p-FACILITY HUFF LOCATION PROBLEM ON NETWORKS *

Rafael Blanquero[1], Emilio Carrizosa[1], Boglárka G.-Tóth[2], and Amaya Nogales-Gómez[1]

[1] Departamento de Estadística e Investigación Operativa
Facultad de Matemáticas
Universidad de Sevilla
Spain
{rblanquero,ecarrizosa,amayanogales}@us.es

[2] Department of Differential Equations
Faculty of Mathematics
Budapest University of Technology and Economics
Hungary
bog@math.bme.hu

Abstract. The p-facility Huff location problem aims at locating facilities on a competitive environment so as to maximize the market share. While it has been deeply studied in the field of continuous location, in this paper we study the p-facility Huff location problem on networks formulated as a Mixed Integer Nonlinear Programming problem that can be solved by a branch and bound algorithm. We propose two approaches for the initialization and division of subproblems, the first one based on the straightforward idea of enumerating every possible combination of p edges of the network as possible locations, and the second one defining sophisticated data structures that exploit the structure of the combinatorial and continuous part of the problem. Bounding rules are designed using DC (difference of convex) and Interval Analysis tools.

In our computational study we compare the two approaches on a battery of 21 networks and show that both of them can handle problems for $p \leq 4$ in reasonable computing time.

Keywords: Huff location problem, location on networks, p-facility, branch and bound, DC, global optimization.

*This work has been partially supported by projects MTM2012-36163 of Ministerio de Economía y Competitividad, Spain, P11-FQM-7603 and FQM-329 of Junta de Andalucía, Spain.

1 Introduction

Competitive location models [12, 23] were originally introduced by Hotelling in [15], considering the location of two competing facilities on a linear market. In the seminal work of Hotelling, users patronize the facility closest to them. In contrast with this all·or·nothing assumption, it was introduced the Huff location model [16], in which the probability that a user patronizes a facility is proportional to its attractiveness and inversely proportional to a power of the distance to it. The Huff location problem has been extensively studied in the field of continuous location [6, 11, 13, 16, 17] and successfully applied in the marketing field, in problems such as location of petrol stations, shopping centers or restaurants [14, 20, 22].

Network optimization models [5] are widely used in practice due to their methodological aspects and intuitive formulations. They arise naturally in the context of assignment, flow, transportation or location problems among others [1, 19]. For a comprehensive introduction to location models on networks see [18].

The combination of the Huff location problem and network optimization has been already addressed in the literature [4, 8] and applied to market area analysis [20] and demand estimation [21]. The single-facility case has been solved in [4] by means of Interval Analysis (IA) bounds, and in [8] using IA and difference of convex (DC) bounds. Different metaheuristics have been proposed for the p-facility case in [25]. In this paper we solve the p-facility Huff location problem on networks formulated as a Mixed Integer Nonlinear Programming (MINLP) problem.

The remainder of this paper is organized as follows. In Section 2 we set up the notation for networks and introduce the p-facility Huff location problem. In Section 3, a branch and bound method with different initialization and branching rules is described. Section 4 is devoted to procedures for calculating lower and upper bounds. Computational results are reported in Section 5, where the p-facility Huff location problem is solved using the different branching and bounding rules for 12 real-life and 9 artificial networks. Finally, Section 6 contains a brief summary, final conclusions and some lines for future research.

2 The model

Let $N = (V, E)$ be a network, with node set V and edge set E. The length of the edge $e \in E$ is denoted by l_e. The distance between two nodes $a_i, a_j \in V$ is calculated as the length of the shortest path [18] from a_i to a_j. For each $e \in E$, with end-nodes a_i, a_j, we identify each $x \in [0, l_e]$ with the point in the edge e at distance x from a_i and $l_e - x$ from a_j. This way, we obtain that, for any vertex $a_k \in V$ and $x \in e$, the distance $d(x, a_k)$ from x to a_k, as a function of x, is a concave piecewise linear function, given by $d(x,\ a_k) = \min\{x + d(a_i,\ a_k),\ (l_e - x) + d(a_j,\ a_k)\}$.

In the p-facility Huff location model, the finite set V of vertices of the network represents users, asking for a certain service. Each user $a \in V$ has demand $\omega_a \geq 0$, that is patronized by different existing facilities, located at points $y_1, \ldots, y_r$ on the network. The demand captured by facility at y_i from user a is assumed to be inversely proportional to

a positive nondecreasing function of the distance $d(a, y_i)$, namely, $1/d(a, y_i)^2$ is used as the utility or attraction function of y_i. Therefore, the demand captured by the facility at y_i from the user at a is given by

$$\omega_a \frac{1/(d(a, y_i))^2}{\sum_{j=1}^{r} 1/(d(a, y_j))^2}. \tag{1}$$

A new firm is entering the market, by locating p new facilities at some points $x_1, \ldots, x_p$ on the network. This perturbs how the market is shared, since the new facilities will capture part of the demand from $a \in V$,

$$\omega_a \frac{\sum_{j=1}^{p} 1/(d(a, x_j))^2}{\sum_{j=1}^{p} 1/(d(a, x_j))^2 + \sum_{j=1}^{r} 1/(d(a, y_j))^2}. \tag{2}$$

Our goal is the maximization of the market share of the entering firm. Thus, the problem we need to solve can be formulated as

$$\max_{\substack{x_1 \in [0, l_{e_1}], \ldots, x_p \in [0, l_{e_p}] \\ e_1, \ldots, e_p \in E}} \sum_{a \in V} \omega_a \frac{\sum_{j=1}^{p} 1/(d(a, x_j))^2}{\sum_{j=1}^{p} 1/(d(a, x_j))^2 + \sum_{j=1}^{r} 1/(d(a, y_j))^2}. \tag{3}$$

Let us denote the total attraction of the existing facilities for each $a \in V$ by the positive constant

$$\beta_a = \sum_{j=1}^{r} \frac{1}{(d(a, y_j))^2}. \tag{4}$$

Problem (3) can be rewritten as the following MINLP:

$$\max_{\substack{x_1 \in [0, l_{e_1}], \ldots, x_p \in [0, l_{e_p}] \\ e_1, \ldots, e_p \in E}} F(x_1, \ldots, x_p) \tag{5}$$

where F is defined as

$$F(x_1, \ldots, x_p) = \sum_{a \in V} \omega_a \frac{1}{1 + \frac{\beta_a}{\sum_{j=1}^{p} \frac{1}{(d(a, x_j))^2}}}. \tag{6}$$

The MINLP problem (5) is formed by a combinatorial and continuous part. First, we need to solve the combinatorial problem of choosing a set of p edges to locate the facilities, and then solve a continuous location problem on the edges.

