A new local search for continuous location problems[☆]

Jack Brimberg

Department of Mathematics and Computer Science, Royal Military College of Canada, Kingston, ON, K7K7B4, Canada

Zvi Drezner

Steven G. Mihaylo College of Business and Economics, California State University, Fullerton, CA 92634, USA

Nenad Mladenović

Department of Mathematical Sciences, School of Information Systems, Computing and Mathematics, University of Brunel, Uxbridge, Middlesex UB8 3PH, United Kingdom

Said Salhi[*]

Centre for Logistics & Heuristic Optimization, Kent Business School, University of Kent, Canterbury CT2 7PE, United Kingdom

Abstract

This paper presents a new local search approach for solving continuous location problems. The main idea is to exploit the relation between the continuous model and its discrete counterpart. A local search is first conducted in the continuous space until a local optimum is reached. It then switches to a discrete space that represents a discretisation of the continuous model to find an improved solution from there. The process continues switching between the two problem formulations until no further improvement can be found in either. Thus, we may view the procedure as a new adaption of *formulation*

[☆]This research has been supported in part by a Natural Sciences and Engineering Research Council of Canada Discovery Grant (NSERC #205041-2008) and by the UK Research Council EPSRC(EP/I009299/1).

[*]Corresponding author

Email addresses: Jack.Brimberg@rmc.ca (Jack Brimberg), zdrezner@fullerton.edu (Zvi Drezner), nenad.mladenovic@brunel.ac.uk (Nenad Mladenović), S.Salhi@kent.ac.uk (Said Salhi)

space search. The local search is applied to the multi-source Weber problem where encouraging results are obtained. This local search is also embedded within Variable Neighbourhood Search producing excellent results.

Keywords: Continuous Location, Weber problem, Space search formulation, Variable neighbourhood

---

1. Introduction

Location models generally require finding the location of a given number, say p, of new facility sites in order to serve in some optimal way (e.g., minimum cost) a given set of existing facilities, also known as customers or demand (or fixed) points. If the model is formulated in continuous space, a distance function is required to calculate the distance between pairs of points. Since the new facilities may be located anywhere in the continuous space or regions thereof, these models are referred to as site generating models (e.g., see Love *et* al. [32]). The distance functions most commonly used are the Euclidean norm and the rectangular (or Manhattan) norm; however, more sophisticated models of distance are available when more accurate estimates of actual travel distances are desired (e.g., Brimberg and Walker [11]).

The same location problem may be formulated in discrete space by restricting the potential new facility sites to a specified finite set of points in the continuous space. If these sites are chosen well, and a good algorithm or heuristic is available to solve the discrete formulation, we may anticipate a "good" solution to the original problem. For example, if we restrict the candidate facility sites to the given set of fixed points, the classical multi-source Weber problem, also known as the continuous location-allocation problem, converts to the classical (discrete) p-median problem. We may then try to obtain a good solution to the discrete model, and use it as a starting point for the continuous model.

Exploiting the relation between the p-median model and the continuous location-allocation model has been suggested as early as in the original work of Cooper ([14, 15]). Hansen *et* al. [27] tested a heuristic that first solves the p-median problem exactly using a primal-dual algorithm by Erlenkotter [21], and then completes one iteration of "continuous-space adjustment" by solving the p continuous single facility problems identified in the first phase. Brimberg *et* al. [10] examined this heuristic among others, and concluded that computation time became a limiting factor on larger problem instances.

Gamal and Salhi [24] used a similar approach where in the first phase, an effective heuristic is applied instead of an exact solution approach to solve the p-median problem.

In certain cases it may be shown that the continuous problem has a finite dominating set. For example, if the rectangular norm (l_1-norm) is used as the distance function, it is well known that an optimal solution of the continuous location-allocation problem exists where each of the facilities is located at a vertex of a grid formed by drawing horizontal and vertical lines (the fundamental directions of the l_1-norm) through each of the demand points. Since an optimal solution in the plane must also exist with all facilities inside the convex hull of the demand points (e.g., see Hansen *e*t al. [28]), or the smaller rectangular hull for the l_1-norm (Love *e*t al. [32]), a discrete formulation of the continuous location-allocation problem with a fewer number of nodes is also possible that guarantees an optimal solution of the original problem. This idea can be extended to the class of polyhedral (or block) norms (e.g., Ward and Wendell [45]; Ward *e*t al. [46]), although the grid will be more complicated in general due to a higher number of fundamental directions attributed to the norm. This discretisation of the continuous space does not extend to round metrics such as the Euclidean norm.

In practice the discrete formulation, whether or not it contains an optimal solution of the original continuous location problem, may become rather large to be tackled optimally. Aras *e*t al. [2] propose a discrete approximation to solve the capacitated multi-source Weber problem (CMSWP) with Euclidean, squared Euclidean and l_p distances with $1 < p < 2$. The authors discretise the solution space while increasing the number of potential sites by using the rectangular grid points that are within the convex hull of the customers. Two MILP formulations are proposed using this new set of potential sites, including some attempts in choosing a subset. Heuristic approaches such as a Lagrangean relaxation-based method, the $p-$ median heuristic of Hansen *e*t al.[27] and the cellular Heuristic of Gamal and Salhi [23] are also investigated. Aras *et al.* [3] adapt the previous approaches to the case of rectilinear distances whereas Durmaz *e*t al.[18] extend this discretisation approach to cater for uncertainty due to changes in the customer set. Very recently, Akyüz *et al.* [1] studied the CMSWP using two branch-and-bound techniques where one is related to the discretisation of the location space. Heuristics based on solving the discrete approximation of the CMSWP by Lagrangean Relaxation are proposed by Boyaci *et al.* [4].

For an overview of the continuous location-allocation problem, the in-

terested reader is referred to the survey paper by Brimberg *et* al. [9] and the references therein, while for the discrete p-median model and solution approaches the review by Mladenović *et* al. [34] can be useful.

The relation between discrete and continuous formulations may be extended to many other location models. For example, the less-studied continuous p-center problem becomes the better-known discrete p-center problem when candidate facility sites are once again restricted to the set of fixed points. The classical (discrete) simple plant location problem has more recently been modelled in continuous space by Brimberg and Salhi [13], and in a related paper by Brimberg *et* al. [12]. Indeed, the idea of exploring the relation between discrete and continuous location problems presents in our view a rich new area of research.

In this paper we present a new local search for solving continuous location problems that is based on reformulations of the problem in continuous and discrete space. The basic idea is to find a local optimum in continuous space using any convenient local search algorithm. The search space is then modified by reformulating the problem in discrete space. Here we introduce the idea of augmenting a specified set of fixed points (the current set) with the local optima obtained in the continuous phase. Thus, we solve exactly or heuristically a discrete problem where the nodes of the network now include the new facility sites obtained in the previous step. We switch back to continuous space using the discrete solution as the starting point. The procedure alternates between continuous and discrete spaces, always adding newly acquired facility sites to the current set in the discrete formulation, until no further improvement is found.

The local search outlined above incorporates elements of a metaheuristic known as formulation space search (FSS). The basic idea here as presented in Mladenović *et* al. [36] is to use different formulations of a combinatorial or global optimization problem in an iterative fashion, where in each formulation suitable local searches are used. For example, the authors applied two formulations in different coordinate systems of the circle packing problem with excellent results.

In formulation space search, the different formulations are all equivalent to each other. However, in our case, the discrete model is an approximation of the continuous model, thus presenting a fundamental departure from FSS. We may also argue, meanwhile, that the discrete formulation is equivalent to the continuous formulation in an asymptotic sense, as more facility sites generated in the continuous phase are added to the network.

The paper is organized as follows. In the next section we provide a basic framework for the proposed local search. Section 3 illustrates the local search on the multi-source Weber problem (MWP) using a well-known 50-customer problem from the literature. Larger problem instances of MWP are examined later in this section. Section 4 develops a variable neighbourhood search (VNS) heuristic for solving MWP that employs the proposed local search in its local search step. The same data sets are also tested here and superior computational results are obtained. The last section summarizes our conclusions and highlights some suggestions for further research.

