Yin AH, Zhou TQ, Ding JW *et al.* Greedy randomized adaptive search procedure with path-relinking for the vertex p-center problem. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 32(6): 1319–1334 Nov. 2017. DOI 10.1007/s11390-017-1802-3

Greedy Randomized Adaptive Search Procedure with Path-Relinking for the Vertex p-Center Problem

Ai-Hua Yin[1], *Member*, *CFF*, Tao-Qing Zhou[2,3], Jun-Wen Ding[2,*], Qing-Jie Zhao[2], and Zhi-Peng Lv[2]

[1] *School of Software and Communication Engineering, Jiangxi University of Finance and Economics Nanchang 330013, China*

[2] *School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China*

[3] *Department of Computer Science, School of Information Engineering, Zhejiang Agriculture and Forestry University Hangzhou 311300, China*

E-mail: aihuayin@jxufe.edu.cn; qqis@163.com; dingjunwen@hust.edu.cn; qingjie.zqj@alibaba-inc.com zhipeng.lui@gmail.com

Received August 29, 2016; revised March 28, 2017.

Abstract The p-center problem consists of choosing a subset of vertices in an undirected graph as facilities in order to minimize the maximum distance between a client and its closest facility. This paper presents a greedy randomized adaptive search procedure with path-relinking (GRASP/PR) algorithm for the p-center problem, which combines both GRASP and path-relinking. Each iteration of GRASP/PR consists of the construction of a randomized greedy solution, followed by a tabu search procedure. The resulting solution is combined with one of the elite solutions by path-relinking, which consists in exploring trajectories that connect high-quality solutions. Experiments show that GRASP/PR is competitive with the state-of-the-art algorithms in the literature in terms of both solution quality and computational efficiency. Specifically, it virtually improves the previous best known results for 10 out of 40 large instances while matching the best known results for others.

Keywords p-center problem, tabu search, path-relinking, facility location

1 Introduction

The objective of the p-center problem is to locate p facilities and assign clients to them so as to minimize the maximum distance between a client and its closest facility. Each client is only served by one facility. In mathematical form, the definition of the p-center problem can be described as follows.

Let $G = (V, E)$ be an undirected graph with vertex set $V = \{v_1, ..., v_n\}$ and edge set E. The distance between any two vertices v_i and v_j is given by d_{v_i,v_j}. The p-center problem is to find $S \subseteq V$ ($|S| = p$) such that the objective function:

$$f(S) = \max\{\min d_{v_i,v_j}\},$$

where $v_i \in S, v_j \in V - S$, is minimized.

The p-center problem has proven to be NP-hard problem[1-2] and a large number of methods for solving this problem have been reported in the literature. Among them are several exact approaches: Hakimi[3] first defined the p-center problem and proposed several graphical based methods to solve it. Kariv and Hakimi[1-2] designed an $O(|E|^p n^{2p-1}/(p-1)! \lg n)$ and an $O(|E|^p n^{2p-1}/(p-1)!)$ ("$|\,|$" means the number of the elements in the set) algorithm for finding an absolute p-center in a vertex-weighted and a vertex-unweighted network respectively, and extended these results by giving an $O(n \lg^{p-2} n)$ algorithm for finding an absolute p-center (where $3 \leqslant p < n$) and an $O(n \lg^{p-1} n)$ algorithm for finding a vertex p-center (where $2 \leqslant p < n$). Minieka[4] introduced an algorithm to determine the minimum threshold distance where all clients are cov-

Regular Paper

The research was supported by the National Natural Science Foundation of China under Grant Nos. 61370183 and 61262011.

*Corresponding Author

ered by facilities by solving a finite series of minimum set covering problems. Based on this method, Daskin[5] presented the mathematical formulation of the p-center problem and developed a bisection method to reduce the gap between the upper and lower bounds of the optimal solutions. Later, Daskin[6] improved his algorithm by replacing the set covering problem with the maximal set covering subproblem and solved it with Lagrangian relaxation. Ilhan and Pınar[7] proposed a two-phase method for this problem, where it first solves the feasibility problems to find lower bounds based on linear programming formulation, and then utilizes feasibility problems to verify feasible solutions. They solved the 84 instances from OR-Library① and TSPLIB② with up to 900 vertices to optimality for the first time. Elloumi *et al.*[8] presented an integer linear programming formulation for the p-center problem with a polynomial number of variables and constraints, and proposed a binary search algorithm to solve instances with up to 1 817 vertices for the first time. Recently, Al-Khedhairi and Salhi[9] proposed modifications (IP* and Daskin*) to the algorithms proposed in [5, 7] to improve their efficiency. Specifically, for the algorithm proposed in [5], the author used tighter initial lower and upper bounds and a more appropriate binary search method to reduce the number of subproblems. For the algorithm proposed in [7], the authors reduced the number of ILP iterations needed to find the optimal solution. IP* and Daskin* were tested on the well-known benchmark sets OR-Library and TSPLIB with encouraging results.

Besides the delightful achievements in the area of exact methods for the p-center problem, a variety of heuristic and metaheuristic algorithms have been proposed to solve it. Hochbaum *et al.*[10] introduced a 2-approximation algorithm for this problem. Martinich[11] proposed a vertex-closing approach to investigate this problem and proved that it can optimally solve some special cases and obtain very good solutions for problems where p is large relative to the number of vertices n. Plesnik[12] presented a polynomial algorithm with a worst-case error ratio of 2 for the p-center problem. Recently, Mladenović *et al.*[13] presented a basic variable neighborhood search (VNS) and two tabu search (TS) heuristics for the p-center problem without triangle inequality. Experiments on all the OR-Library instances and parts of the TSPLIB instances showed that the proposed VNS and two TS algorithms outperform the previous binary search methods, and VNS is averagely better than TS for instances with large p. Hassin *et al.*[14] introduced a local search strategy to solve the p-center problem with a min-max (or max-min) objective where solutions are compared lexicographically rather than by their worst coordinates. Caruso *et al.*[15] presented Dominant, a metaheuristic algorithm to solve the p-center problem. Experiments on all the OR-Library instances indicated that Dominant has good performance in most cases for finding optimal or near optimal solutions in a reasonable amount of time. Pacheco and Casado[16] proposed a metaheuristic algorithm (SS) based on the scatter search approach for two location problems with few facilities ($p \leqslant 10$) and applied it to solve real data instances from health resources. Davidović *et al.*[17] proposed a variant of bee colony optimization (BCOi) algorithm based on the improvement concept for the p-center problem. Experiments on the OR-Library instances showed that BCOi is competitive compared with VNS and SS. Yurtkuran and Emel[18] proposed a modified artificial bee colony (M-ABC) algorithm that benefits from a variety of search strategies to balance search intensification and diversification and applied random key based coding schemes to solve the p-center problem. Experiments on the OR-Library instances revealed the effectiveness of M-ABC when compared with VNS, SS, and BCOi. Besides, some theoretical studies of the p-center problem are presented in [19-21]. Arostegui *et al.*[22] compared the relative performance of tabu search (TS), simulated annealing (SA), and genetic algorithms (GA) on various types of the p-center problem under time-limited, solution-limited, and unrestricted conditions. Pullan[23] proposed a memetic genetic algorithm (PBS) for the p-center problem, which applies phenotype crossover and directed mutation operators to generate new starting points for a local search. PBS was considered to be the best heuristic for it can obtain the optimal solutions for all the small instances ($|V| < 1\,000$) from OR-Library and TSPLIB and high quality solutions for large instances ($|V| \geqslant 1\,000$) from TSPLIB when compared with the other heuristics mentioned above.

This paper presents a GRASP with path-relinking algorithm (denoted by GRASP/PR) to solve the p-center problem. GRASP/PR was first introduced in [24] which has been successfully applied to several classic combinatorial problems including the three index assignment problem[25], the Steiner problem in graphs[26],

①http://people.brunel.ac.uk/~mastjjb/jeb/orlib/pmedinfo.html, Jan. 2017.
②http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/, Jan. 2017.

the job-shop scheduling problem[27], the quadratic assignment problem[28], the max-cut problem[29], etc. Each iteration of the GRASP/PR algorithm seeks to obtain a local optimal solution, followed by a path-relinking which explores trajectories between the current local optimal solution and the previous optimal solutions preserved in the elite set. In both the GRASP and the path-relinking procedure, a tabu search is adopted to intensify the search. We perform the proposed GRASP/PR algorithm on two sets of 124 benchmark instances commonly used in the literature and find that GRASP/PR virtually improves the previous best solutions for 12 instances and matches previous best known solutions for the others, demonstrating the efficiency of the proposed algorithm.