3 The methodology

The natural way to solve the MINLP formulation of the p-facility Huff location problem is to use a branch and bound method. In our methods we differenciate two main phases: the initialization phase and the branch and bound phase. In the initialization phase the initial

exploration tree is prepared. In the branch and bound phase, an element of the list is selected iteratively (until the termination rule is fulfilled) according to a selection criterion, and then is divided into new elements that are included into the list if they cannot be eliminated by their bounds. In this phase, division, bounding, selection, elimination and termination rules are required.

In this paper we propose different approaches for the initialization phase, division and bounding rules. As selection, elimination and termination rules, we always apply the usual ones from the literature [4]: the element to be evaluated is selected as the one with the largest upper bound, elements whose upper bound are lower than the current lower bound are eliminated, and the optimization is terminated when the relative error between the largest upper bound and the current lower bound is less than a fixed tolerance. This section is aimed at describing two types of initialization and division rules. Bounding rules will be discussed in Section 4.

3.1 Total enumeration

The straightforward way of solving Problem (6) is to separate the combinatorial and the continuous part of the problem: we first fix a set of p edges to locate the facilities, and then solve a continuous location problem on the edges. This means the branch and bound approach starts with a partition of the search space formed by the cartesion product of p-uples. The p-uples are formed by every possible combination of p-edges, taking into account that several facilities can be located at the same edge, i.e., repetitions of the same edge are allowed in the elements of the partition. But obviously, permutations of the p-uples are not taken into account.

We denote by $\underline{s} = (s_1, \ldots, s_k)$ an element of the partition, where each component s_i is a (sub)edge that has a multiplicity $m(s_i)$, i.e., the number of facilities located at s_i is $m(s_i)$. Hence, $m(s_1) + \ldots + m(s_k) = p$. To avoid symmetric sets, for any element of the partition $\underline{s} = (s_1, \ldots, s_k)$, and any $s_i = [l, u] \subseteq [0, l_e], e \in E, s_i \in \underline{s}$, the cartesian product $\Pi_{j=1}^{m(s_i)} s_i$ is replaced by $\{l \leq x_1 \leq \ldots \leq x_{m(s_i)} \leq u\}$.

The subdivision of each element of the partition is done by splitting each (sub)edge by its midpoint, obtaining two new smaller segments for each (sub)edge, namely lower and upper segments. Then, the new elements of the partition are built by replacing each (sub)edge s_i by either its lower or upper segment, s_i^L, s_i^U respectively. In the case of (sub)edges with multiplicity greater than 1, the above-described method is used to avoid symmetric sets. For instance, Figure 1 depicts the subdivision process, for $p = 2$, of the element $\underline{s} = (s_1)$, with $m(s_1) = 2$, identified with the blue coloured area of the big square. Then, the subdivision of $\underline{s}$ leads to three new elements, identified with the blue coloured area of the small squares.

3.2 Superset

A more sophisticated data structure for location problems on networks has been proposed in [9], exploiting together the structure of the combinatorial and continuous part of a

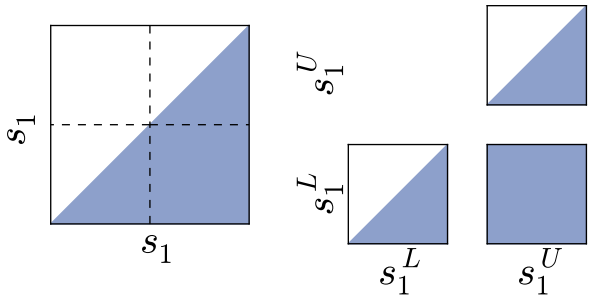

Figure 1: Subdivision process of $\underline{s} = (s_1)$ with $m(s_1) = 2$.

covering problem on networks.

In order to avoid the enumeration of every possible combination of p edges, [9] proposes to construct clusters of (sub)edges, called hereafter *edgesets*, and define a subproblem of (5) over a collection of edgesets called a *superset*.

To be precise, an edgeset is a finite collection of (sub)edges of E; a superset S is any uple of the form $(E_1, p_1; \ldots; E_k, p_k)$, where $E_1, \ldots, E_k$ are disjoint edgesets, p_j 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 (sub)edges in E_j.

For this data structure, the subproblem to be solved at this stage on superset S has the form

$$\max_{(x_1,\ldots,x_p)\in S} F(x_1, \ldots, x_p)$$

with F defined as in (6), and $(x_1, \ldots, x_p) \in S$ understood as $x_1, \ldots, x_{p_1} \in E_1$; $x_{p_1+1}, \ldots, x_{p_1+p_2} \in E_2$; $\ldots$; $x_{p-p_k+1}, \ldots, x_p \in E_k$.

Supersets will be identified with nodes in the branch and bound tree. The root node of the branch and bound tree is the original superset $S_0 = (E, p)$. E is first subdivided into a given partition $E^{(1)}, \ldots, E^{(p)}$ of E: we add to the branch and bound exploration tree the $\binom{2p-1}{p}$ supersets of the form $(E_1, p_1; \ldots; E_k, p_k)$, where $\{E_1, \ldots, E_k\} \subset \{E^{(1)}, \ldots, E^{(p)}\}$ and $p_1 + \ldots + p_k = p$.

First, we need to define how the edges of the network conforming S_0, are split into the partition of p edgesets $E^{(1)}, \ldots, E^{(p)}$. In the first step, E is divided into 2 edgesets by

a distance criterion, namely the diameter of the edgeset, defined as the maximum of the minimal distance between each pair of nodes. Then, the nodes giving the diameter are selected as centres of the two new (sub)edgesets. Each edge of the edgeset is assigned to the closest (sub)edgeset, where the distance from an edge to an (sub)edgeset is measured as the distance from the edge to the (sub)edgeset centre. Then, we will repeat the process until obtaining p edgesets, selecting the largest edgeset to be subdivided at each step, where the size of the edgeset is understood as the sum of the (sub)edgelengths.

