TOP (2022) 30:34–59
https://doi.org/10.1007/s11750-021-00599-w

ORIGINAL PAPER

Less is more: discrete starting solutions in the planar *p*-median problem

Pawel Kalczynski[1] · **Jack Brimberg**[2] · **Zvi Drezner**[1]

Received: 10 October 2020 / Accepted: 20 March 2021 / Published online: 6 April 2021

Abstract
This paper examines the performance of improvement search as a function of the quality of the starting solution in the planar (or continuous) *p*-median problem. We show that using optimal solutions of the analogue discrete p-median problem as the starting solution for heuristic improvement algorithms, as recommended in the literature, can actually lead to inferior performance. That is, good starting solutions obtained in the discrete space with a fraction of the effort can actually be better, a counter-intuitive result that illustrates in a different context the less is more principle recently advocated in the literature.

Keywords Multiple facility location · P-median · Starting solutions · Heuristics

Mathematics Subject Classification 90B85 · 90C59 · 46N10

1 Introduction

An efficient way to solve many evaluation and optimization problems is through discretization of the continuous model. For example, integration of a continuous function is evaluated by the Simpson rule or Gaussian quadrature formulas (Abramowitz and Stegun 1972). Ordinary differential equations are solved by the Runge–Kutta method (Runge 1895; Kutta 1901; Ince 1926). Many models that

✉ Zvi Drezner
zdrezner@fullerton.edu

Pawel Kalczynski
pkalczynski@fullerton.edu

Jack Brimberg
Jack.Brimberg@rmc.ca

[1] Steven G. Mihaylo College of Business and Economics, California State University-Fullerton, Fullerton, CA 92834, USA

[2] Department of Mathematics and Computer Science, The Royal Military College of Canada, Kingston, ON, Canada

are based on normal distributions are solved by discretizing the distribution (e.g. Drezner and Zerom 2016). Aboolian et al. (2007) solved a competitive location model with concave demand by converting the continuous function to a sequence of discrete lines termed the TLA (tangent line approximation) method. Brimberg et al. (2017a) solved the *p*-median problem by injecting potential locations for the facility in an iterative approach. There are exceptions to this rule. For example, integer linear programming solutions may not be close to the continuous ones. However, it is reasonable to expect that solving multiple facility location problems by discretizing the problem formulation will be an effective approach. In this paper we investigate the application of discrete *p*-median solutions as starting solutions for solving the continuous *p*-median problem. We show that using the best discrete solutions as starting solutions for continuous optimization may not be the best strategy for planar p-median problems.

The continuous *p*-median problem (Drezner et al. 2016; Drezner and Salhi 2017), also known as the multi-source Weber problem (Brimberg et al. 2000; Kuenne and Soland 1972; Hansen et al. 1998), or continuous location-allocation problem (Love et al. 1988; Brimberg et al. 2008), requires finding p sites for facilities in Euclidean space in order to minimize a weighted sum of distances from a set of demand points (fixed points, existing facilities, customers) to their closest facility. Let X_i denote the location of facility $i \in \{1, \ldots, p\}$, and A_j the known location of demand point $j \in N = \{1, \ldots, n\}$. In the typical scenario, which is assumed here, the $X_i = (x_i, y_i)$ and $A_j = (a_j, b_j)$ for all i, j are points in the plane. As well, distances are assumed to be measured by the Euclidean norm, so that

$$d(X_i, A_j) = \sqrt{(x_i - a_j)^2 + (y_i - b_j)^2}. \tag{1}$$

The weights (or demands) at the A_j's are given by $w_j > 0, j \in \{1, \ldots, n\}$. We formalize the planar *p*-median problem as follows:

$$\min_{X \subset \mathbb{R}^2} \left\{ f(X) = \sum_{j=1}^{n} w_j \min_{1 \le i \le p} \{d(X_i, A_j)\} \right\}, \tag{2}$$

where $X = \{X_1, \ldots, X_p\}$ denotes the set of location variables.

This model was originally proposed by Cooper (1963, 1964), who also observed that the objective function $f(X)$ is non-convex, and may contain several local optima. The problem was later shown to be NP-hard (Megiddo and Supowit 1984; Garey and Johnson 1979; Kariv and Hakimi 1979). For recent reviews of the discrete *p*-median problem see Daskin and Maass (2015), and for the planar *p*-median problem see Brimberg and Hodgson (2011).

The single facility 1-median problem termed the Weber problem (Weber 1909) has a long history dating back to the French mathematician Pierre De Fermat of the 1600s. Recent reviews of the Weber problem are Wesolowsky (1993); Church (2019); Drezner et al. (2002).

The connection between the discrete *p*-median problem, where potential facility locations are restricted to a given set of nodes (or demand points), and its

continuous counterpart, where the facility sites are modeled as unknown points in the plane, has been a topic of study since the introduction of the continuous problem by Cooper (1963, 1964). Hansen et al. (1998) propose a heuristic that first finds an optimal solution of the discrete problem, and then performs one step of continuous adjustment starting from this solution to obtain the final one. Very good computational results are reported, but as observed in Brimberg et al. (2000), the CPU time to find an optimal solution to the discrete p-median becomes prohibitively high for larger problem instances. In this paper we show, through extensive computational tests, a rather counter-intuitive result that good discrete starting solutions can actually be better than optimal (or best-known) ones. The implication is that heuristics that interface between discrete and continuous versions of the p-median (e.g., Brimberg et al. 2014), should not focus efforts on obtaining exact solutions in the discrete phase, but rather introduce some randomness by using multiple good discrete solutions as starting points for the continuous phase. This will save considerable execution time, and may, depending on the improving search, produce as good or better continuous solutions.

The remainder of the paper is organized as follows. In Sect. 2, a binary linear program is formulated to find the optimal solution to the discrete p-median problem. The formulation is extended to find the k best solutions to the discrete p-median problem using CPLEX recursively. A genetic algorithm is also presented to find 'good quality' discrete solutions, but still, notably inferior to the solutions to the discrete p-median problem. Section 3 presents extensive computational results followed by some discussion in Sect. 4. A summary of conclusions is presented in Sect. 5. The main conclusion is that, although good discrete starting solutions result in better continuous solution quality than random discrete solutions (i.e., poor ones), there is a point reached where further improvement of the discrete solution can be counter-productive in the continuous phase. That is, "good" can sometimes be better than "best". This also corroborates the 'less is more' philosophy adopted in some recent papers on heuristic design (e.g., Mladenović et al. 2016; Brimberg et al. 2017b).

2 Generating starting solutions

We investigate an approach suggested in the literature (e.g., Hansen et al. 1998; Brimberg et al. 2014) of finding the best selection of p demand points out of n given points, to form the starting solution for locating p facilities in the plane $\mathbb{R}^2$. Let d_{ij} be the distance between demand points i and j. We minimize the value of the discrete objective function by selecting set P containing p out of the n demand points. The standard formulation for the discrete p-median problem is:

$$\min_{P:|P|=p} \left\{ \sum_{i=1}^{n} w_i \min_{j \in P} \{d_{ij}\} \right\} \tag{3}$$

We also extended this idea by finding the second best, third best, etc., until say the $k = 100$th best, and then using each of these as starting solutions to solve the

continuous *p*-median problem. Two approaches are tested: (i) finding the *k* best solutions exactly by a binary linear program, and (ii) finding heuristically good solutions by a genetic algorithm.

2.1 A binary linear programming formulation

The following formulation for finding the best set is similar to the one in ReVelle and Swain (1970); Kalczynski and Drezner (2020); Daskin (1995); Daskin and Maass (2015). Let $x_j \in \{0,1\}$ be a binary variable. $x_j = 1$ if demand point j is selected for locating a facility, and zero otherwise. $y_{ij} = 1$ if selected demand point j is the closest facility to demand point i. The BLP (binary linear programming) optimization problem is:

$$\min \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} [w_i d_{ij}] y_{ij} \right\}$$

subject to:

$$\begin{aligned}
&\sum_{j=1}^{n} x_j = p \\
&y_{ij} \le x_j && \text{for } i,j = 1,\ldots,n \\
&\sum_{j=1}^{n} y_{ij} = 1 && \text{for } i = 1,\ldots,n \\
&x_j \in \{0,1\} && \text{for } j = 1,\ldots,n \\
&y_{ij} \in \{0,1\} && \text{for } i,j = 1,\ldots,n
\end{aligned} \tag{4}$$

For the mixed version MBLP (mixed binary linear program), $0 \le y_{ij} \le 1$ are continuous variables for $i = 1,\ldots,n$, $j = 1,\ldots,n$. In the computational experiments we applied BLP as it required less CPU time than MBLP. In formulation (4) there are $n^2 + n$ variables, and $n^2 + n + 1$ constraints not including the last three.

