OR Spectrum (2009) 31:679–705
DOI 10.1007/s00291-008-0133-8



REGULAR ARTICLE

# On a branch-and-bound approach for a Huff-like Stackelberg location problem

**M. Elena Sáiz · Eligius M. T. Hendrix · José Fernández · Blas Pelegrín**

Published online: 12 March 2008


**Abstract** Modelling the location decision of two competing firms that intend to build a new facility in a planar market can be done by a Huff-like Stackelberg location problem. In a Huff-like model, the market share captured by a firm is given by a gravity model determined by distance calculations to facilities. In a Stackelberg model, the *leader* is the firm that locates first and takes into account the actions of the competing chain (*follower*) locating a new facility after the leader. The follower problem is known to be a hard global optimisation problem. The leader problem is even harder, since the leader has to decide on location given the optimal action of the follower. So far, in literature only heuristic approaches have been tested to solve the leader problem. Our research question is to solve the leader problem rigorously in the sense of having a guarantee on the reached accuracy. To answer this question, we develop a branch-and-bound approach. Essentially, the bounding is based on the zero

This work has been supported by the Ministry of Education and Science of Spain through grant SEJ2005/06273/ECON. M. Elena Sáz was supported by a junior research grant of Mansholt Graduate School (Wageningen Universiteit).

M. Elena Sáiz (✉)
Radboud Universiteit Nijmegen, Thomas van Aquinostraat 3.01.04, P.O.Box 9108, 6500 HK, Nijmegen, The Netherlands
e-mail: E.Saiz@fm.ru.nl

E. M. T. Hendrix
Wageningen Universiteit, Hollandseweg 1, 6706 KN Wageningen, The Netherlands
e-mail: Eligius.Hendrix@wur.nl

J. Fernández · B. Pelegrín
Universidad de Murcia, Campus Universitario de Espinardo, 30071 Espinardo, Murcia, Spain
e-mail: josefdez@um.es

B. Pelegrín
e-mail: pelegrin@um.es

sum concept: what is gain for one chain is loss for the other. We also discuss several ways of creating bounds for the underlying (follower) sub-problems, and show their performance for numerical cases.

**Keywords** Continuous location · Nonlinear programming · Global optimisation algorithms · Stackelberg competitive location

## 1 Introduction

Many factors must be taken into account when locating a new facility which provides goods or a service to the customers of a given area. One of the most important points is the existence of competitors in the market providing the same goods or service. When no other competitor exists, the facility to be located will have the monopoly of the market in that area. However, if in the area there already exist other facilities offering the same goods, then the new facility will have to compete for the market.

Many competitive location models are available in the literature, see for instance the survey papers Eiselt and Laporte (1996); Eiselt et al. (1993); Plastria (2001) 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. In this case it is necessary to make decisions with foresight about this competition, leading to a Stackelberg-type model (competition model in which a leader firm moves first and then the follower firm moves sequentially). Demand is usually supposed to be concentrated in a discrete set of points, called *demand points*.

The *patronising behaviour* of the customers must also be taken into account, since the *market captured* by the facilities depends on it. In some models customers select among the facilities in a *deterministic* way, i.e., the full demand of the customer is served by the facility to which he/she is *attracted* most. In other cases, the customer splits his/her demand among more than one facility, leading to *probabilistic* patronising behaviour. On the other hand, 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*.

In this paper, we consider a planar facility location problem with foresight, having a so-called Huff-like probabilistic consumer behaviour, based on an attraction function depending on both the locations and the qualities of the facilities to be located. The demand quantities are assumed to be known and fixed. For the current study, also the quality values of the new facilities to be located are assumed to be given. There are two competitors (chains) that act as in a so-called Stakelberg model. First, the leader makes a decision on where to locate its facility in the plane (the location of the facility is considered the variable of the problem). Second, the follower makes a decision with full knowledge of the decision of the leader. The objective of the leader is to maximise its market share after the entrance of the follower.

The follower problem has been studied in Drezner (1994) and Plastria (1997) under deterministic customer behaviour, using attraction functions of gravity type, and in Plastria and Carrizosa (2004) using different kinds of attraction functions. For probabilistic customer behaviour, the problem has been studied in Drezner and Drezner (1994), where the location problem is solved for a wide range of quality values (see also Drezner and Drezner 2004).

However, due to its difficulty, the literature on the leader problem is rather scarce. To our knowledge, the leader problem with deterministic behaviour on the plane has only been addressed in Drezner (1982) and Bhadury et al. (2003), and with probabilistic behaviour only in Drezner and Drezner (1998), where three heuristics are described for a variant of the model considered in this paper. The question addressed in this paper is whether the leader problem can be solved up to a guaranteed accuracy. We will show in the current paper that one can make use of the zero-sum perspective to construct a branch-and-bound method that achieves that aim.

In Sect. 2, the notation is introduced and both the leader and the follower problem are formulated. In Sects. 3 and 4, a detailed description of the branch-and-bound algorithms to solve the follower and leader problem (respectively) is given. The algorithms are illustrated by instances in Sect. 5 and the efficiency is investigated for different parameter settings. Conclusions and future work are discussed in Sect. 6.

## 2 Description of the problem

The following notation will be used throughout:

*Indices*

- $i$ index of demand points, $i = 1, \ldots, n$
- $j$ index of existing facilities, $j = 1, \ldots, m$ (the first $k$ of those $m$ facilities, $0 \leq k \leq m$, belong to the leader chain, and the rest to the follower)
- $l$ index for the new facilities, $l = 1, 2$.

*Variables*

- $x_l = (x_{l1}, x_{l2})$ location of the leader ($l = 1$) and follower ($l = 2$)

*Data*

- $\alpha_l$ quality of the leader ($l = 1$) and follower ($l = 2$)
- $p_i$ location of the $i$-th demand point
- $w_i$ demand (or buying power) at $p_i$
- $q_j$ location of the $j$-th existing facility
- $d_{ij}$ distance between $p_i$ and $q_j$
- $a_j$ quality of facility $j$
- $g(\cdot)$ a positive non-decreasing function
- $a_j/g(d_{ij})$ attraction that $i$ feels for facility $j$
- $S$ location space where the leader and the follower will locate the new facility.

*Miscellaneous*

- $\delta_{il}$ distance between $p_i$ and $x_l$, $l = 1, 2$
- $\alpha_l/g(\delta_{il})$ attraction that $i$ feels for new facility $l$
- $M_l(x_1, x_2)$ market capture by the leader ($l = 1$) and follower ($l = 2$).

The best location in attraction models is usually situated in the convex hull of the demand points. In this paper we consider as the feasible location space $S$ a rectangle enclosing that convex hull. Notice that $M_1(x_1, x_2) + M_2(x_1, x_2) = \sum_{i=1}^{n} w_i$. This 'zero-sum' character of the model is essential in the method used to solve it. In the model, the market share captured by the leader chain after the leader locates in $x_1$ and the follower in $x_2$ is

$$M_1(x_1, x_2) = \sum_{i=1}^{n} \omega_i \frac{\frac{\alpha_1}{g(\delta_{i1})} + \sum_{j=1}^{k} \frac{a_j}{g(d_{ij})}}{\frac{\alpha_1}{g(\delta_{i1})} + \frac{\alpha_2}{g(\delta_{i2})} + \sum_{j=1}^{m} \frac{a_j}{g(d_{ij})}}$$

and the corresponding market share captured by the follower chain is

$$M_2(x_1, x_2) = \sum_{i=1}^{n} \omega_i \frac{\frac{\alpha_2}{g(\delta_{i2})} + \sum_{j=k+1}^{m} \frac{a_j}{g(d_{ij})}}{\frac{\alpha_1}{g(\delta_{i1})} + \frac{\alpha_2}{g(\delta_{i2})} + \sum_{j=1}^{m} \frac{a_j}{g(d_{ij})}} \tag{1}$$

Given $x_1$, problem $(FP(x_1))$ of the follower is the so-called $(1|x_1)$-medianoid problem introduced by Hakimi (1983)

$$\max_{x_2 \in S} \{G(x_2) = M_2(x_1, x_2)\} \tag{2}$$

Since $M_1(x_1, x_2) + M_2(x_1, x_2) = \sum_{i=1}^{n} w_i$, $(FP(x_1))$ in (2) is equivalent to

$$\min_{x_2 \in S} M_1(x_1, x_2) \tag{3}$$

Let $x_2^*(x_1)$ represent an optimal solution of $(FP(x_1))$. Problem $(LP)$ for the leader is the $(1|1)$-centroid problem (see Hakimi 1983)

$$\max_{x_1 \in S} \{F(x_1) = M_1(x_1, x_2^*(x_1))\}$$

In Drezner and Drezner (2004) and Fernández et al. (2007), procedures are given to maximise the market share captured by a given chain when the facility locations of the competitors are fixed as in problem $(FP(x_1))$. As studied by Fernández et al. (2007), we are dealing with a global optimisation problem; see Fig. 1, which shows the multimodal behaviour of problem $(FP(x_1))$.

