A New Compact Formulation for Discrete p-Dispersion

David Sayah[*,a], Stefan Irnich[a]

[a] *Chair of Logistics Management, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*

Abstract

This paper addresses the discrete p-dispersion problem (PDP) which is about selecting p facilities from a given set of candidates in such a way that the minimum distance between selected facilities is maximized. We propose a new compact formulation for this problem. In addition, we discuss two simple enhancements of the new formulation: Simple bounds on the optimal distance can be exploited to reduce the size and to increase the tightness of the model at a relatively low cost of additional computation time. Moreover, the new formulation can be further strengthened by adding valid inequalities. We present a computational study carried out over a set of large-scale test instances in order to compare the new formulation against a standard mixed-integer programming model of the PDP, a line search, and a binary search. Our numerical results indicate that the new formulation in combination with the simple bounds is solved to optimality by an out-of-the-box mixed-integer programming solver in 34 out of 40 instances, while this is neither possible with the standard model nor with the search procedures. For instances in which the line and binary search fail to find a provably optimal solution, we achieve this by adding cuts to our enhanced formulation.

Key words: facility location, dispersion problems, max-min objective, integer programming

1. Introduction

In the p-dispersion problem (PDP), we are given a set of candidate locations $I = \{1, 2, \dots, n\}$ and an $n\times n$ matrix $(d_{ij})_{i,j\in I}$ with distances d_{ij} between facility i and j. The optimization task is to select $1 < p < n$ facilities from I such that the minimum distance between any pair of selected facilities is maximized.

In practice, this location problem occurs whenever a close proximity of facilities is less desirable. A standard application is concerned with the location of nuclear power plants. Therein, one is interested in minimizing the risk of losing multiple plants in the event that only one plant is subjected to an enemy attack. To achieve this, a selection of plants is desired so that interplant distances are as large as possible. Similar applications can be found in the military sector. In more peaceful contexts, one seeks for facilities of the same franchise system or for public facilities which have overlapping areas of service, e.g., schools, hospitals, waste collection plants, or electoral districts. We refer the reader to Kuby (1987) and to the comprehensive survey of Erkut and Neuman (1989) for an overview on the variety of applications of the PDP. Another area of application is recognized if distances are not interpreted physically but as a measure of the diversity between members of a group, e.g., products of the same portfolio (Saboonchi *et al.*, 2014).

The contribution of this paper is a new compact formulation of the PDP. To highlight the main objective pursued with this model, note that we intend to provide a competitive exact approach for the PDP in which a major part of the overall optimization task is undertaken by an out-of-the-box software package. To make the new model competitive, it is delivered along with two simple enhancements that do not add too much coding effort on top: We exploit simple bounds on the optimal distance to reduce the size and to increase the tightness of the model. The bounds are obtained by very simple heuristics that are already known in

[*]Corresponding author.
Email address: sayah@uni-mainz.de (David Sayah)

the literature. We show that clique inequalities are valid for the new model and can be used to further strengthen it. For the separation of the clique cuts, we also suggest a greedy heuristic in order to keep the coding effort and the computational burden as low as possible.

We carry out computational experiments over large-scale test instances in order to compare the new formulation against a standard mixed-integer programming model. The enhanced formulation is solved to optimality by a mixed-integer programming solver in 34 out of 40 test instances, while this is not possible with the standard model. We also compare our enhanced model against two standard search procedures for the PDP, i.e., a line search and a binary search. This comparison is interesting because these search procedures are easy-to-implement and exact making use of the relationship between the PDP and the maximum independent set problem. For instances in which either line or binary search, or both, fail to find a provably optimal solution, we achieve this by adding the clique cuts to our formulation.

The remainder of the paper is structured as follows: In Section 2, we present PDP formulations from the literature and introduce our new formulation. Section 3 describes the setup of the computational study and its results. The paper closes with final conclusions drawn in Section 4.

2. Formulations

Without loss of generality, we assume that the distance matrix (d_{ij}) is symmetric and that any non-diagonal value is strictly positive. All formulations are based on a graph representation of the problem. Let (I, E) be the complete graph in which locations I are the vertices and $E = \{(i, j) \in I \times I : i < j\}$ are the edges. Given any distance d, we further define subsets of edges as

$$E(d) = \{(i, j) \in E : d_{ij} < d\} \subseteq E.$$

The PDP is a bottleneck optimization problem with a max-min objective function (Hsu and Nemhauser, 1979). We now briefly review two existing non-linear formulations exploiting this fact before we present a standard mixed integer linear programming (MILP) model and our new formulation.