The subdivision of a superset $S = (E_1, p_1; \ldots; E_k, p_k)$ during the branch and bound is done by partitioning the largest edgeset E_i. If E_i contains only one (sub)edge, the subdivision is done by bisecting the (sub)edge at its midpoint, otherwise E_i is partitioned to edgesets E_{i_1}, E_{i_2} by its diameter as done in the initial subdivision of S_0. Thus, the following supersets substitute S:

$$S_j = (E_1, p_1; \ldots; E_{i-1}, p_{i-1}; E_{i_j}, p_i; E_{i+1}, p_{i+1}; \ldots; E_k, p_k), j = 1, 2$$

and additionally if $1 < p_i$, for $j = 1, \ldots, p_i - 1$

$$S_{2+j} = (E_1, p_1; \ldots; E_{i-1}, p_{i-1}; E_{i_1}, j; E_{i_2}, p_i - j; E_{i+1}, p_{i+1}; \ldots; E_k, p_k).$$

This means, that in each step $p_i + 1$ new supersets are created.

4 Lower and Upper bounds

A branch and bound algorithm requires the calculation of tight upper and lower bounds. In this section we present different bounding approaches for the branch and bound used to solve (5). We propose two upper bounds and two lower bounds: IA bound, DC bound as upper bounds, Huff discrete bound (HuffDisc) and midpoint bound (MidPoint) as lower bounds. Note that when a superset contains edgesets with $|E_j| = 1 \; \forall j$, it corresponds to a p-uple of (sub)edges. Each p-uple of (sub)edges has its unique superset correspondance. Thus, the following bounds are valid for p-uples of edges as well, i.e., for the enumeration approach in Section 3.1.

4.1 Upper bounds

The IA bound considers only endpoints of (sub)edges as possible location of facilities. For a superset $S = (E_1, p_1; \ldots; E_k, p_k)$ we obtain the IA bound by replacing in (6) $d(a, x_j)$ by the distance from a to the closest vertex of the edges that belong to E_j, i.e., by

$$d(a, E_j) = \min_{e=[v_1,v_2], e \in E_j} \{d(a, v_1), d(a, v_2)\}.$$

For any $x \in E_j$ it holds that $d(a, E_j) \leq d(a, x) \; \forall j = 1, \ldots, p$. Hence, the following is a valid upper bound for (6):

$$UB^{IA}(S) := \sum_{a \in V} \omega_a \frac{1}{1 + \frac{\beta_a}{\sum_{j=1}^{p} \frac{p_j}{(d(a,E_j))^2}}}.$$

The second upper bound approach is based on a DC bound of the single facility Huff location problem on networks,

$$\max_{x \in [0,l_e],\, e \in E} F_{single}(x)$$

with $F_{single}(x) = \sum_{a \in V} \omega_a \frac{1}{1+\beta_a (d(a,x))^2}$, a particular case of (6) for $p = 1$. This problem has been already studied in [8]. First, for a given edge e of the network, $F_{single}(x)$ is expressed as a difference of convex functions, $F_{single}(x) = \sum_{a \in V} (F_a^+(x) - F_a^-(x))$, namely, its DC decomposition. Then, an upper bound $UB^{DC}_{single}(s)$ for any segment $s \in e,\ e \in E$ with $v_1,\ v_2$ being endpoints of s is defined as

$$UB^{DC}_{single}(s) = \max\{U(v_1), U(v_2)\}$$

with

$$U(x) = \sum_{a \in V} (F_a^+(x) - F_a^-(x_0) - \xi_a (x - x_0))$$

for $\xi_a \in \partial F_a^-(x_0)$ where $\partial F_a^-(x_0)$ denotes the set of subgradients of F_a^- at x_0 [26]. Therefore, it holds that

$$UB^{DC}_{single}(e) \geq F_{single}(x), \forall x \in [0, l_e],\ e \in E. \tag{7}$$

A DC bound over an edgeset E_j is defined as the maximum DC bound of the edges from E_j, i.e.,

$$UB^{DC}(E_j) := \max_{e \in E_j} UB^{DC}_{single}(e) \geq F_{single}(x), \forall x \in [0, l_e],\ e \in E.$$

Given a superset $S = (E_1, p_1; \ldots; E_k, p_k)$, a DC bound of (6) is calculated as

$$UB^{DC}(S) := \sum_{j=1}^{p} p_j \cdot UB^{DC}(E_j) \tag{8}$$

This DC bound is a valid upper bound since it holds that

$$\sum_{j=1}^{p} F_{single}(x_j) = \sum_{j=1}^{p} \sum_{a \in V} \omega_a \frac{1}{1 + \beta_a (d(a, x_j))^2} \tag{9}$$

Since $\frac{1}{(d(a,x_j))^2} \leq \sum_{j=1}^{p} \frac{1}{(d(a,x_j))^2}$, we have:

$$(9) = \sum_{j=1}^{p} \sum_{a \in V} \omega_a \frac{1/(d(a,x_j))^2}{1/(d(a,x_j))^2 + \beta_a} \geq \sum_{j=1}^{p} \sum_{a \in V} \omega_a \frac{1/(d(a,x_j))^2}{\sum_{i=1}^{p} 1/(d(a,x_i))^2 + \beta_a} =$$

$$= \sum_{a \in V} \omega_a \frac{\sum_{j=1}^{p} 1/(d(a,x_j))^2}{\sum_{i=1}^{p} 1/(d(a,x_i))^2 + \beta_a} = \sum_{a \in V} \omega_a \frac{1}{1 + \frac{\beta_a}{\sum_{j=1}^{p} \frac{1}{(d(a,x_j))^2}}} = F(x_1, \ldots, x_p).$$

4.2 Lower bounds

Both of our lower bounding approaches are based on the calculation of the objective function at a feasible solution $(\tilde{x}_1, \ldots, \tilde{x}_p) \in S$. Then, a valid lower bound is given by

$$LB(S) := F(\tilde{x}_1, \ldots, \tilde{x}_p) \leq \max_{(x_1,\ldots,x_p) \in S} F(x_1, \ldots, x_p).$$

Let us now focus on possible feasible solutions. The first lower bound, namely Huff discrete bound ($LB^{HuffDisc}$), is a greedy procedure based on solving iteratively p times the single facility Huff location problem at the vertices of the edges of the superset. For a given superset $S = (E_1, p_1; \ldots; E_k, p_k)$, let $\tilde{x}_1$ be the optimal solution of a single facility Huff location problem on the vertices of the (sub)edges of E_1. In the next step, we will consider that a facility is already located at $\tilde{x}_1$, and will locate $\tilde{x}_2$ solving the single facility Huff location problem at the vertices of the edges of the corresponding edgeset. In the last step, we will choose location $\tilde{x}_p$ as the optimal solution of the single facility Huff location problem on the vertices of the edges of E_k, considering that $p-1$ facilities are already located at $\tilde{x}_1, \ldots, \tilde{x}_{p-1}$. Since only vertices are considered as candidates, each step of the greedy procedure is executed by complete enumeration of the candidate points.