In the solution procedure that we have designed to cope with the leader problem, we are also interested in solving a similar problem to that of the follower, in which the leader wants to locate a new facility at $x_1$, given the location and the quality of all the facilities of the competitor (the follower). In this case, the leader has to solve a medianoid problem in which the roles of leader and follower are interchanged. We will call this problem a *reverse medianoid* problem.

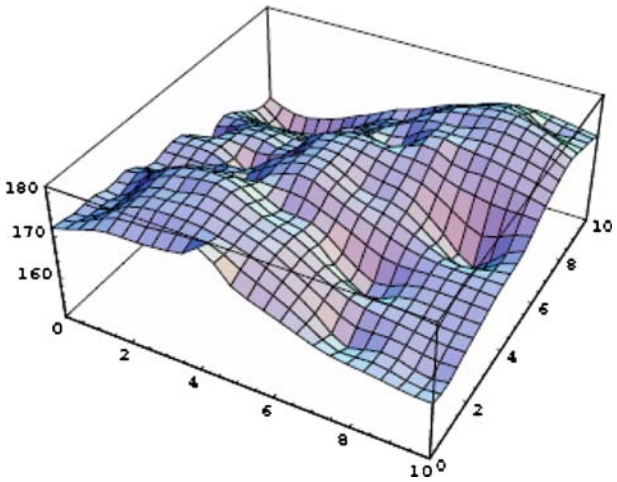


**Fig. 1** Plot of the objective function of a follower problem

The leader problem ($LP$) is much more difficult to solve than the follower problem. To the extent of our knowledge, the leader problem with probabilistic behaviour on the plane has only been addressed in Drezner and Drezner (1998), where heuristic procedures were presented for a similar version of the problem considered here. Among others, they applied variants of multistart and grid search to generate solutions of the leader and follower problems. In Sect. 3, a branch-and-bound algorithm for the medianoid (follower) and reverse medianoid problems with four different ways of obtaining an upper bound are introduced. In Sect. 4, a branch-and-bound algorithm for the (1|1)-centroid problem (leader) is described.

## 3 A branch-and-bound algorithm for the medianoid (follower) problem

In the medianoid problem ($FP(x_1)$), the follower wants to locate a new facility, knowing the location and the quality of all the facilities of the competitor (the leader). Next we describe the details of the algorithm for the follower problem. For the reverse medianoid problem of the leader, the algorithm is similar.

The basic idea in B&B methods consists of a recursive decomposition of the original problem into smaller disjoint subproblems until the solution is found. The method avoids visiting those subproblems which are known not to contain a solution. B&B methods can be characterised by four rules: *branching, selection, bounding*, and *elimination* (see Ibaraki 1976; Mitten 1970). For problems where the solution is determined with a desired accuracy, a *Termination rule* has to be incorporated. The method works as follows. The initial set $C_1 = S$ is subsequently partitioned in more and more refined subsets (branching) over which upper and lower bounds of the objective function are determined (bounding). In a maximisation problem, subsets with upper bounds lower than the best lower bound are eliminated for subsequent partitions (pruning), since these subsets cannot contain the maximum. At every iteration, the B&B method has a

list $\Lambda$ of subsets $C_k$ of $C_1$. The method stops when the list is empty. For every subset $C_k$ in $\Lambda$, upper bounds $z^{kU}$ of the objective function on $C_k$ are determined. Moreover, a global lower bound $z^L$ is updated. Next, we give a more detailed description of the steps of the algorithm.

### 3.1 The algorithm

To take both the medianoid and the reverse medianoid problems into account, we will denote by $M$ the objective function of the problem at hand and by $C$ its feasible set.

**Algorithm 1** : Branch-and-Bound algorithm for the (*reverse*) *medianoid problem*

**Funct** B&B($M, x, C, \varepsilon_f$)

1. $\Lambda := \emptyset$
2. $C_1 := C$
3. Determine an upper bound $z^{1U}$ on $C_1$
4. Compute $y^1$:=midpoint($C_1$), $BestPoint := y^1$
5. Determine lower bound: $z^1 := M(y^1)$, $z^L := z^1$
6. Put $C_1$ on list $\Lambda$, $r := 1$
7. **while** ( $\Lambda \neq \emptyset$ )
8. Take a subset $C$ (selection rule) from list $\Lambda$ and bisect into $C_{r+1}$ and $C_{r+2}$
9. **for** $t := r + 1$ to $r + 2$
10. Determine upper bound $z^{tU}$
11. **if** $z^{tU} > z^L + \varepsilon_f$
12. Compute $y^t$:=midpoint($C_t$) and $z^t := M(y^t)$
13. **if** $z^t > z^L$
14. $z^L := z^t$, $BestPoint := y^t$ and remove all $C_i$ from $\Lambda$ with $z^{iU} < z^L$
15. **if** $z^{tU} > z^L + \varepsilon_f$
16. save $C_t$ in $\Lambda$
17. $r := r + 2$
18. **endwhile**
19. OUTPUT: $\{BestPoint, z^L\}$

The B&B method is described in Algorithm 1. Its output is the best point found during the process and its corresponding function value. The best point is guaranteed to differ less than $\varepsilon_f$ in function value from the optimal solution of the problem (by considering the difference between lower and upper bounds).

### 3.2 Branching rule

The branching rule applied uses rectangles and new rectangles are generated by bisecting a subset $C$ over its longest edge. Two variants are implemented. Either we start with the initial rectangle $S$, or we start with an initial partition of it into rectangles such that none of the demand points is interior with respect to a rectangle. As will be outlined, this may improve the upper bounding applied, but on the other hand may generate more partition sets than strictly necessary.

### 3.3 Selection rule

The selection rule is important in the sense of efficiency measured by computational time and memory requirements. Within selection rules, one can find: depth-first-search, breadth-first-search and best-bound-search. In Sect. 5.1, the effect on efficiency of those rules is measured.

### 3.4 Lower bound

The classical lower bound is obtained as the best objective value at a finite set of feasible solutions $\{x_2^1, \ldots, x_2^r\}$

$$z^L = \max\{G(x_2^1), \ldots, G(x_2^r)\}.$$

for the follower problem. For the reverse medianoid problem, instead of taking $G(x_2)$ one should take $M_1(x_1, x_2)$. A good initial lower bound can be obtained by applying the (local search) Weiszfeld-like algorithm described in Drezner and Drezner (1994) from 20 or 50 starting random points. We simply use the best objective function value found at the evaluated points.

### 3.5 Upper bounds for the follower problem $(FP(x_1))$

The idea of the upper bound is to overestimate $M_2$ over a rectangle $C$. The market share captured by the follower (Eq. 1) can be rewritten as

$$M_2(x_1, x_2) = \sum_{i=1}^{n} \omega_i \frac{1 + \frac{1}{\alpha_2}\left(\sum_{j=k+1}^{m} \frac{a_j}{g(d_{ij})}\right) g(\delta_{i2})}{1 + \frac{1}{\alpha_2}\left(\frac{\alpha_1}{g(\delta_{i1})} + \sum_{j=1}^{m} \frac{a_j}{g(d_{ij})}\right) g(\delta_{i2})}. \tag{4}$$

Introducing

$$h_i = \frac{1}{\alpha_2} \sum_{j=k+1}^{m} \frac{a_j}{g(d_{ij})}$$

$$k_i = \frac{1}{\alpha_2}\left(\frac{\alpha_1}{g(\delta_{i1})} + \sum_{j=1}^{m} \frac{a_j}{g(d_{ij})}\right)$$

and defining

$$f_i(g(\delta_{i2})) = \frac{1 + h_i g(\delta_{i2})}{1 + k_i g(\delta_{i2})} \tag{5}$$

Eq. (4) becomes

$$M_2(x_1, x_2) = \sum_{i=1}^{n} \omega_i f_i(g(\delta_{i2})).$$

An upper bound for $M_2$ is

$$\overline{M}_2(x_1, x_2) = \sum_{i=1}^{n} \omega_i UB_i(C)$$

where $UB_i(C)$ is an overestimation of $f_i(g(\delta_{i2}))$ over rectangle $C$. Notice that $h_i < k_i$ and $f_i$ is monotonously decreasing in $g(\delta_{i2})$ with a limit of $\frac{h_i}{k_i}$.

We now describe various possible variants of the upper bounding. We will also evaluate numerically which bound is sharper than the others. The first upper bound is simply based on underestimating distance. The second and third upper bounds exploit the D.C. (difference of convex functions) structure of the objective function. The fourth upper bound builds a convex overestimating function based on the third one.

