Random Search Algorithm for the p-Median Problem

Alexander N. Antamoshkin and Lev A. Kazakovtsev
Siberian State Aerospace University
prosp. Krasnoyarskiy rabochiy, 31, Krasnoyarsk 660014, Russian Federation
E-mail: levk@ieee.org

Keywords: continuous location problem, Weber problem, random search, genetic algorithms, discrete optimization

Received: November 5, 2012

Authors investigate the p-median location problem on networks and propose a heuristic algorithm which is based on the probability changing method (a special case of the genetic algorithm) for an approximate solution to the problem. The ideas of the algorithm are proposed under the assumption that, in the large-scale networks with comparatively small edge lengths, the p-median problem has features similar to the Weber problem. The efficiency of the proposed algorithm and its combinations with the known algorithms were proved by experiments.

Povzetek: Avtorji predlagajo nov hevristični algoritem za iskanje p-mediane v omrežjih.

1 Introduction

The aim of the location problem [13] is to determine the location of one or more new facilities in a set of possible locations (discrete or continuous). The main parameters of such problems are the coordinates of the facilities and distances between them [37, 14, 15]. Examples of the location problems include the location of warehouses [15], computer and communication networks, base stations of wireless networks [30] etc. They are also useful in the approximation theory, statistical estimation problem [28], signal and image processing and other engineering applications.

The Fermat-Weber problem (Weber problem) [35, 37] is the problem of searching for such a point that a sum of weighted distances from this point to the given points (demand points) is minimal. In the location theory, several generalizations of these problems are known [11]. The first one is the multi-Weber problem where the aim is to find optimal locations of p new facilities:

$$F(X_1, ..., X_p) = \sum_{i=1}^{N} w_i \min_{j \in \{\overline{1,p}\}} L(X_j, A_i) \to min. \quad (1)$$

Here, $\{A_i | i = \overline{1, N}\}$ is a set of the demand points, $\{X_j | j = \overline{1, p}\}$ is a set of new placed facilities, w_i is a weight coefficient of the ith demand point, $L()$ is a distance function.

One of the problems of the discrete location theory, a p-median problem, can also be considered as a generalization of the Weber problem [17, 23]. The medians are searched for in a finite set of graph vertices. Generally this problem is $\mathcal{NP}$-hard [24]. The polynomial-time algorithm is developed for trees only [20]. A procedure for network flow searching (an algorithm for the p-median problem solving) is adapted for location problems in a continuous space with the rectangular metric [10].

Despite the complexity of the problem, various heuristic algorithms could give good results for most problems in reasonable time. One of the simplest but efficient heuristics for the p-median problem is local search [31, 32]. Rabbani [29] proposes an algorithm based on new graph theory for small size problems. Using Lagrangian relaxation allows an approximate solving of huge-scale problems [5], up to 90000 vertices in a network. However, "good" solutions [6] were achieved by the analogous technique for problems with $n = 3795$ which were also considered as large-scale problems.

Hosage and Goodchild [19] proposed the first genetic algorithm for the p-median problem. In [9], authors propose a genetic algorithm providing rather precise results but its convergence is slow. In [1], authors propose a quick and precise genetic algorithm for the p-median problem. However, solving large-scale problems still takes too much time.

In [14], a continuous problem with an arbitrary l_p metric is solved. In [24] and [25], authors prove that the unconstrained Weber problems with Euclidean or rectangular metric are $\mathcal{NP}$-complete.

For the continuous location problems with Euclidean (l_2), rectangular (l_1) and Chebyshev (l_∞) metrics, the algorithms based on the Weiszfeld procedure are proposed [36]. However, a solution of the same problems with restricted zones [39], barriers [8] or modified distance functions is not trivial. In practically important problems, models based on the Euclidean or rectangular metric are usually rough approximations [27, 38] since they do not take into account characteristics of the space and transportation means, in particular, the presence and quality of roads, barriers, relief etc. Sometimes, the distance function is given algorithmically as a solution of another optimization problem [38]. Thus, if the problem is formulated as a Weber

problem, its discretization should be often useful [21].

We can always consider practical problems as discrete. Any scheme (or map) has finite resolution, digital copy of any map is always discrete. The implementation of the obtained solution of a problem is also performed by the tools with finite precision. However, such a discrete problem is a large-scale problem and any algorithms which guarantee an exact solution in polynomial time do not exist (polynomial time algorithm exists for a discretized generalized Weber problem with rectangular metric only [10]).

The unconstrained location problems with mixed coordinate systems (discrete and continuous) are considered in [33, 16].

Heuristic random search algorithms do not guarantee an exact solution. However, they are statistically optimal, i.e., the percentage of the "near optimal" solutions increases with growth of the problems dimension [3]. In case of discrete location problems, genetic algorithms [30, 26], greedy search [26, 18] and other methods are implemented.

The probability changing method initially proposed for unconstrained optimization is a random search method described by the pseudo-code below.

Algorithm 1. *Basic probability changing method*

1: Set $k = 0$; set the initial values of the components of the probability vector $P_0 = \{p_{0,1}, p_{0,2}, ..., p_{0,N}\}$ (here, the value of each element at the kth step is the probability (expectation) of generating a vector X with the corresponging element equal to 1: $p_{k,j} = \mu\{x_j = 1\}$);

2: In accordance with the distribution given by the elements of the vector P, generate a set of N_{POP} vectors $X_{k,i}$, $i \in \{\overline{1, N_{POP}}\}$;

3: Calculate the value of the objective function $F(X_{k,i})$ $\forall i \in \{\overline{1, N_{POP}}\}$.

4: Select some vectors $X_{k,i}$ (for example, a vector with the best and the worst value of the objective function);

5: Based on the results of the Step 4, modify the probability vector P;

6: $k = k + 1$; if $k < N_{STEPS}$ then goto 2 (other stop conditions are also possible);

7: STOP.

This simple method is a special variant of the genetic algorithm [12]. The modifications of this algorithm for the constrained optimization problems proposed in [22] can solve pseudo-Boolean problems (multi-knapsack problem, travelling salesman problem) with dimensions up to millions of Boolean variables.

2 Problem statement, known algorithms

Let $G = (V, E)$ be an undirected adjacent graph (a network), $V = \{v_1, ..., v_n\}$ be a set of its vertices (Fig. 1), $E = \{e_i | i = \overline{1, m}\}$ be a set of its edges, $e_i = (v_j, v_k)$, $j \in \{\overline{1, n}\}$, $k \in \{\overline{1, n}\}$, $i \in \{\overline{1, m}\}$ without loops $(e_i \neq (v_j, v_j)\ \forall i = \overline{1, m}, j = \overline{1, n})$. For each edge e_i, its length l_i is defined, $l_i \geq 0 \forall i = \overline{1, m}$. For an edge $e_i = (v_j, v_k)$, let us denote $l_{j,k} = l_i$. Weight $w_j \geq 0$ is defined for each vertex v_j. For each pair of the vertices (v_j, v_k), a distance function $L(j, k)$ is defined as the length of the shordest path from v_i to v_j.

$$L(j,k) = \sum_{q \in P^*_{j,k}} l_q = \min_{P \in P_{j,k}} \sum_{q \in P} l_q \tag{2}$$

Here, $P^*_{j,k}$ is a set of the edges of the shortest path between v_j and v_k, $P_{j,k}$ is a set of all possible paths between these vertices. We can formulate the the p-median problem as

$$\begin{aligned} \arg \quad & \min_{m_1,...,m_p \in \{\overline{1,n}\}} f(m_1, ..., m_p) \\ & = \arg \min_{m_1,...m_p \in \overline{1,n}} \sum_{i=1}^{n} w_i \min_{i=\overline{1,p}} L(m_j, i), \\ & p < n. \end{aligned} \tag{3}$$

The aim of this problem is to select p vertices so that the sum of weighted distances from each of the vertices of the graph to the nearest selected vertex is minimal.

