Computers & Operations Research ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

A new heuristic for solving the p-median problem in the plane

Jack Brimberg [a], Zvi Drezner [b,*]

[a] The Royal Military College of Canada, Kingston, ON, Canada
[b] Steven G. Mihaylo College of Business and Economics, California State University-Fullerton, Fullerton, CA 92834, United States

ARTICLE INFO

Keywords:
Continuous p-median
Location–allocation
Heuristics

ABSTRACT

This paper presents a new local search for solving the continuous p-median problem in the plane. The basic idea is to first find a good starting solution by overlaying the area containing the set of demand points with a grid and solving heuristically the location problem on this grid. The solution is then used as an initial point for running an improved version of Cooper's well-known alternating local search.

1. Introduction

The p-median problem is probably the most widely used model in location analysis and the most researched one [1,13,20,21,34]. The problem is to locate p facilities to serve n demand points each with an associated weight. Each demand point patronizes the closest facility and the objective is to minimize the weighted sum of distances of all demand points. The more researched version of the problem is in a network environment where demand points are located on vertices and distances are measured as shortest distances between any two points on the network [13]. Hakimi [26,27] proved that optimal locations of the facilities exist at the nodes of the network. In the discrete version of the p-median problem it is assumed that the locations of the facilities are restricted to a finite set of candidate sites. By the Hakimi property the p-median problem in a network environment is thus converted to a discrete problem.

The p-median problem in a planar environment is commonly called the continuous "location–allocation" problem [11,12]. It is also referred to as the multisource Weber problem (e.g., see [7]). Some earlier papers on the subject are Babich [3], Love and Morris [33], Sherali and Shetty [43] for rectilinear distances, Cooper [11,12], Chen [10], Drezner [16] for Euclidean distances, Sherali and Tuncbilek [44] for squared Euclidean distances, and Bongartz et al. [4] for general ℓ_p distances. Love [32] solved the problem on a line using dynamic programming.

The state of the art on the Euclidean distance problem in the plane is the extensive work by Brimberg et al. [7]. Interested readers are also referred to survey papers of the p-median problem given in Mladenović et al. [37] for the discrete model and Brimberg et al. [6] for the continuous one. For a recent overview of solution approaches see Brimberg and Hodgson [8] and its list of references.

The problem is to find p locations for facilities in the plane. A set of n demand points each with an associated weight $w_i > 0$ is given. Each demand point gets its service from the closest facility to it. The objective is to minimize the total sum of weighted minimum distances to the facilities. Let $d_i(X_j)$ be the distance between demand point i and facility j located at $X_j = (x_j, y_j)$. The vector of unknown locations is $\mathbf{X} = \{X_1, \ldots, X_p\}$, and thus, the objective function to be minimized is

$$F(\mathbf{X}) = \sum_{i=1}^{n} w_i \min_{1 \le j \le p} \{d_i(X_j)\} \tag{1}$$

The nonconvexity of the objective function and the existence of multiple local minima were observed by Cooper, the originator of the model [11,12], who proposed several heuristic approaches to solve it. The best known of these methods is the elegant alternating heuristic of Cooper, although at the time the author was not much impressed with it. This method takes advantage of the two simple sub-problems embedded in the model. That is, given the locations of the facilities (vector $\mathbf{X}$), the demand points are simply allocated to their closest facility (with ties broken arbitrarily). Once the allocations or partition of the customer set is fixed, the problem reduces to p independent single facility problems with convex objective functions that can be solved numerically by an iterative process such as the Weiszfeld procedure [47,48]. Thus, Cooper's alternating approach switches between location and allocation phases until no further improvement in the objective function is found.

Cooper's alternating local search starts by dividing the customer set into p subsets of approximately equal size. To overcome this rather messy initialization, Scott [42] suggested that the procedure start by randomly selecting p trial facility locations. The original intent of Cooper was to apply the algorithm once.

* Corresponding author. Tel.: +1 714 278 2712.
E-mail addresses: Jack.Brimberg@rmc.ca (J. Brimberg), zdrezner@fullerton.edu (Z. Drezner).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.07.012

Maranzana [35], working on a similar alternating scheme for the network version of the problem, was the first to propose a multi-start application of the local search, where repeated trials are conducted from different randomly generated initial solutions. The random multi-start version of Cooper's algorithm combined with Scott's starting solution remained the state-of-the-art for solving the planar problem, in spite of the development of several competing local searches. These include the work by Tornqvist et al. [46] who suggested employing a gradient search rather than alternating between location and allocation. Here the objective function is improved by moving the configuration of facility locations in the direction of steepest descent. Other gradient-based heuristics are found in Murtagh and Niwattisyawong [38], Chen [10], and the projection method of Bongartz et al. [4]. It was not until the advent of metaheuristics such as tabu search, variable neighborhood search and the genetic algorithm that better heuristics were developed to solve the problem, with significant improvements obtained especially on larger scale instances (e.g., see [7] for further details).

It is interesting to note the similarity between Cooper's alternating method and the K-means algorithm widely used in statistics and clustering [29,30]. This similarity was pointed out by Arthanari and Dodge [2]. Both heuristics apply the same idea of switching between location and allocation phases. However, in clustering and statistical models, the fixed points (referred to here as demand points) are considered homogeneous, and hence, all assigned a unit weight. Also the K-means algorithm uses squared Euclidean distance to measure the distance between the centers (facilities) and the fixed points. This makes the location phase much easier to solve, since the center of gravity of a given subset of fixed points may be found by simple closed formulas, instead of the more time-consuming numerical procedure for finding the median point when straight Euclidean distances are used.

The location–allocation problem defined in (1) is known to be NP-hard [36]. Therefore, approximate methods are needed to find 'good' solutions to problems of larger scale that may occur in practice. In this paper we present a new heuristic for solving (1). The main idea is to replace random starting points in Cooper's algorithm by high-quality starting points. This is accomplished by overlaying the area containing the demand points with a grid, and solving heuristically a relaxation of model (1) where the facility locations are restricted to the nodes of the grid. The mesh size must be selected with care in order to allow high quality starting solutions to be obtained. On the other hand, the grid should not be too fine since this will increase computation time unnecessarily. The solution found on the grid is used in the next phase as a starting point in a Cooper-type local search.

The heuristic presented here has a similar flavor as the p-median heuristic in [28]. In the latter case, the authors solve exactly a discrete version of (1) where the facility sites are restricted to the set of demand points. They complete the solution with a single adjustment in continuous space using the allocations obtained in the discrete phase. Although very good solutions are obtained (also see [7]), the computation time in the first phase quickly becomes excessive so that results are only reported for smaller instances. We believe our approach has several important advantages. First, the grid allows much more flexibility than the restriction to the set of demand points. Second, the heuristic solution of the discrete relaxation allows larger instances to be solved in reasonable time. Finally our approach terminates with a complete local search in continuous space instead of a single adjustment step.

The main heuristic presented here may be classified as a composite heuristic, in that it combines two heuristics in sequence. The first (discrete) phase is used to find a high-quality starting solution for the second phase where an improved alternating local search is applied to find a local optimum in the original continuous space. Of course we could consider several other possibilities for either phase. For example, there are greedy algorithms proposed in the literature for the network model that could be used alternatively in the first phase to find the starting solution. These include the algorithm by Kuehn and Hamburger [31] where facilities are added one at a time at the fixed points (or nodes) in a greedy way that causes the biggest reduction in the objective function each time; this process continues until p facilities are opened. An efficient implementation of this heuristic is given by Whitaker [49]. Another approach is the stingy approach (e.g., [22,41]), which starts with more than p open facilities at the nodes and removes them one at a time in a way that increases the objective function as little as possible each time; this would continue until p facilities remain. In fact any heuristic applied to a discretized version of the continuous model could be selected for the initial phase, and there are several to choose from (e.g., see the survey paper by Mladenović et al. [37]). Furthermore, other techniques could be considered in the second phase for augmenting the alternating search, such as applying a "drop and add" step as discussed in Densham and Rushton [14] for the network model, and Brimberg et al. [7] for the continuous model, instead of the transfer follow-up procedure that we propose. But this deviates from the main purpose of the paper, which is to show that by combining a good starting solution with a simple local search, we may obtain very fast and competitive heuristics for the location–allocation problem in the continuous plane.

Another interesting question is whether the quality of the final solution (measured by the value of the objective function) depends on the quality of the initial solution. We found that there is a highly significant correlation between the quality of the starting solution and the final solution of Cooper's alternate method. This suggests that it is beneficial to generate good starting solutions for Copper's algorithm.

