An Adaptive Multiphase Approach for Large Unconditional and Conditional p-Median Problems

Chandra Ade Irawan • Said Salhi • Maria Paola Scaparra

Centre for Logistics & Heuristic Optimization (CLHO), Kent Business School,

University of Kent, Canterbury, Kent CT2 7PE, UK

Abstract A multiphase approach that incorporates demand points aggregation, Variable Neighbourhood Search (VNS) and an exact method is proposed for the solution of large-scale unconditional and conditional *p*-median problems. The method consists of four phases. In the first phase several aggregated problems are solved with a "Local Search with Shaking" procedure to generate promising facility sites which are then used to solve a reduced problem in Phase 2 using VNS or an exact method. The new solution is then fed into an iterative learning process which tackles the aggregated problem (Phase 3). Phase 4 is a post optimisation phase applied to the original (disaggregated) problem. For the *p*-median problem, the method is tested on three types of datasets which consist of up to 89,600 demand points. The first two datasets are the BIRCH and the TSP datasets whereas the third is our newly geometrically constructed dataset that has guaranteed optimal solutions. The computational experiments show that the proposed approach produces very competitive results. The proposed approach is also adapted to cater for the conditional *p*-median problem with interesting results.

Keywords Variable neighbourhood search, exact method, aggregation, large p-median problems, adaptive learning

1. Introduction

The *p*-median problem is a discrete location problem where the objective is to find the location of *p* facilities among *n* discrete potential sites in such a way to minimise the sum of the weighted distances between customers and their nearest facilities. The *p*-median problem becomes the conditional problem when some (say *q*) facilities already exist in the study area and the aim is to locate *p* new facilities given the existing *q* facilities. This problem is also known as the (*p, q*) median problem. A customer can be served by one of the existing or the new open facilities whichever is the closest to the customer. When $q = 0$, the problem

reduces to the unconditional problem (the *p*-median problem for short). A further but brief description related to the conditional *p*-median problem will be presented in Section 6 where some results are also given.

The *p*-median problem is categorized as NP-hard (Kariv and Hakimi, 1969). For relatively large problems, optimal solutions may not be found and hence heuristic or metaheuristic methods are usually considered to be the best way forward for solving such problems. Mladenovic *et al*. (2007) provided an excellent review on the *p*-median problem focusing on metaheuristic methods. The *p*-median problem was originally formulated by ReVelle and Swain (1970). However, Rosing *et al*. (1979) enhanced the *p*-median problem formulation to reduce its solution time. In their model, the furthest *p*-1 assignments associated with each demand point are ignored. This reduction scheme is based on the observation that in the worst case, a demand point *i* is served by its $(n\text{-}p+1)^{th}$ closest site. The enhanced *p*-median formulation is formulated as follows:

Minimise
$$\sum_{i\in I}\sum_{j\in F_i} w_i d(i,j) Y_{ij} \qquad (1)$$

Subject to
$$\sum_{j\in F_i} Y_{ij} = 1 \qquad \forall i \in I \qquad (2)$$
$$\sum_{j\in J} X_j = p \qquad (3)$$
$$Y_{ij} - X_j \le 0, \qquad \forall i, j \in F_i \qquad (4)$$
$$X_j \in \{0,1\} \qquad \forall j, j \in J \qquad (5)$$
$$Y_{ij} \in \{0,1\} \qquad \forall i, j \in F_i \qquad (6)$$

Where

(I,J): set of customers $(i \in I = \{1,\ldots,n\})$ and set of potential sites $(j \in J = \{1,\ldots,M\})$ (i.e. : $n = |I|$ and $M = |J|$) respectively

w_i : demand or weight of customer *i*;

$d(i,j)$: distance between customer *i* and potential site *j* (Euclidian distance is used here);

p : the required number of facilities to locate;

Y_{ij} = 1, if customer *i* is fully served by a facility at site *j* and = 0 otherwise;

X_j = 1, if a facility is opened at potential site *j* and = 0 otherwise;

F_i : set of all sites except the $p-1$ furthest sites from demand point *i*.

The objective function (1) minimises the total demand-weighted distance. Constraints (2) guarantee that each customer *i* is assigned to one facility only. Constraint (3) states that the

number of facilities to be located is p. Constraints (4) ensure that customer i can only be allocated to facility j (i.e., $Y_{ij} = 1$) if a facility is opened at site j (i.e., $X_j = 1$). The use of the sets F_i in constraints (2), (4) and (6) yields a more compact formulation, requiring a fewer number of variables and constraints than the classical formulation.

In some applications, p-median problems may involve a large number of demand points and potential facility sites. These problems arise, for example, in urban or regional areas where the demand points are individual private residences. Francis *et al*. (2009) stated that it may be impossible and time consuming to solve location problems consisting of a large number of demand points. To simplify the problem, it is quite common to aggregate demand points (and/or potential facility sites) when solving large-scale location problems. In other words, the number of demand points (and/or potential facility sites) can be reduced from n to m points ($m << n$) so that the approximated problem can be solved within a reasonable amount of computing time. However, aggregation introduces errors in the data as well as in the models output, thus resulting in less accurate results.

The main contributions of this paper include: (i) a novel multiphase approach that incorporates aggregation, Variable Neighbourhood Search (VNS) and an exact method for solving large p-median problems, (ii) new best solutions for some benchmark problems, (iii) the construction of a new large dataset for p-median problems with guaranteed optimality, and (iv) an adaptation of the proposed approach for the conditional p-median problem.

The paper is organized as follows. Section 2 presents a brief review of the past efforts at solving large p-median problems. Section 3 describes the ingredients that make up our method as well as the overall algorithm. Detailed explanations of the main steps and the "Local Search with Shaking" procedure are described in Section 4. Computational results are presented in Section 5 using large datasets including the one with guaranteed optimal solutions which we constructed. Section 6 presents a brief review on the conditional p-median problem followed by the adaptation and the implementation of our approach for this related problem. The last section provides a summary of our findings and highlights some avenues for future research.

2. Past efforts at solving large p-median problems

This section presents an overview of past efforts at solving large p-median problems (see Francis *et al*., 2009, for an excellent review). Hillsman and Rhoda (1978) introduced a

classification of aggregation errors using three types, namely source A, B, and C errors. Source A error occurs when the distance between an Aggregate Spatial Unit (ASU) and a facility is utilized in the model, instead of the true distance between a Basic Spatial Unit (BSU) and a facility. Source B error exists in the special case when a facility is located at an ASU whereas source C error appears when a BSU is assigned to the wrong facility.

Goodchild (1979) stated that aggregation tends to produce more dramatic effects on location than on the values of the objective function while also noting that there is no aggregation scheme without a possible resulting error. Bach (1981) mentioned that "the level of aggregation exerts a strong influence on the optimal locational patterns as well as on the values of the locational criteria". Mirchandani and Reilly (1986) examined the effect of replacing distances to demand points (BSUs) in a region by the distance to a single point (ASU) representing that region.

Current and Schilling (1987) proposed a method for eliminating source A and source B errors. They introduced a novel way of measuring aggregated weighted travel distances for *p*-median problems. Let $d(i, j)$ denote the distance between the i^{th} and the j^{th} BSUs and $\tilde{d}(k, j)$ the distance between the representative point of the k^{th} ASU and the j^{th} BSU. The distance between the k^{th} ASU and the j^{th} facility is traditionally defined as:

$$\hat{d}(k, j) = W_k \tilde{d}(k, j) \tag{7}$$

where $W_k = \sum_{i \in A_k} w_i$ with A_k being the set of aggregated BSUs at the k^{th} ASU.

To eliminate source A and B errors, the distance proposed in Current and Schilling (1987) is set as:

$$\hat{d}(k, j) = \sum_{i \in A_k} w_i d(i, j) \tag{8}$$

However, this method is not able to eliminate source C errors.

Casillas (1987) introduced two measures to assess the accuracy of aggregated models. These include the cost error $(ce = f(F':C) - f(F':C'))$ and the optimality error $(oe = f(F:C) - f(F':C))$ where F and F' represent the optimal locations of the p facilities found with the original and the aggregated models respectively, while C and C' denote the list of BSUs and ASUs. The objective functions $f(F:C)$, $f(F':C)$ and $f(F':C')$ represent the objective function evaluated using F and C, F' and C, and F' and C' respectively.

Oshawa *et al* (1991) studied the location error and the cost error due to "rounding" in the unweighted 1-median and 1-centre problems in the one-dimensional continuous space.

Aggregation error bounds for the median and the centre problems were developed by Francis and Lowe (1992). A Geographical Information System (GIS) method for eliminating source C error was proposed by Hodgson and Neuman (1993). Transport costing errors for the median problems were investigated by Ballou (1994) who demonstrated that cost errors increase with p but decrease with m. An investigation by Fotheringham *et al* (1995) suggested that the level of aggregation affects the location error more significantly than the objective function value. Francis *et al*. (1996) introduced a median row-column aggregation method to find an aggregation which gives a small error bound. In addition to the A, B, and C errors, Hodgson *et al*. (1997) introduced source D error which arises when the BSU locations act as potential sites.

Murray and Gottsegen (1997) investigated the influence of data aggregation on the stability of facility locations and the objective function for the planar *p*-median model. Demand point aggregation procedures for the *p*-median and the *p*-centre network location models were studied by Andersson *et al*. (1998). Hodgson and Salhi (1998) proposed a quadtree-based technique to eliminate source A, B, and C errors in the allocation process. Bowerman *et al*. (1999) investigated the demand partitioning method for reducing source A, B, and C aggregation errors in *p*-median problems. Erkut and Bozkaya (1999) provided a review of aggregation errors for the *p*-median problem. Francis *et al.* (2000) computed error bounds for several location models. Plastria (2001) investigated how to minimise aggregation errors when selecting the ASUs location at which to aggregate given groups of BSUs. Hodgson (2002) introduced data surrogation error in the *p*-median problem which appears when an original population's demand is substituted by inappropriate values.

To solve large *p*-median problems without aggregation, Church (2003) put forward an enhanced Mixed Integer Linear Programming formulation called COBRA. He also proved that there are redundant assignment variables that can be consolidated if they satisfy some equivalent assignment conditions. These conditions are based on the order of closeness of facility sites with respect to pairs of demand points. This property leads to a reduction that can be up to 80% of the original number of variables. An enhanced model formulation referred to as Both Exact and Approximate Model Representation (BEAMR) was later proposed by Church (2008). Hansen *et al*. (2009) introduced a primal-dual VNS metaheuristic for large *p*-median clustering problems where a Reduced VNS is used to get good initial solutions which are then fed into a VNS with decomposition. The worst-case analysis of demand point aggregation for the Euclidean *p*-median problem on the plane was

investigated by Qi and Shen (2010). An alternative covering based formulation which has a small subset of constraints and variables is studied by Garcia *et al*. (2010). This method is relatively more efficient when p is large.

Avella *et al*. (2012) designed an aggregation heuristic based on Lagrangean relaxation. They proposed three main procedures, namely a sub-gradient column generation, a core heuristic, and an aggregation heuristic. The first procedure solves the Lagrangean relaxation by combining subgradient optimisation with column generation. The core heuristic is defined by a subset of the most promising variables found according to the Lagrangean reduced costs associated with the open facilities as well as those associated with the allocation variables. An aggregation heuristic is then introduced to tackle the problem when the value of p is relatively small. Very recently, Irawan and Salhi (2013) introduced an approach using demand points aggregation and variable neighbourhood search for solving large-scale p-median problems. Their method used a multi-batch methodology where a learning process that feeds information from one batch to another is utilised. A batch consists of aggregated problems. In this paper, we propose a multiphase approach instead of a multi-batch approach. The first batch of the method by Irawan and Salhi (2013) is similar to our Phase 1 except that a more efficient implementation of the local search is adopted. Subsequent phases are also designed to guide the search in exploring new areas while retaining promising regions. Moreover, we also enhance the method used to solve the aggregated p-median problem; we present an efficient way in aggregating the demand points; and we put forward an effective implementation of the local search that is used to solve the disaggregated (original) problem.