Let

$$S_i = \{j | \exists e_j = (v_i, v_k), j \in \{\overline{1, m}\}, k \in \{\overline{1, n}\}\}$$

be a set of the edges of the vertices incident to the ith vertex;

$$C_i = \{k | \exists e_j = (c_i, v_k), j \in \{\overline{1, m}\}, k \in \{\overline{1, n}\}\}$$

be a set of the indexes of the vertices adjacent to the ith vertex;

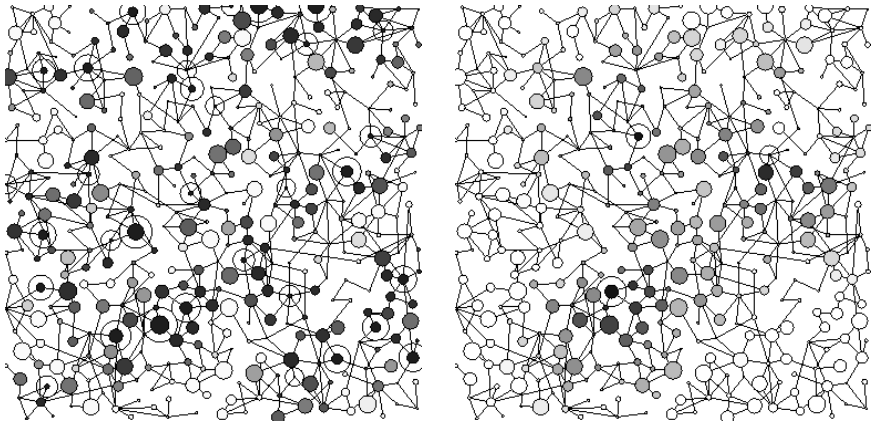
$$l^*_i = \min_{j \in \{\overline{1,p}\}} L(m_j, i)$$

be the length of the shortest path from the ith vertex to the nearest of p vertices $m_1, ..., m_p$.

For calculating the value of the objective function $f(m_1, ..., m_p)$, we use the algorithm below.

Algorithm 2. *p-median objective function calculation*

Require: indexes $m_1, ..., m_p$ of p selected vertices. **1:** $l^*_i = +\infty \forall i = \overline{1, n}$;
2: $l^*_{m_i} = 0 \forall i = \overline{1, p}$;
3: $V^* = \{m_1, ..., m_p\}$; $V^{**} = V^*$;
4: while $|V^{**}| \neq 0$ do
4.1: $V' = \emptyset$;
4.2: for $i \in V^{**}$ do
4.2.1: for $j \in C_i$ do
4.2.1.1: if $v_j \notin V^*$ then $V' = V' \cup \{j\}$; $l^*_j = l^*_i + l_{i,j}$;
4.2.1.2: else if $l^*_j = l^*_i + l_{i,j}$ then $l^*_j = l^*_l + l_{i,j}$;
4.2.1.3: next 4.2.1;
4.2.2: next 4.2;
4.3: $V^{**} = V'$; $V^* = V^* \cup V'$;
4.4: next 4;
5: return $f(m_1, ..., m_p) = \sum_{i=1}^{n} w_i l^*_i$.



a) solution for $p = 32$ b) solution for $p = 3$

Figure 1: Scheme of a problem and its solutions, $n = 400$

For comparison, we used the local search algorithm [32] with a random order of vertices evaluation (Algorithm 3) as one of the simplest but efficient algorithms.

Algorithm 3. *Local search*

Require: array of indexes $\mathcal{M} = \{m_1, ..., m_p\}$ of the vertices (initial solution), value of the objective function $f^* = f(m_1, ..., m_p)$.
1: shuffle elements of $\mathcal{M}$; $r = 0$;
2: for each element m of the array $\mathcal{M}$ do
2.1: for each vertex m^* which is adjacent to m do
2.1.1: $f^{**} = f(m_1, ..., m^*, ..., m_p)$ (here, the vertex m is replaced by m^*);
2.1.2: if $f^{**} < f^*$ then replace m by m^* in $\mathcal{M}$; $f^* = f^{**}$; $r = 1$;
2.1.3: next 2.1;
2.2: next 2;
3: if $r = 1$ then goto 1;
5: return new solution $(m_1, ..., m_p)$.

The genetic algorithm with greedy heuristic proposed in [1] includes a special crossover procedure (Algorithm 4). The "chromosomes" of this algorithm are sets of the vertices (solutions of the problem).

Algorithm 4. *Crossover procedure for the GA with greedy heuristic*

Require: sets of indexes $\mathcal{M}_1 = \{m_{1,1}, ..., m_{1,p}\}$, $\mathcal{M}_2 = \{m_{2,1}, ..., m_{2,p}\}$ of the vertices ("chromosomes").
1: $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2$; $p_{\mathcal{M}} = |\mathcal{M}|$;
2: while $p_{\mathcal{M}} > p$ do
2.1: $f* = +\infty$;
2.2: for each vertex m^* in $\mathcal{M}$ do
2.2.1: $\mathcal{M}^* = \mathcal{M} \setminus \{m^*\}$;
2.2.2: $f** = f(\mathcal{M}*)$;
2.2.2: if $f^{**} < f^*$ then $m^{**} = m^*$;
2.1.3: next 2.2;
2.3: $\mathcal{M} = \mathcal{M} \setminus \{m^{**}\}$; $p_{\mathcal{M}} = p_{\mathcal{M}} - 1$;
2.3: next 2;
5: return new solution ("chromosome") $\mathcal{M}$.

This method uses an original procedure of the initial population generation [1]. It does not use any mutation procedure.

The probability changing method is a pseudo-Boolean optimization method, the objective function must be a function of Boolean variables.

Let us introduce new variables $x_1, ..., x_n$:

$$x_i = \begin{cases} 1, & i \in \{m_1, ..., m_p\} \\ 0, & i \notin \{m_1, ..., m_p\} \end{cases}. \tag{4}$$

In this case, our problem has a constraint

$$\sum_{i=1}^{n} x_i = p. \tag{5}$$

The transformation of the problem back into the problem with integer variables is performed as follows:

$$\{m_1, ..., m_p\} = \{i | x_i = 1, i = \overline{1, n}\}. \tag{6}$$

The problem with the pseudo-Boolean objective function is stated as follows:

$$\begin{aligned} f_b(x_1, ..., x_n) &= f(\{j | x_j = 1, j = \overline{1, n}\}) \\ &= \sum_{i=1}^{n} w_i \min_{j | x_j = 1, j = \overline{1,n}} L(i, j) \end{aligned} \tag{7}$$

with the constraint (5).

In this form, our problem can be solved using many methods [3, 4, 2, 22] including the probability changing method.