The rest of this paper is organized as follows. Section 2 describes the general framework and details of the GRASP/PR algorithm. Section 3 reports the computational results and comparison with the state-of-the-art algorithms in the literature. Section 4 analyzes and discusses some essential ingredients of the GRASP/PR algorithm. Finally, Section 5 gives a review of the main contribution of our work and outlines the future research directions.

2 GRASP/PR Algorithm

2.1 General Framework of the GRASP/PR Algorithm

GRASP is a multi-start metaheuristic for combinatorial problems in which each iteration consists of two phases: construction and local search. The construction phase builds a feasible solution, where the defined neighborhood is investigated until a local minimum is found during the local search phase. Balancing intensification and diversification plays an essential role in any heuristic methods[30-33]. Path-relinking is an approach to integrating intensification and diversification in the search. It consists of exploring trajectories that connect high-quality solutions. The trajectory is generated by selecting moves introduced in the initial solution attributes of the guiding solution. The objective of path-relinking is to integrate features of good solutions obtained by each iteration of GRASP, into new solutions generated in subsequent iterations. In pure GRASP (i.e., GRASP without path-relinking), all iterations are independent and therefore most good solutions are simply "forgotten". Path-relinking tries to change this situation by retaining previous solutions and using them as "guides" to speed up convergence to a high-quality solution.

The general architecture of GRASP/PR is described in Algorithm 1. For convenience, we adopt the following notations in the GRASP/PR algorithm:

• N_v: an ordered list of the adjacent vertices of v sequenced by non-decreasing order of the distance to v;

• N_{vk}: the first k vertices in N_v where k is the index of the facility in N_v that serves v ($N_v[0], \cdots, N_v[k-1]$);

• $d(S^i, S^j)$: the distance $d(S^i, S^j)$ between two solutions S^i and S^j, which is the size of the symmetric difference between the two solutions S^i and S^j given

Algorithm 1 . Pseudo-Code of GRASP/PR Algorithm for p-Center

Input: problem instance
Output: the best found solution S^*
$P \leftarrow \emptyset$
while stopping condition is not satisfied **do**
    $S \leftarrow GreedyRandomized()$
    $S^t \leftarrow TabuSearch(S)$
    if P is full **then**
        Select an elite solution $S^e \in P$ at random
        $S^r \leftarrow PathRelinking(S^t, S^e)$
        if $S^r \notin P$ and $f(S^r) \leqslant \max\{f(S)|S \in P\}$ **then**
            Let $P' = \{S|S \in P, f(S) \geqslant f(S^r)\}$
            Let $S^w \in P'$ be the most similar solution to S^r, i.e., $S^w = \text{argmin}\{d(S, S^r)|S \in P'\}$
            If there is more than one S^w, randomly select one as S^w
            $P \leftarrow P \cup \{S^r\}$, $P \leftarrow P \setminus \{S^w\}$
        end if
    else
        if $S^t \notin P$ **then**
            $P \leftarrow P \cup \{S^t\}$
        end if
    end if
end while
$S^* \leftarrow \text{argmin}\{f(S)|S \in P\}$

by:

$$d(S^i, S^j) = p - |S^i \cap S^j|. \quad (1)$$

At the beginning, the elite set P is empty, and the local optimal solutions obtained by GRASP are added into P if they are different from the solutions already in P. Once the elite set is full, path-relinking is invoked after each GRASP procedure, where an elite solution $S^e \in P$ is randomly selected and combined with the solution S obtained by GRASP through path-relinking.

If the combined solution S^r obtained by path-relinking is not in the elite set and its objective value is not greater than that of the worst solution in the elite set, then it is inserted into the elite set. Besides, the solution that is the most similar to S^r (with the smallest distance according to (1)) and not better than S^r is deleted from the elite set. Note that if there is more than one most similar solution which is not better than S^r, GRASP randomly selects one and deletes it from the elite set. This scheme keeps the size of the elite set constant and attempts to maintain the diversification of the set. The stop condition of GRASP/PR is that the search reaches a predefined number of maximal iterations $I_{\max}$. In the following subsections, the main components of GRASP/PR (greedy and randomized initial solution procedure, tabu search, and path-relinking) are described in detail.

2.2 Initial Solution

In the construction phase of GRASP, an initial solution is iteratively constructed, one element at a time. Algorithm 2 gives the pseudo-code of the greedy and randomized constructive procedure for GRASP/PR. First, it randomly selects a vertex $v \in V$ as the first facility and adds it to S. Obviously, v is the closest facility for all the vertices in V. Second, it selects the client vertex w ($w \in V, w \notin S$) that has the largest distance to its facility w_f. Then it generates a random number between 0 and 1. If this number is smaller than α, it randomly selects a vertex v from the list N_{wk} and adds it to S; otherwise, it randomly selects a vertex v from the list N_w and adds it to S. Parameter α controls the amounts of greediness and randomness in the algorithm. A value $\alpha = 1$ corresponds to a greedy construction procedure, while $\alpha = 0$ produces random construction. This procedure is repeated until the size of S reaches p.

2.3 Tabu Search Procedure

Tabu search is an intelligent optimization algorithm initially introduced in [34], and has been applied to various combinatorial optimization problems[35-37]. Our tabu search procedure is formed by adding a specific tabu strategy to the local search presented in [23]. The neighborhood used in [23] is defined as swapping one vertex in N_{wk} with one facility in current solution S. Let $swap(i, j)$ ($i \in N_{wk}, j \in S$) denote a move that swaps a vertex i in N_{wk} and a facility j in S. In order to avoid cycling, we forbid the move $swap(i, j)$ to be performed in the next tt iterations (called tabu tenure) when a move $swap(i, j)$ is performed. The information for move prohibition is maintained in the tabu list TL where the value of $TL(i, j)$ is the iteration number when

Algorithm 2 . Pseudo-Code of the Greedy and Randomized Constructive Procedure

Input: problem instance, parameter α
Output: an initial solution S
$S \leftarrow \emptyset$
Select a vertex $v \in V$ at random as the first facility and add it to S, i.e., $S \leftarrow S \cup \{v\}$
while $|S| < p$ **do**
  Select the client vertex w ($w \in V, w \notin S$) that has the largest distances to its facility w_f
  if rand$(0, 1) < \alpha$ **then**
    Identify N_{wk} for vertex w according to w_f
    Select a vertex v from N_{wk} ($v \notin S$) at random and add it to S, i.e., $S \leftarrow S \cup \{v\}$
  else
    Select a vertex v from N_w ($v \notin S$) at random and add it to S, i.e., $S \leftarrow S \cup \{v\}$
  end if
  Identify the closet facility for all the vertices in V
end while
return S

move $swap(i, j)$ is performed. The tabu search always selects a non-tabued move in each iteration which minimizes the objective value to the greatest extent. The tabu status of a move is neglected only if the move leads to a new solution better than the best solution found so far, or all the moves are in tabu status. Our tabu search stops when it reaches a maximal number of iterations $L_{\max}$ (called the depth of the tabu search).

2.4 Path-Relinking

Path-relinking (PR) was originally proposed by Glover[38] as an intensification strategy exploring trajectories connecting elite solutions obtained by tabu search or scatter search[39-41]. Starting from one or more elite solutions, paths in the solution space leading towards other elite solutions are generated and explored in the search for better solutions. To generate paths, moves are selected to introduce attributes that are contained in the elite guiding solution into the current solution. Path-relinking may be viewed as a strategy that seeks to incorporate attributes of high-quality solutions, by favoring these attributes in the selected moves.

Algorithm 3 gives the implementation of the path-relinking for the p-center problem. Let S^{c1} and S^{e1} be the sets that contain different vertices between S^c and S^e, respectively. Let Q be a set that contains specific triads defined as (S, v_i, v_j), where S is a solution of the problem, and v_i and v_j are two different vertices. For each vertex v_c in S^{c1} and each vertex v_e in S^{e1}, it removes v_c out of S^r and adds v_e into S^r. The resulting solution S^t and the corresponding vertices v_c and v_e are composed as an element (S^t, v_c, v_e) which is added to Q (line 8). The best solution in the element (S, v_i, v_j) of Q (i.e., the solution with the least increment of the objective function value) is recorded in S^r (line 11). Then it removes v_c and v_e out of S^{c1} and S^{e1}, respectively. This procedure repeats $\beta \times d(S^c, S^e)$ times, where β is a parameter that represents the ratio of the position of S^r in the path from the initial solution to the guiding solution. Afterwards, the resulting solution S^r is optimized by the tabu search procedure.