2. The local search

We consider an unconstrained location problem of the general form

$$\min\ f(X_1, X_2, \ldots, X_p). \qquad (GLP)$$

where $X_i \in \mathbb{R}^N$ gives the unknown location of new facility $i, i = 1, \ldots, p$, and the objective function $f(.)$ represents some performance measure, such as total cost. Typically the location problem occurs in the plane, so that $N = 2$, and X_i is given by the Cartesian coordinates (x_i, y_i).

Consider as an illustration the classical multi-source Weber problem, which may be formulated as follows:

$$\min\ f(X_1, X_2, \ldots, X_p) = \sum_{j=1}^{n} w_j \min_{i=1,\ldots,p}\{\|X_i - A_j\|\}. \qquad (MWP1)$$

Here A_j denotes the known coordinates of customer j, $w_j > 0$, the known demand at A_j, and $\|X_i - A_j\|$ the Euclidean distance between the pair of points X_i and A_j, $i = 1, \ldots, p, j = 1, \ldots, n$. The objective function gives a sum of weighted distances from the demand points to their nearest facilities, and thus, represents a measure of the total cost of the current solution.

As a second illustration, consider the continuous weighted p-center problem, which may be formulated as follows:

$$\min\ g(X_1, X_2, \ldots, X_p) = \max_{j=1,\ldots,n}\{v_j \min_{i=1,\ldots,p}\{\|X_i - A_j\|\}\}, \qquad (MCP)$$

where weight $v_j > 0$ reflects the "importance" of demand point A_j, and the remaining notation is the same as for $(MWP1)$. The objective function gives the maximum (weighted) distance between the demand points and their nearest facilities, and thus, represents a measure of the quality of service of the current solution.

Other examples include the use of ordered medians (e.g., Nickel and Puerto [38]), or the use of negative-valued weights in $(MWP1)$ or (MCP) to model obnoxious facilities (e.g., Erkut and Neuman [20]). In the latter case, restrictions on the location of the facilities may be required in order to guarantee that an optimal solution exists. Capacity constraints and flow variables may also be included in the model.

We now describe the basic steps of the proposed local search for problems of type (GLP). To differentiate between the continuous and discrete formulations of the problem, we let (GLP) denote the original continuous formulation and $(GLP)'$ the discrete approximation. Let S denote a finite set of identified potential sites for the new facilities, and X a subset of p of these sites. For example, $S = \{A_1, \ldots, A_n\}$, where typically $n >> p$, has been recommended in earlier works as noted above. The discrete formulation of the problem is then given by:

$$\min_{X \subset S} f(X). \qquad (GLP)'$$

Assuming w.l.o.g. that the new facilities are identical, there are $\begin{pmatrix} M \\ p \end{pmatrix}$ combinations of candidate solutions of $(GLP)'$, where M is the cardinality of the set S. (The case of non-homogeneous new facilities is readily handled using a solution space that contains up to a factor of $p!$ more points.) Finally, we let L_C and L_D denote the selected local search operators for (GLP) and $(GLP)'$, respectively. These search engines stop at a current solution if, and only if, a better solution cannot be found in the specified neighbourhood of this point.

A simple approach combining the discrete and continuous formulations of the problem is given next. (The main heuristic will be given afterwards.) The general idea is to solve $(GLP)'$ first either heuristically or exactly, and use the obtained solution as the starting point for the local search in (GLP). Although the idea of solving $(GLP)'$ first has been suggested before (e.g., Hansen *et* al. [27]), the inclusion of a second phase of local search in the continuous space is, to the best of our knowledge, a general approach yet to

be formalized.

Algorithm 1: Basic local search (BLS)

Step 1: Select an initial set S for $(GLP)'$.

Step 2: Solve $(GLP)'$ heuristically or exactly to obtain an initial solution $X^0 = \{X_1^0, \ldots, X_p^0\}$.

Step 3: $L_C(X^0) \rightarrow X^C$, and stop (final solution = X^C).

In effect, algorithm 1 is a 2-phase approach that may be viewed as a hybrid heuristic. The solution of the discrete problem $(GLP)'$ is used as a good starting solution for the local search in continuous space (step 3 above) in much the same way that a constructive heuristic could be used.

We also emphasize that BLS is distinctly different from the p-median heuristic used in Brimberg *et* al. [10] in the following ways: (i) In the latter method the discrete p-median problem is solved exactly, followed by one step of continuous adjustment only (i.e., it solves p independent single facility problems using the customer partition found in the discrete solution). Thus the p-median heuristic does not guarantee to reach a local minimum in continuous space as our heuristic does. (ii) The exact solution of the discrete problem limits the p-median heuristic to smaller problem instances. (iii) The discrete formulation in the p-median heuristic defines the set of candidate sites as given by the set of customer locations. In our heuristic the set of candidate sites is defined in a general way. (It can be the customer set, part of the customer set or can have other "attractive" points added to it.)

The next algorithm provides a new way of combining the discrete and continuous phases of the search, which we term ''Reformulation local search".

Algorithm 2: Reformulation local search (RLS)

Step 1: Select an initial set S for $(GLP)'$ and an initial solution $X^0 = \{X_1^0, \ldots, X_p^0\}$.

Step 2 (solving the continuous problem): $L_C(X^0) \rightarrow X^C$ (where $X^C \neq X^0$, only if $f(X^C) < f(X^0)$).

Step 3 (augmenting S): $S \leftarrow S \cup X^C$.

Step 4 (solving the discrete problem): $L_D(X^C) \rightarrow X^D$ (where $X^D \neq X^C$, only if $f(X^D) < f(X^C)$).

Step 5: If $X^D = X^C$, stop (final solution = X^D); else $X^0 \leftarrow X^D$ and return to step 2.

Note that we could use the solution found in step 2 of BLS as the starting solution within the general framework of RLS. This would consequently guarantee that RLS is never worse than BLS. However, the aim of our study is to present an adaptive approach that augments the set of potential sites in discrete space by adding newly found points in continuous space in a systematic and simple way and not to promote efficient implementations which could without any doubt be useful at producing better solutions. This latter objective could be achieved by the combined BLS/RLS approach noted above, or by using powerful global optimisation techniques such as metaheuristics to produce such initial discrete high quality solutions (see Mladenović *e*t al. [34]).

3. Application to the multi-source Weber problem

The multi-source Weber problem, also referred to as the continuous location-allocation problem, is a well-studied model in location theory. As observed above in ($MWP1$), the objective is to generate optimal sites in continuous space, notably $\mathbb{R}^2$, for a given number of new facilities in order to minimize a sum of transportation (or service) costs to a given set of customers at known point locations and with known demands.

Thus, we wish to locate p new facility sites in the plane in order to service a set of n customers at known locations $A_j = (a_j, b_j)$ and with given demands (weights) $w_j > 0, j = 1, \ldots, n$. The basic version of the (uncapacitated) multi-source Weber problem given in ($MWP1$) may be rewritten in the following equivalent form (e.g., Love *e*t al. [32]), which is more suitable for gradient-based search methods:

$$\min_{w,X} \sum_{i=1}^{p} \sum_{j=1}^{n} w_{ij} \|X_i - A_j\| \qquad (MWP2)$$

subject to

$$\sum_{i=1}^{p} w_{ij} = w_j \quad \forall j,$$
$$w_{ij} \geq 0 \ \ \forall i \ , \ j \ ,$$

where $X = (X_1, \ldots, X_p)$ designates the set of location decision variables, with $X_i = (x_i, y_i)$ being the unknown location of facility $i, i = 1, \ldots, p; w = (w_{ij})$ designates the set of allocation decision variables, where w_{ij} gives the flow to customer j from facility $i, i = 1, \ldots, p, j = 1, \ldots, n$; and $\|X_i - A_j\| = \sqrt{(x_i - a_j)^2 + (y_i - b_j)^2}$ is the Euclidean distance between facility i and customer j. Note that the objective function measures the total service cost while the constraints ensure that all the customer demands are satisfied. The model above uses the Euclidean norm to estimate travel distances or times, but other distance functions, such as the Manhattan norm or, more generally, the l_p norm, have also been employed (e.g., see Love *et* al. [32] or Francis *et* al. [22] for a review).