*3.5.1 Upper bound 1*

A first upper bound for $f_i(g(\delta_{i2}))$ over a rectangle $C$ is calculated in the following way. For demand point $p_i$, the distance to the follower $x_2$ is underestimated by assuming that $x_2$ delivers from the complete rectangle $C$. In this way the market share of the demand point for the follower is overestimated. The demand points within rectangle $C$ have a distance $\Delta_i(C) = 0$ from $C$. For demand points out of rectangle $C$, $p_i \notin C$, the shortest distance $\Delta_i(C)$ of $p_i$ to the rectangle is calculated. An upper bound $UB_i^1(C)$ for $f_i(g(\delta_{i2}))$ over rectangle $C$ for demand point $p_i$ is given by

$$UB_i^1(C) = \frac{1 + h_i g(\Delta_i(C))}{1 + k_i g(\Delta_i(C))}$$

where $\Delta_i(C)$ is the distance from demand point $p_i$ to rectangle $C$, $\Delta_i(C) = \min_{x \in C} d(x, p_i)$. The distance $\Delta_i(C)$ can be determined as follows. Rectangle $C$ is defined by two points: lower-left point $L = (L_1, L_2)$ and upper-right point $U = (U_1, U_2)$. The shortest distance from demand point $p_i$ to the rectangle $C = [L, U]$ can be computed by:

$$\begin{aligned} \Delta_{i1} &= \max\{L_1 - p_{i1}, p_{i1} - U_1, 0\} \\ \Delta_{i2} &= \max\{L_2 - p_{i2}, p_{i2} - U_2, 0\} \\ \Delta_i &= \sqrt{\Delta_{i1}^2 + \Delta_{i2}^2} \end{aligned}$$

Summarising,

$$\Delta_i(C) = \begin{cases} 0 & \text{if } p_i \in C \\ \sqrt{\Delta_{i1}^2 + \Delta_{i2}^2} & \text{if } p_i \notin C \end{cases} \tag{6}$$

This distance calculation is easily extendible to higher dimensions. A similar description is used in Plastria (1992). Equation (6) underestimates the distance from demand point $p_i$ to facilities in $C$. Since the new facility is only located at one point within the rectangle, we obtain an overestimation (upper bound) of the market capture of the new facility ($f_i(g(\delta_{i2}))$ is decreasing in $\delta_{i2}$).

### *3.5.2 Upper bound 2*

The second upper bound is more sophisticated and it is based on convexity of the functions $f_i$ and $g$. From now on, we will use the convex function $g(\delta_{i2}) = \sqrt{\delta_{i2}^2 + K_i^2}$ that was suggested in Drezner and Drezner (1997), where $K_i$ is a constant representing demand agglomeration. Equation (5) can be seen as a composition of functions $f_i$ and $g$. We will define an upper bound by using D.C. decomposition. A d.c. decomposition of a function $s$ defined on a convex $C \subset \mathbb{R}^n$ can be expressed, for all $x \in C$, in the form

$$s(x) = s_1(x) - s_2(x)$$

where $s_1$ and $s_2$ are convex functions on $C$. The following lemma is adapted from Lemma 1 in Tuy et al. (1995). Let $f'_+(x)$ be the right derivative of $f(x)$, $x \in \mathbb{R}$.

**Lemma 1** *Let $g(\delta(x))$ be a convex function on a convex and compact subset $C \subset \mathbb{R}^2$ such that $g(\delta(x)) \geq 0$ for all $x \in C$. If $f : \mathbb{R}_+ \mapsto \mathbb{R}$ is a convex nonincreasing function such that $f'_+(0) > -\infty$, then $f(g(\delta(x)))$ is a d.c. function in $C$ and can be expressed as*:

$$f(g(\delta(x))) = b(x) - Rg(\delta(x))$$

*where $b(x) = f(g(\delta(x))) + Rg(\delta(x))$ is a convex function for each positive constant $R$ satisfying $R \geq |f'_+(0)|$.*

By using Lemma 1 we can obtain a d.c. decomposition for each $f_i$. In particular, if $g(\delta_{i2}) = \sqrt{\delta_{i2}^2 + K_i^2}$, a d.c. decomposition for $f_i(g(\delta_{i2}))$ is defined by

$$f_i(g(\delta_{i2})) = b_i(x) - R_i g(\delta_{i2}) = b_i(x) - R_i\sqrt{\delta_{i2}^2 + K_i^2}$$

where $b_i(x) = f_i(g(\delta_{i2})) + R_i\sqrt{\delta_{i2}^2 + K_i^2}$ and $R_i = k_i - h_i$. Market capture for the follower can be expressed by

$$G(x) = M_2(x_1, x) = \sum_{i=1}^{n} \omega_i f_i(g(\delta_{i2})) = \sum_{i=1}^{n} \omega_i \left[ b_i(x) - R_i\sqrt{\delta_{i2}^2 + K_i^2} \right]$$

$$= \sum_{i=1}^{n} \omega_i \left\{ \frac{1 + h_i \sqrt{\delta_{i2}^2 + K_i^2}}{1 + k_i \sqrt{\delta_{i2}^2 + K_i^2}} + (k_i - h_i) \sqrt{\delta_{i2}^2 + K_i^2} \right\}$$
$$- \sum_{i=1}^{n} \omega_i (k_i - h_i) \sqrt{\delta_{i2}^2 + K_i^2}.$$

Let $\delta_i^2(x) = (\|x - p_i\|_2)^2$ be the squared Euclidean distance between $x$ and demand point $p_i$ and $V(C)$ be the set of vertices (corners) $v$ of rectangle $C$. An upper bound is defined as

$$UB = \max_{v \in V(C)} \left\{ \sum_{i=1}^{n} \omega_i \left\{ \frac{1 + h_i \sqrt{\delta_i^2(v) + K_i^2}}{1 + k_i \sqrt{\delta_i^2(v) + K_i^2}} + (k_i - h_i) \sqrt{\delta_i^2(v) + K_i^2} \right\} \right\}$$
$$- \min_{x \in C} \left\{ \sum_{i=1}^{n} \omega_i (k_i - h_i) \sqrt{\delta_{i2}^2 + K_i^2} \right\}$$

$UB$ is a valid upper bound of $M_2$ over $C$. To facilitate computation, one can underestimate $\min_{x \in C} \left\{ \sum_{i=1}^{n} \omega_i (k_i - h_i) \sqrt{\delta_{i2}^2 + K_i^2} \right\}$ by $\sum_{i=1}^{n} \omega_i (k_i - h_i) \sqrt{\Delta_i^2(C) + K_i^2}$. Then, $UB^2$ is defined as

$$UB^2(C) = \max_{v \in V(C)} \left\{ \sum_{i=1}^{n} \omega_i \left\{ \frac{1 + h_i \sqrt{\delta_i^2(v) + K_i^2}}{1 + k_i \sqrt{\delta_i^2(v) + K_i^2}} + (k_i - h_i) \sqrt{\delta_i^2(v) + K_i^2} \right\} \right\}$$
$$- \sum_{i=1}^{n} \omega_i (k_i - h_i) \sqrt{\Delta_i^2(C) + K_i^2}$$

*3.5.3 Upper bound 3*

For the ease of notation, let $z_i(x) = g(\delta_{i2})$. In this way, $G(x) = M_2(x_1, x)$ can be written as

$$G(x) = M_2(x_1, x) = \sum_{i=1}^{n} \omega_i f_i(z_i(x)) = \sum_{i=1}^{n} \omega_i \frac{1 + h_i z_i(x)}{1 + k_i z_i(x)}$$

Let $x^0$ be the midpoint of rectangle $C$ and $z_i^0 = z_i(x^0)$. According to Taylor's theorem there exist $g(\Delta_i) \leq \tilde{z}_i$ such that

$$G(x) = G(x^0) + \sum_{i=1}^{n} \omega_i \left[ \frac{h_i - k_i}{(1 + k_i z_i^0)^2} (z_i(x) - z_i^0) + \frac{k_i (k_i - h_i)}{(1 + k_i \tilde{z}_i)^3} (z_i(x) - z_i^0)^2 \right]$$

The first bounding operation is based on replacing $\tilde{z}_i$ by $g(\Delta_i)$,

$$G(x) \leq G(x^0) + \sum_{i=1}^{n} \omega_i \left[ \frac{h_i - k_i}{(1 + k_i z_i^0)^2} (z_i(x) - z_i^0) + \frac{k_i(k_i - h_i)}{(1 + k_i g(\Delta_i))^3} (z_i(x) - z_i^0)^2 \right]$$

By introducing

$$r_i = w_i \frac{k_i - h_i}{(1 + k_i z_i^0)^2}$$
$$s_i = w_i \frac{k_i(k_i - h_i)}{(1 + k_i g(\Delta_i))^3}$$
$$t_i = r_i + 2 s_i z_i^0$$

and rearranging terms we obtain

$$G(x) \leq G(x^0) + \sum_{i=1}^{n} (r_i z_i^0 + s_i (z_i^0)^2) - \sum_{i=1}^{n} t_i z_i(x) + \sum_{i=1}^{n} s_i z_i(x)^2 \tag{7}$$

