol{\nabla f}(\boldsymbol{x})},\ \forall v \in \mathcal{V}. \end{aligned} \tag{2.60}$$

Let (z^*, y^*) be the optimum of (2.60). Then we take base point $bs^- = y^*$ with the corresponding lower bound $\underline{\boldsymbol{f}_{bs^-}}(\boldsymbol{x}) = \boldsymbol{f}(bs^-) + z^*$.

Notation 2.3.18. $\nabla^w \boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}^n$ denotes gradient bounds with components $\nabla^w \boldsymbol{f}_i(\boldsymbol{x}) = \underline{\nabla \boldsymbol{f}_i}(\boldsymbol{x})$ if $w_i = \underline{x}_i$ and $\nabla^w \boldsymbol{f}_i(\boldsymbol{x}) = \overline{\nabla \boldsymbol{f}}_i(\boldsymbol{x})$ if $w_i = \overline{x}_i$.

Remark 2.3.19. In Notation 2.3.18, all possible variations of lower and upper bounds of the gradients are taken into account when considering all vertices w of $\boldsymbol{x}$.

Writing (2.60) as a linear program requires 2^n constraints for each vertex $v \in \mathcal{V}$:

$$\begin{aligned} \max_{y \in \boldsymbol{x}, z \in \mathbb{R}} \quad & z \\ \text{s.t.} \quad & z \leq (v-y)^T \nabla^w \boldsymbol{f}(\boldsymbol{x}), \quad \forall v \in \mathcal{V},\ \forall w \text{ vertex of } \boldsymbol{x}. \end{aligned} \tag{2.61}$$

The constraints in (2.61) can be written as 2^n linear inequalities

$$\begin{aligned} \max_{y\in\boldsymbol{x},z\in\mathbb{R}} \quad & z \\ \text{s.t.} \quad & z + y^T\nabla^w \boldsymbol{f}(\boldsymbol{x}) \leq \min_{v\in\mathcal{V}} v^T\nabla^w \boldsymbol{f}(\boldsymbol{x}),\ \forall w \text{ vertex of } \boldsymbol{x}. \end{aligned} \tag{2.62}$$

Note that we do not force bs^- to be in simplex S, because it may happen that a point outside S would give the best lower bound. In case we want to use $\overline{f}(bs^-)$ to update the upper bound of the global minimum in a B&B algorithm, bs^- has to be in the initial search region. In our experiments we force bs^- to be in S by adding $y \in S$ using simplex inclusion constraints (2.1) to (2.62) in a similar way as it is done in (2.65).

Notice that (2.60), (2.61), and (2.62) are equivalent descriptions of the same problem, thus providing the same optimum corresponding to the same bound.

2.3.2.2 Linear relaxation based lower bounds

Following earlier results in interval based B&B [144, 178, 179], we can now define other lower bounds for simplicial subsets.

Standard linear relaxation of f over a box

Let w be a vertex of $\boldsymbol{x}$=$\square S$ and consider a first order Taylor expansion

$$\underline{\boldsymbol{f}_w}(x) = \underline{\boldsymbol{f}}(w) + (x-w)^T\nabla^w \boldsymbol{f}(\boldsymbol{x}) \leq f(x) \quad \forall x \in \boldsymbol{x}. \tag{2.63}$$

Since we have 2^n vertices of $\boldsymbol{x}$, we obtain 2^n inequalities from equation (2.63), see [135] for more details. Consider the linear program

$$\begin{aligned} \min_{x\in\boldsymbol{x},z\in\mathbb{R}} \quad & z \\ \text{s.t.} \quad & z \geq \underline{\boldsymbol{f}}(w) + (x-w)^T\nabla^w \boldsymbol{f}(\boldsymbol{x}),\ \forall w \text{ vertex of } \boldsymbol{x}. \end{aligned} \tag{2.64}$$

Let (x^*, z^*) be the optimal solution of (2.64), then

$$f(x) \geq z^*, \forall x \in \boldsymbol{x},$$

such that z^* is a lower bound of f over $\boldsymbol{x}$. z^* is also a lower bound of f over $S \subset \boldsymbol{x} = \square S$.

Linear relaxation of f over a simplex

We now focus on the bounds of f over simplex $S = \text{conv}(\mathcal{V})$. The earlier bound in (2.64) is valid for f over $\boldsymbol{x} = \square S$, such that it is also a bound over the simplex S. However, it is interesting to force $x \in \boldsymbol{x}$ to be inside S, like in (2.1). Introducing the corresponding linear equations into problem (2.64) provides linear program

$$\begin{aligned} \min_{\substack{x\in\boldsymbol{x},z\in\mathbb{R} \\ \lambda\in[0,1]^{n+1}}} \quad & z \\ \text{s.t.} \quad & z \geq \underline{\boldsymbol{f}}(w) + (x-w)^T\nabla^w \boldsymbol{f}(\boldsymbol{x}),\ \ \forall w \text{ vertex of } \boldsymbol{x} \\ & x = \sum_{j=1}^{h} \lambda_j v_j \\ & \sum_{j=1}^{h} \lambda_j = 1. \end{aligned} \tag{2.65}$$

Let (x^*, z^*, λ^*) be the solution of (2.65). Then we have that

$$f(x) \geq z^*, \forall x \in S$$

and therefore, z^* is a lower bound of f over S.

A straightforward idea is to consider the vertices of the simplex instead of the vertices of the enclosing box. Unfortunately, such a formulation leads to a Mixed Integer Programming problem, as the piece-wise linear lower bounding function is neither convex nor concave anymore.

2.3.2.3 Bounding technique using Affine Arithmetic

This section describes the use of Affine Arithmetic (see [4, 53, 134, 136, 169, 180]) to generate a linear underestimation of function f over $\boldsymbol{x}$=$\square S$. We add the constraint that the solution has to be inside the simplex $S = \text{conv}(\mathcal{V})$, see (2.1). This provides a linear program.

First, we focus on the transformation of an interval vector into a vector of affine forms. Second, we describe how the computations are made using Affine Arithmetic to provide linear equations. Third, we sketch how the so-obtained linear equations are used to provide linear underestimations of f over $\boldsymbol{x}$ and then we provide the linear program to find a lower bound of f over the simplex S. Fourth, we show a simple way to solve the linear program.

Conversion into affine forms

The interval vector $\boldsymbol{x}$=$\square S$ can be converted to an affine form vector, denoted by $\hat{x}$, as follows

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_i \\ \vdots \\ \boldsymbol{x}_n \end{pmatrix} = \begin{pmatrix} [\underline{x}_1, \overline{x}_1] \\ \vdots \\ [\underline{x}_i, \overline{x}_i] \\ \vdots \\ [\underline{x}_n, \overline{x}_n] \end{pmatrix} \rightarrow \hat{x} = \begin{pmatrix} \hat{x}_1 \\ \vdots \\ \hat{x}_i \\ \vdots \\ \hat{x}_n \end{pmatrix} = \begin{pmatrix} \text{mid}\,(\boldsymbol{x}_1) + \text{rad}\,(\boldsymbol{x}_1)\varepsilon_1 \\ \vdots \\ \text{mid}\,(\boldsymbol{x}_i) + \text{rad}\,(\boldsymbol{x}_i)\varepsilon_i \\ \vdots \\ \text{mid}\,(\boldsymbol{x}_n) + \text{rad}\,(\boldsymbol{x}_n)\varepsilon_n \end{pmatrix}, \tag{2.66}$$

where $\varepsilon_i \in [-1, 1]$ for all $i \in \{1, \ldots, n\}$. The affine form $\hat{x}$ can be transformed back into an interval by changing ε_i to $[-1, 1]$. Moreover, for all $x \in \boldsymbol{x}$, there is exactly one corresponding value for ε in the affine description,

$$x = T(\boldsymbol{x}, \varepsilon) = \text{mid}\,(\boldsymbol{x}) + \text{rad}\,(\boldsymbol{x})\varepsilon,$$

where $\varepsilon_i = \frac{x_i - \text{mid}\,(\boldsymbol{x}_i)}{\text{rad}\,(\boldsymbol{x}_i)}, i = 1, \ldots, n$.

Affine Arithmetic

By replacing all the occurrences of the variable x_i by the corresponding affine form $\hat{x}_i$ in an expression of f, and by performing the computations using Affine Arithmetic, we obtain a resulting affine form, denoted by

$$\hat{f}(T(\boldsymbol{x}, \varepsilon)) = r_0 + \sum_{i=1}^{n} r_i \varepsilon_i + \sum_{k=n+1}^{N} r_k \varepsilon_k, \tag{2.67}$$

where ε_j is in $[-1, 1]$ for all $j \in \{1, \ldots, N\}$. Note that some error terms $r_k \varepsilon_k$ are added for all $k \in \{n+1, \ldots, N\}$, which come from non affine operations in f.

Linear underestimation of f over $\boldsymbol{x}$

Using Affine Arithmetic, (2.67) underestimates f over $\boldsymbol{x}$

$$f(x) = f(T(\boldsymbol{x}, \varepsilon)) \geq \underline{\hat{f}}(T(\boldsymbol{x}, \varepsilon)) = r_0 + \sum_{i=1}^{n} r_i \varepsilon_i - \sum_{k=n+1}^{N} |r_k|, \tag{2.68}$$

because all error terms are taken into account using their worst value.

Remark 2.3.20. Equation (2.68) is a linear underestimation of f over $\boldsymbol{x}$ using the new variables ε_i.

Linear program to provide lower bounds

In order to compute a lower bound of f over the simplex S (and not only on the $\boldsymbol{x}$=$\square S$), we constrain the point x to be inside S by adding (2.1). In this case, we describe x by its affine form $T(\boldsymbol{x}, \varepsilon)$ and thus, we

obtain the linear program

$$\begin{aligned} \min_{\substack{\varepsilon\in[-1,1]^n \\ \lambda\in[0,1]^{n+1}}} \quad & \sum_{i=1}^{n} r_i\varepsilon_i \\ \text{s.t.} \quad & T(\boldsymbol{x},\varepsilon) = \sum_{j=1}^{h} \lambda_j v_j \\ & \sum_{j=1}^{h} \lambda_j = 1. \end{aligned} \tag{2.69}$$

Denoting the exact solution of (2.69) by $(\varepsilon^*, \lambda^*)$, we have that

$$f(x) = f(T(\boldsymbol{x},\varepsilon)) \geq \sum_{i=1}^{n} r_i\varepsilon_i^* + r_0 - \sum_{k=n+1}^{N} |r_k|,\ \forall x \in S \tag{2.70}$$

and therefore, this is a lower bound of f over S.

Note that to solve the above linear program, we just need to evaluate f at each vertex of S and then take the minimum value of the linear underestimation (2.68). If $\text{rad}\,(\boldsymbol{x}_i) \neq 0$ (else $r_i = 0$), $\varepsilon_i(v_j)$ is a transformation of component i of vertex v_j into variable ε

$$\varepsilon_i(v_j) = \begin{cases} \dfrac{v_{ji} - \text{mid}\,(\boldsymbol{x}_i)}{\text{rad}\,(\boldsymbol{x}_i)}, & \text{if rad}\,(\boldsymbol{x}_i) > 0 \\ 0, & \text{if rad}\,(\boldsymbol{x}_i) = 0 \end{cases} \quad \forall i \in \{1,\ldots,n\}, \text{ and } \forall v_j \in \mathcal{V}.$$

Then, the lower bound of (2.70) becomes

$$\min_{j\in\{1,\ldots,h\}} \sum_{i=1}^{n} r_i\varepsilon_i(v_j) + r_0 - \sum_{k=n+1}^{N} |r_k|. \tag{2.71}$$

Therefore, instead of solving linear program (2.69), we can determine (2.71) and this yields directly the lower bound of f over S. The solution corresponds to the solution of linear program (2.69).

2.3.3 Monotonicity test

Now, we describe a concise monotonicity test which excludes an interior partition set S if it does not contain a stationary point. To be more precise:

Proposition 2.3.21. *Let $S \subset int(D)$ be a simplex in the interior of the search area D. If $\exists i \in \{1,\ldots,n\}$ with $0 \notin \boldsymbol{\nabla} f_i(\square S)$ then S does not contain a global minimum point of* (2.55).

Proof. The condition implies that

$$\forall x \in S, \frac{\partial f}{\partial x_i}(x) \neq 0$$

such that S cannot contain an interior minimum of D. Moreover, S does not touch the boundary, i.e. $S \cap \partial D = \emptyset$, such that neither it can contain a boundary optimum point. □

The test is not strong as initial partition sets typically touch the boundary. Stated it differently, for a simplex $S \subseteq D$, if $\exists i \in \{1,\ldots,n\}$ with $0 \notin \boldsymbol{\nabla} f_i(\square S)$ then $\text{rint}(S)$ does not contain a global minimum point of (2.55). The practical consequence is that the interior of S cannot contain the optimum, so S can be replaced by its border facets.

A slight relaxation is the following corollary, where a simplicial partition set has no facets on the boundary.

Corollary 2.3.22. *Let $S \subseteq D$ be a simplex as partition set in a branch-and bound-algorithm with corresponding gradient enclosure $\boldsymbol{\nabla} f(\square S)$. If $\exists i \in \{1,\ldots,n\}$ with $0 \notin \boldsymbol{\nabla} f_i(\square S)$ and S has no border facets, then S can be rejected.*

Proof. The same reasoning as in the proof of Proposition 2.3.21 applies with respect to the interior of S. The relative boundary of S may contain a global minimum point, but the same point is enclosed in the relative border of another partition set, such that we do not have to store S anymore. □

This corollary is not strong, but it is relatively easy to check. For practical tests, Proposition 2.3.21 offers the conditions for removing an interior simplicial partition set S. We can also remove it, if just several vertices of S touch the boundary of D according to Corollary 2.3.22. Otherwise, we should store the facets of S which are completely in a face of the search region as new simplicial partition sets with less than h vertices.

Even not all border facets are interesting partition sets. It might be shown that the relative interior of a facet F cannot contain a minimum point.

For this property, we can focus on bounds of the directional derivative in a direction d, denoted by $\underline{d^T \nabla \boldsymbol{f}(\square S)}$ and $\overline{d^T \nabla \boldsymbol{f}(\square S)}$. For instance, an upper bound of the directional derivative

$$\overline{d^T \nabla \boldsymbol{f}(\square S)} = \sum_{i=1}^{n} \max\{d_i \underline{\nabla \boldsymbol{f}_i}(\square S), d_i \overline{\nabla \boldsymbol{f}_i}(\square S)\}. \tag{2.72}$$

Notice that a necessary condition for $\overline{d^T \nabla \boldsymbol{f}(\square S)} \leq 0$ according to (2.72) is that f is monotone on $\square S$, i.e. $0 \notin \nabla \boldsymbol{f}(\square S)$.

Moreover, let us introduce the so-called normal vector p with respect to facet F. Let $S = \text{conv}(\mathcal{V})$, $v \in \mathcal{V}$, $\mathcal{U} = \mathcal{V} \setminus \{v\}$, facet $F = \text{conv}(\mathcal{U})$. Consider $u \in \mathcal{U}$ and construct a set of vectors $\mathcal{Y} := \{w - u, w \in \mathcal{U} \setminus \{u\}\}$ with corresponding matrix representation Y. The normal vector of F in the direction of v can be expressed by defining (see e.g. [5])

$$p = \left[I - Y(Y^T Y)^{-1} Y^T\right](v - u). \tag{2.73}$$

The projection procedure (2.73) is sketched in Figure 2.23. Vector p points into the same half-space where

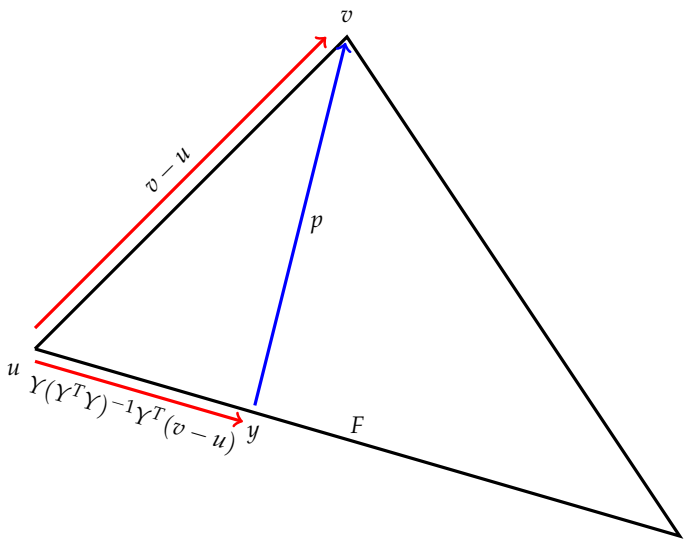


Figure 2.23: Normal vector calculation via projection of an edge $v - u$

S is located. From each point in $\text{rint}(F)$ a direction d with $p^T d > 0$ points into $\text{rint}(S)$. Considering the gradient enclosure $\nabla \boldsymbol{f}(\square S)$ component wise, we can come to several conclusions.

Proposition 2.3.23. *Given simplex $S = \text{conv}(\mathcal{V})$ with a facet F generated by removing vertex v from $\mathcal{V}$ with normal vector p as in* (2.73)*. If for one component i we have $\nabla \boldsymbol{f}_i(\square S) p_i < 0$ (either $\overline{\nabla \boldsymbol{f}}_i(\square S) < 0$ and $p_i > 0$, or $\underline{\nabla \boldsymbol{f}_i}(\square S) > 0$ and $p_i < 0$), then $\text{rint}(F)$ cannot contain a minimum point of* (2.55)*.*

Proof. Suppose that the assumptions of Proposition 2.3.23 hold. We know that for any $x \in \text{rint}(F)$ exists an $\varepsilon > 0$ such that $y = x + \varepsilon p_i e_i \in S$. No point $x \in \text{rint}(F)$ can be a global optimizer, as

$$f(y) = f(x + \varepsilon p_i e_i) \leq f(x) + \nabla \boldsymbol{f}(\square S) p_i e_i = f(x) + \nabla \boldsymbol{f}(\square S) p_i < f(x).$$

□

In the branch-and-bound procedure, if the search space is a full-dimensional simplex, the partition sets are full-dimensional n-simplices, up to reduction to its facets. Using the property of $0 \notin \nabla \boldsymbol{f}(\square S)$, if $\text{rint}(S)$

can be removed, we can reduce it to its border facets. However, for lower dimension $(h-1)$-simplices, $h \leq n$, the condition $0 \notin \boldsymbol{\nabla} \boldsymbol{f}(\square S)$ is a necessary condition to have monotonicity, but not sufficient. Therefore, we have to focus on directional derivative bounds. For the ease of notations, from now on, we will denote the inclusion of the gradient for the interval hull of simplex S by $\boldsymbol{G} = \boldsymbol{\nabla} \boldsymbol{f}(\square S)$.

Now consider a directional vector d as the difference of two points in S, then the corresponding directional derivative $d^T \nabla f(x)$ is also included in the inner product

$$\langle d, \boldsymbol{G} \rangle = \left[\underline{d^T \boldsymbol{G}}, \overline{d^T \boldsymbol{G}} \right] = \left[\sum_{i=1}^{n} \min\{d_i \underline{G}_i, d_i \overline{G}_i\}, \sum_{i=1}^{n} \max\{d_i \underline{G}_i, d_i \overline{G}_i\} \right]. \tag{2.74}$$

The most general result for an $(h-1)$-simplex is the following.

Proposition 2.3.24. *Let $S \subseteq D$ be an $(h-1)$-simplex with $h \leq n+1$ and gradient enclosure $\boldsymbol{G}$. If $\exists\, x, y \in S$, such that direction $d = x - y$ has corresponding directional derivative bounds* (2.74) *with $0 \notin [\underline{d^T \boldsymbol{G}}, \overline{d^T \boldsymbol{G}}]$ then* $\operatorname{rint}(S)$ *does not contain a global minimum point of* (2.55).

Proof. Consider $z \in \operatorname{rint}(S)$. As z is in the relative interior, there exists a feasible direction d in which lower function values can be found, i.e. $\exists \varepsilon \in \mathbb{R}$ small enough, such that $z + \varepsilon d \in S$ and $f(z + \varepsilon d) < f(z)$. So z cannot be a minimum point of f. □

The elaboration for the algorithm depends on the choice of the direction d and the way to compute it. Again, in a branch and bound context, Proposition 2.3.24 has some practical impact.

Corollary 2.3.25. *Let $S \subseteq D$ be an $(h-1)$-simplex as partition set in a branch-and-bound algorithm with corresponding gradient enclosure $\boldsymbol{G}$. If the conditions of Proposition 2.3.24 apply and S has no border facets, then S can be rejected.*

The argument is that the relative border of S may contain a global minimum point, but the same point is enclosed in the relative border of another partition set.

Given that the minimum is not in $\operatorname{rint}(S)$, we have to decide which of the facets to focus on.

Proposition 2.3.26. *Given $(h-1)$-simplex $S = \operatorname{conv}(\mathcal{V})$, $1 \leq h \leq n+1$ and a facet F generated by removing vertex v from $\mathcal{V}$, consider direction $d = v - y$, where $y \in S$. If $\underline{d^T \boldsymbol{G}} > 0$, i.e. the directional derivative in the direction $d = v - y$ is positive, then the facet F contains all minimum points in S, i.e.* $\operatorname{argmin}_{x \in S} f(x) \subseteq F$.

Proof. Without loss of generality, consider the vertex set $\mathcal{V} = \{v_1, \ldots, v_h\}$ and for $v = v_1$ the facet $F = \operatorname{conv}(\{v_2, \ldots, v_h\})$. First we show that $\forall x \in S \setminus F$, $\exists \rho > 0$ such that $z = x - \rho d \in F$. We have $x, y \in S$, such that there exist vectors λ, γ with $x = \sum_{j=1}^{h} \lambda_j v_j$ with $\lambda_1 > 0$ and $y = \sum_{j=1}^{h} \gamma_j v_j$. Looking for the value ρ, such that $z = x - \rho(v_1 - y) \in F$, we have that $\lambda_1 - \rho(1 - \gamma_1) = 0$. Hence, taking $\rho = \frac{\lambda_1}{1-\gamma_1}$, we obtain $z \in F$. According to the mean value theorem, there exists a point ξ in between x and z such that

$$f(x) = f(z + \rho d) = f(z) + \langle \rho d, \nabla f(\xi) \rangle \geq f(z) + \rho \underline{d^T \boldsymbol{G}} > f(z).$$

This means that each point $x \in S \setminus F$ corresponds to a point $z \in F$ with a lower function value. □

Remark 2.3.27. Notice that if there are several facets fulfilling the condition of Proposition 2.3.26, then the consequence is also valid for the intersection of the facets, which is a face of S. So if Proposition 2.3.26 applies to facets F and H, then it also applies to $F \cap H$.