The second lower bound, namely the midpoint bound $LB^{MidPoint}$, is calculated by randomly choosing an edge from an edgeset E_j, and locating p_j facilities at its midpoint $\forall j \leq k$.

5 Computational results

The approaches described in Sections 3 and 4 were implemented in Fortran and executed on an Intel Core i7 computer with 16.00 Gb of RAM memory. The solutions were found within an accuracy of 10^{-3}.

We tested the approaches on a battery of 21 networks, whose characteristics are shown in Table 1. The first 9 networks are artificial networks generated as described in [2]. The following 5 networks are proposed for p-median problems in [3] and also used in [4]. Finally, the last 7 networks are taken from [10, 24]. The number r of existing facilities, is set as $r = 10\%$ of the number of edges of the network, $|E|$. Each instance is obtained by randomly and independently generating the demands (each vertex of the network is assumed to have a demand uniformly distributed in the interval $(0, 1)$ and in $(0, 20)$ for the artificial networks from [2]) and the location of the existing facilities. To generate the locations of the existing facilities, r edges are randomly chosen with replacement; on each selected edge, the facility location is generated following uniform distribution.

Tables 2-9 show a comparison between the two branching rules: total enumeration, Section 3.1, and superset, Section 3.2; and the different bounding approaches: IA bound, IA bound with DC bound (IA+DC), midpoint evaluation (MidPoint) and Huff discrete bound (HuffDisc). Results for DC bound as the only upper bound are not reported because they were systematically outperformed by the results in Tables 2-9. The results for the combinations of upper and lower bounds are shown in four blocks of columns. The first column shows the maximum size of the branch and bound tree (MaxList) during execution. The sign "×" in the MaxList column means that the size limit (10^8) was exceeded so the method was stopped. The second column reports the CPU time in seconds. Time limit is set to 6 hours (21600 seconds) and when it is exceeded, it is denoted with "×". In such case, the third column shows the gap in % achieved by the approach, measured as $\frac{UB-LB}{LB}100\%$.

We start with the analysis of $p = 2$, in Tables 2 and 3. All strategies are able to solve the problem on the 21 networks in less than an average time of 2 seconds. For both approaches, the best upper and lower bound choice is IA+DC and MidPoint respectively, with superset being the fastest approach and enumeration achieving the best MaxList size.

For $p = 3$, using supersets we achieve the best computing time, while using enumeration we achieve the best Maxlist result. For the superset approach, the choice of lower bound affects the behaviour of computing times, with an average improvement of about 200 seconds of MidPoint bound compared to HuffDisc bound. In the case of the enumeration approach, the choice of upper bound affects the MaxList size. Using IA+DC bound halves the MaxList size compared to using only IA bound. For both approaches, the best upper and lower bound choice is IA+DC and MidPoint respectively.

Tables 6 and 7 report results for $p = 4$. Using enumeration with HuffDisc bound solves 15 networks regardless of the upper bound used, while with MidPoint it solves 17 with IA bound and 18 if using IA+DC bound. Supersets with HuffDisc bound solves 15 networks regardless of the upper bound used, while with MidPoint we achieve successful results for 20 out of 21 networks. The $RAT195G$ network is not solved by any of the approaches. Using enumeration, its gap reduces from 32.24% to 17.67% if IA+DC bound is used. Using supersets with IA+DC bound its gap reduces from 15.94% to 8.24% when HuffDisc bound is used and from 12.57% to 9.93% for MidPoint bound. In terms of time, the best choice is to use MidPoint as lower bound, while the choice of upper bound does not make big difference for the superset approach, but for the enumeration approach, IA bound is the best choice. If we compare the results only for $art1 - art9$ networks, which are very small, see Table 1, the enumeration approach becomes the best in terms of all criteria. However, when the size of the network increases, the superset approach outperforms the enumeration one. For the enumeration approach, the difference between the bounding approaches in terms of MaxList size is hardly noticeable, being the best choice IA+DC as upper bound and MidPoint as lower bound. For the superset approach, slightly better MaxList sizes are achieved when using HuffDisc bound.

Finally, we analyze results for $p = 5$, Tables 8 and 9. Using enumeration we are able to solve the problem on 8 networks while with supersets on 7 networks. Using enumeration,

Table 1: Properties of the networks taken from [2, 3, 10, 24]

Network	nodes	edges
art1	20	38
art2	20	43
art3	20	51
art4	30	56
art4	30	71
art5	30	84
art7	40	74
art8	40	95
art9	40	115
pmed1	100	196
pmed2	100	191
pmed3	100	196
pmed4	100	194
pmed5	100	194
KROB150G	150	296
KROA150G	150	297
PR152G	152	296
RAT195G	195	336
KROB200G	200	386
KROA200G	200	392
TS225G	225	306

there is a big outperformance in terms of gap achieved by IA bound over IA+DC bound. In terms of computing time, lower bound makes a small difference, MidPoint bound being the fastest one. In terms of MaxList size, there is no big difference between the bounding approaches. Using supersets, the choice of lower bound makes a big difference in terms of MaxList size and time. There is an average improvement of more than 3 hours of MidPoint bound over HuffDisc bound. On the contrary, in terms of MaxList size, HuffDisc bound outperforms MidPoint bound, the MaxList size of the latter being about the double of HuffDisc MaxList size. These big differences in the behaviour when changing lower bound are due to HuffDisc bound being computationally expensive, which makes the approach stop due to time limit, and MidPoint bound less efficient, which explodes the size of the MaxList tree. Better gaps are achieved using IA+DC bound. For the enumeration approach, we achieve better results when using IA bound with HuffDisc bound. For the superset approach, the best lower bound choice is using MidPoint bound while the choice of upper bound is irrelevant.