β is an important parameter that controls the distance between the intermediate solution and the starting solution. It represents the diversification effect of path-relinking. It is reasonable to consider that too large or small value of β may not provide a proper dose of diversification because it produces a solution that are too close to the guiding solution or the starting solution. We conduct extensive experiments to analyze the influence of parameter β on the performance of GRASP/PR in Subsection 2.4.

3 Experimental Results and Comparisons

In this section we report extensive experimental results of applying GRASP/PR on the well-known standard benchmark instances and compare the performance of GRASP/PR with the state-of-the-art algorithms in the literature.

3.1 Problem Instances and Experimental Protocol

We carried out computational experiments on two sets of test problems. The first set is the p-median uncapacitated problem instances drawn from the OR-

Algorithm 3 . Pseudo-Code of Path-Relinking for p-Center

1: **Input**: the current GRASP solution S^c, the guiding solution S^e, parameter β
2: **Output**: the best solution S^r found by path-relinking
3: $S^r \leftarrow S^c$, $Q \leftarrow \emptyset$
4: $S^{c1} \leftarrow S^c \setminus (S^c \cap S^e)$, $S^{e1} \leftarrow S^e \setminus (S^c \cap S^e)$
5: **for** i from 1 to $\beta \times d(S^c, S^e)$ **do**
6:     **for** each vertex $v_c \in S^{c1}$ **do**
7:         **for** each vertex $v_e \in S^{e1}$ **do**
8:             $S^t \leftarrow S^r \setminus \{v_c\}$, $S^t \leftarrow S^r \cup \{v_e\}$, $Q \leftarrow Q \cup \{(S^t, v_c, v_e)\}$
9:         **end for**
10:     **end for**
11:     $(S^r, v_c, v_e) \leftarrow$ argmin $\{f(S) | (S, v_i, v_j) \in Q\}$
12:     $S^{c1} \leftarrow S^{c1} \setminus \{v_c\}$, $S^{e1} \leftarrow S^{e1} \setminus \{v_e\}$, $Q \leftarrow \emptyset$
13: **end for**
14: $S^r \leftarrow TabuSearch(S^r)$

Library③. The benchmark set consists of 40 instances with $|V|$ ranging from 100 to 900 and p ranging from 5 to 90. The second set is the TSP instances from the TSPLIB④. This set consists of 84 instances with $|V|$ ranging from 226 to 1 817 and p ranging from 5 to 150. A detailed description of the instances can be found in [42]. In all the instances, the shortest distance matrix $\boldsymbol{d}$ is not given directly, thereby we initialize matrix $\boldsymbol{d}$ with the Floyd algorithm[43] for the instances in the OR-Library which provides the lengths of the edges, and Euclidean distances for the instances in the TSPLIB which provides the coordinates of the vertices.

Our algorithm was programmed in C++ and performed on a PC with 3.40 GHz CPU and 4 GB RAM. We conducted preliminary experiments to find out the best values of all the parameters used in our algorithm on all the tested instances. Besides, we presented a detailed analysis and adjustment of two important parameters (α, β) in Subsection 4.2. In the following experiments, we set $I_{\max}$, tt, $L_{\max}$, α, β, and the maximum size of elite set P to 10 000, $p \times (|V|-p)/100+\text{rand}(10\times p)$, 10 000, 0.7, 0.5, and 10, respectively. All the computational results are obtained without special tuning of the parameters. We run GRASP/PR on each problem instance for 20 independent runs and compare it with several state-of-the-art algorithms in the literature.

3.2 Computational Results

We compared the performance of GRASP/PR with three exact algorithms (ELP, IP*, and Daskin*) and two metaheuristic algorithms (VNS and PBS). ELP was run on a PC with a 400 MHz Pentium II CPU and 384 MB RAM[8]. IP* and Daskin* were run on a Sun Enterprise Workstation 450[9]. VNS was run on a Sun Sparc Station 10. The only parameter of VNS, i.e., the number of neighborhood structures, is set to p[13]. PBS was run on a PC with AMD Opteron 252 2.6 GHz CPU[13]. However, a completely fair comparison is impossible since we have a nondeterministic approach (i.e., our approach) on one hand and the deterministic approaches on the other hand. In addition, different computing environments constitute another major source of difficulty for a fair comparison. Therefore, the reported computational time are only given for indicative purposes.

According to the size of V, we classify the test instances into two categories: small instances ($|V| <$ 1 000) and large instances ($|V| \geqslant$ 1 000). The small instances include the 40 p-median instances and 11 instances from the TSPLIB: "pr226, pr264, pr299, pr439, pcb442, kroA200, kroB200, lin318, gr202, d493, d657". The large instances are "u1060, rl1323, u1817".

3.2.1 Computational Results of the Small Instances

Table 1 and Table 2 report the computational results of the proposed algorithm on small instances. Specifically, Table 1 presents the computational results of 40 p-median instances and makes comparison with ELP, IP*, Daskin*, VNS, and PBS, where column $t(s)$ shows the CPU time for ELP, IP*, and Daskin*, and average CPU time for VNS, PBS, and GRASP/PR, and column t_{std} in PBS and GRASP/PR shows the standard deviation of the CPU time for each instance. Table 2 presents the small instances from TSPLIB and makes comparison with IP*, Daskin*, and PBS. Column f_{best} presents the best solution value obtained by PBS and GRASP/PR. As both PBS and GRASP/PR use floating point data and the reference exact algorithms use integer data, a result obtained by PBS or GRASP/PR is deemed to be identical to the result obtained by the exact algorithms if the rounded PBS or GRASP/PR result equals the result obtained by the exact algorithm.

From Table 1 and Table 2, one observes that although both GRASP/PR and PBS can obtain the optimal solutions for all the instances in OR-Library and TSPLIB with a success rate of 100%, GRASP/PR outperforms PBS and the other algorithms for it has the smallest average computational time. Besides, GRASP/PR has a smaller value of t_{std} compared with PBS. Furthermore, for instances pr299 with 40 facilities and gr202 with 6 facilities, GRASP/PR can obtain a relatively better result than PBS (indicated in bold).

3.2.2 Computational Results of the Large Instances

In this subsection, we conducted experiments of GRASP/PR on the large instances with up to 1 817 vertices and 150 facilities. Table 3 presents the computational results of GRASP/PR and makes comparison with PBS, ELP, and VNS on the large instances. Column f_{opt} shows the optimal solution values obtained by ELP, where an optimal solution value that is not identified by ELP is marked with "?". Column f_{best} shows the best solution value obtained by the referenced algorithm. Columns $t(s)$, t_{std}, and f_{dev} show the average

③http://people.brunel.ac.uk/~mastjjb/jeb/orlib/pmedinfo.html, Jan. 2017.

④http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/, Jan. 2017.

Table 1. Computational Results and Comparison with the Reference Algorithms on Small Instances from OR-Library