3. An Adaptive Approach (AA) for solving large p-median problems

We propose an adaptive approach which consists of four phases. The main steps of these phases are depicted in Figure 1 but a brief overview is given below. Moreover, a visualisation of our methodology is presented in Appendix A whereas a detailed description of the main steps is given in the next section. In this study, for simplicity we consider potential facility sites as customer sites (i.e. $M=n$).

In the first phase a learning process is conducted. Here, a clustering procedure is used to aggregate n BSUs into m ASUs, with $m << n$. As each customer site acts as a potential facility site, the aggregated problem reduces to having m customers and m potential facility sites. This phase consists of solving a number of aggregated problems of m ASUs using a "Local Search with Shaking" procedure with the aim of choosing p facility locations. This will

be described further in subsection 4.3. Let *L* denote a list of distinct facilities obtained from the solutions of the aggregated problems.

Initialization

Define the parameters *m*, *itermax*, L_{max}, T and set L = Ø.

Phase 1

Do T times the following steps (t = 1, …, T)

(i) Aggregate *n* BSUs into *m* ASUs and construct *m* clusters by allocating all BSUs to their nearest ASUs.

(ii) Calculate the distance $\hat{d}(k, j)$, *k=1,..,m; j=1,…,m*.

(iii) Solve the t^{th} aggregated *p*-median problem using "Local Search with Shaking (*m, m, p*)", let $F_t = (\sigma_1^t, \sigma_2^t, ..., \sigma_p^t)$ be the obtained facility locations with σ_i^t denoting the i^{th} facility at iteration *t* and set $L = L \bigcup F_t$.

Phase 2

(i) **C**onstruct |*L*| clusters around these |*L*| promising facilities, and compute the distance $\hat{d}(k, j)$, *k=1,..,|L|; j=1,…,|L|*.

(ii) If $|L| \le L_{max}$ solve the *p*-median problem with "CPLEX (|*L*|,|*L*|, *p*)", otherwise apply a "VNS (|*L*|,|*L*|, *p*)". Let $\hat{F}'_{best}$ be the obtained facility configuration.

(iii) Solve the *p*-median problem with "Local Search with Shaking (*n*, |*L*|, *p*)" using $\hat{F}'_{best}$ as the initial facility configuration. Let *F'* be the obtained set of open facilities and $f(F':C)$ its corresponding cost.

(iv) Set $\hat{F}'_{best}$ = *F'* and $f_{best} = f(F':C)$. Let E = $\hat{F}'_{best}$ and *iter* = 0.

Phase 3

(i) Aggregate *n* into *m* potential sites by including the facility locations in E.

(ii) Solve the *p*-median problem using "Local Search with Shaking (*n,m,p*)" using $\hat{F}'_{best}$ as the initial solution. Let F' be the obtained facility configuration and $f(F':C)$ its cost.

(iii) If $f(F':C) < f_{best}$ then

Set $\hat{F}'_{best} = F'$, $f_{best} = f(F':C)$ and $E = E \bigcup \hat{F}'_{best}$. Set *iter* = 0 and go to Step (i) of Phase 3.

Else set *iter* = *iter* + 1. If *iter* ≥ *itermax* go to Phase 4, else go to Step (i) of Phase 3.

Phase 4

Solve the *p*-median problem using "Local Search (*n,n,p*)" using $\hat{F}'_{best}$ as the initial solution.

Figure 1. An Adaptive Approach (AA)

In Phase 2, $|L|$ facilities are considered as the 'promising' facilities to set up an aggregated p-median problem which is then solved with a VNS or with CPLEX, depending on the size of the augmented problem. Namely, if $|L|$ is relatively small then the problem is solved by CPLEX ($|L|$, $|L|$, p), otherwise a VNS ($|L|$, $|L|$, p) is adopted where Method (g, s, p) refers to the procedure 'Method' for locating 'p' facilities, using 's' potential sites and serving 'g' customers. When the VNS is applied, the best solution found in Phase 1 is used as the initial solution. To get a feasible solution to the p-median problem with the original customers set, the "Local Search with Shaking" is then used with $|L|$ potential facility sites starting from the solution returned by VNS or CPLEX in the previous step. We refer to this procedure as the "Local Search with Shaking (n, $|L|$, p)". The best solution in this phase is then fed into the next phase (Phase 3).

The third phase is an iterative process that incorporates potential facility sites aggregation and the use of the "Local Search with Shaking". Unlike Phase 1, the aggregation here includes the promising sites found in the previous iteration. This set of promising sites is denoted by E. The resulting aggregated problem with n customers and m potential facility sites is then solved by "Local Search with Shaking (n, m, p)". The obtained solution is then used as an initial solution for the next iteration and the process is repeated until a stopping criterion is met. In our study, the process stops when there is no improvement after a prescribed number of consecutive iterations which we denote by *itermax*.

In the final phase (Phase 4), a post optimisation is carried out. Here, a local search is used to solve the original problem (without aggregation) starting from the best solution obtained in the previous phase. To speed up the search, a reduction scheme, which is described in subsection 4.5, is also incorporated into the search.

4. Description of the main phases of the Adaptive Approach

In this section we present the aggregation scheme and the distance calculation method. These are followed by the description of the "Local Search with Shaking", the VNS, the exact method, and the local search which is used in the original problem.

4.1. The aggregation methodology (Phases 1(i) and 3(i))

This subsection describes the procedure to aggregate n BSUs into m ASUs used in Phases 1(i) and 3(i). The set of the m ASUs includes the followings:

- the promising facility locations obtained from previous iterations in Phase 3 (i.e. the set E). Note that in Phase 1, E = Ø.
- ($m\gamma$) pseudo randomly generated points, where γ is a parameter ($\gamma > 0$).
- (m-|E|-$m\gamma$) randomly generated points.

Firstly, the method includes the promising facility locations (E) as part of the aggregated points. We assume the use of these points may increase the probability of obtaining a good solution. Secondly the *Basic Cell Approach* (BCA), as shown in Figure 2 and briefly described below, is used to generate the subsequent $m\gamma$ aggregated points. All demand points are covered by square cells and the cell information is used for determining m ASUs. This scheme overcomes the weaknesses of a simple random process when dealing with clustered demand points. In addition, it ensures that the generated ASUs are not too close to each other. This is achieved by imposing that the distance between any pair of ASU points is larger than a certain threshold which is based on the side of the cell. Finally, some randomly generated points are added to the set of ASUs to increase the diversity of the solutions.

The BCA method is adapted from the approach given in Irawan and Salhi (2013) which is originally based on the one by Salhi and Gamal (2003) for the multisource Weber problem. An illustration of the BCA is shown in Figure 2. The main steps of the method are formally given in Figure 3.

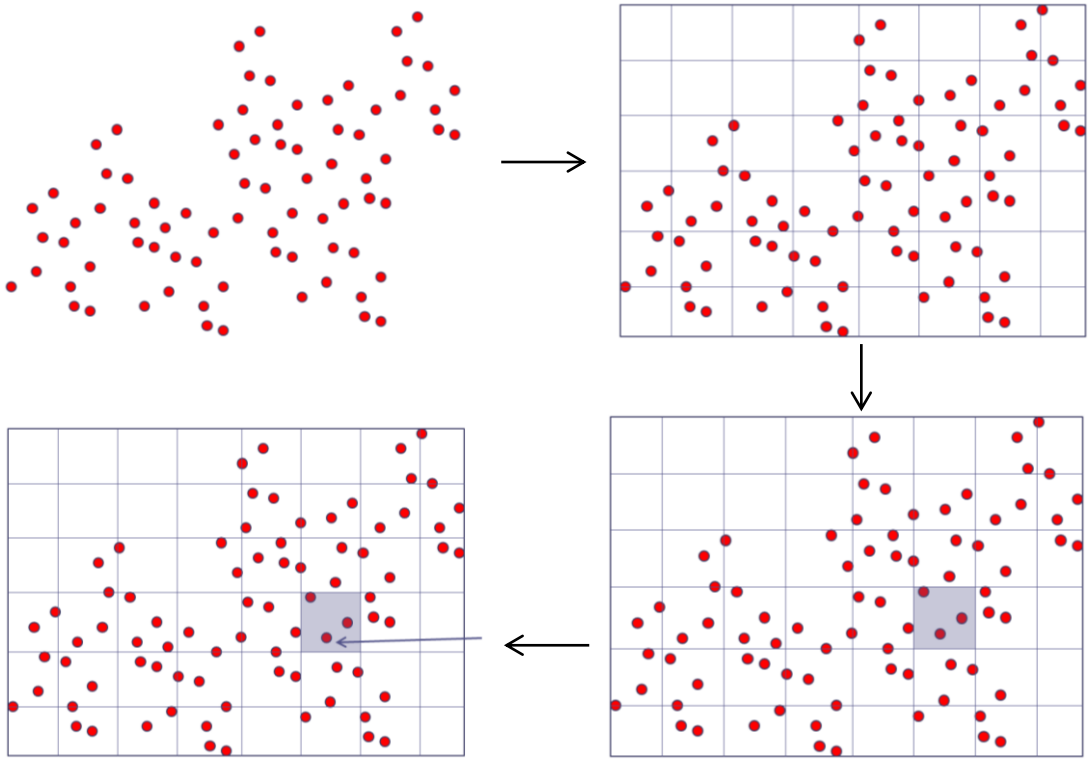


Figure 2. The basic cell approach (BCA)

Step 1 Construct $c_L \cdot c_W$ square cells to get approximately *m* cells of length δ which will cover all demand points. Set parameters ρ, γ, and ς.

Step 2 Record the number of demand points in each cell z as G_z and determine its corresponding probability distribution, say $P_z = G_z / n$, $z = 1,\ldots,(c_L \cdot c_W)$.

Step 3 Define $C' = \mathrm{E}$ as the set of initial promising sites (in Phase 1, E = Ø).

Step 4 Set count = 0.

Step 5 Generate randomly $\beta \in (0,1)$ and choose $\tilde{z}$ *st* $\tilde{z} = F_{(z)}^{-1}(\beta)$ with $F_{(z)} = \sum_{v=1}^{z} P_v$.

Step 6 Choose randomly a demand point *k* in the cell $\tilde{z}$ and calculate $\bar{d}$ (the distance between demand point *k* and the nearest point in *C'*).

Step 7 If $\bar{d} \geq \rho\delta$ set $C' = C' \cup \{k\}$. If $|C'| = m\gamma$ stop the search
Otherwise
Set count = count + 1.
If count = ς then set $\rho = \rho/2$ and go to Step 4 else go to Step 5

Figure 3. The main steps of the BCA

Initially, $(c_L \cdot c_W)$ square cells are constructed. We set the number of cells to be approximately *m*. Let δ denote the length of the side of the cell which is given by:

$$\delta = (x_{\max} - x_{\min}) / \sqrt{m\left(\frac{x_{\max} - x_{\min}}{y_{\max} - y_{\min}}\right)}$$

where x_{max} and x_{min} refer to the maximum and the minimum x coordinate of the points respectively. Similarly, y_{max} and y_{min} refer to the maximum and the minimum y coordinate respectively. A cell is identified by its bottom-left corner. In other words, the coordinates of the bottom-left corner of the z^{th} cell is denoted by (X_z, Y_z), $z = 1,\ldots,(c_L \cdot c_W)$. The bottom-left corner of cell 1 is $(X_1, Y_1) = (x_{\min}, y_{\min})$ and successive cells are defined as follow:

$$(X_z, Y_z) = (x_{\min} + \delta(z \bmod c_L), y_{\min} + \delta(z \bmod c_W))$$

The number of demand points in each cell, G_z, is then recorded and its corresponding probability distribution is calculated as $P_z = G_z / n$, $z = 1,\ldots,(c_L \cdot c_W)$.