In summary, we can say that both approaches are comparable for $p = 2, 3$ while for $p = 4, 5$ the superset approach outperforms the enumeration approach. When faced with the choice of the best upper bound, using IA+DC bound as upper bound is the best choice for both approaches and all values of p, except for $p = 5$ for the enumeration approach. In terms of lower bound, we observe that using MidPoint as lower bound is, in general, the best choice, except for $p = 5$ for the enumeration approach.

6 Conclusions

In this paper we have addressed the p-facility Huff location problem on networks. We propose two branch and bound based approaches and show results for $p \leq 5$. Computational results show that both division approaches are able to solve problems of rather realistic size up to $p = 4$ facilities while for $p = 5$ only small problems are solved. For small values of p, both approaches are comparable, and when the number of facilities increases, the superset approach outperforms the enumeration approach. We conclude with three promising extensions.

As shown in Section 5, for high values of p and for both approaches, some problems remain unsolved because the MaxList size limit is reached. It could be interesting to design a heuristic approach able to reduce the number of elements of the partition, and exploit the benefits of the branch and bound tree evolution.

As second extension, both approaches could be applied to different p-facility location problems on networks, such as the p-median problem with continuous demand on a network [7].

Finally, parallelization techniques deserve further study. Parallelizing the approach can solve the problem of reaching the MaxList size limit and may reduce the computational cost linearly, which will definitely lead to solving the problem for higher values of p.

References

[1] S. Alumur and B.Y. Kara. Network hub location problems: The state of the art. *European Journal of Operational Research*, 190(1):1–21, 2008.

[2] I. Averbakh, O. Berman, D. Krass, J. Kalcsics, and S. Nickel. Cooperative covering problems on networks. *Networks*, 63(4):334–349, 2014.

[3] J.E. Beasley. OR-library: Distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41:1069–1072, 1990.

[4] O. Berman, Z. Drezner, and D. Krass. Big segment small segment global optimization algorithm on networks. *Networks*, 58(1):1–11, 2011.

[5] D.P. Bertsekas. *Network Optimization: Continuous and Discrete Models*. Athenas Scientific, Belmont, Mass, 1998.

[6] R. Blanquero and E. Carrizosa. Continuous location problems and big triangle small triangle: Constructing better bounds. *Journal of Global Optimization*, 45:389–402, 2009.

[7] R. Blanquero and E. Carrizosa. Solving the median problem with continuous demand on a network. *Computational Optimization and Applications*, 56(3):723–734, 2013.

[8] 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, 2014.

[9] R. Blanquero, E. Carrizosa, and B.G. Tóth. Maximal covering location problems on networks with regional demand. *Optimization Online*, 2014.

[10] A. Corberán and J.M. Sanchis. A branch & cut algorithm for the windy general routing problem and special cases. *Networks*, 49:245–257, 2007.

[11] T. Drezner and Z. Drezner. Finding the optimal solution to the Huff competitive location model. *Computational Management Science*, 1:193–208, 2004.

[12] H.A. Eiselt, G. Laporte, and J.-F. Thisse. Competitive location models: A framework and bibliography. *Transportation Science*, 27(1):44–54, 1993.

[13] J. Fernández, B. Pelegrín, F. Plastria, and B.G. Tóth. Solving a Huff-like competitive location and design model for profit maximization in the plane. *European Journal of Operational Research*, 170:1274–87, 2007.

[14] A. Ghosh, S. McLafferty, and C.S. Craig. Multifacility retail networks. In Z. Drezner, editor, *Facility Location. A Survey of Applications and Methods*, pages 301–330, New York, 1995. Springer.

[15] H. Hotelling. Stability in competition. *Economic Journal*, 39:41–57, 1929.

[16] D.L. Huff. Defining and estimating a trading area. *Journal of Marketing*, 8:28–34, 1964.

[17] D.L. Huff. A programmed solution for approximating an optimum retail location. *Land Economics*, 8(42):293–303, 1966.

[18] M. Labbé, D. Peeters, and J.F. Thisse. Location on Networks. In M.O. Ball et al., editor, *Handbooks in operations research and management science*, volume 8, pages 551–624, Amsterdam, 1995. Elsevier.

[19] V. Marianov and D. Serra. Location-allocation of multiple-server service centers with constrained queues or waiting times. *Annals of Operations Research*, 111:35–50, 2002.

[20] A. Okabe and M. Kitamura. A computational method for market area analysis on a network. *Location Science*, 5(3):198–198, 1997.

[21] A. Okabe and K.-I. Okunuki. A computational method for estimating the demand of retail stores on a street network and its implementation in GIS. *Transactions in GIS*, 5(3):209–220, 2001.

[22] K.-I. Okunuki and A. Okabe. Solving the Huff-based competitive location model on a network with link-based demand. *Annals of Operations Research*, 111(1-4):239–252, 2002.

[23] F. Plastria. Static competitive facility location: An overview of optimisation approaches. *European Journal of Operational Research*, 129(3):461–470, 2001.

[24] G Reinelt. TSPLIB - A traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.

[25] S. Roksandić, E. Carrizosa, D. Urošević, and N. Mladenović. Solving multifacility Huff location models on networks using variable neighborhood search and multi-start local search metaheuristics. *Electronic Notes in Discrete Mathematics*, 39(0):121 – 128, 2012.

[26] H. Tuy. *DC Optimization: Theory, Methods and Algorithms, Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht, Holland, 1995.

