Balancing the arrival times of users in a two-stage location problem

Maria Barbati[1,4] Giuseppe Bruno[2] Alfredo Marín[3]

[1] Portsmouth Business School, University of Portsmouth, UK
[2] Department of Industrial Engineering, University of Naples Federico II, Italy
[3] Departamento de Estadística e Investigación Operativa, Universidad de Murcia, Spain
[4] Corresponding author, maria.barbati@port.ac.uk, Richmond Building, Portland Street, Portsmouth PO13DE, UK

Abstract

There has been a number of facility location problems dealing with the introduction of the equity issue in the travel distances distribution. In this paper we analyze a new aspect of equity concerning the distribution of the arrival times of customers. Given a depot and a set of demand points generating flow which also represent potential locations, we consider a discrete two-stage location problem whose aim is to locate a given number of facilities and to allocate the demand points to a facility. We assume as objective the maximization of the minimum difference between two consecutive arrival times of flows to the depot through the patronized facility. This particular equity measure is introduced in order to reduce risks of congestion in the dynamic of flow arrivals at the common destination. The problem is described through two Integer Programming formulations. Computational results for solution methods based on both formulations are then shown and analyzed.

Keywords: Discrete Location, Equity, Integer Programming, Two-Stage

1 Introduction

Location Theory is a very active research area due to the increasing demand of decision-making support systems in several application fields. A location problem consists of positioning one or more facilities within a given space. The decision is made on the basis of an objective function, which usually concerns the minimization of costs or the maximization of benefits. However, in the last few decades a new class of location problems has arisen considering as objective equity, namely balancing the distances among users and facilities. After some initial proposals ([29], [23], [24]), [8] addressed the issue using taxes to redress benefit inequities. In [16], it was proposed a general framework for quantifying inequality and some axioms for the appropriateness of inequality measures were presented. The survey conducted by [28] listed a set of equality measures and introduced a framework for their classification. In addition, they indicated some characteristics that equality measures should have. A very comprehensive survey on the topic was performed in [15]

Afterwards several authors focused on the analysis of the properties and effects derived by using equality measures. For example, [21] analyzed properties of one of the proposed equality measures in the previous surveys, and more recently [13] conducted a specific analysis on one of the most widely used equality measures, the Gini coefficient, highlighting its behaviour for a single facility problem with demand points uniformly distributed.

Equality measures have been also adopted as objective function or as constraint in the formulation of many different facility location models. [30] showed algorithms for single facility location problems on networks using each time a different equality measure. [11] adopted as objective function the variance of total demand attracted from each facility while [22] exploited the concept of a particular formulation, called the ordered weighted averaging formulation, for defining a model which unifies and generalizes several inequality measures on several kinds of networks. [32] proposed the equitable dispersion problem that minimizes the range and the mean absolute deviation of the distribution of the distances. Lastly [3] formulated and analyzed a new version of a p-maxian problem with side constraints on dispersion, population, and equality constraints.

Also different application problems have been solved using models that involve equality measure as in [10] that found the best location of casualty collection points or [19] that formulated a bi-criterion model of perinatal facilities with a balanced loading of services provided by each facility. More recently [9] proposed a model for the location of hospitals in four U.S. states. Moreover, many times the equality measures are used in multi-objective models where the other measure is a typical efficiency measure. [12] proposed a multi-objective formulation for the Casualty Collection Points Location Problem defining 5 different objectives including the minimisation of the variance. [31] adopted as equality objective the sum of the absolute differences between all pairs of squared Euclidean distances from demand points to the facility and, as efficiency criteria, the sum of the squared Euclidean distances between demand points and facilities.

A new research field in the recent years is the definition of new forms of equity and the formulation of appropriate equality measures. [1] considered the problem of locating a given number of facilities on a continuous space so as to minimize the maximum demand faced by each facility subject to closest assignments and coverage constraints. [7] found a position for a given number of facilities in order to minimize the maximum total weight attracted by each facility on a network. A discrete location problem with a new form of equality measure called customers' envy was formulated in [17]. General equitable objective functions were also studied in [27], extending and improving a previous formulation developed in [26]. Recently, [25] introduced equity issues to balance the difference between the maximum and minimum number of customers allocated to each facility. A previous paper on load balancing is [4].

In this paper we address a new aspect of equity concerning the arrival times of customers in a discrete two-stage location problem. In the first stage demands have to be allocated to a facility, and in the second stage it must be transferred from the patronized facility to a fixed depot. The goal is to balance flows arriving at the depot in order to avoid congestion. The problem can occur in different practical contexts. For example, assume that some material (e.g., petrol) has to be sent from several origins (like refineries) to production plants (pump stations) where it is manufactured (pumped) and afterwards sent to a distribution center (storage tank), the depot, that receives the production from all the plants. Because of the limited operational capacity of the distribution center (pipes bottlenecks), it is necessary that consecutive arrivals of material to the depot be separated in time as much as possible. In order to describe the problem we consider as objective the maximization of the minimum difference between consecutive arrivals to the depot. In particular we propose two different Integer Programming models based on different approaches frequently used in the Discrete Location literature.

Our problem can be considered an alternative to the collection depots problem, proposed by [14] and [5], where they locate a new facility that is serving demand points. Here the demand has to travel from the facility to the demand point, then from the demand point to one of the collection depots, and back to the facility. The first difference is that in our case the demand, collected in the demand points, is only directed to the facility and then to the depot. The other important difference is in the type of objective function defined. [14] minimize the weighted sum of the transportation cost, while we propose a new equity objective. In addition, at least in this first approach, we did not consider the possibility of locating simultaneously facilities and depots as analysed by [6] and [33] in two different extensions of the collection depots problem. Another paper related to this one is [2], where the authors analyze the problem of optimal location of a set of facilities in the presence of stochastic demand and congestion.