Next, a cell is chosen in a pseudo random manner based on the cumulative probability distribution. In other words, we generate randomly $\beta \in (0,1)$ and choose $\tilde{z}$ *st* $\tilde{z} = F_{(z)}^{-1}(\beta)$ with $F_{(z)} = \sum_{v=1}^{z} P_v$. For instance as an illustration, $\tilde{z} = 3$ in Figure 4. A demand point (say point *a*) is then chosen randomly in the cell $\tilde{z}$ as long as it satisfies the threshold distance separation criterion $\bar{d}_a = \underset{j \in C'}{Min}\ d(a, j) \geq \tau = \rho\delta$ with ρ being a parameter whose value is dynamically decreased if no aggregated point is found after a number of attempts being made (say ς, in our

study $\varsigma = m$). The selection of a cell and of a point within the cell is repeated until a prescribed number of ASUs is reached. The centroid of the points in a cell was also attempted but a preliminary study showed that the quality of the solution was found to be slightly inferior.

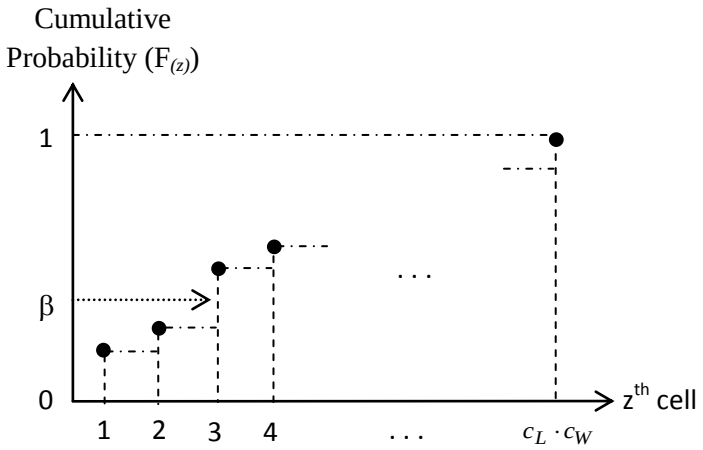


Figure 4. The illustration of determining $\tilde{z}$

4.2. The distance calculation method (Phases 1(ii) and 2(i))

When using the "Local Search with Shaking", the VNS, or CPLEX to solve the aggregated *p*-median problem, the distance matrix between points in *C'* (ASUs) has to be determined first. The way this is performed depends on the type of aggregated *p*-median problems used. The aggregated problems can be categorised as (*m, m, p*), (*n, m, p*), or (|*L*|, |*L*|, *p*) where (*a,b,p*) refers to solving the *p*-median with *a* customers and *b* potential sites.

In Phase 1(iii), the (*m, m, p*) aggregated problem is solved. Here, the procedure to calculate the distance between points in *C'* is performed first by constructing *m* clusters, and allocating all demand points to the nearest points in *C'*. Secondly, the total weight of each cluster W_k, $k = 1,\ldots,m$ is computed. Finally, the approximate distance between each pair of points in *C'*, denoted by $\hat{d}(k, j)$, is calculated using (7).

In Phase 2(ii), the (|*L*|, |*L*|, *p*) aggregated problem is solved instead. In this case, we calculate the distance between each pair of points in *L* using (8) which is practical in this case as the (|*L*|, |*L*|, *p*) aggregated problem is solved only once. In Phases 2(iii) and 3(ii), where no clustering is needed, the true distance between the i^{th} BSU and the j^{th} facility, $d(i, j)$, is used for both the (*n*, |*L*|, *p*) and the (*n, m, p*) aggregated problems.

4.3. The "Local Search with Shaking" for the aggregated p-median problem (Phases 1(iii), 2(iii), and 3(ii))

We use a "Local Search with Shaking" to speed up the search. This choice is due to the fact that our method is an iterative-based approach and therefore finding solutions to the aggregated problems with the VNS or CPLEX would be too time consuming. The method utilises one shaking and one call to the local search only. We explore the following two approaches:

(a) Only the first neighbourhood (N_1) is used. This shaking process can be considered as a perturbation. This is then enhanced by a local search.

(b) Similar to (a), but instead of using the first neighbourhood (N_1), the k^{th} neighbourhood (N_k) is randomly generated where $k \in (1, k_{max})$.

We refer to (a) and (b) as Var1 and Var2 respectively. We carried out some preliminary experiments to test the performance of these two variants. The results, reported in the computational results section, show that Var2 is relatively superior.

The shaking process adapted here applies the shaking algorithm used by Hansen and Mladenovic (1997). Let X denote the facility configuration of the current solution and H the set of potential facility sites. The k^{th} neighbourhood structure N_k is defined as: $N_k(X)$ = use of $N_1(X)$ k times with $k = 1,\ldots,k_{max}$ and $N_1(X) = X - \sigma' \cup \sigma''$ where σ'' is chosen randomly in H-X and $\sigma' \in X$ is selected to yield the best improvement.

The "Local Search with Shaking" is used to solve the (m, m, p), the (n, $|L|$, p), and the (n, m, p) p-median problems. In both Phases 2 and 3, this procedure takes the best solution from the previous steps as the initial solution when solving the (n, $|L|$, p), and the (n, m, p) p-median problems. For the (m, m, p) p-median problem solved in Phase 1, p randomly chosen points are considered instead.

For the (m, m, p) problem, the local search process uses the procedure "FindBestCustomer" proposed by Irawan and Salhi (2013) combined with the use of an efficient data structure initially presented by Resende and Werneck (2007). The latter records intermediate calculations so to eliminate any unnecessary recomputations. A similar data structure was also successfully implemented by Osman and Salhi (1996) when solving the vehicle fleet mix problem. We refer to this local search as "IS-RW". This procedure is based on the *fast interchange heuristic* introduced by Whitaker (1983) which is then adapted to increase the computational speed at the expense of a small loss in quality. The procedure

identifies a point (say point i) among the potential facility sites to be inserted and a facility (say facility j, $j \in X$) that yields the highest saving to be removed. The procedure restricts the search as follow: it considers point $i \in S_j$ with S_j being the set of potential sites that are nearer to facility j than the other open facilities in the current solution ($|S_j| < |H|-p$). For the (n, $|L|$, p) and the (n, m, p) problems, we use the well-known fast swap-based local search procedure of Resende and Werneck (2007). We refer to this local search as "RW". We can afford to use this local search here without the reduction scheme used in "IS-RW" as good solutions are usually obtained from Phase 1 which are then used as initial solutions for the (n, $|L|$, p) and the (n, m, p) problems..

Figure 5 presents the "Local Search with Shaking" when solving the aggregated p-median problems with the use of Var2.

Step 1 Initialization

Set the initial solution (X). Choose p points randomly for Phase 1 whilst for Phases 2 and 3 take the best solution from the previous steps. Define $N_k(X)$, $k = 1,\ldots,k_{max}$ with $k_{max} = p$.

Step 2 Shaking

Generate $k \in (1,k_{max})$ randomly and determine $X' \in N_k(X)$.

Step 3 Local Search

For the (m, m, p) problem: Apply the local search "IS-RW" on X'.
For the (n, $|L|$, p) and the (n, m, p) problems: Apply the local search "RW" on X'.

Figure 5 The "Local Search with Shaking" (Var2) for solving the aggregated problems

We conducted preliminary experiments to test the performance of these two local searches namely IS-RW and RW. The results are reported in the computational results section.

4.4. The VNS and CPLEX (Phase 2(iii))

Variable Neighbourhood Search (VNS) is a metaheuristic first introduced by Brimberg and Mladenovic (1996) for solving continuous location-allocation problems. Hansen and Mladenovic (1997) formally formulated this heuristic and applied it to solve the p-median problem. VNS combines both local search and neighbourhood search. The first search looks for local optimality, while the latter aims to escape from these local optima by systematically using a larger neighbourhood if an improvement is not found and then reverts back to the smaller one otherwise. Initial VNS implementations are given in Hansen and Mladenovic

(2001), but newer variants of VNS and successful applications can be found in Hansen *et al*. (2010).

The VNS or CPLEX is utilized to solve the augmented (|*L*|, |*L*|, *p*) *p*-median problems. The use of these relatively more intensive methods is acceptable as one run of VNS/CPLEX is needed only. Moreover, the size of the *p*-median problem (|*L*|, |*L*|, *p*) is still relatively small.

When the VNS is used, we limit the computing time for solving the problem to T_{vns} seconds. In this study, we set $T_{vns} = 10n^{0.25}p^{0.5}m^{0.5}/1000$ which is found based on a preliminary study. The algorithm of the VNS is based on the one by Hansen and Mladenovic (1997) incorporating the fast swap-based local search (RW). The enhanced formulation (1) – (6) of the *p*-median problem, as given by Rosing *et al.* (1979), is used in the CPLEX implementation.

4.5. The Local Search for the original p-median problem (Phase 4)

An additional post optimisation step to solve the disaggregated problem (original problem) starting from the best solution found in the previous phase is introduced. The main steps are similar to the ones proposed by Irawan and Salhi (2013) except here we adopt a more efficient implementation of the procedure "FindBestCustomer" where we restrict the search even further by imposing that the substituted location must lie within a certain covering radius (*r*). The value of *r* is based on the average of the longest distances from the facilities to their associated potential sites. In other words, we set $r = \mathrm{Min}\left(r_o, \underset{j \in X}{\mathrm{Min}}\left(R_j\right)\right)$

where $R_j = \max_{i \in S_j} d(i,j)$ and $r_o = (\lambda / p)\sum_{j=1}^{p} R_j$. This setting is used to ensure that the search is more restrictive while remaining within each R_j, j=1,…,p. λ is a correction parameter which we set, in our experiments, to 0.25. This was found empirically using a small sample problem. By using this restriction, the number of potential sites used in the procedure "FindBestCustomer" is drastically decreased which led to a massive reduction in the computing time of the local search without a significant loss in solution quality. Here, we do not apply the data structure of Resende and Werneck as the number of potential facility sites is relatively large (*n*) which creates an excessive memory problem due to the use of a two dimensional matrix as part of its data structure.

5. Computational Results

To assess the performance of our solution method, we carried out an extensive computational study. The code was written in C++ .Net 2010 and used the IBM ILOG CPLEX version 12.5 Concert Library. The tests were run on a PC with an Intel Core i5 CPU 650@ 3.20GHz processor, 4.00 GB of RAM and under Windows 7(32bit).

In our computational experiments, we used two existing datasets from the literature and a new dataset with guaranteed optimal solutions which we constructed. We first provide preliminary computational results for the two local searches and the two variants. Full computational experiments on the *p*-median problem are given next.

The existing datasets

These consist of the BIRCH and the TSP datasets. The BIRCH dataset is kindly provided by Avella et al. (2012) in http://iv.icc.ru/Papers.hatml whereas the TSP dataset can be downloaded from http://www.tsp.gatech.edu/world/countries.html.