Instance	n	p	Density	f_{opt}	$t(s)$				PBS		GRASP/PR	
					ELP	IP*	Daskin*	VNS	$t(s)$	t_{std}	$t(s)$	t_{std}
pmed01	100	5	0.040 4	127	0.70	4.05	2.09	0.04	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pmed02	100	10	0.040 4	98	0.20	1.52	1.45	4.45	0.01	0.01	$< \varepsilon$	$< \varepsilon$
pmed03	100	10	0.040 4	93	0.10	1.81	1.35	0.17	0.06	0.05	0.01	0.02
pmed04	100	20	0.040 4	74	0.10	1.01	0.92	0.30	$< \varepsilon$	0.01	$< \varepsilon$	0.01
pmed05	100	33	0.040 4	48	0.10	1.49	0.73	0.11	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pmed06	200	5	0.040 2	84	0.30	13.53	9.01	1.51	0.02	0.02	$< \varepsilon$	$< \varepsilon$
pmed07	200	10	0.040 2	64	0.50	5.09	4.31	1.30	0.01	0.02	$< \varepsilon$	$< \varepsilon$
pmed08	200	20	0.040 2	55	0.40	5.31	3.34	2.22	0.01	0.01	$< \varepsilon$	0.01
pmed09	200	40	0.040 2	37	0.10	3.46	2.66	10.89	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pmed10	200	67	0.040 2	20	0.30	2.76	2.57	9.04	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pmed11	300	5	0.040 1	59	1.00	11.67	16.25	1.52	0.04	0.06	0.01	0.03
pmed12	300	10	0.040 1	51	1.30	12.03	12.25	5.39	0.01	0.02	$< \varepsilon$	$< \varepsilon$
pmed13	300	30	0.040 1	36	0.80	14.43	8.23	10.21	0.05	0.05	$< \varepsilon$	0.02
pmed14	300	60	0.040 1	26	0.90	6.61	6.81	70.67	0.01	0.01	$< \varepsilon$	$< \varepsilon$
pmed15	300	100	0.040 1	18	1.00	4.43	4.40	55.86	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pmed16	400	5	0.040 1	47	1.60	30.01	28.10	0.08	0.01	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pmed17	400	10	0.040 1	39	2.10	30.88	27.06	26.00	0.02	0.02	$< \varepsilon$	0.02
pmed18	400	28	0.040 1	28	1.40	12.49	13.17	119.61	0.13	0.12	0.01	0.07
pmed19	400	80	0.040 1	18	0.40	9.89	10.16	300.11	1.08	1.01	0.57	0.93
pmed20	400	133	0.040 1	13	1.80	13.53	9.30	218.61	0.10	0.08	0.03	0.07
pmed21	500	5	0.040 1	40	5.20	56.40	56.01	1.24	0.01	0.01	0.01	0.01
pmed22	500	10	0.040 1	38	4.30	495.20	60.78	82.63	1.12	1.18	0.24	1.02
pmed23	500	50	0.040 1	22	1.20	28.52	16.45	112.33	2.11	1.84	0.14	1.43
pmed24	500	100	0.040 1	15	4.50	14.64	12.59	264.46	0.06	0.04	0.04	0.03
pmed25	500	167	0.040 1	11	2.70	13.06	10.28	175.67	0.05	0.04	0.02	0.02
pmed26	600	5	0.040 1	38	6.10	401.60	104.20	0.58	0.03	0.03	$< \varepsilon$	$< \varepsilon$
pmed27	600	10	0.040 1	32	8.20	78.24	65.23	5.07	0.04	0.04	$< \varepsilon$	0.04
pmed28	600	60	0.040 1	18	2.10	39.51	19.31	25.09	0.13	0.11	0.04	0.05
pmed29	600	120	0.040 1	13	5.10	32.00	23.61	762.44	0.05	0.03	0.03	0.01
pmed30	600	200	0.040 1	9	5.40	34.72	17.22	196.95	0.80	0.60	0.35	0.23
pmed31	700	5	0.040 1	30	8.10	303.00	122.60	0.58	0.03	0.01	$< \varepsilon$	$< \varepsilon$
pmed32	700	10	0.040 1	29	45.20	1 447.00	116.80	165.26	0.31	0.44	0.05	0.31
pmed33	700	70	0.040 1	15	3.10	94.07	33.11	806.77	81.75	107.25	1.56	2.56
pmed34	700	140	0.040 1	11	6.50	50.23	29.66	160.15	0.04	0.01	0.04	0.01
pmed35	800	5	0.040 1	30	13.70	183.90	123.30	6.67	0.10	0.12	0.04	0.10
pmed36	800	10	0.040 1	27	34.50	3 602.00	110.50	105.99	0.96	0.97	0.55	0.68
pmed37	800	80	0.040 1	15	2.00	105.80	49.02	1 197.86	0.27	0.21	0.08	0.13
pmed38	900	5	0.031 6	29	18.50	251.00	273.10	1.92	0.03	0.01	0.03	$< \varepsilon$
pmed39	900	10	0.040 0	23	27.30	5 817.00	208.70	5.98	26.30	27.12	0.77	1.44
pmed40	900	90	0.040 0	13	7.80	240.80	462.90	493.79	0.46	0.29	0.25	0.27
Avg.					5.67	336.87	51.99		2.91	3.55	**0.12**	**0.24**

Table 2. Computational Results and Comparison with the Reference Algorithms on Small Instances from TSPLIB

Instance	n	p	f_{opt}	f_{best}		$t(s)$		PBS		GRASP/PR	
				PBS	GRASP/PR	IP*	Daskin*	$t(s)$	t_{std}	$t(s)$	t_{std}
pr226	226	40	650	650.00	650.00	7.39	5.68	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pr226	226	20	1 366	1 365.65	1 365.65	9.55	7.28	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pr226	226	10	2 326	2 326.48	2 326.48	8.85	9.29	0.03	0.03	0.010	0.02
pr226	226	5	3 721	3 720.55	3 720.55	19.92	8.08	0.01	0.02	$< \varepsilon$	0.02
pr264	264	40	316	316.23	316.23	-	-	$< \varepsilon$	0.01	$< \varepsilon$	0.01
pr264	264	20	515	514.78	514.78	11.19	9.50	0.02	0.02	$< \varepsilon$	0.01
pr264	264	10	850	850.00	850.00	12.00	11.73	0.01	0.01	$< \varepsilon$	$< \varepsilon$
pr264	264	5	1 610	1 610.12	1 610.12	14.62	16.27	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
pr299	299	40	355	355.52	**355.32**	12.95	11.70	0.18	0.16	0.050	0.08
pr299	299	20	559	559.02	559.02	14.29	13.10	0.22	0.23	0.140	0.15
pr299	299	10	889	888.84	888.84	23.95	18.62	0.43	0.36	0.300	0.33
pr299	299	5	1 336	1 336.27	1 336.27	18.24	18.74	0.09	0.11	0.050	0.02
pr439	439	40	672	671.75	671.75	32.52	33.03	0.66	0.76	0.270	0.34
pr439	439	20	1 186	1 185.59	1 185.59	35.11	29.21	0.12	0.14	0.020	0.05
pr439	439	10	1 972	1 971.83	1 971.83	39.96	36.16	0.09	0.12	0.010	0.09
pr439	439	5	3 197	3 196.58	3 196.58	53.08	51.74	1.84	2.02	0.040	0.06
pcb442	442	40	316	316.23	316.23	2 714.00	1 302.00	0.04	0.04	0.020	0.04
pcb442	442	20	447	447.21	447.21	37.62	151.20	0.89	0.95	0.230	0.11
pcb442	442	10	671	670.82	670.82	34.30	128.80	0.27	0.33	0.060	0.26
pcb442	442	5	1 025	1 024.74	1 024.74	40.31	41.37	0.47	0.52	0.080	0.18
kroA200	200	40	258	258.26	258.26	5.19	5.02	0.14	0.12	0.610	0.09
kroA200	200	20	389	389.31	389.31	5.51	5.38	0.12	0.11	0.050	0.09
kroA200	200	10	599	598.82	598.82	6.57	6.83	0.58	0.61	0.110	0.23
kroA200	200	5	911	911.41	911.41	8.22	8.28	0.10	0.12	0.020	0.11
kroB200	200	40	253	253.24	253.24	5.08	4.29	0.04	0.04	0.010	0.01
kroB200	200	20	382	382.28	382.28	8.27	4.75	0.04	0.03	$< \varepsilon$	$< \varepsilon$
kroB200	200	10	582	582.10	582.10	6.69	5.84	0.05	0.05	0.001	0.03
kroB200	200	5	898	897.67	897.67	7.63	7.53	0.01	0.01	$< \varepsilon$	$< \varepsilon$
lin318	318	40	316	315.92	315.92	219.40	11.67	0.01	0.01	$< \varepsilon$	$< \varepsilon$
lin318	318	20	496	496.45	496.45	13.54	14.49	15.92	20.18	0.860	1.34
lin318	318	10	743	743.21	743.21	17.46	18.52	0.47	0.50	0.060	0.38
lin318	318	5	1 101	1 101.34	1 101.34	18.68	22.06	0.26	0.35	0.050	0.12
gr202	202	40	3	2.97	2.97	5.15	3.81	0.05	0.05	$< \varepsilon$	0.02
gr202	202	20	6	5.97	**5.57**	5.65	4.63	0.01	0.01	$< \varepsilon$	$< \varepsilon$
gr202	202	10	9	9.33	9.33	6.79	4.89	0.05	0.07	$< \varepsilon$	$< \varepsilon$
gr202	202	5	19	19.38	19.38	8.59	5.83	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$	$< \varepsilon$
d493	493	40	206	206.02	206.02	45.38	79.97	36.25	32.78	6.950	5.96
d493	493	20	313	312.74	312.74	1 406.00	75.54	19.36	21.75	2.580	8.23
d493	493	10	458	458.30	458.30	60.13	82.25	5.21	4.79	1.270	1.21
d493	493	5	753	752.91	752.91	72.24	75.41	19.96	15.86	3.490	2.74
d657	657	40	250	249.52	249.52	3 126.00	351.70	196.90	354.34	22.340	7.36
d657	657	20	375	374.70	374.70	301.15	255.80	5.30	5.07	0.460	4.55
d657	657	10	575	574.74	574.74	751.01	156.60	28.58	25.70	4.030	3.31
d657	657	5	881	880.91	880.91	100.56	154.70	209.97	28.66	0.610	1.27
Avg.						212.52	74.30	12.38	11.75	**1.020**	**0.88**