The rest of this work is structured as follows. The technical details of the problem are given in Section 2. Two formulations are then introduced in Sections 3 and 4. Optimality cuts based on a known lower bound on the optimal value of the problem are incorporated to both formulations in Section 5, where a solution algorithm is also detailed. The paper ends with a comprehensive computational study (Section 6) and some conclusions.

2 The problem

Consider a model with the following elements: (i) a set $J = \{1, ..., M\}$ of demand points which also represent potential locations for the facilities, (ii) a depot sited at point 0, (iii) a fixed number p of facilities (plants) to be located, and (iv) an $M \times (M+1)$ distances matrix $d = (d_{ij})$ that represent either the distance (cost, time) between a demand point situated in i and the facility in j (if $i, j \in J$) or the distance from the facility in i to the depot (when $j = 0$). Here we assume $d_{ii} = 0 \ \forall i \in J$, $d_{ij} > 0 \ \forall i, j \in J$ and $d_{i0} \geq 0 \ \forall i \in J$. We also assume that d-values satisfy the triangle inequality.

The aim is to locate p plants among the M candidates and to allocate the remaining $M - p$ demand points to a plant (which is not necessarily the closest one). Let a_i be the plant to which demand point i has been allocated (assuming $a_i = i$ if i is a plant itself). Then, M distances from demand points to the depot will be obtained as $\delta_i := d_{ia_i} + d_{a_i0} \ \forall i \in M$. We call these *travel distances.* The goal is to maximize the minimum absolute difference between any two values in the vector $(\delta_1, \ldots, \delta_M)$.

Note that, when d represents times, the model can be easily extended by considering an additional processing time in the plants. Also note that, when two demand points are allocated to the same plant, the optimal value of the problem will be lower than or equal to the absolute difference between the distances from these two points to the assigned plant. Since we are maximizing the objective, arrival times to the plants are also spread out by the optimal solution.

A possible alternative is to force closest allocation of demand points to facilities. To this end, Closest Allocation Constraints (CAC) have to be added to the formulations introduced in the next sections, drastically worsening the solutions of the problem. CAC in discrete location have been deeply studied in [18], where a complete classification of all possibilities previously considered in the literature was carried out.

In order to formulate the problem as an Integer Programming model it is important to note that maximizing the minimum absolute difference between any pair of travel distances (associated with different demand points) is equivalent to maximizing the minimum difference between consecutive sorted travel distances. The drawback when formulating this problem is to identify which travel distance is greater than or equal to the other, that is to say, absolute values of differences between travel distances have to be considered. It is also necessary to force plants to be allocated to themselves, that is to say, avoiding the flow generated in a point i chosen as a plant to use a route other than the direct route $i \to 0$.

From the distances matrix (d_{ij}) we define the matrix (D_{ij}) which measures the distances from point i to the depot through plant j, that is to say,

$$D_{ij} := d_{ij} + d_{j0} \qquad \forall i, j \in J.$$

We name the entries of (D_{ij}) potential travel distances. From now on, we will call this problem the Balancing Two-Stage Location Problem (BTLP).

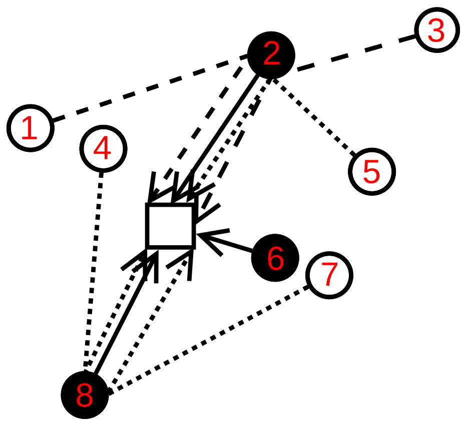

Figure 1: Example. Square is the depot, black circles are plants

Example 1. *In Figure 1 we show an example with $M = 8$ and $p = 3$. We have eight demand points and potential facilities, numbered by 1 to 8; among these, the three plants have been represented with black filled circles (2, 6 and 8). The depot has been depicted with a square. The demand points 2, 6 and 8 are allocated to themselves. The demand points 1, 3 and 5 are allocated to facility 2, while the demand points 4 and 7 are allocated to 8. The travel distance is the sum of the distance from the demand point to the corresponding facility plus the distance from the patronized facility to the depot; these are indicated for the plants with continuous arrows and for the demand points which are not plants with a segmented arrow. The value of the objective function can be determined comparing the travel distance of each demand point with those of the others. Among all these differences the minimum will be the objective value to be maximized.* $\triangle$

3 Classical style formulation for BTLP

In this section what we mean with classical style is a formulation based on the variables commonly used when formulating discrete location problems. Then, allocation decisions are represented through the following x-variables:

$$x_{ij} = \begin{cases} 1 & \text{if demand point } i \text{ is allocated to facility } j \\ 0 & \text{otherwise} \end{cases} \qquad \forall i, j \in J : \ i \neq j,$$

and location decisions are represented with

$$x_{jj} = \begin{cases} 1 & \text{if a facility is located at point } j \\ 0 & \text{otherwise} \end{cases} \qquad \forall j \in J.$$