The newly generated dataset

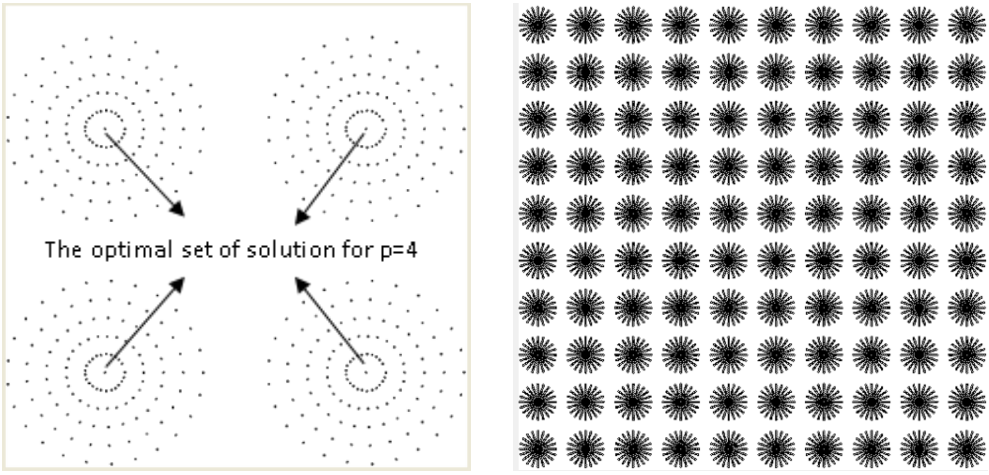
This is constructed using a well-defined though trivial geometric structure so to guarantee optimality when the value of p is equal to the number of groups/clusters in the dataset. We refer to the new dataset as the 'Circle dataset'. Two examples of this dataset (n = 500, p = 4 and n = 20000, p = 100) are shown in Figure 6. The algorithm for generating the Circle dataset is given in Appendix B and its proof of optimality is provided in Appendix C. This is based on the following two items: (i) the *p*-median problem reduces to p 1-median problems; (ii) the optimal centre of each cluster reduces to the same optimal centre of each ring which is the point as defined in our construction. Different instances of this dataset can be downloaded from the CLHO (2013) website (http://www.kent.ac.uk/kbs/research/research-centres/clho/datasets.html).

Some notation

The results of our experiments are presented in several tables. The notation in the tables is as follows:

- n: number of demand points
- p: number of medians

- Z: objective function value
- Time: computational time
- Deviation(%): this is the percent gap from the best known solution and is computed as: $Deviation = 100\left(\frac{Z_c - Z_b}{Z_b}\right)$, where Z_c and Z_b correspond to the Z value obtained with method '*c*' and the best Z value respectively. This is equivalent to the optimality error as defined by Casillas (1987).
- 'Bold' values in the table refer to the best solutions.



n = 500, *p* =4 *n* = 20000, *p* = 100

Figure 6. Two examples of the Circle Dataset (CLHO, 2013)

5.1. Preliminary Empirical Comparison

In this subsection we conduct small experiments to assess the performance of the two local searches as well as the two variants which we discussed in the earlier section.

a) The two local searches (RW vs IS-RW)

Small experiments on 3 TSP datasets (mu1979, tz6117, and ym7663) were conducted to compare the performance of RW and IS-RW. Values of *p* varying from 10 to 100 with an increment of 10 are used. Each instance was executed 10 times for both local searches starting from the same initial solution. Table 1 shows the performance of both local searches where Saving (%) refers to the percentage saving in CPU time of IS-RW over RW. From this table, it can be noted that IS-RW runs approximately 15-36% faster than RW. However, the quality of the solution obtained by IS-RW is, as expected, slightly affected with a

deterioration between 2% and 11%. We use IS-RW when solving the aggregated problems (Phase 1) as the obtained facility locations of the aggregated problem do not necessary yield the corresponding best solution of the original problem. In other words, the main objective in Phase 1 is to identify the promising facility locations while consuming a smaller amount of computing time.

Table 1 The average of CPU time of RW and IS-RW on the TSP data (in seconds)

p	mu1979 (*n* = 1979)			tz6117 (*n* = 6117)			ym7663 (*n* = 7663)		
	RW	IS+RW	Saving (%)	RW	IS+RW	Saving (%)	RW	IS+RW	Saving (%)
10	0.51	0.54	-6.09	5.56	5.57	-0.07	11.32	12.76	-12.75
20	0.46	0.48	-4.64	4.27	4.99	-16.93	8.14	8.56	-5.12
30	0.48	0.40	16.98	4.14	4.60	-11.22	7.52	7.80	-3.66
40	0.56	0.41	26.48	4.25	4.25	0.11	7.81	6.79	13.02
50	0.65	0.42	35.31	4.61	4.13	10.52	7.56	6.43	14.98
60	0.71	0.39	45.56	4.91	4.11	16.40	8.43	6.48	23.20
70	0.86	0.39	54.52	5.41	3.96	26.82	8.88	7.15	19.48
80	1.00	0.40	60.18	6.22	3.94	36.59	9.90	6.86	30.70
90	1.17	0.42	64.09	6.96	4.18	39.87	10.99	6.76	38.44
100	1.32	0.43	67.49	8.16	4.23	48.12	12.02	6.84	43.06
Average	0.77	0.43	**35.99**	5.45	4.40	**15.02**	9.26	7.64	**16.14**

b) The two Variants (Var1 and Var2)

We also tested Var1 and Var2 using IS-RW as a local search on the TSP dataset varying in size from n = 734 to 9,976. We increase the value of p with n. Each instance was executed 10 times and every run in both approaches used the same initial solution. The summary results are presented in Table 2 which shows the average of the objective function (Z), the deviation (%), the average total CPU and the average shaking time. As the "Local Search with Shaking" is used only once for each subproblem, referring to the average behaviour rather than the best is, in our view, more reliable.

In general, Var2 generates better results than Var1 as it produces both a higher number of smaller average objective values and a smaller deviation (0.09841%). This means that conducting the shaking process k times does improve the quality of the solution. To our surprise, the table also shows that Var2 runs faster than Var1. This could be due to the fact that the shaking process does not only affect the performance of the local search but makes the task of the local search relatively easier. Moreover, the shaking time is found to be negligible when compared to the total CPU time (approx. 1.89% extra CPU time only for

Var2 in the worst case, see for example instance ei8246). These results support our choice of using Var2 within the "Local Search with Shaking".

Table 2 Comparison between the Var1 and the Var2 Methods

File Name	*n*	*p*	*Z*		Deviation (%)		Average Total CPU Time (milliseconds)		Average Shaking Time (milliseconds)	
			Var1	Var2	Var1	Var2	Var1	Var2	Var1	Var2
uy734	734	25	209,214	**207,647**	0.75460	-	64.90	54.20	0.00	1.00
zi929	929	30	**208,002**	209,374	-	0.65959	108.50	98.40	0.10	1.70
mu1979	1979	35	320,885	**320,777**	0.03363	-	401.50	336.40	0.30	5.40
ca4663	4663	40	6,885,883	**6,817,921**	0.99681	-	2,878.80	2,202.70	1.20	16.50
tz6117	6117	45	2,412,269	**2,371,727**	1.70937	-	4,171.10	3,597.60	2.00	37.10
eg7146	7146	50	1,010,761	**1,003,380**	0.73559	-	7,529.00	5,733.10	2.20	49.30
ym7663	7663	55	1,430,669	**1,416,127**	1.02695	-	6,934.80	5,578.30	2.90	55.90
ei8246	8246	60	**1,235,810**	1,241,036	-	0.42290	8,851.60	6,137.40	3.20	116.30
ja9847	9847	65	3,365,986	**3,311,024**	1.65999	-	12,925.50	10,102.90	3.90	106.50
gr9882	9882	70	1,869,116	**1,859,538**	0.51504	-	13,235.00	9,655.40	3.70	84.50
kz9976	9976	75	6,378,764	**6,340,439**	0.60445	-	13,657.00	12,725.60	3.60	71.80
				Average	0.73058	**0.09841**	6,432.52	**5,111.09**	2.10	49.64

5.2. Experiments on large *p*-median problems

In our computational study, we set the parameters as follows: $m = 0.1n$, T = 10, L_{max} = 300, and *itermax* = 5. Those parameters were chosen based on a small preliminary study. The number of aggregated points is only 10% of the number of demand points. The value of *m* affects the quality of the solution. The higher the value of *m*, the higher is the chance of getting a better solution. However, the computing time also increases with increasing values of *m*. The number of iterations (T) in Phase 1 also influences the quality of the obtained solution. The likelihood of getting a good solution increases when T is high, which also increases diversification, but at the expense of a longer computing time. We set L_{max} = 300 as CPLEX runs relatively long when the size of the problem exceeds this value. The method terminates when there is no improvement in five consecutive iterations (*itermax* = 5). In addition, we fixed the seed for the random generator to a constant, say *m*, so the results can be reproducible if need be.

Different settings were considered for the parameter ρ used to determine the threshold distance τ in the BCA method ($\tau = \rho\delta$). Based on some preliminary tests, we set ρ = 0 for the clustered datasets and ρ = 0.25 for the non-clustered ones. The BIRCH and Circle datasets belong to the clustered dataset category whereas the TSP fits the non-clustered category. This

choice of the parameter ρ implies that the threshold distance is relatively small for the clustered dataset as the demand points are spread in a certain area. For the non-clustered dataset, we set the number of aggregated points generated from the promising facility locations based on the BCA method to be 75% (γ = 0.75), whereas the remaining 25% were generated randomly. For the clustered dataset, we set γ = 0.90.

5.2.1. Experiments on the existing datasets

The BIRCH dataset includes the largest instances tested in the literature (*n* ranges from 25,000 to 89,600). For the TSP dataset, we use the Italy, Sweden, Burma, and China instances (*n* ranges from 16,862 to 71,009).

Case 1: BIRCH Dataset

The results of our experiments on the BIRCH dataset are compared with the ones obtained by Irawan and Salhi (2013), Avella et al. (2012), and Hansen et al. (2009). We refer to these 3 methods as IS, AV, and VNSH respectively. The computational results of the AV and VNSH methods are taken from Avella et al. (2012). The value of *p* ranges between 25 and 64.

The specification of the computer used to execute IS is the same as the one used here. Computational experiments for AV and VNSH were carried out by Avella et al. (2012) on an Intel Core 2Quad CPU 2.6 GHz, 4.00 GB of RAM and under Windows XP64. Dongarra's transformation is used to provide a fair comparison in terms of CPU time. The formulation of this transformation is as follow: $T_2 = T_1 \frac{Nf_1}{Nf_2}$, where T_1 denotes the reported time in Machine 1 and T_2 the estimated time in Machine 2. Nf_1 and Nf_2 represent the number of Mflops in Machines 1 and 2 respectively. The software used to record the values of Nf_1 and Nf_2 can be downloaded from http://www.roylongbottom.org.uk. In that software, we record the value of 32 bit SSE MFLOPS. As we could not obtain precisely the number of Mflops of the computer used by Avella et al. (2012), we provide an approximation based on a slightly slower but similar computer available to us, namely a PC Intel Core 2Duo 2.6GHz, 4 GB of RAM.

The computational results for our method (AA) on the BIRCH dataset are presented in Table 3 where the summary results of the four methods (AA, IS, AV, and VNSH) are shown: the best known objective function (Z), the deviation (%), and the time (in seconds). In these

experiments, we used two types of BIRCH instances, namely BIRCH instances of type 1 and BIRCH instances of type 3.