Although $z_i$ is convex, the function in the right part of (7) is not. However, it is clearly a D.C. function. Let $V(C)$ be the set of vertices (corners) $v$ of rectangle $C$. Then, one can overestimate (7) by taking

$$UB = Const1 - \min_{x \in C} \sum_{i=1}^{n} t_i z_i(x) + \max_{v \in V(C)} \sum_{i=1}^{n} s_i z_i(v)^2$$

where $Const1 = G(x^0) + \sum_{i=1}^{n} (r_i z_i^0 + s_i (z_i^0)^2)$. As with upper bound $UB^2$, one can underestimate $\min_{x \in C} \sum_{i=1}^{n} t_i z_i(x)$ by $\sum_{i=1}^{n} t_i g(\Delta_i(C))$. Then, $UB^3$ is defined as

$$UB^3(C) = Const1 - \sum_{i=1}^{n} t_i g(\Delta_i(C)) + \max_{v \in V(C)} \sum_{i=1}^{n} s_i z_i(v)^2$$

*3.5.4 Upper bound 4*

In this section, a convex overestimation $\Gamma_C(x)$ of $G(x)$ over a rectangle $C$ is derived starting from (7). One can linearly overestimate the term $-t_i z_i(x)$ due to convexity of function $z_i(x)$ as follows

$$z_i(x) \geq z_i^0 + \nabla z_i^0 (x - x^0)$$

Substitution gives

$$G(x) \leq G(x^0) + \sum_{i=1}^{n}(r_i z_i^0 + s_i (z_i^0)^2) - \sum_{i=1}^{n} t_i z_i^0 - \sum_{i=1}^{n} t_i \nabla z_i^0 (x - x^0) + \sum_{i=1}^{n} s_i z_i(x)^2$$
$$= G(x^0) - \sum_{i=1}^{n} s_i (z_i^0)^2 - \sum_{i=1}^{n} t_i \nabla z_i^0 (x - x^0) + \sum_{i=1}^{n} s_i z_i(x)^2 = \Gamma_C(x)$$

Function $\Gamma_C(x)$ is convex. An upper bound over rectangle $C$, $UB^4(C)$, can be expressed by

$$UB^4(C) = Const2 + \max_{v \in V(C)} \left\{ \sum_{i=1}^{n} s_i z_i(v)^2 - \sum_{i=1}^{n} t_i \nabla z_i^0 (v - x^0) \right\}$$

where $Const2 = G(x^0) - \sum_{i=1}^{n} s_i (z_i^0)^2$.

## 4 A branch-and-bound algorithm for the leader problem

In this section, a new method based on Branch-and-Bound is formulated to generate a solution of the (1|1)-centroid problem. The final outcome is guaranteed to differ less in function value than a preset accuracy $\varepsilon_l$ from the optimum solution. Next, we introduce the algorithm and its ingredients.

### 4.1 The algorithm

The branching and selection rules used were the same as in Algorithm 1. The output of the B&B method (see Algorithm 2) is again the best point found during the process and its corresponding function value, which differs less than $\varepsilon_l$ from the optimum value of the problem.

### 4.2 Lower bound

The classical lower bound is obtained as the best objective value at a finite set of feasible solutions $\{x_1^1, \ldots, x_1^r\}$ for the leader problem,

$$z^L = \max\{F(x_1^1), \ldots, F(x_1^r)\}$$

One can follow the objective function value $F(x_1^p)$ of the iterates, or alternatively define an initial lower bound $z^L$ based on running another algorithm that generates a good approximate solution.

**Algorithm 2** : Branch-and-Bound algorithm for *Leader problem*

**Funct** B&BLeader($\varepsilon_l, \varepsilon_f$)

1. $\Lambda := \emptyset$
2. $C_1 = S$
3. Compute $x_1^1$:=midpoint($C_1$), $BestPoint := x_1^1$
4. Solve the problem for the follower: $\{x_2^1, z\} := \underline{\mathbf{B\&B}}(M_2, x_1^1, C_1, \varepsilon_f)$
5. Determine an upper bound $z_1^{1U}$ on $C_1$ solving a reverse medianoid problem: $\{y, z_1^{1U}\} := \underline{\mathbf{B\&B}}(M_1, x_2^1, C_1, \varepsilon_l)$
6. Determine lower bound: $z_1 := F(x_1^1) = M_1(x_1^1, x_2^1)$, $z^L := z_1$
7. Put $C_1$ on list $\Lambda$ , $r := 1$
8. **while** ( $\Lambda \neq \emptyset$ )
9. Take a subset $C$ (selection rule) from list $\Lambda$ and bisect into $C_{r+1}$ and $C_{r+2}$
10. **for** $t := r + 1$ to $r + 2$
11. Compute $x_1^t$:=midpoint($C_t$)
12. Solve the problem for the follower: $\{x_2^t, z\} := \underline{\mathbf{B\&B}}(M_2, x_1^t, C_1, \varepsilon_f)$
13. Determine upper bound $z_1^{tU}$ solving a reverse medianoid problem: $\{y, z_1^{tU}\} := \underline{\mathbf{B\&B}}(M_1, x_2^t, C_t, \varepsilon_l)$
14. **if** $z_1^{tU} > z^L + \varepsilon_l$
15. Determine $z_t := F(x_1^t) = M_1(x_1^t, x_2^t)$
16. **if** $z_t > z^L$
17. $z^L := z_t$, $BestPoint := x_1^t$, and remove all $C_i$ from $\Lambda$ with $z_1^{iU} < z^L$
18. **if** $z_1^{tU} > z^L + \varepsilon_l$
19. save $C_t$ in $\Lambda$
20. $r := r + 2$
21. **endwhile**
22. OUTPUT: $\{BestPoint, z^L\}$

### 4.3 Upper bounds

Let $C \subseteq \mathbb{R}^2$ denote a subset of the search region of $(LP)$, and assume that $x_2$ is given. An upper bound of $F(x_1)$ over $C$ can be obtained by having the leader solve the reverse medianoid problem.

**Lemma 2** *$UB(C, x_2) = \max_{x_1 \in C} M_1(x_1, x_2)$ is an upper bound of $F(x_1)$ over $C$.*

*Proof* According to (3), $F(x_1) = M_1(x_1, x_2^*(x_1)) \leq M_1(x_1, x_2)$ such that

$$\max_{x_1 \in C} F(x_1) \leq \max_{x_1 \in C} M_1(x_1, x_2) = UB(C, x_2).$$

□

Given a finite set $\{x_2^1, \ldots, x_2^r\}$ of feasible solutions for the follower, then

$$\min\{UB(C, x_2^1), \ldots, UB(C, x_2^r)\}$$

is an upper bound of $F(x_1)$ over $C$.

For a specific rectangle $C$, the choice of $x_2$ for the upper bound calculation is done as follows. We take $x^C = \text{midpont}(C)$ as the midpoint of the rectangle. Now one

solves $(FP(x^C))$ obtaining $\hat{x}_2$. An upper bound is determined by solving the problem

$$ub_1(C) = UB(C, \hat{x}_2) = \max_{x_1 \in C}\{M_1(x_1, \hat{x}_2)\}$$

Another easy possibility is to set $x_2$ equal to $x_1$ (that is, to assume co-location). In that way, one obtains the following upper bound.

**Lemma 3** *$ub_2(C) = UB(C, x_1) = \max_{x_1 \in C} M_1(x_1, x_1)$ is an upper bound of $F(x_1)$ over $C$.*

In the next two sections, we use numerical cases to illustrate the outcomes and efficiency of the algorithm.

## 5 Numerical examples

The effectiveness and efficiency of the algorithms are investigated with the aid of numerical cases. In a first case, we experiment with algorithm settings (variants of the algorithm) and study the performance. In the following cases, the performance is studied with a good algorithm setting. The effectiveness question concerns the algorithms and several ways of upper bounding. Performance indicators of the efficiency are the number of iterations used by the algorithms and the memory requirement. In general, branch-and-bound algorithms deliver a guarantee of detecting the global optimum up to a pre-set accuracy, but the cost of the memory requirement may be high if the dimension is going up or the accuracy is increasing, see e.g. Casado et al. (2007). In the first study, we will vary carefully the selection rule and the accuracy and inspect values of the performance indicators and effectiveness of the different bounds. Moreover, we evaluate a variant where an initial partition is generated to improve bound number 4. The second case is an illustration from literature. In the last case, we generate many instances at random where the size of the problem is varied to validate the viability of the approach with increasing number of demand points and existing facilities.