Variable z will represent the minimum difference between travel distances, i.e., the objective function to be maximized. The proposed formulation is

$$
\begin{aligned}
&\text{(CBTLP)} \\
&\max\ z \\
&\text{s.t.} \sum_{j\in J} x_{ij} = 1 && \forall i \in J, \quad (1)\\
&x_{ij} \le x_{jj} && \forall i,j \in J, i \neq j, \quad (2)\\
&\sum_{j\in J} x_{jj} = p && (3)\\
&z \le \sum_{\ell\in J} |D_{ia} - D_{j\ell}| x_{j\ell} + (z_{UB} - \min_{\ell\in J} |D_{ia} - D_{j\ell}|)(1 - x_{ia}) && \forall i,j,a \in J: \ i \neq j, \quad (4)\\
&x_{ij} \in \{0,1\} && \forall i,j \in J. \quad (5)
\end{aligned}
$$

Constraints (1) ensure that all the demand points are allocated and also that plants are self-allocated. Constraints (2) guarantee that a point receives allocation only if it is a plant. Constraint (3) fixes the number of plants to p. Constraints (4) are used to obtain the value of the objective function. In particular, due to constraints (1), for any $j \in J$ the first addend in the right hand side of (4) will take the value $|D_{ia} - D_{j\ell}|$ for that plant ℓ to which j is allocated. If, additionally, site i is allocated to plant a, z will be upperly bounded by $|D_{ia} - D_{j\ell}|$, as wished. Here, the term $\min_{\ell\in J} |D_{ia} - D_{j\ell}|$ is used to tighten the constraint, since the first addend in the right hand side of (4) will take at least this value. Otherwise z will be bounded by z_{UB}, a known upper bound on the optimal value of the problem, plus a non negative amount.

This formulation has M^2 binary variables and $M^3 + 1$ constraints (excluding binarity constraints). It is well known that in many other discrete location problems, it suffices with forcing the binarity of x_{jj} $\forall j \in J$, reducing in this way the complexity of some resolution methods. This is not the case with formulation (CBTLP), as we show in the following example.

Example 2. *Consider an instance with $M = 4$, four points in the plane located at $(2,2)$, $(1,1)$, $(1,4)$ and $(5,0)$, respectively. The depot is at $(2,3)$. For ease of computation, we use the Manhattan distance $d((i_1,i_2),(j_1,j_2)) := |i_1 - j_1| + |i_2 - j_2|$. The optimal solution to the instance with $p = 2$ is to locate facilities in points 1 and 4, and to allocate 2 and 3 to 4. The corresponding travel distances are 1, 11, 14 and 6, giving an optimal value of 3. Relaxing in (CBTLP) the integrity of x_{ij} with $i \neq j$, the optimal solution is*

$$
x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 1 \end{pmatrix},
$$

i.e., exactly the same solution except that each half of point 3 is assigned to a different facility. This fractional solution gives an objective value of 5. △

4 Reduced formulation for BTLP

In order to build the reduced formulation, some preprocessing is needed. First, given any point i, all its different potential travel distances are sorted in increasing order:

$$
0 \le D^i_{(1)} < \cdots < D^i_{(g_i)} := \max_{j\in J}\{D_{ij}\}.
$$

The corresponding sets of indexes are named $G_i := \{1, \ldots, g_i\}$, $i \in J$.

For this second formulation, allocation decisions are represented through y-variables as follows ($i \in J, k \in G_i$):

$$y_{ik} = \begin{cases} 1 & \text{if the travel distance for point } i \text{ is } D^i_{(k)}, \\ 0 & \text{otherwise.} \end{cases}$$

Then the reduced formulation is

(RBTLP)

$$\max z$$

$$\text{s.t.} \sum_{k \in G_i} y_{ik} = 1 \qquad \forall i \in J, \tag{6}$$

$$y_{ik} \leq \sum_{j \in J:\ D_{ij} = D^i_{(k)}} y_{j1} \qquad \forall i \in J,\ \forall k \in G_i, \tag{7}$$

$$\sum_{j \in J} y_{j1} = p \tag{8}$$

$$z \leq \sum_{\ell \in G_b} |D^a_{(k)} - D^b_{(\ell)}| y_{b\ell} + (z_{UB} - \min_{\ell \in G_b} |D^a_{(k)} - D^b_{(\ell)}|)(1 - y_{ak}) \qquad \forall a < b \in J,\ \forall k \in G_a, \tag{9}$$

$$y_{ik} \in \{0, 1\} \qquad \forall i \in J, \forall k \in G_i. \tag{10}$$

Constraints (6) say that demand point i has to be allocated at some given distance $D^i_{(k)}$. Constraints (7) force y_{ik} to take value 0 if no plant is opened at a distance equal to $D^i_{(k)}$. Constraint (8) ensures that exactly p plants are opened (note that, due to triangle inequality, y_{i1} is equal to 1 if and only if the demand point i is allocated with a distance $D^i_{(1)}$, i.e., allocated to itself). Constraints (9) force the objective function z to assume the correct value. Binarity constraints (10) close the formulation.

The number of variables and constraints in (RBTLP) depends on the number of different potential travel distances. In the worst case (all potential travel distances from each point different), this formulation is exactly equal to (CBTLP) and contain the same number of variables and constraints. In case of ties, (RBTLP) is actually a reduced formulation, since the sum of two x-variables $x_{ij_1} + x_{ij_2}$ with $D_{ij_1} = D_{ij_2}$ is reduced to a single y-variable y_{ik} with $D^i_{(k)} = D_{ij_1} = D_{ij_2}$.