The paper is organized as follows. In the next section, the details of our heuristic are presented. First we describe the alternating local search that is used after the grid search, and which incorporates several improvements to the original Cooper algorithm. We also propose heuristic algorithms for the grid search that follow the algorithms suggested in [15] for the multiple competitive location problem in the plane. Improvements are also given for the grid search and for the Weiszfeld procedure that is embedded in the alternating local search with the aim of making the new procedure as efficient as possible. Section 3 describes the computational experiments that were carried out. A discussion of the computational results is also provided. We finish with some conclusions and directions for further research.

2. The algorithms

We propose improvements to the alternate approach suggested by Cooper [11,12]. We also propose heuristic algorithms that follow the algorithms suggested in Drezner et al. [15] for the solution of the multiple competitive location problems in the plane.

2.1. The alternate approach

Cooper [11,12] suggested the alternate approach to heuristically solve the location–allocation (p-median) problem. As noted above, the idea is to alternate between allocating the demand points to facilities and finding the optimal location for each subset and to repeat the process until convergence. This process must converge to a local minimum because the value of the objective function decreases every iteration. We suggest a modified version

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

of the Cooper [11,12] approach which provides better results than the original approach. To avoid bias in the updates of the locations, the order of updating the locations is performed randomly. We also maintain an indicator vector of length p associated with the facilities. A zero indicator value means that the facility is not necessarily at the optimal location for the set of demand points attracted to it. It is equal to 1 if the facility is located at the optimal location for the set of demand points attracted to it.

2.1.1. New alternating local search

1. Initial locations for p facilities are generated (randomly or otherwise).
2. Each demand point is assigned to the closest facility partitioning the set of demand points to subsets each attracted to a facility.
3. An assignment vector is maintained along with an indicator vector of length p associated with the facilities that is initially set to all zeroes.
4. A facility with an indicator of zero is randomly selected.
5. The Weiszfeld algorithm [47] as accelerated in Drezner [17] is used to find the optimal location of the selected facility for the subset of demand points currently attracted to that facility. The facility is relocated and the indicator for the relocated facility is set to one.
6. The demand points are re-allocated to their closest facilities. For each demand point that changed an assignment, the two facilities involved in the change (including possibly the facility that just has been relocated) get an indicator of zero.
7. If all indicators are equal to one, stop. Otherwise, go to Step 4.

2.2. The transfer follow-up

When the alternate approach terminates, we propose an exchange algorithm as an attempt to improve the solution. Demand points are transferred one at a time from one subset to another. Examining all transfers of demand points to other subsets is inefficient. We select a pre-specified number L of demand points to be transferred to a particular different subset one at a time. These L transfers are selected as follows. Each facility attracts demand points in a Voronoi region [45,39] that is defined by perpendicular bisectors with all other facilities. We select demand points that are close to the boundary between two facilities and transfer them to the facility on the other side of the boundary.

2.2.1. The transfer follow-up

- For each demand point i the difference between the shortest distance to a facility and the second shortest distance to another facility δ_i is calculated.
- The L smallest values of δ_i are selected for transfer starting at the smallest δ_i and continuing in order.
- The L transfers to be examined are transferring the assignment of a demand point from its closest facility to its second closest. Note that the objective function initially increases with the transfer, but subsequently may be reduced by re-positioning the two affected facilities.
- If a transfer fails to improve the value of objective function, the transferred point is returned to its original subset.
- If a transfer leads to an improved value of the objective function, it is performed, and the transfer follow-up terminates.
- If all L transfers fail to improve the value of the objective function the algorithm terminates.

Once the transfer follow-up finds an improved solution, the alternate method is restarted followed by the transfer follow-up. The two vectors used by the alternate approach are updated considering the implication of the executed transfer on these vectors. If the transfer follow-up fails to find an improved solution, the algorithm terminates. The combined alternate approach and the transfer follow-up is termed the improved alternate approach, or in short, IALT.

2.3. Pseudo-code of IALT

An initial vector $\mathbf{X}$ of locations X_j for $j=1,\ldots,p$ is given. S_j for $j=1,\ldots,p$ is the set of all demand points attracted to facility j. a_i is the facility closest to demand point i. I_j for $j=1,\ldots,p$ is an indicator vector. When X_j is located at the optimal solution of the Weber problem based on S_j then $I_j=1$. Otherwise $I_j=0$. Initially $I_j=0$, $\forall j$. A parameter L is given.

1. Facility $j\in[1,\ldots,p]$ with $I_j=0$ is randomly selected.
2. The optimal solution X^* for the Weber problem based on S_j is found. Set $X_j=X^*$ and $I_j=1$.
3. The demand points are re-allocated to their closest facilities yielding an assignment vector a'_i for $i=1,\ldots,n$ and updating S_j for $j=1,\ldots,p$.
4. For all $i=1,\ldots,n$ do: if $a'_i\neq a_i$ set $I_{a_i}=I_{a'_i}=0$ and set $a_i=a'_i$.
5. If at least one $I_j=0$, go to Step 1.
6. Calculate D_i, the distance to the closest facility a_i and D'_i the second closest distance which is to facility b_i. Calculate $\delta_i=D'_i-D_i$ for $i=1,\ldots,n$.
7. The vector $\{\delta_i\}$ is sorted yielding $\delta_{(1)}\leq\cdots\leq\delta_{(n)}$. Set $k=1$.
8. Perform a transfer of demand point (k) from facility $a_{(k)}$ to facility $b_{(k)}$ by setting $S'_{a_{(k)}}=S_{a_{(k)}}-(k)$ and $S'_{b_{(k)}}=S_{b_{(k)}}\cup(k)$. Solve two Weber problems based on $S'_{a_{(k)}}$ and $S'_{b_{(k)}}$ yielding $X'_{a_{(k)}}$ and $X'_{b_{(k)}}$ defining the vector $\mathbf{X}'$.
9. If $F(\mathbf{X}')<F(\mathbf{X})$, set $\mathbf{X}=\mathbf{X}'$ and go to Step 3.
10. If $F(\mathbf{X}')\geq F(\mathbf{X})$, set $k=k+1$. If $k\leq L$ go to Step 8. Otherwise, the algorithm terminates.

2.4. Metaheuristics

As mentioned above, it is important to choose an appropriate grid size in the relaxation of model (1) in order to be able to obtain starting solutions of high quality. For our purposes, suppose we use a 101 by 101 grid, which for the problems investigated provides a number of grid points that is at least an order of magnitude greater than the number of demand points. The set of demand points is enclosed in a square (it can be a rectangle or a polygonal region in general) of side S and its lower left corner is located at (x_0,y_0). The size of the grid is set to $\Delta=S/100$. As a starting solution p facilities are randomly selected on points $(x_0+k\Delta,y_0+m\Delta)$ where k and m are integers randomly generated in the range [0,100]. A "move" is defined as moving one facility to eight possible locations: changing x by $-\Delta$, 0, $+\Delta$ and the same options for changing y except for the combination of a (0,0) change. We also stipulate that moves to the outside of the square are not considered. When only such moves are allowed, all facilities stay on the grid. Our goal for the heuristic is to find the best solution on the grid and use it as a starting solution to IALT. We tested a descent heuristic, tabu search, and simulated annealing.

2.4.1. Descent

The descent algorithm is straightforward. Evaluate all eight moves for each facility and select the best improving move. If there is no improving move among the $8p$ possible moves, stop.

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

2.4.2. Tabu search

Tabu search [23–25] proceeds from the terminal solution of the descent algorithm by allowing upward moves hoping to obtain a better solution in subsequent iterations. A tabu list of forbidden moves is maintained. Tabu moves stay in the tabu list for tabu tenure iterations. To avoid cycling, most tabu search algorithms use the reverse of recent moves as forbidden moves. Similar to the descent algorithm, the changes in the value of the objective function in the neighborhood are evaluated. If there is at least one move leading to a solution better than the best found solution, the best of such moves is executed regardless of the tabu list. If none of the moves leads to a solution better than the best found solution, the best permissible move (disregarding moves in the tabu list), whether improving or not, is executed. The process continues for a pre-specified number of iterations.