### 5.1 Case I: varying algorithm setting

This case has been generated randomly with $n = 10$ demand points, $m = 4$ existing facilities and a varying number $k$ of those facilities belonging to the leader's chain, $k = 0, \ldots, 4$. The generated demand points can be found in Appendix A (Table 11). The other parameters are chosen as follows:

- buying power: $w_i = 100,\ i = 1, \ldots, 10$
- quality of existing facilities: $a_j = 5.5,\ j = 1, \ldots, 4$
- quality of new facilities: $\alpha_l = 5,\ l = 1, 2$
- $g(d_{ij}) = \sqrt{(q_{j1} - p_{i1})^2 + (q_{j2} - p_{i2})^2 + (10^{-5})^2},\ i = 1, \ldots, 10,\ j = 1, \ldots, 4$
- $g(\delta_{il}) = \sqrt{(x_{l1} - p_{i1})^2 + (x_{l2} - p_{i2})^2 + (10^{-5})^2},\ l = 1, 2$
- accuracy for leader and follower: $\varepsilon_l = \varepsilon_f = 10^{-2}$.

**Table 1** Optimal locations and market capture for different number of leader facilities, $k = 0, \ldots, 4$. Parameter $z_l^* =$ market capture for the leader after locating facility, $Mb_l$ before; locations and market captures are rounded to two decimals

| | $k=0$ | $k=1$ | $k=2$ | $k=3$ | $k=4$ |
|---|---|---|---|---|---|
| Optimal location | | | | | |
| Leader | $\begin{pmatrix}2.44\\3.97\end{pmatrix}$ | $\begin{pmatrix}5.03\\0.69\end{pmatrix}$ | $\begin{pmatrix}5.33\\4.34\end{pmatrix}$ | $\begin{pmatrix}5.33\\4.34\end{pmatrix}$ | $\begin{pmatrix}5.03\\0.69\end{pmatrix}$ |
| Follower | $\begin{pmatrix}2.44\\3.97\end{pmatrix}$ | $\begin{pmatrix}5.03\\0.69\end{pmatrix}$ | $\begin{pmatrix}1.41\\4.65\end{pmatrix}$ | $\begin{pmatrix}1.75\\3.79\end{pmatrix}$ | $\begin{pmatrix}1.75\\3.79\end{pmatrix}$ |
| Market capture | | | | | |
| Leader | 186.29 | 367.87 | 497.70 | 611.07 | 773.44 |
| Follower | 813.71 | 632.13 | 502.30 | 388.93 | 226.56 |
| $z_l^* - Mb_l$ (gain or loss for the leader) | 186.29 | 100.67 | 14.17 | –72.46 | –226.56 |

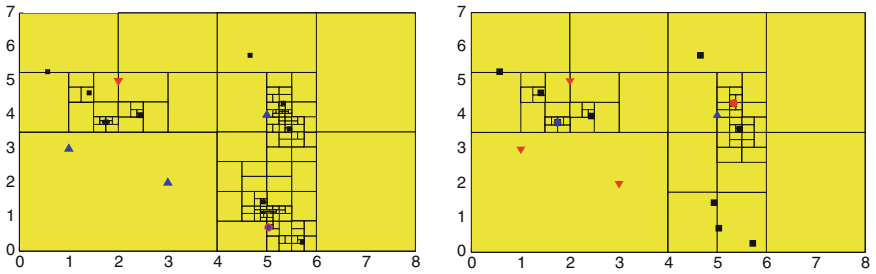

**Fig. 2** Generated partition by the algorithm. Cases with $k = 1$ (*left*) and $k = 3$ (*right*)

The resulting optimal locations are shown in Table 1, which also gives the market capture of both chains, when the number $k$ of existing facilities of the leader chain is increasing. One can observe a characteristic of the model, where leader and follower tend to co-locate when the number of existing facilities of the leader is low. In fact, the follower by locating at the same position, mitigates the effect of the relatively newcomer in the market who is going to compete for market capture. Notice also that when the leader is dominant in the market (it owns $k = 3$ of the $m = 4$ existing facilities, or all of them, $k = 4$) then the leader suffers a decrease in market share after the location of the two new facilities (see the negative values in the last line of Table 1). This is because in those cases the follower increases its market share more than the leader.

Figure 2 illustrates how the algorithm proceeds. It gives: location of the demand points (squares); location of the existing facilities (triangle up, belongs to the follower, triangle down, belongs to the leader); the optimum for the locations of leader (diamond) and the follower (circle) and the final partition of the search space for the leader for the cases when the number of existing facilities of the leader are $k = 1$ and $k = 3$. Each of the boxes has been evaluated and it has been proven by bounding that the optimum location of the leader cannot be there.

**Table 2** Efficiency of base case algorithm. Iterations. Upper bound $UB^1$ in Algorithm 1, selection rule: breadth-first-search in both algorithms

| $k$ | Leader problem | Medianoid problems | | | |
|---|---|---|---|---|---|
| | | Follower medianoid problems | | Reverse medianoid problems | |
| | | Max | Avg | Max | Avg |
| 0 | 1325 | 503 | 308.62 | 3645 | 215.48 |
| 1 | 1017 | 427 | 313.98 | 3107 | 248.09 |
| 2 | 1161 | 545 | 439.71 | 2709 | 166.13 |
| 3 | 209 | 501 | 447.42 | 2421 | 296.95 |
| 4 | 131 | 675 | 515.11 | 1009 | 190.15 |

In Tables 2 and 3 we focus on the efficiency of the algorithm and the different ways of bounding. Table 2 concerns the base case, where only $UB^1$ is used as upper bound in Algorithm 1, and breadth-first-search is used as selection rule in both Algorithms 1 and 2. It shows the number of iterations for the leader problem and the maximum and average number of iterations for Algorithm 1 when it is called at each iteration of Algorithm 2 to solve the corresponding (reverse) medianoid problems. First of all, one can observe from the number of iterations, that it is relatively easier for the algorithm to detect what is the global optimum for the leader when it has already many existing facilities. The intuition is as follows. When the leader is a newcomer, it has many options to gain market capture by going close to existing facilities of the competitor; there are many local optima. The result is that it is harder for the algorithm (requires more splitting) to verify that an already found location is the best one. Typically, this is easier when the leader has already several facilities. The global optimum is far more pronounced and defined by staying away from its own facilities. Accordingly, the number of iterations required for solving the follower medianoid problems increases with $k$.

In Table 3, we focus on the effectiveness of the upper bounds of Algorithm 1. At each iteration, it computes the four upper bounds described in Section 3.5 and chooses the minimum of the upper bounds. In all the cases, upper bounds $UB^1$ and $UB^4$ were used. Upper bounds $UB^2$ and $UB^3$ which are based on the d.c. concept appeared not to be efficient since they were never lower than $UB^1$ or $UB^4$. Observing the computations during the process, we found that $UB^4$ mainly improves the bounding of $UB^1$ when the partition sets get small. In this way, it contributes to speeding up the algorithm compared to only using $UB^1$. As in the previous table, the first two columns of Table 3 give the maximum and average number of iterations for Algorithm 1 when it is called at each iteration of Algorithm 2 to solve the corresponding (reverse) medianoid problems . The next four columns show the maximum and average number of iterations that the bounds $UB^1$ and $UB^4$ were the ones giving the minimum upper bound when solving the medianoid problems, whereas the last four columns give similar values when solving the reverse medianoid problems. Comparing Tables 2 and 3 we can see that the use of the both bounds reduces

**Table 3** Number of iterations and upper bounds used. Selection rule: breadth-first-search in both algorithms. Gives number of times $UB^1$ is the sharpest and $UB^4$ is the sharpest

| $k$ | Iterations | | | | Upper bounds used | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Follower medianoid problems | | Reverse medianoid problems | | Follower medianoid problems | | | | Reverse medianoid problems | | | |
| | | | | | $UB^1$ | | $UB^4$ | | $UB^1$ | | $UB^4$ | |
| | Max | Avg | Max | Avg | Max | Avg | Max | Avg | Max | Avg | Max | Avg |
| 0 | 497 | 295.70 | 3645 | 218.32 | 479 | 278.62 | 49 | 17.08 | 3645 | 208.39 | 695 | 9.93 |
| 1 | 411 | 302.16 | 3107 | 241.31 | 392 | 280.92 | 40 | 21.24 | 3107 | 222.89 | 1471 | 18.42 |
| 2 | 527 | 414.59 | 2709 | 164.11 | 496 | 390.28 | 58 | 24.31 | 2709 | 160.59 | 241 | 3.52 |
| 3 | 467 | 410.79 | 2421 | 291.36 | 418 | 367.99 | 60 | 42.80 | 2398 | 275.37 | 328 | 15.99 |
| 4 | 571 | 471.90 | 1009 | 190.91 | 495 | 412.98 | 91 | 58.92 | 1009 | 184.93 | 172 | 5.98 |