Example 3. *Consider again the instance proposed in Example 2, four points in the plane located at* $(2, 2)$, $(1, 1)$, $(1, 4)$ *and* $(5, 0)$, *respectively, the depot at* $(2, 3)$ *and Manhattan distance. The distances matrix here is given by*

$$\begin{pmatrix} 0 & 1 & 3 & 2 & 6 \\ 1 & 0 & 2 & 3 & 5 \\ 3 & 2 & 0 & 3 & 5 \\ 2 & 3 & 3 & 0 & 8 \\ 6 & 5 & 5 & 8 & 0 \end{pmatrix},$$

and the potential travel distances are

$$\begin{pmatrix} 1 & 5 & 5 & 11 \\ 3 & 3 & 5 & 11 \\ 4 & 6 & 2 & 14 \\ 6 & 8 & 10 & 6 \end{pmatrix}.$$

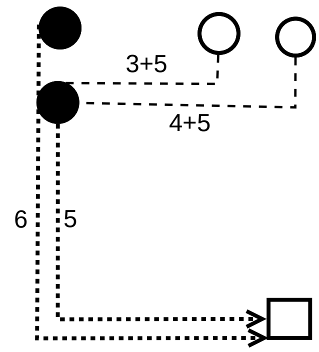

Figure 2: Optimal solution to instance of Example 4. Square is the depot, black circles are plants

Associated with $i = 1$ *we have sorted potential travel distances*

$$D^1_{(1)} = 1 < D^1_{(2)} = 5 < D^1_{(3)} = 11.$$

Then y_{12} *will take value 1 if point 1 is allocated to some plant in such a way that its travel distance is* $D^1_{(2)} = 5$*, i.e., if 1 is allocated to plants 2 or 3.* △

Furthermore, the reduction in the size of the formulation can also provide us with a better linear relaxation upper bound, as can be seen in the following example.

Example 4. *Consider an instance with four points located in the plane, with coordinates (1,4), (3,4), (4,4) and (1,3). The depot is located at (4,1), p is fixed to 2 and Manhattan distance is considered. Then, the potential travel distances are*

$$\begin{pmatrix} 6 & 6 & 6 & 6 \\ 8 & 4 & 4 & 8 \\ 9 & 5 & 3 & 9 \\ 7 & 7 & 7 & 5 \end{pmatrix}.$$

The optimal solution to this instance (Figure 2) gives optimal value 1. Here, $g_i = (1, 2, 3, 2)$*, and constraints (6) of (RBTLP) are in the shape of*

$$\begin{aligned} &y_{11} = 1 \\ &y_{21} + y_{22} = 1 \\ &y_{31} + y_{32} + y_{33} = 1 \\ &y_{41} + y_{42} = 1. \end{aligned}$$

Taking the constraint in family (9) with $a = 1$*,* $b = 4$ *and* $k = 1$ *it holds*

$$z \leq y_{41} + y_{42} + (z_{UB} - 1)(1 - y_{11}),$$

thus $z \leq 1$*. On the other hand,*

$$x = \begin{pmatrix} 1/4 & 1/4 & 1/4 & 1/4 \\ 0 & 1/4 & 0 & 3/4 \\ 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

is a feasible solution to the linear relaxation of (CBTLP) that gives an objective value of 1.75. △

Encouraged by the good results obtained when applying *ordered formulations* to discrete location problems (see e.g. [20]), we built still a third formulation of BTLP using the variables

$$y_{ik} = \begin{cases} 1 & \text{if the travel distance for point } i \text{ is } D^i_{(k)}, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$w_{jk} = \begin{cases} 1 & \text{if the } (n-j+1)\text{-th travel distance is less than or equal to the } k\text{-th } d\text{-distance} \\ & \text{and the } (n-j)\text{-th travel distance is strictly greater than the } k\text{-th } d\text{-distance,} \\ 0 & \text{otherwise,} \end{cases}$$

defined over the adequate sets of indexes. Even though we strove to make this formulation competitive, implementing a preprocessing phase and several families of valid inequalities, the computational results never were good enough to make this third formulation competitive. Then we gave up the idea of including it in the article. Ordered formulations contain more variables than its *unordered* counterparts, and need to take a large advantage of the problem structure to be competitive, what was not the case with this problem.

5 Bounds, optimality cuts and solution strategy

An upper bound on the optimal value of the problem is needed for the formulations. We get one in the following way:

$$z_{UB} := (\max_{i,j\in J}\{D_{ij}\} - \min_{i\in J}\{D_{ii}\})/(M-1).$$

Note that z_{UB} corresponds with the perfect situation where all sorted travel distances differ in the same amount.

Assuming we know a lower bound z_{LB} on the optimal value of the problem (possibly associated with a feasible solution), some new inequalities can be added to the formulations. Take two different points, $i, j \in J$, and two potential travel distances associated to them, $D^i_{(k)}$ and $D^j_{(\ell)}$. It is obvious that, if $|D^i_{(k)} - D^j_{(\ell)}|$ is lower than the lower bound z_{LB}, allocating i at a plant in such a way that the travel distance $D^i_{(k)}$ is obtained and, at the same time, allocating j at a plant such that the corresponding travel distance is $D^j_{(\ell)}$, will give a worse feasible solution. Then, it can be concluded that variables y_{ik} and $y_{j\ell}$ will not take value 1 at the same time in any optimal solution, that is to say,

$$y_{ik} + y_{j\ell} \leq 1 \tag{11}$$

will be satisfied by any optimal solution to (RBTLP). Cutting planes that remove part of the feasible region as long as at least one integer optimal solution remains intact are typically referred as optimality cuts. The addition of these cuts can reduce the search space and improve the linear relaxation upper bound. Inequality (11) is a set packing constraint, what do it very appropriate for the standard preprocess that commercial solvers carry out before starting the search. Several binary variables can be added up and bounded by 1 if they correspond with travel distances which are pairwise lower than the available lower bound or correspond with the same origin. Moreover, commercial solvers can get very good feasible solutions (based on rounding heuristics) that can be used to generate constraints of type (11). Assuming we are considering formulation