2.2 Finding the best *k* discrete solutions

Let $P^{(1)}$ be the optimal set. In order to get the second best solution add the constraint

$$\sum_{j \in P^{(1)}} x_j \le p - 1$$

getting the second best set $P^{(2)}$. Once the best $k \ge 1$ sets are found, the $(k+1)^{th}$ best solution is obtained by adding to the original problem (4) k constraints:

$$\sum_{j \in P^{(m)}} x_j \le p - 1 \text{ for } m = 1,\ldots,k \tag{5}$$

2.3 A genetic algorithm

We create a population of *pop* solutions, such as $pop = 100$, and once the genetic algorithm terminates, we have a sorted list of solutions by the objective function value of the discrete p-median. We can apply up to *pop* of the best population members as starting solutions. We propose a variant of a genetic algorithm (Goldberg 2006; Holland 1975), which is similar to such algorithms that were designed and successfully applied to problems of selecting p items out of n with various objective functions (Alp et al. 2003; Berman et al. 2003; Drezner et al. 2020).

2.3.1 Outline of the genetic algorithm

Every population member is a list of p demand points where facilities are located. The iterations are repeated until the best population member is not improved in a pre-specified number of consecutive iterations N. In most experiments we used $N = 2p$ or $5p$.

1. A population of *pop* (we used $pop = 100$) sets of p demand points is either randomly generated or generated by a construction algorithm (described below). Set $iter = 0$.
2. Two population members are selected by the parents selection rule detailed below.
3. An offspring is produced by the merging process detailed below.
4. If the offspring is worse than the worst population member, or is identical to an existing population member, go to Step 7.
5. If the offspring is better than the best population member, set $iter = 0$.
6. The offspring replaces the worst population member.
7. Set $iter = iter + 1$, and if $iter \leq N$, go to Step 2.
8. The final population is the desired list of starting solutions. It can be sorted by the value of the discrete objective function, which is actually the same as the continuous objective function value.

The following parents selection rule was proposed in Drezner and Marcoulides (2003). It is based on the biological concepts of inbreeding and outbreeding depression (Edmands 2007; Fenster and Galloway 2000; Drezner and Drezner 2020). A successful offspring is formed in nature when the parents are not too close genetically, such as siblings, (inbreeding depression) or too dissimilar (outbreeding depression). We assume that $2p < n$.

The Parents Selection Rule:

Define:

s: Number of potential second parents (a parameter). In Drezner and Marcoulides (2003), and later applications, $s = 2$ or 3 provided the best performance. A large value of s may lead to outbreeding depression. $s = 1$ is the "standard" parent selection rule of randomly selecting two parents. The selected individual is

typically neither the most similar nor the most different member of the population. In nature, breeding with individuals that are genetically very similar (e.g. siblings, parents) tends to produce inferior offspring, termed *inbreeding depression* (Freeman et al. 2014). Conversely, breeding with individuals that are very different, also tends to be disadvantageous. This is termed *outbreeding depression* (Edmands 2007; Fenster and Galloway 2000).

c: Similarity count between two population members. It is the number of demand points that are included in both members.

The process:

1. One population member is randomly selected as the first parent.
2. $s \geq 1$ potential second parents are randomly selected from the remaining $pop - 1$ population members.
3. For each potential second parent, the similarity count with the first parent, c, is found.
4. The potential second parent with the smallest value of c is selected with ties broken arbitrarily, so that the parents are as dissimilar as possible,.

The Merging Process:

1. The combined set of demand points in the two parents is created. c points are common to the two parents; so there are $2p - c$ points in the combined set.
2. Randomly select c points, which are not in the combined set (out of the remaining $n - (2p - c)$ points), and add them to the combined set. This is done to avoid the possibility that a "good" demand point is not present in any population member and thus will not be present in future generations. The extended set has $2p$ members.
3. A reverse descent algorithm is applied on the extended set:
   (a) Select the point in the set, not among the c common points, whose removal increases the objective function the least.
   (b) The removal of points is repeated for p iterations, so that the set is reduced to p points defining the offspring.

2.3.2 Efficient calculation of the merging process

1. The set P of $2p$ demand points for facility locations is given. The c members that are common to the two parents are moved to the beginning of the list and duplicates removed. Set the number of demand points in P to $m = 2p - c$.
2. A vector D of the distances D_i between demand point i and the closest facility is found for $i = 1, \ldots, n$ and the objective function F for the m-median problem is calculated.
3. Set the facility to be removed $j = c + 1$, and the minimum increase in the objective function ΔF to a large number.

4. The increase in the value of the objective function ΔF_j by removing facility j from P is calculated as follows:

   (a) Set $\Delta F_j = 0, i = 1$.
   (b) If $d_{ij} > D_i$ go to Step 4D
   (c) If $d_{ij} = D_i$, find the minimum distance $\hat{D}_i$ to the demand points in $P - \{j\}$ and set $\Delta F_j = \Delta F_j + w_i(\hat{D}_i - D_i)$.
   (d) Set i to $i + 1$.
   (e) If $i \leq n$ go to Step 4b.

5. If $\Delta F_j < \Delta F$ set $\Delta F = \Delta F_j$ and record the value of $\hat{j} = j$.
6. Set $j = j + 1$.
7. If $j \leq m$ go to Step 4
8. Remove Facility $\hat{j}$ from P; set $m = m - 1$. Update the vector D: not changing it when $d_{ij} > D_i$, and if $d_{ij} = D_i$ replace D_i by $\hat{D}_i$ as in Step 4c.
9. Update $F = F + \Delta F$.
10. If $m > p$ go to Step 3. Otherwise, stop with the set P and its objective function F.

If the merging process is performed in a straightforward way, i.e., calculating the value of the objective function for each potential removal of demand point j from the set P, there are $O(p^2)$ such evaluations and each requires $O(np)$ operations for a total complexity of $O(np^3)$. In the efficient calculations: Step 4b requires $O(1)$ operations, and Step 4c requires $O(p)$ operations but is performed on average only $\frac{1}{p}$ of the time. Therefore, each evaluation of the objective function is of complexity $O(n)$. The number of the objective function calculations remains the same, reducing the complexity to $O(np^2)$.

2.4 Starting solutions for the genetic algorithm

We tested two starting solution approaches for the discrete problem: a random selection of p demand points (RAND), and the construction algorithm (CONS) suggested in Brimberg and Drezner (2020); Kalczynski et al. (2020b, 2020a). The construction algorithm, which applies the GRASP approach (Feo and Resende 1995), is outlined as follows:

1. The first two demand points are randomly selected.
2. The demand point with the largest minimum distance to the already selected points is selected with probability $\frac{2}{3}$, and the one with the second-largest minimum distance is selected with probability $\frac{1}{3}$.
3. Repeat Step 2 until p demand points are selected.

2.4.1 Improving the starting solution

Once the starting solutions were determined, we applied the RATIO algorithm (Brimberg and Drezner 2020), which is a small modification of the IALT algorithm (Brimberg and Drezner 2013), on the set of starting solutions. Cooper (1963, 1964) suggested the ALT (alternating) algorithm. Since each demand point gets its services from the closest facility, each facility attracts a subset of demand points. The algorithm alternates between determining the subsets attracted by each facility, and re-locating the facilities to the optimal location for each subset, until stabilization. Once Cooper's algorithm terminates, the IALT algorithm finds for each demand point the difference between the distances to the closest facility and the second closest, and considers re-assigning the demand points with the L smallest differences (we use $L = 20$) individually from the closest facility to the second closest. Another iteration of Cooper's algorithm is then performed on the two adjusted subsets. If no improvement for all the transfers is found, the process terminates. The RATIO algorithm selects the L smallest *ratios* between the distances rather than differences. It was found in Brimberg and Drezner (2020) that this small modification significantly improves the performance of the IALT algorithm.

Since the merging process consumes most of the calculation time, and is performed about p times, the complexity of the genetic algorithm using the efficient calculations is $O(np^3)$ rather than $O(np^4)$. The complexity of the RATIO procedure (Brimberg and Drezner 2020) is the same as the IALT algorithm (Brimberg and Drezner 2013), which is $O(np^2)$ per iteration.

3 Computational experiments

The FORTRAN programs used double precision arithmetic. They were compiled by an Intel 11.1 FORTRAN Compiler with no parallel processing. They were run on a desktop with the Intel i7-6700 3.4GHz CPU processor and 16GB RAM. Only one processor was used. The BLP was run on IBM's CPLEX (CPLEX and IBM ILOG 2009) Optimization Studio 12.8 environment. We used the default CPLEX MIP solver settings. The solvers were run on a virtualized Windows environment with 16 vCPUs and 128GB of vRAM. The physical server used was a 2 CPU (8 cores each) PowerEdge R720 Intel E5-2650 CPUs with 128 GB RAM using shared storage on MD3620i via 10GB interfaces.

We first tested the approaches on the 50 randomly generated instances of problems used in Brimberg and Drezner (2020). The number of demand points is $n = 100, 200, \ldots, 1000$, and the number of facilities is $p = 5, 10, 15, 20, 25$, for a total of 50 instances. We also tested problems that were extensively investigated in the literature, and report results on two of them with $n = 654, 1060$ demand points (Reinelt 1991), for a total of 58 instances. In total, 108 instances were tested.

The starting solutions were found in two ways: (i) the best 100 discrete solutions found optimally by the binary linear program; and (ii) the final population members of the genetic algorithm.