2.1. Non-Linear Formulations

The first formulation is the mixed integer non-linear program of Pisinger (2006): Define a vector of location variables $x = (x_i)_{i \in I}$ and let $x_i = 1$ indicate that candidate location $i \in I$ is open (0, otherwise). Using a continuous variable $d \geq 0$ for the minimum distance between open locations, the PDP can be written as

$$\begin{aligned}
z = \ & \max d && && \text{(1a)}\\
\text{s.t.} \ & \sum_{i \in I} x_i = p && && \text{(1b)}\\
& d x_i x_j \leq d_{ij} && (i, j) \in E && \text{(1c)}\\
& x_i \in \{0, 1\} && i \in I && \text{(1d)}\\
& d \geq 0. && && \text{(1e)}
\end{aligned}$$

The objective (1a) maximizes the minimum distance d, and exactly p candidate locations are opened because of (1b). The non-linear constraints (1c) impose that any two locations i and j are only opened simultaneously ($x_i x_j = 1$) if their distance d_{ij} is at least d. The variable domains are given by (1d) and (1e).

For a given value of d, the set of feasible solutions to PDP with minimum distance at least d is given by

$$\mathcal{X}(d) = \left\{ x \in \{0, 1\}^n : \sum_{i \in I} x_i = p \quad \text{and} \quad x_i + x_j \leq 1 \ \forall (i, j) \in E(d) \right\}.$$

A vector $x \in \mathcal{X}(d)$ is the incidence vector of an independent set (IDS) of size p in the graph $(I, E(d))$. It is equivalent to state that $\{i \in I : x_i = 1\}$ is a clique (of size p) in the complement graph $(I, E \setminus E(d))$. This notation allows us to state the PDP in the form

$$
\begin{aligned}
z = \ & \max d && \text{(2a)}\\
\text{s.t.}\ & \mathcal{X}(d) \neq \varnothing && \text{(2b)}\\
& d \geq 0. && \text{(2c)}
\end{aligned}
$$

The minimum distance d is maximized in (2a), while constraint (2b) states that a feasible choice of d has to ensure that an IDS of size p exists in $(I, E(d))$. We refer to the problem of deciding whether $\mathcal{X}(d)$ is non-empty for any d as the IDS problem. Erkut (1990) proposed another non-linear formulation similar to (2). Though neither of the above non-linear formulations was supposed to be solved directly. The authors motivate the two subsequent categories of exact solution approaches to the PDP which can be found in the literature.

MILP-Based Approaches. These approaches are driven by compact linearized versions of model (1). We describe a standard MILP formulation of the PDP in Section 2.2. Some authors suggest to solve the compact model straightaway using any off-the-shelf MILP solver (Kuby, 1987; Daskin, 1995). Erkut (1990) tailored a branch-and-bound algorithm for the PDP.

Search Procedures. Model (2) motivates a simple search algorithm, e.g., line or binary search, to find a largest minimum distance in combination with an efficient method to perform the feasibility tests in each iteration of the search. For a continuous version of the PDP defined on a tree, Chandrasekaran and Daughety (1981) propose a search procedure which requires consecutive solutions of anti-cover problems. The anti-cover problem (Chaudhry *et al.*, 1986) and the d-separation problem (Erkut, 1990; Erkut *et al.*, 1996) are synonyms for the maximum IDS problem. Pisinger (2006) suggests a binary search and considers cliques of size p for the feasibility test.

We position the contribution of this paper in the first category because a new compact formulation for the PDP is presented (see Section 2.3). Along with the new formulation, we provide in Section 2.3.2 a greedy but usually effective procedure to strengthen its linear relaxation by separating valid inequalities. In our computational tests, we benchmark the new formulation against the standard MILP formulation and against the search procedures known from the literature.

2.2. Kuby Formulation

Using an appropriately large number M, a linearization of (1) suggested by Kuby (1987) can be written as

$$
\begin{aligned}
z = \ & \max d && && \text{(3a)}\\
\text{s.t.}\ & \sum_{i \in I} x_i = p && && \text{(3b)}\\
& d \leq M(2 - x_i - x_j) + d_{ij} && (i, j) \in E && \text{(3c)}\\
& x_i \in \{0, 1\} && i \in I && \text{(3d)}\\
& d \geq 0. && && \text{(3e)}
\end{aligned}
$$

(3a), (3b), (3d), and (3e) are identical to formulation (1), while constraints (3c) replace (1c). The latter constraints guarantee $d \leq d_{ij}$ whenever both locations i and j are chosen via $x_i = x_j = 1$. This model is fairly compact, since it has $n + 1$ variables and $|E| + 1 = n(n-1)/2 + 1$ constraints.