Using reverse moves in the tabu list performed very poorly. We realized that the poor performance of the tabu search was due here to an improper definition of the tabu list. Normally, the tabu list is updated with the opposite move to the one that was just selected. So, if a facility moves in the "east" direction, i.e., x is changed by $+\Delta$ and y remains the same, then the tabu move entered into the tabu list is moving that facility "west" thus returning to the same location. However, with the special structure of the moves in the tabu algorithm, this may not be an effective tabu list. Consider Fig. 1. If three such moves are executed, the tabu list will not prevent such a loop from being repeated indefinitely. There are many such loops with the same property. When a tabu search based on such a tabu list was constructed, its performance was not much better than the descent algorithm. We therefore designed a different definition of moves in the tabu list. The following tabu list led to a much improved tabu search algorithm. There are $101\times101=10{,}201$ possible locations for facilities. The tabu list consists of *locations* recently visited by a facility (regardless of which facility it was).

The following is a short description of the tabu search.

1. A tenure vector consisting of 10,201 entries for each potential location is maintained.
2. The resulting solution of the descent algorithm is selected as a starting solution for the tabu search and as the best found solution. The number of iterations for the tabu search is set to $IT=5000p$ and $iter=0$.
3. Every potential location in the tenure vector is assigned a large negative number.
4. The tabu tenure, T, is randomly selected in the range [100,1000].
5. All $8p$ moves (fewer possible moves if we have border facilities) are evaluated and the value of the objective function is calculated for each.
6. If a move entry i to entry j yields a solution better than the best found one, continue to evaluate all the moves and perform the best improving move. Update the best found solution and go to Step 8.
7. If no move yields a solution better than the best found solution, select the move which yields the best value of the objective function (whether improving or not) as long as the difference between the current iteration and the entry of j in the tenure vector exceeds T. If all entries j of the $8p$ moves do not exceed T, select the best move disregarding the tabu list.
8. The current iteration number is entered into entry i of the selected move in the tenure vector.
9. Set $iter=iter+1$. If $iter=IT$ stop with the best found solution as the tabu solution. Otherwise, go to Step 4.

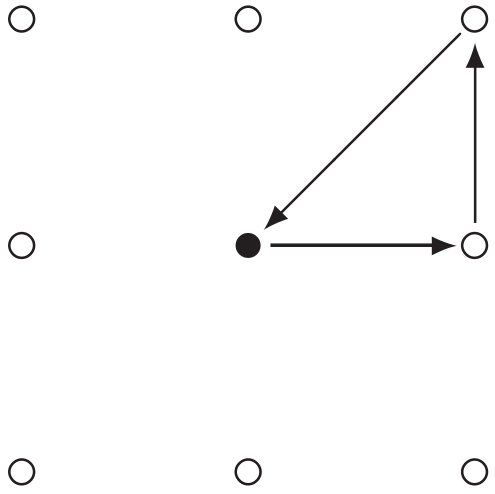

Fig. 1. A tabu list example.

2.4.3. Simulated annealing

The simulated annealing approach performed best in [15] and it also performed best, as reported in the computational experiments section, for the p-median problem in the plane. We set three parameters: the starting temperature T_0, the number of iterations N, and the factor $\alpha<1$ by which the temperature is lowered every iteration. The temperature T is set to T_0. N iterations are repeated. Every iteration the following steps are performed:

- One of the $8p$ moves is selected at random and the change Δf in the value of the objective function evaluated.
- If $\Delta f\le 0$ the move is performed, otherwise the move is accepted and performed with a probability of $e^{-\Delta f/T}$. If the move is not accepted, there is no change in the solution.
- The temperature T is multiplied by α.

The best solution encountered throughout the iterations is the solution of the algorithm.

Following extensive experiments the following parameters were used in the computational experiments

- $N=500{,}000p$.
- $T_0=1$.
- $\alpha=1-7/N$.

This selection of α leads to a final temperature which is $e^7\approx1096$ times smaller than T_0.

2.4.4. Efficient calculations

Calculating the change in the value of the objective function Δf can be done more efficiently than calculating the objective function itself. This is applicable to all heuristics. The objective function value is obtained by calculating the distance from each demand point to all p facilities, selecting the shortest distance and adding the term associated with the demand point. This requires $O(np)$ time. To calculate Δf more efficiently, two vectors are maintained. For each demand point we save the minimum distance to all facilities and the facility for which this minimum is obtained. When a location of a facility is changed, the distance to that facility is calculated and compared with the minimum distance.

- If the distance is smaller than the minimum distance, the difference multiplied by the weight is subtracted from Δf. The contribution of this demand point to Δf is negative. The

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

minimum distance vector and the facility assignment vector are updated.

- If the distance is greater than the minimum distance then
  (a) If the facility is not the one to which the minimum distance is obtained, there is no change in Δf.
  (b) Otherwise, a new minimum distance is found for this demand point by calculating all distances, and Δf is changed. In this case the contribution of this demand point to Δf is positive.

Also, in most calculations using the square of the distance suffices. When finding the minimum distance, squares of distances are compared, and the square root is applied only on the minimum distance. In addition to the vector containing the minimum distances for all demand points, another vector containing the squares of the minimum distances is maintained so comparison between distances is done by comparing squares of distances. This saves an appreciable amount of computer time.

Table 1
Comparing three methods.

p	ALT						IALT					
	(1)	(2)	(3)	(4)	(5)	(6)	(1)	(2)	(3)	(4)	(5)	(6)
5	674	261	4.89	51.31	0.1384	1.1E−44	4339	14	2.95	26.08	0.1476	1.2E−50
10	54	4701	10.31	68.85	0.1233	9.4E−36	144	424	6.25	42.82	0.1391	4.0E−45
15	10	8727	17.95	60.38	0.1010	1.6E−24	29	3191	11.66	37.56	0.1341	5.0E−42
20	0	9164	21.50	69.36	0.0986	2.0E−23	6	4613	11.41	42.71	0.1037	9.7E−26
25	0	9035	27.37	81.94	0.0560	1.0E−08	11	4627	11.65	52.82	0.0400	3.1E−05
p	Simulated annealing											
5	10,000	1	0	0	–	–						
10	1301	10	0.41	4.64	0.0038	0.3514						
15	2	16	0.33	4.15	0.0143	0.0763						
20	2189	66	1.01	7.57	0.0336	0.0004						
25	5890	120	1.08	11.04	0.0137	0.0855						

(1) Number of times optimum encountered.
(2) Number of different local optima encountered.
(3) Percent of average solution above the optimum.
(4) Percent of maximum solution above the optimum.
(5) Correlation between initial value of the objective function and the final value.
(6) One-tail p-value of the correlation being non-negative.

Table 2
Results for the IALT algorithm.

n	p	Best known	[a]	[b]	Time[c]	n	p	Best known	[a]	[b]	Time[c]
100	2	14.13353	708,152	1.27	7.50	600	2	89.74039	508,722	0.52	39.99
100	3	10.08185	1,000,000	0.00	4.32	600	3	68.62544	1,000,000	0.00	34.01
100	5	7.19404	631,946	3.96	5.57	600	5	50.86844	187,984	0.79	31.87
100	10	4.64606	25,258	5.48	6.38	600	10	34.96318	25,430	1.89	62.74
100	20	2.65704	848	9.62	10.25	600	20	23.02266	117	3.09	112.41
100	30	1.66946	2	17.79	13.40	600	30	17.93616	1	4.59	171.40
200	2	29.83759	494,318	0.90	12.06	700	2	105.06867	434,335	0.26	45.20
200	3	22.85924	507,950	0.23	10.88	700	3	80.78105	981,551	0.10	60.19
200	5	16.69636	650,513	1.70	10.63	700	5	59.55514	139,885	0.87	37.14
200	10	10.70744	53,267	3.88	14.77	700	10	41.23822	28,270	1.50	78.40
200	20	6.64971	17	7.03	23.78	700	20	27.25898	225	2.63	144.33
200	30	4.92899	6	8.65	35.86	700	30	21.38212	1	4.00	225.63
300	2	42.93134	393,289	0.10	16.85	800	2	117.73201	165,401	0.36	52.61
300	3	33.59106	982,640	0.16	24.51	800	3	90.61576	556,247	0.24	53.71
300	5	25.18148	458,833	1.01	18.34	800	5	67.60815	243,585	0.93	47.85
300	10	16.84649	107,256	2.82	25.10	800	10	46.98038	14,832	1.41	100.41
300	20	10.76098	166	5.52	44.01	800	20	31.16189	2,033	2.61	176.51
300	30	8.22219	0	6.86	65.52	800	30	24.84122	2	3.55	274.14
400	2	58.66174	300,605	0.37	25.96	900	2	132.89287	254,350	0.34	61.58
400	3	45.38518	423,633	0.36	21.57	900	3	103.06271	528,132	0.10	65.63
400	5	33.51992	369,597	0.88	22.75	900	5	76.12663	274,136	0.81	58.35
400	10	22.85775	14,686	2.04	38.33	900	10	53.49045	52,308	1.20	115.80
400	20	14.95262	36	3.56	64.41	900	20	35.40132	249	2.34	211.74
400	30	11.54676	1	5.64	96.17	900	30	28.20450	0	3.11	341.89
500	2	73.96976	487,276	0.61	27.53	1000	2	145.85870	82,539	0.54	63.45
500	3	56.15244	974,430	0.22	27.92	1000	3	114.20354	95,729	0.19	69.00
500	5	41.35487	95,467	0.68	28.22	1000	5	84.50642	176,031	0.53	62.93
500	10	28.41502	183,785	2.01	51.36	1000	10	59.30524	5,407	0.96	130.40
500	20	18.83322	23	3.59	89.03	1000	20	39.27941	234	2.27	245.61
500	30	14.47131	3	5.52	134.76	1000	30	31.44394	0	2.98	394.91