Proposition 2.3.28. *Given $(h-1)$-simplex S fulfilling the conditions in Proposition 2.3.26. If F is a non-border facet, then global minimum points of* (2.55) *cannot be in* $\operatorname{rint}(F)$, *i.e. global minimum points can only be found at the relative boundary of F.*

Proof. According to Proposition 2.3.26, the minimum points of f over S can be found on facet F. We first show that $x \in \operatorname{rint}(F)$ does not contain global minimum points. Consider a point $x \in \operatorname{rint}(F)$. Because F is non-border, there exists an $\varepsilon > 0$, such that $z(\varepsilon) := x - \varepsilon(v - y) \in D$ and $z(\varepsilon) \notin S$. As $\underline{d^T \boldsymbol{G}} > 0$ and f is continuously differentiable, there exists an $\varepsilon > 0$ (sufficiently small) such that for all $\delta \in [0, \varepsilon]$, we have $d^T \nabla f(z(\delta)) > 0$. Therefore, following the mean value theorem, there exists a point $\delta \in [0, \varepsilon]$ such that $f(z(\varepsilon)) = f(x) - \varepsilon(v - y)^T \nabla f(z(\delta)) < f(x)$. This means that in an environment of x outside S we can find a feasible point with a lower function value than x. The relative boundary of F could contain global minimum points of (2.55). □

Remark 2.3.29. First, let us note that the boundary of F is shared by other partition sets, thus we can disregard simplex S in a branch-and-bound process.

Second, notice that there can be several facets fulfilling the condition of Proposition 2.3.26, however, we need only one non-border facet fulfilling the condition to remove simplex S from consideration.

Corollary 2.3.30. *Given $(h-1)$-simplex S fulfilling the conditions of Proposition 2.3.26 with centroid c, considering direction $d = v - c$. If $\underline{d^T \boldsymbol{G}} > 0$, then the facet F contains all minimum points in S, i.e.* $\operatorname{argmin}_{x \in S} f(x) \subseteq F$. *Moreover, if F is a non-border facet, then simplex S cannot contain the global minimum of* (2.55).

This follows from Proposition 2.3.26 and 2.3.28 just by using the centroid $y = c$.

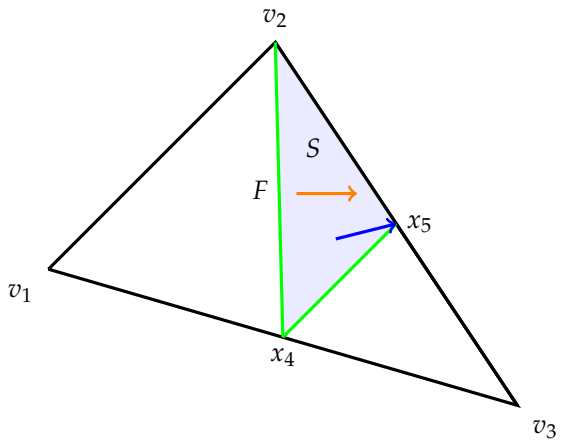


Figure 2.24: Three partition sets generated by bisection, using bisection points x_4 and x_5. We focus on monotonic directions in S.

Example 2.3.31. For the illustration of the concept, consider the simplices in Figure 2.24. It shows three partition sets generated by bisection, using bisection points x_4 and x_5. Consider the light blue simplex $S = \operatorname{conv}(\mathcal{V})$ with $\mathcal{V} = \{v_2, x_4, x_5\}$, where we assume that the orange direction provides a monotonously increasing direction. According to Proposition 2.3.24, the interior of S does not contain a global minimum point. Blue direction provides an increasing monotonicity for facet $F = \operatorname{conv}(\mathcal{V} \setminus \{v\})$ with $v = x_5$. According to Proposition 2.3.26, facet $F = \operatorname{conv}(\{v_2, x_4\})$ contains all minimum points on S. Specifically, they are at lower dimensional faces, as $\operatorname{rint}(F)$ cannot contain minima either. Now, Proposition 2.3.28 states, that we can even remove S, as there is another partition set at its left, that encloses all minimum points.

An alternative is to use the projection point $y = (v - p) \in S$ of the normal vector p of (2.73) as illustrated in Figure 2.23 for the choice of y. However, this is only correct if the simplex has angles of less than 90 degrees. This is typically the case when we follow the longest edge bisection process.

Corollary 2.3.32. *Given a simplex partition set $S = \operatorname{conv}(\mathcal{V})$ in the bisection and dimension reduction process and a facet F generated by removing vertex v from $\mathcal{V}$ with normal vector p according to* (2.73). *If $\underline{p^T \boldsymbol{G}} > 0$, then F contains all minimum points over S. Moreover, if F is a non-border facet, then simplex S cannot contain the global minimum of* (2.55).

This follows from Proposition 2.3.26 and 2.3.28 by using the normal vector p in place of $d = v - y$.

Proposition 2.3.33. *Given simplex $S = \operatorname{conv}(\mathcal{V})$, $1 \leq h \leq n+1$, consider a facet F generated by removing vertex v from $\mathcal{V}$. If for direction $d = v - y$ with $y \in S$ we have $\overline{d^T \boldsymbol{G}} < 0$, then $\operatorname{rint}(F) \subset S$ does not contain a minimum point of* (2.55).

Proof. Following the proof of Proposition 2.3.26 by taking $\nabla f(\xi) \leq \overline{d^T \boldsymbol{G}}$, we have

$$f(x) = f(z + \rho d) = f(z) + \langle \rho d, \nabla f(\xi) \rangle \leq f(z) + \rho \overline{d^T \boldsymbol{G}} < f(z)$$

and the result follows. □

This leads to a nice test in case there is a monotone direction in the simplex, but we cannot show that any of the facets contain all the minima using Proposition 2.3.26. In such case we have to replace the simplex by all its border facets, unless Proposition 2.3.33 holds. Thus, we need to consider only those border facets where we cannot find a point y for which Proposition 2.3.33 holds.

Corollary 2.3.34. *Given simplex $S = \text{conv}(\mathcal{V})$, $1 \leq h \leq n+1$, fulfilling the conditions of Proposition 2.3.33 with centroid c, considering direction $d = v - c$. If $\underline{d^T \boldsymbol{G}} < 0$, then facet F can be disregarded.*

2.3.3.1 Checking on monotonicity

To prove that there exists a monotone direction in a simplex, at least we should have $0 \notin \boldsymbol{G}$. This is a necessary, but not sufficient condition for an $(h-1)$-simplex, $h \leq n$. To prove that such a direction exists, according to Proposition 2.3.24, we need to find a direction $d = x - y$, with $x, y \in S$ corresponding to a positive lower bound of the directional derivative

$$0 < \underline{d^T \boldsymbol{G}} = \sum_{i=1}^{n} \min\{d_i \underline{G}_i, d_i \overline{G}_i\}. \tag{2.75}$$

Finding such a direction can be done by searching for the steepest monotone direction $\max_d \underline{d^T \boldsymbol{G}}$. Consider the terms $z_i = \min\{d_i \underline{G}_i, d_i \overline{G}_i\}$. This means we can write $\underline{d^T \boldsymbol{G}} = \sum_{i=1}^{n} z_i$. If we fix one of the points $x \in S$ in $d = x - y$, then the term $z_i(y) = \min\{(x_i - y_i)\underline{G}_i, (x_i - y_i)\overline{G}_i\}$ is a concave function in y, as it is the minimum of two affine functions. Therefore, the lower bound on the directional derivative $\underline{g}(y) = \underline{(x-y)^T \boldsymbol{G}}$ is a concave function, being the sum of concave terms. Similarly, it can be shown that the upper bound $\overline{g}$ on the directional derivative is a convex function. We will illustrate this with an example and then show how an LP problem can be formulated to find a maximum of $\underline{g}(y)$.

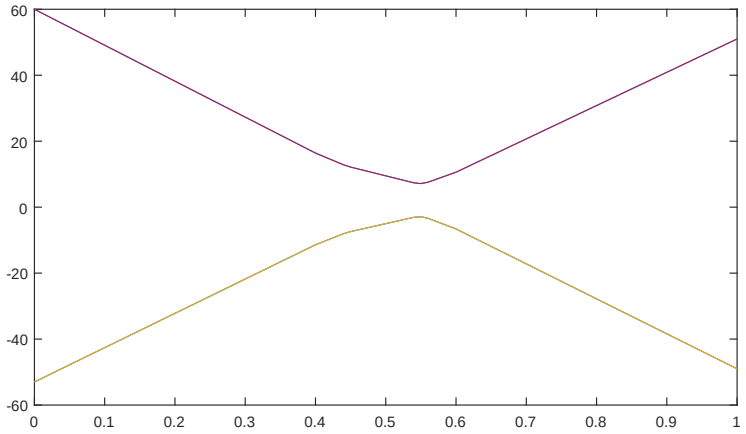


Figure 2.25: Shape of concave directional derivative lower bounding function $\underline{g}(y) = \underline{g}(\lambda v_2 + (1-\lambda)v_3)$ as function of λ in orange and the corresponding convex upper bounding $\overline{g}$ in purple.

Example 2.3.35. Consider a simplex $S = \text{conv}(\mathcal{V})$ with $\mathcal{V} = \{v_1, v_2, v_3\}$ in $\mathbb{R}^6$ with $v_1 = 0, v_2 = (1, -2, 3, -4, 5, 6)^T$ and $v_3 = (0, 3, -2, 5, -4, -5)^T$. In $d = x - y$, we take as fixed point $x = v_1$ and vary y over the edge between v_2 and v_3 as suggested in Proposition 2.3.26, so $y = \lambda v_2 + (1-\lambda)v_3, 0 \leq \lambda \leq 1$. Figure 2.25 sketches the concave piece-wise linear shape of $\underline{g}$ as function of λ and the convex shape of $\overline{g}$.

Looking for the existence of a positive value of $\underline{d^T \boldsymbol{G}}$ for some direction d, we can fix x to the centroid in the directional vector, i.e. $d = c - y$. The maximization of concave piece-wise linear function $\underline{g}(y)$ over

$y \in S$ can be formulated as the following LP problem.

$$\begin{aligned} & \max \sum_{i=1}^{n} z_i \\ \text{s.t.} \quad & z_i \leq (c_i - y_i)\underline{G}_i,\ i = 1, \ldots, n \\ & z_i \leq (c_i - y_i)\overline{G}_i,\ i = 1, \ldots, n \\ & \sum_{j=1}^{h} \lambda_j = 1 \\ & y = \sum_{j=1}^{h} \lambda_j v_j \\ & \lambda_j \geq 0, j = 1, \ldots, h. \end{aligned} \tag{2.76}$$

If there is no monotone direction, then $y = c$ and $\sum_{i=1}^{m} z_i = 0$.

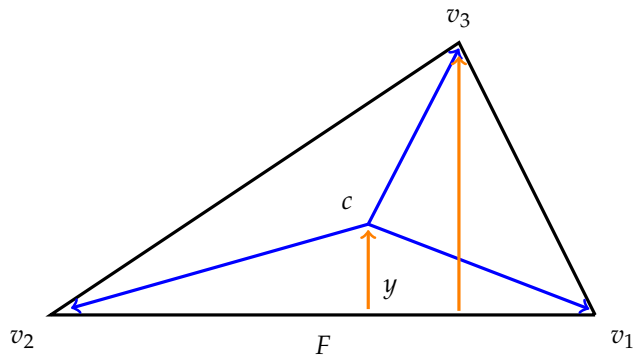


Figure 2.26: 2D image of the 2-simplex S. The point y provides a positive lower bound on the directional derivative in direction $d = c - y$.

Example 2.3.36. For the illustration of the concept, consider the 2-simplex S defined by three vertices $\mathcal{V} = \{(4,0,1)^T, (0,0,0)^T, (3,2,1)^T\}$ in 3-dimensional space, projected in 2D for Figure 2.26. Its centroid is given by $\frac{1}{3}(7,2,2)^T$. Now let the bounds of the gradient be given by $\underline{G} = (-3,1,0)^T$ and $\overline{G} = (1,2,1)^T$. For none of the directions $d = v_j - c$, we have that lower bound $\underline{d^T \boldsymbol{G}}$ is positive. Running the LP (2.76) provides us with a positive directional derivative bound of $\frac{2}{3}$ for the point $y = (\frac{7}{3}, 0, \frac{7}{12})^T$. The monotone direction $c - y$ is drawn by an orange arrow in Figure 2.26. This means that f is monotone on S. This illustrates that checking a finite number of directions over the simplex is not necessarily sufficient to prove that f is monotone. The LP (2.76) provides a numerical proof of f being monotone over S or not. Notice again that $y = c$ is a feasible solution of the LP yielding an objective function value of zero as soon as monotonicity cannot be proven.

Following the line of reasoning of the example, according to Proposition 2.3.24, we can conclude that simplex S can be replaced by its facets. Now at least one of the facets is of interest, although that cannot be concluded by only considering the directions $d = v - c$ for this specific example. The question is now whether there is a monotone direction from a vertex v to a point y on the facet $F = \text{conv}(\mathcal{V} \setminus \{v\})$ for which $d = v - y$ has a positive lower bound on the directional derivative $\underline{d^T \boldsymbol{G}} > 0$. If such direction can be found, according to a similar reasoning as in Proposition 2.3.26, the minimum of the simplex is in F.

To answer the question whether such a direction would exist, we can solve the following LP for a specific facet F maximizing the lower bound of a directional derivative. Consider the ordered vertex set $\mathcal{V} := \{v_1, v_2, \ldots, v_h\}$ and the vertex set of F given as $\{v_2, \ldots, v_h\} := \mathcal{V} \setminus \{v_1\}$. Focusing on the direction $v_1 - y$ with $y = \sum_{j=2}^{h} \lambda_j v_j$, we can demonstrate that there is a (maximum) positive directional derivative,

if it exists, by solving the LP

$$\begin{aligned} & \max \sum_{i=1}^{n} z_i \\ \text{s.t.} \quad & z_i \le \underline{G}_i(v_{1i} - y_i),\ i = 1, \ldots, n \\ & z_i \le \overline{G}_i(v_{1i} - y_i),\ i = 1, \ldots, n \\ & \sum_{j=2}^{h} \lambda_j = 1 \\ & y = \sum_{j=2}^{h} \lambda_j v_j \\ & \lambda_j \ge 0, j = 2, \ldots, h. \end{aligned} \tag{2.77}$$

If the result is positive, we have proven theoretically that all minima of f over S are on F. However, computationally, better to ensure that the resulting direction is monotone by checking $\underline{d^T \boldsymbol{G}} > 0$ by interval computations.

Example 2.3.37. We can now show how LP (2.77) is working, following the illustration in Figure 2.26 for facet $F = \text{conv}\{v_1, v_2\}$. Although the lower bound on the directional derivative of $d = v_3 - c$ is not positive, the LP will provide a solution $y = (3, 0, \frac{3}{4})^T$ with an objective function value of 2. The corresponding direction is also illustrated with an arrow between $(3, 0, \frac{3}{4})^T$ and v_3 in Figure 2.26.

In a procedure for searching for such a facet, in the worst case we need to solve LP (2.77) h times. Instead, we might solve only one Mixed Integer Programming problem (MIP) where a binary variable δ_j selects the facet corresponding to the most positive directional derivative:

$$\begin{aligned} & \max \sum_{i=1}^{n} z_i \\ \text{s.t.} \quad & z_i \le \underline{G}_i d_i,\ i = 1, \ldots, n \\ & z_i \le \overline{G}_i d_i,\ i = 1, \ldots, n \\ & \sum_{j=1}^{h} \lambda_j = 1 \qquad \sum_{j=1}^{h} \delta_j = 1 \\ & d = \sum_{j=1}^{h} \delta_j v_j - \sum_{j=1}^{h} \lambda_j v_j \\ & \lambda_j \ge 0, \delta_j \in \{0, 1\}, j = 1, \ldots, h. \end{aligned} \tag{2.78}$$

MIP (2.78) can be used to replace LP problem (2.77) if such a monotonously increasing direction towards one of the vertices exists. If this is not the case, the solution is 0 with the direction $d = 0$. However, there still may exist an increasing direction according to LP (2.76).

Example 2.3.38. For our example in Figure 2.26 with $S = \text{conv}\{v_1, v_2, v_3\}$ the MIP (2.78) finds indeed the positive objective function value of $\sum_{i=1}^{n} z_i$, stating that the facet corresponding to $\delta_3 = 1$ provides the maximum derivative lower bound.

The two steps, looking whether a monotone direction exists and identifying which facet contains all the minima (if any), can also be done in one step. Thus, instead of solving LP (2.76) and (2.77) h times (or

MIP (2.78)), we can solve directly one MIP:

$$\begin{aligned} & \max \sum_{j=1}^{h} \delta_j \\ \text{s.t.} \quad & z_i \le \underline{G}_i d_i,\ i = 1, \dots, n \\ & z_i \le \overline{G}_i d_i,\ i = 1, \dots, n \\ & \sum_{j=1}^{h} \lambda_j = 1 \qquad \sum_{j=1}^{h} \mu_j = 1 \qquad \sum_{j=1}^{h} \delta_j \le 1 \\ & d = \sum_{j=1}^{h} (\lambda_j - \mu_j) v_j \\ & \sum_{i=1}^{n} z_i \ge \varepsilon \\ & \mu_j \ge \delta_j \quad \forall j = 1, \dots, h \\ & \mu_j, \lambda_j \in [0,1], \delta_j \in \{0,1\}, j = 1, \dots, h, \end{aligned} \tag{2.79}$$

where $d = x - y$, for $x, y \in S$, so $x = \sum_{j=1}^{h} \lambda_j v_j$ and $y = \sum_{j=1}^{h} \mu_j v_j$, thus $d = \sum_{j=1}^{h} (\lambda_j - \mu_j) v_j$.

A solution with $\mu_k = 1$ and $\mu_j = 0, j \neq k$ represents a direction d pointing to vertex v_k. A solution with $\lambda_k = 1$ and $\lambda_j = 0, j \neq k$ represents a direction pointing from vertex v_k. We connect μ_j with binary variables δ_j such that $\delta_j = 1$ implies $\mu_j = 1$. The inequality $\sum_{i=1}^{n} z_i \ge \varepsilon$ with $\varepsilon > 0$ assures d is a monotone direction. Moreover, it forces $\mu \neq \lambda$, because otherwise $z = 0$ as well. If no monotone direction exists, (2.79) has no feasible solution.

The objective is to maximize the sum of δ_j, meaning that we aim at finding a monotone direction with $\mu_k = 1$ corresponding to $\delta_k = 1$. In this case, we know that facet $F = \text{conv}(\mathcal{V} \setminus \{v_k\})$ contains all minima according to Proposition 2.3.26. If the objective is zero, there is no facet containing all the minima, i.e. there is no $\delta_k = 1$, but there is a monotone direction d.

Example 2.3.39. Following our example in Figure 2.26 with $S = \text{conv}\{v_1, v_2, v_3\}$ the MIP (2.79) finds the positive objective function value 1 for $\sum_{i=1}^{n} z_i$, stating that the facet corresponding to $\delta_3 = 1$ provides a positive directional derivative for $d = v_3 - v_1$. Notice, that this direction is not the maximum directional derivative, but still positive, which is the main question.

Interestingly, solving the LP-s and MIP-s in Matlab, the necessary time for this example was counter-intuitive: LP (2.76) 0.521, LP (2.77) 0.033, while MIP (2.79) 0.048 seconds.

We investigated whether the counter-intuitive result of a smaller solution time for the MIP than for the LP is a general trend. Therefore, we compared the solution time and effectiveness of formulations (2.76), (2.77), (2.78), and (2.79). We took 447 simplices from a branch-and-bound process over the functions Hartman 3, 4, and 6 that, as the name suggests, have dimension 3, 4, and 6. We have used the routines `linprog` and `intlinprog` in Matlab setting IntegerTolerance to 1e-6 and ConstraintTolerance to 1e-8. The result is given in Table 2.6. In each line of the table we give for the LP and MIP formulations the percentage of the effectiveness measured as proven monotonicity. For instance, for problem Hartman 3, LP (2.76) proved in 81.6% that there is a monotone directional derivative.

We can prove there is a monotone decreasing direction from any facet by solving LP (2.77) for all vertices, or by solving any of the MIP-s once. Comparing the formulations, LP (2.77) is the strongest, while MIP (2.79) is the weakest due to the ε in its formulations, which is hard to set together with the tolerances. The percentage development shows that, as the found monotone directions are less and less solving LP (2.77), MIP (2.78), and MIP (2.79), and the percentage where the best results are not found goes up in the same order.

Surprisingly, the average computing time is the smallest for MIP (2.78), followed by MIP (2.79) or LP (2.76), and the slowest is LP (2.77). The latter is no surprise as in that case we added up the time needed to solve LP (2.77) for all facets, or until it found a monotone decreasing direction from a facet.

2.3.4 Keeping track of border facets

To know the border status of a given facet, we use a labelling system to find out which minimum dimensional face of D the facet or simplex is included in. Here, we consider only simplicial feasible regions, as [104] deals with hyper-rectangle feasible sets, which is easier to handle.

Table 2.6: Time and effectiveness of the LP and MIP formulations

	LP (2.76)	LP (2.77)	MIP (2.78)	MIP (2.79)
Hartman 3				
Monotone dir exists	81.6%	-	-	77.6%
No monotone dir	18.4%	-	-	18.4%
Mon.neg.dir.from_F exists	-	73.5%	73.5%	69.4%
No Mon.dir.from_F exists	-	24.5%	24.5%	18.4%
Best result not found		2.0%	2.0%	4.1%
Time	0.018	0.029	0.009	0.012
Hartman 4				
Monotone dir exists	92.3%	-	-	80.2%
No monotone dir	7.7%	-	-	7.7%
Mon.neg.dir.from_F exists	-	67.0%	60.4%	57.1%
No Mon.dir.from_F exists	-	33.0%	33.0%	23.1%
Best result not found	-	-	6.6%	12.1%
Time	0.014	0.030	0.008	0.016
Hartman 6				
Monotone dir exists	98.0%	-	-	87.3%
No monotone dir	2.0%	-	-	2.0%
Mon.neg.dir.from_F exists	-	83.1%	73.0%	72.6%
No Mon.dir.from_F exists	-	16.9%	16.9%	14.7%
Best result not found	-	-	10.1%	10.7%
Time	0.015	0.038	0.009	0.011

For a simplicial feasible set $D = \text{conv}(\mathcal{W})$, this is done by assigning to each face φ of D a label $\mathcal{B}(\varphi)$, starting with the vertex faces labelled $\mathcal{B}(v_j) = 0..010..0$ where the only 1 is the jth bit, for $j = 1, \ldots, n+1$. Each face φ, which is a convex combination of vertices $\mathcal{V} \subseteq \mathcal{W}$, the corresponding bit-string $\mathcal{B}(\varphi)$ has a value 1 for each vertex $v \in \mathcal{V}$ in the same position as in $\mathcal{B}(v)$. For instance, in Figure 2.27, the edge (v_2, v_3) has label 011, and the simplex itself has label 111. In fact, for any simplex face φ of the feasible set, its label is given by the bitwise or of the label of all its vertices. The complete face graph is given in Figure 2.28 for a 4-vertex simplex.

After bisecting the original set of vertices $\mathcal{W}$, we store the bisection points in set X. For instance, in Figure 2.27, we have $X = \{v_1, v_2, v_3, x_4, x_5\}$. We label all generated points $x \in X$, which serve as vertices for the partition sets. The label of point x is the same as the label of the minimum dimensional face φ of the feasible set x is in. For instance, the label of x_4 is 101, the same as the label of face $\text{conv}(v_1, v_3)$. In the bisection, a new vertex $x = \frac{v+w}{2}$ gets label $\mathcal{B}(x) = \textbf{BitOr}(\mathcal{B}(v), \mathcal{B}(w))$.