Step 1. Get upper bound $z_{UB} := (\max_{i,j\in J}\{D_{ij}\} - \min_{i\in J}\{D_{ii}\})/(M-1)$,
fix lower bound $z_{LB} := 0$ and a time limit $T > 0$.

Step 2. Build formulation (12) and run the preprocessing and heuristic phases of the solver.
Let z_1 be the objective value associated with the best feasible solution the solver obtained.
If $z_1 > 1.1 z_{LB}$, do $z_{LB} := z_1$ and go back to Step 2.

Step 3. Run the branch-and-bound phase of the solver. If a new feasible solution with objective value $z_2 > 1.1 z_{LB}$ is obtained before time T is reached, stop the search, update $z_{LB} := z_2$ and go back to Step 2. Otherwise, if after time T the lower bound has not been improved in at least 10%, let the branch-and-bound algorithm run until the end.

Figure 3: Solution algorithm

(RBTLP), and we have found a lower bound z_{LB}, the enforced formulation we will use is then

$$\text{(ERBTLP)} \quad \max z \tag{12}$$
$$\text{s.t. } (6),\ (7),\ (8),\ (9),\ (10)$$
$$y_{ik} + y_{j\ell} \le 1 \qquad \forall i < j \in J,\ \forall k \in G_i,\ \forall \ell \in G_j :\ |D^i_{(k)} - D^j_{(\ell)}| < z_{LB}. \tag{13}$$

In the case of (CBTLP), two variables x_{ia} and x_{jb} will not take simultaneously value 1 when $|D_{ia} - D_{jb}| < z_{LB}$. Therefore we also consider the enforcement of the classical formulation in the shape of

$$\text{(ECBTLP)} \quad \max z \tag{14}$$
$$\text{s.t. } (1),\ (2),\ (3),\ (4),\ (5)$$
$$x_{ia} + x_{jb} \le 1 \qquad \forall i < j \in J, a, b \in J :\ |D_{ia} - D_{jb}| < z_{LB}. \tag{15}$$

We observed that the quality of the lower bound is crucial for the efficient resolution of the problem, since a powerful set of constraints (13)/(15) largely reduce the search space and, consequently, the computational time. On the other hand, after adding constraints (13)/(15) to the corresponding formulation, the lower bound can be significantly improved. Therefore we designed a solution method that starts with one of the formulations (CBTLP) or (RBTLP), gets a lower bound –simply the bound given by the best feasible solution found by the solver during the preprocessing phase–, generates optimality cuts, adds them to the formulation and starts again. Many times, much better feasible solutions are generated in the first nodes of the branching tree, what do it very advisable to re-start the process with additional cuts. A trade-off between the time needed by the branch-and-bound algorithm to find the bound and the time saved with the improvement of the formulation due to the new inequalities has to be evaluated. After some trials, an algorithm was designed whose details are given in Figure 3.

Note that a time limit T has to be fixed in the algorithm. Note also that the number of potential inequalities in family (13) is large (up to $\frac{M^3(M-1)}{2}$), and typically about 10% of them satisfy the condition required for adding them to the formulation. Nevertheless, they can be easily combined in a smaller number of tighter constraints. For instance, consider $i, j \in J$, $k \in G_i$ and let J_{ijk} be a set with cardinality c defined as

$$J_{ijk} := \{\ell \in G_j :\ |D^i_{(k)} - D^j_{(\ell)}| < z_{LB}\}.$$

Then, instead of c cuts of type (13), a single tighter cut in the shape of

$$y_{ik} + \sum_{a \in J_{ijk}} y_{ja} \leq 1$$

can be added to the formulation. However, it seems that the solver is more able to work with the disaggregated constraints, producing better feasible solutions, and also reduces the number of cuts (13) during the preprocessing phase. For this reason we discarded the possibility of using aggregated constraints.

6 Computational study

Computational experiments were carried out in order to check the performance of both formulations in terms of time, with and without optimality cuts. All the results of the experiment were obtained using a Intel Core 2 Quad CPU Q9300, 2.50GHz × 4, with 3 GB of RAM memory, running linux Ubuntu 12.04. The solver used was Xpress mosel 32-bit v3.4.3, with the default parameters.