Table 3. Computational Results and Comparison with the Reference Algorithms on Large Instances from TSPLIB

Instance	n	p	ELP		f_{best}		PBS			GRASP/PR			VNS		
			f_{opt}	$t(s)$	PBS	GRASP/PR	$t(s)$	t_{std}	f_{dev}	$t(s)$	t_{std}	f_{dev}	f_{best}	$t(s)$	f_{dev}
u1060	1 060	10	2 273	53	2 273.08	2 273.08	138.11	56.30	0.00	1.31	24.11	0.00	2 280.09	94.93	0.31
u1060	1 060	20	1 581	2 778	1 580.80	1 580.80	659.44	1 449.55	0.00	14.88	85.67	0.00	1 611.95	20.49	1.07
u1060	1 060	30	1 208	298	1 207.77	1 207.77	36.84	25.89	0.00	3.19	30.43	0.00	1 220.41	373.46	0.24
u1060	1 060	40	1021	366	1 020.56	1 020.56	47.73	43.65	0.00	3.26	41.32	0.00	1 050.45	279.75	2.93
u1060	1 060	50	905	383	904.92	904.92	233.13	129.32	0.00	218.85	104.87	0.00	922.14	477.18	0.00
u1060	1 060	60	781	233	781.17	781.17	103.12	74.46	0.00	7.75	89.55	0.00	806.52	446.89	3.25
u1060	1 060	70	711	135	710.76	**710.75**	109.56	20.23	0.00	116.91	12.40	0.00	721.37	422.73	1.49
u1060	1 060	80	652	60	652.16	652.16	142.11	54.56	0.00	316.57	38.53	0.00	670.53	398.84	2.81
u1060	1 060	90	608	38	607.88	**607.87**	63.15	27.14	0.00	7.09	20.78	0.00	640.23	111.08	5.32
u1060	1 060	100	570	29	570.01	570.01	17.54	16.43	0.00	19.04	8.29	0.00	582.92	430.33	2.26
u1060	1 060	110	539	30	538.84	538.84	160.73	74.35	0.00	66.46	57.33	0.00	565.72	186.60	4.99
u1060	1 060	120	510	44	510.28	**510.27**	107.65	28.90	0.00	397.85	18.70	0.00	551.90	218.84	8.16
u1060	1 060	130	500	44	499.65	499.65	118.71	77.53	0.00	58.18	83.08	0.00	500.14	473.65	0.10
u1060	1 060	140	452	46	452.46	452.46	318.48	150.04	0.00	127.39	55.86	0.00	500.12	214.06	10.37
u1060	1 060	150	447	50	447.01	447.01	10.59	12.87	0.00	4.37	11.50	0.00	453.16	428.16	1.38
rl1323	1 323	10	3 077	1 380	3 077.30	3 077.30	4 760.10	1 905.62	0.00	38.02	342.59	0.00	-	-	-
rl1323	1 323	20	2 016	480	2 016.40	2 016.40	605.90	226.38	0.00	104.89	129.04	0.00	-	-	-
rl1323	1 323	30	1 632	900	1 631.50	1 631.50	1 200.20	1 138.23	0.00	169.47	473.51	0.00	-	-	-
rl1323	1 323	40	1 352	3 000	1 352.36	1 352.36	292.00	154.02	0.00	21.90	184.90	0.00	-	-	-
rl1323	1 323	50	1 187	8 580	1 187.27	1 187.27	619.40	211.66	0.00	119.63	110.75	0.00	-	-	-
rl1323	1 323	60	1 063	9 120	1 063.01	1 063.01	8 184.90	8 206.12	0.03	4 190.92	394.07	0.00	-	-	-
rl1323	1 323	70	972	1 740	971.93	971.93	7 427.00	4 157.30	0.03	6 287.04	129.23	0.00	-	-	-
rl1323	1 323	80	895	420	895.06	895.06	8 783.00	6 698.80	0.28	5 265.81	384.50	0.09	-	-	-
rl1323	1 323	90	832	120	832.00	832.00	929.90	584.37	0.00	776.23	453.29	0.00	-	-	-
rl1323	1 323	100	787	120	789.70	789.70	1 840.50	1 070.96	0.34	2 010.67	225.68	0.00	-	-	-
u1817	1 817	10	458	2 700	457.91	457.91	5 316.60	728.10	0.00	604.53	87.25	0.00	-	-	-
u1817	1 817	20	3 10?	4 920	309.01	309.01	10 243.00	3 842.53	0.00	4 068.06	398.87	0.00	-	-	-
u1817	1 817	30	2 50?	16 500	240.99	240.99	1 605.50	371.02	0.00	1 239.97	20.43	0.00	-	-	-
u1817	1 817	40	2 10?	6 420	209.46	**209.45**	193.70	206.73	0.00	308.29	67.34	0.00	-	-	-
u1817	1 817	50	1 87?	9 840	184.91	184.91	1 128.90	714.66	0.00	471.94	234.90	0.00	-	-	-
u1817	1 817	60	163	1 260	162.65	**162.64**	837.30	287.25	0.00	469.43	55.98	0.00	-	-	-
u1817	1 817	70	148	420	148.11	148.11	191.80	193.45	0.00	19.66	87.36	0.00	-	-	-
u1817	1 817	80	137	1 140	136.80	136.80	127.50	105.50	0.00	12.42	110.40	0.00	-	-	-
u1817	1 817	90	1 30?	7 202	129.54	**129.51**	2 963.50	2 646.63	0.00	3 859.05	287.61	0.00	-	-	-
u1817	1 817	100	127	300	127.01	**126.99**	146.40	142.66	0.00	2.35	39.51	0.00	-	-	-
u1817	1 817	110	109	420	109.25	109.25	13 772.40	16 610.21	0.00	6 954.89	434.03	0.00	-	-	-
u1817	1 817	120	108	120	107.78	**107.76**	80.10	62.16	0.00	5.25	19.34	0.00	-	-	-
u1817	1 817	130	1 08?	3 720	107.75	107.75	11.20	6.46	0.00	7.04	13.45	0.00	-	-	-
u1817	1 817	140	1 05?	4 020	101.61	**101.60**	4 949.30	2 349.44	0.00	30.95	137.31	0.00	-	-	-
u1817	1 817	150	94?	5 640	101.60	**92.44**	314.00	54.35	0.00	1 236.55	23.97	0.16	-	-	-
Avg.				2 376.73			1 969.78	1 372.90	0.02	**990.95**	**138.19**	**0.01**	-	-	-

computational time in seconds, the standard deviation of the computational time, and the average deviation of the solution value from the best found solution value, respectively. The result that is not reported by VNS in [13] is marked with "-".

From Table 3, one observes that GRASP/PR can improve the previous best known results for 10 cases compared with PBS (u1060 with p = 70, 90, and 120, and u1817 with p = 40, 60, 80, 90, 100, 120, and 150), and match the previous best known results for the others. Besides, GRASP/PR outperforms PBS for it has smaller values of the average computational time, the standard deviation from the computational time, and the deviation from the best known solution value. Compared with VNS on instances u1060, GRASP/PR has the superiority of both solution quality and computational efficiency for it has smaller values of f_{best}, $t(s)$, and f_{dev}. Note that GRASP/PR can obtain the same solution values with 100% success rate for 20 independent runs for all the instances except rl1323 with p = 80 and u1817 with p = 150. Furthermore, for instance u1817 with p = 150, PBS fails to reach the best result obtained by ELP, while GRASP/PR can obtain a better result than ELP. These results demonstrate the good performance of GRASP/PR in terms of both solution quality and computational efficiency.