Given a simplex $S = \text{conv}(\mathcal{V})$, the question is what is the label of the (smallest dimensional) face φ it

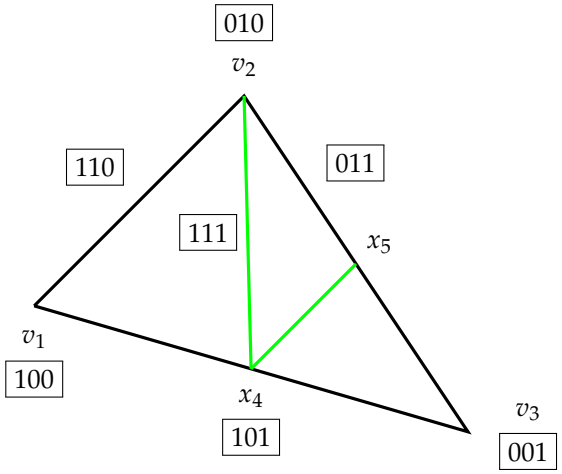


Figure 2.27: Faces of the feasible set with bisection points x_4 and x_5.

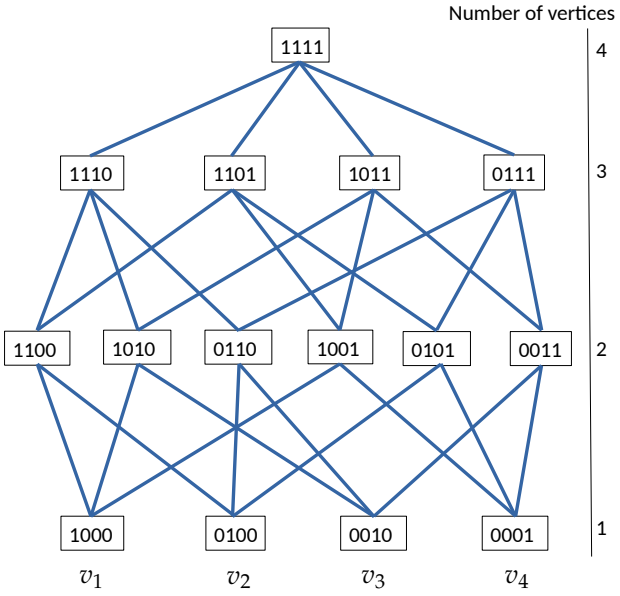


Figure 2.28: Face graph for 4-vertex simplex. Binary labels are shown in boxes for each face.

is included in. This is determined by the label $\mathcal{B}(\varphi) = \textbf{BitOr}(\mathcal{B}(\mathcal{V}))$, to be interpreted as a bitwise or on all its vertex labels. The number of ones of a bitstring $\mathcal{B}(\varphi)$ is denoted by $|\mathcal{B}(\varphi)|$, giving the number of vertices of φ. According to Definition 2.3.3, $(h-1)$-simplex S is border if there exists an $(h-1)$-simplex face φ of the feasible set including S ($h \leq n$).

Proposition 2.3.40. *Given $(h-1)$-simplex $S = \text{conv}(\mathcal{V})$ with $h \leq n$, if $|\textbf{BitOr}(\mathcal{B}(\mathcal{V}))| = h$, then simplex S is border.*

Proof. Consider the face φ, which is the minimal dimensional face containing S, i.e. label $\mathcal{B}(\varphi) = \textbf{BitOr}(\mathcal{B}(\mathcal{V}))$. As $|\textbf{BitOr}(\mathcal{B}(\mathcal{V}))| = h$, we have $|\mathcal{B}(\varphi)| = h$, thus φ is an $(h-1)$-simplex. Therefore, S is enclosed by an $(h-1)$-simplex face of the feasible set and thus is border. □

For an example, let us check the edges in Figure 2.27. The edge (x_4, v_3), half of the original edge (v_1, v_3), is border, as $|\textbf{BitOr}(\mathcal{B}(\{x_4, v_3\}))| = |\textbf{BitOr}(101, 001)| = |101| = 2$ corresponding to face $\text{conv}(\{v_1, v_3\})$. In contrast, edge (x_4, x_5) is not border, because $|\textbf{BitOr}(\mathcal{B}(\{x_4, x_5\}))| = |111| = 3 \neq 2$. In fact, the minimum dimensional face it is included in, is D itself.

As we have seen in the theoretical results, for monotonicity testing the border certification of facets is of importance. One way to do so is to follow Proposition 2.3.40. One of our research questions is whether it might be more efficient in an algorithm to store the border status of the facets in a Boolean vector β indicating for each facet whether they are border or not. Then, we can transfer the status β to facets of newly generated simplicial subsets instead of evaluating the border status $|\textbf{BitOr}(\mathcal{B}(\mathcal{V}))| = h$ for all facets. Therefore, we introduce a border status for each facet of $(h-1)$-simplicial subset S. Each facet F corresponds to a vertex $v \in \mathcal{V}$ which is dropped from the vertex set S to obtain F. Let us denote it by F_v, and its marker by $\beta(F_v) \in \{true, false\}$. Storing this information with subset S requires storing a binary vector $\beta = (\beta_1, \ldots, \beta_h) = (\beta(F_{v_1}), \ldots, \beta(F_{v_h}))$ ordered in the same way as the vertex set and the corresponding facets.

2.3.4.1 Setting the border status for facets in Longest Edge Bisection refinement

Consider a simplicial subset S with ordered vertex set $(v_1, \ldots, v_i, \ldots, v_j, \ldots, v_h)$ and the corresponding facet border status vector $(\beta_1, \ldots, \beta_i, \ldots, \beta_j, \ldots, \beta_h)$. Longest Edge Bisection (LEB) along edge (v_i, v_j) generates the new point $x = \frac{v_i + v_j}{2}$ and two new subsets $S_1 = \text{conv}(\{v_1, \ldots, x, \ldots, v_j, \ldots, v_h\})$ and $S_2 = \text{conv}(\{v_1, \ldots, v_i, \ldots, x \ldots, v_h\})$. Both new simplices inherit the border status β from parent S, but we set the border status of the new facet to false: for S_1, we set β_j :=false and for S_2 we put β_i :=false. So, apart from the newly generated shared facet, all other facets inherit their border status from the bisected simplex. Figure 2.29 shows the first bisection of the original 3-simplex D with the border status of its facets.

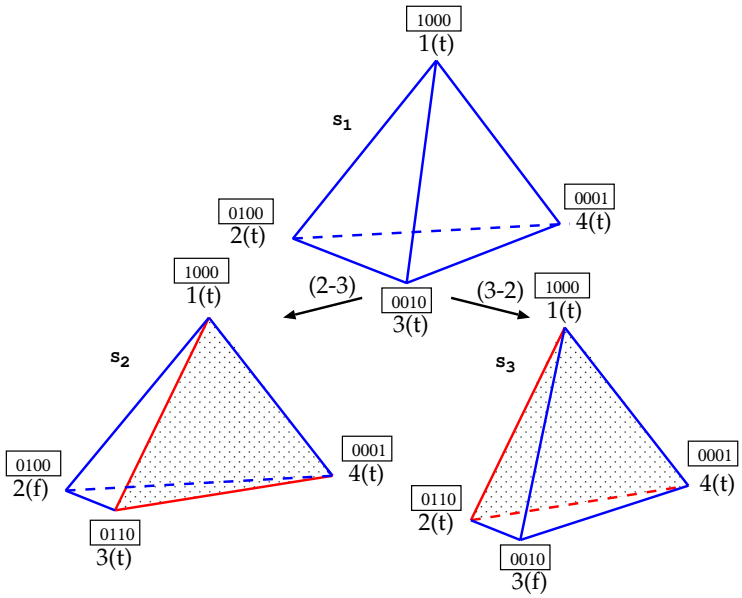


Figure 2.29: Part of the search tree where Longest Edge Bisection splits simplex S into two new simplices. Blue edges are border and red ones are non-border. The border status of vertex v_j and corresponding facet F_j is represented by j(t) or j(f): if F_j is border (t) or non-border (f). The dotted facet is the new facet generated by the bisection.

2.3.5 Setting border status for facets in simplicial dimension reduction

Consider a simplicial subset S with ordered vertex set $(v_1, \ldots, v_h)$ and corresponding border status $(\beta_1, \ldots, \beta_h)$. Let us focus on dropping one of the vertices and reducing the simplex to the corresponding border facet. Without loss of generality consider dropping v_1. The remaining facet $F_1 = \text{conv}(\{v_2, \ldots, v_h\})$ is border when β_1=true, otherwise F_1 is not border. What will be the new border status $(\beta_2, \ldots, \beta_h)$?

Proposition 2.3.41. *Given $(h-1)$-simplex $S = \text{conv}(\mathcal{V})$ and border facets $F = \text{conv}(\mathcal{V} \setminus \{v\})$ and $E = \text{conv}(\mathcal{V} \setminus \{w\}), v \neq w$. The intersection $F \cap E = \text{conv}(\mathcal{V} \setminus \{v, w\})$ is border.*

Proof. According to Proposition 2.3.40, the including faces φ_F, φ_E corresponding to F and E have a label $\mathcal{B}$ with $|\mathcal{B}(\varphi_F)| = |\mathcal{B}(\varphi_E)| = h-1$, because F and E are border. The intersection $F \cap E$ is an $h-3$-simplex, as we removed two vertices from $\mathcal{V}$.

Assume that $F \cap E$ is not border, then $|\mathcal{B}(\varphi_{F\cap E})| > h-2$. As $\varphi_{F\cap E} \subset \varphi_F$, we have that $|\mathcal{B}(\varphi_{F\cap E})| \leq h-1$ bits, so $|\mathcal{B}(\varphi_{F\cap E})| = h-1$.

As $(F \cap E) \cup \{w\} = F$, also $\textbf{BitOr}(\mathcal{B}(\varphi_{F\cap E}), \mathcal{B}(w)) = \mathcal{B}(\varphi_F)$. From $|\mathcal{B}(\varphi_{F\cap E})| = |\mathcal{B}(\varphi_F)| = h-1$ follows $\mathcal{B}(\varphi_{F\cap E}) = \mathcal{B}(\varphi_F)$. Similarly, we obtain $\mathcal{B}(\varphi_{F\cap E}) = \mathcal{B}(\varphi_E)$, therefore $\mathcal{B}(\varphi_F) = \mathcal{B}(\varphi_E)$ and $\varphi_E = \varphi_F$. Two border facets F and E cannot be included in the same $(h-1)$-face of the feasible set. Thus, $F \cap E$ must be border. □

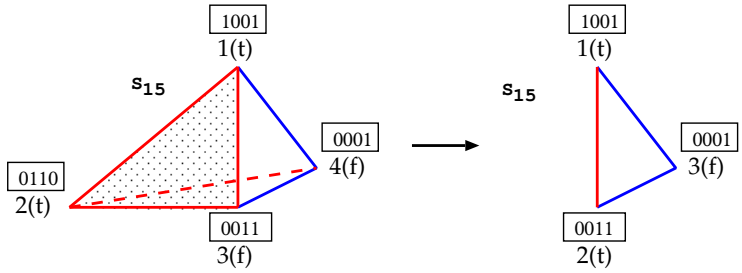


Figure 2.30: Reduction of S_{15} by removing vertex 1. Before reduction, S_{15} is included in face φ with label $\textbf{BitOr}(\mathcal{B}(\mathcal{V})) = 1111$. So S_{15} contains interior points in D. After reduction, $\textbf{BitOr}(\mathcal{B}(\mathcal{V})) = 1011$. The border status of facet F_1 is true and stays the same. Vertex 3 (4) becomes vertex 2 (3) and because their facet border status is false, their new status requires recalculation.

The consequence of Proposition 2.3.41 is that a border status $\beta(F_v) = true$ remains true after reduction and $\beta(F_v) = false$ requires recalculation using Proposition 2.3.40. Figure 2.30 shows an illustrative example.

2.3.6 Simplicial B&B algorithm (SBB)

Algorithm 2.2 SBB(f, P, α)

Input:
f: the n dimensional objective function.
P: initial simplicial partition of the search region D.
α: termination criterion.

1: $\Lambda = \{\emptyset\}$ ▷ *Storage structure*
2: $\tilde{f} = \infty$ ▷ *Incumbent value*
3: **for** $S \in P$ **do**
4: Evaluate $\boldsymbol{f}(S), \nabla \boldsymbol{f}(S)$ ▷ *+ other lower bounds*
5: $\tilde{f} \leftarrow \min\{\tilde{f}, \min_{v_j \in S} \overline{f}(v_j)\}$
6: **if** $\underline{f}(S) \leq \tilde{f}$ **and not** ReduceToFacets($\boldsymbol{f}$,S,Λ)
7: $\Lambda \leftarrow S$ ▷ *Store S and its bounds in Λ*
8: $S \leftarrow \Lambda$ ▷ *Retrieve S from Λ with smallest $\boldsymbol{f}(S)$ value*
9: **while** wid $([\underline{f}(S)), \tilde{f}]) > \alpha$ **do**
10: **if not** ReduceToFacets($\boldsymbol{f}$,S,Λ)
11: $\{S_1, S_2\} \leftarrow$ Divide(S) ▷ *Longest Edge Bisection*
12: **if** $\overline{f}($ new vertex $) < \tilde{f}$
13: $\tilde{f} = \overline{f}$(new vertex)
14: $\Lambda =$ CutOff(Λ) ▷ *Remove $S \in \Lambda : \underline{f}(S) > \tilde{f}$*
15: **for** each subset S_j **do**
16: Evaluate $\boldsymbol{f}(S_j), \nabla \boldsymbol{f}(S_j)$ ▷ *+ other lower bounds*
17: **if** $\underline{f}(S_j) \leq \tilde{f}$ **and not** (Monotone **and** $\nexists$ vertex labelled *border*) ▷ *See Proposition 2.3.21*
18: $\Lambda \leftarrow S_j$ ▷ *Store S_j and its bounds in Λ*
19: $S \leftarrow \Lambda$
20: **return** $[\underline{f}(S), \tilde{f}]$

Algorithm 2.2 uses an AVL tree[1] [2] Λ, a self-balancing binary search tree, for storing partition sets. Such a structure has a computational complexity of sorted insertion and extraction of an element of $\mathcal{O}(\log_2 |\Lambda|)$. Evaluated and not rejected simplices are sorted in Λ by non decreasing order of the bounds on the objective using any of the methods from Subsection 2.3.2. This means $[\underline{x}, \overline{x}] < [\underline{y}, \overline{y}]$ when $\underline{x} < \underline{y}$ or when $\underline{x} = \underline{y}$ and $\overline{x} < \overline{y}$. Simplicial partition sets having the same bounds are stored in the same node of the AVL tree using a linked list.

All vertices of a simplex are also stored in an AVL tree. Vertices may be shared among several simplices, such that we avoid duplicate storage. Although Algorithm 2.2 describes vertices to be evaluated in order to update the incumbent $\tilde{f}$ (see Algorithm 2.2, lines 5 and 13), their evaluation depends on the actual lower bounding method. The simplex S with the lowest value of $\boldsymbol{f}(S)$ is extracted from Λ (lines 8 and 19). The lower bound of $\boldsymbol{f}(S)$ is used in the stopping criterion of the algorithm (line 9).

Evaluation of a simplex S always includes computation of the natural inclusion $\boldsymbol{f}(S) = \boldsymbol{f}(\square S)$ of the objective function and inclusion of the gradient $\nabla \boldsymbol{f}(S) = \nabla \boldsymbol{f}(\square S)$ using Automatic Differentiation (see Algorithm 2.2, lines 4 and 16, and Algorithm 2.3 line 6). Other bounding methods can be applied afterwards in order to improve the calculated bounds in $\boldsymbol{f}$.

Simplices with a lower bound greater than the incumbent $\tilde{f}$ are rejected. They are also rejected using Proposition 2.3.21 when they are in the relative interior of the search space D and $0 \notin \nabla \boldsymbol{f}_i(\square S)$ (see Algorithm 2.2, lines 14 and 17, and Algorithm 2.3, line 7).

In case f is monotone on $\square S$ and $S \cap \partial D \neq \emptyset$, S can be reduced to a number of facets by Corollary 2.3.34 (see calls to Algorithm 2.3 from Algorithm 2.2, lines 6 and 10). From computational perspective it is better to label the vertex opposed to a facet to be *border* or *not-border*. Thus, a *border* vertex means that when it is removed from S, the remaining facet is on the boundary of D. Setting the border status later on is done by the procedures written in Subsection 2.3.4. If the search region is a simplex, P contains just the

[1] named after the inventors Adelson-Velsky and Landis

initial simplex, and all initial facets are at the boundary, such that all vertices are labelled *border*. In case the search region is a box, P contains the result of the combinatorial vertex triangulation of the box into $n!$ simplices [186, 200].

This technical detail has not been included in Algorithm 2.2 for the sake of simplicity. The specific triangulation is not appealing for large values of n. We use this here because box constrained problems are used to compare methods. Each of the $n!$ initial simplices has two border facets. They are determined by removing the smallest and largest vertex (numbered in a binary system), see the grey nodes in Figure 2.31. In the binary system, 0 is the lower bound and 1 is the upper bound of the given component of the box.

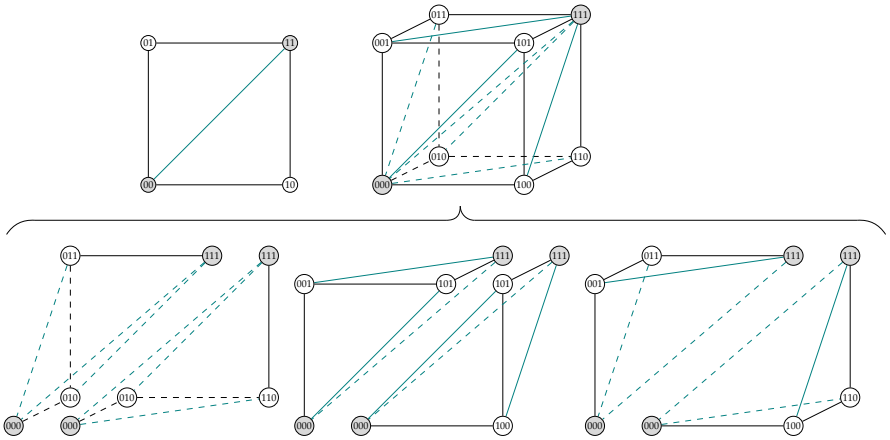


Figure 2.31: Combinatorial vertex triangulation of an hyper-rectangle. Vertices 000 and 111 are labelled *border* in all sub-simplices. Removing one of them leaves a facet that is completely on the boundary of D.

A simplicial partition set, which was neither rejected nor reduced, is divided using Longest Edge Bisection (LEB), see Algorithm 2.2, line 11. When several longest edges exist, the longest edge with a vertex with the lowest value of $\underline{f}$ and the other vertex having the highest value of $\underline{f}$ is selected. In case vertices are not evaluated, the first longest edge is selected.

In the described algorithm, we do not use all the potential of the monotonicity described in Subsection 2.3.3, as at this point we want to compare the inclusion functions.

2.3.7 Numerical results

This subsection is based on the work done by Leocadio G. Casado, but is presented for completeness.

Algorithm 2.2 was run on an Asus UX301L NoteBook with Intel(R) Core(TM) i7-4558U CPU and 8 GB of RAM running Fedora 32 Linux distribution. The algorithm was coded with g++ (gcc version 10.1.1) and it uses Kv-0.4.50 for Interval Arithmetic and Affine Arithmetic (AA). Kv uses boost libraries. Algorithms were compiled with `-O3 -DNDEBUG -DKV_FASTROUND` options and AA uses `#define AFFINE_SIMPLE 2` and `#define AFFINE_MULT 2` in Kv. For the Linear Programming, we use PNL 1.10.4 with `-DCMAKE_BUILD_TYPE = Release -DWITH_MPI = OFF`, as a C++ wrapper to LPsolve 5.5.2.0.

Notice that Kv Affine Arithmetic is slow in execution speed: *When the direction of rounding is fixed as "upward", the downward calculation is performed as "sign inversion"*, and it currently does not support division

Algorithm 2.3 ReduceToFacets(f,S,Λ)

```
1: Reduced=false
2: if Mon(f,S) and S ∩ ∂D ≠ ∅                          ▷ See Proposition 2.3.21
3:     for each border facet F do
4:         if not d^T ∇f(S) < 0                         ▷ See Corollary 2.3.34
5:             Reduced=true
6:             Evaluate f(F), ∇f(F)                     ▷ + other lower bounds
7:             if f(F) ≤ f̃ and not ( Monotone and ∄ vertex labelled border )
                                                        ▷ See Proposition 2.3.21
8:                 Λ ← F                                ▷ Store F and its bounds in Λ
9: return Reduced
```

by affine variables containing zero. Additionally, the execution time for Interval Arithmetic can be reduced on processors supporting Advanced Vector Extensions SIMD (AVX-512) (see last table at kv-rounding web page), which is not our case. Moreover, the PNL library has support for MPI, which is not used here.

Table 2.7 describes the studied instances. Their detailed description can be found in [116] and the optimization web page.

Table 2.7: Test problems. The problems are box constrained apart from KE2-1 and KE2-2 with search regions {(-3,-1),(1,1),(1.5,-2)} and {(-2,0),(0,-3),(2,3)}, respectively. An asterisk at n indicates that this is the selected dimension for a varying dimension test instance.

Instance	Description	n
KE2-1	Karhbet example 6 over simplex 1	2
KE2-2	Karhbet example 6 over simplex 2	2
GP2	Goldstein-Price	2
THCB2	Three Hump Camel Back	2
SHCB2	Six Hump Camel Back	2
G7	Griewank	7*
S4	Shekel 10	4
H3	Hartmann 3	3
H4	Hartmann 4	4
H6	Hartmann 6	6
L8	Levy	8*
SCH2	Schubert	2
MC2	McCormick	2
RB2	Rosenbrock	2*
MCH2	Michalewicz	2*
MCH5	Michalewicz	5*
ST2	Styblinski-Tang	2*
ST5	Styblinski-Tang	5*
DP2	Dixon-Price	2*
DP5	Dixon-Price	5*

The used termination accuracy is $\alpha = 10^{-6}$ and Interval Arithmetic is applied with Automatic Differentiation to obtain bounds of f and ∇f on $\square S$. The following notation is used to describe the variants to calculate lower bounds:

IA : Natural IA.

+CFcb : IA + Centered form on a box (see Not. 2.3.12) using the center of $\square S$.

+CFbb : IA + Centered form on a box (see Not. 2.3.12) using the Baumann point bb^- on $\square S$ (see Not. 2.3.14).

+CFcs : IA + Centered form on a simplex (see Prop. 2.3.15) using the centroid as the base-point and the gradient on $\square S$.

+CFbs : IA + Centered form on a simplex (see Prop. 2.3.15) using base-point $y = bs^-$ (see Def. 2.3.17) and the gradient on $\square S$.

+CFvs : IA + Centered form on a simplex (see Prop. 2.3.15) using base-point $y = \text{argmax}_{v \in S}\{\overline{f}(v)\}$ and the gradient on $\square S$.

+AA : IA + Affine Arithmetic lower bound (2.71) over $\square S$.

+LR : IA + Linear Relaxation bound (2.64) on $\square S$.

+LRS : IA + Linear Relaxation bound (2.65) on $\square S$, forcing the solution on S.

Rejection tests like the ones on monotonicity, are checked after the bound calculations. This is not efficient, but it allows us to compare the calculated bounds.