3 An algorithm based on the probability changing method

Continuous location problems such as multi-Weber problem with Euclidean, rectangular or Chebyshev meric are

amenable to analysis and analytical solution methods. Weiszfeld [36] procedure and Trubin's procedure for rectangular metric [34] are based on the assumption that the partial derivatives of the objective function are defined, i.e., a small change of the location of any point in a feasible solution leads to some small change in the value of the objective function. If $X = (x_1, x_2)$ is a solution of some $2D$ unconstrained location problem then

$$\frac{\Delta f(X)}{\Delta X} = \frac{\Delta f(x_1, x_2)}{\Delta\sqrt{\Delta x_1 + \Delta x_2}} < const.$$

Let L_{max} be the maximum distance between 2 vertices:

$$L_{max} = \max_{i,j\in\{\overline{1,n}\}} L(i,j), \tag{8}$$

l_{avg} be the average length of the edge:

$$l_{avg} = \sum_{j=1}^{m} l_j/m. \tag{9}$$

Our algorithm is based on 2 hypotheses:

Hypothesis 1. *If l_{avg}/L_{max} is small ($l_{avg}/L_{max} \to 0$) then the character of the p-median problem shows features of the continuous problem. In particular, if $\{m_1, ..., m_p\}$ is a solution of the p-median problem then, having replaced any m_ith vertex of this solution to any jth point such that $L(j, m_i)$ is small, the change in the objective function value is also small:*

$$\exists l_\Delta, \Delta_{Fmax} > 0 : (j, m_i) < l_\Delta \Rightarrow$$
$$|f(m_1, ..., m_i, ..., m_p) - f(m_1, ..., j, ..., m_p)|$$
$$< \Delta_{Fmax}$$

Hypothesis 2. *If the solution $\{m_1, ..., m_i, ..., m_j, ..., m_p\}$ contains 2 vertices v_i and v_j such that $L(m_i, m_j)$ is small then there is high probability (expectation) that for the solution $\{m_1, ..., m_i, ..., k, ..., m_p\}$ with a vertex v_j replaced by another arbitrary chosen vertex v_k, the value of the objective function is "better" than for the original solution:*

$$\exists l_{min} :$$
$$L(m_i, m_j) < l_{min} \wedge L(m_i, k) > l_{min} \Rightarrow$$
$$\Rightarrow \mu \left\{ \begin{array}{l} f(m_1, ..., m_i, ..., m_j, ..., m_p) \\ \geq f(m_1, ..., m_i, ..., k, ..., m_p) \end{array} \right\} > 0.5$$

and

$$\lim_{l_{min}\to 0} \mu \left\{ \begin{array}{l} f(m_1, ...m_i, ...m_j, ...m_p) \\ \geq f(m_1, ...m_i, ...k, ...m_p) \end{array} \right\} \to 1.$$

Let us prove the consistency of the hypotheses for the special cases.

Lemma 1. *Let $L(m_i, j) = 0$. Then*

$$f(m_1, ..., m_i, ..., m_p) = f(m_1, ..., j, ..., m_p).$$

Proof. Let us choose arbitrarily the i^*th vertex, $i^* \in \{\overline{1,n}\}$.

If $i^* \in \{m_1, ..., m_i, ..., m_p\}$ then, obviously, $\min_{i''\in\{m_1,...m_i,...m_p\}} L(i^*, i'') = 0$.

$$\min_{i''\in\{m_1,...,j,...,m_p\}} L(i^*, i'')$$
$$= \min \left\{ \min_{i''\in\{m_1,...m_i,...m_p\}} L(i^*, i'') + L(m_i, j); \right.$$
$$\left. L(i^*, j) \right\}$$
$$= \min\{0 + 0; L(i^*, j)\} = 0.$$

If $i^* \notin \{m_1, ..., m_i, ..., m_p\}$, let us introduce the notation:
$P_{i^*,j}$ is a set of all possible paths from the i^*th vertex to the jth one,
P_{i^*,m_i} is a set of all possible paths from the i^*th vertex to the m_ith,
$P_{i^*(m_i)j}$ is a set of all possible paths from the i^*th vertex to the jth one through the m_ith vertex,
$P_{i^*(\overline{m_i})j}$ is a set of all possible paths from the i^*th vertex to the jth one which do not include the m_ith vertex.

$$L(i^*, j) = \min_{P\in P_{i^*,j}} \sum_{e_k\in P} l_k$$
$$= \min\{\min_{P\in P_{i^*(m_i)j}} \sum_{e_k\in P} l_k; \min_{P\in P_{i^*(\overline{m_i})j}} \sum_{e_k inP} l_k\}$$
$$= \min\{ \min_{P\in P_{i^*,m_i}} \sum_{e_k\in P} l_k + \min_{P\in P_{m_i,j}} \sum_{e_k\in P} l_k;$$
$$\min_{P\in P_{i^*(\overline{m_i})j}} \sum_{e_k\in P} l_k \}$$
$$= \min\{\min_{P\in P_{i^*,m_i}} \sum_{e_k\in P} l_k + 0; \min_{P\in P_{i^*(\overline{m_i})j}} \sum_{e_k inP} l_k\}$$
$$= \min\{L(i^*, m_i); \min_{P\in P_{i^*(\overline{m_i})j}} \sum_{e_k inP} l_k\}$$
$$\leq L(i^*, m_i).$$

$$L(i^*, m_i) = \min_{P\in P_{i^*,m_i}} \sum_{e_k\in P} l_k$$
$$= \min\{\min_{P\in P_{i^*(j)m_i}} \sum_{e_k\in P} l_k; \min_{P\in P_{i^*(\overline{j})m_i}} \sum_{e_k inP} l_k\}$$
$$= \min\{ \min_{P\in P_{i^*,j}} \sum_{e_k\in P} l_k + \min_{P\in P_{j,m_i}} \sum_{e_k\in P} l_k;$$
$$\min_{P\in P_{i^*(\overline{j})m_i}} \sum_{e_k\in P} l_k \}$$
$$= \min\{\min_{P\in P_{i^*,j}} \sum_{e_k\in P} l_k + 0; \min_{P\in P_{i^*(\overline{j})m_i}} \sum_{e_k inP} l_k\}$$
$$= \min\{L(i^*, j); \min_{P\in P_{i^*(\overline{j})m_i}} \sum_{e_k inP} l_k\} \leq L(i^*, j).$$

$$\begin{array}{lll} L(i^*, m_i) & \leq L(i^*, j) \wedge L(i^*, j) \leq L(i^*, m_i) & \\ \Rightarrow & L(i^*, m_i) = L(i^*, j). & (10) \end{array}$$

□

Lemma 2. *Let $\{m_1, ..., m_i, ..., m_j, ..., m_p\}$ be a solution of the p-median location problem on a network with n vertices, $w_i > 0 \forall i = \overline{1,n}$, $L(m_i, m_j) = 0$ and $\exists k \in \{\overline{1,n}\}$: $L(m_q, k) > 0 \forall q = \overline{1,p}$.*

Then

$$f \quad (m_1, ..., m_i, ..., m_j, ..., m_p) > f(m_1, ..., m_i, ..., k, ..., m_p). \qquad (11)$$

Proof. Let us introduce the notation

$$M_0 = \{m_1, ..., m_i, ..., m_j, ..., m_p\};$$

$$M_1 = \{m_1, ..., m_i, ..., k, ..., m_p\}.$$

Let us define a function

$$f_m(i', S) = \min_{i'' \in S} L(i'', i').$$

Its value for the sets denoted above is

$$f_m \quad (i', M_0) = \min\{ f_m(i', \{m_i\}); f_m(i', \{m_j\}); f_m(i', M_0 \setminus \{m_i\}) \} \ \forall i' \in \{\overline{1,n}\}.$$

Taking into account $L(m_i, m_j) = 0$, from Lemma 1, for the set of vertices of the solution

$$\begin{aligned} f_m \quad & (i', \{m_1, ..., m_i, ..., m_j, ..., m_p\}) \\ = \quad & f_m(i', \{m_1, ..., m_i, ..., m_i, ..., m_p\}) \\ = \quad & f_m(i', M_0) \ \ \forall i' \in \{\overline{1,n}\}. \end{aligned}$$