Since, to the best of our knowledge, this is the first paper in studying BTLP, we could not compare our results with other benchmarks and we had to generate our own instances for the experiment. Demand points were randomly generated inside a square and Euclidean distances were first considered. We tested the solution methods on a testbed of 81 instances, 3 for each possible position of the depot (center, corner, random) and for each combination of M in {20, 25, 30} with different values of p (2, 4, 6). Note that, when using Euclidean distances, it is very unlikely to obtain ties between potential travel distances. In such a case both formulations, (CBTLP) and (RBTLP) are the same. What we compared in this case is one of the formulations ((RBTLP), in fact) with the solution method given in Figure 3. First we used the solution method to exactly solve the instances. The time limit T used to improve the lower bounds were fixed to 100 seconds in the smaller instances ($M \leq 25$) and 200 seconds in the larger instances ($M = 30$). The results are shown in Tables 1, 2 and 3, grouped by the size of the instance ($M = 20$, 25 and 30, respectively). Each line corresponds to the average of three instances. **Type** indicates if the depot is located in the center (CE), corner (CO) or at a random position (RA). The number of plants, **p**, is 2, 4 or 6. **LB** is the last lower bound that was used to re-start the solver, and **RS** is the number of times the solver was re-started. The optimal value is given in column **OPT**, and the number of nodes of the branching tree needed to solve the instances (in the last run of the solver) is in column **bbn**. The number of optimality cuts added in the last run of the solver is given in column **cuts**, and the overall time in seconds of all runs is in column **time**.

The computational results indicate that the difficulty of the instances grows fast with the number of points. The computational times required for solving the instances vary from seconds in the instances with 20 points, through minutes in the case of instances with 25 points, to hours for the 30-points instances. The difficulty also increases when the number of plants increases. The location of the depot does not seem to influence the results, and the bounds are tight in general. Several thousands of optimality cuts are usually added to the formulation in the first part of the algorithm, and the number of times the solver was re-started varied between 1 and 4. The time devoted to this first part of the algorithm was generally tiny in comparison with the total computational time. We observed that the worst results (larger computational times) were obtained when the lower bound was weak. In some cases, after the time limit T was reached, a new lower bound was obtained that clearly would help to reduce the total time. The time limit of 200 seconds for the large instances was correct for the easy instances ($p = 2$, say), but re-starting

Type	p	LB	RS	OPT	bbn	cuts	time
CE	2	7.4	1.7	7.6	73	3177	28
CE	4	10.1	1.7	11.7	3857	6578	59
CE	6	11.2	1.3	12.2	948	7275	20
CO	2	10.6	1.0	11.0	13	5494	17
CO	4	13.3	1.7	15.6	2661	6817	39
CO	6	14.7	1.0	16.2	1728	7463	27
RA	2	8.2	1.7	8.5	57	4785	21
RA	4	10.7	1.0	13.0	3209	62412	38
RA	6	12.2	1.0	14.2	566	71654	14

Table 1: Computational results for the solution algorithm, 20 points

Type	p	LB	RS	OPT	bbn	cuts	time
CE	2	4.7	1.0	5.7	195	7640	50
CE	4	8.5	1.7	9.5	18889	13651	519
CE	6	9.4	2.0	10.1	20324	15107	329
CO	2	6.7	2.0	6.7	66	8639	66
CO	4	11.2	2.0	12.1	7620	14345	257
CO	6	12.2	2.3	13.2	876	16340	106
RA	2	6.6	1.7	6.6	63	9707	57
RA	4	9.5	2.7	10.1	12034	14144	398
RA	6	10.5	2.0	11.1	48492	15509	874

Table 2: Computational results for the solution algorithm, 25 points

Type	p	LB	RS	OPT	bbn	cuts	time
CE	2	4.6	2.0	5.0	96	15281	137
CE	4	7.6	3.7	8.3	46403	25481	3012
CE	6	7.9	2.0	9.0	383075	26412	14383
CO	2	5.7	3.0	5.8	70	15935	122
CO	4	9.0	2.0	10.3	64995	25088	3444
CO	6	9.9	2.3	11.3	280557	27276	8878
RA	2	4.9	2.0	5.1	141	14817	138
RA	4	7.9	2.0	9.1	88299	23921	4255
RA	6	8.7	2.3	10.1	218273	26801	9241

Table 3: Computational results for the solution algorithm, 30 points

			Algorithm		Solver	
Type	p	OPT	bbn	time	bbn	time
CE	2	7.6	73	28	1167	16
CE	4	11.7	3857	59	46322	133
CE	6	12.2	948	20	15824	29
CO	2	11.0	13	17	343	17
CO	4	15.6	2661	39	29361	96
CO	6	16.2	1728	27	39599	94
RA	2	8.5	57	21	1027	17
RA	4	13.0	3209	38	44790	123
RA	6	14.2	566	14	8721	29

Table 4: Computational results for the reduced formulation, 20 points

			Algorithm		Solver	
Type	p	OPT	bbn	time	bbn	time
CE	2	5.7	195	50	2670	50
CE	4	9.5	18889	519	(2)	2191
CE	6	10.1	20324	329	(3)	1647
CO	2	6.7	66	66	2587	60
CO	4	12.1	7620	257	(2)	1028
CO	6	13.2	876	106	(3)	528
RA	2	6.6	63	57	1932	59
RA	4	10.1	12034	398	(2)	1973
RA	6	11.1	48492	874	(3)	4368

Table 5: Computational results for the reduced formulation, 25 points

the search after 200 seconds, when the new lower bound is much better than the current one, could be carried out to significantly reduce the running times of the difficult instances.