Table 2: Maximum branch and bound tree size and running times for $p = 2$ for the enumeration approach.

	enumeration							
Upper bound	IA				IA+DC			
Lower bound	HuffDisc		MidPoint		HuffDisc		MidPoint	
Network	MaxList	time	MaxList	time	MaxList	time	MaxList	time
art1	535	0.00	642	0.00	682	0.00	798	0.02
art2	320	0.02	346	0.00	664	0.00	664	0.02
art3	378	0.02	382	0.00	478	0.02	478	0.00
art4	377	0.02	390	0.00	861	0.03	861	0.00
art5	1256	0.02	1346	0.00	1301	0.03	1302	0.02
art6	1351	0.03	1362	0.03	1145	0.03	1145	0.02
art7	1594	0.03	1612	0.03	931	0.02	934	0.00
art8	1417	0.03	1512	0.03	922	0.03	938	0.03
art9	1497	0.06	1538	0.03	1636	0.08	1640	0.03
pmed1	1094	0.25	1130	0.16	425	0.19	429	0.17
pmed2	2747	0.22	2796	0.17	805	0.16	809	0.17
pmed3	1184	0.22	1184	0.16	334	0.17	334	0.17
pmed4	675	0.20	677	0.16	175	0.17	175	0.17
pmed5	3323	0.31	3365	0.17	1164	0.20	1164	0.17
KROB150G	4143	1.53	4561	0.61	381	0.76	451	0.58
KROA150G	4897	1.70	5003	0.66	1039	0.95	1039	0.61
PR152G	712	0.95	1014	0.56	494	0.87	586	0.61
RAT195G	17877	7.13	17916	1.51	2149	1.84	2149	1.09
KROB200G	3489	2.76	3792	1.31	1237	2.22	1315	1.40
KROA200G	2675	2.36	2828	1.28	546	1.93	546	1.40
TS225G	3822	1.61	3843	0.90	1325	1.20	1325	0.97
Average	2636	0.93	2725	0.37	890	0.52	908	0.36

Table 3: Maximum branch and bound tree size and running times for $p = 2$ for the superset approach.

	superset							
Upper bound	IA				IA+DC			
Lower bound	HuffDisc		MidPoint		HuffDisc		MidPoint	
Network	MaxList	time	MaxList	time	MaxList	time	MaxList	time
art1	298	0.03	324	0.02	298	0.02	324	0.02
art2	920	0.03	1087	0.03	920	0.03	1087	0.02
art3	569	0.03	1602	0.02	466	0.03	1336	0.00
art4	580	0.03	835	0.02	580	0.03	835	0.00
art5	1310	0.09	2106	0.03	1185	0.08	1895	0.02
art6	1299	0.09	1884	0.03	1299	0.09	1884	0.02
art7	1641	0.11	1972	0.03	1605	0.12	1969	0.03
art8	2370	0.22	2432	0.05	2370	0.23	2426	0.06
art9	3367	0.31	4709	0.06	2425	0.30	4184	0.06
pmed1	3423	1.22	3738	0.2	3260	1.20	3380	0.20
pmed2	2371	0.80	3703	0.14	2137	0.80	3706	0.14
pmed3	1950	0.64	2286	0.11	1854	0.61	2247	0.11
pmed4	5883	1.54	8262	0.27	5682	1.51	8273	0.27
pmed5	3466	1.08	4045	0.17	2050	1.08	3046	0.19
KROB150G	5373	3.07	6482	0.37	3006	2.64	5055	0.30
KROA150G	5070	3.06	6113	0.37	4432	2.54	4970	0.33
PR152G	465	0.67	638	0.08	381	0.39	533	0.05
RAT195G	21719	13.73	25581	1.61	11951	13.56	15576	1.51
KROB200G	4529	5.02	5499	0.51	2477	2.54	3018	0.28
KROA200G	3548	4.49	4687	0.47	1624	4.07	2134	0.39
TS225G	4336	3.81	5733	0.42	3887	3.78	5135	0.42
Average	3547	1.91	4463	0.24	2566	1.70	3477	0.21

Table 4: Maximum branch and bound tree size and running times for $p = 3$ for the enumeration approach.

	enumeration							
Upper bound	IA				IA+DC			
Lower bound	HuffDisc		MidPoint		HuffDisc		MidPoint	
Network	MaxList	time	MaxList	time	MaxList	time	MaxList	time
art1	8267	0.11	9910	0.09	9866	0.39	15477	0.23
art2	6188	0.08	7800	0.08	13756	0.55	13922	0.23
art3	10584	0.36	11793	0.17	18592	0.62	20144	0.31
art4	7916	0.41	9198	0.23	28851	1.90	29185	0.73
art5	40940	0.90	41153	0.51	59592	3.56	59663	1.29
art6	41664	1.47	43088	0.80	71196	3.67	71751	1.51
art7	45641	1.25	45862	0.84	57556	2.56	57750	1.14
art8	58719	1.54	59989	0.97	98178	3.39	98979	1.75
art9	52667	3.84	53788	1.73	135925	13.17	135925	4.04
pmed1	69966	18.99	70310	13.28	58693	22.95	58923	16.94
pmed2	156224	16.99	156992	13.10	93928	17.33	93952	15.10
pmed3	55064	19.06	55064	13.24	36455	19.83	36455	16.58
pmed4	46950	17.11	47374	12.76	22048	18.19	22048	15.91
pmed5	201832	34.30	205948	16.05	116076	24.74	116218	17.25
KROB150G	336291	268.09	353294	86.27	48164	112.79	48164	84.57
KROA150G	379461	324.26	385955	90.07	137050	194.86	137050	93.49
PR152G	50488	131.56	51722	69.84	66332	159.98	66332	90.03
RAT195G	1822569	1306.57	1823360	231.01	249618	280.55	249618	167.53
KROB200G	390512	597.89	399780	217.18	223063	513.26	223063	262.47
KROA200G	207484	324.93	208160	203.71	58563	290.55	58563	254.16
TS225G	270334	216.11	270894	115.80	123379	191.80	124312	141.23
Average	202846	156.47	205306	51.80	82232	89.36	82738	56.50

Table 5: Maximum branch and bound tree size and running times for $p = 3$ for the superset approach.