Finding an optimal solution to $(MWP2)$ is a difficult proposition due to the non-convexity of the objective function and the existence of multiple local minima. This fact was known to the originator of the model; see Cooper [14, 15]. It was shown later that the problem is NP-hard (Megiddo and Supowit [33]). Brimberg *et* al. [8] demonstrate the difficult nature of $(MWP2)$ on a 50-customer problem taken from the classical textbook by Eilon *et* al. [19] by using 10,000 random restarts of Cooper's well-known alternating heuristic for $p = 5, 10, 15$, to generate $272, 3008$ and 3363 different local minima, respectively. Furthermore, the worst deviation from the optimal solution was respectively, 47%, 66% and 70%, while the optimal solution was obtained 690 times for $p = 5, 34$ times for $p = 10$ and only once for $p = 15$. Such relatively small instances are useful in demonstrating the tendency for the number of local minima to increase exponentially with problem size as defined by n and p.

Early attempts to solve the problem exactly are given in Kuenne and Soland [31] and Ostresh [29]; however the branch-and-bound algorithms were capable of solving at the time only very small instances, of the order of

$n = 15, p = 4$ and $n = 50, p = 3$. Rosing [40] was able to incorporate improvements in the methodology that solved problems with $n = 30, p = 5$ and $n = 25, p = 6$. It was not until quite recently that the original 50-customer problem in Eilon *et* al. [19] was solved exactly using a novel column generation approach combined with global optimization and branch-and-bound (Krau [30]). This author was also able to obtain exact solutions for instances with up to $n = 287$ customers (ambulance problem from Bongartz *et* al.[5]) and up to $p = 100$ facilities by utilizing a dual formulation of the problem. A bundle method in the l_1 norm (du Merle *et* al.[17]) is added to stabilize the solution of the dual, leading to an algorithm in Hansen *et* al. [25] that successfully solves problems up to $n = 1000$ and $p = 100$.

Despite these advances, large scale problems found in the literature remain unsolvable by exact methods. Furthermore, the newer algorithms tend to be highly sensitive to the starting solution, and so, require state-of-the-art heuristics to obtain the best initial solution possible. Thus, advances in heuristic approaches are continually sought. For instance, Taillard [43] solves very large centroid problems by using efficient clustering techniques.

Brimberg and Drezner [6] suggest an improved alternate algorithm which is a modified version of the well known locate-allocate of Cooper[14, 15] enhanced by a transfer follow-up. In brief, instead of finding the optimal location within each cluster using the Weiszfeld recursive equation [47] and then re-allocating customers based on the new facility locations, the authors randomly choose one facility at a time to re-locate and re-assign customers to their nearest facilities after each re-location. A flag stating whether or not a facility has been affected is activated and the process is repeated until all flags remain unchanged. This scheme has the advantage of being less rigid and hence avoids entrapment in the same local optimum obtained by the systematic approach of Cooper's locate-allocate. An exchange of customers based only on those that lie on the boundary of two nearest facilities is then examined for possible reallocation (allocate a customer to its second nearest facility) and the saving due to these two affected facilities is evaluated to see whether or not the swap is worthwhile. The examination of boundary customers is a powerful and efficient allocation scheme as also shown in Salhi and Sari [41] when studying a class of multi-depot vehicle routing problems.

Very recently, Drezner *et* al. [16] propose an effective constructive heuristic that finds a good initial solution by combining the drop method and the gravity concept (see Brimberg *et* al. [7] for its details), a decomposition method that relies upon those triangles that constitute the nodes of the De-

launay triangulation of the set of facilities, and a concentric tabu search whose set of neighbourhoods is defined by different radii around the current set of facility locations. Sophisticated methods such as these and others may be used as local search operators (L_C or L_D) in algorithm RLS, resulting in a rich variety of heuristics embedded in the RLS framework.

3.1. An illustration of the advantage of RLS

Consider a simple problem comprising 8 customers with coordinates and weights tabulated below (also see Figure 1):

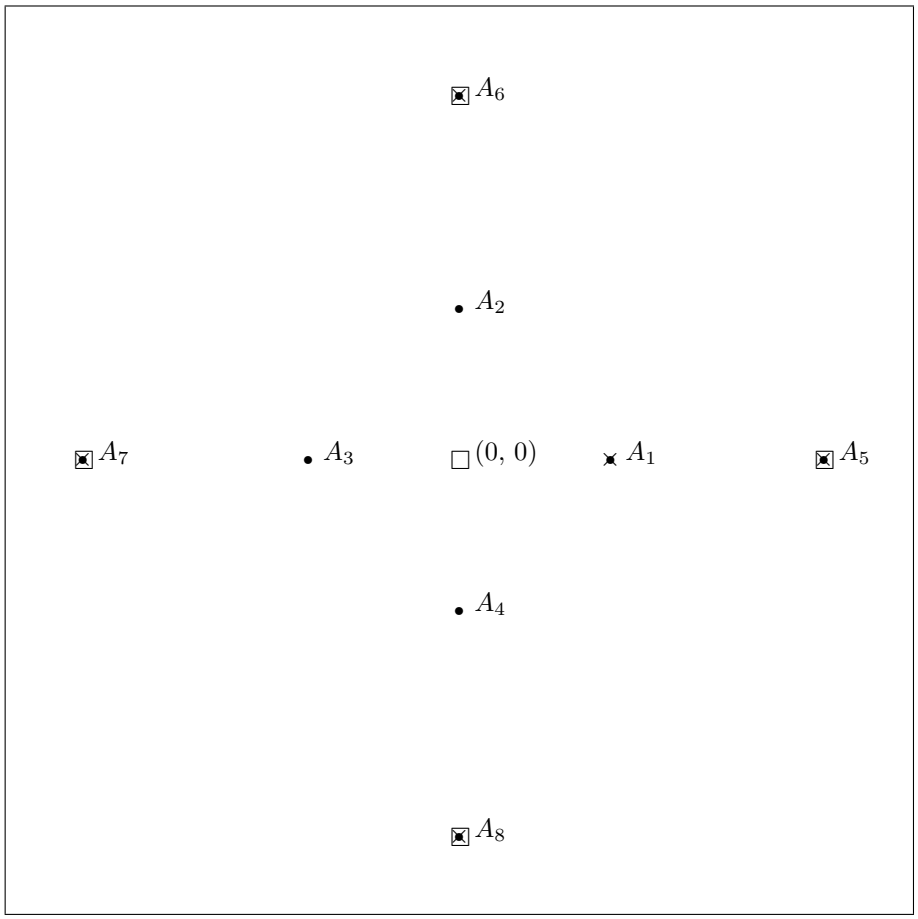
i	x_i	y_i	w_i
1	1	0	1
2	0	1	1
3	-1	0	1
4	0	-1	1
5	2.41	0	10
6	0	2.41	10
7	-2.41	0	10
8	0	-2.41	10

Note that the external demand points are each given a weight of 10 while the internal ones have unit weights. The objective is to find optimal locations for $p = 5$ facilities.

Let the BLS algorithm begin by randomly selecting five demand points from the set $S = \{A_j, j = 1, \ldots, 8\}$. Thus, there must be at least one external point and one internal point in this initial solution. A simple vertex exchange heuristic as the local search in discrete space will then move facilities from internal points to the heavier external points until the four external points are covered and a single facility (of the five) remains at an internal point (see the configuration in Figure 1). This yields a local minimum which is also a global solution of the given discrete problem. A subsequent Cooper-style heuristic will fail to move the facilities from their current positions since the current solution is also a local minimum in continuous space. Thus the BLS terminates with an objective value of $3 \times 1.41 = 4.23$. Meanwhile if we let RLS start from five random points in continuous space, it is possible to capture the Weber point (0,0) and subsequently the optimal solution denoted by small squares in Figure 1 with objective value of 4.0.