[a] Number of times best known solution found.
[b] Percent of average solution above the best known solution.
[c] Time in minutes for all one million runs.

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

2.5. Additional improvements

We suggest applying the following minor modifications to improve the performance of the algorithms.

Empty subsets: When a facility in the IALT process has no demand points closest to it and therefore the subset associated with it is empty, a new location for that facility is randomly generated. This is quite common when n/p is relatively small. For further discussion of this degeneracy problem, see Brimberg and Mladenović [9].

Optimal demand point: Convergence of any Weiszfeld scheme can be slow when the solution is on a demand point. Such a occurrence is quite common (Drezner and SimchiLevi [19] proved that the probability that a solution is on a demand point is about $1/n$). Thus, the number of iterations in such cases can be quite large. When an iteration of the Weiszfeld procedure is implemented, the distances to all demand points from the current iteration are calculated. If the distance to a demand point is less than $0.001S$, where S is the side of the square, we test whether this demand point is the optimal solution by the formula in [18,19,34]. If it is optimal, the demand point is returned as the solution and if it is not, that demand point is flagged and if the distance to that demand point is less than $0.001S$ in a subsequent iteration, the check for optimality is avoided.

Improved accelerated Weiszfeld: The idea behind the accelerated Weiszfeld [17] is to estimate the limit of the iterations by two successive Weiszfeld iterations. Suppose that the original Weiszfeld iterations for the x-coordinate are $x^{(1)},x^{(2)},\ldots$. If the differences $x^{(k+1)}-x^{(k)}$ form a geometric series, the limit point of the sequence can be estimated. Two successive Weiszfeld iterations are performed. Suppose that the present iteration is (x_1,y_1). The next Weiszfeld iterate is (x',y'). The next iterate is $(x_2,y_2)=[x_1+2(x'-x_1),y_1+2(y'-y_1)]$. We then calculate (x_3,y_3) by replacing (x_1,y_1) with (x_2,y_2) in the Weiszfeld iteration. We then calculate the estimate of the geometric series quotient $\theta_x=(x_3-x_2)/(x_2-x_1)$ and similarly θ_y. The next iterate (the estimate of the limit point of the Weiszfeld iterations) is then $x_1+(1/(1-\theta_x))(x_2-x_1)$ if $\theta_x\le 0.5$ and $|x_2-x_1|>10^{-10}S$. Otherwise, the geometric series assumption is ignored, and the next iterate is x_3. The same is performed for y. We then proceed to the next iteration by performing a sequence of two Weiszfeld iterations using the limit estimate as (x_1,y_1). The algorithm terminates when the distance between two successive Weiszfeld iterations is less than $10^{-7}S$.

Initial solution: p-Median solutions tend to be spaced uniformly with no two facilities close to one another. We therefore generated sparse initial solutions. The first facility is randomly generated in the square. Each subsequent facility is generated as follows. K randomly generated locations are drawn (we used $K=3$). The selected location is the one with the largest minimum distance to already selected facilities. The process continues until p locations are selected.

Individual grid: We experimented with each facility having its individual grid. That means that the starting solution for any

Table 3
Heuristic results (grid search + IALT) for small values of n.

n	p	Descent			Tabu search			Simulated annealing		
		[a]	[b]	Time[c]	[a]	[b]	Time[c]	[a]	[b]	Time[c]
100	2	6966	1.32	0.19	82	0.78	0.25	100	0.00	2.58
100	3	10,000	0.00	0.27	100	0.00	0.47	100	0.00	3.32
100	5	6165	4.27	0.41	100	0.00	1.01	100	0.00	4.52
100	10	414	5.72	0.76	16	3.13	2.93	28	0.35	7.33
100	20	5	10.27	1.79	7	2.07	9.43	53	0.21	12.59
100	30	0	22.80	2.96	1	3.47	19.45	3	1.18	17.38
200	2	5190	0.82	0.35	100	0.00	0.51	100	0.00	4.91
200	3	5604	0.14	0.56	100	0.00	0.93	100	0.00	6.16
200	5	6461	1.75	0.81	84	0.64	1.98	100	0.00	8.30
200	10	351	4.05	1.53	60	1.20	5.73	100	0.00	13.09
200	20	0	7.21	3.19	1	1.94	18.16	3	0.11	21.91
200	30	0	8.76	5.47	4	1.73	36.90	7	0.79	30.40
300	2	3800	0.11	0.58	100	0.00	0.77	100	0.00	7.03
300	3	9903	0.09	0.95	100	0.00	1.40	100	0.00	8.78
300	5	4772	1.04	1.20	100	0.00	2.97	100	0.00	12.06
300	10	958	2.95	2.34	80	0.44	8.53	100	0.00	18.56
300	20	6	5.45	5.00	4	1.28	26.88	56	0.24	31.07
300	30	0	6.68	8.11	2	1.29	54.83	16	0.18	43.30
400	2	4794	0.32	0.74	100	0.00	1.02	100	0.00	9.19
400	3	5115	0.24	1.13	100	0.00	1.86	99	0.00	11.44
400	5	4068	0.80	1.63	100	0.00	3.96	100	0.00	15.36
400	10	277	2.06	3.21	80	0.19	11.30	97	0.02	23.88
400	20	0	3.49	6.66	1	0.86	35.55	2	0.13	39.93
400	30	0	5.48	10.66	1	0.77	72.25	6	0.16	55.56
500	2	4462	0.61	0.83	84	0.16	1.28	100	0.00	11.28
500	3	9781	0.19	1.42	100	0.00	2.32	100	0.00	13.99
500	5	2131	0.63	2.02	97	0.03	4.95	96	0.00	18.83
500	10	1954	2.01	4.11	75	0.54	14.11	100	0.00	28.99
500	20	0	3.50	8.36	1	0.84	44.19	15	0.04	48.55
500	30	0	5.20	13.66	2	1.07	89.91	9	0.19	67.54

[a] Number of times best known solution found.
[b] Percent of average solution above the best known solution.
[c] Time in minutes for all runs.