	superset							
Upper bound	IA				IA+DC			
Lower bound	HuffDisc		MidPoint		HuffDisc		MidPoint	
Network	MaxList	time	MaxList	time	MaxList	time	MaxList	time
art1	3880	0.39	5397	0.09	3880	0.39	5397	0.09
art2	16975	1.05	27467	0.27	16975	1.06	27467	0.28
art3	24193	2.17	53640	0.53	23876	2.17	53525	0.51
art4	9803	0.92	18388	0.22	9803	0.94	18388	0.20
art5	41377	4.07	66646	0.94	41148	4.07	66605	0.95
art6	58757	6.12	75996	1.19	58757	6.16	75996	1.23
art7	67579	7.69	73556	1.61	67579	7.74	73556	1.65
art8	88324	11.62	117422	2.18	88324	11.65	117422	2.26
art9	139697	18.83	227709	3.29	139596	18.84	227624	3.31
pmed1	271844	169.71	288840	21.76	271947	167.11	288840	21.90
pmed2	222849	111.34	378409	14.59	222851	110.09	378514	14.63
pmed3	135698	64.88	187136	8.22	135370	63.82	186754	8.30
pmed4	418453	167.26	631093	21.82	409319	165.45	621040	21.81
pmed5	223885	106.75	264197	13.65	219336	105.53	260945	13.65
KROB150G	494950	394.23	593159	39.98	438955	368.16	522850	36.40
KROA150G	478892	410.13	578185	42.04	470628	408.25	568164	41.04
PR152G	58848	81.81	68886	8.03	61306	78.16	67491	7.88
RAT195G	2190021	2026.48	2768940	189.42	1242889	1877.61	1687197	166.50
KROB200G	436217	647.47	549967	54.12	340731	479.09	429621	38.58
KROA200G	262735	435.37	341576	35.07	214576	397.79	290266	30.37
TS225G	315713	379.91	399584	36.02	306847	376.56	388696	34.24
Average	283842	240.39	367438	23.57	227843	221.46	302684	21.23

Table 6: Maximum branch and bound tree size, running times and achieved gaps for $p = 4$ for the enumeration approach.

	enumeration											
Upper bound	IA						IA+DC					
Lower bound	HuffDisc			MidPoint			HuffDisc			MidPoint		
Network	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap
art1	87598	1.68	–	117441	1.22	–	101277	8.63	–	198557	4.29	–
art2	74095	1.23	–	101784	1.58	–	163003	15.16	–	167512	5.40	–
art3	177508	8.55	–	293743	4.43	–	307426	29.08	–	490398	12.36	–
art4	159413	8.46	–	185915	5.24	–	454552	68.27	–	459312	21.04	–
art5	873673	39.11	–	881711	16.61	–	1150102	199.56	–	1177881	59.86	–
art6	975158	51.75	–	984445	23.57	–	2080085	327.80	–	2129849	84.52	–
art7	1028331	58.73	–	1036206	24.90	–	1347049	268.88	–	1406366	71.64	–
art8	1616556	92.37	–	1651432	40.20	–	3423277	659.45	–	3426785	158.54	–
art9	1100215	141.68	–	1111150	63.32	–	5890324	1527.42	–	5890324	322.58	–
pmed1	3920397	1314.92	–	3927269	851.34	–	6667549	3020.66	–	6667781	1292.31	–
pmed2	7644966	959.14	–	7646396	763.20	–	9792019	1369.41	–	9792181	1027.34	–
pmed3	2342677	1217.93	–	2344253	832.70	–	4280821	2344.15	–	4280821	1212.02	–
pmed4	2652444	1252.69	–	2685214	817.60	–	2767490	1840.00	–	2767490	1100.93	–
pmed5	8830603	2538.98	–	8979523	1041.57	–	9393530	3119.91	–	9398491	1350.89	–
KROB150G	27979117	×	0.04	29666987	9798.09	–	8609286	20606.66	–	8609286	9043.41	–
KROA150G	×	5256.88	34.17	×	5348.99	34.17	24167062	×	0.03	24203649	11270.48	–
PR152G	2993638	13190.67	–	3099354	6683.47	–	8203867	×	0.01	8203867	9327.78	–
RAT195G	×	2956.78	32.24	×	3000.29	32.24	×	12331.88	17.67	×	12191.70	17.67
KROB200G	×	21119.03	26.10	×	20559.01	26.10	×	20059.76	27.89	24150000	×	0.27
KROA200G	22110803	×	0.18	22149983	×	0.23	8670000	×	0.26	8670000	×	0.26
TS225G	22200741	×	0.02	22243742	12332.64	–	18379844	×	0.04	18379844	15321.28	–
Average	19369901	5476.69	4.42	19481264	3990.95	4.42	15040407	7342.70	2.19	11450971	5098.97	0.87

Table 7: Maximum branch and bound tree size, running times and achieved gaps for $p = 4$ for the superset approach.

	superset											
Upper bound	IA						IA+DC					
Lower bound	HuffDisc			MidPoint			HuffDisc			MidPoint		
Network	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap
art1	157141	19.22	–	152898	4.23	–	157141	20.22	–	152898	4.35	–
art2	181037	17.05	–	326290	3.81	–	181037	17.89	–	326290	3.87	–
art3	1093070	156.98	–	2659877	34.73	–	1093070	165.64	–	2659878	36.05	–
art4	101627	13.63	–	248993	2.65	–	101627	14.23	–	248998	2.70	–
art5	784382	109.43	–	1730167	21.61	–	784350	114.36	–	1730170	21.96	–
art6	1346499	196.92	–	1845530	33.52	–	1346499	202.04	–	1845530	34.68	–
art7	1415772	222.43	–	1581166	39.30	–	1415772	225.20	–	1581166	40.17	–
art8	3459070	702.52	–	5218221	116.56	–	3459070	708.77	–	5218221	120.68	–
art9	4195351	802.08	–	7068510	125.58	–	4195351	809.40	–	7068510	125.33	–
pmed1	10765649	7786.70	–	12980089	859.97	–	10765649	7677.53	–	12980089	857.10	–
pmed2	11143314	7410.62	–	19962438	834.42	–	10984446	7273.61	–	19944050	819.15	–
pmed3	6524060	4148.35	–	9334140	464.82	–	6522592	4116.63	–	9332340	458.92	–
pmed4	16174587	9091.02	–	24213155	1031.35	–	16174587	9017.36	–	24213155	1023.01	–
pmed5	8687562	5509.46	–	10884353	604.88	–	8687776	5464.92	–	10884353	599.86	–
KROB150G	35348181	×	3.26	42047737	3354.30	–	35289997	×	3.24	41974612	3347.03	–
KROA150G	31579343	×	2.44	38836554	3146.35	–	31579345	×	2.43	38836554	3172.58	–
PR152G	2879248	5482.97	–	3759971	460.70	–	2796219	5290.21	–	3657358	450.09	–
RAT195G	39163633	×	15.94	×	3893.07	12.57	40806156	×	8.27	×	3779.84	9.93
KROB200G	27205462	×	4.64	38996783	4209.31	–	25044941	×	3.44	36820724	3896.58	–
KROA200G	18774302	×	3.08	24445462	2700.81	–	16949758	×	2.43	22853487	2503.93	–
TS225G	22020591	×	2.73	28041441	2878.48	–	22020758	×	2.73	28041441	2869.51	–
Average	11571423	8155.68	1.53	17825418	1181.93	0.60	11445531	8129.43	1.07	17636658	1150.83	0.47