This simple example illustrates the advantage of adding Weber points (i.e., solutions from a continuous phase) in the discrete approximation of

Figure 1: The Illustration Problem



• Customer □ Optimal solution × BLS solution

the original continuous problem. This example also shows that BLS is not globally convergent, whereas RLS has the inherent flexibility to be so.

3.2. An extensive computational experiment on a small instance

We now test the basic local search (Algorithm BLS) and the reformulation local search (Algorithm RLS) on a relatively small instance, the well-known 50-customer problem from Eilon *e*t al. [19]. The benchmark used for comparison is the classical multi-start Cooper method (MALT).

In brief, MALT consists of a) randomly generating the locations of p facilities, either from the rectangle that encloses all customers, or from potential sites such as the demand points, b) allocating each customer, to its nearest

facility to determine p clusters, c) applying the Weiszfeld procedure to obtain an optimal location within each cluster, and d) repeating the allocation and the location steps (b) and (c) until there is no change in the total weighted distance or in the facility configuration. The entire process is repeated several times from random starting points, and the best solution is selected.

All methods were programmed in Fortran 95 compiler release 5.50e and run on a PC with 1.8GHz processor. Cooper's alternating algorithm is chosen as the local search operator in continuous space (L_C). The discrete search operator (L_D) is a simple vertex swap originally proposed by Teitz and Bart [44]. This local search is based on the exchange of one facility from its current location on a vertex to an unoccupied vertex on the network, with customers re-allocated to their closest facilities in each configuration. All such vertex exchanges are examined and the one giving the best improvement is taken. The process is repeated until there is no further improvement with respect to this facility exchange neighbourhood. In our implementation of this local search, the change in the objective function resulting from a vertex exchange (or 'swap') is calculated using an efficient procedure by Whitaker [48], that requires the retention of second nearest facility distances. The Whitaker improvement avoids unnecessary repetition of calculations on parts of the objective function that are unchanged.

In summary, we note that the most basic local searches are employed in the two phases of RLS. We repeat the experiment 100 times for each value of p and for each heuristic to gather basic statistics such as the number of different local minima found, the number of times the best, second best and third best solutions of each respective method are obtained, as well as the number of times a local minimum is obtained whose deviation from the optimum is within three thresholds which are set to 0.0005, 0.005 and 0.05. Table 1 summarises these statistics. Note that the first best solution is also globally optimal in all instances for BLS and RLS but not for MALT.

It can be observed that the idea of augmenting the set of potential sites with Weber locations in RLS increases the power of the method at obtaining a larger number of occurrences of the best solution as well as guaranteeing a larger number of occurrences of local minima whose fractional deviation is less than 0.0005. Note that this does not guarantee that there will always be many successes at reaching a global minimum even if the method is robust at generating excellent solutions around the global minimum. This observation can be seen in Table 1 for RLS and $p = 20$ where there are only 2 occurrences of 'first-best' but 71 of second-best that happens to lie within a

tiny deviation of 0.0005 from the global minimum. With regard to MALT, we see that it obtained many more distinct local minima, and as a result, very few occurrences of its first, second and third-best solutions. BLS seems to perform for this small example nearly as well as RLS. This shows the improvements over MALT are quite substantial when the search is extended to discrete space. It is also worth noting that RLS produces a larger number of solutions within 0.0005 in all cases except for $p = 25$. For instance, when $p = 20$, the number of very good solutions (within 0.0005) is extremely high at 73 compared with 3 only for BLS. The exceptional case at $p = 25$ may have a simple explanation. As the number of facilities increases, more and more of them will be located at fixed points in an optimal solution. Thus the relative performance of BLS improves for these very large values of p. Though the aim is not to compare these two approaches, the above result demonstrates the robustness of the innovative and simple approach used in RLS.

In our implementation, MALT and RLS both begin with uniformly random continuous facility locations within the rectangle that encloses all the fixed points (eg., Scott [42]). BLS on the other hand starts with a random selection of p nodes from S in its first phase when solving the discrete p-median problem. RLS and BLS both use the set of fixed points as the initial set S; but BLS stops after finding the continuous location (or Weber) points in phase 2, whereas RLS continues augmenting the set of potential sites which then allows the new Weber points to be used in the next local search on the larger network. According to these encouraging empirical results, further experiments are conducted next to test our proposed BLS and RLS approaches on larger data sets and with varying values of p.

3.3. Computational results on larger data sets

We conducted an extensive empirical experiment on four data sets commonly used for the multi-source Weber problem (see Brimberg *e*t al.[10]) as a platform to test our methodology. These include the 50-customer problem in Eilon *e*t al. [19], the 287-customer ambulance problem from Bongartz *e*t al. [5], and the 654- and the 1060-customer problems listed in the TSP library (see [39]). Note that the summary results of the first data set are provided in the previous subsection in Table 1.

We run each instance for the same amount of CPU time for each method described in the previous section. For each instance, the time allowed for n=287, 654 and 1060 is 20, 120 and 300 seconds respectively. The results are

p	Local Minima Infos	MALT	BLS	RLS
5	# Distinct Local Minima (LM)	50	5	4
	# 1st best	7	18	19
	# 2nd best	3	31	32
	# 3rd best	4	25	20
	# LM with dev ≤ 0.0005	10	18	**19**
	# LM with dev ≤ 0.005	19	49	51
	# LM with dev ≤ 0.05	57	90	96
10	# Distinct Local Minima (LM)	97	3	3
	# 1st best	2	48	49
	# 2nd best	1	33	33
	# 3rd best	1	19	18
	# LM with dev ≤ 0.0005	0	48	**49**
	# LM with dev ≤ 0.005	4	81	49
	# LM with dev ≤ 0.05	16	100	100
15	# Distinct Local Minima (LM)	100	2	2
	# 1st best	1	67	70
	# 2nd best	1	33	30
	# 3rd best	1	0	0
	# LM with dev ≤ 0.0005	0	67	**70**
	# LM with dev ≤ 0.005	1	67	70
	# LM with dev ≤ 0.05	4	100	100
20	# Distinct Local Minima (LM)	100	4	5
	# 1st best	1	3	2
	# 2nd best	1	67	71
	# 3rd best	1	18	12
	# LM with dev ≤ 0.0005	0	3	**73**
	# LM with dev ≤ 0.005	0	70	97
	# LM with dev ≤ 0.05	1	88	97
25	# Distinct Local Minima (LM)	98	2	2
	# 1st best	1	16	11
	# 2nd best	1	84	89
	# 3rd best	1	0	0
	# LM with dev ≤ 0.0005	0	**16**	11
	# LM with dev ≤ 0.005	0	16	11
	# LM with dev ≤ 0.05	0	16	11

Table 1: Detailed comparison of local searches on smallest data set ($n = 50$)

given in Tables 2-4 where the first two columns list the value of p and the best known solution as reported in Table 1 of Brimberg *et* al. [10]; this is followed by 3 columns giving the deviation of the best solution from the best known and then 3 columns providing the times (in secs) when the best solution was found. A bold format in Table 3 is used to denote instances where an improvement over the best known solution was obtained. For $n = 287$, 'Best' represents the optimal solution as shown by Krau [30].

We observe again a substantial improvement in solution quality of BLS and RLS over the conventional MALT. Meanwhile BLS and RLS yield comparable results. Furthermore, the quality of the solutions obtained by both local searches is very good. For the 287-customer problem (Table 2), MALT fails miserably while BLS and RLS both are on average 0.04% above the optimal solution. For the larger data sets (Tables 3 and 4), the average deviations of BLS and RLS from the best-known solutions obtained by the relatively sophisticated metaheuristic methods in Brimberg *et* al.[10]) are only of the order of 1/5%. Also, it is observed that RLS, on the larger instances, produces its best solution slightly earlier than the other two approaches.