Table 3 Computational Results for the AA method on the BIRCH dataset

File Name	*n*	*p*	*Z* Best Known	Deviation (%)				Time (seconds)			
				VNSH	AV	IS	AA	VNSH	AV	IS	AA
BIRCH instances of type 1											
1	25,000	25	31,229.3	0.430	0.171	**0.000**	**0.000**	206	447	157	93
2	36,000	36	45,115.6	**0.000**	0.245	**0.000**	**0.000**	590	780	373	215
3	49,000	49	61,384.1	**0.000**	0.302	**0.000**	**0.000**	818	1,216	612	421
4	64,000	64	79,987.3	**0.000**	0.438	0.083	0.083	1,527	2,258	1,110	939
5	30,000	25	37,563.6	0.001	0.142	**0.000**	**0.000**	321	559	223	133
6	43,200	36	54,191.4	**0.000**	0.211	**0.000**	**0.000**	767	1,003	434	244
7	58,800	49	73,626.8	**0.000**	0.310	**0.000**	**0.000**	1,454	1,691	792	566
8	76,800	64	95,989.1	**0.000**	0.421	0.052	0.052	2,931	2,834	1,510	1,020
9	35,000	25	43,902.1	**0.000**	0.159	**0.000**	**0.000**	569	768	353	197
10	50,400	36	63,169.2	**0.000**	0.253	**0.000**	**0.000**	1,185	1,472	645	460
11	68,600	49	85,833.5	**0.000**	0.289	**0.000**	**0.000**	1,787	2,441	1,149	666
12	89,600	64	112,059.2	**0.000**	0.380	**0.000**	**0.000**	3,678	4,501	2,069	1,487
# best				**10**	0	**10**	**10**				
Average				0.0360	0.2769	**0.0113**	**0.0113**	1,319.42	1,664.17	785.62	536.80
BIRCH instances of type 3											
21	25,000	25	17,696.2		0.1268	**0.0000**	**0.0000**		527	125	138
22	36,000	36	27,423.0		0.1935	**0.0000**	**0.0000**		913	365	216
23	49,000	49	44,149.0		0.3025	0.1208	**0.0000**		1,760	526	398
24	64,000	64	58,840.3		0.2570	0.1055	**0.0000**		2,624	1,049	968
25	30,000	25	21,829.9		0.1614	**0.0000**	**0.0000**		832	454	170
26	43,200	36	32,339.4		0.1614	**0.0000**	**0.0000**		1,873	492	342
27	58,800	49	50,857.9		0.2502	**0.0000**	**0.0000**		2,692	899	946
28	76,800	64	66,573.6		0.5574	0.2525	**0.0000**		4,393	1,892	1,836
29	35,000	25	24,811.0		0.0913	**0.0000**	0.0078		972	288	290
30	50,400	36	38,102.6		0.1566	0.0001	**0.0000**		2,297	611	659
31	68,600	49	61,857.1		0.2430	0.0410	**0.0000**		3,556	1,035	879
32	89,600	64	78,777.5		0.5938	**0.0000**	0.0092		5,779	2,189	1,688
# best					0	7	**10**				
Average					0.2579	0.0433	**0.0014**	T1	2,351.50	827.15	710.86
								Mflops	3,545.00	4,415.00	4,415.00
								T2	1,888.12	827.15	710.86

On the BIRCH instances of type 1, the AA's results are similar to the ones of IS but AA is approximately 15% faster than IS. AA provides better solutions compared to AV. Compared to VNSH, it produces similar objective function values, while yielding a slightly smaller deviation (0.0113%). On the BIRCH instances of type 3, the upper bound of VNSH was not provided by Avella et al. (2012). AA outperforms IS and AV where AA found 10 best solutions and yielded the smallest deviation (0.0014%). As stated in Hansen et al. (2009), our experiments also show that the BIRCH instances of type 3 are harder to solve compared to type 1 instances.

Compared to AV, according to Dongarra's transformation, the specification of the computer used to execute our method (AA) is approximately 25% faster than the one used by Avella et al. (2011). Table 3 also presents the transformed computing time (T_2). Note that AA is faster than both IS and AV while generating relatively better results. Overall, AA found 5 new best solutions for this difficult type of instances.

Case 2: TSP Dataset

The computational results for the TSP dataset are given in Table 4. There are 4 instances (Italy, Sweden, Burma, and China), where each instance is solved with p varying from 25 to 100 with an increment of 25, totalling 16 instances. Our results are compared with the ones of IS. For $p \geq 100$, we set $\gamma = 0.90$ and $\rho = 0.10$.

Table 4 Computational Results for the AA method on the TSP dataset

Description	p	The Best Known Z	Deviation (%)		Time (seconds)	
			IS	AA	IS	AA
Italy Data (n=16,862)	25	7,406,925.33	0.05762	**0.00000**	96.82	70.34
	50	5,100,031.58	0.21061	**0.00000**	119.01	109.02
	75	4,087,255.68	0.01946	**0.00000**	233.80	136.37
	100	3,490,908.55	0.05437	**0.00000**	350.33	167.82
Sweden Data (n=24,978)	25	14,098,813.68	**0.00000**	0.54322	348.43	215.89
	50	9,665,260.28	0.02729	**0.00000**	278.59	256.11
	75	7,783,165.38	**0.00000**	0.05237	450.64	287.71
	100	6,677,281.78	**0.00000**	0.10613	637.66	288.46
Burma Data (n=33,708)	25	18,226,756.71	0.00160	**0.00000**	418.36	273.13
	50	12,597,568.51	0.04881	**0.00000**	479.49	451.90
	75	10,187,180.83	0.18312	**0.00000**	651.47	527.29
	100	8,731,907.79	0.18702	**0.00000**	1,107.66	449.62
China Data (n=71,009)	25	113,794,799.80	0.01586	**0.00000**	2,227.85	2,556.07
	50	78,526,961.86	0.13582	**0.00000**	2,901.61	3,404.75
	75	63,786,587.67	0.37598	**0.00000**	3,217.66	3,339.16
	100	54,865,606.72	0.00809	**0.00000**	4,030.75	2,958.15
	#Best		3	**13**		
		Average	0.08285	**0.04386**	1,096.88	968.24

In general, our method (AA) produces better results than the ones of IS. Here, AA yields a higher number of best solutions and a smaller deviation (0.04386%). The proposed approach also produces 13 new best solutions which can be used for further benchmarking. Based on the average computing time, AA runs approximately 12% faster than IS.

5.2.2. Experiments on the newly constructed dataset with guaranteed optimality

By conducting experiments on the Circle dataset, the optimal errors can be obtained. The number of demand points (n) ranges from 20,000 to 60,000 with an increment of 10,000 whilst the number of opened facilities (p) is equal to 0.5%n and 1%n. There are 10 instances denoted by C1 to C10. Table 5 presents the results of these experiments. Z^* refers to the optimal objective function value.

Table 5 shows that AA produces the optimal solutions for all instances. Overall, it seems that the Circle dataset can be solved quite easily by our method. With respect to the computing time, solving the problems with larger p requires more time than the one with smaller p. This could be partly due to the fact that in the "Local Search with Shaking", k_{max} is set to p and as $k \in (1, k_{\max})$, consequently a larger value of p will obviously increase the computational burden. It is worth noting that although these instances are visually trivial, they were constructed purposely this way to assess the ability of our approach to find optimal solutions. It is not that obvious for the algorithm to find the solutions as no extra information is fed into the search.

Table 5 Computational Results for the AA method on the Circle dataset

File Name	n	p	Z*	Z_{AA}	Deviation (%)	Time (Seconds)
C1	20,000	100	1,648,585.87	1,648,585.87	0.0000	86.57
C2	20,000	200	897,033.86	897,033.86	0.0000	172.51
C3	30,000	150	2,472,878.81	2,472,878.81	0.0000	245.55
C4	30,000	300	1,234,194.78	1,234,194.78	0.0000	372.03
C5	40,000	200	3,297,171.74	3,297,171.74	0.0000	475.53
C6	40,000	400	1,645,593.04	1,645,593.04	0.0000	592.67
C7	50,000	250	4,214,610.74	4,214,610.74	0.0000	843.53
C8	50,000	500	2,056,991.29	2,056,991.29	0.0000	1,108.13
C9	60,000	300	4,945,757.61	4,945,757.61	0.0000	1,250.56
C10	60,000	600	2,468,389.55	2,468,389.55	0.0000	1,767.75
				Average	**0.0000**	691

6. The conditional *p*-median problem

In this section we review some papers focusing on the conditional p-median problem followed by the adaptation of our approach to this related problem and a summary of some computational results.

6.1. A brief review of the conditional *p*-median problem

The conditional location problem was first formally introduced by Minieka (1980) where conditional centers and medians on a graph were investigated. Chen (1990) developed a method for solving minisum and minimax conditional location-allocation problems with $p \geq 1$. An algorithm that requires the one-time solution of an unconditional $(p+1)$ center or $(p+1)$ median for solving the conditional $(p+1)$ center or $(p+1)$ median on networks was suggested by Berman and Simchi-Levi (1990).

Drezner (1995) proposed a general heuristic for the conditional *p*-median problem on both the network and the plane. In his paper, the term "(*p*,*q*) median problem" was introduced. Let Q denote the set of existing facilities where $Q \subset J$. The objective function for the *p*-median problem, equation (1), can be modified as follow (see Drezner, 1995):

$$Z = \sum_{i \in I} w_i \left[\operatorname{Min} \left\{ \operatorname*{Min}_{j \in Q} \{d(i,j)\}, \operatorname*{Min}_{j \in J, j \notin Q} \{d(i,j)\} \right\} \right] \tag{9}$$

As $D_i = \operatorname*{Min}_{j \in Q} \{d(i,j)\}$ can be computed for each $i \in I$ beforehand, equation (9) can be rewritten as :

$$Z = \sum_{i \in I} w_i \left[\operatorname{Min} \left\{ D_i, \operatorname*{Min}_{j \in J, j \notin Q} \{d(i,j)\} \right\} \right] \tag{10}$$

The use of equation (10) is computationally more efficient as it avoids unnecessary calculations.

Berman and Drezner (2008) suggested a method for solving both the conditional *p*-median and *p*-center problems. The method needs the one-time solution of an unconditional *p*-median and *p*-center problem using the shortest distance matrix. A hybridization approach combining a harmony search and a greedy heuristic for solving *p*-median problems was recently proposed by Kaveh and Esfahani (2012).

6.2. The adaptation of AA for the large (*p*,*q*) median problems

Our proposed method (AA) which is designed to solve large *p*-median problems can easily be adapted for tackling large (*p*,*q*) median problems. Our revised approach, which is referred to as AAq, contains the following minor modifications.

a) The aggregation method

The q existing facility locations are always included in the promising facility locations (E) which are then used to aggregate the points. These locations are also used in Phase 1, meaning that $E \neq \emptyset$ in Phase 1 but $E = Q$.

b) The "Local Search with Shaking"

In both the shaking and the local search, the existing facilities are always retained open in the solutions. In other words, the existing facilities are not even checked for possible removal.

- *The shaking*

  When finding the best facility to be removed (say facility j) from the current solution, facility j is not one of the existing facilities (i.e. $j \notin Q$).