3.1 Testing randomly generated instances

In Table 1 we report the continuous solution found by starting at the optimal (best) discrete solution. The best discrete solution was obtained by solving (4) using CPLEX. Some of the largest problem sets required several days of computing time. Applying RATIO on the best discrete solution required a negligible fraction of one second by a FORTRAN program. The results are disappointing. The best known (BK) continuous solution was found for only 12 of the 50 instances. The average percentage above the best known solution was 0.048%.

For comparison, we report in Table 2 the analogous results obtained from repeating the genetic algorithm 10 and 100 times using $N = 2p$, $s = 1, 2 \ldots, 5$ second parents, and the best 10 members of each final population. When 100 runs are

Table 1 Optimal discrete solution as a starting solution for the test problems

n	p	(1)	(2)	(3)	n	p	(1)	(2)	(3)
100	5	164.6011	167.3227	164.6179	600	5	1030.9282	1032.1770	1031.0041
100	10	100.7650	101.7818	101.0657	600	10	694.2726	697.3327	694.2980
100	15	74.4746	75.5618	**74.4746**	600	15	547.8102	549.8051	547.9021
100	20	59.4779	60.1859	**59.4779**	600	20	460.6433	463.3245	461.2379
100	25	49.1846	49.7157	49.2830	600	25	408.3926	410.1870	408.6577
200	5	329.0968	331.1405	**329.0968**	700	5	1198.9113	1200.7430	**1198.9113**
200	10	213.1025	214.2705	213.1246	700	10	807.4504	810.0341	808.1567
200	15	167.1654	168.5897	167.3351	700	15	647.6007	649.6282	**647.6007**
200	20	140.0728	141.4799	140.3091	700	20	548.0676	550.6144	548.2130
200	25	120.5562	122.5305	120.5626	700	25	482.5661	485.3041	482.8113
300	5	505.9990	508.2661	**505.9990**	800	5	1372.8710	1374.7470	1373.9489
300	10	331.5499	333.1892	**331.5499**	800	10	928.7004	930.4604	929.1542
300	15	259.6754	262.7352	**259.6754**	800	15	743.1017	745.3164	743.1632
300	20	216.8050	219.2277	216.8138	800	20	633.9782	636.4713	634.2868
300	25	191.5259	193.5395	191.9743	800	25	557.1867	560.3380	557.6293
400	5	685.1978	686.3057	685.2155	900	5	1545.5993	1549.4860	**1545.5993**
400	10	458.8549	460.8376	458.8954	900	10	1053.7279	1055.6010	1054.3150
400	15	362.7120	364.6220	362.9923	900	15	844.0657	846.3643	**844.0657**
400	20	304.1061	306.2300	304.1783	900	20	718.9711	721.6564	719.6132
400	25	266.3945	269.1881	266.5848	900	25	634.8785	637.9545	634.9551
500	5	856.1153	858.1612	**856.1153**	1000	5	1731.6308	1735.3630	1731.9183
500	10	575.6737	577.4147	576.0708	1000	10	1177.9664	1180.2960	1178.7499
500	15	449.8948	453.1653	450.1084	1000	15	942.4672	944.4968	**942.4672**
500	20	382.6915	384.4923	382.8467	1000	20	798.5461	802.0976	798.5688
500	25	337.3002	339.1829	337.6522	1000	25	705.8626	708.3867	705.9880

(1) Best known continuous objective

(2) Best discrete objective

(3) Continuous objective from best discrete solution. Best known marked in boldface

Table 2 Variants of the genetic approach

	Number of Second Parents				
	1	2	3	4	5
Random Starting Solutions, 10 runs					
Average of best found solution above best known	0.004%	0.002%	0.004%	0.003%	0.005%
Number of best known solutions found	38	41	44	42	40
Total time (seconds) for all runs of the smallest problem	0.02	0.02	0.02	0.02	0.02
Total time (seconds) for all runs of the largest problem	11.1	14.0	12.2	10.9	11.8
Average total time (seconds) for all runs	1.7	1.8	1.7	1.7	1.7
Construction Starting Solutions, 10 runs					
Average of best found solution above best known	0.005%	0.004%	0.004%	0.005%	0.006%
Number of best known solutions found	42	42	41	43	37
Total time (seconds) for all runs of the smallest problem	0.03	0.03	0.02	0.02	0.02
Total time (seconds) for all runs of the largest problem	12.1	15.5	11.5	12.3	12.1
Average total time (seconds) for all runs	2.0	2.1	2.0	2.0	2.0
Random Starting Solutions, 100 runs					
Average of best found solution above best known	0%	0%	0%	0.001%	0%
Number of best known solutions found	50	50	50	49	50
Total time (seconds) for all runs of the smallest problem	0.19	0.19	0.17	0.20	0.19
Total time (seconds) for all runs of the largest problem	116.6	118.6	118.9	122.4	114.1
Average total time (seconds) for all runs	17.5	17.3	17.4	17.5	17.4
Construction Starting Solutions, 100 runs					
Average of best found solution above best known	0%	0.000%	0%	0%	0%
Number of best known solutions found	50	49	50	50	50
Total time (seconds) for all runs of the smallest problem	0.20	0.20	0.22	0.19	0.20
Total time (seconds) for all runs of the largest problem	130.8	136.4	123.9	130.6	128.9
Average total time (seconds) for all runs	19.8	20.2	19.7	19.8	19.9

performed, the best known continuous solution was found for all 50 instances by any number of parents, except in two cases where it was found in 49 instances. All variants performed about equally well. The best performing variant (with 100 replications) used random starting solutions, and $s = 2$ second parents.

As noted above, the best known continuous solution was not obtained in many cases using the optimal discrete solution as the starting solution for RATIO. We further investigated this phenomenon on six instances of $n = 100, \ldots, 600$, and $p = 5$. The 1000 best discrete solutions for these instances were found by total enumeration with a FORTRAN program. The number of feasible combinations ranges from about 75 million for $n = 100$ to 637 billion for $n = 600$. The value of the objective function for each combination was calculated, and the sorted vector of the 1000 best objective function values created. Run times for these instances ranged from about 3 seconds for $n = 100$ to about 61 hours for $n = 600$.

In Table 3 we report the 10 highest ranked (best) discrete solutions, and the 1000^{th} one. We examined all 1000 of them, but it is not reasonable to report on all

Table 3 One thousand best discrete objectives for $p = 5$

Rank of Solution	Number of Demand points (n)					
	100	200	300	400	500	600
1	167.3227	331.1405	508.2661	686.3057	858.1612	1032.1770
2	167.3522	331.1546	508.3602	686.3366	858.1716	1032.2529
3	167.3604	331.1551	508.3676	686.4217	858.2183	1032.2832
4	167.3631	331.3353	508.4049	686.5040	858.2716	1032.3591
5	167.3899	331.4797	508.5313	686.7107	858.3287	1032.4699
6	167.3927	331.5968	508.6018	686.8402	858.3333	1032.5024
7	167.4015	331.5974	508.6510	686.8999	858.3893	1032.5645
8	167.4310	331.6160	508.6540	686.9308	858.3997	1032.5902
9	167.4838	331.6335	508.6584	686.9516	858.4437	1032.6228
10	167.4861	331.6381	508.6624	686.9825	858.4997	1032.6404
1000	169.9705	335.0922	512.0414	690.7987	862.8357	1036.7017
†	7.53E+07	2.54E+09	1.96E+10	8.32E+10	2.55E+11	6.37E+11
‡	0.05	3.57	41.35	263.00	1134.18	3661.02
Parents	Best found genetic discrete solution					
1	167.3522	336.1606	516.2656	698.9384	874.3234	1047.9237
2	167.5479	333.7619	520.6779	702.1613	865.5019	1047.5878
3	168.9990	333.6394	518.1258	700.1646	872.1444	1051.3648
4	168.5249	332.8579	517.4716	696.1967	874.7907	1047.7047
5	168.0556	332.4777	514.8653	698.8044	875.1926	1051.4130

† Number of feasible solutions

‡ Time in minutes for scanning all feasible solutions and getting the best 1000 ones

of them. The best found genetic discrete solution was ranked several times even worse than the 1000$^{\text{th}}$ one. In Table 4 the ranks of the best found genetic solution for $n = 100, 200$ are reported. For example, the best genetic solution using $s = 3$ second parents for $n = 100$ was the 266$^{\text{th}}$ best discrete solution. For $n \geq 300$ all solutions were ranked higher than the 1000$^{\text{th}}$ one. It is interesting and perhaps surprising that even though the values of the discrete objective function of the genetic algorithm solutions were rather inferior, they resulted in excellent continuous solutions (see Table 2).