The computational results above support the notion that a combined local search in discrete and continuous spaces can be a powerful heuristic approach. The combined local search may be as simple as the BLS algorithm presented here, or as flexible as the proposed RLS algorithm. In spite of the comparable results of BLS and RLS on the limited data sets examined here, we recommend RLS over BLS for the following reasons:

(i) RLS provides a general framework for building local searches, and hence, is a more adaptable methodology. In fact, BLS may be viewed as a stripped-down implementation of RLS.

(ii) RLS has a stronger theoretical basis than BLS, as the simple example in subsection 3.1 amply demonstrates.

4. Variable Neighbourhood Search

In this section, we incorporate our new local search within a powerful metaheuristic known as variable neighbourhood search (VNS). VNS is a metaheuristic for solving combinatorial and global optimization problems. Its basic idea is to systematically change neighbourhoods in the search for a better solution (see Mladenović and Hansen [35]). The main loop consists of

		Deviation (%)			*CPU Times (sec.)*		
p	Best*	MALT	BLS	RLS	MALT	BLS	RLS
2	14427.59	0.00	0.00	0.00	0.02	0.02	0.03
3	12095.44	0.00	0.00	0.00	7.02	0.16	0.23
4	10661.48	0.00	0.00	0.00	8.09	0.28	0.27
5	9715.63	0.01	0.01	0.01	7.58	0.23	0.09
6	8787.56	0.03	0.02	0.02	1.89	0.25	0.78
7	8160.32	0.03	0.03	0.03	6.33	0.14	0.17
8	7564.29	0.09	0.01	0.01	16.81	0.05	1.02
9	7088.13	2.09	0.01	0.01	2.64	0.81	0.53
10	6705.04	2.95	0.00	0.00	7.89	0.06	10.02
11	6351.59	3.18	0.11	0.11	18.11	0.14	0.16
12	6033.05	5.66	0.24	0.24	15.31	1.20	1.23
13	5725.19	6.39	0.26	0.26	7.66	0.06	2.08
14	5469.65	6.52	0.01	0.01	18.33	2.97	7.06
15	5224.70	7.88	0.01	0.01	16.19	0.70	6.25
20	4148.84	15.50	0.03	0.03	15.89	1.48	2.92
25	3348.71	29.51	0.00	0.00	6.30	2.78	7.67
30	2716.90	41.15	0.03	0.03	0.08	12.14	1.73
35	2238.18	62.25	0.01	0.01	3.80	10.05	2.78
40	1900.84	79.37	0.01	0.01	10.31	3.41	15.16
45	1630.31	85.53	0.01	0.01	1.67	1.61	3.39
50	1402.58	99.77	0.00	0.00	11.58	2.13	0.23
55	1203.99	122.90	0.00	0.00	5.17	12.17	3.53
60	1055.14	156.81	0.01	0.01	16.78	17.36	8.92
65	924.56	146.31	0.02	0.02	15.17	1.09	4.28
70	814.22	187.20	0.03	0.03	6.45	2.97	0.66
75	730.04	198.94	0.05	0.05	4.86	8.34	6.17
80	655.38	182.22	0.06	0.06	12.56	3.22	15.19
85	588.37	202.34	0.13	0.13	17.97	15.47	19.17
90	529.21	252.54	0.07	0.07	18.30	10.31	7.11
95	480.86	258.83	0.02	0.01	5.73	17.20	8.70
100	441.24	286.42	0.01	0.01	19.13	12.14	10.81
	Average (%)	78.79	0.04	0.04	9.86	4.55	4.79

Table 2: Comparison of MALT, BLS and RLS ($n = 287$)

* These are optimal solutions as shown by Krau [30].

		Deviation (%)			*CPU Times (sec.)*		
p	Best	MALT	BLS	RLS	MALT	BLS	RLS
5	209068.80	0.00	0.00	0.00	0.23	0.23	0.27
10	115339.03	0.00	0.00	0.00	7.06	1.47	2.48
15	80177.04	0.05	0.01	0.01	4.47	8.20	14.34
20	63389.02	1.95	0.01	0.01	30.16	23.16	27.13
25	52209.51	6.24	0.01	0.01	98.19	109.59	113.92
30	44705.19	7.74	0.01	0.01	22.33	24.33	80.28
35	39257.27	12.48	0.11	0.11	7.41	28.80	21.94
40	35704.41	14.10	0.16	0.13	117.14	107.03	61.31
45	32306.97	13.98	0.15	0.15	117.19	40.09	69.16
50	29338.01	14.67	0.01	0.05	22.08	68.34	13.23
55	26699.17	23.76	0.06	0.02	15.42	98.16	55.27
60	24504.39	25.60	0.03	0.03	51.66	43.87	33.67
65	22747.10	31.06	0.10	0.06	76.13	61.05	8.44
70	21465.44	32.59	0.06	0.12	88.88	41.89	34.56
75	20312.97	28.78	0.11	**-0.08**	60.13	8.34	48.47
80	19193.86	28.71	0.83	0.70	100.69	28.06	80.44
85	18316.54	35.93	0.43	0.46	117.25	91.20	4.48
90	17514.42	36.34	0.54	0.48	111.66	62.55	29.00
95	16786.39	37.42	0.19	0.10	28.97	65.34	10.41
100	16083.54	41.40	0.21	0.16	110.88	115.83	79.64
105	15436.40	38.86	0.43	0.30	2.92	103.64	76.83
110	14826.58	39.27	0.70	0.60	26.53	36.31	71.28
115	14381.06	37.21	0.09	0.65	28.48	100.20	35.89
120	13887.74	39.66	1.01	1.01	69.83	43.53	67.13
125	13568.40	38.90	**-0.06**	0.23	20.11	64.88	85.77
130	13127.54	43.86	0.57	0.65	17.11	42.88	81.53
135	12812.87	34.84	0.21	0.28	33.98	56.00	79.73
140	12396.74	39.90	0.88	0.80	37.66	93.23	94.97
145	12132.16	44.04	0.25	0.17	105.94	110.42	75.95
150	11668.53	38.98	1.02	1.19	56.64	59.72	26.92
	Average (%)	22.82	0.22	0.21	54.82	54.63	43.32

Table 3: Comparison of MALT, BLS and RLS ($n = 654$)

		Deviation (%)			*CPU Times (sec.)*		
p	Best	MALT	BLS	RLS	MALT	BLS	RLS
5	1851877.25	0.00	0.00	0.00	10.72	12.55	9.77
10	1249564.75	0.00	0.03	0.02	23.38	26.50	113.30
15	980131.69	0.01	0.01	0.03	152.98	83.70	116.56
20	828685.63	0.03	0.03	0.01	112.23	17.81	69.55
25	721988.19	0.03	0.01	0.01	190.58	135.78	224.97
30	638212.31	0.40	0.01	0.01	299.55	62.05	88.81
35	577496.63	0.91	0.07	0.02	186.13	215.70	78.00
40	529660.12	1.56	0.16	0.18	192.11	236.80	201.44
45	489483.75	1.75	0.19	0.15	240.08	66.59	188.69
50	453109.56	2.23	0.20	0.14	101.88	89.80	70.83
55	422638.69	2.90	0.20	0.10	126.33	128.48	243.45
60	397674.53	3.59	0.07	0.11	60.13	186.09	283.75
65	376630.28	2.87	0.08	0.05	281.14	166.78	71.67
70	357335.13	3.35	0.17	0.12	90.61	99.05	65.05
75	340123.50	4.91	0.09	0.09	121.59	49.98	33.84
80	325971.25	5.32	0.08	0.15	18.94	129.47	108.81
85	313446.59	4.35	0.21	0.16	131.77	70.53	151.56
90	302479.03	5.23	0.23	0.29	244.47	148.78	229.97
95	292282.63	5.73	0.35	0.20	205.17	58.44	214.39
100	282536.44	5.62	0.36	0.51	66.20	302.13	97.88
105	273463.31	4.30	0.28	0.27	260.83	244.48	106.53
110	264959.97	6.14	0.29	0.48	60.69	108.59	169.16
115	256763.02	6.77	0.48	0.32	205.91	191.30	156.23
120	249050.48	7.66	0.43	0.38	37.67	58.02	142.84
125	241880.38	7.73	0.47	0.36	44.75	186.22	84.36
130	235203.39	8.45	0.46	0.42	222.25	239.83	22.41
135	228999.80	8.18	0.43	0.35	99.88	194.00	43.50
140	223062.63	7.40	0.28	0.47	37.11	110.48	67.08
145	217462.80	8.68	0.49	0.36	107.66	68.58	23.72
150	212236.27	9.04	0.37	0.50	80.98	208.47	120.33
	Average (%)	4.17	0.22	0.21	133.79	129.90	119.95