- *The local search*

  The implementation of the best improvement strategy does not include the existing facilities as these locations are always part of the solution.

c) The exact method

The implementation of the exact method with CPLEX is still using equations (1) – (6). However, constraints (11) are added to ensure that the existing facilities are always in the solution.

$$X_j = 1 \qquad \forall j \in Q \tag{11}$$

The introduction of such constraints (11) into the p-median formulation makes the problem relatively much easier to solve. In our study, we are now able to increase the value of L_{max} from 300 to 1100 while using a similar amount of computational time.

d) The post-optimisation (the local search on the original problem)

The modification in the post-optimisation is quite similar to the local search in the "Local Search with Shaking" described earlier in part *(b)*.

6.3. Computational results on the (p,q) median problem

The configuration of the parameters used for solving the (p,q) median problem is similar to the one for the p-median problem except we set L_{max} = 1,100. We test our modified method on the TSP dataset (Italy, Sweden, Burma, and China) that has already been tested on the unconditional p-median problem. We opted for this dataset so that we could use the solutions obtained by solving the p-median problem to set the existing q facilities in the (p,q) median problem. Namely, the q existing facility locations are obtained from the solution of the p-

median problem solved in the previous section. For example for the (p=25, q=25) median problem, the existing 25 facility locations are from the solution of the (p=25) median problem. The objective function of the (p=25, q=25) median problem is then compared to the one of the unconditional (p=50) median problem which is taken as a lower bound. In this case, the objective function value of the (p=25, q=25) median problem should be worse than or equal to the one of the (p=50) median problem. Note that such a claim is only valid if an exact method is used instead of a heuristic.

The computational results of AAq on the TSP dataset are given in Table 6. The deviation (%) is the gap between the objective function value found by the AAq and the one by AA. The results show that solving (p,q) using the AAq requires almost a third less amount of computing time than AA. This is quite expected as in the local search, the existing facilities are already fixed and consequently the number of combinations in the swapping procedure decreases drastically.

Table 6 Computational Results for the (p,q) median problem on the TSP dataset

Description	p-Median problem (AA)			(p, q)-Median problem (AAq)				
	p	Z	Time (seconds)	p	q	Z	Deviation (%)	Time (seconds)
Italy Data (n=16,862)	50	5,100,031.58	109.02	25	25	5,407,411.93	6.03	42.18
	75	4,087,255.68	136.37	50	25	4,272,997.47	4.54	99.25
				25	50	4,302,311.14	5.26	60.98
	100	3,490,908.55	167.82	50	50	3,698,225.76	5.94	65.57
				25	75	3,605,335.36	3.28	28.18
Sweden Data (n=24,978)	50	9,665,260.28	256.11	25	25	10,163,911.00	5.16	100.58
	75	7,783,165.38	287.71	50	25	8,098,016.88	4.05	140.64
				25	50	8,155,349.71	4.78	116.45
	100	6,677,281.78	288.46	50	50	7,020,252.82	5.14	200.01
				25	75	6,898,906.64	3.32	100.95
Burma Data (n=33,708)	50	12,597,568.51	451.90	25	25	13,099,836.27	3.99	191.39
	75	10,187,180.83	527.29	50	25	10,557,462.19	3.63	322.85
				25	50	10,582,656.48	3.88	152.14
	100	8,731,907.79	449.62	50	50	9,155,818.79	4.85	351.97
				25	75	9,052,184.99	3.67	164.34
China Data (n=71,009)	50	78,526,961.86	3,404.75	25	25	82,437,517.47	4.98	911.16
	75	63,786,587.67	3,339.16	50	25	66,448,542.12	4.17	2,068.54
				25	50	66,367,405.04	4.05	792.73
	100	54,865,606.72	2,958.15	50	50	57,296,333.65	4.43	1,353.44
				25	75	56,649,744.56	3.25	705.66
		Average	1,031.36				4.42	398.45

On average, the deviation between the objective function value of the (p,q) median problem and the one of the p-median problem is 4.42%. Table 6 also shows interesting results where the objective function value of a more restricted problem happens to be, in some cases, smaller than the less restricted. For instance, the Z value for (p=25, q=75) median problem is smaller than the one of (p=50, q=50) problem. This could be due to two reasons: (i) the q facilities for the two cases are not necessarily the same and hence may have different effect and (ii) using a heuristic approach could also yield different solution for the two cases.

We have also attempted another variant of the AAq where in the post-optimization phase (Phase 4), a VNS is used instead of the local search. This was possible as the computing time of the AAq's local search is usually quite fast. For the VNS, we set $k_{max} = p$. The use of the VNS in the post-optimisation yields slightly better deviation (4.41%) at the expense of a much longer computing time (almost 6 times). Out of 20 instances, the VNS improves slightly the quality of the solutions on 8 instances. The objective values are reported here for benchmarking purposes only. Italy: Z(p=50,q=25) = 4,272,752.66; Z(50,50) = 3,697,823.05; Sweden: Z(25, 25) = 10,163,883.03; Z(50, 25) = 8,093,829.14; Z(50, 50) = 7,017,515.03; China: Z(25, 25) = 82,437,310.68; Z(50, 25) = 66,444,176.54; Z(50, 50) = 57,296,280.91.

7. Conclusion and suggestions

An adaptive approach based on data aggregation, the use of a "Local Search with Shaking" and an efficient implementation of VNS/CPLEX is proposed to solve large unconditional and conditional p-median problems. The method consists of four phases. The first two phases are part of a learning process where demand point aggregation, a "Local Search with Shaking", and a VNS/CPLEX are utilised to obtain a better initial solution. The third phase is an iterative-based phase which incorporates demand point aggregation and the "Local Search with Shaking" to improve the solution. The last phase is a post-optimisation process performed on the original problem.

The computational results show that our approach performs well and runs relatively fast. For the unconditional problem, the proposed approach was tested on three types of datasets. The first one is the BIRCH dataset. On the BIRCH instances of type 1, our method produces better solution compared to the one by Avella et al. (2012) and similar solutions to the ones by Irawan and Salhi (2013) and by Hansen et al. (2009). On the BIRCH instances of type 3, our method is superior than the ones of Avella et al. (2012) and Irawan and Salhi (2013). On the TSP dataset, the results show that our method clearly outperforms the Irawan and Salhi

(2013) method as it finds 13 new best known solutions for the 16 instances and reduces the average deviation. On the Circle dataset, which we constructed to guarantee geometrically optimal solutions, the proposed method is able to find all the optimal solutions. For the conditional problem, the adapted method was only assessed on the TSP dataset. The results also reveal that our method performs quite well when compared against the results of the unconditional *p*-median problem which are used as lower bounds.

This study could be extended to investigate other related location problems such as large vertex *p*-center problems and their counterparts on the plane. The proposed method could also be adapted for clustering of large datasets with higher dimension as part of data mining.

Acknowledgment

The authors would like to thank both referees for their useful suggestions that improved both the content as well as the presentation of the paper. We are also grateful to the Indonesian Government and Jurusan Teknik Industri-ITENAS Bandung, Indonesia for the sponsorship of the first author.

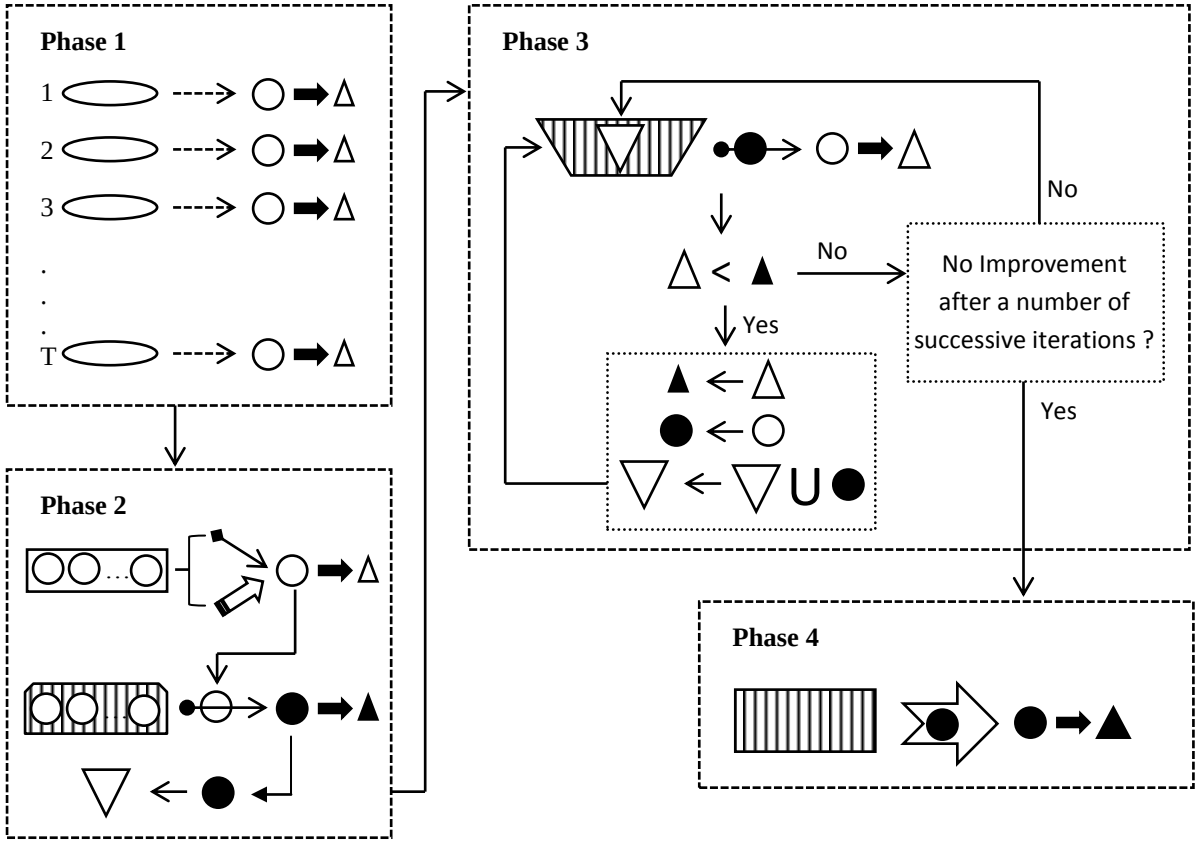
References

1. Andersson, G., Francis, R. L., Normark, T., & Rayco, M. B. (1998). Aggregation method experimentation for large-scale network location problems. *Location Science, 6*, 25-39.
2. Avella, P., Boccia, M., Salerno, S., & Vasilyev, I. (2012). An aggregation heuristic for large scale p-median problem. *Computers and Operations Research, 39*, 1625-1632.
3. Bach, L. (1981). The problem of aggregation and distance for analysis of accessibility and access opportunity in location-allocation models. *Environment and Planning A, 13*, 955–978.
4. Ballou, R. H. (1994). Measuring transport costing error in customer aggregation for facility location. *Transportation Journal, 33*, 49-59.
5. Berman, O. and Simchi-Levi, D. (1990). The conditional location problem on networks. *Transportation Science, 24*, 77-78.
6. Berman, O. and Drezner, Z. (2008). A new formulation for the conditional *p*-median and *p*-center problems. *Operations Research Letters, 36*, 481-483.
7. Brimberg, J., & Mladenovic, N. (1996). A variable neighbourhood algorithm for solving the continuous location-allocation problem. *Studies of Locational Analysis, 10*, 1-12.
8. Bowerman, R. L., Calamai, P. H., & Brent Hall, G. (1999). The demand partitioning method for reducing aggregation errors in p-median problems. *Computers and Operations Research, 26*, 1097-1111.
9. Casillas, P. (1987). Aggregation problems in location-allocation modeling. In Gosh, A., & Rushton, G. (Eds.), *Spatial analysis and location-allocation models* (pp. 327-344). New York: Van Nostrand Reinhold.