Table 4 Ranks of best genetic discrete solutions, $p = 5$

Number Parents	$n = 100$		$n = 200$	
	Objective	Rank	Objective	Rank
1	167.3522	2	336.1606	>1000
2	167.5479	12	333.7619	261
3	168.9990	266	333.6394	220
4	168.5249	112	332.8579	68
5	168.0556	49	332.4777	37

In Table 5 we report the distribution of the 1000 continuous objectives obtained by RATIO starting at the best 1000 discrete starting solutions for $n \leq 600$, and $p = 5$. For $n = 200, 300, 500$, the best known continuous solution was obtained from the best discrete solution, as was also reported in Table 1. For $n = 500$ all 1000 best discrete solutions yielded the best continuous solution. A different behavior is observed for $n = 100, 400, 600$. For $n = 100$, the best continuous solution was obtained for the 15$^{\text{th}}$ discrete solution, meaning that the best 14 ones did not yield the best continuous solution. For $n = 400$ the 3$^{\text{rd}}$, and for $n = 600$ the 5$^{\text{th}}$, found the best known continuous solution. The solution obtained by the best discrete solution

Table 5 Continuous results of the 1000 best discrete starting solutions, $p = 5$

Cont.	$n = 100$			$n = 200$			$n = 300$		
Rank	Objective	†	‡	Objective	†	‡	Objective	†	‡
1	164.6011	15	129	329.0968	1	306	505.9990	1	191
2	164.6028	10	138	329.1135	2	694	506.0105	2	527
3	164.6179	1	275				506.5548	14	210
4	164.6612	376	2				506.5651	20	72
5	164.8097	164	14						
6	165.2178	18	71						
7	165.3513	129	137						
8	165.3514	17	206						
9	167.5991	200	23						
	$n = 400$			$n = 500$			$n = 600$		
1	685.1978	3	825	856.1153	1	1000	1030.9282	5	220
2	685.2155	1	175				1030.9495	21	35
3							1030.9688	3	270
4							1031.0041	1	194
5							1032.6397	203	8
6							1032.6422	230	17
7							1032.6495	185	5
8							1032.6514	520	2
9							1032.8203	241	9
10							1032.8781	62	16
11							1033.2775	57	188
12							1033.3372	202	5
13							1033.3716	116	8
14							1033.5151	431	6
15							1033.5154	400	1
16							1033.5408	344	5
17							1033.5637	644	11

† First rank of the discrete starting solution

‡ Number of times the continuous solution obtained

is 0.010% above the best known for $n = 100$, and 0.003%, 0.007% for $n = 400, 600$. For $n = 100$, ten different solutions were obtained, while for $n = 600$ there were seventeen different solutions.

In Table 6 we report on the continuous solutions found using the top 10 discrete solutions obtained by CPLEX with (4) and additional constraints (5). The improvement in the results is disappointing, compared with the results starting at the best discrete solution reported in Table 1. The best known continuous solution was obtained at least once in 21 of the 50 instances, an increase of only 9 instances. The average best result was 0.035% above the best known solution, down from 0.048%. There are also some unusual results. In eleven instances the same continuous solution was

Table 6 Best ten discrete solutions as starting solutions

n	p	(1)	(2)	(3)	(4)	(5) (%)	n	p	(1)	(2)	(3)	(4)	(5) (%)
100	5	164.6011	164.6028	1	10	0.001	600	5	1030.9282	1030.9282	4	3	0
100	10	100.7650	101.0657	8	1	0.298	600	10	694.2726	694.2726	4	2	0
100	15	74.4746	74.4746	6	1	0	600	15	547.8102	547.9021	5	1	0.017
100	20	59.4779	59.4779	10	1	0	600	20	460.6433	461.1782	1	2	0.116
100	25	49.1846	49.2830	10	1	0.200	600	25	408.3926	408.6577	9	1	0.065
200	5	329.0968	329.0968	7	1	0	700	5	1198.9113	1198.9113	10	1	0
200	10	213.1025	213.1025	4	4	0	700	10	807.4504	807.4504	3	6	0
200	15	167.1654	167.3351	5	1	0.101	700	15	647.6007	647.6007	10	1	0
200	20	140.0728	140.2270	6	3	0.110	700	20	548.0676	548.0676	3	6	0
200	25	120.5562	120.5626	10	1	0.005	700	25	482.5661	482.6098	4	2	0.009
300	5	505.9990	505.9990	5	1	0	800	5	1372.8710	1373.9489	10	1	0.079
300	10	331.5499	331.5499	10	1	0	800	10	928.7004	929.1526	3	2	0.049
300	15	259.6754	259.6754	10	1	0	800	15	743.1017	743.1632	2	1	0.008
300	20	216.8050	216.8138	10	1	0.004	800	20	633.9782	634.0161	1	5	0.006
300	25	191.5259	191.9743	10	1	0.234	800	25	557.1867	557.3658	2	8	0.032
400	5	685.1978	685.1978	4	3	0	900	5	1545.5993	1545.5993	5	1	0
400	10	458.8549	458.8954	7	1	0.009	900	10	1053.7279	1053.7279	3	3	0
400	15	362.7120	362.9760	1	10	0.073	900	15	844.0657	844.0657	9	1	0
400	20	304.1061	304.1783	10	1	0.024	900	20	718.9711	719.0012	1	5	0.004
400	25	266.3945	266.5848	10	1	0.071	900	25	634.8785	634.8785	5	4	0
500	5	856.1153	856.1153	10	1	0	1000	5	1731.6308	1731.9183	10	1	0.017
500	10	575.6737	575.6832	2	5	0.002	1000	10	1177.9664	1178.5329	1	5	0.048
500	15	449.8948	449.9610	3	2	0.015	1000	15	942.4672	942.4672	9	1	0
500	20	382.6915	382.8467	10	1	0.041	1000	20	798.5461	798.5461	1	2	0
500	25	337.3002	337.6522	10	1	0.104	1000	25	705.8626	705.9880	10	1	0.018

(1) Best known continuous objective

(2) Best continuous found starting from the best ten discrete objectives

(3) The number of times out of 10 that the (2) result obtained

(4) The smallest discrete rank yielding the (2) result

(5) Percent of (2) above (1)

found for all ten discrete starting solutions, but it was not the best known solution. *In conclusion, for this set of test problems, the best discrete solutions are not recommended as starting solutions.*

3.2 Testing the $n = 654$ and 1060 Instances

To further investigate the conclusion that optimal, or close to optimal, discrete starting solutions do not perform well for the planar *p*-median problem, we tested the procedures on the $n = 50, 287, 654, 1060$ instances, with various values of p, which were tested in many papers. The $n = 50, 287$ (Eilon et al. 1971; Bongartz et al. 1994) instances, which were solved optimally by Krau (1997), are quite easy. Therefore, we report the results for the $n = 654, 1060$ (Reinelt 1991) instances. The best known solutions for the $n = 654, 1060$ problems were reported in Drezner and Drezner (2020). In Table 7 we compare the best known continuous objective to the optimal discrete objective for the $n = 654,\ 1060$ instances.

The distribution of the demand points for the $n = 654, 1060$ instances is definitely not uniformly random (see Figs. 1,2) as in the instances examined in Sect. 3.1. In