Table 4: Comparison of MALT, BLS and RLS ($n = 1060$)

three steps: Shaking or perturbation of the incumbent solution X to a random solution in some neighbourhood of X, a local search from the perturbed solution, and a neighbourhood change. For various VNS variants and their successful applications, see Hansen *et* al. [26] and Mladenovic *et* al. [37]. For a number of different variants of VNS applied to the multi-source Weber problem, see Brimberg *et* al. [10].

The *Basic VNS* (BVNS) method we employ is outlined in Algorithm 3, where two parameters are needed: t_{max} relating to the limit on execution time and k_{max} to the number of neighbourhoods used in the shaking operation.

Algorithm 3: Steps of the Basic VNS (BVNS)

Step 1: Specify k_{max}, t_{max} and define the neighbourhood structures $\mathcal{N}_k; k = 1, \ldots, k_{max}$; set $t = 0$, and obtain an initial solution X.

Step 2: While $t < t_{max}$ do

Step 2a: Set $k = 1$

Step 2b: Repeat until $k = k_{max}$

(i) Generate a random $X' \in \mathcal{N}_k(X)$ *(Shaking)*

(ii) Apply a local search from X' to determine X'' *(Local search)*

(iii) If the solution is improved, move there ($X \leftarrow X''$) and go to Step 2a, else set $k = k+1$. *(Change neighbourhood)*

Step 2c: Set $t =$ `CpuTime()`

The distance function we use to define the different neighbourhood structures counts the number of facilities that are not in the same locations as in the incumbent X(e.g., Brimberg *et* al. [10]). Thus, a shake to neighbourhood $\mathcal{N}_k$ involves randomly re-locating k facilities to unoccupied fixed points.

4.1. Computational Results

In the reformulation local search (RLS), we simply add new Weber points to the set of potential sites for the discrete problem each time the continuous phase reaches a local minimum. Also, after 10 unsuccessful big iterations (full cycles of the VNS), we start again from the incumbent solution while

deleting all added Weber points; that is, we return to the original set $S = \{A_j, j = 1, \ldots, n\}$. In this way the discrete problem is kept at a manageable size.

Two algorithms are compared. In the first, termed VNS1, we implement the successful variant of VNS given in Brimberg *et* al. [10] which relocates facilities to unoccupied fixed points in the shaking operation as discussed above (this refers to VNS3 in that paper). The second method, VNS2, uses our new RLS as the local search. Note that the discrete phase of RLS uses a Teitz and Bart vertex swap [44] with Whitaker's accelerated implementation [48] starting from the solution just obtained in the continuous phase.

Both versions of VNS are tested on the same 4 data sets described in the previous section. For both implementations of VNS, the following parameter values are used: $t_{max} = 300$ seconds, $k_{max} = p$, and as noted above, we set the maximum number of unsuccessful full cycles to 10 for re-initializing S.

As the best-known results for the small data set ($n = 50$) are all optimal and our RLS based approach found all those solutions, we do not reproduce them here. The results for the other data sets ($n = 287, 654$ and 1060) are summarized in Tables 5 to 7. We also highlight the best percentage deviation in bold when it shows an improvement on the best known solution from Brimberg *et* al. [10]. On average VNS2 outperforms VNS1 slightly. For the case of $n = 287$, VNS2 proved to be slightly better than VNS1 when compared against the optimal solutions. Average deviations of 0.025% and 0.011% with the worst deviations of 0.24% and 0.05% are recorded for VNS1 and VNS2, respectively. For the 654-customer instances, 22 and 21 best solutions are found including 4 new best solutions for VNS1 and VNS2 respectively. Their respective average deviations are 0.024% and 0.028% above the best known, while the largest improvements are -0.27% and -0.22%, and worst deviations are 0.38% and 0.35% respectively. For the largest instances ($n = 1060$) 28 and 29 best solutions are identified including 20 and 24 new ones for VNS1 and VNS2 respectively. Besides this, their average deviation shows improvement of -0.093% and -0.107% and the worst deviation is found to be negligible at 0.04% and 0.02% respectively. Again we see that VNS2 performs slightly better than VNS1. The average computing times when the best solution is found are similar for $n = 287$ for both approaches but VNS2 is faster as n increases requiring an approximate CPU time of 2/3 of that of VNS1 in the largest data set ($n = 1060$).

	Objective function values			*Deviation (%)*		*CPU Times (sec.)*	
p	Best*	VNS1	VNS2	VNS1	VNS2	VNS1	VNS2
2	14427.59	14427.97	14427.97	0.00	0.00	0.02	0.02
3	12095.44	12095.44	12095.44	0.00	0.00	0.06	0.75
4	10661.48	10661.91	10661.91	0.00	0.00	2.15	0.08
5	9715.63	9716.22	9716.22	0.01	0.01	0.05	0.03
6	8787.56	8789.08	8789.10	0.02	0.02	0.02	0.20
7	8160.32	8162.51	8161.97	0.03	0.02	0.67	7.57
8	7564.29	7564.67	7564.67	0.00	0.00	5.80	1.29
9	7088.13	7088.48	7088.28	0.00	0.00	4.02	5.66
10	6705.04	6705.36	6705.16	0.00	0.00	0.14	0.58
11	6351.59	6351.60	6351.59	0.00	0.00	4.21	7.74
12	6033.05	6047.31	6033.06	0.24	0.00	0.92	7.61
13	5725.19	5725.23	5725.21	0.00	0.00	0.08	4.45
14	5469.65	5474.97	5469.94	0.10	0.01	3.03	5.88
15	5224.70	5225.00	5225.00	0.01	0.01	9.09	0.75
20	4148.84	4149.94	4149.08	0.03	0.01	31.86	7.35
25	3348.71	3348.81	3348.79	0.00	0.00	43.48	28.31
30	2716.90	2717.61	2717.60	0.03	0.03	1.37	0.44
35	2238.18	2238.39	2238.21	0.01	0.00	1.93	40.37
40	1900.84	1900.99	1900.92	0.01	0.00	7.97	3.76
45	1630.31	1630.42	1630.42	0.01	0.01	32.40	4.09
50	1402.58	1402.64	1402.61	0.00	0.00	50.26	18.19
55	1203.99	1204.03	1203.99	0.00	0.00	1.50	28.56
60	1055.14	1055.24	1055.24	0.01	0.01	18.77	6.32
65	924.56	924.67	924.67	0.01	0.01	29.61	8.75
70	814.22	814.43	814.43	0.03	0.03	31.84	4.46
75	730.04	730.38	730.35	0.05	0.04	46.89	15.21
80	655.38	655.75	655.64	0.06	0.04	1.15	4.80
85	588.37	588.63	588.65	0.04	0.05	47.30	45.32
90	529.21	529.31	529.28	0.02	0.01	2.12	55.10
95	480.86	480.94	480.93	0.02	0.01	41.15	22.79
100	441.24	441.27	441.26	0.01	0.00	44.71	55.60
	Average			0.025	0.011	14.98	12.65
	# best*			11	15		

Table 5: Comparison of VNS1 and VNS2 ($n = 287$)

* These are optimal solutions as shown by Krau [30].