4 Analysis and Discussions

4.1 Importance of Tabu Strategy

In order to evaluate the performance of the tabu search in the GRASP/PR algorithm, we modify GRASP/PR by replacing the tabu search procedure with the local search procedure introduced in [23]. The new algorithm is denoted by GRASP/PR-I. We apply GRASP/PR and GRASP/PR-I on each of the instances u1817 with 10∼150 facilities for 20 independent runs and report the comparative results in Table 4, where column *hit* shows the success rate for reaching the best known result f_{best} over 20 runs.

From Table 4, one observes that GRASP/PR outperforms GRASP/PR-I for it can obtain better results for eight cases (i.e., p = 40, 60, 90, 100, 120, 130, 140, and 150), and the average computational time and the standard deviation of the computational time of GRASP/PR are both smaller than these of GRASP/PR-I. Besides, GRASP/PR is more stable than GRASP/PR-I for it has a higher hit rate and a smaller average deviation from the best found solution. These indicate the effectiveness of tabu search as a local search procedure in GRASP/PR for the p-center problem.

Table 4. Comparison Between GRASP/PR and GRASP/PR-I on Instance u1817 with 10∼150 Facilities

Instance	n	p	GRASP/PR-I					GRASP/PR				
			f_{best}	f_{dev}	$t(s)$	t_{std}	*hit*	f_{best}	f_{dev}	$t(s)$	t_{std}	*hit*
u1817	1 817	10	457.91	0.00	2 084.47	384.90	20/20	457.91	0.00	604.53	87.25	20/20
u1817	1 817	20	309.01	0.00	5 672.01	1 023.84	20/20	309.01	0.00	4 068.06	398.87	20/20
u1817	1 817	30	240.99	0.00	3 048.76	304.85	20/20	240.99	0.00	1 239.97	20.43	20/20
u1817	1 817	40	209.46	0.59	1 093.80	128.53	16/20	**209.45**	0.00	308.29	67.34	20/20
u1817	1 817	50	184.91	0.12	1 395.41	582.49	18/20	184.91	0.00	471.94	234.90	20/20
u1817	1 817	60	162.65	0.48	763.55	243.56	15/20	**162.64**	0.00	469.43	55.98	20/20
u1817	1 817	70	148.11	0.00	137.49	114.70	20/20	148.11	0.00	19.66	87.36	20/20
u1817	1 817	80	136.80	0.00	88.73	95.11	20/20	136.80	0.00	12.42	110.40	20/20
u1817	1 817	90	129.54	0.22	2 014.87	1 645.83	13/20	**129.51**	0.00	3 859.05	287.61	20/20
u1817	1 817	100	127.01	0.00	76.48	124.65	20/20	**126.99**	0.00	2.35	39.51	20/20
u1817	1 817	110	109.25	0.00	7 723.49	1 395.20	20/20	109.25	0.00	6 954.89	434.03	20/20
u1817	1 817	120	107.78	0.00	57.66	164.09	20/20	**107.76**	0.00	5.25	19.34	20/20
u1817	1 817	130	107.76	0.28	33.98	14.58	15/20	**107.75**	0.00	7.04	13.45	20/20
u1817	1 817	140	102.39	0.00	2 487.76	879.34	20/20	**101.60**	0.00	30.95	137.31	20/20
u1817	1 817	150	101.60	0.32	302.54	47.56	12/20	**92.44**	0.16	1 236.55	23.97	17/20
Avg.				0.13	1 798.73	476.62	17.27/20		**0.01**	**1 286.03**	**134.52**	**19.820**

4.2 Influence of Parameters α and β

As indicated in Section 2, parameter α controls the amounts of greediness and randomness in the initial solution procedure, and parameter β represents the diversification effect of path-relinking. In this subsection, we analyze the influence of parameters α and β on the performance of GRASP/PR. For parameter α, we take 11 different values $\alpha \in [0, 1]$ (step size of 0.1) and keep other parameters fixed. For parameter β, we take 21 different values of $\beta \in [0, 1]$ (step size of 0.05) and keep other parameters fixed. We perform 20 independent runs for each parameter and each of the 15 large instances u1060 with 10∼150 facilities, and stop our algorithm when it reaches 5 000 iterations. Fig.1(a) shows the average normalized objective value[3] of the initial solutions (obtained by the initial solution procedure, denoted by "*norm.f* of initial solution") and the final solutions (obtained by GRASP/PR, denoted by "*norm.f* of final solution") corresponding to different values of parameter α. Fig.1(b) shows the average normalized objective value and computational time, and their quadratic fitting curve corresponding to different values of parameter β.

From Fig.1(a), one observes that the average objective value of the initial solution gradually increases with the increment of α from 0 to 1 while the average objective value of the final solution does not change too much when $\alpha \in [0, 1]$. This implies that α can influence the quality of the initial solution but has little impact on the quality of the final solution of GRASP/PR. Considering that the average objective value of the final solution is the smallest when $\alpha = 0.7$, α is suggested to be 0.7.

From Fig.1(b), one observes that GRASP/PR can find relatively high-quality solution when $\beta \in [0.35, 0.7]$, while the corresponding average computational time increases when $\beta \in [0, 0.65]$ and decreases when $\beta \in [0.65, 1]$. This indicates that, for parameter β, only the values in the middle of region $[0, 1]$ can introduce a considerable diversification into the search but requires more computational time. The reason may be that, if β is too large or small, the intermediate solutions are too close to the starting or the guiding solution, and thus it is easy for tabu search to be trapped into previous local optima. In order to testify

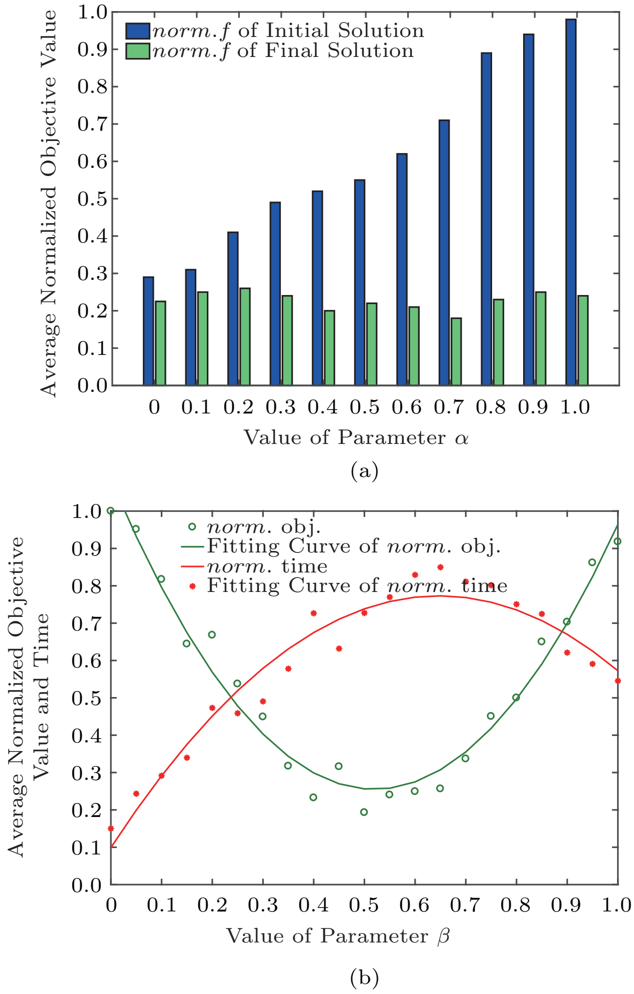


Fig.1. Average normalized objective function value and computational time corresponding to different values of parameters α and β. *norm.* means average normalized value, and obj. is short for objective.

this phenomenon, we apply GRASP/PR on the 15 large instances u1060 with 10∼150 facilities with 11 different values of $\beta \in [0, 1]$ (step size of 0.1) and record the number of times that the returned solution of path-relinking encounters (is the same as) the previous visited local optima (here the local optima are solutions obtained by tabu search). The results are presented in Table 5, where column "Value of β" represents the value of β and column "Number of Encounter" represents the average times that the returned solution of path-relinking encounters the previous visited local optima. From Table 5, one observes that the number of encounters decreases when $\beta \in [0, 0.5]$, while it in-

[3]Average normalized objective value: first, finding the maximum and minimum objective values (denoted by $f_{\max}$ and $f_{\min}$ respectively) over the 20 runs for each instance; second, normalizing the objective value of each run for each instance using $norm.f = (f - f_{\min})/(f_{\max} - f_{\min} + 1)$; third, averaging the $norm.f$ for all the 15 instances for each value of parameter α or β.

creases when $\beta \in [0.5, 1]$. This confirms that the tabu search procedure in path-relinking is easy to be trapped into previous optima with too large or small value of β. Considering both solution quality and computational time, β is suggested to be 0.5.