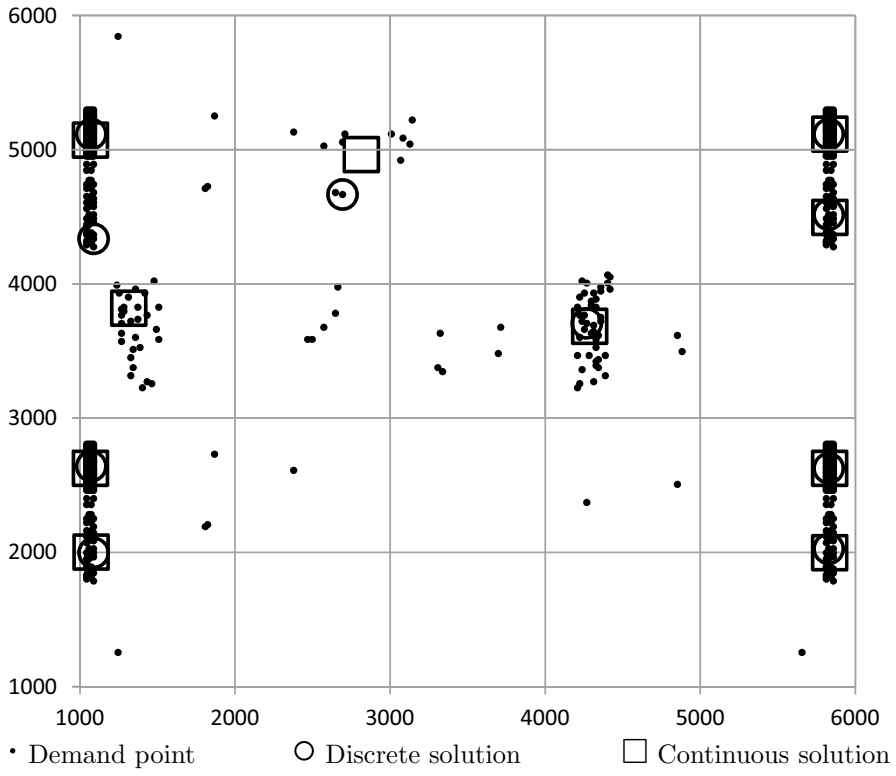

Fig. 1 The $n = 654,\ p = 10$ Instance

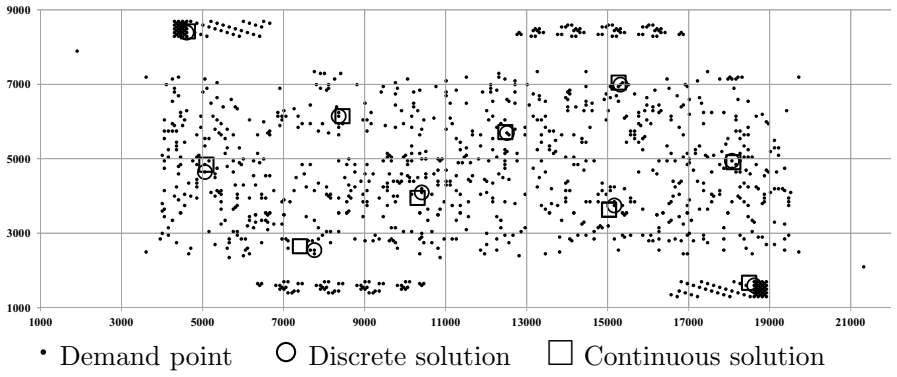

Fig. 2 The $n = 1060$, $p = 10$ Instance

Tables 8,9 we report the best continuous result obtained by applying RATIO on the best 100 discrete solutions for the $n = 654$, 1060 instances, compared with the results of applying RATIO on the genetic algorithm final population members. In Fig. 1, the optimal discrete solution and best known continuous solution for $n = 654$, $p = 10$ facilities are depicted. The best discrete solution is quite far from the continuous one. In fact, all 100 best discrete solutions resulted in the same inferior continuous solution by applying RATIO, as reported in Table 8. In Fig. 2, the best known continuous and optimal discrete solutions for $n = 1060$, $p = 10$ are depicted. The solution points are closer to each other for $n = 1060$.

3.2.1 Analysis and discussion of the $n = 654$, $p = 10$ instance

The optimal discrete objective of 115788.7512 is 0.39% above the best known continuous objective (BK). The 100^{th} best discrete solution is 0.44% above BK. All these 100 starting solutions resulted in the same continuous objective of 115384.6427, which is 0.04% above BK.

When using the genetic results as starting solutions with $N = 2p$, out of 25,000 starting solutions (5,000 starting solutions for each of the five second parents), 2,318 resulted in BK. The distribution of the discrete values of these 2,318 starting solutions is: the best one is 117595.2606 which is 1.96% above BK, the 100^{th} best of these is 3.98% above BK, and the last one on the list of 2318 is 152.55% above BK!

When N in the genetic algorithm was increased from $2p$ to $5p$, the results were better. There were 6,327 BK's, with the best discrete objective among these results of 116042.5786, which is 0.61% above BK, number 100 is 0.90% above BK, and the last one, number 6,327, is 95.45% above BK. *In conclusion, the best continuous solutions can often be found using moderately good discrete starting solutions, and in some cases even bad ones. If the discrete starting solutions are optimal or near optimal, the continuous results may be inferior.*

Table 7 Comparing discrete to continuous objectives

p	Best Known	Opt. Discrete	(1) (%)	(2)	*p*	Best Known	Opt. Discrete	(1) (%)	(2)
$n = 654$									
5	209068.7935	209155.2963	0.04	2.45	35	39257.2685	39861.9684	1.54	6.13
6	180488.2126	180613.4840	0.07	2.31	40	35704.4076	36228.2778	1.47	7.23
7	163704.1681	163880.0282	0.11	2.45	45	32306.9721	32779.0578	1.46	5.96
8	147050.7904	147275.6162	0.15	2.45	50	29338.0106	29774.1401	1.49	4.38
9	130936.1241	131182.2763	0.19	2.61	55	26699.1208	27200.0921	1.88	2.62
10	115339.0328	115788.7512	0.39	2.77	60	24504.3952	24984.0306	1.96	2.30
11	100133.2007	100708.7261	0.57	2.20	65	22733.2923	23129.2031	1.74	2.21
12	94152.0549	94634.4786	0.51	2.18	70	21465.4391	21851.8388	1.80	2.15
13	89454.7613	89930.1438	0.53	2.25	75	20269.9644	20740.4013	2.32	3.11
14	84807.6690	85255.1202	0.53	2.45	80	19193.8610	19748.1558	2.89	2.17
15	80177.0422	80595.4106	0.52	2.49	85	18313.8703	18837.1674	2.86	2.19
20	63389.0238	63894.6649	0.80	4.13	90	17514.4227	18008.4259	2.82	2.84
25	52209.5106	52875.7746	1.28	4.37	95	16770.1973	17259.7985	2.92	2.60
30	44705.1920	45307.1196	1.35	7.71	100	16083.5345	16544.1315	2.86	2.40
$n = 1060$									
5	1851877.3	1854329.7	0.13	21.42	80	325971.3	329073.4	0.95	4.93
10	1249564.8	1252141.9	0.21	14.10	85	313446.6	316449.5	0.96	9.00
15	980131.7	982399.3	0.23	10.24	90	302479.1	304974.3	0.82	24.96
20	828685.7	831419.3	0.33	9.12	95	292282.6	294660.2	0.81	19.40
25	721988.2	725005.9	0.42	9.34	100	282536.5	284814.9	0.81	22.59
30	638212.3	641851.1	0.57	6.42	105	273463.3	275576.3	0.77	4.80
35	577496.7	581365.0	0.67	5.66	110	264959.6	267025.8	0.78	5.13
40	529660.1	532431.5	0.52	6.00	115	256735.7	258803.0	0.81	5.15
45	489483.8	492441.2	0.60	6.39	120	249050.5	251111.7	0.83	5.19
50	453109.6	455587.7	0.55	6.57	125	241880.4	243873.4	0.82	5.01
55	422638.7	425166.0	0.60	6.37	130	235203.4	237192.7	0.85	4.81
60	397674.5	399964.2	0.58	5.75	135	228999.2	230919.6	0.84	4.81
65	376630.3	379072.1	0.65	5.23	140	223062.0	225017.0	0.88	4.67
70	357335.1	360126.8	0.78	5.10	145	217462.8	219397.9	0.89	4.90
75	340123.5	343260.5	0.92	5.50	150	212230.5	214201.6	0.93	4.85

(1) Percent of optimal discrete objective above best known continuous solution

(2) Run time in minutes for finding optimal discrete solution

In Table 10, various statistical results and run times of various approaches are depicted for the $n = 654, 1060$ instances. In the genetic variants, all 100 final population members were used as starting solutions.

Referring to Table 10, the following observations may be made:

Table 8 Comparing results from different starting solutions for the $n = 654$ Instances

p	Best Known	100 Best Discrete Solutions					Genetic Algorithm†		
		Best Found	(1)	(2)	(3)	(4)	Best Found	(3)	(4)
5	209068.7935	209068.7935	100	1	0%	281.50	209068.7935	0%	0.06
6	180488.2126	180488.2126	100	1	0%	269.78	180488.2126	0%	0.05
7	163704.1681	163704.1681	100	1	0%	282.15	163704.1681	0%	0.06
8	147050.7904	147050.7905	100	1	0%	296.20	147050.7904	0%	0.07
9	130936.1241	130936.1241	100	1	0%	310.54	130936.1241	0%	0.07
10	115339.0328	115384.6427	100	1	0.04%	335.31	115339.0328	0%	0.09
11	100133.2007	100133.2007	100	1	0%	263.07	100133.2007	0%	0.10
12	94152.0549	94152.0550	100	1	0%	264.89	94152.0549	0%	0.10
13	89454.7613	89454.7613	95	1	0%	274.82	89454.7613	0%	0.12
14	84807.6690	84807.6690	79	1	0%	288.40	84807.6690	0%	0.15
15	80177.0422	80177.0422	100	1	0%	298.34	80177.0422	0%	0.18
20	63389.0238	63389.8074	23	5	0.00%	481.45	63389.0238	0%	0.43
25	52209.5106	52209.5106	9	5	0%	657.37	52209.5106	0%	0.69
30	44705.1920	44705.1921	6	42	0%	877.10	44705.1920	0%	1.05
35	39257.2685	39257.2685	2	54	0%	851.93	39257.2685	0%	1.51
40	35704.4076	35773.2693	12	18	0.19%	886.95	35704.4076	0%	2.32
45	32306.9721	32306.9721	45	4	0%	496.45	32306.9721	0%	3.22
50	29338.0106	29338.0106	70	1	0%	715.04	29338.0106	0%	4.08
55	26699.1208	26703.1845	7	2	0.02%	300.42	26699.1208	0%	4.81
60	24504.3952	24505.4082	100	1	0.00%	307.89	24505.4082	0.00%	6.52
65	22733.2923	22734.3053	100	1	0.00%	287.73	22733.2923	0%	7.55
70	21465.4391	21473.9547	100	1	0.04%	302.85	21470.8346	0.03%	9.89
75	20269.9644	20341.2528	100	1	0.35%	393.27	20270.4011	0.00%	11.79
80	19193.8610	19217.9862	51	1	0.13%	316.96	19261.8774	0.35%	13.92
85	18313.8703	18346.0036	100	1	0.18%	314.21	18413.3586	0.54%	15.61
90	17514.4227	17573.3116	100	1	0.34%	404.50	17617.0847	0.59%	19.17
95	16770.1973	16850.0891	100	1	0.48%	352.08	16823.1395	0.32%	19.72
100	16083.5345	16084.3842	36	7	0.01%	336.06	16083.5345	0%	24.09