Table 8: Maximum branch and bound tree size, running times and achieved gaps for $p = 5$ for the enumeration approach.

	enumeration											
Upper bound	IA						IA+DC					
Lower bound	HuffDisc			MidPoint			HuffDisc			MidPoint		
Network	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap
art1	761505	22.62	–	954269	14.45	–	850668	164.28	–	2884315	70.31	–
art2	833806	15.69	–	937528	11.62	–	1533922	324.22	–	1655465	102.99	–
art3	2218569	200.29	–	5398052	101.84	–	3472626	799.30	–	8899181	296.18	–
art4	2072133	210.15	–	2228330	81.31	–	5461464	1934.30	–	5461464	509.90	–
art5	13916512	1285.29	–	13942105	433.95	–	17259376	7172.07	–	18616932	1732.50	–
art6	17322753	1884.49	–	17534221	681.16	–	39108015	17341.40	–	×	763.47	83.84
art7	17115252	1830.39	–	21231438	685.81	–	21110879	11568.02	–	25765937	3450.06	–
art8	×	286.62	28.17	×	388.10	34.34	×	416.04	80.16	×	509.84	91.73
art9	17510258	3538.34	–	17525808	1663.44	–	×	443.31	99.23	×	590.92	99.23
pmed1	×	5610.09	34.84	×	5318.60	34.84	×	2075.53	73.16	×	2099.87	73.16
pmed2	×	2043.19	40.74	×	1877.88	40.74	×	1071.52	48.64	×	1089.60	48.64
pmed3	×	3195.31	32.60	×	5224.86	33.86	×	2865.89	53.84	×	2843.13	53.84
pmed4	×	1703.33	27.98	×	5537.69	28.46	×	3068.20	53.54	×	3033.74	53.54
pmed5	×	6864.04	35.36	×	1704.56	39.75	×	1733.62	52.25	×	1713.50	52.25
KROB150G	×	3326.66	31.34	×	3234.95	31.34	×	8541.38	26.04	×	8453.09	26.04
KROA150G	×	4453.25	28.90	×	4371.62	28.90	×	2252.59	31.32	×	2242.95	31.32
PR152G	17820377	×	0.17	17968869	×	0.20	×	13184.08	45.96	×	13009.80	45.96
RAT195G	×	2947.69	28.09	×	2885.96	28.09	×	3035.05	23.54	×	3019.45	23.54
KROB200G	×	12617.27	26.03	×	12737.00	26.03	×	6584.91	32.84	×	6603.30	32.84
KROA200G	×	3645.04	40.16	×	3671.02	40.16	×	8926.33	43.28	×	8845.65	43.28
TS225G	×	9715.15	26.58	×	9851.17	26.58	×	8832.73	37.19	×	8742.36	37.19
Average	61408150	4142.61	18.14	61796220	3908.43	18.73	70895092	4873.08	33.38	74442061.62	3320.12	37.92

Table 9: Maximum branch and bound tree size, running times and achieved gaps for $p = 5$ for the superset approach.

	superset											
Upper bound	IA						IA+DC					
Lower bound	HuffDisc			MidPoint			HuffDisc			MidPoint		
Network	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap	MaxList	time	gap
art1	1452057	310.71	–	1451329	58.63	–	1452056	307.24	–	1451328	59.09	–
art2	1994545	270.01	–	3696608	55.65	–	1994543	270.26	–	3696616	55.88	–
art3	36610591	6922.33	–	92922451	1502.35	–	36610591	6910.81	–	92922451	1490.92	–
art4	1228104	235.41	–	2984440	43.12	–	1228104	235.27	–	2984440	44.16	–
art5	20255341	4142.36	–	45869245	785.61	–	20255346	4144.18	–	45869251	799.97	–
art6	22539544	4639.56	–	29757712	727.59	–	22539545	4643.0	–	29757713	733.52	–
art7	25822697	5571.06	–	31479371	896.23	–	25822697	5580.98	–	31479371	897.07	–
art8	×	9939.57	4.60	×	1094.05	8.10	×	9937.81	4.60	×	1088.79	8.10
art9	×	10181.58	8.96	×	1058.50	24.56	×	10103.78	8.96	×	1049.59	24.56
pmed1	68661960	×	18.38	×	2901.09	19.27	68470552	×	18.40	×	2839.00	19.27
pmed2	70797011	×	19.05	×	2774.52	22.43	70727962	×	18.98	×	2738.64	22.35
pmed3	68662438	×	15.47	×	2865.02	19.57	68029779	×	15.54	×	2893.29	19.57
pmed4	70145377	×	23.65	×	2827.24	26.15	71493389	×	22.38	×	2727.99	27.76
pmed5	69754129	×	18.18	×	2756.32	17.42	69989853	×	18.08	×	2780.73	17.41
KROB150G	31327042	×	26.10	×	4152.59	24.74	36024130	×	21.09	×	3880.06	20.15
KROA150G	32557892	×	24.46	×	4095.34	21.55	35501051	×	20.57	×	3957.60	20.96
PR152G	34521048	×	6.92	×	4522.50	4.34	35281520	×	5.57	×	4524.22	4.13
RAT195G	20016605	×	48.90	×	5492.69	44.99	27657715	×	20.40	×	4631.51	24.53
KROB200G	17064415	×	29.03	×	5865.73	20.12	22428893	×	15.32	×	5085.60	15.25
KROA200G	16281856	×	29.79	×	5778.46	22.11	20716500	×	15.35	×	5165.29	14.37
TS225G	18788148	×	30.97	×	6013.59	23.07	21635793	×	21.94	×	5668.51	20.94
Average	39451467	14352.98	14.50	76579103	2679.37	14.21	40850477	14349.60	10.82	76579103	2529.12	12.35