Table 5. Average Times that the Returned Solution of Path-Relinking Encounters the Previous Visited Local Optima in GRASP/PR on u1060 with 10∼150 Facilities Corresponding to Different Values of β

Value of β	Number of Encounters ($\times 10^5$)
0.0	12.36
0.1	10.89
0.2	9.58
0.3	7.33
0.4	6.01
0.5	4.75
0.6	5.24
0.7	7.18
0.8	8.63
0.9	9.93
1.0	11.50

4.3 Fitness-Distance Correlation and Distribution of Local Optima

The fitness-distance correlation (FDC) coefficient ρ[44] is a well-known tool for landscape analysis and can provide useful indications about the problem hardness. The FDC captures the correlation between the fitness (solution quality) of a solution and its distance to the nearest global optimum (or best-known solution if no global optimum is available). For a minimization problem, $\rho = 1$ ideally indicates a perfect correlation between fitness and distance to the optimum, implying that improving the fitness reduces the distance to the global optimum. For landscape $-1 < \rho < 1$, there is virtually no correlation between fitness and distance, while for $\rho = -1$, there is no correlation at all, which means that using the fitness to guide the search towards global optimum may be misleading.

Table 6 reports the results of FDC analysis on instances rl1323 and u1817 with 10∼150 facilities. For each instance, we run GRASP/PR on it for 30 minutes and collect the number of distinct local optima (column $num\_d_{\rm lo}$), the average distance between local optima (column $avg\_d_{\rm lo}$), the average distance between

Table 6. FDC Results on Instances rl1323 and u1817 with 10∼150 Facilities

Instance	n	p	$num\_d_{\rm lo}$	$avg\_d_{\rm lo}$	$avg\_d_{\rm lb}$	$avg\_d_{\rm be}$	ρ
rl1323	1 323	10	843	7.586 50	6.73	6.73	0.56
rl1323	1 323	20	1 590	14.154 80	10.81	10.81	0.47
rl1323	1 323	30	2 512	27.416 00	23.04	23.04	0.53
rl1323	1 323	40	3 301	28.001 50	27.55	27.55	0.64
rl1323	1 323	50	4 219	36.096 90	35.62	35.62	0.62
rl1323	1 323	60	5 296	44.502 60	44.41	44.41	0.77
rl1323	1 323	70	6 198	58.400 30	51.52	51.52	0.61
rl1323	1 323	80	6 979	59.304 30	59.88	59.88	0.64
rl1323	1 323	90	7 679	61.269 20	62.86	62.86	0.74
rl1323	1 323	100	9 626	64.240 20	67.65	67.65	0.70
u1817	1 817	10	428	7.582 29	7.39	7.39	0.55
u1817	1 817	20	852	17.267 60	15.33	15.33	0.35
u1817	1 817	30	1 260	24.961 90	23.85	19.49	0.48
u1817	1 817	40	1 814	32.651 60	35.46	32.60	0.53
u1817	1 817	50	2 325	44.891 60	42.57	39.74	0.56
u1817	1 817	60	3 019	58.885 40	54.46	51.84	0.73
u1817	1 817	70	3 494	68.638 80	64.12	66.39	0.59
u1817	1 817	80	3 919	69.865 10	72.21	70.92	0.58
u1817	1 817	90	4 068	79.169 90	80.70	81.15	0.64
u1817	1 817	100	4 271	89.875 40	85.92	92.23	0.60
u1817	1 817	110	4 779	95.303 80	93.80	96.38	0.80
u1817	1 817	120	6 160	105.595 80	102.93	110.67	0.75
u1817	1 817	130	6 853	121.976 10	113.70	119.95	0.24
u1817	1 817	140	6 419	132.718 80	129.07	130.23	0.61
u1817	1 817	150	6 614	138.355 90	140.49	138.86	0.15

local optima and the nearest best found solution (column $avg\_d_{\rm lb}$), the average distance between the nearest best found solutions (column $avg\_d_{\rm be}$), and the FDC coefficient (column ρ). Besides, we also draw FDC plots for six different instances (rl1323 with 40, 80, 100, and u1817 with 40, 90, 150 facilities) in Fig.2, where the same data used for estimating ρ is displayed graphically. Such plots have been used to estimate the distribution of local optima for a number of problems including TSP[45], QAP[46], graph partitioning[47-49], and

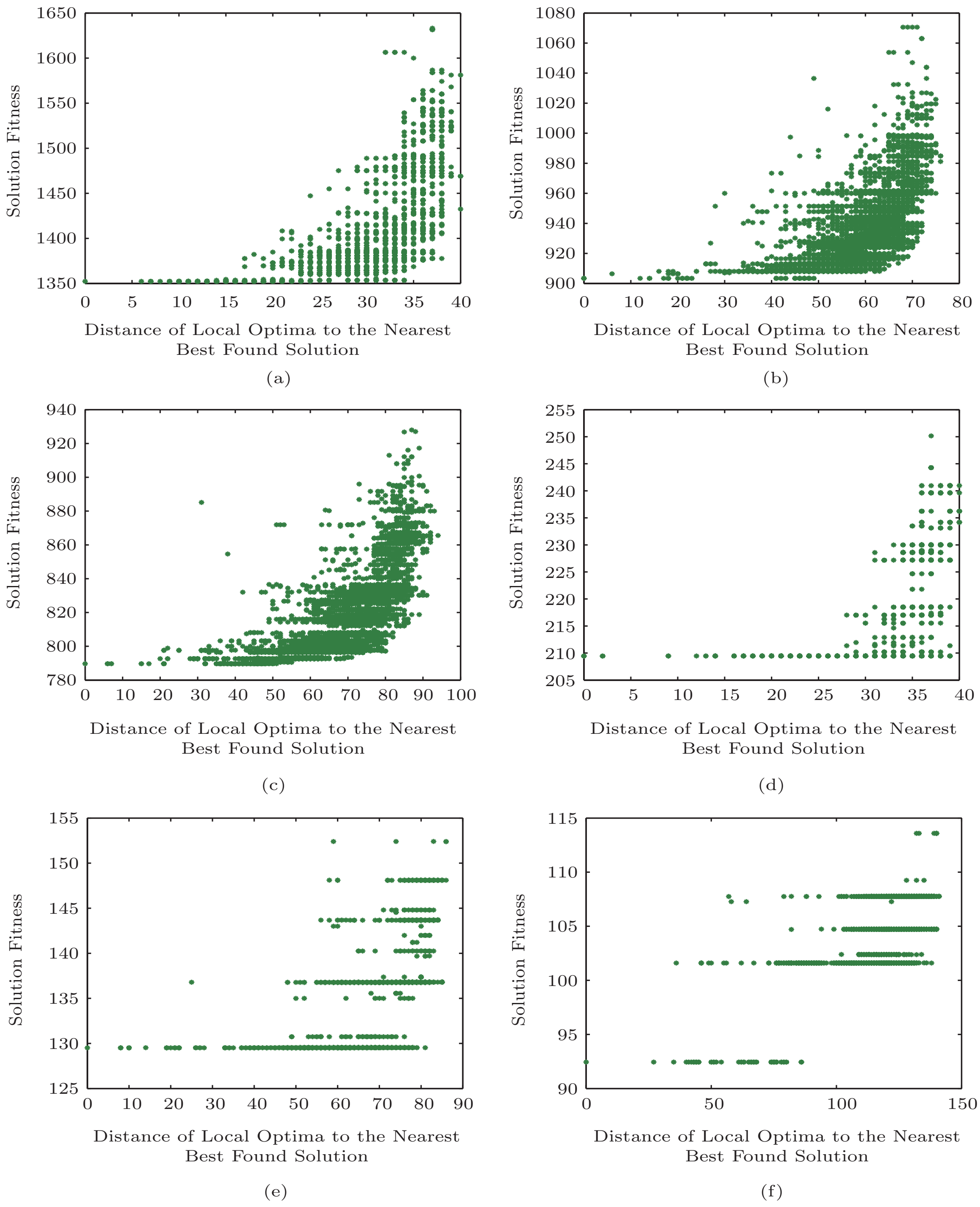


Fig.2. FDC plots of distance of local optima to the nearest best known solution for instances. (a) rl1323, $p = 40$. (b) rl1323, $p = 80$. (c) rl1323, $p = 100$. (d) u1817, $p = 40$. (e) u1817, $p = 90$. (f) u1817, $p = 150$.

flow-shop scheduling problem[50].