Improvement of the best function value found $\tilde{f}$ is done by point evaluation. Together with the IA bound calculation we evaluate simplex vertices. When other lower bound methods are added to IA, the evaluation of simplex vertices can be disabled in order to save computation. However, this may imply another (worse) update of $\tilde{f}$ and a different course of the algorithm, due to Longest Edge Bisection (LEB) by the first longest edge, instead of the best LEB [173].

The following points are evaluated for each method. +CFc* methods (*=b or s) evaluate only the center and +CFb* evaluate only base-points bb^- or bs^-. Such points are not stored. Notice that base point bb^- might be located outside the simplicial search region. +CFvs and +AA evaluate and store simplex vertices. +LR and +LRS evaluate and store box vertices when the search region is a box. Additionally, simplex vertices are evaluated when the search region is a simplex, because vertices of $\square S$ may be outside the search region and should not be used to improve $\tilde{f}$.

The +CF*s (*=c,b or v) methods only update lower bounds. The other methods also update upper bounds, which may affect the partition set storage order.

Figures 2.32 and 2.33 show the normalized (to the range [0,1]) number of simplex evaluations (NS) and execution time (T), respectively. The number of simplex evaluations can be considered as the number of iterations, as in each iteration one simplex is evaluated. The problems are sorted by NS in both figures. Reduction to border facets due to monotonicity does not occur in box constrained problems. It happens in the simplex constrained instances (see Corollary 2.3.34). The monotonocity test reduces the number of simplex evaluations significantly for all test problems. Without that test, the algorithm lasts more than the limit of 15 minutes for several problems. Therefore, we always apply the monotonicity test.

A value of $> 15m$ in Figures 2.32 and 2.33 means that i) the algorithm reached the 15 minute time limit, or ii) there was a problem with the Linear Programming solver in methods +LR and +LRS or iii) a division by zero occurred. The latter only happens for the +AA method for problem L8, because the Kv library does not implement division by zero in Affine Arithmetic.

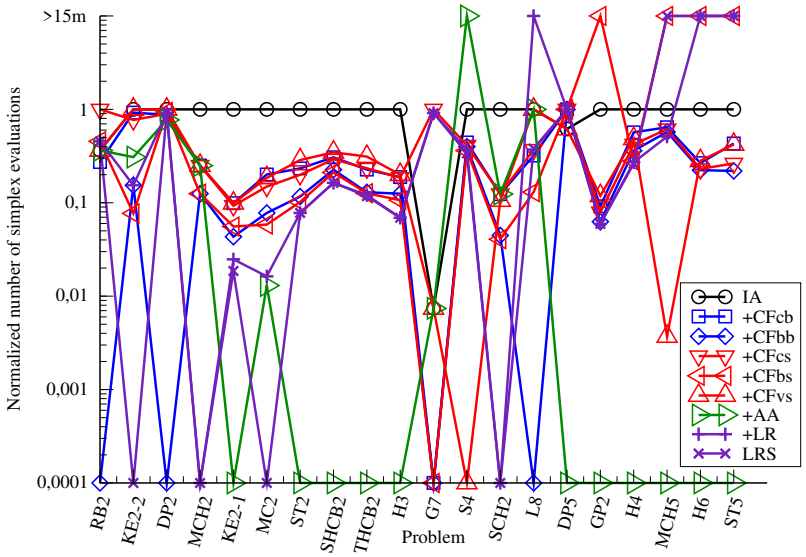


Figure 2.32: Normalized number of simplex evaluations in log scale. A value of $> 15m$ means time out of 15 minutes or execution error. The ranges of evaluated simplices per problem are as follow: RB2$\in$ [44, 66], KE2-2$\in$ [47, 60], DP2$\in$ [16, 112], MCH2$\in$ [128, 192], EX2-1$\in$ [186, 510], MC2$\in$ [434, 1,052], ST2$\in$ [558, 1,382], SHCB$\in$ [556, 1,646], THCB$\in$ [626, 1,986], H3$\in$ [2,286, 4,430], G7$\in$ [5,040, 5,314], S4 $\in$ [3,984, 5,288], SCH2$\in$ [4,862, 6,834], L8$\in$ [40,462, 40,662], DP5$\in$ [97,060, 188,476], GP2$\in$ [2,272, 167,800], H4$\in$ [53,368, 170,622], MCH5$\in$ [189,198, 210,356], H6$\in$ [2,641,024, 4,944,040], and ST5$\in$ [2,569,082, 6,358,328].

Focusing on the number of required simplex evaluations (NS), Figure 2.32 shows that the +AA method requires the least evaluations for most of the test cases. The second best methods regarding the NS metric are those using LP (+LR* and +CFbs). The +LRS lower bounding requires less simplex evaluations than +LR for some cases, and +CFcb has the best NS values for only a few test cases.

For smooth functions, the algorithm converges to a region which is captured by a convex quadratic function. To study the limit convergence behaviour of the algorithm, we run all variants over the so-called Trid function from [184], which represents a convex quadratic function. One can observe for this limit situation that the Linear Relaxation variants are relatively close models and require less simplex evaluations than other lower bounding methods. This means that, for all cases, the +LR variants have an advantage in the final stages of the algorithm. It is worth mentioning that, when the dimension increases, the required Linear Programming gets more time consuming and also the +AA variant starts to do better.

The execution time is a difficult performance indicator, as it depends on the used external subroutines.

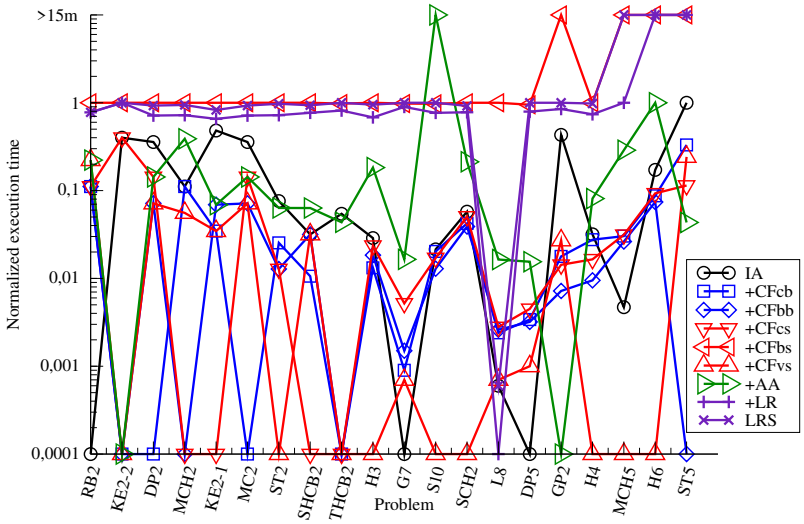


Figure 2.33: Normalized execution time (seconds) in log scale. A value of $> 15m$ means time out of 15 minutes or execution error. The ranges of execution time per problem are as follow: RB2$\in$ [0.005, 0.014], EX2-2$\in$ [0.005, 0.010], DP2$\in$ [0.005, 0.019], MCH2$\in$ [0.006, 0.024], EX2-1$\in$ [0.005, 0.034], MC2$\in$ [0.005, 0.019], ST2$\in$ [0.006, 0.085], SHCB2$\in$ [0.006, 0.101], THCB2$\in$ [0.008, 0.1], H3$\in$ [0.032, 0.415], G7$\in$ [0.048, 6.69], S4 $\in$ [0.056, 0.995], SCH2$\in$ [0.104 0.696], L8$\in$ [0.566, 159.124], DP5$\in$ [0.526, 61.352], GP2$\in$ [0.051, 1.461], H4$\in$ [1.196, 19.844], MCH5$\in$ [3.611, 60.438], H6$\in$ [51.057, 187.538], and ST5$\in$ [58.476, 184.924].

Figure 2.33 provides normalized values. In the first 9 test cases (ordered according to NS), the execution time is similar for most of the methods apart from those using LP (+LR* and +CFbs). In fact, methods using LP are in general the most time consuming due to the called routines, followed by +AA which avoids solving an LP due to (2.71). According to the Kv library documentation, Affine Arithmetic is slow and its implementation could be improved.

The +CFvs method requires the least execution time for most of the instances. Comparing +CFvs with other +CF* methods, the centered form used in +CFvs has to evaluate one sum term less and the base-point vertex can already have been evaluated and stored. On average, +CFbb is the best method, but this is because it is the best for the ST5 test problem, which is one of the most time consuming instances.

2.3.8 Conclusions

In simplicial branch-and-bound methods, the determination of the lower bound is of great importance. The Interval Arithmetic lower bound on the interval hull of a simplex can be tightened by additional calculations at a given cost. Several methods have been described and investigated. We have used the centered form with several base-points over a simplex and the interval hull of a simplex. The use of Affine Arithmetic and a Linear Relaxation over the interval hull and over the simplex has also been presented. Moreover, we introduced several theoretical results about monotonicity that can be applied to construct new rejection tests.

Results on a set of well-known low dimensional test instances show that Affine Arithmetic is a promising method to get lower bounds over a simplex. It requires the least number of simplex evaluations in many problems. However, its computational time is larger than several other methods. In general, methods using a Linear Programming solver suffer the same drawback requiring more time. We found that the monotonicity tests were essential for the reduction of computing time.

The method requiring the least computing time in several test problems is the one based on the centred form on a simplex using the vertex of a simplex with the highest function value as a base-point. The vertex can already have been evaluated and stored, and the centered form requires one less additional function evaluation.

This means that it is preferable to evaluate cheap lower bounds that reuse previous information over more simplices than expensive lower bounds over less simplices for low dimensional instances.

Chapter 3

Modelling costumer selection rules in competitive facility location problems

The estimation of market share that can be captured by a facility in a competitive environment where other facilities offer the same product is a topic of major concern for managers, as the survival of a facility depends on the revenues it can obtain, and those revenues largely depend on the market share. Where to locate a facility is a strategic decision which cannot easily be altered, as the location of a facility usually requires a massive investment. But how do we choose the right location for a new facility?

Competitive location problems concerning optimally placing facilities in a competitive environment have been widely developed for a number of contextual applications in the traditional retail sector, see for instance the survey papers of [70, 71] and [152], and the references therein. They vary in the ingredients which form the model. For instance, the *location space* may be the plane, a network or a discrete set. We may want to locate just one or more than one new facility. The competition may be *static*, which means that the competitors are already in the market and the owner of the new facility knows their characteristics, or *with foresight*, in which the competitors are not in the market yet but they will be soon after the new facility enters. Demand is usually supposed to be concentrated in a discrete set of points, called *demand points*, and it can be either *inelastic* or *elastic*, depending on whether the goods are essential or inessential.

It is also necessary to specify what the *attraction (or utility) function* of a customer towards a given facility is. Usually, the attraction function depends on the distance between the customer and the facility, as well as on other characteristics of the facility which determine its *quality*.

The *patronising behaviour* of the customers must also be taken into account, since the *market share* captured by the facilities depends on it. This is the topic this chapter is devoted to. For instance, it is not uncommon to see in the literature papers where consumers shop at the closest store supplying a specific product or service. But, does this assumption reflect consumer behaviour? It seems more realistic to admit that consumers do not merely consider distance when choosing retail outlets. Also, consumers may patronise more than one facility to satisfy their demand. Consumer choice behaviour literature studies the key variables that a customer takes into account to patronise one or another facility, and how these variables interact.

A common classification of the consumer choice behaviour states that this can be done in three groups [48]:

- The first one includes models that rely on some "normative assumption" regarding consumer travel behaviour. This hypothesis is too simple and is useful only in a limited number of applications. The classic example is the so-called *deterministic rule*, which says that "consumers patronise the nearest outlet that provides the required goods or service". This hypothesis has not found much empirical support, except in areas where shopping opportunities are limited and transportation is difficult.

- The second group uses information revealed by past behaviour to understand the dynamics of retail competition and how consumers choose among alternative shopping opportunities. [113] was the first one to use the revealed preference approach to study retail store choice. The Huff probability formulation, known as the *probabilistic rule*, uses distance (or travel time) from consumer zones to retail centres and the size of retail centres as inputs to find the probability of consumers shopping at a given retail outlet.

- The third group of models estimates the consumer utility function from simulated choice data using information integration, conjoint, or logit techniques. Instead of observing past choices, these methods use consumer evaluations of hypothetical store descriptions to calibrate the utility function. The best representative model of this group is the one developed by [92] based on game theory.

Clarkon [42] have pointed out that firms prefer the revealed preference approach to model consumer store-choice behaviour. This approach is preferred to normative models since it more faithfully reflects real consumer behaviour, and to the direct utility approach because it is simpler since it uses surveys and linear regression instead of conjoint, logit techniques, or game theory. We follow the revealed preference approach in this chapter.

The two customer choice rules commonly used in literature are the following:

Deterministic (or binary) rule: it assumes that the full demand of a customer is satisfied by *only one* centre, the one to which he/she is attracted most, disregarding all other facilities which are less attractive, even those whose difference in attraction is very small.

Probabilistic rule: it assumes that a customer splits his/her demand probabilistically over *all* facilities in the market proportionally to his/her attraction to each facility.

Hotelling [112] was the first to propose the deterministic choice rule for a simple model on a line. That is why competitive location models using this rule are also referred to as Hotelling models. The first two papers that introduced location models in a more general space assuming that customers patronize the closest facility were [66] in the plane and [96] on a network.

Huff [113, 114] described the gravity model suggested by Reilly [164], although he did not investigate any location problem. The first paper that considered the location problem based on the Huff rule was [58]. Later on, [76] and [1] introduced the design as an additional variable of the model, although an earlier version of location and design was already introduced in [60].

Deciding the location and the quality of the new facilities to be opened is a hard task. There are many competitive location models in literature for that purpose. See for instance [70, 71, 152] and the references therein. Depending on the location space, the number of facilities to be located, the type of competition, the demand and the way the attraction is measured, different location models have been proposed. Recently, in some of them, the quality of the new facility is also included as a variable to be determined (see [1, 64, 76, 161, 162, 172]).

The chapter is organized as follows. The classical probabilistic choice rule is reviewed in the next section in the context of a continuous competitive facility location and design model. The partially probabilistic choice rule is then introduced in Section 3.2 based on the paper [80]. In Section 3.3 we present another new choice rule, named multi-deterministic choice rule [79]. These new rules may, in some cases, model the patronising behaviour of customers closer to reality than other existing rules in many practical applications. The exact B&B method will be used in Section 3.4 to research up to what extent the use of a particular customer choice rule may affect the decision about the optimal location of a new facility. The chapter ends with Section 3.5 offering some conclusions.

This chapter contains those parts of papers [80, 79] where my contribution was significant, although done together with José Fernández. These articles have been very well received in the literature, having many citations compared to their age.

3.1 The probabilistic choice rule

In the rest of this chapter, in order to fix ideas, we assume the following scenario: a chain wants to locate a new single facility in a given area of the plane, where there already exist other facilities in the vicinity offering the same goods or product. Some of those facilities may belong to the locating chain. The demand is supposed to be fixed and concentrated at given demand points, whose locations and buying powers are known, as well as the location and quality of the existing facilities. The attraction of a demand point towards a facility is modelled multiplicatively as quality divided by perceived distance. This generalizes the law of retail gravitation of Reilly [164], who considered the perceived distance to be the squared distance. Quality was first estimated as store surface by Huff [113], and later several other store characteristics were incorporated by Jain [115] and Nakanishi [143]. For details see [65].

The objective is to maximize the *profit* obtained by the chain after the location of the new facility, to be understood as the income due to the market share captured by the chain minus its operational costs. Both the location and the quality of the new facility are to be found.

Notice, however, that the main conclusion remains valid for other competitive location models as well, that is the selection of the right customer choice rule is a critical issue for the location decisions of a firm that wants to set up new facilities.

In order to give a mathematical formulation of location models using the different customer choice rules, the following notation will be used:

Indices

- i index of demand points, $i \in I = \{1, \ldots, i_{\max}\}$.
- c index of competing chains, $c \in C = \{1, \ldots, c_{\max}\}$ where chain $c = 1$ is the locating chain.
- j index of existing facilities, $j \in J = \{1, \ldots, j_{\max}\}$ where $j \in J_c = \{j^c_{\min}, \ldots, j^c_{\max}\}$ belongs to chain $c \in C$ with $j^1_{\min} = 1$, $j^1_{\max}$ is the number of facilities in the locating chain, $j^c_{\max} = j^{c+1}_{\min} - 1$ and $j^{c_{\max}}_{\max} = j_{\max}$. Clearly, $J = \cup_{c \in C} J_c$

Variables

- x location of the new facility, $x = (x_1, x_2)$.
- α quality of the new facility.

Input data

- p_i location of demand point i.
- w_i demand (or buying power) at p_i, $w_i > 0$.
- f_j location of existing facility j.
- d_{ij} distance between demand point p_i and facility f_j, $d_{ij} > 0$.
- α_j quality of facility f_j, $\alpha_j > 0$.
- $d_i^{\min}$ minimum distance from p_i at which the new facility can be located, $d_i^{\min} > 0$.
- $\alpha_{\min}$ minimum level of quality for the new facility, $\alpha_{\min} > 0$.
- $\alpha_{\max}$ maximum level of quality for the new facility, $\alpha_{\max} \geq \alpha_{\min}$.
- S region of the plane where the new facility can be located.
- G^{ex} fixed costs due to the existing chain-owned facilities.

Miscellaneous

- $g_i(\cdot)$ a continuous non-negative non-decreasing function, which modulates the decrease in attractiveness as a function of distance.
- $d_i(x)$ distance between demand point p_i and the new facility.
- $u_{i0}(x, \alpha)$ attraction that p_i feels for the new facility; $u_{i0}(x, \alpha) = \alpha / g_i(d_i(x))$.

Computed parameters

- u_{ij} attraction that p_i feels for f_j (or utility of f_j perceived by the people at p_i). In this study, $u_{ij} = \alpha_j / g_i(d_{ij})$.
- u_i^c maximum attraction that p_i feels for any of the existing facilities of chain c, $u_i^c = \max\{u_{ij} : j \in J_c\}$.

Based on these assumptions, the market share captured by the chain when a deterministic rule is used is

$$M_D(x, \alpha) = \sum_{i \in I_D} w_i, \qquad \text{where} \qquad I_D = \{i \in I : \max\{u_i^1, u_{i0}(x, \alpha)\} \geq \max_{c \in C \setminus \{1\}} u_i^c\}.$$

In the previous formula we have assumed that, in case of ties in the attraction, customers choose the locating chain. Notice that in the deterministic rule it is assumed that the attraction of the customers at p_i towards a chain is determined only by the facility to which they are attracted most. The rest of the facilities do not play any role.

When a probabilistic rule is used, the market share captured by the chain is given by

$$M_P(x, \alpha) = \sum_{i \in I} w_i \frac{u_{i0}(x, \alpha) + \sum_{j \in J_1} u_{ij}}{u_{i0}(x, \alpha) + \sum_{j \in J} u_{ij}}.$$

In the probabilistic rule the attraction of the customers at p_i towards a chain is determined by all the facilities belonging to the chain. As we can see, it is assumed that the utility is *additive*: for instance, the utility for the first chain is given by $U_i^1(x, \alpha) = u_{i0}(x, \alpha) + \sum_{j \in J_1} u_{ij}$.

The problem to be solved is then

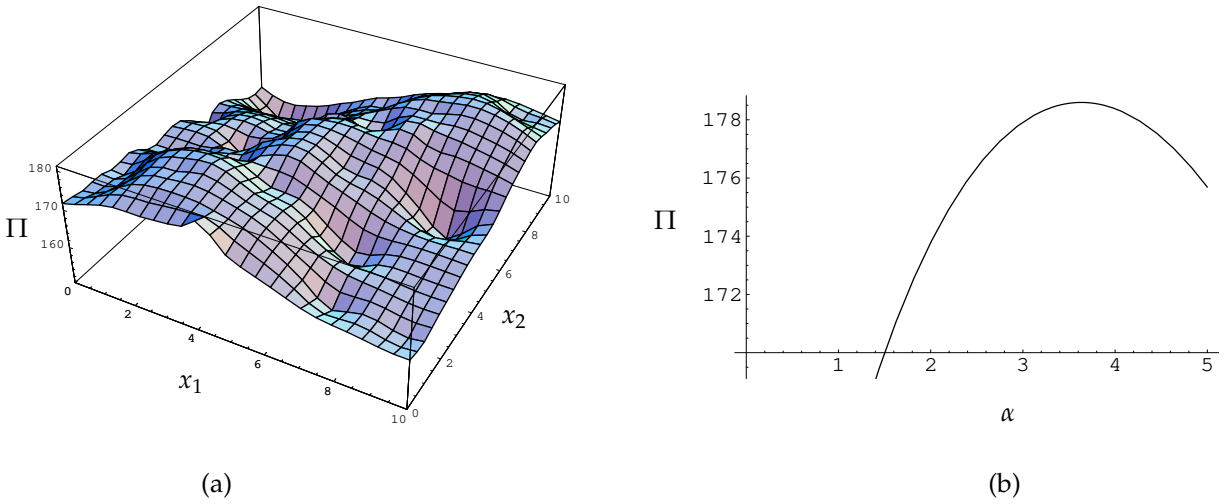
$$\begin{cases} \max & \Pi_P(x, \alpha) = F(M(x, \alpha)) - G(x, \alpha) - G^{ex} \\ \text{s.t.} & d_i(x) \geq d_i^{\min} \ \forall i \\ & \alpha \in [\alpha_{\min}, \alpha_{\max}] \\ & x \in S \subset \mathbb{R}^2, \end{cases} \tag{3.1}$$

where $F(\cdot)$ is a strictly increasing differentiable function which transforms the market share into expected sales. $M(x,\alpha)$ stands for either $M_D(x,\alpha)$ or $M_P(x,\alpha)$, $G(x,\alpha)$ is a function which gives the operating cost of a facility located at x with quality α, and $\Pi(x,\alpha)$ is the profit obtained by the chain.

In this work the function F is assumed to be linear, $F(M_P(x,\alpha)) = c \cdot M_p(x,\alpha)$, where c is the income per unit of goods sold. And for simplicity it is also assumed that c is the same across different facilities (which may not be true in the real world cases). Of course, other functions can be more suitable depending on the real problem considered (see [76]). Function $G(x,\alpha)$ should increase as x approaches one of the demand points, since it is rather likely that at around those locations the operational costs of the facility will be higher (due to the value of land and premises, which will make the cost of buying or renting the location higher). On the other hand, G will be in many cases a non-decreasing and convex function in the variable α, since the more quality we require of the facility, the higher the costs will be, at an increasing rate. But other more general functions could be considered if appropriate. In this study we have assumed G to be separable, i.e. of the form $G(x,\alpha) = G_1(x) + G_2(\alpha)$, with $G_1(x) = \sum_{i\in I} \Phi_i(d_i(x))$, with $\Phi_i(d_i(x)) = w_i/((d_i(x))^{\phi_{i0}} + \phi_{i1})$, $\phi_{i0}, \phi_{i1} > 0$ given parameters, and $G_2(\alpha) = e^{\frac{\alpha}{\beta_0}+\beta_1} - e^{\beta_1}$, with $\beta_0 > 0$ and β_1 given values (see [76] for more details and other possible expressions).