	Objective function values			*Deviation (%)*		*CPU Times (sec.)*	
p	Old best	VNS1	VNS2	VNS1	VNS2	VNS1	VNS2
2	815313.31	815313.31	815313.31	0.00	0.00	0.00	0.00
3	551062.88	551062.88	551062.88	0.00	0.00	0.25	0.00
4	288191.00	288191.00	288191.00	0.00	0.00	0.05	0.06
5	209068.80	209068.80	209068.80	0.00	0.00	0.00	0.00
6	180488.22	180488.22	180488.22	0.00	0.00	0.00	0.00
7	163704.17	163704.17	163704.17	0.00	0.00	0.14	0.47
8	147050.80	147050.80	147050.80	0.00	0.00	0.12	0.19
9	130936.13	130936.13	130936.13	0.00	0.00	0.16	0.27
10	115339.03	115339.03	115339.03	0.00	0.00	0.23	0.75
11	100133.20	100133.20	100133.20	0.00	0.00	0.00	0.00
12	94152.05	94152.05	94152.05	0.00	0.00	0.27	0.19
13	89376.81	89454.76	89454.76	0.09	0.09	1.84	0.80
14	84807.67	84807.67	84815.88	0.00	0.01	1.00	2.31
15	80177.04	80177.04	80177.04	0.00	0.00	0.17	9.05
20	63389.02	63389.03	63389.03	0.00	0.00	28.20	9.78
25	52209.51	52212.83	52212.83	0.01	0.01	19.23	8.81
30	44705.19	44708.52	44708.52	0.01	0.01	0.31	14.73
35	39257.27	39260.60	39260.60	0.01	0.01	11.95	156.38
40	35704.41	35707.74	35710.88	0.01	0.02	176.75	12.11
45	32306.97	32307.02	32307.02	0.00	0.00	36.03	65.30
50	29338.01	29338.06	29338.06	0.00	0.00	17.33	6.80
55	26699.17	26699.17	26699.17	0.00	0.00	13.81	12.11
60	24504.39	24504.45	24504.45	0.00	0.00	20.23	18.86
65	22747.10	22733.40	22733.40	**-0.06**	**-0.06**	6.56	72.92
70	21465.44	21473.60	21471.36	0.04	0.03	164.06	264.88
75	20312.97	20298.00	20285.09	**-0.07**	**-0.14**	180.58	75.83
80	19193.86	19222.92	19233.93	0.15	0.21	114.28	34.27
85	18316.54	18343.50	18372.16	0.15	0.30	89.86	98.91
90	17514.42	17566.15	17575.62	0.30	0.35	239.53	157.63
95	16786.39	16790.21	16814.01	0.02	0.16	41.48	203.50
100	16083.54	16084.40	16087.76	0.01	0.03	254.25	102.50
105	15436.40	15416.11	15422.99	**-0.13**	**-0.09**	239.80	260.83
110	14826.58	14854.90	14828.10	0.19	0.01	117.47	47.41
115	14381.06	14342.77	14349.74	**-0.27**	**-0.22**	228.56	219.09
120	13887.74	13940.72	13922.82	0.38	0.25	145.78	131.09
	Average			0.024	0.028	61.44	56.79
	# best			22	21		
	# new best			4	4		

Table 6: Comparison of VNS1 and VNS2 ($n = 654$)

	Objective function values			*Deviation (%)*		*CPU Times (sec.)*	
p	Old best	VNS1	VNS2	VNS1	VNS2	VNS1	VNS2
5	1851879.88	1851877.25	1851879.88	0.00	0.00	141.32	29.19
10	1249564.75	1249564.75	1249564.75	0.00	0.00	4.82	17.27
15	980132.13	980151.81	980131.69	0.00	0.00	89.00	10.19
20	828802.00	828697.13	828685.69	**-0.01**	**-0.01**	156.87	69.92
25	722061.19	722029.50	722001.44	0.00	**-0.01**	211.42	147.86
30	638263.00	638234.56	638218.13	0.00	**-0.01**	290.20	256.81
35	577526.63	577510.75	577496.63	0.00	**-0.01**	154.55	192.27
40	529866.19	529733.31	529696.94	**-0.03**	**-0.03**	123.30	121.81
45	489650.00	489584.16	489527.91	**-0.01**	**-0.02**	296.30	252.19
50	453164.00	453167.63	453149.47	0.00	0.00	84.77	18.61
55	422770.00	422787.63	422728.75	0.00	**-0.01**	248.40	18.13
60	397784.41	397722.53	397704.09	**-0.02**	**-0.02**	187.88	14.94
65	376759.50	376731.88	376700.44	**-0.01**	**-0.02**	198.58	32.31
70	357385.00	357360.25	357375.28	**-0.01**	0.00	176.29	97.63
75	340242.00	340177.72	340215.66	**-0.02**	**-0.01**	203.20	292.48
80	326053.19	326105.72	326107.78	0.02	0.02	293.27	185.97
85	313738.19	313851.50	313687.50	0.04	**-0.02**	223.21	54.29
90	302837.00	302743.75	302786.09	**-0.03**	**-0.02**	98.92	276.60
95	292875.09	292751.81	292454.81	**-0.04**	**-0.14**	139.21	228.69
100	283113.00	282769.22	282778.19	**-0.12**	**-0.12**	186.52	163.52
105	274576.00	273758.63	273811.84	**-0.30**	**-0.28**	205.76	60.00
110	265801.00	265316.47	265261.00	**-0.18**	**-0.20**	297.66	31.14
115	257605.00	257199.06	257098.28	**-0.16**	**-0.20**	129.79	136.64
120	249584.00	249345.88	249386.44	**-0.10**	**-0.08**	172.19	70.61
125	242930.00	242241.38	242205.61	**-0.28**	**-0.30**	274.28	194.29
130	236154.00	235598.81	235505.97	**-0.24**	**-0.27**	268.37	194.00
135	230431.00	229428.47	229307.31	**-0.44**	**-0.49**	264.50	148.61
140	224504.00	223412.30	223416.47	**-0.49**	**-0.48**	191.08	88.78
145	218279.00	217762.50	217657.17	**-0.24**	**-0.28**	215.79	222.33
150	212926.00	212643.89	212519.81	**-0.13**	**-0.19**	162.25	189.94
	Average			-0.093	-0.107	189.66	127.23
	# best			28	29		
	# new best			20	24		

Table 7: Comparison of VNS1 and VNS2 ($n = 1060$)

5. Discussion and conclusion

A new local search for continuous location problems that systematically oscillates between the continuous space and a discretised space is proposed. In each iteration new potential sites obtained in the continuous phase are added to the discrete space in order to improve the quality of the discrete approximation. We implemented this idea on the well known multi-source Weber problem with encouraging results. In brief, once the new Weber locations are found at a local minimum in the continuous space, these are added to the set of potential sites in the discrete space. If an improved solution is found in the discrete space, it then serves as an initial solution in the continuous space and a new iteration begins. This process is repeated until there is no change in the solution in either space. We also incorporated this local search within VNS and this proved to be very competitive when tested on the classical data sets from the literature.

The proposed methodology can be adapted and applied to many other location models. For example, consider the continuous p-center problem and the relation to its well-known discrete vertex p-center problem, or the simple plant location problem on the plane and its relation to its discrete counterpart. Another variant related to discrete location problems includes the capacitated case on the plane. Indeed, the idea of exploring the relation between discrete and continuous location problems presents in our view a rich new area of research.

Acknowledgement The authors would like to thank two anonymous referees for their interesting suggestions that improved the content as well as the presentation of the paper.