Further,

$$\begin{aligned} & f_m(i', M_1) \\ & = \min \quad \{ f_m(i', \{m_i\}); \ f_m(i', \{k\}); \\ & \qquad f_m(i', M_1 \setminus (\{m_i\} \cup \{k\})) \} \\ & \qquad \forall i' \in \{\overline{1,n}\}; \end{aligned}$$

$$\begin{aligned} f_m \quad & (i', M_0) \\ = \quad & \min\{f_m(i', \{m_i\}); f_m(i', M_0 \setminus \{m_i\})\} \\ = \quad & \min\{f_m(i', \{m_i\}); f_m((M_1 \setminus \{k\}) \setminus \{m_i\})\} \\ = \quad & \min\{f_m(i', \{m_i\}); f_m(M_1 \setminus (\{k\} \cup \{m_i\}))\} \\ \geq \quad & \min\{ f_m(i', \{m_i\}); \ f_m(i', \{k\}); \\ & \qquad f_m(i', M_1 \setminus (\{m_i\} \cup \{k\})) \ \} \\ = \quad & f_m(i', M_1) \ \ \forall i' \in \{\overline{1,n}\}; \end{aligned}$$

Thus,

$$\begin{aligned} f \quad & (m_1, ..., m_i, ..., k, ..., m_p) \\ = \quad & \sum_{i'=1}^{n} f_m(i', M_1) \leq \sum_{i'=1}^{n} f_m(i', M_0) \\ = \quad & f(m_1, ..., m_i, ..., m_j, ..., m_p). \end{aligned}$$

□

In our version of Algorithm 1, steps 2 and 5 are modified. At Step 2 (generation of the samples of vectors X), the constraint (5) must be taken into consideration. The solutions generated by algorithm below are always feasible.

Algorithm 5. *Step 2 of Algorithm 1*

Require: Probability vector $P = (p_1, ..., p_n)$.
1: $\chi = \emptyset$;
2: for each $i \in \{\overline{1,p}\}$ do
2.1: $r = Random() \cdot \sum_{j=1}^{n} p_i$; $S = 0$;
2.2: for each $j \in \{\overline{1,n}\}$ do
2.2.1: $S = S + p_i$;
2.2.2: if $S \geq r$ then $\chi = \chi \cup \{j\}$; goto 2.3;
2.2.3: next 2.2;
2.3: next 2;
3: return χ.

The result of this algorithm is a set χ. The corresponding vector X of boolean variables can be calculated in accordance with (4). Here, $Random()$ is a generator of the random value with continuous uniform distribution ($Random() \sim U[0, 1)$).

Let L_0 be the maximum distance considered as "small" in terms of Hypothesis 1 and Hypothesis 2. In accordance with Hypothesis 2,

$$\begin{aligned} \mu \quad & \{\exists m_i, m_j \in \chi : L(m_i, m_j) < L_0\} \\ < \quad & \mu\{L(m_i, m_j) \geq L_0 \forall m_i, m_j \in \chi\}; \end{aligned}$$

$$\lim_{L^* \to 0} \mu\{\exists m_i, m_j \in \chi : L(m_i, m_j) < L^*\} = 0.$$

Let us modify Algorithm 5.

Algorithm 6. *Step 2 of Algorithm 1, v. 2*

Require: Probability vector $P = (p_1, ..., p_n)$.
1: $\chi = \emptyset$;
2: for each $i \in \{\overline{1,p}\}$ do
2.1: $P^* = P$;
2.2: $r = Random() \cdot \sum_{j=1}^{n} p_i^*$; $S = 0$;
2.3: for each $j \in \{\overline{1,n}\}$ do
2.3.1: $S = S + p_i^*$;
2.3.2: if $S \geq r$ then $\chi = \chi \cup \{j\}$; $j' = j$; goto 2.3;
2.3.3: next 2.3;
2.4: for each $j \in \{k | k \in \{\overline{1,n}\} \wedge L(j', k) < L_0\}$ do: $p_j^* = p_j^* \cdot L(j, j') \cdot L_0$; next 2.4;
2.5: next 2;
3: return χ.

Step 5 of he Algorithm 1 (adaptation of the probability vector P) in its kth iteration must be performed in accordance with Hypothesis 2. We use the multiplicative adaptation.

$$P_{k,i} = p_{(k-1),i} \cdot d_{k,i}^b / d_{k,i}^w, \qquad (12)$$

$$d^b_{k,i} = \begin{cases} 1 + \frac{L_0}{1+L(i^b(i),i)}, & L(i,i^b(i)) < L_0 \\ 1, & L(i,i^b(i)) \geq L_0 \end{cases}, \quad (13)$$

$$d^w_{k,i} = \begin{cases} 1 + \frac{L_0}{1+L(i^w(i),i)}, & L(i,i^w(i)) < L_0 \\ 1, & L(i,i^w(i)) \geq L_0 \end{cases}. \quad (14)$$

Here,

$$i^b(i) = \arg_{i'} \min_{i' \in \chi^b} L(i,i'), \quad (15)$$

$$i^w(i) = \arg_{i'} \min_{i' \in \chi^w} L(i,i'), \quad (16)$$

χ^b and χ^w are the best and the worst samples of the sets of vertex indexes χ generated by Algorithm 2 or Algorithm 5. In the simplest case,

$$\chi^b = \arg_{\chi'} \min_{\chi' \in \mathcal{X}} f(\chi'), \quad (17)$$

$$\chi^b = \arg_{\chi'} \max_{\chi' \in \mathcal{X}_k} f(\chi'). \quad (18)$$

Here, $\mathcal{X}_k$ is a set of all samples of the vector χ at the kth iteration of Algorithm 1.

Note that the discretized Weber problem described in [21] is a special case of the p-median problem considered in this paper.

4 First results, adjusting parameters

For testing purposes, we used the p-median problems generated automatically by the special Algorithm 7.

Algorithm 7. *Sample problem generating*

Require: n.

1: for i in $\{\overline{1,n}\}$ do

1.1: $c^i_x = Random() \cdot 500$; $c^i_y = Random() \cdot 500$; $w_i = 1 + 9Random()$;

1.2: if $\exists j \in \{\overline{j, n-1}\}$: $\sqrt{(c^i_x + c^j_x)^2 + (c^i_y + c^j_y)^2} < 10$ then goto 1.1;

1.3: next 1;

2 Fill the adjacency matrix A with the zero values; $E = \emptyset$;

3: for i in $\{\overline{1,n}\}$ do

3.1:

$$D_i = \begin{cases} 1, & i \in \{\overline{[0.6n+1], n}\}, \\ 2, & i \in \{\overline{[0.4n]+1, [0.6n]}\}, \\ 3, & i \in \{\overline{[0.25n]+1, [0.4n]}\}, \\ 4 + [Random() \cdot 4], & i \leq [0.25n]. \end{cases}$$

3.2: for d in $\{\overline{1, \sum_{j=1}^{n} A_{i,j}}\}$;

3.2.1:

$j = \arg\min_{j \in \{\overline{1,n}\}, A_{i,j}=0} \sqrt{(c^i_x - c^j_x)^2 + c^i_y - c^j_y)^2}$;

$A_{i,j} = 1$; $A_{j,i} = 1$; $E = E \cup \{(i,j)\}$; $l_{i,j} = \sqrt{(c^i_x - c^j_x)^2 + c^i_y - c^j_y)^2}$;

3.2.2: next 3.2;

4: return adjacency matrix A, edges set E, edge lengths $\{l_{i,j}\} \forall (i,j) \in E$ and weights $w_i \forall i \in \{\overline{1,n}\}$.

Scheme of of such problem example is shown in Fig. 1. The lengths of the edges are proportional to ones shown in the scheme, the diameters of the vertices show their weights. In addition, in Fig. 1, the solutions calculated by our algorithm for $p = 32$ and $p = 3$ are shown. The vertices selected as the solution are shown in a circle. For each of the vertices, the color of the vertex shows the distance to the nearest selected vertex. The nearest vertices are shown in dark, the farthest are shown in white.