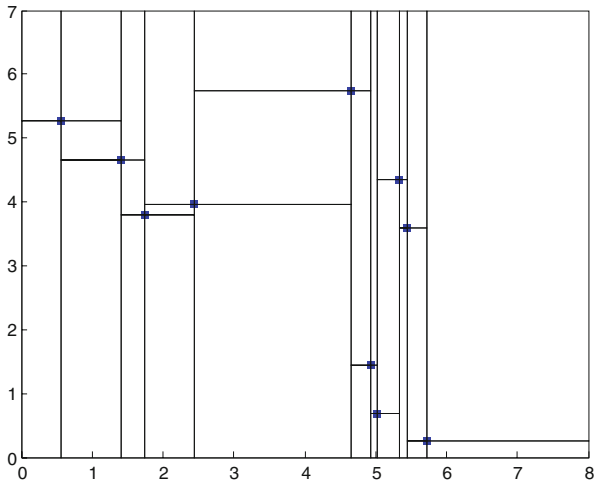


**Fig. 3** Initial partition generated for the follower medianoid

**Table 4** Efficiency changing to best bound selection. Iterations. Upper bound $UB^1$ in Algorithm 1, selection rule: best-bound-search in both algorithms

| $k$ | Leader problem | Medianoid problems | | | |
|---|---|---|---|---|---|
| | | Follower medianoid problems | | Reverse medianoid problems | |
| | | Max | Avg | Max | Avg |
| 0 | 689 | 613 | 184.25 | 2945 | 115.70 |
| 1 | 675 | 497 | 241.24 | 2893 | 71.21 |
| 2 | 1739 | 539 | 299.91 | 2519 | 58.59 |
| 3 | 463 | 401 | 362.57 | 8363 | 120.87 |
| 4 | 85 | 561 | 434.12 | 3871 | 140.64 |

the number of iterations required for solving the corresponding (reverse) medianoid problems.

In a next computational analysis we vary two rules of the algorithm. First of all, we compare the efficiency of the selection rule changing from breadth-first-search to best-bound-search, i.e., the rectangle $C_k$ such that $z^{kU} = \max\{z^{iU} : C_i \in \Delta\}$ is selected to be split next in Step 8 of Algorithm 1 and Step 9 of Algorithm 2. Secondly, we evaluate the performance when initially a partition is generated such that none of the demand points is interior as illustrated in Fig. 3. The idea is that the upper bounds $UB^4$ get sharper.

Comparing Tables 2 and 4, one can observe that Algorithm 1 clearly improves over the thousands of problems solved with the selection rule best-bound-search. Algorithm 2 for the leader problem does not always improve for this particular case. For the algorithm variant where the best upper bound is used, comparison of Tables 3

**Table 5** Efficiency when the best of the 4 upper bounds is used. Selection rule: best-bound-search. Gives number of times $UB^1$ is the sharpest and $UB^4$ is the sharpest

| $k$ | Iterations | | | | Upper bounds used | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Follower medianoid problems | | Reverse medianoid problems | | Follower medianoid problems | | | | Reverse medianoid problems | | | |
| | | | | | $UB^1$ | | $UB^4$ | | $UB^1$ | | $UB^4$ | |
| | Max | Avg | Max | Avg | Max | Avg | Max | Avg | Max | Avg | Max | Avg |
| No initial partition | | | | | | | | | | | | |
| 0 | 589 | 184.13 | 2943 | 116.72 | 537 | 163.97 | 81 | 20.16 | 2943 | 105.41 | 234 | 11.31 |
| 1 | 479 | 209.67 | 2891 | 70.07 | 466 | 192.50 | 54 | 17.17 | 2891 | 64.05 | 80 | 6.02 |
| 2 | 389 | 249.43 | 2517 | 50.95 | 325 | 226.04 | 76 | 23.39 | 2517 | 49.36 | 106 | 1.59 |
| 3 | 277 | 236.35 | 8363 | 116.47 | 233 | 214.83 | 44 | 21.52 | 8363 | 112.45 | 221 | 4.02 |
| 4 | 471 | 282.69 | 3871 | 141.90 | 390 | 249.48 | 84 | 33.20 | 3871 | 138.23 | 29 | 3.67 |
| With initial partition | | | | | | | | | | | | |
| 0 | 495 | 308.14 | 2856 | 146.22 | 473 | 269.90 | 101 | 38.24 | 2856 | 134.96 | 233 | 11.26 |
| 1 | 517 | 356.47 | 2938 | 76.81 | 415 | 297.37 | 115 | 59.10 | 2938 | 70.88 | 80 | 5.93 |
| 2 | 707 | 492.82 | 2578 | 53.54 | 617 | 407.77 | 148 | 85.05 | 2578 | 51.94 | 77 | 1.60 |
| 3 | 525 | 443.36 | 8363 | 126.74 | 480 | 392.15 | 79 | 51.21 | 8363 | 123.50 | 221 | 3.24 |
| 4 | 647 | 455.16 | 3871 | 143.81 | 525 | 391.40 | 142 | 63.76 | 3871 | 137.94 | 30 | 5.87 |

**Table 6** Memory requirement. The best of the 4 upper bounds is used. Selection rule: best-bound-search and $\varepsilon_l = 0.01$ and $\varepsilon_f = 0.01$

| $k$ | Leader problem | Follower medianoid problems | | Reverse medianoid problems | |
|---|---|---|---|---|---|
| | No. Rec. | Max | Avg | Max | Avg |
| 0 | 15 | 22 | 9.92 | 26 | 7.43 |
| 1 | 20 | 15 | 11.84 | 24 | 6.23 |
| 2 | 23 | 30 | 13.04 | 27 | 5.08 |
| 3 | 17 | 15 | 14.00 | 26 | 9.10 |
| 4 | 5 | 22 | 14.56 | 22 | 8.38 |

and 5 confirms that best-bound-search is better for Algorithm 1 than breadth-first-search.

Comparing efficiency between generating an initial partition or not, Table 5 shows that the case "No initial partition" is better for the medianoid problems. This effect is less for the reverse medianoid problems, because for this problem Algorithm 1 is applied to smaller rectangles.

We now focus on the memory requirement as performance indicator. As said, branch-and-bound algorithms are usually hindered by huge search trees that need to be stored in memory. This part of complexity usually increases rapidly with dimension and with accuracy. Table 6 shows the memory requirements when the best of the four upper bounds is used. Selection rule applied is best-bound-search for both algorithms and the accuracies are $\varepsilon_l = 0.01$ and $\varepsilon_f = 0.01$. The second column shows the number of rectangles required by Algorithm 2 as the maximum number stored during the iterations. In the columns 3 to 6 the maximum and average number (over the solved problems) are given of memory requirement for the medianoid and reverse medianoid problems, respectively.

One can observe that the memory requirement of the branch-and-bound approach for these continuous location problems is not dramatic for the used accuracy; there are never more than 30 subsets in the storage tree. Is this still the case if we increase accuracy? Notice that to have valid upper and lower bounds of the leader problem, the follower problem (giving lower bounds) and reverse medianoid (giving upper bounds) should be solved with an accuracy that is at least as tight as that of the leader problem. We evaluate the number of iterations as well as the memory requirement if the accuracy is tightened for the case where the number of existing facilities is taken as $k = 4$. The results in Table 7 show that the number of iterations of the algorithms increases less than linear with the used accuracy in terms of $1/\varepsilon$. The memory requirement hardly goes up, showing that the best bound selection rule is efficient.

Given the evaluations of different variants of the algorithm on this case, in the next cases we apply a best-bound selection rule, the best upper bound at each iteration and no initial partitioning of the domain is generated.