The parameters $d_i^{\min} > 0$ and $\alpha_{\min} > 0$ are given thresholds, which guarantee that the new facility is not located over a demand point and that it has a minimum level of quality, respectively. The parameter $\alpha_{\max}$ is the maximum value that the quality of a facility may take in practice. By S we denote the region of the plane where the new facility can be located. Distances are assumed to be computed with the help of a distance predicting function induced by a norm (see [73]). As for $g_i(d)$, it is usually considered to be of the form $g_i(d) = d^\lambda$ for some $\lambda > 0$ (see [113, 114, 143]) or $g_i(d) = \exp(\kappa d^\tau)$ for some $\kappa, \tau > 0$ (see [10, 107]). Notice that since $d_i^{\min} > 0$, then $g_i(d_i(x))$ is strictly positive for any feasible location x.

An example of the objective function for $i_{\max} = 100$, $j_{\max} = 10$, $j^1_{\max} = 4$ can be seen in Figure 3.1. As our function is three-dimensional, we can only depict it for a chosen quality or for a chosen location.



(a) (b)

Figure 3.1: (a) The objective function for a fixed quality, $\Pi(f_0, 2.9)$, and (b) for a fixed location $\Pi((5,5),\alpha_0)$.

It can be seen from Figure 3.1(a) that the objective function Π is neither convex nor concave and may have several local optima. Figure 3.1(b) illustrates the concavity of Π as a function of α, which can be easily proved for any fixed f_0.

3.2 The partially probabilistic choice rule

A demand point aggregates the demand of all the consumers that it represents. The probabilistic choice rule assumes that the demand at a demand point (or the buying power concentrated at it) is split probabilistically over *all* the facilities in the market proportionally to his/her attraction to each facility. Alternatively, it can be interpreted that customers patronize *all* the existing facilities (not just the most attractive to them), but the amount spent at each facility is proportional to the attraction that the customers feel towards the facility. This is the interpretation used throughout this work, as well as in most of the location papers in the literature.

However, does a customer really go to *all* the available facilities offering the product to satisfy his/her needs? To what extent does this assumption have an influence on the selection of the location and the quality of the new facilities to be located? This is the research question studied here. If the attraction that a demand point feels towards a facility is strictly positive, then, no matter how small that attraction is, according to the probabilistic rule the facility will capture part of the demand of the demand point. This may be unrealistic in some cases.

The idea of patronizing only some of the existing facilities is not new. For example, in marketing literature, some theories of consumer behavior involve thresholds or discontinuities. For instance, in [94] the authors investigate consumers' use of screening rules as part of a discrete-choice model and they present a model that accommodates conjunctive, disjunctive, and compensatory screening rules. However, in the location literature this topic has been hardly investigated. We are only aware of three location papers. In the first one, the so-called Pareto–Huff rule (see [147]) was introduced. According to it, a customer patronizes a more distant facility only if that facility has a higher quality. The second paper [183] introduces another variant, in which customers only patronize a facility provided that it offers a minimum attraction to them. Both papers deal with *network* location problems. The third paper is [93], where the related concept of reservation distance is used in a *discrete* location model. In all the three papers it is assumed that the quality of the new facilities to be located are fixed and given beforehand.

In this section we follow the idea in [183], i.e., customers at each demand point have associated a minimum level of attraction in order to patronize a facility, and then they share their buying power among the facilities that pass this threshold. However, this work differs from [93, 147] and [183] in two key points: (i) we analyze location problems in the plane, and (ii) the quality of the new facilities is not given beforehand, but it is considered to be an additional variable of the problem to be determined. Both details make the problem much more challenging from the optimization point of view. It also differs from [147] and [183] in two additional points: (iii) the function of the distance used in the attraction function does not have to be necessarily concave, and (iv) the locating chain does not have to be a new entrant in the market, it may already own some of the existing facilities. The variant of the probabilistic rule used in this work, which will be referred to as *partially probabilistic choice rule* hereinafter, is in many cases a more realistic assumption than that of the probabilistic rule. Notice that when the attraction threshold is very low, this rule is similar to the proportional one. For higher threshold values this procedure may coincide with the deterministic rule. For intermediate threshold values, the customers' choices may be different from both cases.

The use of attraction thresholds is related to the constrained choice-set concept used in [185] for the analysis of non-cooperative competition between multi-branch networks when consumers have heterogeneous preferences. The implications of thresholds has also been investigated in some discrete choice models studies from different perspectives, considering them as minimum perceptible changes in attributes required for a decision maker (DM) to differentiate among alternatives (see for example [30, 31, 32]). Those studies reveal that where perception thresholds exist in the population, the use of models without them leads to errors in estimation and prediction. However, in those papers the purpose is to analyse how the thresholds affect the decisions made by DMs, whereas in this study we assume that perception thresholds exist for the customers and want to investigate its final influence in the location of the new facility.

In the location literature the use of thresholds is usually related to the distance. For example, in the location of some emergency facilities, the emergency facility is assumed not to provide a good response to a demand point if it is at a distance greater than a given threshold (see, for instance, [108] or [128]). In the location of some undesirable facilities, the undesirable effects produced by the facility are supposed to be negligible for a demand point if the facility is further than a given threshold distance (see for instance [154] or [197]). In covering location problems, a demand point is assumed to be covered by a facility if their interdistance is less than a given threshold (see [15] and references therein). There are also competitive location papers where thresholds are used. For instance, there are location models in which the facilities to be located are required to capture a minimum level of demand (see, for instance, [44]). When the deterministic choice rule is used, the location of the competitive facilities is fixed, and only the location is to be determined, then profit changes only occur at particular threshold qualities (see [155]). There are also competitive location models where two facilities with the same quality are regarded as similar for a customer if the difference between the distances from the customer to the facilities is less than a given threshold (see [121] for a survey of that type of models on networks). The requirement of minimum level of attraction is also suggested in [6] (apart from [185] and [183]).

Notice that if for a given demand point none of the facilities attains the minimum attraction level, then the demand at that demand point will not be served. Hence, in our model, it may exist some unmet

demand. This feature also makes our model different from most of the models in the literature, where it is usually assumed that the whole demand is fully served. The most remarkable exceptions are the papers by Drezner and Drezner [62, 63], where the authors try to model the lost demand in a competitive environment similar to the one used in our work (the probabilistic choice rule is explicitly considered in those two papers). However, the way in which lost demand is considered in those papers is different from the way it is done in our model. In those papers, (i) *all* customers are *partially* served (part of their demand is not served) and (ii) the demand actually served at any given demand point is served *from all* the existing facilities. Furthermore, (iii) in [62] an exponential distance decay function is assumed to model the disutility of the facility as the distance between the demand point and the facility increases. On the contrary, in our model (i) some customers are fully served, whereas others may be not served at all; (ii) the demand at a given demand point may be served by only some of the existing facilities (not necessarily all) and the facilities which serve a demand point depend on the demand point; and (iii) a general distance decay function can be used in our model.

In order to give a mathematical formulation, we consider the same location scenario and notation as in the previous section. Let us denote $\overline{u}$ the minimum level of attraction that a facility must have for a customer to spend some of his/her buying power there. For simplicity, we assume that minimum level to be the same for all the demand points, but we could have a different value $\overline{u}_i$ for each demand point i. Let us define

$$\tilde{u}_{ij} = \begin{cases} u_{ij} & \text{if } u_{ij} \geq \overline{u} \\ 0 & \text{otherwise} \end{cases}$$

and

$$\tilde{u}_{i0}(x,\alpha) = \begin{cases} u_{i0}(x,\alpha) & \text{if } u_{i0}(x,\alpha) \geq \overline{u} \\ 0 & \text{otherwise.} \end{cases}$$

Then, the market share captured by the chain with the modified choice rule is

$$M_{PP}(x,\alpha) = \sum_{i \in I_{PP}} w_i \frac{\tilde{u}_{i0}(x,\alpha) + \sum_{j \in J_1} \tilde{u}_{ij}}{\tilde{u}_{i0}(x,\alpha) + \sum_{j \in J} \tilde{u}_{ij}}, \tag{3.2}$$

where

$$I_{PP} = \{i \in I : \max\{u_{i0}(x,\alpha), \max\{u_{ij} : j \in J\}\} \geq \overline{u}\}.$$

Notice that if for a given demand point i, $i \notin I_{PP}$, i.e.

$$\max\{u_{i0}(x,\alpha), \max\{u_{ij} : j \in J\}\} < \overline{u},$$

then the demand at i is not served by any facility. Hence, in this model, some of the demand may be unmet. Notice that, in particular, this means that the model is suitable only for inessential goods.

The corresponding continuous competitive facility location and design problem is the same as (3.1), except that $M_P(x,\alpha)$ is replaced by $M_{PP}(x,\alpha)$ as given by (3.2). Accordingly, its objective function will be denoted by Π_{PP}.

It is known that (3.1) is a hard-to-solve global optimization problem, with many local maximum points which are not global optimal points [76]. The corresponding problem with the modified choice rule is even harder, as in addition to this, it also has many discontinuities. As an example, consider Figure 3.2, which gives the graphs of the objective function on the location domain for a problem with setting ($i_{max} = 71, j_{max} = 5, j^1_{\max} = 2$) when the variable α is fixed to its optimal value for the partially probabilistic problem when $\overline{u} = 2$. The white holes in the graphs correspond to the forbidden regions around the demand points. As can be seen, this problem is also highly nonlinear (as the one in Figure 3.1) and require global optimization techniques to be solved, although the discontinuities of the problem due to the capture or loss of new customers, make it much more challenging. Something similar can be seen in Figure 3.3, where the graphs of the objective functions are shown for both the probabilistic and partially probabilistic rule when the location of the new facility is fixed at the optimal location of the partially probabilistic problem. As can be seen, whereas for the probabilistic choice rule the function is differentiable and concave with regard to α (left picture), this is no longer true for the problem with the partially probabilistic choice rule (right picture).

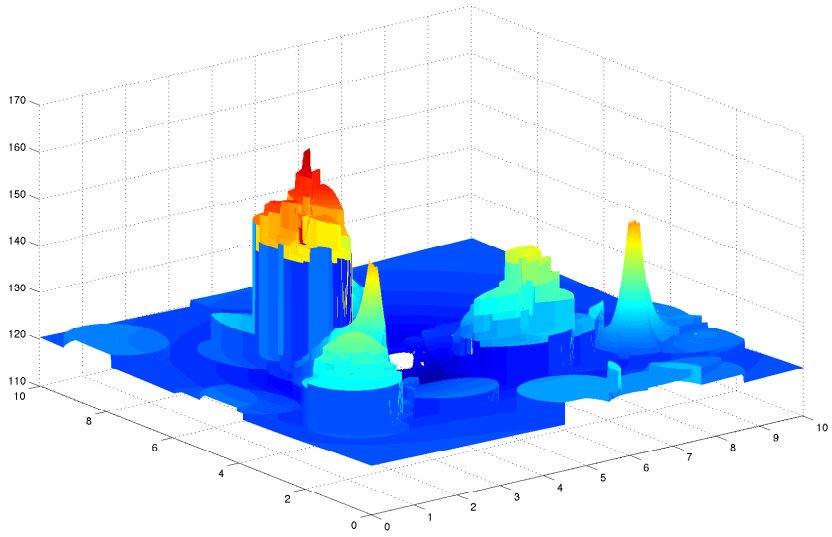


Figure 3.2: Objective function of an instance with setting $(i_{max} = 71, j_{max} = 5, j^1_{\max} = 2)$ when $\alpha = 1.545898$ with the partially probabilistic choice rule with $\overline{u} = 2$.

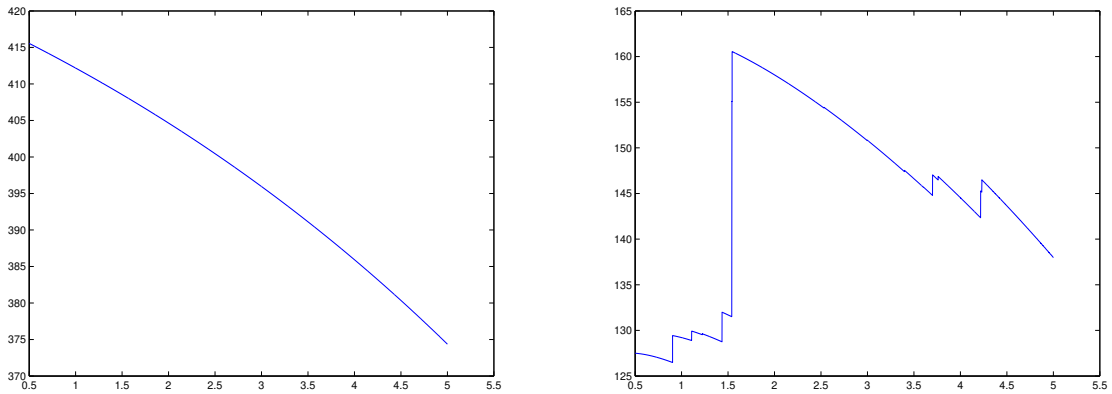


Figure 3.3: Objective function of an instance with setting $(i_{max} = 71, j_{max} = 5, j^1_{\max} = 2)$ when $(x_1 = 3.989257, x_2 = 7.065429)$: in the left figure, using the probabilistic rule, in the right figure with the partially probabilistic choice rule with $\overline{u} = 2$.

3.3 Multi-deterministic selection rule

In this section, we derive another costumer choice rule, based on a different aspect of costumer behaviour. Consider, for instance, the case of a customer who needs to do his/her weekly shopping. There are five supermarkets around his/her home, two of them belonging to chain A, and the other three to a different chain, B. Most likely, the customer will not do all the weekly shopping in a single supermarket, as some products may not be available there, or their price is lower in the supermarkets of the other chain. However, he/she will not go to all the supermarkets either, as he/she will find the same products, even with the same price, in all the supermarkets belonging to the same chain. So, the customer will decide to go to one of the supermarkets in chain A and to one of the supermarkets in chain B. In particular, the supermarket from each chain that he/she will choose will be the one for which he/she is attracted most. And the customer will do his/her weekly shopping in those two supermarkets not on a 50% basis: most likely, he/she will spend more money in the supermarket for which he/she feels more attraction. The multi-deterministic rule that we introduce next in this section tries to model this behaviour.

Hakimi already proposed something like this back in 1990, see Section 10.4 in [97]. He called it 'partially binary rule' (a name that we consider a bit misleading). Following this idea, Serra and colleagues ([45, 176]) presented a *discrete* location model using the multi-deterministic rule, but in which the utility of a facility

for a demand point was determined by the distance between them only. The multi-deterministic rule has also been addressed in *networks* in papers by Suárez-Vega and colleagues [181, 182], where some discretization results are shown. However, to the extent of our knowledge, this is the first study to address the problem in a *continuous* setting, and we do it using a general attraction function and including the quality as a third variable to be determined in the problem.

As in Section 3.1, we consider the problem of locating a single facility in the plane, with static competition and inelastic demand, where the attraction function depends on both the location and the quality of the facilities. These two last factors are the variables of the problem. The objective is again to maximize the *profit* obtained by the chain, to be understood as the income due to the market share captured by the chain minus its operational costs. As before, several firms are present in the market, but now customers split their demand among the firms by patronising only one facility from each firm, the one with the highest utility, and the demand is split among those facilities proportionally to their attraction.

The market share captured by the locating chain (chain $c = 1$) is

$$M_{MD}(x,\alpha) = \sum_{i\in I} w_i \frac{\max\{u_{i0}(x,\alpha), u_i^1\}}{\max\{u_{i0}(x,\alpha), u_i^1\} + \sum_{c=2}^{c_{\max}} u_i^c}. \tag{3.3}$$

As can be seen in the formula, it is assumed here that the attraction of the customers at p_i towards a chain is determined only by the facility of the chain to which they are attracted most. The rest of the facilities of the chain do not play any role. But unlike the deterministic rule, now all the chains capture part of the demand at p_i.

The market share captured by the new facility is

$$m_{MD_0}(x,\alpha) = \sum_{i\in I} w_i \frac{\tilde{u}_{i0}(x,\alpha)}{\max\{u_{i0}(x,\alpha), u_i^1\} + \sum_{c=2}^{c_{\max}} u_i^c},$$

where

$$\tilde{u}_{i0}(x,\alpha) = \begin{cases} u_{i0}(x,\alpha) & \text{if } u_{i0}(x,\alpha) \geq u_i^1 \\ 0 & \text{otherwise.} \end{cases}$$

The corresponding continuous competitive facility location and design problem is given by (3.1), where $M(x,\alpha)$ is given by (3.3). Figure 3.4 gives the graph of the objective function on the location domain for a problem with setting $(i_{max} = 71, j_{max} = 5, c_{max} = 2, j^1_{max} = 2)$ for a fixed value of the variable α. Ascan be seen, this problem is again a highly nonlinear optimization problem which requires global optimization techniques to be solved. Notice that when the number of facilities of each competing chain is equal to one, and the locating chain is a newcomer, the model reduces to the standard probabilistic model introduced in [76], described in Section 3.1.

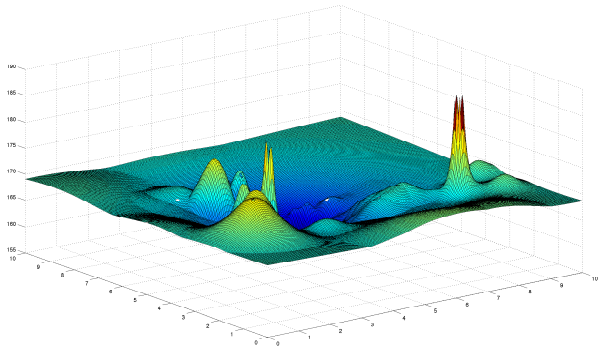


Figure 3.4: Objective function of an instance with setting $(i_{max} = 71, j_{max} = 5, c_{max} = 2, j^1_{max} = 2)$ when $\alpha = 0.5$.

3.4 Empirical comparison of the costumer selection rules

To what extent is the difference between the probabilistic and the new choice rules important when deciding the location of the new facility? How much does it affect the optimal profit? We study these and

other issues first using a quasi-real example of the location of a hypermarket, and then analysing some random problems. In both studies we will solve the same location problems using all rules, and using the exact interval branch-and-bound method mentioned in Subsection 1.3.2.1, so as to have the optimal solutions and carry out a fair comparison.

Let us denote by $\Pi_P(\cdot)$ the objective function of location problem (3.1) when the probabilistic choice rule is employed, by $\mathcal{L}_\mathcal{P}$ the list of solution boxes provided by iB&B, and by (x_P^*, α_P^*) the best point found by iB&B during the execution, and by $\Pi_N(\cdot)$, $\mathcal{L}_\mathcal{N}$, and (x_N^*, α_N^*) the corresponding items when a new choice rule is employed ($N = PP$ for the partially-probabilistic and $N = MD$ for the multi-deterministic rule).

We will compute the Euclidean distance between x_P^* and x_N^*, denoted by $dist_{loc}$, and the difference between the qualities, $dist_{qual} = |\alpha_P^* - \alpha_N^*|$ to measure the difference between the optimal solutions for $N = PP$ and $N = MD$.

We will also compute the relative profit loss incurred when the probabilistic choice rule is assumed in a problem where one of the new rule should have been chosen,

$$loss(P|N) = 100 \cdot (\Pi_N(x_N^*, \alpha_N^*) - \Pi_N(x_P^*, \alpha_P^*))/\Pi_N(x_N^*, \alpha_N^*),$$

and the relative profit loss incurred when one of the new choice rule is assumed in a problem where the probabilistic rule should have been chosen,

$$loss(N|P) = 100 \cdot (\Pi_P(x_P^*, \alpha_P^*) - \Pi_P(x_N^*, \alpha_N^*))/\Pi_P(x_P^*, \alpha_P^*),$$

to measure the cost of choosing the wrong model for the chain as a whole.

Finally, in order to measure the cost of choosing the wrong model in the profit increment because of the new facility, the relative profit lost due to the new facility only when the probabilistic choice rule is assumed in a problem where the new rule should have been chosen,

$$loss(P|N)_0 = 100 \cdot (Incr\Pi_N(x_N^*, \alpha_N^*) - Incr\Pi_N(x_P^*, \alpha_P^*))/Incr\Pi_N(x_N^*, \alpha_N^*),$$

is computed, where $Incr\Pi_N(x_N^*, \alpha_N^*) = \Pi_N(x_N^*, \alpha_N^*) - \Pi_N(before)$ and $Incr\Pi_N(x_P^*, \alpha_P^*) = \Pi_N(x_P^*, \alpha_P^*) - \Pi_N(before)$, and $\Pi_N(before)$ stands for the profit obtained by the chain before the location of the new facility. Analogously, $loss(N|P)_0$ will be computed, too.

3.4.1 A case study

First, we will investigate a quasi-real example dealing with the location of a shopping mall in an area around the city of Murcia, in south-eastern Spain. A working radius of 25 km around Murcia was considered. 632 558 people live within the circle, distributed over $i_{\max} = 71$ population centres, with population varying between 1 138 and 178 013 inhabitants. In this study, we have considered each population centre as a demand point, with buying power proportional to its total population (one unit of buying power per 17 800 inhabitants). Their position and population can be seen in Figure 3.5: each demand point is shown as a grey circle (or a black dot), whose radius is proportional to the buying power. Note that here the grey circles also show the forbidden regions. There are five shopping malls present in the area: two from a first chain A (marked with a red •, and three from another chain B, marked with a green ×). Figure 3.5 shows the location of each mall. The feasible set S was taken exactly as depicted in Figure 3.5, i.e., the smallest rectangle containing all demand points. This is approximately a square centred in Murcia and of sides close to 45 km.

The coordinates of the population centres and the malls were obtained with the geographical information system called VisualMap [193], and were rescaled from coordinates $([200, 245], [243, 285])$ to an approximate standard square $([0, 10], [0, 10])$. Thus, the units correspond approximately to 4.5 km. The minimum distance $d_i^{\min}$ at which the new facility must be from the population centre i was chosen to be $w_i/30$. The qualities of the existing facilities lie in the interval $[0.4, 4]$ and for the new facility in the interval $[0.5, 5]$. For more details about the data set, the interested reader is referred to [190].

The basic data described above have been used to define several different competitive market structures:

Scenario 'newcomer 1': $c_{\max} = 2$ (number of chains), and the number of existing facilities belonging to each chain is 0 and 5, respectively. Notice that in this case the locating chain (chain $c = 1$) has no existing facilities and all the existing facilities are assumed to belong to the same chain.

Table 3.1: Differences in the solutions obtained by the probabilistic and the partially probabilistic choice rules for the scenario 'small chain A'.