It should be remarked that we formulate the linking constraints (3c) following the suggestion of Erkut (1990) and the most recent literature, e.g., Saboonchi *et al.* (2014). Kuby (1987) and also Daskin (1995) state these constraints in a slightly different way. However, we attribute Kuby's name to the model, since his work is the first documenting the "big M" constraints in the PDP context to our knowledge.

Finally, Ağca *et al.* (2000) developed a third mixed-integer formulation in order solve it via a Lagrangian relaxation-based heuristic. Their model is less compact than Kuby's, thus not considered in this paper.

2.3. New Compact Formulation

We now present our new compact formulation which exploits the fact that an optimal distance is identical to at least one of the entries of the distance matrix. Let $D^0 < D^1 < \ldots < D^{k_{max}}$ be the different non-zero values in (d_{ij}). The associated index sets are $K = \{1, 2, \ldots, k_{max}\}$ and $K_0 = \{0\} \cup K$. By definition, $\varnothing = E(D^0) \subsetneq E(D^1) \subsetneq E(D^2) \subsetneq \cdots \subsetneq E(D^{k_{max}}) \subsetneq E$ holds.

The new compact formulation uses two types of binary variables: As before, the binary location variable x_i indicates whether location $i \in I$ is opened. For $k \in K$, the binary variable z_k indicates whether the location decisions satisfy a minimum distance of at least D^k. The pure binary program reads as follows:

$$z = \max D^0 + \sum_{k \in K} (D^k - D^{k-1}) z_k \tag{4a}$$

$$\text{s.t.} \quad \sum_{i \in I} x_i = p \tag{4b}$$

$$z_k \leq z_{k-1} \qquad k \in K, k > 1 \tag{4c}$$

$$x_i + x_j + z_k \leq 2 \qquad k \in K, (i,j) \in E(D^k) \setminus E(D^{k-1}) \tag{4d}$$

$$x_i \in \{0, 1\} \qquad i \in I \tag{4e}$$

$$z_k \in \{0, 1\} \qquad k \in K \tag{4f}$$

Constraints (4b) ensure that exactly p locations are chosen. The consistency between the z_k variables is modeled via (4c) in the sense that the z_k variables are non-increasing in k. Consequently, any feasible solution fulfills that there exists a unique $k \in K_0$ with $z_1 = \cdots = z_k = 1$ and $z_{k+1} = \cdots = z_{k_{max}} = 0$ (in the extreme case of $k = 0$, $z_1 = \cdots = z_{k_{max}} = 0$). Whenever the minimum distance is at least D^k, i.e., $z_1 = z_2 = \cdots = z_k = 1$, the constraints (4d) ensure that no pair (i, j) of locations with distance $d_{ij} < D^k$ is chosen simultaneously. The domains of the variables are given by (4e) and (4f).

Any feasible solution to formulation (3) implies a feasible solution to formulation (4). Suppose that x is feasible in (3) and define the set $P = \{i \in I : x_i = 1\}$ of open locations. Then, x satisfies (4b) because P has the right cardinality p. Moreover, let D^ℓ for an $\ell \in K_0$ be the minimum distance between two locations in P, i.e., $D^\ell = \min_{i \neq j \in P} d_{ij}$. Then, any pair $i \neq j \in P$ fulfills $(i, j) \notin E(D^k)$ for all $k = 0, 1, \ldots, \ell$. It means that one can set $z_k = 1$ for all $k = 1, \ldots, \ell$ without violating any constraints in (4d) for these values of k. For all other values $k \in K, k > \ell$, setting $z_k = 0$ is also consistent with (4c). Due to (4a) the resulting objective value is $D^0 + \sum_{k=1}^{\ell} (D^k - D^{k-1}) z_k = D^\ell$, which shows that x is valued with the same minimum distance in (4). The reverse statement follows analogously. Hence, formulations (3) and (4) are equivalent as MILP formulations.

The new formulation (4) is compact, since it has exactly $n + k_{max}$ variables and $k_{max} + |E(D^{k_{max}})|$ constraints. Both values do not exceed n^2.