References

[1] M.H. Akyüz, I.K. Altinel, and T. Öncan. Location and allocation based branch and bound algorithms for the capacitated multi-facility Weber problem . *Annals of Operations Research*, doi: 10.1007/s10479-012-1221-3, 2013.

[2] N. Aras, I.K. Altinel, and M. Orbay. New heuristic methods for the capacitated multi-facility Weber problem . *Naval Research Logistics*, 54(1):21–32, 2007.

[3] N. Aras, M. Orbay, and I.K. Altinel. Efficient heuristics for the rectilinear distance capacitated multi-facility Weber problem . *Journal of the Operational Research Society*, 59(1):64–79, 2008.

[4] B.Boyaci, I.K. Altinel, and N. Aras. Approximate solutions methods for the capacitated multi-facility Weber problem. *IIE Transactions*, 43(1):97–120, 2013.

[5] I. Bongartz, P.H. Calamai, and A.R. Conn. A projection method for lp norm location-allocation problems. *Mathematical Programming*, 66:283–312, 1994.

[6] J. Brimberg and Z. Drezner. A new heuristic for solving the p-median problem in the plane. *Computers and Operations Research*, 40:427–437, 2013.

[7] J. Brimberg, Z. Drezner, N. Mladenović, and S. Salhi. Generating good starting solutions for the p-median problem in the plane. *Electronic Notes in Discrete Mathematics*, 39:225–232, 2013.

[8] J. Brimberg, P. Hansen, and N. Mladenović. Attraction probabilities in variable neighbourhood search. *4OR- Quarterly Journal of Operations Research*, 8:181–194, 2010.

[9] J. Brimberg, P. Hansen, N. Mladenovic, and S. Salhi. A survey of solution methods for the continuous location-allocation problem. *International Journal of Operations Research*, 5:1–12, 2008.

[10] J. Brimberg, P. Hansen, N. Mladenović, and E.D. Taillard. Improvement and comparison of heuristics for solving the uncapacitated multisource Weber problem. *Operations Research*, 48(3):444–460, 2000.

[11] J. Brimberg and J. H.Walker. Estimation of travel distance. In N. Balakrishnan, editor, *Methods and Applications of Statistics in Business, Finance and Management Science*, pages 139–157. John Wiley and Sons, 2010.

[12] J. Brimberg, N. Mladenovic, and S. Salhi. The multi-source Weber problem with constant opening cost. *Journal of the Operational Research Society*, 55:640–646, 2004.

[13] J. Brimberg and S. Salhi. A continuous location-allocation problem with zone-dependent fixed cost. *Annals of Operations Research*, 136:99–115, 2005.

[14] L. Cooper. Location-allocation problems. *Operations Research*, 11:331–343, 1963.

[15] L. Cooper. Heuristic methods for location-allocation problems. *SIAM Review*, 6:37–53, 1964.

[16] Z. Drezner, J. Brimberg, N. Mladenović, and S. Salhi. Effective solutions to the p-median problem in the plane, 2013. in preparation.

[17] O. du Merle, D. Villeneuve, J. Desrosiers, and P. Hansen. Stabilization dans le cadre de la génération de colonnes. *Les cahiers du GERAD G-97-08, University of Montreal*, 1997.

[18] E. Durmaz, N. Aras, and I.K. Altinel. Discrete approximation heuristics for the capacitated continuous location-allocation problem with probabilistic customer locations. *Computers and Operations Research*, 36(7):2139–2148, 2009.

[19] S. Eilon, C.D.T. Watson-Gandy, and N. Christofides. *Distribution Management*. Hafner, New York, 1971.

[20] E. Erkut and S. Newman. Analytical models for locating undesirable facilities. *European Journal of Operational Research*, 40:275–291, 1989.

[21] D. Erlenkotter. A dual-based procedure for uncapacitated facility location. *Operations Research*, 26:992–1009, 1978.

[22] R.L. Francis, L.F. McGinnis, and J.A. White. *Facility layout and location: an analytical approach*. Prentice Hall, Englewood Cliffs, 1992.

[23] M. D. H. Gamal and S. Salhi. Constructive heuristics for the uncapacitated continuous location-allocation problem. *Journal of the Operational Research Society*, 52:821–829, 2001.

[24] M. D. H. Gamal and S. Salhi. A cellular heuristic for the multisource Weber problem. *Computers and Operations Research*, 30:1603–1624, 2003.

[25] P. Hansen, S Krau, and O. du Merle. Stabilized column generation algorithm for the multisource Weber problem. *Les cahiers du GERAD G-2000-62, University of Montreal*, 2000.

[26] P. Hansen, N. Mladenović, and J. A. Moreno Perez. Variable neighborhood search: methods and applications. *Annals of OR*, 175:367–407, 2010.

[27] P. Hansen, N. Mladenović, and E. Taillard. Heuristic solution of the multisource Weber problem as a p-median problem. *Operations Research Letters*, 22:55–62, 1998.

[28] P. Hansen, J. Perreur, and J.-F. Thisse. Location theory, dominance, and convexity: some further results. *Operations Research*, 28:1241–1250, 1980.

[29] L.M. Ostresh Jr. *Multi-exact solutions to the M-center location-allocation problem.* In G.Rushton, M.F.Goodchild and L.M.Ostresh Jr (Eds), Computer Programs for Location-Allocation Problems, Monograph No.6, University of Iowa, IA, 1973.

[30] S. Krau. *Extensions du problème de Weber.* PhD Thesis, Ecole Polytechnique de Montreal, Montreal, 1997.

[31] R.E. Kuenne and R.M. Soland. Exact and approximate solutions to the multisource Weber problem. *Mathematical Programming*, 3:193–209, 1972.

[32] R. F. Love, J. G. Morris, and G. O. Wesolowsky. *Facilities Location: Models and Methods.* North-Holland, New York, 1988.

[33] M. Megiddo and K.J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13:182–196, 1984.

[34] N. Mladenović, J. Brimberg, P. Hansen, and J.A. Moreno-Perez. The p-median problem: A survey of metaheuristic approaches. *European Journal of Operational Research*, 179:927–939, 2007.

[35] N. Mladenović and P. Hansen. Variable Neighbourhood Search. *Computers and Operations Research*, 24:1097–1100, 1997.

[36] N. Mladenović, F. Plastria, and D. Urosevic. Reformulation descent applied to a circle packing problem. *Computers and Operations Research*, 32:2419–2434, 2005.

[37] N. Mladenović, R. Todosijevic, and D. Urosevic. An efficient general variable neighborhood search for large TSP problems with time windows. *Yugoslav Journal of Operations Research*, 23:19–31, 2013.

[38] S. Nickel and J. Puerto. *Location Theory: A unified approach*. Springer, Heidelberg, 2005.

[39] G. Reinelt. TSLIB-a travelling salesman library. *ORSA J Computing*, 3:376–384, 1991.

[40] K. E. Rosing. An optimal method for solving the (generalized) multi-Weber problem. *European Journal of Operational Research*, 58:414–426, 1992.

[41] S. Salhi and M. Sari. A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, 103:95–112, 1997.

[42] A.J. Scott. *Combinatorial programming, spatial analysis and planning*. Methuen, London, 1971.

[43] E. Taillard. Heuristic methods for large centroid clustering problems. *Journal of Heuristics*, 9:51–73, 2003.

[44] M. B. Teitz and P. Bart. Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations Research*, 16:955–961, 1968.

[45] J. E. Ward and R. E. Wendell. A new norm for measuring distance which yields linear location problems . *Operations Research*, 28:836–844, 1980.

[46] J. E. Ward, R. E. Wendell, and E. Richard. Using block norms for location modeling . *Operations Research*, 33:1074–1090, 1985.

[47] E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal*, 43:355–386, 1937.

[48] R. Whitaker. A fast algorithm for the greedy-interchange for large-scale clustering and median location problems. *INFOR*, 21:95–108, 1983.