† Genetic algorithm applying: CONS, $N = 5p$, $s = 2$ parents, 50 populations of 100 members

(1) The number of times out of 100 that the best found result was obtained

(2) The smallest discrete solution rank yielding the best found result

(3) Percent of best found result above best known

(4) Total run time in minutes for all runs

1. When RATIO is applied directly to random discrete solutions, the best obtained continuous objective values for each p are bad. The average for the deviation of the best solution for each p is 12.33% for $n = 654$, and 3.15% for $n = 1060$. Note that these averages are based on best found results from 5000 starting solutions for each instance. The results improve somewhat for CONS starting solutions (9.19% for $n = 654$, and 1.93% for $n = 1060$). We conclude that the application

Table 9 Comparing results from different starting solutions for the $n = 1060$ instances

p	Best Known	100 Best Discrete Solutions					Genetic Algorithm†		
		Best Found	(1)	(2)	(3)	(4)	Best Found	(3)	(4)
5	1851877.3	1851877.3	34	4	0%	3179.77	1851877.3	0%	0.18
10	1249564.8	1249590.4	3	31	0.00%	1666.11	1249564.8	0%	0.27
15	980131.7	980319.0	1	42	0.02%	1184.81	980131.7	0%	0.52
20	828685.7	828775.0	1	95	0.01%	1101.15	828685.7	0%	0.92
25	721988.2	722029.6	2	26	0.01%	1093.31	721995.3	0.00%	1.57
30	638212.3	638236.3	26	9	0.00%	816.10	638212.3	0%	2.51
35	577496.7	577496.7	6	64	0%	689.53	577496.7	0%	3.69
40	529660.1	529970.4	8	53	0.06%	732.35	529660.1	0%	4.75
45	489483.8	489554.0	80	1	0.01%	744.24	489512.6	0.01%	6.12
50	453109.6	453297.3	68	1	0.04%	785.63	453109.6	0%	8.58
55	422638.7	422860.7	6	14	0.05%	800.72	422638.7	0%	11.19
60	397674.5	397674.9	3	68	0.00%	742.56	397674.5	0%	13.22
65	376630.3	377041.9	60	1	0.11%	703.45	376630.3	0%	16.61
70	357335.1	357335.1	3	56	0%	700.94	357335.1	0%	17.72
75	340123.5	340333.4	80	1	0.06%	730.60	340175.3	0.02%	23.43
80	325971.3	326077.3	32	1	0.03%	694.44	325998.4	0.01%	25.35
85	313446.6	313790.6	53	1	0.11%	2256.37	313512.6	0.02%	27.56
90	302479.1	302733.3	16	5	0.08%	2897.88	302613.8	0.04%	33.67
95	292282.6	292590.8	1	98	0.11%	2879.34	292625.9	0.12%	39.91
100	282536.5	282738.6	7	35	0.07%	2878.77	282825.5	0.10%	47.51
105	273463.3	273738.9	2	49	0.10%	759.91	273821.7	0.13%	53.79
110	264959.6	265361.7	1	81	0.15%	786.38	265445.8	0.18%	52.95
115	256735.7	256908.9	4	10	0.07%	787.34	257195.2	0.18%	65.48
120	249050.5	249068.8	1	73	0.01%	781.67	249155.2	0.04%	77.35
125	241880.4	241936.3	22	5	0.02%	738.50	241955.5	0.03%	85.32
130	235203.4	235477.4	1	96	0.12%	768.88	235469.2	0.11%	98.30
135	228999.2	229026.6	1	75	0.01%	766.40	229138.0	0.06%	105.75
140	223062.0	223238.8	1	28	0.08%	776.05	223207.2	0.07%	112.18
145	217462.8	217572.3	6	24	0.05%	773.84	217955.5	0.23%	120.68
150	212230.5	212401.0	1	62	0.08%	779.30	212510.5	0.13%	136.78

† Genetic algorithm applying: CONS, $N = 5p$, $s = 2$ parents, 50 populations of 100 members

(1) The number of times out of 100 that the best found result obtained

(2) The smallest discrete rank yielding the best found result

(3) Percent of best found result above best known

(4) Total run times in minutes for all runs

Table 10 Average for all p's of best results by applying RATIO on discrete starting solutions

Parents	$n = 654$		$n = 1060$		Average
	RAND	CONS	RAND	CONS	
Genetic using $N = 2p$, 50 populations of 100					
1	0.079%	0.063%	0.060%	0.054%	0.064%
2	0.083%	0.073%	0.065%	0.060%	0.070%
3	0.089%	0.068%	0.058%	0.055%	0.067%
4	0.079%	0.071%	0.059%	0.059%	0.067%
5	0.078%	0.066%	0.062%	0.062%	0.067%
Average	0.082%	0.068%	0.061%	0.058%	0.067%
Time‡	2.70	2.70	19.44	19.95	
Genetic using $N = 5p$, 50 populations of 100					
1	0.077%	0.066%	0.062%	0.052%	0.064%
2	0.070%	0.065%	0.052%	0.049%	0.059%
3	0.084%	0.070%	0.058%	0.053%	0.066%
4	0.081%	0.071%	0.057%	0.056%	0.066%
5	0.076%	0.071%	0.053%	0.051%	0.063%
Average	0.078%	0.069%	0.056%	0.052%	0.064%
Time‡	5.44	5.33	39.12	39.34	
No genetic on 5000 starting solutions					
Average	12.33%	9.19%	3.15%	1.93%	6.65%
Time‡	1.17	0.51	5.70	4.71	
Applying RATIO on the 100 Best Discrete solutions					
Average	0.063%		0.049%		0.056%
Time‡	408.83*		1166.54*		

‡Average time per instance (one parent selection) in minutes

* CPLEX time (minutes) for 100 best. RATIO time is negligible

of the genetic algorithm to obtain "good" discrete starting solutions is well worth the extra effort, as seen by the substantial improvement in the averages of best results.

2. The best overall solution quality reported in Table 10 is obtained by using the 100 best discrete solutions as starting solutions. (Further analysis reported in Table 11 shows a better percentage above the best known solution of 0.052% for $n = 654$ and 0.043% for $n = 1060$ by a very shallow genetic algorithm). However, the reduction in percent deviation over the genetic results is of the order of 0.005%, which is quite negligible in practical terms. Furthermore, as seen in Tables 8 and 9, the improvement applies only to the larger instances (larger values of p), while the genetic starting solutions gave better results for the smaller ones.

Table 11 Comparing results from different depth values (Five Random Seeds)

Depth	Starting	$n = 654$				$n = 1060$			
	Solutions	(1)	(2)	(3)	(4)	(1)	(2)	(3)	(4)
1	24000	0.048%	8.26	2.45	5.81	0.046%	61.49	22.61	38.88
2	12000	0.061%	6.37	1.22	5.14	0.051%	47.85	11.30	36.55
3	8000	0.065%	5.67	0.82	4.85	0.052%	42.16	7.54	34.62
4	6000	0.069%	5.37	0.61	4.75	0.052%	39.68	5.65	34.03
5	4800	0.067%	5.12	0.49	4.63	0.050%	38.14	4.52	33.62
6	4000	0.071%	4.98	0.41	4.57	0.053%	36.64	3.77	32.87
8	3000	0.073%	4.72	0.31	4.42	0.055%	35.10	2.83	32.27
10	2400	0.075%	4.55	0.24	4.30	0.055%	33.66	2.26	31.40
12	2000	0.076%	4.46	0.20	4.25	0.053%	32.54	1.88	30.66
15	1600	0.077%	4.41	0.16	4.25	0.054%	33.13	1.51	31.63
16	1500	0.078%	4.35	0.15	4.20	0.053%	32.77	1.41	31.36
20	1200	0.081%	4.35	0.12	4.22	0.058%	32.13	1.13	31.00
24	1000	0.085%	4.23	0.10	4.13	0.056%	31.15	0.94	30.21
30	800	0.089%	4.12	0.08	4.03	0.059%	30.57	0.75	29.82
40	600	0.082%	4.07	0.06	4.01	0.062%	30.10	0.57	29.54
48	500	0.092%	3.93	0.05	3.88	0.065%	30.45	0.47	29.98
60	400	0.089%	4.01	0.04	3.97	0.065%	29.23	0.38	28.85
80	300	0.098%	3.86	0.03	3.83	0.071%	28.05	0.28	27.77
120	200	0.107%	3.82	0.02	3.80	0.076%	28.59	0.19	28.40
240	100	0.126%	3.30	0.01	3.29	0.093%	27.66	0.09	27.57

(1) Percent of the average of best found result for each random seed, above best known

(2) Average time (minutes per seed) for all runs

(3) Estimated average time (minutes per seed) for CONS and RATIO

(4) Average time (minutes per seed) for the genetic algorithms

3.2.2 Analyzing the depth of the genetic algorithm

Define the depth of the genetic algorithm as: $depth = \frac{N}{p}$. The deeper the genetic algorithm, the greater are the number of iterations executed, and the longer are the run times (approximately proportional to the depth). Therefore, the final population of the genetic algorithm improves with depth. If we wish to have approximately the same genetic algorithms run time, the number of repetitions of the genetic algorithm should be inversely proportional to the depth.