2.3.1. Exploitation of Lower and Upper Bounds

Model (4) can be reduced and tightened if bounds for the optimal distance z are available. If one knows $lb \leq z \leq ub$ with $lb = D^{k_{min}}$ and $ub = D^{k_{max}}$, then the definition of K can be altered into $K = \{k_{min}, k_{min} + 1, \ldots, k_{max}\}$. Moreover, one must redefine $D^0 := D^{k_{min}}$ and $E(D^{k_{min}-1}) := E(D^0)$. Now, formulation (4) has only $n + (k_{max} - k_{min} + 1)$ variables and only $(k_{max} - k_{min} + 1) + |E(D^{k_{max}}) \setminus E(D^{k_{min}})|$ constraints. It is guaranteed that the linear relaxation of formulation (4) delivers a bound between lb and ub.

Upper bounds can be computed with a procedure first suggested by Pisinger (2006). For each location $i \in I$, one first determines the $p - 1$ largest distances d_{ij} to any $j \in I, j \neq i$. Let d_i^{p-1} be the smallest of these distances. If these values are computed for all $i \in I$, one hast to find the pth largest value among these. This is a valid upper bound.

We suggest a simple lower bounding procedure that uses a greedy algorithm to compute maximum cardinality independent sets (Chaudhry *et al.*, 1986). It works as follows: For each $k \in K$ we consider the undirected graph $(I, E(D^k))$. Any independent set $S \subseteq I$ in this graph consists of locations with minimum distance at least D^k. If an independent set with cardinality $|S| \geq p$ is found, then D^k is a valid lower bound. In order to keep the computational effort small, we search for large independent sets S with decreasing values of k using the greedy approach in each outer iteration. In each inner iteration of the greedy algorithm, a minimum degree vertex is chosen, this vertex and all its adjacent vertices are removed from the graph, and the process is repeated. All chosen vertices form an independent set S. Such a greedy algorithm can be implemented using $\mathcal{O}(n^2)$ time.

2.3.2. Valid Inequalities

Formulation (4) can be strengthened by adding additional valid inequalities. The idea is to consider more than two locations at the same time that are incompatible with a certain minimum distance D^k. The valid inequalities have the following form:

$$\sum_{i \in S} x_i + (|S|-1)z_k \leq |S| \qquad k \in K,\ \varnothing \neq S \subset I : (i,j) \in E(D^k) \text{ for all } i < j \in S \tag{5}$$

Note first that for $1 \leq |S| \leq 2$ the inequalities are given by the bounds (4e) and constraints (4d), respectively. For larger $|S| \geq 3$, the validity can be derived considering the two cases $z_k = 0$ and $z_k = 1$. In the first case, $\sum_{i \in S} x_i \leq |S|$ is always true. In the second case, the minimum distance of the solution is D^k or larger so that $\sum_{i \in S} x_i \leq 1$ is valid as imposed by (5).

Violated inequalities (5) can be separated by solving a series of maximum weight clique problems, one for each $k \in K$. First, for a fixed k, weights for all locations $i \in I$ are defined by $w_i = x_i + z_k - 1$. Then, any clique $S \subseteq I$ in the graph $(I, E(D^k))$ is a valid set according to the definition in (5). If the weight of the clique, i.e., $\sum_{i \in S} w_i$ is greater than z_k, a violated inequality (5) is found.

Again, in order to keep the computational effort small, we apply a greedy algorithm for the separation. Starting from the graph $(I, E(D^k))$, we iteratively determine a vertex maximizing the product of vertex degree and weight. This vertex and vertices not adjacent to it are removed from the graph. The procedure is repeated on the resulting graph. All chosen vertices together form the clique. This greedy algorithm can be implemented so that its effort is $\mathcal{O}(n^2)$.

3. Computational Results

The compact formulation proposed in this paper provides a fast and easy-to-implement way to find optimal solutions to the PDP. We show how computation times scale and use the 40 *pmed* instances from the OR-Library (ORLIB) (Beasley, 1990). These symmetric instances have been originally designed to test p-median algorithms. Their distance matrices are quadratic and therefore appropriate for the PDP. Since the ORLIB instances are much larger as those used in recent studies, e.g., Della Croce *et al.* (2009), Porumbel *et al.* (2011), and Saboonchi *et al.* (2014), we think that they are well suited for a comparison.