In a second stage we run the solver without optimality cuts. Since the times were so large, we only checked the small instances. Furthermore, we stopped the run when the time was five times the time used by the algorithm. The results are shown in Tables 4 ($M = 20$) and 5 ($M = 25$). To make the comparison easier, we copied columns **bbn** and **time** from the previous study, under the head **Algorithm**. Under the head **Solver** we show the number of nodes of the branching tree and the computational times of the solver when applied to the difficult instances. In the case of Table 4, $M = 20$, all instances could be solved within the time limit (that is to say, no instance multiplied by five the running time of the algorithm). Instances with $p = 2$ needed similar or even smaller times than the algorithm, since they are extremely easy and the re-starting procedure unnecessarily added some time to the process. In the case of more difficult instances ($p = 4, 6$), there is a significant saving of time and branching nodes when the optimality cuts are used. But the impressive results were obtained with the 25-points instances (Table 5). Again for $p = 2$ the times were similar, but for larger values of p, the times needed by the solver were several orders of magnitude larger than the times needed by the algorithm. Since we observed that these times were huge and, in some cases, produced an *out of memory* error, we decided to stop the search after the solver reached five times the times of the algorithm. Then, in column **time** of Table 5, the times are calculated with this limit, and we indicated between brackets in column **bbn** the number of instances that were stopped without being solved. Instances of size $M = 30$ were beyond the powers of the formulation.

					Classical							Reduced						
		OPT			bounds	bbn			time			bounds	bbn			time		
M	p	min	avg	max	avg	min	avg	max	min	avg	max	avg	min	avg	max	min	avg	max
20	4	1	1.9	2	(1.2,2.8)	515	12164	64447	21	60	245	(1.2,2.8)	1	4048	19071	3	13	43
20	6	2	2	2	(1.4,2.8)	105	3408	10327	15	29	49	(1.7,2.7)	0	367	1409	1	3	10
25	4	1	1.4	2	(1.1,2.4)	1	75743	340032	16	1868	6783	(1.0,2.3)	1	26056	140939	3	103	549
25	6	1	1.6	2	(1.2,2.4)	1	64703	272290	13	996	3687	(1.1,2.4)	1	20315	97400	2	64	210

Table 6: Computational results for both formulations, instances with ties

		Classical			Reduced		
M	p	bbn	time	cuts	bbn	time	cuts
20	4	4659	82	5207	1673	18	881
20	6	1	23	6204	80	3	1420
25	4	33452	713	6445	7142	107	1059
25	6	76326	1363	3035	7670	85	1932

Table 7: Computational results of the algorithm with both formulations, instances with ties

Besides showing the advantage of using the optimality cuts for solving BTLP, the aim of the computational study was to compare formulations (CBTLP) and (RBTLP) when there are ties between potential travel distances. To this aim, we generated a different set of instances, randomly locating integer points inside a square of size 19×19 (pairs in the set $\{1,\ldots,20\} \times \{1,\ldots,20\}$) with a depot in position $(4,4)$, and calculated the Manhattan distance between each pair of points, always an integer number. Notice that we did not pass any parameter to Xpress to indicate the integrity of the optimal value (thus Xpress does not take into account that a duality gap less than one implies optimality). The typical optimal values of these instances were 1 and 2. Ten instances with $M = 20$ and ten other instances with $M = 25$ were generated and solved with both formulations (without optimality cuts). The results are given in Table 6, were each row refers to all the ten instances. The number of plants was fixed in 4 and 6. **min**, **avg** and **max** means the minimum, average and maximum value of the ten instances, respectively. **OPT** if the optimal value, **bounds** are the bounds produced by the solver after the linear relaxation, **bbn** are again the nodes of the branching tree and **time** the computational time in seconds. The results are self-explanatory, even noting that two of the instances run out of memory with the classical formulation and were not included in the calculations. There is a strong reduction of computational times and number of nodes of the branching tree when the formulation we use is the reduced one. The reduction is more significant when the size of the instance is 25 than when it is 20. Still we compared both formulations with the addition of optimality cuts, using the algorithm of Figure 3 in the case of (ERBTLP) and a similar approach in the case of (ECBTLP). Results are presented in Table 7 (averaged for ten instances). When compared with the results in Table 6, we observe that the computational times of the reduced formulation and the algorithm are very similar, although the number of nodes is largely reduced. Some instances benefit from the optimality cuts, whereas some others need more time due to the larger size of the formulation. The reason is that the bounds the solver produces are always 1 or 2. A lower bound of 1 does not help solving the instance when the optimal value is 2. On the contrary, the algorithm based on the classical formulation largely improved the results of the plain formulation when $M = 25$. For the first time in the study, some of the instances were solved in less time with the classical approach than with the reduced approach. These cases do not suffice to make the classical formulation more attractive than the reduced one, as can be seen in the averages of

Table 7.

7 Conclusions and further research

In this paper we propose a new discrete location problem where plants must be located in such a way that arrivals of flows generated from demand points through the plants to a fixed depot are as separated in time as possible.

Starting with a classical style formulation, where the constraints have been tightened as much as possible, we take advantage of the ties in the potential travel distances (from a given demand point) to produce a reduced formulation (containing less variables) that is much more efficient from a computational point of view.

Then we use a lower bound on the optimal value of the problem in both formulations to produce a large number of optimality cuts (in a set packing fashion). Optimality cuts increase the lower bound and then we iteratively get new cuts and lower bounds to improve the formulations as far as possible. This procedure results in a drastic reduction of the computational times.

Nevertheless, the problem proves to be difficult and requires further effort to be effectively solved when the number of points is greater than 30. Heuristic algorithms will be also of great interest to approximately solve large instances and are in our scope of interest.

8 Acknowledgements

Alfredo Marín acknowledges that research reported here was partially supported by *Ministerio de Economía y Competitividad*, project MTM2012-36163-C06-04, and Fundación Séneca project 08716/PI/08.

References

[1] Baron, O., Berman, O., Krass, D., & Wang, Q. (2007). The equitable location problem on the plane. European Journal of Operational Research, 183(2), 578–590.