$\overline{u}$	0	0.5	1.0	1.5	2.0	2.5	3.0
quality	1.45	1.24	4.70	0.70	1.49	1.86	2.24
$dist_{loc}$		0.16	5.31	6.35	5.88	5.88	5.90
$dist_{qual}$		0.22	3.24	0.76	0.03	0.41	0.78
Π_{PP}(before)		176.2	151.4	141.1	133.4	127.4	123.6
$\Pi_{PP}(x^*_{PP}, \alpha^*_{PP})$		197.0	182.6	166.9	162.6	157.1	153.6
$G(x^*_{PP}, \alpha^*_{PP})$		12.5	47.4	11.0	17.8	20.6	23.6
$lost(P\|PP)$		0.1	8.3	5.9	8.1	8.7	9.3
$lost(PP\|P)$		0.1	8.0	9.8	11.4	11.6	12.1
$lost(P\|PP)_0$		1.4	48.6	37.9	45.0	46.1	47.4
$lost(PP\|P)_0$		0.8	77.7	94.2	110.2	112.5	117.1
$\%W_{TB}$		95.0	83.3	76.9	76.2	72.2	66.5
$\%W_{TA}$		95.0	91.4	85.5	84.4	82.0	78.9
$\%W_{CB}$		41.3	35.5	33.1	31.3	29.9	29.0
$\%W_{CA}$		49.1	53.9	41.7	42.3	41.7	41.6
$\%W_{new}$		7.9	19.1	8.6	11.0	11.9	12.6

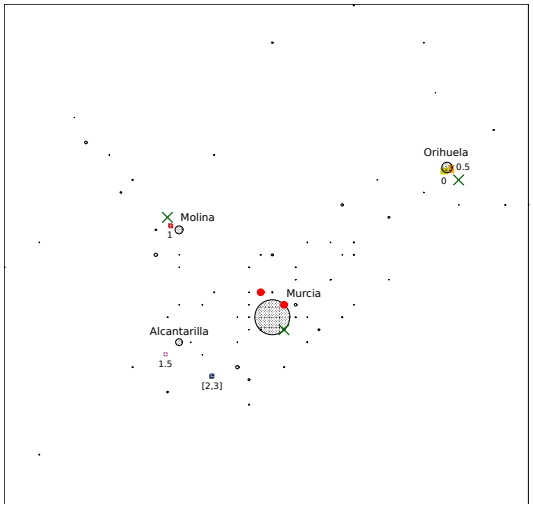


Figure 3.5: Scenario small chain A

Scenario 'newcomer 2': $c_{\max} = 3$, and the number of existing facilities belonging to each chain is 0, 3, and 2, respectively. Again, in this case, the locating chain has no existing facilities, so it is a new entering firm. But now the existing facilities are assumed to belong to two different chains. This scenario is only interesting for the multi-deterministic case, as for the (partially) probabilistic behaviour it is equivalent to 'newcomer 1'.

Scenario 'small chain A': $c_{\max} = 2$, and the number of existing facilities belonging to each chain is 2 and 3, respectively. The locating chain is the small one, chain A.

Scenario 'large chain B': $c_{\max} = 2$, and the number of existing facilities belonging to each chain is 3 and 2, respectively. The locating chain is the greater one, chain B.

3.4.1.1 Probabilistic choice rule versus partially probabilistic choice rule

As the demand actually served by the facilities may vary depending on the threshold value $\overline{u}$ employed for the partially-probabilistic rule (remember that we may have unmet demand), we will also compute the percentage of the total demand really served before ($\%W_{TB}$) and after ($\%W_{TA}$) the location of the new facility, the percentage of the total demand captured by the locating chain before ($\%W_{CB}$) and after ($\%W_{CA}$) the new facility is located, and the percentage of the total demand captured by the new facility ($\%W_{new}$). Notice that $\%W_{CB} + \%W_{new}$ may be different from $\%W_{CA}$, since the new facility may steal part of the demand to the existing chain-owned facilities (an effect known as cannibalization). The total demand in the region in all the cases is $W = \sum_{i=1}^{i_{\max}} w_i = 35.53$. For the newcomer case, $\%W_{CB}$ and $\%W_{CA}$ are omitted, as the locating chain has no existing facilities. For the sake of completeness, we also show the quality of the new facility for each value of $\overline{u}$.

The results obtained for each of the three scenarios are shown in Tables 3.1, 3.2, and 3.3, and Figures 3.5, 3.6, and 3.7, respectively.

As we can see, in the scenario 'small chain A' (see Table 3.1), even for the threshold $\overline{u} = 0.5$ (a small value with which 95% of the total demand is still served) we can observe that both the location and the quality of the new facility to be located are different from those of the probabilistic choice rule (see Figure 3.5, where the optimal locations for the different values of $\overline{u}$ are drawn as small squares in different colors). However, the differences are not too important, as the optimal solution is still to locate the new facility close to Orihuela, the second most populous city, in the North-East of the map, where one of the facilities of the competing chain is set up (the chain already has two facilities in the surroundings of the most populous city, Murcia, so this is not the optimal place to expand the chain, due to the cannibalization). In fact, the differences in objective function value are almost negligible. It is for $\overline{u} = 1$ where we can observe a big change, both in location and quality. In this case, the optimal location changes to the third most populous city, Molina (in the North-West of the map, where another facility of the competing chain operates). The reason is that Molina is closer to Murcia, and the new facility not only competes against the competitor's facility in Molina, but also attracts part of the demand from Murcia and Alcantarilla (the

Table 3.2: Differences in the solutions obtained by the probabilistic and the partially probabilistic choice rules for the scenario 'large chain B'.

$\overline{u}$	0	0.5	1.0	1.5	2.0	2.5	3.0
quality	0.58	0.50	4.33	3.87	0.92	1.16	2.25
$dist_{loc}$		0.07	2.13	1.57	0.28	0.28	0.89
$dist_{qual}$		0.08	3.75	3.29	0.35	0.58	1.67
Π_{PP}(before)		229.0	203.7	186.7	191.6	180.6	159.9
$\Pi_{PP}(x^*_{PP}, \alpha^*_{PP})$		236.0	236.6	212.5	215.8	203.2	189.4
$G(x^*_{PP}, \alpha^*_{PP})$		10.7	48.8	43.8	12.6	14.2	23.6
$lost(P\|PP)$		0.2	7.2	4.0	4.3	4.5	8.5
$lost(PP\|P)$		0.0	1.6	0.8	1.3	1.4	3.8
$lost(P\|PP)_0$		5.4	51.9	33.3	38.2	40.5	54.4
$lost(PP\|P)_0$		1.2	61.9	30.8	51.2	52.3	147.6
$\%W_{TB}$		95.0	83.3	76.9	76.2	72.2	66.5
$\%W_{TA}$		95.7	91.5	84.6	84.9	80.9	78.9
$\%W_{CB}$		53.7	47.8	43.8	44.9	42.3	37.5
$\%W_{CA}$		57.9	66.9	60.1	53.6	51.0	50.0
$\%W_{new}$		7.4	25.1	22.0	8.6	8.6	12.6

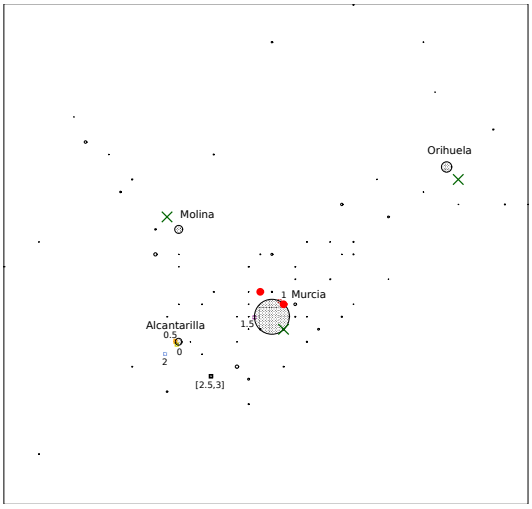


Figure 3.6: Scenario large chain B.

fourth most populous city), thanks to its high quality. Notice that when the facility is located in Orihuela, if $\overline{u} = 1$, even a high quality is not enough to attract demand from Murcia. As can be seen in Table 3.1, when $\overline{u} = 1$ an inadequate choice in the patronizing behavior of customers may lead to a considerable relative profit lost of around 8%. In fact, in this case the new facility captures 19.1% of the total demand, more than 1/3 of the demand captured by the chain. When $\overline{u} = 1.5$ we can observe another change in the location, which moves to the South-West, close to Alcantarilla and closer to Murcia; now the quality of the new facility is small, which reduces the costs, but still allows to capture most of the demand from Alcantarilla and its surroundings. Finally, for $\overline{u} = 2$, 2.5, and 3 a final change can be observed, where the location is the same for all those values, and only the quality changes. Now, although the new facility is at a similar distance from the most populous city, it is set up in a cheaper place and it is closer to the competitor's facility in Murcia, which allows to compete against it and to capture part of the demand from the South-West area (although at the cost of a higher quality).

Note that an increase in the market share captured does not necessarily means an increase in the profit, since the increment in the market share captured can be due to a higher quality or a better and more expensive location, and in both cases this implies a higher cost, and maybe the cost exceeds the income obtained from the market share captured. For instance, from $\overline{u}$ =0.5 to 1.0 the chain capture more market share (from 49.1% to 53.9%, which is an increase of 9.77%), but the profit decreases (from 197.0 to 182.6, a 7.30%) since the cost due to the new facility increases from 12.5 to 47.4, mainly due to the increment in the quality (from 0.22 to 3.24).

In the scenario 'large chain B', for both the probabilistic choice rule ($\overline{u} = 0$) and the partially probabilistic choice rule with threshold value $\overline{u} = 0.5$, the optimal solution is to locate the facility close to Alcantarilla, the fourth most populous city, with a very bad quality: since there exist no other facilities around, the new facility captures most of its demand (see Figure 3.6) with a low cost.

However, when $\overline{u} = 1$ the optimal location moves to the surroundings of Murcia, the most populous city, between the two existing facilities of the competing chain, and this, despite the fact that the chain already owns a facility in the South-East of that city; to compete against them, a high quality is required. Notice that with $\overline{u}$ =0.5, the total market share served before the location of the new facility was 95.0%; whereas with $\overline{u}$ =1.0 it is only 83.3%. This means that part of the demand at Murcia (and other cities around) is not served by the existing facilities, and so, there is an opportunity for the new facility to capture that unserved demand. In fact, after the location of the new facility, the total demand served increases to 91.5%. As we can see, in this case, $lost(P|PP) = 7.2\%$, a very high value taking into account that the locating chain is dominant in the market and that 91.5% of the total demand is served after the location of the new facility. In this case, the new facility captures 25.1% of the total demand, whereas the chain, considering all the facilities, gets 66.9% of the total demand, i.e., the new facility captures 37.5% of the demand of the chain. This is, however, at the expense of suffering some cannibalization: notice that $\%W_{CB} + \%W_{new} = 72.9\%$, more than $\%W_{CA} = 66.9\%$. When $\overline{u} = 1.5$ the optimal location moves to another part of the city of Murcia, far from the existing competing facilities, which allows reducing the quality, and hence, the costs, but still capturing a good part of the demand at Murcia, even though suffering some

Table 3.3: Differences in the solutions obtained by the probabilistic and the partially probabilistic choice rules for the scenario 'newcomer'.

$\overline{u}$	0	0.5	1.0	1.5	2.0	2.5	3.0
quality	5.00	5.00	4.92	4.16	4.37	5.00	2.24
$dist_{loc}$		0.10	0.67	0.04	0.01	0.09	1.29
$dist_{qual}$		0.00	0.08	0.84	0.62	0.00	2.76
Π_{PP}(before)		0.0	0.0	-0.0	0.0	0.0	0.0
$\Pi_{PP}(x^*_{PP}, \alpha^*_{PP})$		45.6	60.9	52.2	50.6	43.3	30.1
$G(x^*_{PP}, \alpha^*_{PP})$		56.7	55.7	46.8	49.3	56.6	23.6
$lost(P\|PP)$		0.9	11.8	3.7	4.1	45.0	17.4
$lost(PP\|P)$		0.7	5.8	3.8	2.3	10.2	64.5
$\%W_{TB}$		95.0	83.3	76.9	76.2	72.2	66.5
$\%W_{TA}$		95.7	91.5	84.6	83.4	80.8	78.9
$\%W_{new}$		24.0	27.3	23.2	23.4	23.4	12.6

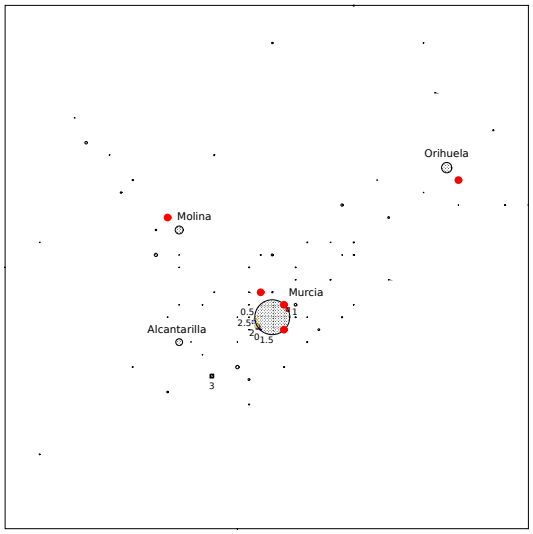


Figure 3.7: Newcomer

cannibalization. When $\overline{u}$ =2.0 the location changes again, close to Alcantarilla, again with a low quality. Since the costs are again much smaller, the profit increases, and this despite capturing a smaller percentage of the market. And finally, for $\overline{u} = 2.5$ and 3, a new location is obtained, similar to that of the small chain scenario, not too close to Murcia nor to Alcanrilla to reduce the location costs (see Figure 3.6), and with a medium quality. Notice that, as expected, the cannibalization decreases as $\overline{u}$ increases.

Concerning the 'newcomer' case, for $\overline{u} = 0$ to 2.5, the optimal solution is to locate the new facility in the surroundings of Murcia, the most populous city (see Figure 3.7). Even though there already exist three facilities around Murcia and there will be fierce competition, for the newcomer it is still the best option, as most of the demand is concentrated there. A high quality is required, though. It is important to highlight, however, that even with very small differences in location and quality, the relative profit loss incurred when an inadequate patronizing behavior of customers is assumed may be very high. See, for instance, the case $\overline{u} = 2.5$. In this case, the difference in location is very small ($dist_{loc} = 0.09$) and the difference in quality with regard to the probabilistic case is almost negligible. Nevertheless, $lost(PP|P) = 10.2\%$ and $lost(P|PP) = 45.0\%$. This clearly shows the big difference between the probabilistic and the partially probabilistic choice rule. For $\overline{u} = 3$ the relative profit lost is even higher, although in this case the location moves to a different place, the same as that of the small and large chain scenarios when $\overline{u}$ is 2.5 or more (not too close to Murcia nor to Alcantarilla to reduce the location costs), and the quality is also different from that of the probabilistic case. In this case, since $\overline{u}$ is very high, none of the existing facilities can capture the demand from Alcantarilla and the cities around, so the new facility can capture it with a low quality and in a cheap location.

Summarizing, we can see that both the location and/or the quality of the facility to be located may change drastically as the threshold value varies, and even when those changes are very small, the relative profit loss incurred for the chain when an inadequate choice rule is employed may be very high. This is due to discontinuities of the objective function, see Figures 3.2 and 3.3: every time a new demand point is captured or lost (and this may happen with a small change in the location and/or the quality), a jump in the objective function happens. This clearly shows that the selection of the choice rule in competitive location models should be made with care, and the assumption of the probabilistic choice rule commonly done in literature should only be considered when it is really the case.

3.4.1.2 The influence of the choice rule using the multi-deterministic choice rule

The results obtained for the multi-deterministic case are shown in Table 3.4. As we can see, for the second and third scenarios the differences are rather slight. However, in the 'newcomer 1' scenario, the relative profit loss incurred when the multi-deterministic choice rule is assumed instead of the probabilistic rule is more than 6%.

For the 'large chain B' scenario, the differences are much higher. The relative profit loss incurred for the chain when the probabilistic choice rule is assumed instead of the multi-deterministic rule is more than 10%. This is a rather high loss, especially taking into account that the locating chain, chain B, is dominant in the market, and after the location will have four facilities, against the two facilities of the competitor.

Table 3.4: Differences in the solutions obtained by the probabilistic and multi-deterministic choice rules.

Scenario	$dist_{loc}$	$dist_{qual}$	$loss(P\|MD)$	$loss(MD\|P)$	$loss(P\|MD)_0$	$loss(MD\|P)_0$
newcomer 1	0.67	0.14	1.06	6.31	1.06	6.31
newcomer 2	0.09	0.14	0.39	0.49	0.39	0.49
small chain A	0.15	0.07	0.02	0.02	0.16	0.17
large chain B	2.23	3.44	10.06	1.13	418.53	23.51

But the loss for the new facility is much higher, more than 400%, which clearly shows that the location chosen for the new facility in that case is completely wrong. If the patronising behaviour of customers was probabilistic, then the corresponding relative profit loss incurred for the chain and for the facility will be 1.13% and 23.51%, respectively. As we can see, although the loss for the chain as a whole is not that big, the loss for the new facility is rather high.

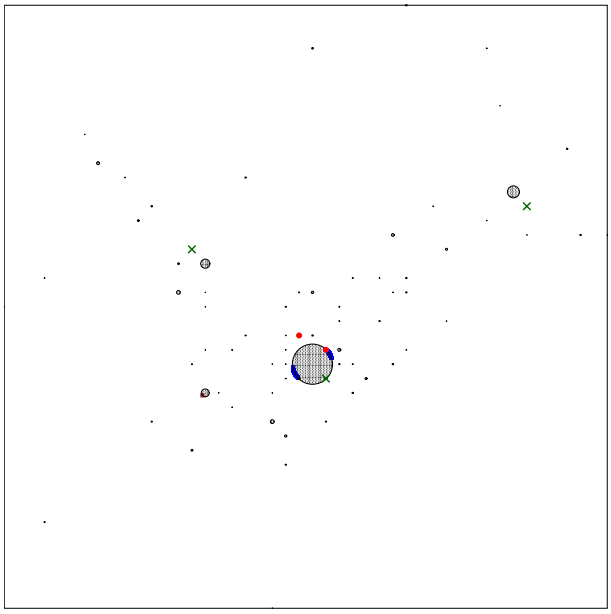


Figure 3.8: Case study: scenario large chain B.

In Figure 3.8, we can see a picture of this last scenario projected onto the 2-dimensional locational space. When the probabilistic choice rule is used, the boxes of the solution list $\mathcal{L}_{\mathcal{P}}$, marked in dark blue in the picture, are around the city of Murcia, the big grey circle where most of the inhabitants of the region live. Even though the chain already has a facility in the South-East of the city, it is still more advantageous for the chain to locate the new facility there, either close to one of the existing facilities of the competitor or opposite it, and also from the second facility of the competitor chain. However, when the multi-deterministic choice rule is used, those areas are no longer an optimal solution, as the existing facility of the chain already captures a large part of the demand from Murcia. Locating the new facility close to Murcia will not increase the captured demand too much (most of the demand that the new facility will capture will be stolen from its own existing facility, an effect known as cannibalisation). That is why the optimal solution in this case is to locate the new facility in the surroundings of the fourth most populous city of the region, where the locating chain does not have any facility yet. The area covered by the list $\mathcal{L}_{\mathcal{M}}$ is drawn in brown colour, and is located in the South-West of the fourth most populous city. As we can see in the example, the probabilistic choice rule is more prone to the concentration of facilities around the areas with more demand, whereas the multi-deterministic choice rule favours the dispersion of facilities more.

Not only the location for the new facility is different, but also its quality. When the probabilistic rule is used, the facility has to be located in Murcia, where there already exist other facilities. So, the new facility needs to have a high quality in order to capture more demand (the optimal value for the parameter α lies in the interval $[3.35, 4.72]$). However, when the multi-deterministic rule is employed, the facility is located

in an area where there are no facilities around, hence a small quality is enough to capture most of the demand of the area (the optimal value of the variable α lies in this case in the interval $[0.60, 0.68]$).

As we have seen in the case study, the assumption of a wrong customer choice rule may, depending on the location of demand points and existing facilities, provoke high losses in profit.

3.4.2 Random problems

We have completed a similar study on a set of random problems, half of them with $n = 500$ demand points and the rest with 1000 demand points for the partially-probabilistic, and $n = 100$, also $n = 1000$ for the multi-deterministic case. For the sake of brevity, here only the differences in the location, quality, and objective value have been analysed. Several settings $(i_{\max}, j_{\max}, j^1_{\max})$ have been considered (see Table 3.5), and for each of them, five problems were generated by randomly choosing the parameters of the problems uniformly within the following intervals for the partially-probabilistic and in parentheses for the multi-deterministic case, if it is different:

- $p_i, f_j \in S$,
- $w_i \in [0, 100/\sqrt{n}], ([1, 10])$
- $\alpha_j \in [0.4, 6], ([0.4, 4])$
- $G(x, \alpha) = \sum_{i=1}^{i_{\max}} \Phi_i(d_i(x)) + G_2(\alpha)$ where
  - $\Phi_i(d_i(x)) = w_i \frac{1}{(d_i(x))^{\phi_{i0}+\phi_{i1}}}$ with $\phi_{i0} = \phi_0 = 2, \phi_{i1} \in [0.5, 1.5], ([0.5, 2])$
  - $G_2(\alpha) = e^{\frac{\alpha}{\beta_0}+\beta_1} - e^{\beta_1}$ with $\beta_0 \in [7, 9], ([5, 7])\beta_1 \in [6, 6.5], ([4, 5])$
- $c \in [10, 11], ([2, 3.5])$, the parameter for $F(M(x, \alpha)) = c \cdot M(x, \alpha)$.

Those intervals were obtained by varying up and down the value of the parameters of the quasi-real problem studied in Subsection 3.4.1.

3.4.2.1 Probabilistic choice rule versus partially probabilistic choice rule

Table 3.5: Random problems: differences in the solutions obtained by the probabilistic and partially probabilistic choice rules.