Table 4
Heuristic results (grid search + IALT) for large values of n.

n	p	Descent			Tabu search			Simulated annealing		
		[a]	[b]	Time[c]	[a]	[b]	Time[c]	[a]	[b]	Time[c]
600	2	4851	0.55	1.08	100	0.00	1.54	100	0.00	13.16
600	3	10,000	0.00	1.75	100	0.00	2.77	100	0.00	16.35
600	5	2022	0.73	2.31	100	0.00	5.93	100	0.00	22.14
600	10	372	1.92	4.91	44	0.29	16.91	40	0.01	34.30
600	20	2	3.01	10.15	11	0.68	52.89	64	0.07	56.90
600	30	0	4.30	16.47	1	0.59	107.48	3	0.15	79.12
700	2	3843	0.27	1.20	44	0.22	1.79	100	0.00	15.18
700	3	9889	0.06	2.09	100	0.00	3.24	100	0.00	18.82
700	5	1911	0.83	2.64	98	0.00	6.94	98	0.00	25.47
700	10	387	1.46	5.75	60	0.19	19.64	75	0.00	39.51
700	20	4	2.51	12.02	16	0.35	61.52	48	0.10	65.39
700	30	0	3.70	19.45	4	0.28	125.22	24	0.09	90.96
800	2	2379	0.31	1.40	81	0.15	2.06	100	0.00	17.27
800	3	6113	0.19	2.23	100	0.00	3.71	100	0.00	21.29
800	5	3491	0.75	3.20	83	0.25	7.92	100	0.00	28.69
800	10	441	1.37	6.76	61	0.38	22.45	89	0.00	44.60
800	20	32	2.43	14.15	12	0.22	70.19	50	0.06	73.51
800	30	0	3.20	22.14	7	0.19	142.42	32	0.05	102.69
900	2	3075	0.30	1.60	86	0.06	2.31	91	0.00	19.17
900	3	5186	0.10	2.48	100	0.00	4.21	100	0.00	23.92
900	5	2994	0.78	3.45	86	0.18	8.90	100	0.00	31.99
900	10	632	1.14	7.46	76	0.18	25.34	100	0.00	49.84
900	20	10	2.18	15.91	9	0.20	78.79	27	0.09	81.87
900	30	0	2.82	25.56	7	0.18	160.01	16	0.02	114.03
1000	2	3526	0.44	1.75	59	0.10	2.57	69	0.00	21.24
1000	3	1845	0.17	2.70	65	0.00	4.66	81	0.00	26.40
1000	5	2300	0.47	4.01	97	0.01	9.89	99	0.00	35.37
1000	10	88	0.91	8.28	46	0.06	28.11	60	0.01	54.82
1000	20	6	2.07	17.98	15	0.31	87.67	46	0.11	90.07
1000	30	0	2.68	28.40	1	0.08	177.43	3	0.06	125.62

[a] Number of times best known solution found.
[b] Percent of average solution above the best known solution.
[c] Time in minutes for all runs.

Table 5
Summary of results.

Property	IALT	Desc.	Tabu	SA
# of times each problem solved	1,000,000	10,000	100	100
% of runs that best known (BK) found	24.4	26.4	57.5	70.1
# of Problems BK found at least once	57	47	60	60
% of average solution above BK	2.45	2.49	0.46	0.07
Average run time (min.)	75.79	5.67	28.67	33.96

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

facility can be anywhere in the square but moves of facilities can only be ($\pm 0.01S$, $\pm 0.01S$). This variant produced better results in experiments.

2.6. The pseudo-code for the simulated annealing algorithm

The following is a pseudo-code of the simulated annealing algorithm:

1. Randomly select a vector $\mathbf{X}=\{X_j\}$ in the square enclosing all demand points. Set $\overline{F}=F(\mathbf{X})$. Set the iteration counter $IT=0$, $T=T_0/\alpha$. Set $F_{best}=\overline{F}$.
2. Set $IT=IT+1$ and $T=T\times\alpha$. If $IT>N$ stop with F_{best} as the solution.
3. Randomly select a facility $j\in[1,\ldots,p]$.
4. Randomly select $i_x\in[-1,0,1]$ and $i_y\in[-1,0,1]$. If $i_x=i_y=0$ go to Step 4.
5. Set $X_j'=(x_j+i_x\Delta x, y_j+i_y\Delta y)$ defining $\mathbf{X}'$. If X_j' is outside the square enclosing all demand points go to Step 4.
6. If $F(\mathbf{X}')\le\overline{F}$ then
   (a) if $F(\mathbf{X}')<F_{best}$ set $F_{best}=F(\mathbf{X}')$.
   (b) Set $\mathbf{X}=\mathbf{X}'$ and $\overline{F}=F(\mathbf{X}')$.
   (c) Go to Step 2.
7. Otherwise, calculate $\Delta=[F(\mathbf{X}')-\overline{F}]/T$ and randomly generate $0\le u\le 1$.
8. If $e^{-\Delta}\le u$ go to Step 2. Otherwise, go to Step 6b.

2.7. Testing the algorithms on a small problem

We tested three algorithms on the well-studied $n=50$ problems from [20] for $p=5,10,15,20,25$ and for which the optimal solutions are known [7]. In Table 1 we report for ALT, IALT, and simulated annealing+IALT replicated 10,000 times each from random starting solutions. We report for each method: the number of times the optimum was encountered, the number of different local optima encountered, the percentage of the average solution value above the optimum, and the percentage of the maximum solution above the optimum. We also investigated whether a better starting solution tends to yield a better final solution by calculating the correlation coefficient between the initial value of the objective function and the final value for these 10,000 runs. The IALT algorithm clearly performed better than ALT. The simulated annealing algorithm performed best but required longer computer time. For the ALT and IALT algorithms there is a highly significant correlation between the initial and final solutions. The correlation for simulated annealing is relatively small. This suggests that if one applies the ALT and IALT algorithms, better results are expected if the best among several randomly generated solutions is selected as the starting solution.

Random initial solutions are generally very poor in quality. Hence, starting the local search from such solutions may lead the descent path to an inferior local minimum. On the other hand, by starting the local search from a "high quality" point as obtained by our simulated annealing heuristic on the grid, these inferior local minima are avoided.

Table 6
Improved best known results.

n	p	From [7]	New BK	% improv.
287	7	8160.3230	8160.3203	0.0000
287	30	2716.9071	2716.9038	0.0001
1060	5	1,851,879.9	1,851,877.3	0.0001
1060	15	980,132.1	980,131.7	0.0000
1060	20	828,802.0	828,685.7	0.0140
1060	25	722,061.2	721,988.2	0.0101
1060	30	638,263.0	638,212.3	0.0079
1060	35	577,526.6	577,496.7	0.0052
1060	40	529,866.2	529,660.1	0.0389
1060	45	489,650.0	489,483.8	0.0339
1060	55	422,770.0	422,647.2	0.0290
1060	60	397,784.4	397,718.0	0.0167
1060	65	376,759.5	376,639.3	0.0319
1060	70	357,385.0	357,381.1	0.0011
1060	75	340,242.0	340,123.5	0.0348
1060	80	326,053.2	325,990.5	0.0192
1060	85	313,738.2	313,463.0	0.0877
1060	90	302,837.0	302,600.9	0.0780
1060	95	292,875.1	292,343.1	0.1816
1060	100	283,113.0	282,624.9	0.1724
1060	105	274,576.0	273,611.0	0.3515
1060	110	265,801.0	265,242.3	0.2102
1060	115	257,605.0	256,940.1	0.2581
1060	120	249,584.0	249,142.2	0.1770
1060	125	242,930.0	242,267.4	0.2728
1060	130	236,154.0	235,653.0	0.2121
1060	135	230,431.0	229,399.4	0.4477
1060	140	224,504.0	223,519.0	0.4387
1060	145	218,279.0	217,931.3	0.1593
1060	150	212,926.0	212,717.4	0.0980

3. Computational experiments

Solution methods were programmed in Fortran using double precision arithmetic. The programs were compiled by an Intel 11.1 Fortran Compiler with no parallel processing and run on a desktop with the Intel 870/i7 2.93 GHz CPU Quad processor and 8 GB RAM. Only one thread was used.

3.1. Randomly generated problems

Sixty problems were randomly generated for $n=100,200,\ldots,1000$ and $p=2,3,5,10,20,30$ (the same n demand points were used for all values of p). Demand points were generated in a unit

Table 7
Results for the $n=50$ problems.