10. Chen, R. (1990). Conditional minisum and minimax location-allocation problems in euclidean space. *Transportation Science, 22,* 158-160.
11. Church, R. L. (2003). COBRA: A new formulation for the *p*-median location problem. *Annals of Operations Research, 122*, 103–120.
12. Church, R. L. (2008). BEAMR: An exact and approximate model for the *p*-median problem. *Computers and Operations Research, 35*, 417-426.
13. CLHO (2013). Centre of Logistic and Heuristic Optimization, Kent Business School, University of Kent at Canterbury, UK. http://www.kent.ac.uk/kbs/research/research-centres/clho/datasets.html
14. Current, J. R., & Schilling, D. A. (1987). Elimination of source A and B errors in p-median location problems. *Geographical Analysis, 19*, 95-110.
15. Dongarra, J. J. Performance of various computers using standard linear equation software. http://www.netlib.org/benchmark/performance.pdf (Accessed online 15 April 2013).
16. Drezner, Z. (1995). On the conditional *p*-median problem. *Computers and Operations Reasearch, 22*, 525-530.
17. Erkut, E., & Bozkaya, B. (1999). Analysis of aggregation errors for the p-median problem. *Computers and Operations Research, 26,* 1075-1096.
18. Fotheringham, A. S., Densham, P. J., & Curtis, A. (1995). The zone definition problem in location-allocation modeling. *Geographical Analysis, 27,* 60-77.
19. Francis, R. L., & Lowe, T. J. (1992). On worst-case aggregation analysis for network location problems. *Annals of Operations Research, 40*, 229-246.
20. Francis, R. L., Lowe, T. J., & Rayco, M. B. (1996). Row-column aggregation for rectilinear distance p-median problems. *Transportation Science, 30*, 160-174.
21. Francis, R. L., Lowe, T. J., Rayco, & M. B., Tamir, A. (2009). Aggregation error for location models: survey and analysis. *Annals of Operations Research, 167*, 171-208.
22. Francis, R. L., Lowe, T. J., & Rayco, A. (2000). Aggregation error bounds for a class of location models. *Operations research, 48*, 294.
23. Garcia, S., Labbe, M., & Marin, A. (2010). Solving large p-median problem with a radius formulation. *INFORMS Journal on Computing, 23*, 546-556.
24. GoodChild, M.F. (1979). The aggregation problem in location-allocation. *Geographical Analysis, 11*, 240-255.
25. Hansen, P., Brimberg, J., Urosevic, D., & Mladenovic, N. (2009). Solving large p-median clustering problems by primal-dual variable neighborhood search. *Data Mining and Knowledge Discovery, 19*, 351-375.
26. Hansen, P., & Mladenovic, N. (1997). Variable neighbourhood search for the p-median. *Location Science, 5*, 207-225.
27. Hansen, P., & Mladenovic, N. (2001). Variable neighbourhood search: Principles and applications. *European Journal of Operational Research, 130*, 449-467.
28. Hansen, P., Mladenovic, N., & Perez, J. A. M. (2010). Variable neighbourhood search: methods and applications. *Annals of Operations Research, 175,* 367-407.
29. Hillsman, E. L., & Rhoda, R. (1978). Errors in measuring distances from populations to service centers. *Annals of Regional Science, 12*, 74-88.

30. Hodgson, M. J. (2002). Data surrogation error in p-median models. *Annals of Operations Research, 110*, 153-165.
31. Hodgson, M. J., & Neuman, S. (1993). A GIS approach to eliminating source C aggregation error in p-median models. *Location Science, 1*, 155-170.
32. Hodgson, H., & Salhi, S. (1998). Using a quadtree structure to eliminate aggregation error in point to point allocation. Presented at INFORS Conference, Montreal.
33. Hodgson, M. J., Shmulevitz, F., & Körkel, M. (1997). Aggregation error effects on the discrete-space p-median model: The case of Edmonton, Canada. *Canadian Geographer / Le Géographe canadien, 41*, 415-428.
34. Irawan C. A., & Salhi S. (2013). Solving large p-median problems by a multistage hybrid approach using demand points aggregation and variable neighbourhood search. *Journal of Global Optimization*, (DOI : 10.1007/s10898-013-0080-z)
35. Kariv, O., & Hakimi, S. L. (1969). An algorithmic approach to network location problems; part 2. The p-medians. *SIAM Journal on Applied Mathematics, 37*, 539-560.
36. Kaveh, A. and Esfahani, H. N. (2012). Hybrid harmony search for conditional *p*-median problems. *International Journal of Civil Engineering, 10*, 32-36.
37. Minieka, E. (1980). Conditional Centers and Median on a Graph. *Network, 10*, 265-272.
38. Mirchandani, P. B., & Reilly, J. M. (1986). "Spatial nodes" in discrete location problems. *Annals of Operations Research, 6*, 203-222.
39. Mladenovic, N., Brimberg, J., Hansen, P., & Moreno-Perez, J. A. (2007). The p-median problem: A survey of metaheuristic approaches. *European Journal of Operational Research, 179*, 927-939.
40. Murray, A. T., & Gottsegen, J. M. (1997). The influence of data aggregation on the stability of p-median location model solutions. *Geographical Analysis, 29*, 200-213.
41. Oshawa, Y., Koshizuka, T., & Kurita, O. (1991). Errors caused by rounded data in two simple facility location problems. *Geographical Analysis, 23*, 56-73.
42. Osman, I. H., & Salhi, S. (1996). Local search strategies for the vehicle fleet mix problem. In Rayward-Smith, V. J., Osman, I. H., Reeves, C. R., & Smith, G. D. (Eds.), *Modern Heuristic Search Methods* (pp. 131-154). Chichester: John Wiley & Sons Ltd.
43. Plastria, F. (2001). On the choice of aggregation points for continuous p-median problems: a case for the gravity center. *TOP*, *9*, 217–242
44. Qi, L., & Shen, Z. M. (2010). Worst-case analysis of demand point aggregation for the Euclidean p-median problem. *European Journal of Operational Research, 202*, 434-443.
45. ReVelle, C.S., & Swain, R. (1970). Central Facilities Location. *Geographical Analysis, 2*, 30–42
46. Resende, M. G. C. & Werneck, R. (2007). A fast swap-based local search procedure for location problems. *Annals of Operations Research, 150*, 205-230.
47. Rosing, K. E., ReVelle C. S., & Rosing-Vogelaar, H. (1979). The p-median and its linear programming relaxation: An approach to large problems. *Journal of the Operational Research Society, 30*, 815-822.
48. Salhi, S., & Gamal, M. D. H. (2003). A Genetic Algorithm Based Approach for the Uncapacitated Continuous Location–Allocation Problem. *Annals of Operations Research, 123*, 203-222.
49. Whitaker, R. A. (1983). A fast algorithm for the greedy interchange for large-scale clustering and median location problems. *INFOR, 21*, 95-108

Appendix A. An illustration of the adaptive approach (AA)



An aggregated problem (*m* demand points and *m* potential facility locations)

Solving the aggregated *p*-median problem using a "Local Search with Shaking (*m*, *m*, *p*)"

The obtained facilities (F_t') found by a "Local Search with Shaking (*m*, *m*, *p*)", $t=1,\ldots,T$

Allocating all demand points in *C* to their nearest obtained facilities

The objective function calculated using the original problem

An aggregated problem made up of all solutions ($L = \bigcup_{t=1}^{T} F_t'$)

Solving the aggregated *p*-median problem using a VNS (|*L*|, |*L*|, *p*)

Solving the aggregated *p*-median problem using CPLEX (|*L*|, |*L*|, *p*)

An aggregated problem (*n* demand points and |*L*| potential sites)

Solving the aggregated problem using a "Local Search with Shaking (*n*,|*L*|,*p*)" with O as the initial solution

Promising Facility Locations

Set of facility locations which gives the smallest objective function

The smallest objective function

An aggregated problem (*n* demand points and *m* potential sites including the promising sites)

Solving the aggregated problem using a "Local Search with Shaking (*n*, *m*, *p*)" with ● as the initial solution

The original problem (*n* demand points and *n* potential sites)

Solving the *p*-median problem using a "Local Search (*n*, *n*, *p*)" with ● as the initial solution

Figure A.1. An illustration of the adaptive approach (AA)

Appendix B. The generator for the Circle dataset

```
Input integer A,B,C,R; // where (A mod C = 0) and (B mod R = 0)
// N = the number of points, the number of clusters (Q) = C*R
// C = Number of columns; R = Number of rows;
// n = number of customers in one cluster
N=A*B; n=(A*B)/(C*R);
If (C<R) Then Set α=(int)(B/R) and β=(int)(A/C)
Else Set α=(int)(A/C) and β=(int)(B/R)
γ=n; Yc=γ; //Yc -> y-coordinate of the centre of the circle
ω=0; ν=0;
for r=1 To R do
begin
   δ=0.75γ; // the radius of one clustered dataset is 0.75γ
   Xc=δ; // Xc -> x-coordinate of the centre of the circle
   for c=1 To C do
   begin
       δ=0.75γ; ρ=δ/α; // the radius decreases by ρ
       for k=1 To α do
       begin
          for i=1 To β do
          begin
             ω = ω + 1
             If ((k=α) And (i=β)) Then
                Set ν=ν+1; x= Xc; y=Yc;S[ν]=ω; // the optimal solution
             Else
                θ=i*2*Π/β;
                Set x=Xc+δ*cos(θ); //x coordinate
                Set y=Yc+δ*sin(θ); //y coordinate
             End If
             X[ω]=(int)(x+0.5); Y[ω]=(int) (y+0.5);//x and y coordinate
          end
          δ=δ-ρ; // decrease the radius
       end
       Xc=Xc+2γ; // x-coordinate of the centre of next cluster
   end
   Yc=Yc+2γ; //y-coordinate of the centre of next cluster
end
```

Figure B.1. The algorithm for generating the Circle dataset

Appendix C. The proof of optimality

Let A and B be the inputs. C and R be the number of columns and rows respectively.
Define N = A * B and Q = C * R as the number of customers and clusters respectively.
Define γ = N/Q as the number of customers in one cluster
Let R_{max} denote the radius of the largest circle (ring) in a cluster where $R_{max} = 0.75\gamma$
(X[c,r] and Y[c,r]) : the coordinate of the centre of the cluster defined by [c,r]
where c = 1,…,C and r = 1,…,R
Let $X[1,1] = R_{max}$ and $Y[1,1] = \gamma$.
This leads to :

$$X[c,r] = X[1,1] + 2(c-1)\gamma \text{ and } Y[c,r] = Y[1,1] + 2(r-1)\gamma \quad \text{(C.1)}$$