Settings	$\overline{u}$	$dist_{loc}$	$dist_{qual}$	$lost(P\|PP)$	$lost(PP\|P)$	$lost_0(P\|PP)$	$lost_0(PP\|P)$
(500,6,1)	0.5	(1.5,2.9)	(0.0,0.0)	(9.5,13.2)	(2.2,5.6)	(14.6,21.7)	(4.0,8.7)
	1.0	(1.6,2.2)	(0.1,0.2)	(21.1,30.7)	(3.4,4.7)	(32.0,44.7)	(7.0,10.5)
	2.0	(4.0,7.4)	(0.1,0.2)	(25.3,31.0)	(7.0,11.6)	(42.5,58.3)	(14.2,24.0)
(500,6,3)	0.5	(2.7,5.3)	(0.1,0.4)	(7.1,12.0)	(2.6,4.0)	(20.3,35.0)	(15.5,39.0)
	1.0	(3.2,6.1)	(0.1,0.3)	(14.3,20.2)	(2.2,4.6)	(34.2,66.0)	(14.2,44.3)
	2.0	(3.5,6.9)	(0.2,0.3)	(20.0,27.8)	(2.6,4.9)	(48.5,60.0)	(15.8,47.2)
(500,12,3)	0.5	(1.3,3.7)	(0.1,0.3)	(7.3,18.3)	(0.8,2.1)	(18.3,39.2)	(4.2,14.9)
	1.0	(2.0,3.4)	(0.1,0.1)	(12.0,22.3)	(1.7,2.3)	(25.9,33.5)	(6.7,9.8)
	2.0	(1.8,4.0)	(0.5,2.0)	(18.8,34.0)	(2.6,5.1)	(39.5,50.3)	(9.5,14.4)
(500,12,6)	0.5	(1.6,3.7)	(0.0,0.0)	(5.8,8.8)	(1.0,2.2)	(16.7,26.7)	(6.4,16.7)
	1.0	(2.3,6.2)	(0.0,0.1)	(10.0,15.1)	(2.7,8.5)	(27.8,38.5)	(13.1,38.3)
	2.0	(2.6,3.8)	(0.2,0.3)	(14.6,20.3)	(2.7,4.1)	(44.6,53.9)	(14.8,28.2)
(1000,12,3)	0.5	(0.8,1.2)	(0.0,0.0)	(4.5,8.8)	(0.9,1.6)	(7.5,13.9)	(2.2,4.3)
	1.0	(0.9,1.5)	(0.0,0.0)	(6.3,8.4)	(1.2,2.7)	(11.4,16.1)	(3.4,6.5)
	2.0	(1.4,2.0)	(0.0,0.1)	(10.5,14.0)	(2.7,5.3)	(19.1,25.1)	(6.4,10.8)
(1000,12,6)	0.5	(3.8,9.0)	(0.4,1.8)	(4.5,9.2)	(1.3,3.4)	(15.0,28.1)	(7.2,19.0)
	1.0	(4.1,7.8)	(0.0,0.0)	(9.9,20.0)	(1.4,2.1)	(28.2,50.7)	(8.4,13.7)
	2.0	(3.9,7.7)	(0.1,0.2)	(11.9,24.8)	(1.8,3.6)	(37.3,66.7)	(11.6,26.2)
(1000,25,6)	0.5	(0.5,0.7)	(0.0,0.0)	(1.2,2.1)	(0.5,0.9)	(3.1,5.4)	(2.2,5.0)
	1.0	(1.1,2.1)	(0.0,0.2)	(5.1,9.2)	(1.5,3.0)	(13.6,27.8)	(6.1,16.2)
	2.0	(2.4,6.7)	(0.1,0.4)	(10.3,16.5)	(2.2,3.6)	(29.1,53.3)	(9.5,20.5)
(1000,25,12)	0.5	(3.6,8.5)	(0.4,0.8)	(1.5,2.2)	(0.6,1.1)	(12.7,29.8)	(6.7,9.4)
	1.0	(4.3,8.6)	(0.3,0.7)	(5.0,8.4)	(1.0,1.5)	(31.6,63.0)	(12.1,23.1)
	2.0	(5.8,10.3)	(0.4,1.0)	(8.0,10.3)	(1.1,2.4)	(41.7,66.3)	(13.6,29.7)

The meaning of the columns in Table 3.5 correspond to those of Tables 3.1, 3.2, and 3.3. For each setting and each threshold value $\overline{u}$, the average value of the five problems, followed by the corresponding maximum, are given.

As we can see, even for $\overline{u} = 0.5$ we have instances where the relative profit loss incurred when the probabilistic choice rule is assumed instead of the partially probabilistic choice rule is more than 18% (see the maximum value of $lost(P|PP)$ for the setting $(500, 12, 3)$). In fact, the mean is also rather high for some settings (see the results for (500,6,1)).

We can also see that there are some instances where the relative profit lost is high despite the location and quality being quite close to that of the probabilistic choice rule. See for instance the setting $(1000, 12, 3)$ when $\overline{u} = 1$: the difference in quality is negligible, and in location is rather small, but $lost(P|PP) = 6.3\%$ in average, and the maximum is 8.4%.

As in the example, the relative profit lost increases as the threshold value increases. Also, the figures of the relative profit lost due only to the new facility are higher.

3.4.2.2 The influence of the choice rule using the multi-deterministic choice rule

Results are shown in Table 3.6 in the same way as in Table 3.5.

Table 3.6: Random problems: differences in the solutions obtained by the probabilistic and multi-deterministic choice rules.

Setting	$dist_{loc}$	$dist_{qual}$	$loss(P\|MD)$	$loss(MD\|P)$	$loss(P\|MD)_0$	$loss(MD\|P)_0$
(100,2,5)	(0.49,1.20)	(0.22,0.73)	(0.6,2.0)	(0.4,1.0)	(13.0,40.6)	(8.6,26.3)
(100,2,10)	(0.13,0.62)	(0.10,0.48)	(0.0,0.2)	(0.0,0.2)	(7.5,37.0)	(2.6,11.4)
(100,3,10)	(0.13,0.43)	(0.06,0.15)	(0.1,0.4)	(0.0,0.2)	(6.0,20.7)	(1.8,6.3)
(100,3,15)	(1.83,4.91)	(1.01,2.36)	(1.1,3.6)	(1.1,3.7)	(109.8,454.9)	(33.1,120.0)
(100,4,15)	(3.97,10.00)	(1.05,3.45)	(1.3,2.4)	(1.2,3.8)	(18.7,40.4)	(67.8,301.9)
(1000,3,15)	(0.17,0.31)	(0.00,0.00)	(0.1,0.1)	(0.0,0.0)	(4.2,15.4)	(0.8,1.5)
(1000,4,30)	(0.51,1.13)	(0.55,2.76)	(0.5,1.0)	(0.4,0.8)	(16.9,33.3)	(13.8,37.7)
(1000,6,30)	(7.40,14.14)	(0.30,1.48)	(2.0,5.9)	(0.6,0.8)	(31.8,131.3)	(7.7,15.6)
(1000,5,40)	(4.23,10.29)	(0.00,0.00)	(1.5,4.6)	(1.2,4.9)	(31.0,65.1)	(28.0,53.2)
(1000,8,40)	(2.71,10.47)	(0.52,1.35)	(1.0,2.0)	(1.7,5.4)	(5.8,13.6)	(7.9,18.9)

From the results in column $dist_{loc}$, we can see that the location of the facility may vary considerably depending on the behaviour of customers assumed when solving the problem. This is also true for the quality of the facility to be located (see the results in $dist_{qual}$).

But for a decision maker it is the profit that makes a difference. As we can see, the *average* relative profit loss for the locating chain is not too big, regardless the choice rule assumed. Since the problems are generated at random, with the demand points and the existing facilities uniformly distributed over the feasible set, the chances of having clusters of points with high demand concentration and with facilities belonging to the locating chain around those clusters (as in the case study) is small. Still, notice that the relative profit loss is greater than 4.5% in at least 4 of the problems (see the maximum values of columns $loss(P|MD)$ and $loss(MD|P)$).

The differences are much clearer regarding the average relative profit loss for the new facility. In this case, it is over 5% in most of the settings, regardless the customer choice rule assumed. And in 6 of them the average loss is over 25%. Concerning the maximum values, they are greater than 25% in at least 12 settings.

3.5 Conclusions

Two extensions of the classical probabilistic choice rule have been introduced and studied. According to the first new rule, called partially probabilistic choice rule, a customer, in order to satisfy his/her demand, only patronizes those facilities for which he/she feels an attraction greater than or equal to a threshold value, and the demand is split among them proportionally to their attraction. Unlike most of the choice rules employed in literature, the threshold value implicitly implies that there may be some unmet demand. Hence, the model is suitable for inessential goods. The other new customer choice rule employed in this work, named multi-deterministic choice rule, assumes that customers split their demand among all the

firms by patronizing only one facility from each firm, the one with the highest utility, and the demand is then split among those facilities proportionally to their attraction.

The influence of the choice rule in the location of competitive facilities has been analysed. In particular, the problem of locating a single new facility in the *plane* has been considered. The corresponding location problem for profit maximization has been formulated, and a rigorous interval branch-and-bound method (iB&B) has been used to cope with the problem.

According to the computational studies, the optimal location and quality of the new facility as well as the profit obtained by the chain and by the new facility may vary considerably depending on the customer choice rule employed. Hence, the selection of the choice rule to be used in real applications should be made with care. In particular, although the assumption of the probabilistic choice rule makes the problem more computationally tractable, it should only be used when customers really patronize *all* the existing facilities because even a small threshold value may lead to a very different solution.

A modification of the partially probabilistic choice rule is required to handle essential goods, since in that case all the demand has to be served. But what if at a given demand point the attraction towards any of the facilities falls below the minimum threshold level? One possibility is that the demand at that demand point be served by the most attractive facility for that demand point (hence, all the customers at that demand point will go the same facility, the most attractive for them, similarly to the customers of the demand points for which there is only one facility with an attraction level above the threshold). This modified rule requires a deeper study, as the objective function should be reformulated.

As already stated in [177], competitive location is a difficult field not only because it involves rather complex mathematical models, but also because customer behaviour cannot easily be transcribed into neat equations. Models provide only an approximation to reality. More research on customer behaviour modelling, as we have done here, is required. In particular, the influence of the attraction function in the location of facilities deserves its own study. The extension of the single facility location model with the new choice rules to the case of the location of more than one facility [162], and to the case where competitors react by locating new facilities, too [163], should also be studied. Variable demand [161] is also an extension that should be researched in the future. From a computational point of view, the design of new algorithms that are able to solve larger instances, or the parallelization of the introduced ones, is another promising field for research.

Summary

This dissertation contains novel results in both the field of global optimization and facility location. Some results belong to either only the global optimization or the facility location field, but there are also works where both fields are involved. These classes correspond to Chapter 2, Chapter 3, and Chapter 1, respectively.

In Chapter 1 three challenging problems were solved using spatial branch-and-bound methods. Each problem is from the field of facility location, but all are quite different in the problem settings. Due to that, although all the methods are based on the branch-and-bound principle, they are different in the type of subproblems, and thus also how bounding rules can be defined.

Unsurprisingly, the branching and the bounding rules are the two most important decisions in a B&B framework. In the studied problems, the general methods could not work, hence the structure of the problem had to be taken into account. For the covering problem in Section 1.1, a novel structure for the subproblems has been designed. It was able to describe a set of subproblems where neither the discrete nor the continuous part of the decision was directly fixed, only restricted, but it was still possible to find strong bounding procedures. What was surprising, how the three different bounding rules could complement each other, making the method even more efficient and powerful.

For the firm expansion problem in Section 1.2, both the branching and bounding procedures were kind of standard, with a slight specialization due to the connections between the variables. What was really important to boost the efficiency of the method is the use of discarding tests, which were developed in this work for both continuous nonlinear constraints and integer restrictions. The use of the new monotonicity and projected one-dimensional Newton method made it possible to develop a method which is faster than any commercial solver available for academic use.

Solving the location-price equilibrium problem in Section 1.3, the first step was to transform it to a pure location problem, where prices can be determined dynamically. However, the transformed optimization problem is not differentiable anymore, thus a special rule is needed that can handle it. Using the described inclusion for the min function and its gradient, the problem could be solved up to medium-sized problems by the interval branch-and-bound method. For larger problems, an alternating Weiszfeld-like location algorithm has been designed, which was shown to find the solution efficiently.

The developed methods of the chapter expand the size and type of problems that can be solved using an exhaustive search method.

Chapter 2 showed our main results in simplicial branch-and-bound methods. We have dealt with all the important aspects of the general branch-and-bound framework.

In Section 2.1, a procedure was built for testing infeasibility in the presence of infeasibility spheres at the vertices of the simplices. This so-called Simplex Covering problem was rewritten as an optimization problem, and based on geometrical considerations using power planes some easily computable routines were derived to check infeasibility. Interestingly, the problem becomes much simpler in case of regular simplices.

In continuation, in Section 2.2, investigation on regular simplex refinement was carried out. The research question was how a regular refinement can be done, and what are the advantages and disadvantages of such branching procedure. It turned out, that despite the huge overlapping between simplices, in case of grid-aware refinement the use of regular simplices can be beneficial. Still, in larger dimensions, and when many sub-problems can be discarded, the use of longest edge bisection is more efficient.

Finally, in Section 2.3 bounding rules for simplicial partitions were investigated using either gradient information or Affine Arithmetic. In total, eight inclusion functions were designed and compared on a number of test problems in a simplicial branch-and-bound framework. The developed bounding rules were quite different in computational effort and effectiveness. Not surprisingly, those using linear programming

to obtain a bound, were not efficient enough due to the high computational cost. Affine Arithmetic based bounds were the tightest, but still, cheap centred forms give the best results in general. We saw that monotonicity test plays a big role in all cases and there is potential for even further improvement.

In Chapter 3 our results in modelling competitive facility location problems were described. We construct models where our aim is to describe costumer selection behaviour as close to reality as possible.

We start by describing the classical models in the literature in Section 3.1. Afterwards, we design the partially probabilistic selection rule in Section 3.2, that models the costumer behaviour, when costumers do not want to patronize centres having lower utility than a minimal utility given to the problem.

Our next model, in Section 3.3, describes the multi-deterministic selection rule, that alloy the deterministic and probabilistic rule. We choose deterministically from the facilities of any chain, but among the best facilities of all the chains the costumers distribute their demand probabilistically according to their utility.

Additionally, in Section 1.2, we have modelled the problem of chain expansion, where we consider not only the opportunity to open a new facility but possible adjustments for the existing facilities of the locating chain as well. The chain has some budget restrictions, but can decide to close some existing facilities but even to not open a new facility.

These models expand the state-of-the-art in competitive facility location, rendering them more realistic to solve practical problems.

Bibliography

[1] R. Aboolian, O. Berman, and D. Krass. Competitive facility location and design problem. *European Journal of Operational Research*, 182(1):40–62, 2007.

[2] G. M. Adelson-Velsky and E. M. Landis. An algorithm for the organization of information. *Proceedings of the USSR Academy of Sciences (in Russian)*, 146:263–266, 1962.

[3] B. Adenso-Díaz and F. Rodríguez. A simple search heuristic for the MCLP: Application to the location of ambulance bases in a rural region. *Omega*, 25(2):181–187, 1997.

[4] A. Andrade, J. Comba, and J. Stolfi. Affine arithmetic. *International Conf. on Interval and Computer-Algebraic Methods in Science and Engineering (INTERVAL/94)*, 1994.

[5] G. Aparicio, L. G. Casado, E. M. T. Hendrix, B. G.-Tóth, and I. García. On the minimum number of simplex shapes in longest edge bisection refinement of a regular n-simplex. *Informatica*, 26(1):17–32, 2015.

[6] B. Arenoe, J.-P. I. van der Rest, and P. Kattuman. Game theoretic pricing models in hotel revenue management: An equilibrium choice-based conjoint analysis approach. *Tourism Management*, 51(1):96–102, 2015.

[7] M. Ashtiani. Competitive location: a state-of-art review. *International Journal of Industrial Engineering Computations*, 7:1–18, 2016.

[8] F. Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[9] E. Baumann. Optimal centered forms. *BIT Numerical Mathematics*, 28(1):80–87, 1988.

[10] D. Bell, T.-H. Ho, and C. Tang. Determining where to shop: Fixed and variable costs of shopping. *Journal of Marketing Research*, 35(3), 1998.

[11] P. Belotti, T. Berthold, P. Bonami, S. Cafieri, F. Margot, C. Megaw, S. Vigerske, and A. Wachter. *COUENNE (Convex Over and Under ENvelopes for Nonlinear Estimation)*. `https://projects.coin-or.org/Couenne` [Accessed 17-03-2022].

[12] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013.

[13] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wachter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods & Software*, 24(4-5):597–634, 2009.

[14] T. Benoist, B. Estellon, F. Gardi, R. Megel, and K. Nouioua. LocalSolver 1.x: a black-box local-search solver for 0-1 programming. *4OR*, 9(3):299, 2011.

[15] O. Berman, Z. Drezner, and D. Krass. Generalized coverage: New developments in covering location models. *Computers & Operations Research*, 37(10):1675–1687, 2010.

[16] O. Berman, Z. Drezner, and D. Krass. Continuous covering and cooperative covering problems with a general decay function on networks. *Journal of the Operational Research Society*, 64(11):1644–1653, 2013.

[17] O. Berman and D. Krass. The generalized maximal covering location problem. *Computers & Operations Research*, 29(6):563–581, 2002.

[18] O. Berman, D. Krass, and Z. Drezner. The gradual covering decay location problem on a network. *European Journal of Operational Research*, 151(3):474–480, 2003.

[19] R. Blanquero and E. Carrizosa. Solving the median problem with continuous demand on a network. *Computational Optimization and Applications*, 56(3):723–734, 2013.

[20] R. Blanquero, E. Carrizosa, and B. G.-Tóth. Maximal covering location problems on networks with regional demand. *Omega*, 64:77–85, 2016.

[21] R. Blanquero, E. Carrizosa, A. Nogales-Gómez, and F. Plastria. Single-facility huff location problems on networks. *Annals of Operations Research*, 222(1):175–195, 2013.

[22] P. Bonami, L. Biegler, A. Conn, G. Cornuéjols, I. Grossmann, C. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wachter. An algorithmic framework for convex mixed integer nonlinear programs. *IBM Research Report*, RC23771 (W0511-023), 2005.

[23] P. Bonami, M. Klinç, and J. Linderoth. Algorithms and Software for Convex Mixed Integer Nonlinear Programs. In J. Lee and S. Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154 of *The IMA Volumes in Mathematics and its Applications*, pages 1–39. Springer, New York, 2012.

[24] A. Bonn, A. S. L. Rodrigues, and K. J. Gaston. Threatened and endemic species: are they good indicators of patterns of biodiversity on a national scale?: Threat, endemism and biodiversity. *Ecology Letters*, 5(6):733–741, 2002.

[25] J. Brimberg and Z. Drezner. A new heuristic for solving the p-median problem in the plane. *Computers & Operations Research*, 40:427–437, 2013.

[26] J. Brimberg, P. Hansen, N. Mladenovic, and S. Salhi. A survey of solution methods for the continuous location-allocation problem. *International Journal of the Operations Research*, 5:1–12, 2008.

[27] J. Brimberg, P. Hansen, N. Mlandinović, and E. Taillard. Improvement and comparison of heuristics for solving the uncapacitated multisource Weber problem. *Operations Research*, 48(3):444–460, 2000.

[28] S. Burer and A. Letchford. Non-convex mixed-integer nonlinear programming: a survey. *Surveys in Operations Research and Management Science*, 17(1):97–106, 2012.

[29] R. H. Byrd, J. Nocedal, and R. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. di Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, volume 83 of *Nonconvex Optimization and Its Applications*, pages 35–59. Springer, 2006.

[30] V. Cantillo, J. Amaya, and J. d. D. Ortúzar. Thresholds and indifference in stated choice surveys. *Transportation Research Part B: Methodological*, 44:753–763, 2010.

[31] V. Cantillo, B. Heydecker, and J. d. D. Ortúzar. A discrete choice model incorporating thresholds for perception in attribute values. *Transportation Research Part B: Methodological*, 40(9):807–825, 2006.

[32] V. Cantillo and J. d. D. Ortúzar. Implications of thresholds in discrete choice modelling. *Transport Reviews*, 26(6):667–691, 2006.

[33] E. Carrizosa and B. G.-Tóth. Anti-covering problems. In *Location Science*, pages 115–132. Springer, 2019.

[34] E. Carrizosa, M. Muñoz-Márquez, and J. Puerto. A note on the optimal positioning of service units. *Operations Research*, 46(1):155–156, 1998.

[35] E. Carrizosa and F. Plastria. Optimal expected-distance separating halfspace. *Mathematics of Operations Research*, 33(3):662–677, 2008.

[36] L. G. Casado, I. García, B. G.-Tóth, and E. M. T. Hendrix. On determining the cover of a simplex by spheres centered at its vertices. *Journal of Global Optimization*, 50(4):645–655, 2010.

[37] L. G. Casado, E. M. T. Hendrix, and I. García. Infeasibility spheres for finding robust solutions of blending problems with quadratic constraints. *Journal of Global Optimization*, 39(4):577–593, 2007.

[38] S.-H. Cho, H. Jang, T. Lee, and J. Turner. Simultaneous location of trauma centers and helicopters for emergency medical service planning. *Operations Research*, 62(4):751–771, 2014.

[39] C.-H. Chung. Recent applications of the maximal covering location planning (M.C.L.P.) model. *Journal of the Operational Research Society*, 37(8):735–746, 1986.

[40] R. Church and C. ReVelle. The maximal covering location problem. *Papers of the Regional Science Association*, 32(1):101–118, 1974.

[41] R. L. Church and M. E. Meadows. Location modeling utilizing maximum service distance criteria. *Geographical Analysis*, 11(4):358–373, 1979.

[42] R. Clarkon, C. Clarke-Hill, and J. Robinson. UK Supermarket Location Assessment. *International Journal of Retail & Distribution Management*, 24(6):22–33, 1996.

[43] J. Claussen and A. Zilinskas. Subdivision, sampling and initialization strategies for simplical branch and bound in global optimization. *Computers and Mathematics with Applications*, 44:943–955, 2002.

[44] R. Colomé, H. Lourenco, and D. Serra. A new chance-constrained maximum capture location problem. *Annals of Operations Research*, 122(1-4):121–139, 2003.

[45] R. Colomé and D. Serra. Consumer choice and optimal location models: Formulations and heuristics. *Papers in Regional Science*, 80(4):439–464, 2001.

[46] L. Cooper. Heuristics methods for location-allocation problems. *SIAM Review*, 6:37–53, 1964.

[47] Á. Corberán. Data instances for arc routing problems. Accessed: 06-04-2022.

[48] C. Craig, A. Ghosh, and S. McLafferty. Models of the retail location process: a review. *Journal of Retailing*, 60(1):5–36, 1984.

[49] J. R. Current and D. A. Schilling. Analysis of errors due to demand data aggregation in the set covering and maximal covering location problems. *Geographical Analysis*, 22(2):116–126, 1990.

[50] A. Dasci and G. Laporte. A continuous model for multistore competitive location. *Operations Research*, 53(2):263–280, 2005.

[51] M. S. Daskin and L. K. Dean. Location of health care facilities. In M. Brandeau, F. Sainfort, and W. Pierskalla, editors, *Operations Research and Health Care*, volume 70 of *International Series in Operations Research & Management Science*, pages 43–76. Springer, Boston, MA, 2005.

[52] C. D'Aspremont, J. Gabszewicz, and J. Thisse. On Hotelling's stability in competition. *Econometrica*, 47:1145–1150, 1979.

[53] L. de Figueiredo and J. Stolfi. Affine arithmetic: Concepts and applications. *Numerical Algorithms*, 37(1-4):147–158, 2004.

[54] E. de Klerk. The complexity of optimizing over a simplex, hypercube or sphere: a short survey. *Central European Journal of Operations Research*, 16(2):111–125, 2008.

[55] P. J. Dickinson. On the exhaustivity of simplicial partitioning. *Journal of Global Optimization*, 58(1):189–203, 2014.

[56] G. Dobson and U. S. Karmarkar. Competitive location on a network. *Operations Research*, 35(4):565–574, 1987.

[57] P. Dorta-González, D. Santos-Peñate, and R. Suárez-Vega. Spatial competition in networks under delivered pricing. *Papers in Regional Science*, 84:271–280, 2005.

[58] T. Drezner. Optimal continuous location of a retail facility, facility attractiveness, and market share: An interactive model. *Journal of Retailing*, 70(1):49–64, 1994.

[59] T. Drezner. Competitive facility location in the plane. In Z. Drezner, editor, *Facility location: a survey of applications and methods*, Springer Series in Operations Research and Financial Engineering, pages 285–300. Springer, Berlin, 1995.

[60] T. Drezner. Location of multiple retail facilities with limited budget constraints in continuous space. *Journal of Retailing and Consumer Services*, 5(3):173–184, 1998.

[61] T. Drezner. A review of competitive facility location in the plane. *Logistics Research*, 7:114, 2014.

[62] T. Drezner and Z. Drezner. Lost demand in a competitive environment. *Journal of the Operational Research Society*, 59:362–371, 2008.

[63] T. Drezner and Z. Drezner. Modelling lost demand in competitive facility location. *Journal of the Operational Research Society*, 63:201–206, 2012.

[64] T. Drezner, Z. Drezner, and P. Kalczynski. Strategic competitive location: improving existing and establishing new facilities. *Journal of the Operational Research Society*, 63(12):1720–1730, 2012.