For our experiments, we used a computer Depo X8Sti (6-core CPU Xeon X5650 2.67 GHz, 12Gb RAM), hyperthreading disabled and ifort compiler with full optimization and implicit parallelism (option -O3). Comparison of the results reached with this hardware configuration with the results of the small system with 2-core Atom CPU N2701.6GHz, 1Gb RAM are shown in Fig. 2 (a combined method "probability changing+GA" is explained in Section 6).

Fig. 3 illustrates change in the probability vector for $p = 3$. The vertices with high value of the expectation of being included in the generated solutions are shown in white, the vertices with the smallest value of the expectation are shown in black.

The diameter L_0 of the consistency area of Hypothesis 1 and Hypothesis 2 is an important parameter of the algorithm. For the problem with $p = 12$, the comparison of the algorithm efficiency with various values of L_0 is shown in Fig. 4. The results of running the algorithm with use of Algorithm 5 as the generating procedure is shown as "L0=0". The best results are achieved with $L_o = 90$. The optimal value of L_0 depends on p. With $p = 32$, the best results are achieved with $L_0 = 60$. Nevertheless, the algorithm with a wide variety of the parameter values $L_0 \in [10, 350]$ gives the results better than the "classical" probability changing method (the case $L_0 = 0$).

For the further experiments, we used $L_0 = L_{avg}/3$ where L_{avg} is the expectation of the average distance to the closest facility in a randomly generated solution (arbitrary chosen set of p vertices):

$$L_{avg} = \mu\{\sum_{i=1}^{n} \min_{j \in \{\overline{1,p}\}} L(i,j)/n\}. \quad (19)$$

As the estimation the value L_{avg}, we used the average distance from 10 randomly chosen vertices to the closest vertex from 10 randomly generated sets of p vertices.

The calculation of L_{max} used in the conditions of the Hypotheses 1 and 2 takes significant time. Instead, we used $l_{avg}/L_{avg} \to 0$ (19) as the condition of applicability of our algorithm.

5 Comparison with the existing methods

For testing purposes, we used the local search method (Algorithm 3) with multistart from randomly generated initial

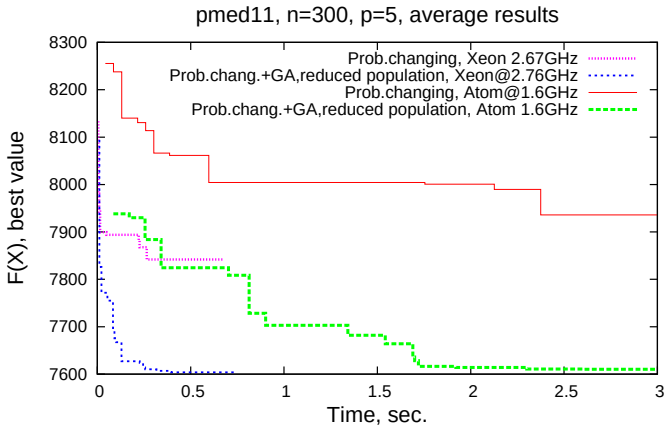
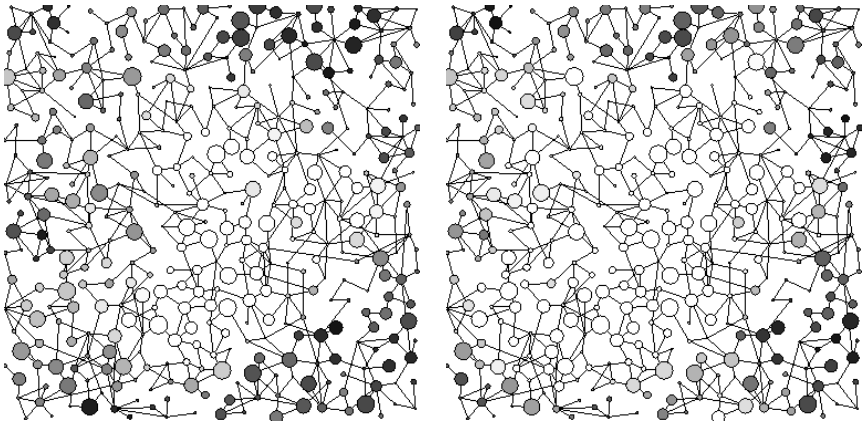


Figure 2: Comparison of the productivity on a medium (Xeon CPU) and small system (Atom CPU)



a) 3rd step b) 6th step

Figure 3: Probability values change $L_0 = 100$, $p = 3$

solution as one of the simplest methods and the genetic algorithm (GA) [1] with the crossover procedure given by Algorithm 4 (greedy heuristic) as one of the most efficient methods . As the testbed, we used the p-median problems from the OR Library [7]. The same library was used in [1]. Since this library contains problems with numbers of vertices up to $n = 900$, we used our Algorithm 7 for generating larger problems.

The results of our algorithm based on the probability changing method used standalone show its low convergence in comparison with the GA (Fig. 5). Problems "pmed22" and "pmed39" are included in the OR Library, a problem with $n = 5000$ was generated by Algorithm 7. This figure shows the average values for 10 runs and the worst result. The results of the combined method ("Probability changing+GA") are explained in the next section. To calculate the quantity of exemplars of the generated solutions in each population N_{POP}, we used formula

$$N_{POP} = \lceil \frac{\sqrt{n}C\left(\begin{array}{c} n \\ p \end{array}\right)}{100\lceil n/p \rceil} \rceil \lceil n/p \rceil. \quad (20)$$

The GA with greedy heuristic uses formula

$$N_{POP} = \lceil \frac{nC\left(\begin{array}{c} n \\ p \end{array}\right)}{100\lceil n/p \rceil} \rceil \lceil n/p \rceil. \quad (21)$$

6 Results of combined methods

The genetic algorithm [1] uses the regular method of filling of the initial population. In case of large-scale problems (pmed39, pmed32, generated problems with $n = 2000$, $n = 5000$), experiments with the randomly filled initial population decrease the accuracy of the results and the convergence insignificantly.

Our experiments with using the last generated population of the probability changing method as the initial population of the GA with greedy heuristic show significant speed-up of such combined algorithm in comparison with the original GA. We used two variants of the population size, standard population (21) and reduced population (20). Both variants show significant speed-up for most problems.

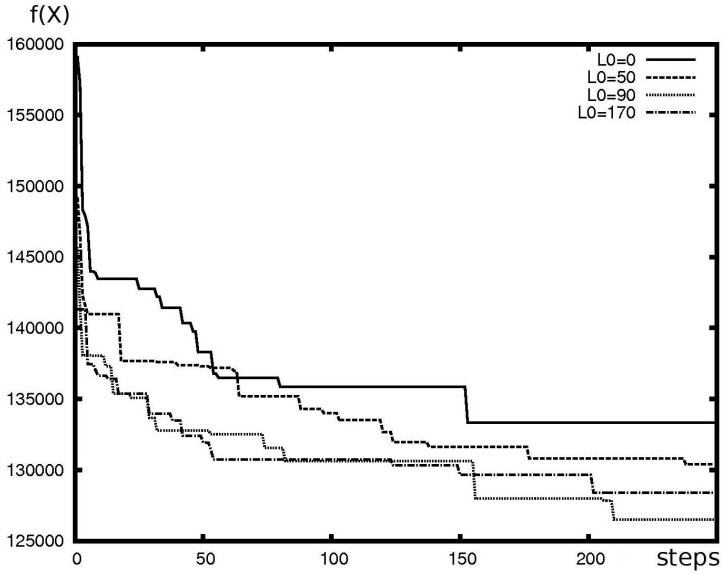


Figure 4: Comparison of the efficiency of the algorithm with various values L_0, $p = 12$