We selected the value of the depth multiplied by the number of populations equal 240, so there are 20 integer values of the depth and number of populations, see Table 11. Note that for each tested population, 100 starting solutions are created for RATIO. Each instance was run five times applying five different random seeds. The *average* of the five best found solutions by each of the five random seeds is reported. Graphs showing the percentage above the best known solution as a function of log(*depth*), obtained from Table 11, are depicted in Fig. 3.

Table 12 Statistical analysis of the average best results

Independent Variable	$n = 654$		$n = 1060$	
	R	p-value	R	p-value
depth	0.8919	1.3×10^{-7}	0.9728	7.0×10^{-13}
log(depth)	0.9624	1.2×10^{-11}	0.8765	4.0×10^{-7}
$\log(\text{depth})^{\lambda}$†	0.9708	1.3×10^{-12}	0.9888	2.4×10^{-16}

†$\lambda = 1.5$ for $n = 654$; $\lambda = 3.4$ for $n = 1060$

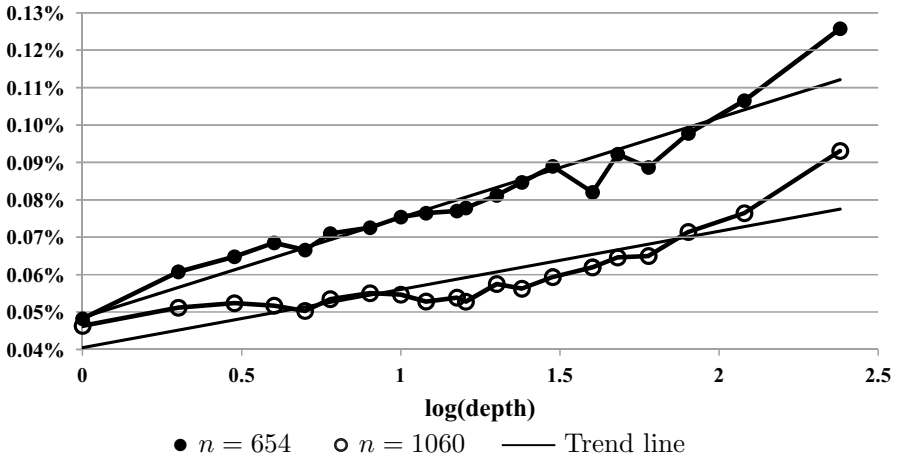

Fig. 3 Percent above the best known solution (average of 5 seeds)

The total run time of the procedures consists of generating the starting solutions, running the genetic algorithms, and running RATIO. Generating starting solutions and running RATIO are performed the same number of times for each p (second column in Table 11). We estimated the run time of generating 5000 starting solutions and 5000 runs of RATIO as 0.51 min for $n = 654$, and 4.71 min for $n = 1060$ (Table 10). For example, (referring to Table 11) for depth=1 and $n = 1060$, the procedure required an average of 61.49 min. Generating 24000 starting solutions and running RATIO 24000 times required $24000 \times \frac{4.71}{5000} = 22.61$ min. Therefore, the genetic algorithms (that were repeated 240 times) required 61.49-22.61=38.88 min, which is about 10 sec per each genetic algorithm. In contrast, one run of the genetic algorithm using depth=240 required 27.57 min.

We calculated simple regression lines, with the dependent variable of the average percentage above the best known solution for $n = 654, 1060$, on the 20 points for each n depicted in Fig. 3. As the independent variable we tested: (i) depth, (ii) log(depth), and (iii) $\left[\log(depth)\right]^{\lambda}$, for the λ that best fits the data (maximum R value). We found, by the Solver in Excel, that the best value for λ is 1.5 for

$n = 640$, and 3.4 for $n = 1060$. The results are depicted in Table 12. All regression lines are statistically significant with very small p-values. This means that the quality deteriorates significantly as the genetic algorithm provides better objective values of the starting solution, when the number of runs are controlled by the run time of the genetic algorithm. More populations with lower quality discrete starting solutions yield better continuous objectives.

4 Discussion

When all the locations of the facilities in the optimal continuous solution coincide with demand points, the optimal discrete solution is the same as the continuous one. How likely is it that a facility is located at a demand point? In the optimal solution (continuous or discrete), the set of demand points is partitioned by a Voronoi diagram (Suzuki and Okabe 1995; Okabe et al. 2000; Voronoï 1908; Aurenhammer et al. 2013) to p non-intersecting subsets, and each has a facility located at the optimal location for the subset minimizing the weighted sum of distances. Therefore, the two solutions will be the same if every single facility solution for a subset is located at a demand point.

When the number of demand points increases, one would expect that the probability that the Weber solution point is on a demand point converges to 1. There is "no space" left between demand points, when the number of demand points increases to infinity. However, Drezner and Simchi-Levi (1992) showed that when n demand points are randomly generated in a unit circle, then the probability that the Weber solution with Euclidean distances is on a demand point is approximately $\frac{1}{n}$. This counter-intuitive result was verified by simulation. Consider the $n = 654$, $p = 10$ case. For $n = 654$ and one facility, the probability that the solution is on a demand point is about 0.15%. When 10 about equally sized subsets are created, the probability that the location of the facility serving a subset is on a demand point is 1.5%. The probability that all 10 facilities are located at demand points is basically zero (10^{-18}). The numbers for $n = 1060$ are even lower. The optimal discrete will hardly ever be the same as the optimal continuous.

The subsets are separated by perpendicular bisectors between the facilities. If the facilities are moved, the perpendicular bisectors can change quite a bit, even for small perturbations. The partitioned subsets for the optimal continuous solution can be significantly different from those of the optimal discrete solution. This may explain why using the optimal discrete solution as a starting solution is not the "best" choice, as we often observed in the computational results.

On the other hand, even if the optimal discrete solution leads to the optimal (or best known) continuous one, there are likely many other discrete solutions of varying quality that will also produce the same result. That is, there may be several discrete solutions that map onto the same continuous solution (local optimum obtained by the RATIO algorithm). For example, consider the extreme case, instance $n = 500$, $p = 5$ in Table 5, where all 1000 best discrete solutions map onto the same (best known) continuous solution.

The computational results show that using "good" discrete starting solutions from our genetic algorithm often yield better continuous solutions than those obtained by optimal or near-optimal discrete solutions. Not only do we get better solutions, but the CPU time is also reduced by one or more orders of magnitude. However, this is not uniformly the case. As noted above, the optimal and near-optimal discrete solutions were able to beat the genetic starting solutions for several larger instances (see Tables 8 and 9). Consider, for example, the $n = 654$, $p = 85$ instance, where the percent deviation obtained with the 100 best discrete solutions and the genetic solutions were, respectively, 0.18% and 0.54%, a difference of 0.36% in favor of the optimal/near optimal discrete solutions.

Similar conclusions are drawn by Drezner and Drezner (2016). They investigated sequential location of two facilities. Once the first facility is located, the second facility is located at its optimal location considering the location of the first one (termed the conditional location model (Minieka 1980; Drezner 1995)). They found that for the 2-median and 2-center problems, random location of the first facility is better, while for locating two new competing facilities, the optimal location for the first facility is better.

5 Conclusions

In this paper we examined the use of discrete starting solutions for solving the continuous p-median problem. One would expect that better discrete starting solutions should result in better continuous solutions. However, we found that this is not the case. Moderately good starting solutions can often yield better continuous solutions. We tested using as starting solutions: (i) the optimal discrete solution, (ii) the 100 best discrete solutions, and (iii) the final population of a genetic algorithm designed to find good discrete solutions.

The 100 best discrete solutions provided quite good continuous solutions. However, run time was two orders of magnitude longer than the time required by the genetic algorithm. If we would allow the genetic algorithm to consume that much run time, it is clear that best continuous solutions obtained would be much improved from the same population of starting solutions. Actually, the best performance was observed when many populations of a very short genetic algorithm, whose results are not so good, were applied. We also evaluated the effect of the quality of the genetic algorithm on the final outcome. It is clear that when using about the same run time, better genetic final populations (fewer final populations though), perform worse. Statistical analysis shows that this inferiority is statistically significant, as evidences by extremely small p-values.