[65] T. Drezner and H. Eiselt. *Facility location: applications and theory*, chapter Consumers in competitive location models, pages 151–178. Springer-Verlag, Berlin, 2002.

[66] Z. Drezner. Competitive location strategies for two facilities. *Regional Science and Urban Economics*, 12(4):485–493, 1982.

[67] Z. Drezner and A. Suzuki. Covering continuous demand in the plane. *Journal of the Operational Research Society*, 61(5):878–881, 2010.

[68] H. Edelsbrunner. The union of balls and its dual shape. *Disctrete and Computational Geometry*, 13(1):415–440, 1995.

[69] H. Eiselt. Hotelling's duopoly on a tree. *Annals of Operations Research*, 40:195–207, 1992.

[70] H. Eiselt and G. Laporte. Sequential location problems. *European Journal of Operational Research*, 96(2):217–231, 1996.

[71] H. Eiselt, G. Laporte, and J. Thisse. Competitive location models: a framework and bibliography. *Transportation Science*, 27(1):44–54, 1993.

[72] H. Eiselt, V. Marianov, and T. Drezner. Competitive location models. In G. Laporte, S. Nickel, and F. Saldanha-da Gama, editors, *Location Science*, pages 365–398. Springer, 2015.

[73] J. Fernández, P. Fernández, and B. Pelegrín. Estimating actual distances by norm functions: a comparison between the $l_{k,p,\theta}$-norm and the $l_{b_1,b_2,\theta}$-norm and a study about the selection of the data set. *Computers & Operations Research*, 29(6):609–623, 2002.

[74] J. Fernández and E. Hendrix. Recent insights in Huff-like competitive facility location and design. *European Journal of Operational Research*, 227(3):581–584, 2013.

[75] J. Fernández and B. Pelegrín. Using interval analysis for solving planar single-facility location problems: new discarding tests. *Journal of Global Optimization*, 19(1):61–81, 2001.

[76] J. Fernández, B. Pelegrín, F. Plastria, and B. Tóth. Solving a Huff-like competitive location and design model for profit maximization in the plane. *European Journal of Operational Research*, 179(3):1274–1287, 2007.

[77] J. Fernández, S. Salhi, and B. G. Tóth. Location equilibria for a continuous competitive facility location problem under delivered pricing. *Computers & Operations Research*, 41(1):185–195, 2014.

[78] J. Fernández, B. Tóth, L. Cánovas, and B. Pelegrín. A practical algorithm for decomposing polygonal domains into convex polygons by diagonals. *Top*, 16(2):367–387, 2008.

[79] J. Fernández, B. G. Tóth, J. Redondo, P. Ortigosa, and A. Arrondo. A planar single-facility competitive location and design problem under the multi-deterministic choice rule. *Computers & Operations Research*, 78:305–315, 2017.

[80] J. Fernández, B. G.- Tóth, J. L. Redondo, and P. M. Ortigosa. The probabilistic customer's choice rule with a threshold attraction value: Effect on the location of competitive facilities in the plane. *Computers & Operations Research*, 101:234–249, 2019.

[81] R. Fourer, D. Gay, and B. Kernighan. *AMPL. A Modeling Language for Mathematical Programming (second edition)*. Duxbury-Thomson, 2003.

[82] B. G.-Tóth, L. Anton-Sanchez, and J. Fernández. A combined model for chain expansion including the possibility of locating a new facility and modification and/or closing of existing facilities. Optimization Online 8970, 2022.

[83] B. G.-Tóth, L. Anton-Sanchez, J. Fernández, J. Redondo, and P. Ortigosa. A Continuous Competitive Facility Location and Design Problem for Firm Expansion. In H. Le Thi, H. Le, and T. Pham, editors, *Optimization of Complex Systems: Theory, Models, Algorithms and Applications*, Advances in Intelligent Systems and Computing 991, pages 1013–1022. Springer, 2020.

[84] B. G.-Tóth, L. Casado, E. Hendrix, and F. Messine. On new methods to construct lower bounds in simplicial branch and bound based on interval arithmetic. *Journal of Global Optimization*, 80(4):779–804, 2021.

[85] B. G.-Tóth, E. M. T. Hendrix, L. G. Casado, and I. García. On refinement of the unit simplex using regular simplices. *Journal of Global Optimization*, 64(2):305–323, 2015.

[86] J. Gabszewicz and J. Thisse. Location. In R. Aumann and S. Hart, editors, *Handbook of Game Theory with Economic Applications*, pages 281–304. Elsevier Science Publishers, 1992.

[87] R. D. Galvão, L. G. A. Espejo, and B. Boffey. A comparison of Lagrangean and surrogate relaxations for the maximal covering location problem. *European Journal of Operational Research*, 124(2):377–389, 2000.

[88] R. D. Galvão and C. ReVelle. A Lagrangean heuristic for the maximal covering location problem. *European Journal of Operational Research*, 88(1):114–123, 1996.

[89] M. García, P. Fernández, and B. Pelegrín. On price competition in location-price models with spatially separated markets. *Top*, 12(2):351–374, 2004.

[90] M. García and B. Pelegrín. All Stackelberg location in the Hotelling's duopoly model on a tree with parametric prices. *Annals of Operations Research*, 122:177–192, 2003.

[91] M. Gavrilova and J. Rokne. An efficient algorithm for construction of the power diagram from the voronoi diagram in the plane. *International Journal of Computer Mathematics*, 61:49–61, 1997.

[92] A. Ghosh and C. Craig. Formulating retail location strategy in a changing environment. *Journal of Marketing*, 47(3):56–68, 1983.

[93] A. Ghosh and C. Craig. FRANSYS: a franchise distribution system location model. *Journal of Retailing*, 67:466–495, 1991.

[94] T. J. Gilbride and G. M. Allenby. A choice model with conjunctive, disjunctive, and compensatory screening rules. *Marketing Science*, 23(3):391–406, 2004.

[95] P. Gupta. Competitive spatial price discrimination with strictly convex production costs. *Regional Science and Urban Economics*, 24:265–272, 1994.

[96] S. Hakimi. On locating new facilities in a competitive environment. *European Journal of Operational Research*, 12(1):29–35, 1983.

[97] S. Hakimi. Locations with spatial interactions: competitive location and games. In R. Francis and P. Mirchandani, editors, *Discrete Location Theory*, pages 439–478. Wiley/Interscience, New York, 1990.

[98] J. Hamilton, J. Thisse, and A. Weskamp. Spatial discrimination, Bertrand vs. Cournot in a model of location choice. *Regional Science and Urban Economics*, 19:87–102, 1989.

[99] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz. *C++ Toolbox for verified computing I: basic numerical problems: theory, algorithms and programs*. Springer-Verlag, Berlin, 1995.

[100] A. Hannukainen, S. Korotov, and M. Křížek. On numerical regularity of the face-to-face longest-edge bisection algorithm for tetrahedral partitions. *Science of Computer Programming*, 90, Part A(0):34–41, 2014.

[101] E. Hansen and W. Walster. *Global Optimization Using Interval Analysis*. Marcel Dekker Inc., New York, Second edition, 2004.

[102] P. Hansen and J. Thisse. The generalized Weber-Rawls problem. In *Operational research '81 (Hamburg, 1981)*, pages 569–577. North-Holland, Amsterdam, 1981.

[103] E. Hendrix, L. G. Casado, and P. Amaral. Global optimization simplex bisection revisited based on considerations by Reiner Horst. In *Computational Science and Its Applications – ICCSA 2012*, volume 7335 of *Lecture Notes in Computer Science*, pages 159–173. Springer Berlin Heidelberg, 2012.

[104] E. Hendrix, B. Tóth, F. Messine, and L. Casado. On derivative based bounding for simplicial branch and bound. *RAIRO - Operations Research*, 55(3):2023–2034, 2021.

[105] E. M. T. Hendrix, L. G. Casado, and I. Garcia. The semi-continuous quadratic mixture design problem: Description and branch-and-bound approach. *European Journal of Operational Research*, 191(3):803–815, 2008.

[106] E. M. T. Hendrix, J. M. G. Salmerón, and L. G. Casado. On function monotonicity in simplicial branch and bound. *AIP Conference Proceedings*, 2070:020007–1,020007–4, 2019.

[107] M. Hodgson. A location-allocation model maximizing consumers' welfare. *Regional Studies*, 15(6):493–506, 1981.

[108] J. Holmes, F. Williams, and L. Brown. Facility location under a maximum travel restriction: an example using day care facilities. *Geographical Analysis*, 4(3):258–266, 1972.

[109] E. Hoover. Spatial price discrimination. *Review of Economic Studies*, 4:182–191, 1936.

[110] R. Horst. On generalized bisection of n-simplices. *Mathematics of Computation*, 66(218):691–698, 1997.

[111] R. Horst. Bisecton by global optimization revisited. *Journal of optimization theory and applications*, 144(3):501–510, 2010.

[112] H. Hotelling. Stability in competition. *Economic Journal*, 39:41–57, 1929.

[113] D. Huff. Defining and estimating a trading area. *Journal of Marketing*, 28(3):34–38, 1964.

[114] D. Huff. A programmed solution for approximating an optimum retail location. *Land Economics*, 42(3):293–303, 1966.

[115] A. Jain and V. Mahajan. Evaluating the competitive environment in retailing using multiplicative competitive interactive models. *Research in Marketing*, 1:217–235, 1979.

[116] S. D. Karhbet and R. B. Kearfott. Range bounds of functions over simplices, for branch and bound algorithms. *Reliable Computing*, 25:53–73, 2017.

[117] R. Kearfott. *Rigourous Global Search: Continuous Problems*. Kluwer, Dordrecht, 1996.

[118] R. Kearfott, M. Nakao, A. Neumaier, S. M. Rump, S. P. Shary, and P. van Hentenryck. Standardized notation in interval analysis. *TOM*, 15(1):7–13, 2010.

[119] O. Knüppel. PROFIL/BIAS - A Fast Interval Library. *Computing*, 53(1):277–287, 1993.

[120] S. Korotov and M. Křížek. Red refinements of simplices into congruent subsimplices. *Computers & Mathematics with Applications*, 67(12):2199–2204, 2014.

[121] D. Kress and E. Pesch. Sequential competitive location on networks. *European Journal of Operational Research*, 217(3):483–499, 2012.

[122] H. Küçükaydin, N. Aras, and I. K. Altinel. A leader-follower game in competitive facility location. *Computers & Operations Research*, 39(2):437–448, 2012.

[123] Languages for interval analysis. Available at `https://www.cs.utep.edu/interval-comp/intlang.html`, [Accessed 17-03-2022].

[124] L. J. LeBlanc, E. K. Morlok, and W. P. Pierskalla. An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research*, 9(5):309–318, 1975.

[125] P. Lederer and A. Hurter. Competition of firms: discriminatory pricing and location. *Econometrica*, 54(3):623–40, 1986.

[126] P. Lederer and J. Thisse. Competitive location on networks under delivered pricing. *Operations Research Letters*, 9(91):147–153, 1990.

[127] S. Leyffer, J. Linderoth, J. Luedtke, A. Mahajan, T. Munson, P. Palkar, and M. Sharma. MINOTAUR: Mixed-Integer Optimization Toolbox-Algorithms, Underestimators, Relaxations. `https://wiki.mcs.anl.gov/minotaur/index.php/MINOTAUR` [Accessed 17-03-2022].

[128] X. Li, Z. Zhao, X. Zhu, and T. Wyatt. Covering models and optimization techniques for emergency response facility location and planning: a review. *Mathematical Methods of Operations Research*, 74(3):281–310, 2011.

[129] V. Marianov and C. ReVelle. The queueing maximal availability location problem: A model for the siting of emergency vehicles. *European Journal of Operational Research*, 93(1):110–120, 1996.

[130] J. Martínez, L. Casado, I. García, and B. Tóth. AMIGO: Advanced Multidimensional Interval analysis Global Optimization algorithm. In *Frontiers in Global Optimization*, volume 74 of *Nonconvex Optimization and Its Applications*, pages 313–326. Kluwer, Dordrecht, 2004.

[131] T. C. Matisziw and A. T. Murray. Siting a facility in continuous space to maximize coverage of a region. *Socio-Economic Planning Sciences*, 43(2):131–139, 2009.

[132] M. Megiddo and K. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13:182–196, 1984.

[133] N. Megiddo, E. Zemel, and S. L. Hakimi. The maximum coverage location problem. *SIAM Journal on Algebraic and Discrete Methods*, 4(2):253–261, 1983.

[134] F. Messine. Extensions of affine arithmetic: application to unconstrained global optimization. *Journal of Universal Computer Science*, 8(11):992–1015, 2002.

[135] F. Messine and J.-L. Lagouanelle. Enclosure methods for multivariate differentiable functions and application to global optimization. *Journal of Universal Computer Science*, 4(6):589–603, 1998.

[136] F. Messine and A. Touhami. A general reliable quadratic form: An extension of affine arithmetic. *Reliable Computing*, 12(3):171–192, 2006.

[137] H. J. Miller. GIS and geometric representation in facility location problems. *International Journal of Geographical Information Science*, 10(7):791–816, 1996.

[138] R. Misener and C. Floudas. ANTIGONE: Algorithms fon coNTinuous / Integer Global Optimization of Nonlinear Equations. *Journal of Global Optimization*, 59(2):503–526, 2014.

[139] O. Mohand. Tighter bound functions for nonconvex functions over simplexes. *RAIRO Operations Research*, 55:S2373–S2381, 2021.

[140] R. Moore. *Interval Analysis*. Prentice-Hall Inc., Englewood Cliffs, 1966.

[141] A. T. Murray and R. L. Church. Applying simulated annealing to location-planning models. *Journal of Heuristics*, 2(1):31–53, 1996.

[142] A. T. Murray, M. E. O'Kelly, and R. L. Church. Regional service coverage modeling. *Computers & Operations Research*, 35(2):339–355, 2008.

[143] M. Nakanishi and L. Cooper. Parameter estimate for multiplicative interactive choice model: Least square approach. *Journal of Marketing Research*, 11:303–311, 1974.

[144] J. Ninin, F. Messine, and P. Hansen. A reliable affine relaxation method for global optimization. *4OR*, 13(3):247–277, 2014.

[145] M. Osborne and C. Pitchik. Equilibrium in Hotelling's model of spatial competition. *Econometrica*, 55:911–922, 1987.

[146] R. Paulavičius and J. Žilinskas. *Simplicial Global Optimization*. Springer, New York, 2014.

[147] P. Peeters and F. Plastria. Discretization results for the Huff and Pareto-Huff competitive location models on networks. *Top*, 6:247–260, 1992.

[148] B. Pelegrín, P. Dorta-González, and P. Fernández. Finding location equilibria for competing firms under delivered pricing. *Journal of the Operational Research Society*, 62:729–741, 2010.

[149] B. Pelegrín, J. Fernández, and B. Tóth. The 1-center problem in the plane with independent random weights. *Computers & Operations Research*, 35(3):737–749, 2008.

[150] J. D. Pintér. LGO — a program system for continuous and Lipschitz global optimization. In *Developments in global optimization*, pages 183–197. Springer, 1997.

[151] H. Pirkul and D. A. Schilling. The maximal covering location problem with capacities on total workload. *Management Science*, 37(2):233–248, 1991.

[152] F. Plastria. Static competitive facility location: an overview of optimisation approaches. *European Journal of Operational Research*, 129(3):461–470, 2001.

[153] F. Plastria. Continuous covering location problems. In Z. Drezner and H. W. Hamacher, editors, *Facility Location: Applications and Theory*, pages 37–79. Springer Berlin, Heidelberg, 2002.

[154] F. Plastria and E. Carrizosa. Undesirable facility location with minimal covering objectives. *European Journal of Operational Research*, 119(1):158–180, 1999.

[155] F. Plastria and E. Carrizosa. Optimal location and design of a competitive facility. *Mathematical Programming*, 100(2):247–265, 2004.

[156] F. Plastria and L. Vanhaverbeke. Discrete models for competitive location with foresight. *Computers & Operations Research*, 35(3):683–700, 2008.

[157] L. Rall. *Automatic Differentiation, Techniques and Applications*, volume 120 of *Lecture Notes in Computer Science*. Springer, Berlin, 1981.

[158] H. Ratschek and J. Rokne. *Computer Methods for the Range of Functions*. Ellis Horwood Ltd, Chichester, 1984.

[159] H. Ratschek and J. Rokne. *New Computer Methods for Global Optimization*. Ellis Horwood, Chichester, 1988.

[160] J. Redondo, J. Fernández, J. Álvarez, A. Arrondo, and P. Ortigosa. Approximating the Pareto-front of a planar bi-objective competitive facility location and design problem. *Computers & Operations Research*, 62:337–349, 2015.

[161] J. Redondo, J. Fernández, A. Arrondo, I. García, and P. Ortigosa. Fixed or variable demand? Does it matter when locating a facility? *Omega*, 40(1):9–20, 2012.

[162] J. Redondo, J. Fernández, I. García, and P. Ortigosa. Solving the multiple competitive facilities location and design problem on the plane. *Evolutionary Computation*, 17(1):21–53, 2009.

[163] J. Redondo, J. Fernández, I. García, and P. M. Ortigosa. Heuristics for the facility location and design (1|1)-centroid problem on the plane. *Computational Optimization and Applications*, 45(1):111–141, 2010.

[164] W. Reilly. *The Law of Retail Gravitation*. Pilsbury Publishers, New York, 2nd edition, 1953.

[165] M. G. Resende. Computing approximate solutions of the maximum covering problem with GRASP. *Journal of Heuristics*, 4(2), 1998.

[166] C. ReVelle. Facility siting and integer-friendly programming. *European Journal of Operational Research*, 65(2):147–158, 1993.

[167] C. ReVelle and H. Eiselt. Location analysis: A synthesis and survey. *European Journal of Operational Research*, 165(1):1–19, 2005.

[168] C. ReVelle, M. Scholssberg, and J. Williams. Solving the maximal covering location problem with heuristic concentration. *Computers & Operations Research*, 35(2):427–435, 2008.

[169] S. M. Rump and M. Kashiwagi. Implementation and improvements of affine arithmetic. *Nonlinear Theory and Its Applications*, 6(3):341–359, 2015.

[170] B. Sadri. *Surface and medial axis topology through distance flows induced by discrete samples*. PhD thesis, University of Illinois, Urbana, Illinois, 2006.

[171] N. V. Sahinidis. *BARON 17.8.9: Global Optimization of Mixed-Integer Nonlinear Programs,* User's Manual, 2017. `http://www.minlp.com/downloads/docs/baron%20manual.pdf` [Accessed 17-03-2022].

[172] N. Saidani, F. Chu, and H. Chen. Competitive facility location and design with reactions of competitors already in the market. *European Journal of Operational Research*, 219(1):9–17, 2012.

[173] J. M. G. Salmerón, G. Aparicio, L. G. Casado, I. García, E. M. T. Hendrix, and B. G.-Tóth. Generating a smallest binary tree by proper selection of the longest edges to bisect in a unit simplex refinement. *Journal of Combinatorial Optimization*, 33(2):389–402, 2015.

[174] D. Schilling, V. Jayaraman, and R. Barkhi. A review of covering problem in facility location. *Location Science*, 1(1):25–55, 1993.

[175] A. Schöbel and D. Scholz. The big cube small cube solution method for multidimensional facility location problems. *Computers & Operations Research*, 37:115–122, 2010.

[176] D. Serra, H. Eiselt, G. Laporte, and C. ReVelle. Market capture models under various customer choice rules. *Environment and Planning B*, 26(5):141–150, 1999.

[177] D. Serra and C. ReVelle. Competitive location and pricing on networks. *Geographical Analysis*, 31(2):109–129, 1999.

[178] H. Sherali and W. Adams. *A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems*. Kluwer, Dordrecht, 1999.

[179] H. Sherali and L. Liberti. Reformulation-Linearization Technique for Global Optimization. In *Encyclopedia of Optimization*, volume 18, pages 3263–3268. Springer-Verlag, 2009.

[180] J. Stolfi and L. de Figueiredo. *Self-Validated Numerical Methods and Applications*. Monograph for 21st Brazilian Mathematics Colloquium. IMPA/CNPq, 1997.

[181] R. Suárez-Vega, D. Santos-Peñate, and P. Dorta-González. Competitive multifacility location on networks: the $(r|X_p)$-medianoid problem. *Journal of Regional Science*, 44(3):569–588, 2004.

[182] R. Suárez-Vega, D. Santos-Peñate, and P. Dorta-González. Discretization and resolution of the $(r|X_p)$-medianoid problem involving quality criteria. *Top*, 12(1):111–133, 2004.

[183] R. Suárez-Vega, D. Santos-Peñate, and P. Dorta-González. The follower location problem with attraction thresholds. *Papers in Regional Science*, 86(1):123–137, 2007.

[184] S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets, 2013. `http://www.sfu.ca/~ssurjano` [Accessed 17-03-2022].

[185] J. Thill. Network competition and branch differentiation with consumer heterogeneity. *The Annals of Regional Science*, 34(3):451–468, 2000.

[186] M. J. Todd. *The Computation of Fixed Points and Applications*. Springer, Heidelberg, 1976.

[187] B. Tóth and L. Casado. Multi-dimensional pruning from the Baumann point in an Interval Global Optimization Algorithm. *Journal of Global Optimization*, 38(2):215–236, 2007.

[188] B. Tóth and J. Fernández. *Interval Methods for Single and Bi-objective Optimization Problems - applied to Competitive Facility Location Problems*. Lambert Academic Publishing, Saarbrücken, 2010.

[189] B. Tóth, J. Fernández, and T. Csendes. Empirical convergence speed of inclusion functions for facility location problems. *Journal of Computational and Applied Mathematics*, 199:384–389, 2007.

[190] B. Tóth, F. Plastria, J. Fernández, and B. Pelegrín. On the impact of spatial pattern, aggregation, and model parameters in planar Huff-like competitive location and design problems. *OR Spectrum*, 31(1):601–627, 2009.

[191] F. Trespalacios and I. Grossmann. Review of mixed-integer nonlinear and generalized disjunctive programming methods. *Chemie Ingenieur Technik*, 86(7):991–1012, 2014.

[192] S. Vigerske and A. Gleixner. SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. *Optimization Methods and Software*, 33(3):563–593, 2018.

[193] Visual G.I.S. Engineering. Basauri 17 (La Florida) 28023 Madrid, 2003.

[194] E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal*, 43:355–386, 1937.

[195] T. Westerlund and R. Pörn. Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. *Optimization and Engineering*, 3(3):253–280, 2002.

[196] P. D. Wright, M. J. Liberatore, and R. L. Nydick. A survey of operations research models and applications in homeland security. *INFORMS Journal on Applied Analytics*, 36(6):514–529, 2006.

[197] H. Yapicioglu, A. Smith, and G. Dozier. Solving the semi-desirable facility location problem using bi-objective particle swarm. *European Journal of Operational Research*, 177(2):733–749, 2007.

[198] S. Zhang. On a profit maximizing location mode. *Annals of Operations Research*, 103:251–260, 2001.

[199] T. Zhang and Y. Xia. Covering a simplex by spheres: complexity and algorithms. *Journal of Global Optimization*, 2022.

[200] J. Žilinskas. Branch and bound with simplicial partitions for global optimization. *Mathematical Modelling and Analysis*, 13(1):145–159, 2008.