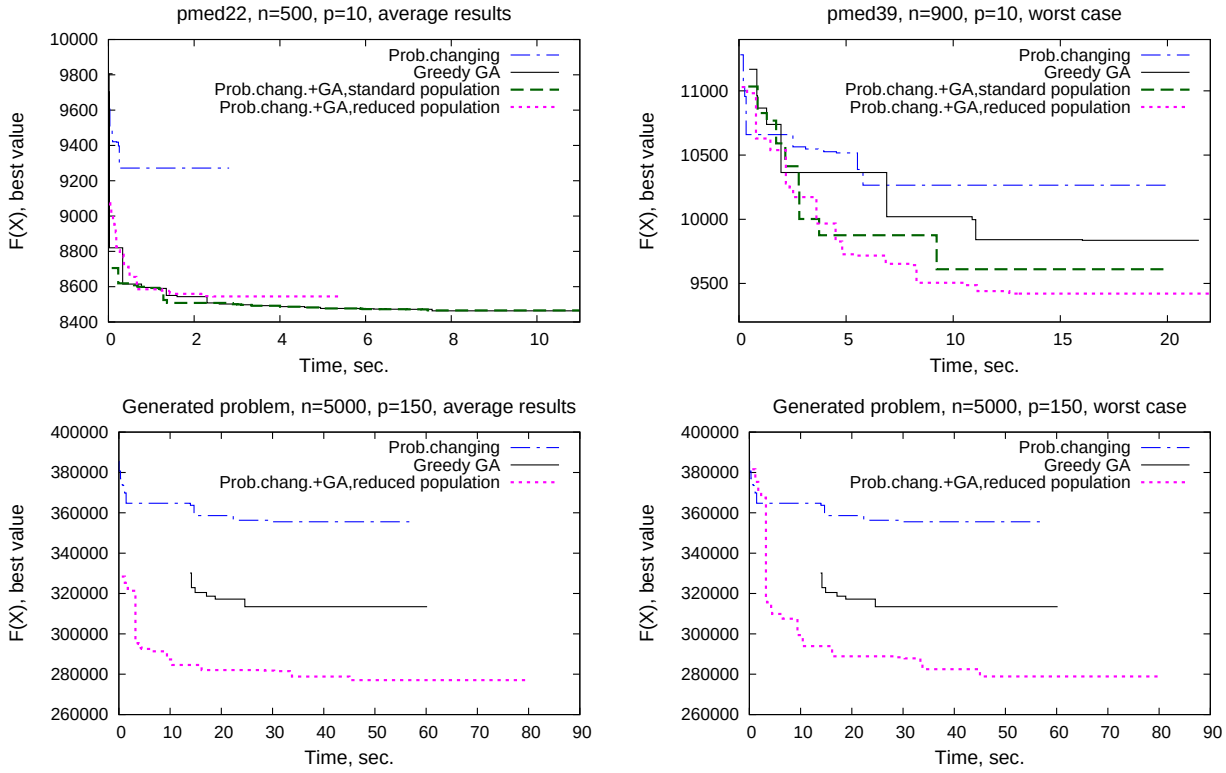


Figure 5: Comparison of the probability changing methods and its combinations with the GA

The variant with the reduced population shows worse accuracy but it can be useful for fast search for an approximate solution of the large-scale problems.

We performed 5 steps of Algorithm 1 ($N_{STEPS} = 5$ in the Step 6) with the probability adaptation (Algorithm 5) and used its last population $\{X_{5,i}|i = \overline{1, N_{POP}}\}$ as the initial population for the GA. The "chromosomes" $\mathcal{M} \in \{X_{5,i}|i = \overline{1, N_{POP}}\}$ are then passed to the crossover procedure (Algorithm 4).

The results shown in Fig. 5 and Fig. 6 were calculated for 10 runs of the original and combined algorithms (3 runs for $n = 7500$). The dependency of the average and worst

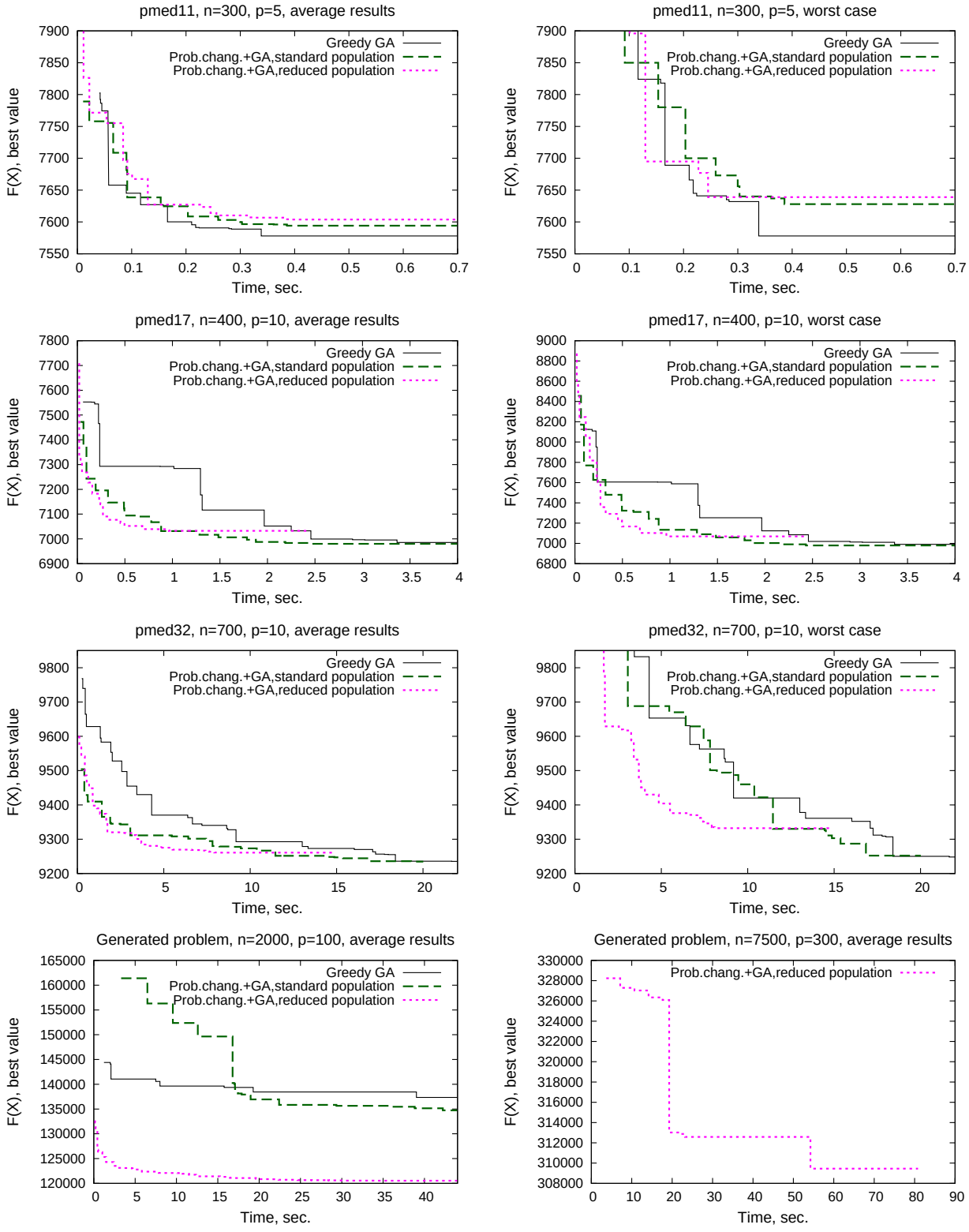


Figure 6: Comparison of the original and combined GAs

values achieved by the algorithms on the spent time was calculated for problems from the OR Library with comparatively small value of l_{avg}/L_{avg} (see (9) and (19)). The results for the problems "pmed11", "pmed12", "pmed14", "pmed16", "pmed18", "pmed19", "pmed21", "pmed23", "pmed35", "pmed31" show the analogous tendencies. We used a combination of 3 stop conditions: reaching the best result announced in the OR Library (if such exists), reaching $[\sqrt{n \cdot p}]$ steps which do not improve the best result or reaching the time limit.