[2] Baron, O., Berman, O., & Krass, D. (2008). Facility location with stochastic demand and constraints on waiting time. Manufacturing & Service Operations Management, 10(3), 484–505.

[3] Batta, R., Lejeune, M., & Prasad, S. (2014). Public facility location using dispersion, population, and equity criteria. European Journal of Operational Research, 234(3), 819–829.

[4] Berman, O., Drezner, Z., Tamir, A., & Wesolowsky, G.O. (2009). Optimal location with equitable loads. Annals of Operations Research, 167(1), 307–325.

[5] Berman, O., Drezner, Z., & Wesolowsky, G.O. (2002). The collection depots location problem on networks. Naval Research Logistics, 49(1), 15–24.

[6] Berman, O., & Huang, R. (2004). Minisum collection depots location problem with multiple facilities on a network. Journal of the Operational Research Society, 55(7), 769–779.

[7] Berman, O., & Huang, R. (2008). The minimum weighted covering location problem with distance constraints. Computers & Operations Research, 35(2), 356–372.

[8] Berman, O., & Kaplan, E.H. (1990). Equity maximizing facility location schemes. Transportation Science, 24(2), 137–144.

[9] Burkey, M.L., Bhadury, J. & Eiselt, H.A. (2012). A location-based comparison of health care services in four U.S. states with efficiency and equity. Socio-Economic Planning Sciences, 46(2), 157–163.

[10] Drezner, T. (2004). Location of casualty collection points. Environment and Planning C: Government and Policy, 22(6), 899–912.

[11] Drezner, T., & Drezner, Z. (2006). Multiple Facilities Location in the Plane Using the Gravity Model. Geographical Analysis, 38(4), 391–406.

[12] Drezner, T., Drezner, Z., & Salhi, S. (2006). A multi-objective heuristic approach for the casualty collection points location problem. Journal of the Operational Research Society, 57(6), 727–734.

[13] Drezner, T., Drezner, Z., & Guyse, J. (2009). Equitable service by a facility: Minimizing the Gini coefficient. Computers & Operations Research, 36(12), 3240–3246.

[14] Drezner, Z., Wesolowsky, G.O. (2001). On the collection depots location problem. European Journal of Operational Research, 130(3), 510–518.

[15] Eiselt, H.A., & Laporte, G. (1995). Objectives in location problems. In Z. Drezner (Ed.), Facility Location: A Survey of Applications and Methods (pp.151–180). New York: Springer-Verlag.

[16] Erkut, E. (1993). Inequality measures for location problems. Location Science, 1(3), 199–217.

[17] Espejo, I., Marín, A., Puerto, J., & Rodríguez-Chía, A.M. (2009). A comparison of formulations and solution methods for the minimum-envy location problem. Computers & Operations Research, 36(6), 1966–1981.

[18] Espejo, I., Marín, A., & Rodríguez-Chía, A.M. (2012). Closest assignment constraints in discrete location problems. European Journal of Operational Research, 219(1), 49–58.

[19] Galvão, R.D., Acosta Espejo, L.G., Boffey, B., & Yates, D. (2006). Load balancing and capacity constraints in a hierarchical location model. European Journal of Operational Research, 172(2), 631-646.

[20] García, S., Labbé, M., & Marín, A. (2011). Solving large p-median problems with a radius formulation. INFORMS Journal on Computing, 23(4), 546-556.

[21] López-de-los-Mozos, M.C., & Mesa, J.A. (2001). The maximum absolute deviation measure in location problems on networks. European Journal of Operational Research, 135(1), 184–194.

[22] López-de-los-Mozos, M.C., Mesa, J.A., & Puerto, J. (2008). A generalized model of equality measures in network location problems. Computers & Operations Research, 35(3), 651–660.

[23] Maimon, O. (1986). The variance equity measure in locational decision theory. Annals of Operations Research, 6(5), 147–160.

[24] Maimon, O. (1988). An algorithm for the Lorenz measure in locational decisions on trees. Journal of Algorithms, 9(4), 583–596.

[25] Marín, A. (2011). The discrete facility location problem with balanced allocation of customers. European Journal of Operational Research , 210(1), 27–38.

[26] Marín, A., Nickel, S., Puerto, J., & Velten, S. (2009). A flexible model and efficient solution strategies for discrete location problems. Discrete Applied Mathematics, 157(5), 1128–1145.

[27] Marín, A., Nickel, S., & Velten, S. (2010). An extended covering model for flexible discrete and equity location problems. Mathematical Methods of Operations Research, 71(1), 125–163.

[28] Marsh, M.T., & Schilling, D.A. (1994). Equity measurement in facility location analysis: A review and framework. European Journal of Operational Research, 74(1), 1–17.

[29] McAllister, D.M. (1976). Equity and efficiency in public facility location. Geographical Analysis, 8(1), 47–63.

[30] Mesa, J.A., Puerto, J., & Tamir, A. (2003). Improved algorithms for several network location problems with equality measures. Discrete Applied Mathematics, 130(3), 437–448.

[31] Ohsawa, Y., Ozaki, N., & Plastria, F. (2008). Equity-efficiency bicriteria location with squared Euclidean distances. Operations Research, 56(1), 79–87.

[32] Prokopyev, O.A., Kong, N., & Martínez-Torres, D.L. (2009). The equitable dispersion problem. European Journal of Operational Research, 197(1), 59–67.

[33] Tamir, A., Halman, N. (2005). One-way and round-trip center location problems. Discrete Optimization, 2(2), 168–184.