The characteristics of the ORLIB instances are summarized in the first two blocks of Table 1. In the first block, p is the number of locations to open, $|K_0|$ the number of distinct distances, and n the overall number of locations. The largest instances have $n = 900$ locations and up to $p = 200$ locations need to be opened. In the second block, we show the lower and upper bounds resulting for the bounding procedures of Section 2.3.1. Lower and upper bounds are given by *lb* and *ub* and the corresponding k-values are given by k_{min} and k_{max}. For convenience, the remaining number of relevant distinct distances is displayed in the column $\#k$.

3.1. Computational Setup

All computations were performed on a standard PC with an Intel(R) Core(TM) i7-2600 running at 3.4 GHz processor with 16 GB of main memory. The bounding, separation, line and binary search procedures were coded in C++ and compiled in release mode with MS-Visual Studio 2010. We used CPLEX 12.5 as general-purpose mixed integer linear programming (MILP) solver and allow CPLEX to allocate two threads.

The following different exact methods were used to study the relative performance of the compact formulation of Section 2.3. We tested the six subsequent settings over all test instances:

Kuby Kuby formulation (3) solved by CPLEX.

NF The new formulation (4) solved by CPLEX.

NF* The reduced formulation (4) solved by CPLEX.

NFC* The reduced formulation (4) solved by CPLEX and separation of cuts (5).

LS* The non-linear formulation (2) solved via line search.

BS* The non-linear formulation (2) solved via binary search.

The parameter M in model (3) is set equal to the largest distance $D^{k_{max}}$. In setting NF, we solve the full model as defined in (4) with z_k variables for all $k = 1, \ldots, k_{max}$. As already mentioned, the clique cuts (5) in setting NFC* are separated by the heuristic described in Section 2.3.2. We kept all default parameters of CPLEX, except for NFC* for which we set the backtracking tolerance to zero (this lets CPLEX focus on improving the upper bound). All starred settings (NF*, NFC*, LS* and BS*) make use of the bounds described in Section 2.3.1 replacing 0 and k_{max} by k_{min} and some smaller k_{max}. In particular, the line and binary search procedures are accelerated by restricting the search interval to the distance values between $D^{k_{min}}$ and $D^{k_{max}}$. Our line search starts from $D^{k_{min}}$ and increases distance values since the lower bounds provided by the iterated greedy procedure are often close to the optimal distance (cf. Table 1). In the following, $\#k = k_{max} - k_{min} + 1$ denotes the resulting number of distinct distance values possible for the minimum distance z.

We notice that Erkut (1990) already suggests a line and a binary search over the minimum distance d. In the worst case, the line search needs at most $\mathcal{O}(\#k)$ and the binary search at most $\mathcal{O}(\log(\#k))$ checks. Of course, the IDS problem in every iteration can be solved with any general-purpose MILP solver. In addition, one can exploit the structure of the feasibility problem by using, e.g., "integer-friendly" models (see Erkut *et al.*, 1996). Pisinger (2006) proposes a special-purpose algorithm based on a dense subgraph representation of the feasibility problem.

Our attempt is to further speed up both search procedures while keeping the effort incurred by their implementation as low as possible. Thus, we realized the feasibility tests using the exact method introduced by Östergård (2002) for which the author distributes a free C library. Note that Östergård's implementation allows to specify lower and upper bounds to terminate the search prematurely.

3.2. Results

We start with an analysis of the upper bounds resulting from the linear relaxations produced by Kuby, NF, NF*, and NFC*. The third block of Table 1 shows these bounds. Note that the LP bounds stated in the second last column (NFC*) were obtained by the heuristic separation of the clique cuts (5) as described in Section 2.3.2.

It is obvious that the new formulation (4) produces significantly tighter upper bounds than Kuby's model (3), in particular when the preprocessing has already restricted the search space to distances D^k with $k_{min} \leq k \leq k_{max}$. It can also be seen that the preprocessing upper bound (ub) is preserved when solving the linear relaxation of the reduced model, see the LP bounds in column NF* of Table 1. The last column of this table reports the optimal distance values z or, where these are not known, the best known lower and upper bounds. Taking these values as benchmark, the linear relaxation of the Kuby formulation (3) produces upper bounds that are on average approximately 75% above, while the LP bounds associated with the new formulation (4) and the reduced version are 54% and 29% above, respectively. From the last two columns of Table 1, we compute an average integrality gap of 11% if the clique cuts (5) are heuristically separated and added, and we see that the PDP is solved at the root node in 19 out of 40 cases.