For the problem with $n = 7500$, the results are shown for the combined algorithm with the reduced population only due to memory allocation problems in case of standard population (21).

In case of using the probability changing method, the standard deviation of the objective function in the population of decreases faster than in case of the GA (Fig. 7). In the combined algorithm, the search continues with a comparatively "good" population.

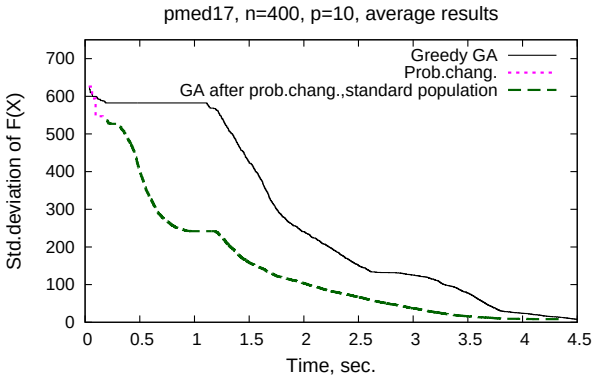


Figure 7: Comparison of the standard deviation of the original and combined GAs

We used the local search (Algorithm 3) with randomly selected initial solutions for testing purposes. Also, the local search was implemented as a procedure of the algorithm based on the probability changing method. We modified Step 2 of Algorithm 1 :

2: In accordance with the distribution given by the elements of the vector P, generate a set of N_{POP} vectors $X_{k,i}, i \in \{\overline{1, N_{POP}}\}$;

2.1: If $[k/5] = k/5$ and $k > 0$ then apply Algorithm 3 to each vector $X_{k,i}$;

The results are shown in Fig. 8. Both, original and combined algorithm ran 10 times. The size of the population of the probability changing algorithm was calculated in accordance with (20). For small size problems, local search in both variants is more efficient than the GA. For most problems included in the OR Library, the results of the combined method are the same as the results of the local search with multistart (case "a" on Fig. 8) because 2-100 starts of the local search procedure are enough for obtaining the exact solution. An exception is the problems with high density (p/n) and comparatively large size ("pmed19", "pmed24", "pmed25", "pmed29"). In this case (case "b" of Fig. 8), combined algorithm allows to reach the exact result faster. For the large scale problems ($n \geq 2000$, case "c") generated by Algorithm 7, the combined algorithm gives better results.

7 Conclusion

The proposed algorithm based on the probability changing method is useful for solving the p-median problem in a network under the assumption that the lengths of the edges are much smaller than the expectation of the path length from a randomly selected vertex to the closest vertex of the solution. Very slow convergence of the algorithm obstructs its implementation as a standalone algorithm. However, its running in combination with other algorithms improves their efficiency significantly. Adjusting the parameters of the algorithm is the subject to the future research.

References

[1] O. Alp, E. Erkut and Z. Drezner (2003) An Efficient Genetic Algorithm for the p-Median Problem, *Annals of Operations Research*, Vol.122, Issue 1–4, pp. 21–42, doi 10.1023/A:1026130003508

[2] A. Antamoshkin and I. Masich (2007) Pseudo-Boolean Optimization in Case of an Unconnected Feasible Set, *in Models and Algorithms for Global Optimization Optimization and Its Applications*, Springer Verlag, Vol. 4, pp 111–122

[3] A. N. Antamoshkin (1987) Optimizatsiya funktsionalov s bulevymi peremennymi (Optimization of Functionals with Boolean Variables), Tomsk University Press

[4] A. Antamoshkin, H. P. Schwefel, A. Toern, G. Yin, A. Zhilinskas (1993) *Systems Analysis, Design and Optimization. An Introduction*, Krasnoyarsk, Ofset

[5] P. Avella, M. Boccia, S. Salerno and I. Vasilyev (2012) An Aggregation Heuristic for Large Scale p-median Problem, *Computers & Operations Research*, 39 (7), pp. 1625–1632, doi 10.1016/j.cor.2011.09.016

[6] P. Avella, A. Sassano and I. Vasil'ev (2007) Computational Study of Large-Scale p-Median Problems, *Mathematical Programming*, 109(1), pp. 89–114, doi 10.1007/s10107-005-0700-6

[7] J. E. Beasley (1990) OR-Library: Distributing Test Problems by Electronic Mail, *Journal of the Operational Research Society*, 41(11), pp. 1069–1072

[8] M. Bischoff, T. Fleischmann and K. Klamroth (2009) The Multi-Facility Location-Allocation Problem with

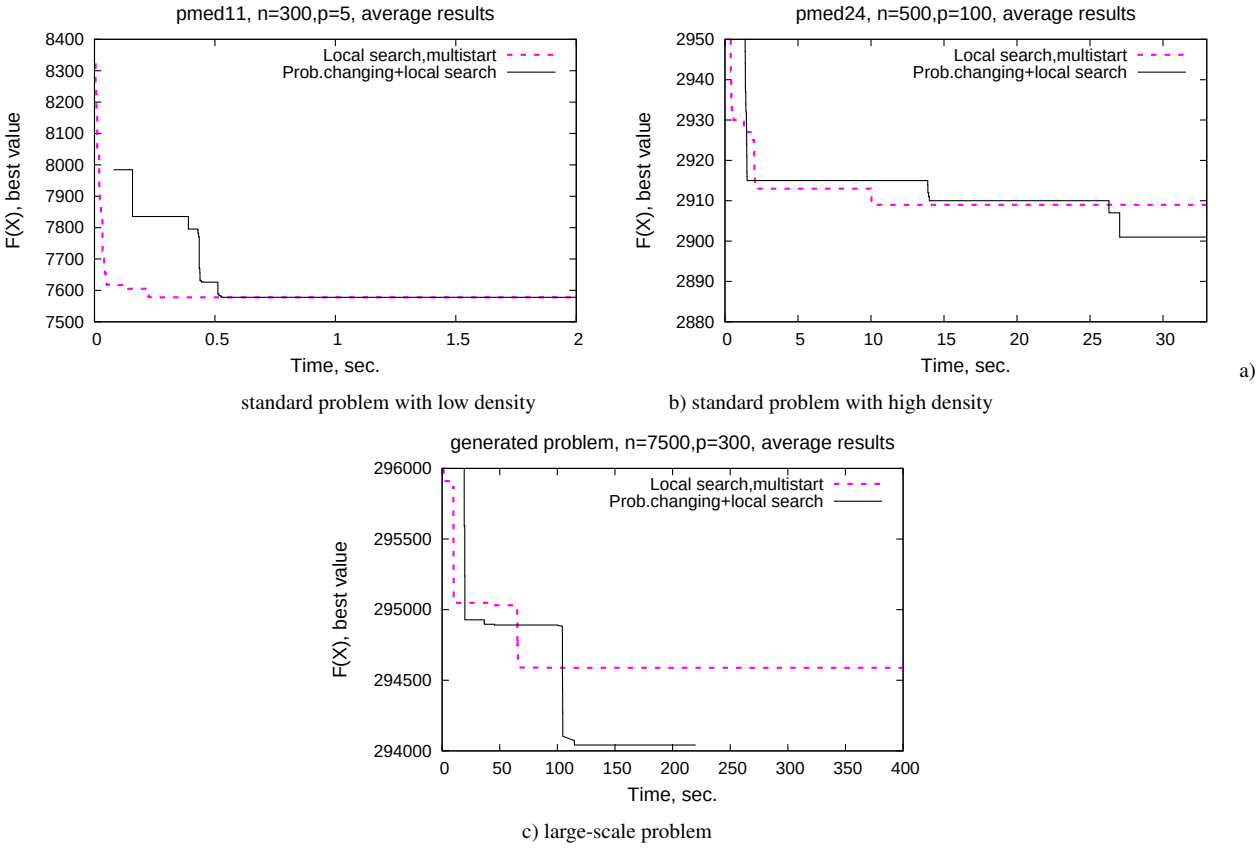


Figure 8: Comparison of the local search and the probability changing method with the local search procedure