We found that a diverse set of "good" discrete starting solutions is a better option than relying on a single optimal (or near-optimal) discrete solution. Furthermore, this diverse set may be obtained with less computational effort. Our research also shows that at some point, further improvement in quality of the discrete starting solution may actually result in inferior continuous solutions. This turning point would depend on the local search being implemented in the continuous space. This notion of a "turning point" may provide some insight on what constitutes a good set

of starting solutions. If the current best continuous solution is reached from starting solution X^*, and there is no improvement obtained from several better quality starting solutions, we might surmise that the current set of starting solutions is a good one for the applied heuristic. This could also become an interesting direction for future research.

Future research should investigate other continuous search algorithms, such as the standard Cooper method (which is faster but less powerful than the RATIO algorithm we used), to determine the effect on our conclusions of the quality of the local search. Also, it would be useful to investigate using discrete starting solutions for solving other continuous multiple facility location problems, such as the *p*-center, and finding whether "best" discrete solutions perform better or worse than "good" discrete solutions.

References

Aboolian R, Berman O, Krass D (2007) Competitive facility location model with concave demand. Eur J Oper Res 181:598–619

Abramowitz M, Stegun I (1972) Handbook of Mathematical Functions. Dover Publications Inc., New York

Alp O, Drezner Z, Erkut E (2003) An efficient genetic algorithm for the *p*-median problem. Ann Oper Res 122:21–42

Aurenhammer F, Klein R, Lee D-T (2013) Voronoi Diagrams and Delaunay Triangulations. World Scientific, New Jersey

Berman O, Drezner Z, Wesolowsky GO (2003) Locating service facilities whose reliability is distance dependent. Comput Oper Res 30:1683–1695

Bongartz I, Calamai PH, Conn AR (1994) A projection method for ℓ_p norm location-allocation problems. Math Program 66:238–312

Brimberg J, Drezner Z (2013) A new heuristic for solving the *p*-median problem in the plane. Comput Oper Res 40:427–437

Brimberg J, Drezner Z (2020) Improved starting solutions for the planar *p*-median problem. Yugoslav J Oper Res. https://doi.org/10.2298/YJOR2003

Brimberg J, Hodgson MJ (2011) Heuristics for location models. In: Eiselt HA, Marianov V (eds) Foundations of Location Analysis: International Series in Operations Research & Management Science, vol 155. Springer, New York, pp 335–355

Brimberg J, Hansen P, Mladenović N, Taillard E (2000) Improvements and comparison of heuristics for solving the uncapacitated multisource Weber problem. Oper Res 48:444–460

Brimberg J, Hansen P, Mladonovic N, Salhi S (2008) A survey of solution methods for the continuous location allocation problem. Int J Oper Res 5:1–12

Brimberg J, Drezner Z, Mladenović N, Salhi S (2014) A new local search for continuous location problems. Eur J Oper Res 232:256–265

Brimberg J, Drezner Z, Mladenovic N, Salhi S (2017a) Using injection points in reformulation local search for solving continuous location problems. Yugoslav J Oper Res 27:291–300

Brimberg J, Mladenović N, Todosijević R, Urošević D (2017b) Less is more: solving the max-mean diversity problem with variable neighborhood search. Inf Sci 382:179–200

Church RL (2019) Understanding the Weber location paradigm. In: Eiselt HA, Marianov V (eds) Contributions to Location Analysis - In Honor of Zvi Drezner's 75th Birthday. Springer Nature, Switzerland, pp 69–88

Cooper L (1963) Location-allocation problems. Oper Res 11:331–343

Cooper L (1964) Heuristic methods for location-allocation problems. SIAM Rev 6:37–53

CPLEX, IBM ILOG (2009). V12. 1: User's Manual for CPLEX. In: International Business Machines Corporation, Incline Village, NV, 46(53):157

Daskin MS, Maass KL (2015). The p-median problem. In: Laporte G, Nickel S, da Gama FS (eds.) Location science. Springer, New York, pp 21–45
Daskin MS (1995) Network and Discrete Location: Models, Algorithms, and Applications. John Wiley & Sons, New York
Drezner Z (1995) On the conditional *p*-median problem. Comput Oper Res 22:525–530
Drezner T, Drezner Z (2016) Sequential location of two facilities: Comparing random to optimal location of the first facility. Ann Oper Res 246:1–15
Drezner Z, Drezner TD (2020) Biologically inspired parent selection in genetic algorithms. Ann Oper Res 287:161–183
Drezner Z, Marcoulides GA (2003) A distance-based selection of parents in genetic algorithms. In: Resende MGC, de Sousa JP (eds) Metaheuristics: Computer Decision-Making. Kluwer Academic Publishers, Boston, pp 257–278
Drezner Z, Salhi S (2017) Incorporating neighborhood reduction for the solution of the planar *p*-median problem. Ann Oper Res 258:639–654
Drezner Z, Simchi-Levi D (1992) Asymptotic behavior of the Weber location problem on the plane. Ann Oper Res 40:163–172
Drezner Z, Zerom D (2016) A simple and effective discretization of a continuous random variable. Commun Stat Simul Comput 45:3798–3810
Drezner Z, Klamroth K, Schöbel A, Wesolowsky GO (2002) The Weber problem. In: Drezner Z, Hamacher HW (eds) Facility Location: Applications and Theory. Springer, Berlin, pp 1–36
Drezner Z, Brimberg J, Salhi S, Mladenović N (2016) New local searches for solving the multi-source Weber problem. Ann Oper Res 246:181–203
Drezner T, Drezner Z, Kalczynski P (2020) Directional approach to gradual cover: the continuous case. CMS. https://doi.org/10.1007/s10287-020-00378-1
Edmands S (2007) Between a rock and a hard place: evaluating the relative risks of inbreeding and outbreeding for conservation and management. Mol Ecol 16:463–475
Eilon S, Watson-Gandy CDT, Christofides N (1971) Distribution Management. Hafner, New York
Fenster CB, Galloway LF (2000) Inbreeding and outbreeding depression in natural populations of Chamaecrista fasciculata (Fabaceae). Conserv Biol 14:1406–1412
Feo TA, Resende MG (1995) Greedy randomized adaptive search procedures. J Global Optim 6:109–133
Freeman S, Harrington M, Sharp JC (2014) Biological Science. Second Canadian Edition, Pearson, Toronto
Garey MR, Johnson DS (1979). Computers and intractability: A guide to the theory of NP-completeness. freeman, San Francisco
Goldberg DE (2006) Genetic algorithms. Pearson Education, Delhi
Hansen P, Mladenović N, Taillard É (1998) Heuristic solution of the multisource Weber problem as a *p* -median problem. Oper Res Lett 22:55–62
Holland JH (1975) Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor
Ince EL (1926) Ordinary Differential Equations. Reprinted in 1956 by Dover Publications, Inc., USA
Kalczynski P Drezner Z (2020). The obnoxious facilities planar *p*-median problem. In review, arXiv:2004.03038 [math.OC]
Kalczynski P, Brimberg J, Drezner Z (2020a). The importance of good starting solutions in the minimum sum of squares clustering problem. In review, arXiv:2004.04593 [cs.LG]
Kalczynski P, Goldstein Z, Drezner Z (2020b). Partitioning items into mutually exclusive groups. In review, arXiv:2002.11536 [math.OC]
Kariv O, Hakimi SL (1979) An algorithmic approach to network location problems. II: The *p*-medians. SIAM J Appl Math 37:539–560
Krau S (1997). *Extensions du problème de Weber*. PhD thesis, École Polytechnique de Montréal
Kuenne RE, Soland RM (1972) Exact and approximate solutions to the multisource Weber problem. Math Program 3:193–209
Kutta W (1901) Beitrag zur näherungweisen integration totaler differentialgleichungen. Z Angew Math Phys 46:435–453
Love RF, Morris JG, Wesolowsky GO (1988) Facilities Location: Models & Methods. North Holland, New York
Megiddo N, Supowit K (1984) On the complexity of some common geometric location problems. SIAM J Comput 18:182–196
Minieka E (1980) Conditional centers and medians on a graph. Networks 10:265–272

Mladenović N, Todosijević R, Urošević D (2016) Less is more: basic variable neighborhood search for minimum differential dispersion problem. Inf Sci 326:160–171

Okabe A, Boots B, Sugihara K, Chiu SN (2000) Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. Wiley Series in Probability and Statistics. John Wiley, Hoboken

Reinelt G (1991) TSLIB a traveling salesman library. ORSA J Comput 3:376–384

ReVelle CS, Swain RW (1970) Central facilities location. Geograph Anal 2:30–42

Runge C (1895) Über die numerische auflösung von differentialgleichungen. Math Ann 46:167–178

Suzuki A, Okabe A (1995) Using Voronoi diagrams. In: Drezner Z (ed) Facility Location: A Survey of Applications and Methods. Springer, New York, pp 103–118

Voronoï G (1908) Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les paralléllοèdres primitifs. Journal für die reine und angewandte Mathematik 134:198–287

Weber A (1909) Über den Standort der Industrien, 1. Teil: Reine Theorie des Standortes. English Translation: on the Location of Industries. University of Chicago Press, Chicago, IL. Translation published in 1929

Wesolowsky GO (1993) The Weber problem: History and perspectives. Locat Sci 1:5–23