It remains to discuss the results with respect to Kuby, NF, NF*, NFC*, LS*, and BS*. The corresponding numbers are reported in the six blocks of Table 2. Each block consists of two columns: The first column shows either the optimal distance or the lower bound associated with the best integer solution and the best upper bound. The numbers in the second column are the computation times in seconds required to prove

Instance				Proprocessing bounds					LP upper bounds				Opt.
Name	p	$\|K_0\|$	n	lb	ub	k_{min}	k_{max}	$\#k$	Kuby	NF	NF*	NFC*	z
pmed1	5	284	100	223	268	221	266	46	599	299	268	228	228
pmed2	10	282	100	181	240	178	237	60	601	316	240	181	181
pmed3	10	316	100	164	236	162	234	73	731	388	236	193.1	167
pmed4	20	289	100	124	225	117	218	102	571.5	335	225	126.5	125
pmed5	33	261	100	75	151	74	150	77	451.7	312	151	75	75
pmed6	5	188	200	159	178	158	175	18	397	198	178	159	159
pmed7	10	170	200	115	145	114	144	31	367.6	184	145	118	118
pmed8	20	204	200	92	136	91	135	45	414.6	220	136	109.3	92
pmed9	40	189	200	60	108	59	107	49	362.5	215	108	65	62
pmed10	67	162	200	33	74	32	73	42	240.1	169	74	33	33
pmed11	5	129	300	112	116	111	115	5	269	134	116	112	112
pmed12	10	154	300	92	109	91	108	18	335	167	109	99.3	92
pmed13	30	139	300	61	91	60	90	31	283.3	150	91	82.3	64
pmed14	60	160	300	42	77	41	76	36	299.7	179	77	56.1	43
pmed15	100	130	300	26	62	25	61	37	194	136	62	33.2	27
pmed16	5	103	400	91	94	90	93	4	215	107	94	91	91
pmed17	10	103	400	71	83	70	82	13	211	105	83	71	71
pmed18	40	118	400	45	72	44	71	28	264.3	141	72	70.2	48
pmed19	80	100	400	29	60	28	59	32	170.9	101	60	43	31
pmed20	133	111	400	20	52	19	51	33	160.6	113	52	21	21
pmed21	5	88	500	74	78	73	77	5	183	91	78	74	74
pmed22	10	111	500	65	78	64	77	14	227	113	78	75.4	66
pmed23	50	94	500	36	60	35	59	25	177.2	94	60	56.5	39
pmed24	100	95	500	24	49	23	48	26	167.7	100	49	30.2	25
pmed25	167	99	500	17	43	16	42	27	144	102	43	17	17
pmed26	5	82	600	67	74	66	73	8	175	87	74	68	68
pmed27	10	90	600	58	66	57	65	9	183	91	66	59	59
pmed28	60	106	600	31	48	30	47	18	204.9	110	48	41.5	31
pmed29	120	87	600	21	42	20	41	22	147.5	88	42	30.6	22†
pmed30	200	95	600	14	39	13	38	26	134.9	96	39	16	15
pmed31	5	65	700	56	60	55	59	5	131	65	60	57	57
pmed32	10	117	700	51	58	50	57	8	249	124	58	52	52
pmed33	70	71	700	26	43	25	42	18	138.7	74	43	42	[27,28†]
pmed34	140	94	700	18	37	17	36	20	162.8	98	37	27.4	19†
pmed35	5	69	800	58	59	57	58	2	149	74	59	58	58
pmed36	10	87	800	50	57	49	56	8	175	87	57	51	51
pmed37	80	77	800	26	41	25	40	16	146	78	41	37.3	[26,27†]
pmed38	5	80	900	57	58	56	57	2	169	84	58	57	57
pmed39	10	95	900	40	47	39	46	8	231	115	47	46	41
pmed40	90	68	900	21	37	20	36	17	129.1	69	37	37	[22,29]

Table 1: Characteristics of instances, lower and upper bounds, LP bounds, and optimal distances; the values marked with † were obtained with NFC* in extended computation time

the optimal distance. An entry "TL" indicates that it was not possible to prove optimality within the time limit of 1800 seconds, and the entry "ME" stands for a CPLEX abort prior to the time limit due to the occurrence of an out-of-memory error. The last three lines of Table 2 are the number of solved instances, the average computation time over all instances, and the average computation time over instances solved to optimality.

Surprisingly, the line search outperforms the binary search on average over the solved instances, see the last line of Table 2. This is due to the fact that the greedy lower bound, from which we start the line search, is often quite close to the optimum. In these cases, the line search may need less iterations to hit the optimum distance than the binary search. Note that this statement does not necessarily hold in general because the solvability of the IDS problems depends on the value d. Thus, even if the line search requires less iterations, we may be better off with the binary search if the feasibility checks are easier to perform.