p	IALT 100 restarts			IALT 1000 restarts			Simul. anneal.+IALT		
	[a]	[b]	Time[c]	[a]	[b]	Time[c]	[a]	[b]	Time[c]
2	0.00	0.00	0.17	0.00	0.00	1.50	0.00	0.00	77.24
3	0.00	0.00	0.16	0.00	0.00	1.59	0.00	0.00	102.41
4	0.00	0.00	0.17	0.00	0.00	1.76	0.00	0.01	124.93
5	0.00	0.00	0.22	0.00	0.00	2.25	0.00	0.00	147.41
6	0.00	0.00	0.23	0.00	0.00	2.28	0.00	0.00	167.16
7	0.00	0.00	0.22	0.00	0.00	2.20	0.00	0.00	191.62
8	0.00	0.00	0.22	0.00	0.00	2.12	0.00	0.26	212.66
9	0.00	0.00	0.20	0.00	0.00	2.17	0.00	0.17	234.05
10	0.00	0.01	0.22	0.00	0.00	2.28	0.00	0.36	254.28
11	0.00	0.00	0.25	0.00	0.00	2.36	0.00	0.46	274.47
12	0.00	0.00	0.25	0.00	0.00	2.45	0.00	0.52	292.97
13	0.00	0.00	0.25	0.00	0.00	2.48	0.00	0.36	312.53
14	0.00	0.59	0.25	0.00	0.00	2.51	0.00	0.56	330.24
15	0.00	0.48	0.25	0.00	0.00	2.51	0.00	0.46	349.99
16	0.00	0.60	0.27	0.00	0.01	2.48	0.00	0.45	368.21
17	0.00	0.33	0.25	0.00	0.06	2.50	0.00	0.55	387.90
18	0.00	0.90	0.25	0.00	0.03	2.50	0.00	0.85	407.18
19	0.00	1.49	0.27	0.00	0.00	2.54	0.00	1.56	427.13
20	0.05	0.58	0.25	0.00	0.03	2.61	0.00	1.06	445.62
21	0.07	0.42	0.27	0.00	0.04	2.64	0.00	0.79	465.79
22	0.00	0.57	0.28	0.00	0.09	2.68	0.00	0.87	484.24
23	0.00	0.51	0.27	0.00	0.10	2.73	0.00	0.70	503.77
24	0.00	0.56	0.28	0.00	0.02	2.81	0.00	0.51	521.70
25	0.00	1.10	0.28	0.00	0.00	2.90	0.00	1.19	540.92
Ave ⇒	0.01	0.34	0.24	0.00	0.02	2.37	0.00	0.49	317.68

[a] Percent of best found solution above best known.
[b] Percent of average solution above best known.
[c] Time in seconds for all runs.

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

square and weights were generated in the interval [0, 1]. For the transfer follow-up we used $L=10$.

We first tested the IALT approach which includes (by definition) the transfer follow-up. The procedure is so efficient that we could run one million replications for each problem. The results are summarized in Table 2. The procedure performed very well (we may say surprisingly well) for problems with $p \leq 10$. The procedure did not work so well on problems with $p=30$. Results are much improved when the transfer follow-up is implemented (results without the transfer follow-up are not reported).

We then tested the heuristic algorithms on the grid followed by the IALT approach. The descent algorithm was repeated 10,000 times for each problem starting from randomly generated starting solutions. The tabu search and simulated annealing were repeated 100 times each. The results are reported in Tables 3 and 4.

A statistical summary of the results is given in Table 5. We observe the following:

1. The IALT method performed very well.
2. The descent algorithm did not provide significantly better starting solutions to IALT as compared with a random starting solution.
3. Simulated annealing performed better than tabu search (both were followed by IALT).
4. The best known solution was obtained at least once by simulated annealing and tabu search in all 60 problem instances. IALT found the best known solution almost all the time (57 out of 60 instances).
5. The deviation of the average solution from the best known was very tight for simulated annealing.

3.2. Test problems used in [7]

The problems used in Brimberg et al. [7] for testing their algorithms are with $n=50$ demand points [20], $n=287$ [4], $n=654$ [40], and $n=1060$ [40]. Each of these problems was solved for varying values of p by

- IALT with 100 restarts repeated 10 times for a total of 1000 replications leading to 10 solutions, each with the minimum value of the objective function among these 100 restarts.
- IALT with 1000 restarts repeated 10 times for a total of 10,000 replications leading to 10 solutions.
- Simulated annealing repeated 100 times.

In Table 6 we list problem instances where our procedure yielded results better than the best known results reported in Brimberg et al. [7]. Note that the two improvements in the objective value for $n=287$ are likely due to a typo in [7] or

Table 8
Results for the $n=287$ problems.

p	IALT 100 restarts			IALT 1000 restarts			Simul. anneal.+IALT		
	[a]	[b]	Time[c]	[a]	[b]	Time[c]	[a]	[b]	Time[c]
2	0.00	0.00	1.29	0.00	0.00	12.65	0.00	0.08	189.24
3	0.00	0.07	1.06	0.00	0.00	10.53	0.00	0.05	242.08
4	0.00	2.59	1.11	0.00	0.00	11.08	0.00	1.50	300.57
5	0.00	0.54	1.29	0.00	0.00	13.18	0.00	0.31	365.78
6	2.64	4.76	1.51	0.00	2.45	15.46	0.00	0.04	412.25
7	2.97	5.88	1.68	2.15	4.26	16.77	0.00	0.02	492.84
8	6.21	7.28	1.70	3.07	6.25	17.60	0.00	0.20	536.30
9	6.94	10.76	1.78	6.25	7.45	17.89	0.00	0.22	584.33
10	9.03	12.66	1.89	8.89	9.80	18.81	0.00	0.33	628.40
11	12.83	14.68	1.97	11.51	12.48	19.75	0.00	0.24	684.59
12	14.48	17.31	2.09	12.16	14.67	20.89	0.00	0.16	730.23
13	17.65	20.40	2.17	15.64	17.21	21.82	0.00	0.58	788.29
14	16.00	20.98	2.26	16.00	18.85	22.92	0.08	0.48	837.93
15	20.34	23.72	2.40	19.39	20.71	23.84	0.16	0.72	902.62
16	21.43	26.20	2.53	19.20	23.00	25.15	0.36	0.98	948.72
17	24.46	30.44	2.70	23.92	26.76	26.71	0.09	1.07	999.58
18	28.96	32.34	2.84	27.42	29.93	28.19	0.02	1.02	1052.07
19	32.35	36.07	2.96	28.32	31.38	29.72	0.26	0.99	1106.06
20	36.22	40.50	3.15	32.14	34.83	31.62	0.00	0.95	1157.34
25	54.38	60.14	4.17	47.68	50.93	42.34	0.69	1.82	1431.89
30	73.43	79.70	5.55	60.95	67.48	55.29	0.00	0.87	1699.69
35	86.18	97.12	6.83	82.36	88.60	68.94	0.00	1.85	1982.71
40	97.43	112.57	8.21	91.87	103.60	82.51	0.66	2.38	2251.31
45	123.65	133.56	9.56	115.24	126.14	96.61	0.61	2.90	2546.95
50	139.46	157.18	11.01	135.37	142.36	111.15	1.06	2.74	2803.84
55	176.67	186.61	12.64	149.26	165.89	126.00	0.05	2.46	3101.30
60	197.73	210.48	14.18	180.56	188.67	141.90	0.30	1.85	3368.28
65	218.90	235.99	16.75	178.81	208.53	167.81	0.61	2.27	3685.77
70	223.37	251.07	18.92	202.93	227.13	188.59	0.35	1.80	3948.46
75	247.27	262.09	21.23	223.45	242.15	212.74	0.02	1.30	4268.94
80	282.17	301.41	23.82	249.81	262.89	239.76	0.42	1.49	4535.46
85	268.05	309.98	26.75	240.13	271.75	269.10	0.18	1.59	4863.86
90	284.20	320.83	30.15	281.72	292.48	302.14	0.16	1.64	5131.37
95	294.32	339.44	33.59	271.19	301.60	337.20	0.11	1.73	5473.79
100	292.43	338.64	37.35	278.07	306.77	375.71	0.29	1.43	5744.30
Ave ⇒	94.63	105.83	9.12	86.16	94.49	91.50	0.19	1.14	1994.20

[a] Percent of best found solution above best known.
[b] Percent of average solution above best known.
[c] Time in seconds for all runs.