Polyhedral Barriers, *Computers and operations Research*, 36, pp. 1376–1392

[9] B. Bozkaya, J. Zhang and E. Erkut (2002) A Genetic Algorithm for the p-Median Problem, *in Z. Drezner and H. Hamacher (eds.), Facility Location: Applications and Theory*, Springer

[10] A. V. Cabot, R. L. Francis and M. A. Stary (1970) A Network Flow Solution to a Rectilinear Distance Facility Location roblem, *American Institute of Industrial Engineers Transactions*, 2, pp. 132–141

[11] L. Cooper (1968) An Extension of the Generalized Weber Problem, *Journal of Regional Science*, Vol. 8, Issue 2, pp.181-197

[12] A. S. Degterev, F. V. Kanashkin and A. D. Sumarokov (2004) Obobshenie geneticheskikh algoritmov i algoritmov skhemy MIVER (Generalization of Genetic Algorithms and Algorithms Based on Probability Changing Method), *Issledovano v Rossii*, vol. 2004, pp. 1658–1665

[13] Z. Drezner and H. Hawacher (2004) *Facility location: applications and theory*, Springer-Verlag, Berlin, Heidelberg.

[14] Z. Drezner and G. O. Wesolowsky (1978) A Trajectory Method for the Optimization of the Multifacility Location Problem with lp Distances, *Management Science*, 24, pp.1507-1514

[15] R. Z. Farahani and M. Hekmatfar, editors (2009) *Facility Location Concepts, Models, Algorithms and Case Studies*, Springer-Verlag Berlin Heidelberg.

[16] M. Gugat and B. Pfeifer (2007) Weber Problems with Mixed Distances and Regional Demand, *Math. Methods of Orerations Research*, issue 66, pp. 419–449

[17] S. L. Hakimi (1964) Optimum Locations Of Switching Centers and the Absolute Centers and Medians of a Graph, *Operations Research*, 12(3), pp. 450-459

[18] P. Hansen, J. Brimberg, D. Urošević, N. Mladenović (2009) Solving large p-median clustering problems by primal–dual variable neighborhood search, *Data Mining and Knowledge Discovery*, vol. 19, issue 3, pp 351–375

[19] C. M. Hosage and M. F. Goodchild (1986) Discrete Space Location–Allocation Solutions from Genetic Algorithms, *Annals of Operations Research* 6, 35–46.

[20] O. Kariv and S. L. Hakimi (1979) An Algorithmic Approach to Network Location Problems. II: The P medians, *SIAM J. Appl. Math.* **37**, pp. 539–560.

[21] L. A. Kazakovtsev (2012) Adaptation of the Probability Changing Method for Weber Problem with an Arbitrary Metric, *Facta Universitatis, series Mathematics and Informatics*, vol. 27 (2), pp. 239–254

[22] L. Kazakovtsev (2012) Random Constrained Pseudo-Boolean Optimization Algorithm for Multiprocessor Systems and Clusters, *Proceedings of the IV International Congress on Ultra Modern Telecommunications and Control Systems 2012 (ICUMT)*, S. Petersburg, 23-25 September 2012, pp. 473–480 doi: 10.1109/ICUMT.2012.6459711

[23] V. Marianov and D. Serra (2009) Median Problems in Networks, *available at SSRN:* http://ssrn.com/abstract=1428362 or http://dx.doi.org/10.2139/ssrn.1428362

[24] S. Masuyama, T. Ibaraki and T. Hasegawa (1981) The Computational Complexity of the m-Center Problems on the Plane, *The Transactions of the Institute of Electronics and Comunication Engineers of Japan*, 64E, pp. 57–64

[25] N. Megiddo and K. Suppowit (1984) On the Complexity of Some Common Geometric Location Problems *SIAM Journal of Computing*, 13, pp. 182–196

[26] N. Mladenović, J. Brimberg, P. Hansen, J. A. Moreno-Perez (2007) The p-median problem: A survey of metaheuristic approaches, *European Journal of Operational Research*, Vol. 179, issue 3, pp.927–939

[27] J. G. Morris (1981) Convergence of the Weiszfeld Algorithm for Weber Problem Using a Generalized "Distance" Function, *Operations Research*, vol. 29 no. 1, pp. 37–48

[28] I. A. Osinuga and O. N. Bamigbola (2007) On the Minimum Norm Solution to the Weber Problem, *SAMSA Conference proceedings*, pp. 27–30

[29] M. Rabbani (2013) A Novel Approach for Solving a Constrained Location Allocation Problem, *International Journal of Industrial Engineering Computations*, published online, doi 10.5267/j.ijiec.2013.02.003, http://www.growingscience.com/ijiec/IJIEC_2013_8.pdf

[30] A. W. Reza, K. Dimyati, K. A. Noordin, A. S. M. Z. Kausar, Md. S. Sarker (2012) A Comprehensive Study of Optimization Algorithm for Wireless Coverage in Indoor Area, *Optimization Letters*, September 2012, published online, doi 10.1007/s11590-012-0543-z, http://link.springer.com/article/10.1007 %2Fs11590-012-0543-z?LI=true

[31] M. G. C. Resende (2008) Metaheuristic hybridization with Greedy Randomized Adaptive Search Procedures, *in TutORials in Operations Research, Zhi-Long Chen and S. Raghavan (Eds.)*, INFORMS, pp. 295–319

[32] M. G. C. Resende, C. C. Ribeiro, F. Glover and R. Marti (2010) Scatter search and path-relinking: Fundamentals, advances, and applications, *Handbook of Metaheuristics, 2nd Edition, M. Gendreau and J.-Y. Potvin, Eds.*, Springer pp. 87–107

[33] P. S. Stanimirovic, M. Ćirić, L. A. Kazakovtsev and I. A. Osinuga (2012) Single-Facility Weber Location Problem Based on the Lift Metric, *Facta Universitatis, series Mathematics and Informatics*, vol. 27 (2), pp. 31–46

[34] V. A. Trubin (1978) Effective algorithm for the Weber problem with a rectangular metric, *Cybernetics and Systems Analysis*, **14(6)**, DOI:10.1007/BF01070282, Translated from Kibernetika, **6** (November-December 1978), pp. 67–70.

[35] A. Weber (1922) *Über den Standort der Industrien, Erster Teil: Reine Theorie des Standortes*, Tübingen, Mohr

[36] E. Weiszfeld (1937) Sur le point sur lequel la somme des distances de n points donnes est minimum, *Tohoku Mathematical Journal*, **43** no.1, pp. 335–386.

[37] G. Wesolowsky (1992) The Weber problem: History and perspectives, *Location Science*, **1**, pp. 5–23.

[38] G. O. Wesolowsky and R. F. Love (1972) A nonlinear Approximation Method for Solving a Generalized Rectangular Distance Weber Problem, *Management Science*, vol. 18 no. 11, pp. 656–663

[39] G. G. Zabudski (2004) A Minimax Planar Location Problem with Forbidden Zones: Its Solution Algorithm, *Autom. Remote Control 65, No. 2, pp. 241–247*