**Table 7** Efficiency when accuracy is increasing. Case with $k = 4$. Selection rule: best-bound-search

| | Accuracy of the leader | | | | | |
|---|---|---|---|---|---|---|
| | $\varepsilon_l = 0.01$ | | | $\varepsilon_l = 0.001$ | | $\varepsilon_l = 0.0001$ |
| | Accuracy of the medianoid and reverse medianoid problems | | | | | |
| | $\varepsilon_f$ | | | | | |
| | 0.01 | 0.001 | 0.0001 | 0.001 | 0.0001 | 0.0001 |
| Iterations | | | | | | |
| Leader | 85 | 95 | 95 | 143 | 151 | 219 |
| Follower med. (Avg) | 282.69 | 314.6 | 416.54 | 305.10 | 397.19 | 386.20 |
| Reverse med. (Avg) | 141.90 | 433.55 | 1186.64 | 296.20 | 784.34 | 549.65 |
| Memory | | | | | | |
| Leader | 5 | 6 | 6 | 8 | 9 | 9 |
| Follower med. (Avg) | 14.56 | 15.54 | 18.6 | 15.36 | 18.38 | 18.26 |
| Reverse med. (Avg) | 8.38 | 11.21 | 14.44 | 9.02 | 12.01 | 9.71 |

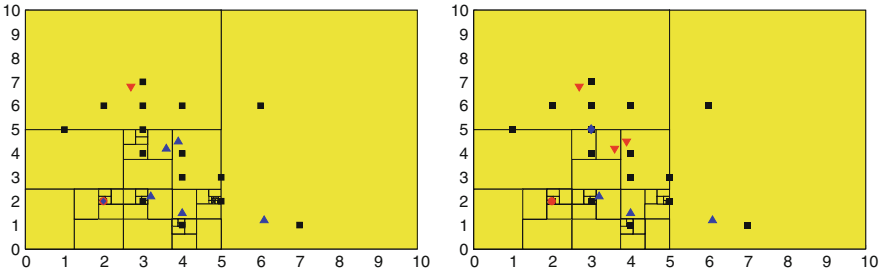

**Fig. 4** Generated partition by the algorithm. Case from Drezner and Drezner (1998): $k = 1$ (*left*), $k = 3$ (*right*)

## 5.2 Case II: from literature

In the second case where $n = 16$ and $m = 6$, data have been taken from Drezner and Drezner (1998). In that paper, the existing facilities all belong to other chains different from the leader or follower. Thus, to adjust the data to our model, we have assigned the first $k$ existing facilities to the leader and the rest to the follower. The data is different from randomly generated examples, as many points are situated along coordinate lines as can be observed from Fig. 4. The exact location of demand points and other facilities can be found in Appendix B (Tables 12, 13). Table 8 shows the results of the algorithm for $k = 0, \ldots, m$. The optimal locations and resulting market capture for both chains are given.

One can observe the co-location effect when the number of existing facilities of the leader is low. Notice that this effect can also be observed when the leader is a newcomer with less facilities than the follower. Co-location of the new facilities does not occur when the follower is a newcomer, albeit co-location occurs with an existing

**Table 8** Optimal locations Case II, market capture and number of iterations for both chains. Parameter $z_l^* =$ market capture after locating facility, $Mb_l$ before; locations and market captures are rounded to two decimals

| | $k=0$ | $k=1$ | $k=2$ | $k=3$ | $k=4$ | $k=5$ | $k=6$ |
|---|---|---|---|---|---|---|---|
| Optimal location | | | | | | | |
| Leader | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}2.00\\2.00\end{pmatrix}$ | $\begin{pmatrix}2.00\\2.00\end{pmatrix}$ | $\begin{pmatrix}2.00\\2.00\end{pmatrix}$ |
| Follower | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}1.99\\1.99\end{pmatrix}$ | $\begin{pmatrix}3.00\\5.00\end{pmatrix}$ | $\begin{pmatrix}3.00\\5.00\end{pmatrix}$ | $\begin{pmatrix}3.00\\5.00\end{pmatrix}$ | $\begin{pmatrix}3.00\\4.99\end{pmatrix}$ |
| Market capture | | | | | | | |
| Leader | 203.36 | 368.82 | 455.09 | 661.24 | 872.68 | 1037.21 | 1087.25 |
| Follower | 1143.14 | 977.68 | 891.41 | 685.26 | 473.82 | 309.29 | 259.25 |
| $z_l^* - Mb_l$ (gain or loss) | 203.36 | 157.31 | 129.67 | 48.31 | −140.26 | −234.34 | −259.25 |

facility of the competitor. Figure 4 gives an impression of the final partition generated by the branch-and-bound algorithm for the leader (cases with $k = 1$ and $k = 3$), together with the locations of demand points, existing facilities and new facilities.

Table 9 shows the number of iterations and the use of the 4 upper bounds. As in Case I, only upper bounds $UB^1$ and $UB^4$ were used.

Finally, Table 10 shows the memory requirements for Case II. The second column shows the maximum number of rectangles stored during the iterations by Algorithm 2. Columns 3 to 6 show the maximum and average number of rectangles stored for the follower medianoid and reverse medianoid, respectively.

### 5.3 Case III: varying problem dimension

In this section, numerical results of the evaluation of the Algorithms 1 and 2 are discussed. The wider question is whether the algorithms are able to solve larger problems in reasonable time. To study the performance of the algorithms, we have generated different types of problems, varying the number $n$ of demand points, the number $m$ of existing facilities and the number $k$ of facilities belonging to the leader chain. For every type of setting, ten problems were randomly generated. The settings are defined by choosing:

- $n = 20, 30, \ldots, 110$
- $m = 5, 10, 15$
- $k = [m/2]$.

For each $n, m$-combination parameter values of ten problems were uniformly chosen within the following intervals:

- $p_i, q_j \in ([0, 10], [0, 10]), i = 1, \ldots, n, j = 1, \ldots, m$
- $w_i \in [1, 10], i = 1, \ldots, n$
- $a_j \in [0.5, 5], j = 1, \ldots, m$.

**Table 9** Number of iterations when the best of the 4 upper bounds are considered. Selection rule: best-bound-search in Algorithm 1 and Algorithm 2. Gives number of times $UB^1$ is the sharpest and $UB^4$ is the sharpest

| $k$ | Iter | | | | | Upper bounds used | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Leader | Iterations | | | | Follower medianoid problems | | | | Reverse medianoid problems | | | |
| | | Follower medianoid problems | | Reverse medianoid problems | | $UB^1$ | | $UB^4$ | | $UB^1$ | | $UB^4$ | |
| | | Max | Avg | Max | Avg | Max | Avg | Max | Avg | Max | Avg | Max | Avg |
| 0 | 1417 | 913 | 450.32 | 4633 | 165.23 | 839 | 413.93 | 119 | 36.39 | 4633 | 128.21 | 2107 | 37.02 |
| 1 | 1127 | 297 | 232.14 | 1517 | 54.40 | 288 | 222.85 | 25 | 9.29 | 1517 | 48.00 | 121 | 6.40 |
| 2 | 715 | 277 | 217.93 | 2001 | 82.97 | 269 | 209.05 | 19 | 8.88 | 2001 | 81.62 | 117 | 1.35 |
| 3 | 249 | 261 | 174.36 | 1513 | 118.04 | 243 | 160.58 | 20 | 13.78 | 1513 | 107.06 | 315 | 10.98 |
| 4 | 177 | 239 | 183.17 | 573 | 83.25 | 214 | 153.96 | 33 | 29.21 | 573 | 75.65 | 103 | 7.60 |
| 5 | 181 | 249 | 190.83 | 405 | 63.19 | 219 | 155.67 | 38 | 35.16 | 405 | 59.58 | 37 | 3.61 |
| 6 | 125 | 389 | 248.33 | 557 | 61.77 | 345 | 215.78 | 44 | 32.55 | 557 | 56.76 | 29 | 5.01 |

**Table 10** Memory requirement Case II. Max number of stored rectangles

| $k$ | Leader | Follower medianoid | | Reverse medianoid | |
|---|---|---|---|---|---|
| | Problem | Max | Avg | Max | Avg |
| 0 | 22 | 29 | 18.32 | 27 | 9.80 |
| 1 | 24 | 12 | 11.15 | 26 | 6.11 |
| 2 | 16 | 11 | 10.92 | 28 | 6.18 |
| 3 | 10 | 12 | 11.16 | 28 | 7.15 |
| 4 | 10 | 12 | 11.77 | 17 | 6.58 |
| 5 | 10 | 12 | 12.00 | 21 | 6.14 |
| 6 | 10 | 15 | 12.73 | 22 | 6.45 |

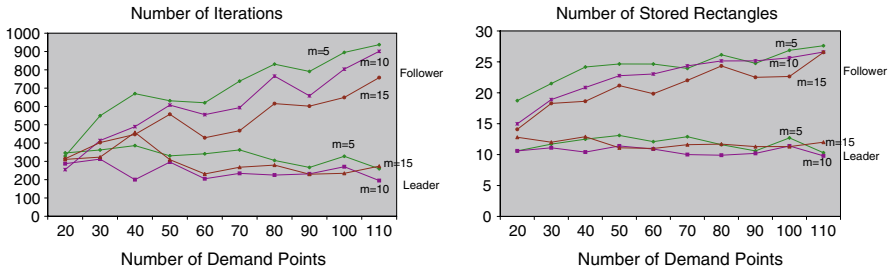


**Fig. 5** Average number of iterations and memory requirement (rectangles) over ten random cases varying number of demand points $n = 20, \ldots, 110$, existing facilities $m = 5, 10, 15$ and $k = m/2$. Selection rule: best-bound-search and $\varepsilon_l = \varepsilon_f = 0.01$

From Fig. 5, one can observe that an increasing number of demand points does not make the problem more complex in terms of the memory requirement for the branch-and-bound. The leader problem neither needs more iterations. The follower problem however, needs more iterations on average to reach the predefined accuracy. The experiment suggests that no exponential effort is required to solve the problems with increasing number of demand points. This confirms the viability of the approach.