It is obvious that, when the new formulation (4) is preprocessed using the simple bounds, it is possible to solve 34 instances, i.e., 10 instances more than with the standard model (3) and 4 instances more than both line and binary search. Moreover, the greedy separation appears quite effective in the sense that the average computation time of NF* could be further shortened by 14,7% over all instances and by 26.8% over the solved instances.

	Kuby		NF		NF*		NFC*		LS*		BS*	
Name	*lb/ub*	time	*lb/ub*	time	*lb/ub*	time	*lb/ub*	time	*lb/ub*	time	*lb/ub*	time
pmed1	228	0.7	228	1.3	228	0.1	228	0.1	228	< 0.1	228	< 0.1
pmed2	181	2.4	181	4.3	181	0.2	181	0.2	181	< 0.1	181	< 0.1
pmed3	167	7.1	167	9.8	167	0.6	167	0.8	167	< 0.1	167	< 0.1
pmed4	125	301.3	125	12.5	125	0.4	125	0.4	125	< 0.1	125	< 0.1
pmed5	75	443.6	75	56.5	75	0.2	75	0.3	75	< 0.1	75	< 0.1
pmed6	159	2.2	159	3.0	159	< 0.1	159	< 0.1	159	< 0.1	159	< 0.1
pmed7	118	10.5	118	18.7	118	0.5	118	0.7	118	< 0.1	118	< 0.1
pmed8	92	105.6	92	39.8	92	3.1	92	4.4	92	< 0.1	92	< 0.1
pmed9	62/107	TL	62/79	389.3	62	3.9	62	8.0	62	0.7	62	< 0.1
pmed10	33/69	TL	33	195.8	33	0.8	33	0.9	33	< 0.1	33	7.0
pmed11	112	22.0	112	7.2	112	< 0.1	112	< 0.1	112	< 0.1	112	< 0.1
pmed12	92	35.0	92	280.2	92	0.2	92	0.5	92	< 0.1	92	< 0.1
pmed13	64/93	TL	64	279.8	64	36.8	64	32.2	64	< 0.1	64	< 0.1
pmed14	43/96	TL	43	1059.2	43	30.9	43	59.4	43	94.6	43	96.5
pmed15	27/78	TL	27	305.5	27	12.1	27	37.8	27	920.3	27	1462.3
pmed16	91	37.1	91	40.0	91	0.1	91	0.1	91	< 0.1	91	< 0.1
pmed17	71	101.8	71	49.6	71	0.4	71	0.8	71	< 0.1	71	< 0.1
pmed18	48/84	TL	48/84	TL	48	1486.2	48	740.6	48	18.2	48	17.1
pmed19	31/79	ME	31/67	TL	31	623.2	31	249.0	29/60	TL	29/36	TL
pmed20	20/77	TL	74	TL	21	25.3	21	95.1	20/52	TL	20/36	TL
pmed21	74	30.1	74	111.8	74	0.2	74	0.2	74	< 0.1	74	< 0.1
pmed22	66	200.0	66	342.7	66	1.0	66	4.0	66	< 0.1	66	< 0.1
pmed23	39	15.3	39/44.8	TL	39	1427.2	39	1054.8	39	51.0	39	15.3
pmed24	24/103	TL	24/103	TL	25	1617.9	25	379.0	24/49	TL	24/36	TL
pmed25	17/85.5	TL	17/94.8	TL	17	15.0	17	18.3	17/43	TL	17/43	TL
pmed26	68	30.2	68	187.9	68	0.5	68	0.8	68	< 0.1	68	< 0.1
pmed27	59	247.4	59	272.2	59	2.6	59	3.5	59	< 0.1	59	< 0.1
pmed28	31	129.0	31/110	TL	31/48	TL	31	1311.4	31	145.3	31	130.2
pmed29	21/91	TL	21/91	TL	22/23	TL	22/23	TL	21/42	TL	21/31	TL
pmed30	14/97.9	TL	14/96	TL	15	211.4	15/18.5	ME	14/39	TL	14/39	TL
pmed31	57	76.3	57	197.9	57	0.8	57	1.1	57	< 0.1	57	< 0.1
pmed32	52	434.1	52	517.1	52	3.5	52	4.8	52	< 0.1	52	< 0.1
pmed33	26/34	TL	26/34	TL	26/43	TL	27/41.9	TL	26/43	TL	26/34	TL
pmed34	18/101.4	TL	18/98	TL	18/21	TL	18/20	TL	18/37	TL	18/27	TL
pmed35	58	< 0.1	58	868.6	58	0.8	58	0.9	58	< 0.1	58	< 0.1
pmed36	51	< 0.1	51	808.5	51	4.6	51	5.6	51	< 0.1	51	< 0.1
pmed37	26/83.9	TL	26/83.9	TL	26/41	TL	26/36.9	TL	26/41	TL	26/33	TL
pmed38	57	155.7	57	155.7	57	0.9	57	1.0	57	< 0.1	57	< 0.1
pmed39	41	< 0.1	41/108.8	TL	41	6.7	41	20.7	41	< 0.1	41	< 0.1
pmed40	21/29	TL	21/69	TL	22/37	TL	21/37	TL	21/37	TL	21/29	TL
#opt.	24		25		34		34		30		30	
avg.(tot.)		777.4		785.5		408.1		348.0		481.0		493.5
avg.(opt.)		99.5		233.0		162.3		118.7		41.0		57.6