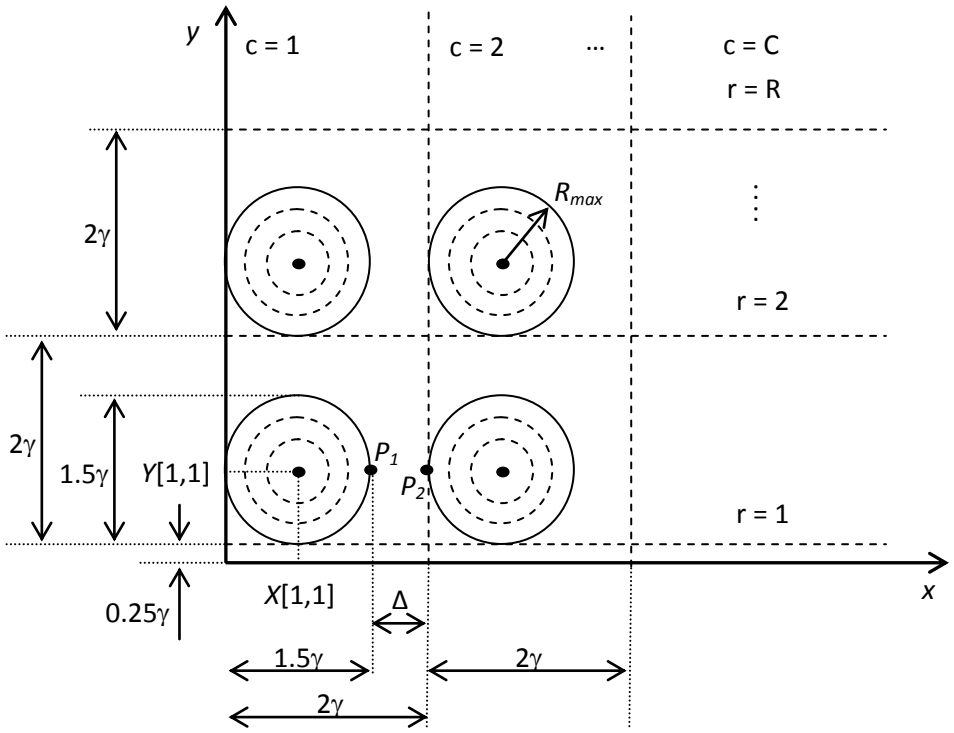


Figure C.1. An illustration of the Circle dataset

We define the number of circles (rings) in one cluster as $K = \begin{cases} A/C, \text{ if } C > R \\ B/R, \text{otherwise} \end{cases}$

and the number of customers in one ring as $s = \begin{cases} B/R, \text{ if } C > R \\ A/C, \text{otherwise} \end{cases}$

The radius of the k^{th} ring of any cluster is constant for all clusters with $r_k = R_{\max} - [(K-k)R_{\max}/K]$

In a given ring, the position of customer i (i=1,…, s) is as follow:

$$x[i] = X[c,r] + r_k \cos\theta \text{ and } y[i] = Y[c,r] + r_k \sin\theta[i] \text{ where } \theta[i] = i2\pi/s,\ i = 0,...,(s-1) \quad \text{(C.2)}$$

The proof of optimality is based on two items:

(i) The *p*-median problem can be reduced to *p* 1-median problems.

Let F_z be the set of points in the z^{th} cluster and $\Delta = \underset{\substack{i\in F_z \\ j\in F_w \\ z\neq w}}{\text{Min}}\, d_{ij} = 0.5\gamma, \forall z = 1,...,Q$.

Let P_1 and P_2 be the two-points where $P_1 \in F_z$ and $P_2 \in F_w$ such that $(P_1, P_2) = Arg \underset{\substack{i\in F_z \\ j\in F_w \\ z\neq w}}{Min}\, d_{ij}$. These are shown in Figure C.1.

Let d_i^z be the distance between point i ($i \in F_z$) and the centre of cluster z.

$$d_{P_1}^{w} = \Delta + 0.75\gamma = 1.25\gamma > d_{P_1}^{z} = 0.75\gamma$$

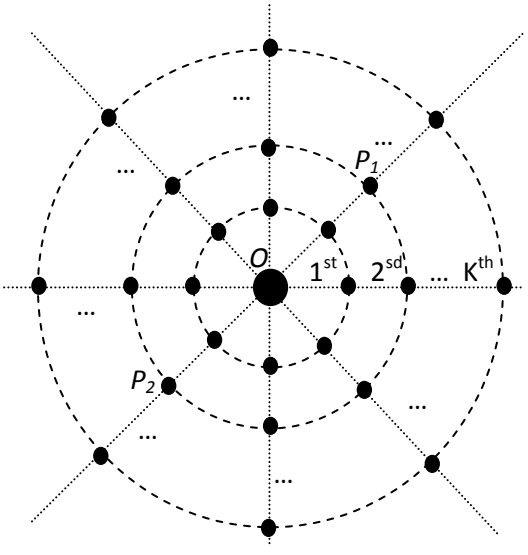
This shows that the furthest point in cluster z remained served by the centre of cluster z. This demonstrates all points in cluster z remained as part of cluster z, $\forall z = 1,\ldots,Q$. Therefore the Q-median problem is reduced to solve Q 1-median problems, with each median representing a centre of a given cluster.

(ii) In each cluster, the optimal point for all circles (rings) needs to be proved to be exactly in the centre defined by (C.1) which is the centre point used for generating the points in the rings.

Let $(\tilde{x}, \tilde{y})$ be the coordinate of the optimal solution (O). To proof that $(\tilde{x}, \tilde{y})$ is the coordinate of the centre of the circle, we need to examine two situations. This is based on whether s is even or odd.

a) s is even.

Given the construction of the points (see Equation C.2), the circle is defined by two critical points (P_1 and P_2) such that $\angle P_1 O P_2 = \pi$. This leads to $(\tilde{x}, \tilde{y})$ as given below:



$$\tilde{x} = \left(x[P_1] + x[P_2]\right)/2 \,;\; \tilde{y} = \left(y[P_1] + y[P_2]\right)/2$$

Let $\theta_0 = 2\pi / s$ and $\theta(i) = i\theta_0, \forall i = 0,...,(s-1)$

$\exists \alpha \; s.t. \theta(P_1) = \alpha\theta_0$ and $\theta(P_2) = \alpha\theta_0 + \pi$

$$x[P_1] = X[r,c] + r_k \cos(\alpha\theta_0)$$

$$x[P_2] = X[r,c] + r_k \cos(\alpha\theta_0 + \pi) = X[r,c] - r_k \cos(\alpha\theta_0)$$

$$\tilde{x} = \left(\left(X[r,c] + r_k \cos(\alpha\theta_0)\right) + \left(X[r,c] - r_k \cos(\alpha\theta_0)\right)\right)/2$$

$$\tilde{x} = \left(2X[r,c]\right)/2 = X[r,c]$$

$$y[P_1] = Y[r,c] + r_k \sin(\alpha\theta_0)$$

$$y[P_2] = Y[r,c] + r_k \sin(\alpha\theta_0 + \pi) = Y[r,c] - r_k \sin(\alpha\theta_0)$$

$$\tilde{y} = \left(\left(Y[r,c] + r_k \sin(\alpha\theta_0)\right) + \left(Y[r,c] - r_k \sin(\alpha\theta_0)\right)\right)/2$$

$$\tilde{y} = \left(2Y[r,c]\right)/2 = Y[r,c]$$

Figure C.2. An example of a cluster in the Circle dataset

b) s is odd. A circle can be defined by three critical points making an acute and isosceles triangle. For example, in the k^{th} ring the three points are P_A, P_B, and P_C with $\angle\left(\overline{P_A O} \text{ and } x-axis\right)=\lambda$, $\angle\left(\overline{P_B O} \text{ and } x-axis\right)=\vartheta$, and $\angle\left(\overline{P_C O} \text{ and } x-axis\right)=\sigma$. The centre of the circle, (O), is the circumcentre of the triangle $P_A P_B P_C$ whose coordinates are as follow:

$$\tilde{x} = \Gamma / \mathrm{T} \tag{C.3}$$

$$\tilde{y} = \Psi / \mathrm{T} \tag{C.4}$$

where

$$\Gamma = \left(\left(x[P_A]^2 + y[P_A]^2\right)\left(y[P_B] - y[P_C]\right) + \left(x[P_B]^2 + y[P_B]^2\right)\left(y[P_C] - y[P_A]\right) + \left(x[P_C]^2 + y[P_C]^2\right)\left(y[P_A] - y[P_B]\right)\right) \tag{C.5}$$

$$\Psi = \left(\left(x[P_A]^2 + y[P_A]^2\right)\left(x[P_C] - x[P_B]\right) + \left(x[P_B]^2 + y[P_B]^2\right)\left(x[P_A] - x[P_C]\right) + \left(x[P_C]^2 + y[P_C]^2\right)\left(x[P_B] - x[P_A]\right)\right) \tag{C.6}$$

$$\mathrm{T} = 2\left(x[P_A]\left(y[P_B] - y[P_C]\right) + x[P_B]\left(y[P_C] - y[P_A]\right) + x[P_C]\left(y[P_A] - y[P_B]\right)\right) \tag{C.7}$$

The coordinate of the points:

$$x[P_A] = X[r,c] + r_k \cos(\lambda) \text{ and } y[P_A] = Y[r,c] + r_k \sin(\lambda) \tag{C.8}$$

$$x[P_B] = X[r,c] + r_k \cos(\vartheta) \text{ and } y[P_A] = Y[r,c] + r_k \sin(\vartheta) \tag{C.9}$$

$$x[P_A] = X[r,c] + r_k \cos(\sigma) \text{ and } y[P_A] = Y[r,c] + r_k \sin(\sigma) \tag{C.10}$$

Substitute (C.8), (C.9), and (C.10) into (C.7)

$$\mathrm{T} = 2r_k^2\left(\cos\lambda\sin\vartheta - \cos\lambda\sin\sigma + \cos\vartheta\sin\sigma - \cos\vartheta\sin\lambda + \cos\sigma\sin\lambda - \cos\sigma\sin\vartheta\right) \tag{C.11}$$

Substitute (C.8), (C.9), and (C.10) into (C.5)

$$\Gamma = X[c,r]2r_k^2\left(\cos\lambda\sin\vartheta - \cos\lambda\sin\sigma + \cos\vartheta\sin\sigma - \cos\vartheta\sin\lambda + \cos\sigma\sin\lambda - \cos\sigma\sin\vartheta\right) \tag{C.12}$$

Substitute (C.8), (C.9), and (C.10) into (C.6)

$$\Psi = Y[c,r]2r_k^2\left(\cos\lambda\sin\vartheta - \cos\lambda\sin\sigma + \cos\vartheta\sin\sigma - \cos\vartheta\sin\lambda + \cos\sigma\sin\lambda - \cos\sigma\sin\vartheta\right) \tag{C.13}$$

Substitute (C.12) and (C.11) into (C.3)

$$\tilde{x} = X[c,r]$$

Substitute (C.13) and (C.11) into (C.4)

$$\tilde{y} = Y[c,r]$$

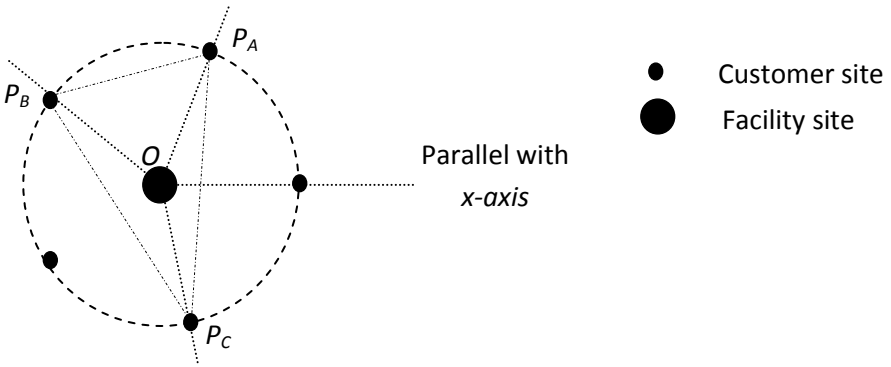


Figure C.3. The centre of the three critical points P_A, P_B, and P_C (s = 5)