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

Table 9
Results for the $n=654$ problems.

p	IALT 100 restarts			IALT 1000 restarts			Simul. anneal.+IALT		
	[a]	[b]	Time[c]	[a]	[b]	Time[c]	[a]	[b]	Time[c]
2	0.00	0.00	2.68	0.00	0.00	27.16	0.00	0.00	500.47
3	0.00	0.00	2.50	0.00	0.00	25.41	0.00	0.12	582.41
4	0.00	0.00	2.71	0.00	0.00	27.07	0.00	12.84	679.70
5	0.00	0.00	2.62	0.00	0.00	26.47	0.00	0.00	795.42
6	0.00	0.00	2.20	0.00	0.00	23.06	0.00	0.00	919.88
7	0.00	0.03	2.11	0.00	0.00	21.54	0.00	0.60	1054.35
8	0.37	0.65	2.12	0.00	0.16	21.06	0.00	0.89	1171.43
9	0.40	0.40	2.07	0.00	0.36	20.72	0.00	0.94	1307.49
10	0.00	5.17	2.09	0.00	0.00	21.03	0.00	0.62	1410.50
11	9.03	11.45	2.17	0.00	7.18	21.42	0.00	0.30	1548.04
12	0.00	9.50	2.23	0.00	2.81	22.11	0.00	0.00	1673.66
13	2.79	8.59	2.29	0.55	3.47	23.15	0.00	0.30	1802.58
14	0.53	7.74	2.43	0.53	3.15	24.35	0.00	0.45	1912.18
15	6.38	7.90	2.54	5.78	6.35	25.58	0.00	0.64	2045.77
20	14.58	18.40	3.56	12.72	14.85	35.26	0.14	1.31	2679.89
25	21.63	26.05	4.98	17.18	21.66	49.83	0.00	1.60	3338.50
30	23.55	28.62	7.13	23.47	25.37	71.99	0.00	1.01	3968.63
35	19.30	27.44	10.55	18.06	21.25	105.22	0.02	1.13	4633.95
40	15.82	21.68	15.43	14.50	16.87	153.40	0.28	0.87	5280.45
45	12.78	16.89	21.67	9.59	13.06	217.04	0.45	1.61	5961.61
50	13.11	16.69	29.87	11.25	12.97	298.52	0.28	1.60	6607.92
55	14.80	16.24	39.58	9.91	13.38	398.30	0.30	1.47	7294.11
60	12.42	15.14	51.84	11.10	12.67	518.14	0.18	1.23	7942.67
65	15.31	16.90	69.64	10.13	12.63	697.62	0.13	1.30	8644.08
70	11.05	15.51	87.10	9.77	12.26	872.53	0.17	0.99	9300.08
75	10.47	13.58	107.67	9.05	11.09	1073.52	0.19	0.81	10,008.28
80	9.78	13.33	130.57	7.57	9.64	1306.24	0.44	1.34	10,674.34
85	9.13	11.85	157.56	7.23	9.55	1568.62	0.42	1.32	11,389.63
90	9.84	11.05	186.67	7.17	8.98	1864.93	0.19	1.10	12,052.14
95	6.66	9.77	220.51	6.66	8.09	2205.71	0.47	1.30	12,783.75
100	8.27	10.01	257.46	6.36	8.10	2580.77	0.42	1.31	13,469.77
Ave ⇒	8.00	10.99	46.28	6.41	8.25	462.83	0.13	1.26	4949.47

[a] Percent of best found solution above best known.
[b] Percent of average solution above best known.
[c] Time in seconds for all runs.

Table 10
Results for the $n=1060$ problems.

p	IALT 100 restarts			IALT 1000 restarts			Simul. anneal.+IALT		
	[a]	[b]	Time[c]	[a]	[b]	Time[c]	[a]	[b]	Time[c]
5	0.00	0.00	12.78	0.00	0.00	126.95	0.00	0.00	1318.96
10	0.00	0.00	19.92	0.00	0.00	196.70	0.00	0.00	2277.52
15	0.00	0.10	29.53	0.00	0.01	292.97	0.00	0.01	3282.23
20	0.01	0.48	38.53	0.01	0.15	383.41	0.00	0.04	4268.61
25	0.37	0.70	50.94	0.02	0.17	508.52	0.00	0.14	5277.34
30	0.30	1.21	64.42	0.13	0.64	646.12	0.00	0.07	6249.32
35	1.39	2.28	79.27	0.41	1.25	792.97	0.00	0.13	7261.44
40	0.48	2.25	96.36	0.48	1.38	953.81	0.00	0.25	8248.58
45	1.64	2.38	113.31	1.05	1.74	1135.38	0.00	0.16	9280.23
50	2.96	3.69	133.14	1.92	2.57	1326.17	0.01	0.28	10,271.78
55	3.60	4.55	153.98	2.40	3.05	1522.50	0.00	0.29	11,451.50
60	2.81	4.24	173.42	2.45	2.98	1736.08	0.00	0.18	12,457.52
65	3.36	4.29	200.98	1.99	2.83	2029.95	0.00	0.17	13,526.44
70	3.55	4.46	224.45	2.64	3.22	2258.70	0.00	0.21	14,566.36
75	3.49	4.59	250.52	2.75	3.53	2499.25	0.00	0.18	15,652.84
80	3.66	4.65	274.27	2.50	3.46	2731.05	0.00	0.15	16,658.05
85	3.69	4.91	298.16	3.27	3.71	3012.03	0.00	0.21	17,747.58
90	4.13	4.92	325.48	2.64	3.50	3263.67	0.00	0.19	18,791.49
95	3.79	4.56	352.98	2.88	3.63	3568.02	0.00	0.21	19,895.21
100	4.01	4.73	379.28	3.12	3.86	3856.80	0.00	0.28	20,931.57
Ave ⇒	2.16	2.93	95.45	1.38	2.09	955.03	0.00	0.16	10,970.73

[a] Percent of best found solution above the new best known.
[b] Percent of average solution above the new best known.
[c] Time in seconds for all runs.

rounding errors. However, some significant improvements are obtained for the $n=1060$ problem sets. In Tables 7–10 we give a summary of all our results for the four problems from Brimberg et al. [7].

In Table 11 we compare the performance of the alternate approach used in Brimberg et al. [7] (MALT) to the IALT approach proposed in this paper. Except for the $n=287$ problems where IALT performed unusually poorly, the IALT algorithm performed much better in a much shorter run time for the 100 restarts case.

3.2.1. Discussion

As observed in Tables 7–10, the IALT algorithm performed very well on the majority of test problems from [7]. We see in fact that this simple local search matches the sophisticated applications of metaheuristics in [7] on several of the test instances! The results also compare very favorably with the standard alternate procedure of Cooper for $n=50$, 654 and 1060 (see Table 11). Run times for the simulated annealing+IALT gave the best results. Run times may look high. However, run times are for all 100 runs for each value of p. For example, the average run time for each value of p for the largest problem of $n=1060$ reported in Table 10 is about 3 h which is less than 2 min for one run of simulated annealing+IALT.

To demonstrate further the complexity of model (1), consider the following quote taken from Brimberg et al. [5] on the same well-known 50 customer problems studied above:

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

Table 11
Comparison of averages between the IALT and MALT in [7].

n	IALT (100 restarts)			IALT (1000 restarts)			MALT (100 restarts) [7]		
	%Best	%Ave.	Time[a]	%Best	%Ave.	Time[a]	%Best	%Ave.	Time[a]
50	0.01	0.34	0.24	0.00	0.02	2.37	1.12	3.10	3.59
287	94.63	105.83	9.12	86.16	94.49	91.50	7.32	10.76	50.38
654	8.00	10.99	46.28	6.41	8.25	462.83	27.34	34.15	142.20
1060	2.16	2.93	95.45	1.38	2.09	955.03	21.47[b]	23.61[b]	725.96

[a] Time in seconds for all runs.
[b] 0.04% added to compare with new best known solution.

It is interesting to note that of the 10,000 iterations of MLS (i.e., Cooper's procedure), an optimal solution was obtained 690, 34, 1 times for $p=5$, 10, 15, respectively, and the worst deviation from the optimal value was, respectively, 46.74%, 65.80%, 70.27%. In all 272, 3008, and 3363 different local solutions were obtained, respectively, for $p=5$, 10, 15.

We extended this analysis, reported in Table 1, to IALT and simulated annealing up to $p=25$ and obtained better characteristics for the methods proposed in this paper.

Thus, it appears that the follow-up transfer in IALT is highly effective in reducing the number of local minima and increasing their quality. Meanwhile we must also recognize the inferior results obtained by IALT for $n=287$ (Table 11). It should be noted that this problem has an unusual weight distribution and most demand points are concentrated near the center of the area, which may diminish the usefulness of the transfer follow-up step. The solution is not sparsely distributed and thus a sparsely distributed starting solution hinders the performance of IALT. When the value of L in the transfer follow-up was increased from $L=10$ to $L=n=287$, the percentages decreased to 30.24% and 36.35% for IALT with 100 restarts but run time increased by about eight fold.

The simulated annealing algorithm is seen to obtain consistently good results in all cases (see Tables 7–10). The best found result was in one worst case, 1% above the best known (Table 8). The average performance in the worst case was 3% above the best known (again Table 8). With the exception of the $n=287$ problems, and only one instance of the $n=1060$ problems, this heuristic was always able to find the best known solution, and in several of the instances with $n=1060$, even improved the state-of-the-art (see Tables 6–10). Thus, the importance of finding good starting solutions for IALT is confirmed.