Table 2: Numerical results for the *pmed* instances of the ORLIB; the time limit TL was set to 1800 seconds; ME indicates the occurrence of an out-of-memory error

4. Conclusions

We proposed a new compact pure binary formulation for the p-dispersion problem along with two options to enhance the formulation at a low computational cost. First, we suggested a preprocessing phase in which the number of constraints and variables can be considerably reduced by exploiting simple bounds on the optimal distance. Second, we showed that clique inequalities are valid for the new model and that their heuristic separation is sufficient to achieve a significant speed-up in many tested instances.

For future research, fast improvement heuristics in the preprocessing phase might further improve our formulation, since in some instances the best upper bounds are already tight. Moreover, it would be interesting to see if our modeling approach also applies to the various related dispersion problems. Among these variants are, for example, the PDP with an additional given set fixed open facilities or with demand locations, from which the new facilities have to be located as far as possible, and the well-known p-dispersion-sum problem of maximizing the total minimum distance between open facilities and demand locations.

References

Ağca, Ş., Eksioglu, B., and Ghosh, J. B. (2000). Lagrangian solution of maximum dispersion problems. *Naval Research Logistics*, **47**(2), 97–114.

Beasley, J. E. (1990). OR-library: Distributing test problems by electronic mail. *The Journal of the Operational Research Society*, **41**(11), 1069–1072.

Chandrasekaran, R. and Daughety, A. (1981). Location on tree networks: P-centre and n-dispersion problems. *Mathematics of Operations Research*, **6**(1), 50–57.

Chaudhry, S. S., McCormick, S. T., and Moon, I. D. (1986). Locating independent facilities with maximum weight: Greedy heuristics. *Omega*, **14**(5), 383–389.

Daskin, M. S. (1995). *Network and Discrete Location*. Wiley, New York.

Della Croce, F., Grosso, A., and Locatelli, M. (2009). A heuristic approach for the max–min diversity problem based on max-clique. *Computers & Operations Research*, **36**(8), 2429–2433.

Erkut, E. (1990). The discrete p-dispersion problem. *European Journal of Operational Research*, **46**(1), 48–60.

Erkut, E. and Neuman, S. (1989). Analytical models for locating undesirable facilities. *European Journal of Operational Research*, **40**(3), 275–291.

Erkut, E., ReVelle, C., and Ülküsal, Y. (1996). Integer-friendly formulations for the r-separation problem. *European Journal of Operational Research*, **92**(2), 342–351.

Hsu, W.-L. and Nemhauser, G. L. (1979). Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, **1**(3), 209–215.

Kuby, M. J. (1987). Programming models for facility dispersion: The p-dispersion and maxisum dispersion problems. *Geographical Analysis*, **19**(4), 315–329.

Östergård, P. R. (2002). A fast algorithm for the maximum clique problem. *Discrete Applied Mathematics*, **120**(1–3), 197–207.

Pisinger, D. (2006). Upper bounds and exact algorithms for p-dispersion problems. *Computers & Operations Research*, **33**(5), 1380–1398.

Porumbel, D. C., Hao, J.-K., and Glover, F. (2011). A simple and effective algorithm for the MaxMin diversity problem. *Annals of Operations Research*, **186**(1), 275–293.

Saboonchi, B., Hansen, P., and Perron, S. (2014). MaxMinMin p-dispersion problem: A variable neighborhood search approach. *Computers & Operations Research*, **52**, 251–259.