## 6 Conclusions and future work

In this paper, we described a competitive Huff-like Stackelberg location model for market share maximisation. There are two competitors (chains); first the leader locates and then the follower makes a decision with full knowledge of choices of the leader. We consider competition with foresight and probabilistic behaviour. Attraction of a customer is depending on the location and the quality of the facility. The location of the leader facility is the variable of the problem. The problem is known to be a Global Optimisation problem. In order to solve it, we have constructed a branch-and-bound algorithm for the follower problem and for the leader problem. The branch-and-bound algorithms guarantee a global optimum within a given accuracy (gap between lower

and upper bound). The introduced bound of the leader problem is based on the zero sum concept where gain of one chain is loss for its competitor. We have developed and compared four different upper bounds for the algorithm of the (reverse) medianoid problem.

The algorithms were illustrated with several cases. In a first case, the algorithm settings and performance were studied. The selection rule and accuracy were varied to study the performance and effectiveness of the different bounds. A variant where an initial partition is generated was also studied. In a second case taken from literature, good algorithm settings from the first case were used. In the last case, many instances were generated at random where the size and the number of existing facilities is varied to validate the viability of the approach.

Looking at effectiveness, one can observe the co-location behaviour of the optimum strategy as one can expect. Also the difficulty on multimodal behaviour is reflected when measuring the efficiency as the number of iterations to solve the problem up to desired accuracy $\varepsilon$. Efficiency has been measured computationally. Comparing bounds and several variants with respect to selection rule and generating an initial partition to improve bounds, we found the following. More sophisticated bounds are not necessarily more effective than simple bounds based on distance comparison over the complete run of the algorithm. One can best focus on measuring the quality of the bound during the run and take the sharpest one. For the selection rule, the focus on the best bound (most promising) selection of the next subset to be split has the tendency to result in minimum effort on number of function evaluations. However, one always has to keep in mind that a depth first search may lead to less memory requirement of a branch-and-bound algorithm. Where memory requirement is usually a problem for higher dimensions, it is not necessarily a focus point for the location problem in two dimensional space.

Future research will include the quality of the leader and follower as variables of the problem.



## Appendix A: Test problems

**Table 11** Locations and distances from demand points to facilities

| Facility | Demand points | | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *X* axis | *Y* axis | 2.44<br>3.97 | 5.33<br>4.34 | 0.57<br>5.27 | 5.03<br>0.69 | 4.66<br>5.75 | 5.72<br>0.25 | 5.41<br>1.65 | 1.75<br>3.79 | 4.93<br>1.44 | 5.45<br>3.59 |
| **1** | 2 | 5 | 1.12 | 3.40 | 1.45 | 5.27 | 2.76 | 6.04 | 4.78 | 1.24 | 4.61 | 3.72 |
| **2** | 3 | 2 | 2.05 | 3.30 | 4.07 | 2.42 | 4.10 | 3.24 | 2.43 | 2.18 | 2.01 | 2.92 |
| **3** | 1 | 3 | 1.73 | 4.53 | 2.31 | 4.65 | 4.58 | 5.47 | 4.61 | 1.09 | 4.23 | 4.49 |
| **4** | 5 | 4 | 2.56 | 0.47 | 4.61 | 3.31 | 1.79 | 3.82 | 2.39 | 3.25 | 2.56 | 0.61 |

## Appendix B: Input data for example from Drezner and Drezner (1998)

**Table 12** Distances from demand points to facilities

| Facility | Demand points | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| **1** | 1.82 | 0.36 | 1.06 | 4.81 | 2.48 | 0.85 | 2.82 | 4.85 | 5.32 | 7.22 | 5.94 | 3.09 | 1.53 | 4.02 | 4.44 | 3.40 |
| **2** | 1.03 | 2.66 | 2.42 | 2.66 | 2.94 | 1.75 | 1.03 | 3.14 | 2.73 | 4.68 | 3.50 | 0.51 | 1.50 | 1.50 | 1.86 | 2.58 |
| **3** | 1.00 | 2.86 | 2.41 | 2.28 | 2.72 | 1.90 | 0.63 | 2.72 | 2.61 | 4.67 | 3.22 | 0.45 | 1.84 | 1.26 | 1.84 | 3.00 |
| **4** | 2.81 | 4.80 | 3.98 | 0.28 | 3.56 | 3.81 | 1.81 | 1.22 | 1.81 | 3.98 | 1.44 | 1.97 | 3.88 | 1.13 | 1.97 | 4.72 |
| **5** | 3.64 | 5.59 | 4.92 | 1.12 | 4.61 | 4.61 | 2.69 | 2.06 | 1.12 | 3.04 | 0.50 | 2.50 | 4.50 | 1.50 | 1.80 | 4.92 |
| **6** | 4.90 | 6.58 | 6.31 | 3.20 | 6.36 | 5.71 | 4.18 | 4.18 | 1.36 | 0.92 | 2.11 | 3.50 | 5.24 | 2.77 | 2.11 | 4.80 |

**Table 13** Location and buying power for demand points and location and attractiveness for existing facilities

| Number | Facility points | | |
|---|---|---|---|
| | $q_1$ | $q_2$ | $a_j$ |
| **1** | 2.7 | 6.8 | 7 |
| **2** | 3.9 | 4.5 | 3 |
| **3** | 3.6 | 4.2 | 7 |
| **4** | 3.2 | 2.2 | 10 |
| **5** | 4.0 | 1.5 | 7 |
| **6** | 6.1 | 1.2 | 3 |
| Number | Demand points | | |
| | $p_1$ | $p_2$ | $w_i$ |
| **1** | 3 | 5 | 163.8 |
| **2** | 3 | 7 | 28.8 |
| **3** | 2 | 6 | 39.0 |
| **4** | 3 | 2 | 77.4 |
| **5** | 1 | 5 | 42.0 |
| **6** | 3 | 6 | 107.0 |
| **7** | 3 | 4 | 64.5 |
| **8** | 2 | 2 | 250.6 |
| **9** | 5 | 2 | 101.4 |
| **10** | 7 | 1 | 57.6 |
| **11** | 4 | 1 | 132.0 |
| **12** | 4 | 4 | 77.6 |
| **13** | 4 | 6 | 29.6 |
| **14** | 4 | 3 | 67.5 |
| **15** | 5 | 3 | 50.7 |
| **16** | 6 | 6 | 57.0 |

## References

Bhadury J, Eiselt H, Jamarillo J (2003) An alternating heuristic for medianoid and centroid problems in the plane. Comput Operat Res 30:553–565

Casado LG, Hendrix EMT, García I (2007) Infeasibility spheres for finding robust solutions of blending problems with quadratic constraints. J Global Optimi 39(2):577–593. doi:10.1007/s10.898-007-9157-x

Drezner T (1994) Locating a single new facility among existing unequally attractive facilities. J Reg Sci 34(2):237–252

Drezner T, Drezner Z (1994) Optimal continuous location of a retail facility, facility attractiveness and market share: an interactive model. J Retail 70:49–64

Drezner T, Drezner Z (1997) Replacing continuous demand with discrete demand in a competitive location model. Naval Res Logist 44:81–95

Drezner T, Drezner Z (1998) Facility location in anticipation of future competition. Location Sci 6:155–173

Drezner T, Drezner Z (2004) Finding the optimal solution to the huff based competitive location model. Comput Manage Sci 1:193–208

Drezner Z (1982) Competitive location strategies for two facilities. Reg Sci Urban Econ 12:485–493

Eiselt H, Laporte G (1996) Sequential location problems. Eur J Opera Res 96:217–231

Eiselt H, Laporte G, Thisse JF (1993) Competitive location models: a framework and bibliography. Transp Sci 27:44–54

Fernández J, Pelegrín B, Plastria F, Tóth B (2007) Solving a huff-like competitive location and design model for profit maximization in the plane. Eur J Oper Res 179:1274–1287

Hakimi S (1983) On locating new facilities in a competitive environment. Eur J Oper Res 12:29–35

Ibaraki T (1976) Theoretical comparisons of search strategies in branch and bound algorithms. Int J Comput Inform Sci 5:315–344

Mitten LG (1970) Branch and bound methods: general formulation and properties. Oper Res 18:24–34

Plastria F (1992) Gbsss, the generalized big square small square method for planar single facility location. Eur J Oper Res 62:163–74

Plastria F (1997) Profit maximising single competitive facility location in the plane. Stud Locat Anal 11:115–126

Plastria F (2001) Static competitive facility location: an overview of optimisation approaches. Eur J Oper Res 129:461–470

Plastria F, Carrizosa E (2004) Optimal location and design of a competitive facility. Mathe Program 100:247–265

Tuy H, Al-Khayyal F, Zhou F (1995) A d.c. optimization method for single facility location problems. J Global Optim 7:209–227