4. Conclusions

This paper develops some powerful heuristics for solving the p-median problem in the plane. First we show that a simple add-on step (IALT) to the classical alternating algorithm attributed to Cooper [11,12] (ALT), and still widely used as a local search (e.g., see [7]), greatly enhances this method. Furthermore, we provide some ideas to improve the efficiency of the local search. This improvement in efficiency is important, as the local search can be used many times as a subroutine in more sophisticated algorithms. We also show that finding good starting solutions on a grid representation of the problem can be very effective when combined with the improved local search. In fact, the combined approach developed here was able to find or improve best known solutions on several test problems.

We found that there is a highly significant correlation between the quality of the starting solution and the final solution of ALT or IALT. This suggests that it is beneficial to generate good starting solutions for each of these heuristics.

Further research in this area may include more detailed studies concerning the effect of grid dimensions on the quality of the solution and on computing times. Other grid types (e.g., triangular) should also be examined. Finally, the procedures developed here may form the basis of a general methodology for solving other types of location–allocation problems. We plan to apply this approach, for example, to the continuous p-center problem. Several other continuous location models come to mind.

References

[1] Alp O, Drezner Z, Erkut E. An efficient genetic algorithm for the p-median problem. Annals of Operations Research 2003;122:21–42.
[2] Arthanari TS, Dodge Y. Mathematical programming in statistics. New York: John Wiley & Sons, Inc.; 1993.
[3] Babich G. An efficient algorithm for solving the rectilinear location–allocation problem. Environment and Planning A 1978;10:1387–95.
[4] Bongartz I, Calamai PH, Conn AR. A projection method for ℓ_p norm location–allocation problems. Mathematical Programming 1994;66:238–312.
[5] Brimberg J, Hansen P, Mladenović N. Attraction probabilities in variable neighborhood search. 4OR—Quarterly Journal of Operations Research 2010;8:181–94.
[6] Brimberg J, Hansen P, Mladenović N, Salhi S. A survey of solution methods for the continuous location–allocation problem. International Journal of Operations Research 2008;5:1–12.
[7] Brimberg J, Hansen P, Mladenović N, Taillard E. Improvements and comparison of heuristics for solving the uncapacitated multisource Weber problem. Operations Research 2000;48:444–60.
[8] Brimberg J, Hodgson MJ. Heuristics for location models. In: Eiselt HA, Marianov V, editors. Foundations of location analysis: industrial series in operations research & management science, vol. 155. New York, NY: Springer; 2011. p. 335–55 (Chapter 15).
[9] Brimberg J, Mladenović N. Degeneracy in the multi-source Weber problem. Mathematical Programming 1999;85:213–20.
[10] Chen R. Solution of minisum and minimax location–allocation problems with Euclidean distances. Naval Research Logistics Quarterly 1983;30:449–59.
[11] Cooper L. Location-allocation problems. Operations Research 1963;11:331–43.
[12] Cooper L. Heuristic methods for location–allocation problems. SIAM Review 1964;6:37–53.
[13] Current J, Daskin M, Schilling D. Discrete network location models. In: Drezner Z, Hamacher HW, editors. Facility location: applications and theory. Berlin: Springer-Verlag; 2002. p. 81–118.
[14] Densham PJ, Rushton G. A more efficient heuristic for solving large p-median problems. Papers in Regional Science 1992;71:307–29.
[15] Drezner T, Drezner Z, Salhi S. Solving the multiple competitive facilities location problem. European Journal of Operational Research 2002;142:138–51.
[16] Drezner Z. The planar two-center and two-median problems. Transportation Science 1984;18:351–61.
[17] Drezner Z. A note on accelerating the Weiszfeld procedure. Location Science 1996;3:275–9.
[18] Drezner Z, Klamroth K, Schöbel A, Wesolowsky GO. The Weber problem. In: Drezner Z, Hamacher HW, editors. Facility location: applications and theory. Berlin: Springer; 2002. p. 1–36.
[19] Drezner Z, Simchi-Levi D. Asymptotic behavior of the Weber location problem on the plane. Annals of Operations Research 1992;40:163–72.
[20] Eilon S, Watson-Gandy CDT, Christofides N. Distribution management. New York: Hafner; 1971.
[21] Eiselt HA, Marianov V. Foundations of location analysis: industrial series in operations research & management science, vol. 155. New York: Springer; 2011.
[22] Feldman E, Lehrer FA, Ray TL. Warehouse location under continuous economies of scale. Management Science 1966;12:670–84.

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012

[23] Glover F. Heuristics for integer programming using surrogate constraints. Decision Sciences 1977;8:156–66.
[24] Glover F. Future paths for integer programming and links to artificial intelligence. Computers & Operations Research 1986;13:533–49.
[25] Glover F, Laguna M. Tabu search. Boston, MA: Kluwer Academic Publishers; 1997.
[26] Hakimi SL. Optimum locations of switching centres and the absolute centres and medians of a graph. Operations Research 1964;12:450–9.
[27] Hakimi SL. Optimum distribution of switching centers in a communication network and some related graph theoretic problems. Operations Research 1965;13:462–75.
[28] Hansen P, Mladenović N, Taillard E. Heuristic solution of the multisource Weber problem as a p-median problem. Operations Research Letters 1998;22:55–62.
[29] Hartigan J, Wong M. Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics) 1979;28:100–8.
[30] Krishna K, Narasimha Murty M. Genetic k-means algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 1999;29:433–9.
[31] Kuehn AA, Hamburger MJ. A heuristic program for locating warehouses. Management Science 1963;9:643–66.
[32] Love RF. One dimensional facility location–allocation using dynamic programming. Management Science 1976;22:614–7.
[33] Love RF, Morris JG. A computation procedure for the exact solution of location–allocation problems with rectangular distances. Naval Research Logistics Quarterly 1975;22:441–53.
[34] Love RF, Morris JG, Wesolowsky GO. Facilities location: models & methods. New York, NY: North Holland; 1988.
[35] Maranzana EE. On the location of supply points to minimize transport costs. Operational Research Quarterly 1964;15:261–70.
[36] Megiddo N, Supowit KJ. On the complexity of some common geometric location problems. SIAM Journal on Computing 1984;13:182–96.
[37] Mladenović N, Brimberg J, Hansen P, Moreno-Perez JA. The p-median problem: a survey of metaheuristic approaches. European Journal of Operations Research 2007;179:927–39.
[38] Murtagh BA, Niwattisyawong SR. An efficient method for the multi-depot location–allocation problem. Journal of the Operational Research Society 1982;33:629–34.
[39] Okabe A, Boots B, Sugihara K, Chiu SN. Spatial tessellations: concepts and applications of Voronoi diagrams. Wiley series in probability and statistics. John Wiley; 2000.
[40] Reinelt G. TSLIB a traveling salesman library. ORSA Journal on Computing 1991;3:376–84.
[41] Salhi S, Atkinson RA. Subdrop: a modified drop heuristic for location problems. Location Science 1995;3:267–73.
[42] Scott AJ. Location-allocation systems: a review. Geographical Analysis 1970;2:95–119.
[43] Sherali HD, Shetty CM. The rectilinear distance location–allocation problem. AIIE Transactions 1977;9:136–43.
[44] Sherali HD, Tuncbilek CH. A squared Euclidean distance location–allocation problem. Naval Research Logistics 1992;39:447–69.
[45] Suzuki A, Okabe A. Using Voronoi diagrams. In: Drezner Z, editor. Facility location: a survey of applications and methods. New York: Springer; 1995. p. 103–18.
[46] Tornqvist G, Nordbeck S, Rystedt B, Gould P. Multiple location analysis. In: Lund studies in geography, Series C, general, mathematical and regional geography, No. 12. University of Lund, Sweden; 1971.
[47] Weiszfeld E. Sur le point pour lequel la somme des distances de n points donnes est minimum. Tohoku Mathematical Journal 1936;43:355–86.
[48] Weiszfeld E, Plastria F. On the point for which the sum of the distances to n given points is minimum. Annals of Operations Research 2009;167:7–41.
[49] Whitaker R. A fast algorithm for the greedy interchange for large-scale clustering and median location problems. INFOR 1983;21:95–108.

Please cite this article as: Brimberg J, Drezner Z. A new heuristic for solving the p-median problem in the plane. Computers and Operations Research (2012), http://dx.doi.org/10.1016/j.cor.2012.07.012