From Table 6, one observes that most of the values of $num\_d_{\rm lo}$, $avg\_d_{\rm lo}$, $avg\_d_{\rm lb}$, and $avg\_d_{\rm be}$ are very large, close to the number of the facilities. This implies that local optima are scattered all over the search space. Besides, for the same number of facilities, the values of ρ of instances rl1323 are larger than these of instances u1817, which means that instances rl1323 with given facilities are comparably easy to solve. The FDC plots in Fig.2 also confirm this observation, where it can be seen that: on one hand, the distributions of local optima in rl1323 are denser than those in u1817; on the other hand, the distance of local optima to the nearest best found solution decreases when the solution fitness decreases in rl1323, while the distance of local optima to the nearest best found solution ranges in a wide region in u1817.

5 Conclusions

In this paper, we presented a GRASP/PR algorithm for solving the p-center problems which combines GRASP and path-relinking. The elite set of GRASP/PR consists of the local optimal solutions obtained by GRASP. Each iteration of GRASP consists of the construction of a randomized greedy solution, followed by a tabu search procedure. The resulting solution is combined with one of the elite solutions by path-relinking, which consists in exploring trajectories that connect high-quality solutions.

Tested on two sets of 124 well-known benchmarks, our GRASP/PR algorithm is competitive with the state-of-the-art algorithms in the literature in terms of both solution quality and computational efficiency. Specifically, it improves the previous best known results for 10 out of 40 large instances while matching the best known results for the others. In addition, the computational results demonstrate the robustness and computational efficiency of our GRASP/PR algorithm.

In addition, we investigated some essential ingredients of the proposed algorithm. First, we carried experiments to demonstrate the effectiveness of the tabu search as the local search procedure in GRASP/PR. Second, we analyzed the influence of parameters α and β on the performance of GRASP/PR and suggested a proper value for it.

The success of the GRASP/PR algorithm on the p-center problem reminds us that it is essential to introduce a mechanism to combine the independent elite solutions obtained by GRASP. Given the merits of GRASP with path-relinking, we hope to design even more robust and effective heuristic algorithms for solving the p-center problem and other similar optimization problems such as capacitated p-center, p-median, maximum covering problems, and so on.

References

[1] Kariv O, Hakimi S L. An algorithmic approach to network location problems. I: The p-centers. *SIAM Journal on Applied Mathematics*, 1979, 37(3): 513-538.

[2] Kariv O, Hakimi S L. An algorithmic approach to network location problems. II: The p-medians. *SIAM Journal on Applied Mathematics*, 1979, 37(3): 539-560.

[3] Hakimi S L. Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research*, 1964, 12(3): 450-459.

[4] Minieka E. The m-center problem. *SIAM Review*, 1970, 12(1): 138-139.

[5] Daskin M S. Network and Discrete Location: Models Algorithms and Applications. John Wiley & Sons, 1995.

[6] Daskin M S. A new approach to solving the vertex p-center problem to optimality: Algorithm and computational results. *Communications of the Operations Research Society of Japan*, 2000, 45(9): 428-436.

[7] Ilhan T, Pinar M C. An efficient exact algorithm for the vertex p-center problem. http://www.optimization-online.org/DB_HTML/2001/09/376.html, June 2017.

[8] Elloumi S, Labbé M, Pochet Y. A new formulation and resolution method for the p-center problem. *INFORMS Journal on Computing*, 2004, 16(1): 84-94.

[9] Al-Khedhairi A, Salhi S. Enhancements to two exact algorithms for solving the vertex p-center problem. *Journal of Mathematical Modelling and Algorithms*, 2005, 4(2): 129-147.

[10] Hochbaum D S, Shmoys D B. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 1985, 10(2): 180-184.

[11] Martinich J S. A vertex-closing approach to the p-center problem. *Naval Research Logistics*, 1988, 35(2): 185-201.

[12] Plesník J. A heuristic for the p-center problems in graphs. *Discrete Applied Mathematics*, 1987, 17(3): 263-268.

[13] Mladenović N, Labbé M, Hansen P. Solving the p-center problem with tabu search and variable neighborhood search. *Networks*, 2003, 42(1): 48-64.

[14] Hassin R, Levin A, Morad D. Lexicographic local search and the p-center problem. *European Journal of Operational Research*, 2003, 151(2): 265-279.

[15] Caruso C, Colorni A, Aloi L. Dominant, an algorithm for the p-center problem. *European Journal of Operational Research*, 2003, 149(1): 53-64.

[16] Pacheco J A, Casado S. Solving two location models with few facilities by using a hybrid heuristic: A real health resources case. *Computers & Operations Research*, 2005, 32(12): 3075-3091.

[17] Davidović T, Ramljak D, Šelmić M, Teodorović D. Bee colony optimization for the p-center problem. *Computers & Operations Research*, 2011, 38(10): 1367-1376.

[18] Yurtkuran A, Emel E. A modified artificial bee colony algorithm for p-center problems. *The Scientific World Journal*, 2014, 2014: 824196.

[19] Scaparra M P, Pallottino S, Scutellà M G. Large-scale local search heuristics for the capacitated vertex p-center problem. *Networks*, 2004, 43(4): 241-255.

[20] Cheng T C E, Kang L Y, Ng C T. An improved algorithm for the p-center problem on interval graphs with unit lengths. *Computers & Operations Research*, 2007, 34(8): 2215-2222.

[21] Krumke S O. On a generalization of the p-center problem. *Information Processing Letters* 1995, 56(2): 67-71.

[22] Arostegui M A Jr, Kadipasaoglu S N, Khumawala B M. An empirical comparison of tabu search, simulated annealing, and genetic algorithms for facilities location problems. *International Journal of Production Economics*, 2006, 103(2): 742-754.

[23] Pullan W. A memetic genetic algorithm for the vertex p-center problem. *Evolutionary Computation*, 2008, 16(3): 417-436.

[24] Laguna M, Marti R. GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing*, 1999, 11(1): 44-52.

[25] Aiex R M, Resende M G C, Pardalos P M, Toraldo G. GRASP with path relinking for three-index assignment. *INFORMS Journal on Computing*, 2005, 17(2): 224-247.

[26] Ribeiro C C, Uchoa E, Werneck R F. A hybrid GRASP with perturbations for the steiner problem in graphs. *INFORMS Journal on Computing*, 2002, 14(3): 228-246.

[27] Aiex R M, Binato S, Resende M G C. Parallel GRASP with path-relinking for job shop scheduling. *Parallel Computing*, 2003, 29(4): 393-430.

[28] Oliveira C A S, Pardalos P M, Resende M G C. GRASP with path-relinking for the quadratic assignment problem. In *Proc. the 3rd Int Workshop on Experimental and Efficient Algorithms*, May 2004, pp.356-368.

[29] Festa P, Pardalos P M, Resende M G C, Ribeiro C C. Randomized heuristics for the max-cut problem. *Optimization Methods and Software*, 2002, 17(6): 1033-1058.

[30] Huang W Q, Lv Z P, Shi H. Growth algorithm for finding low energy configurations of simple lattice proteins. *Physical Review E*, 2005, 72(1): 016704.

[31] Zou P, Zhou Z, Wan Y Y, Chen G L, Gu J. New metaheuristic for combinatorial optimization problems: Intersection based scaling. *Journal of Computer Science and Technology*, 2004, 19(6): 740-751.

[32] Xu H Y, Lv Z, Cheng T C E. Iterated local search for single-machine scheduling with sequence-dependent setup times to minimize total weighted tardiness. *Journal of Scheduling*, 2014, 17(3): 271-287.

[33] Xu H Y, Lv Z P, Yin A H, Shen L J, Buscher U. A study of hybrid evolutionary algorithms for single machine scheduling problem with sequence-dependent setup times. *Computers & Operations Research*, 2014, 50: 47-60.

[34] Glover F. Tabu search-part I. *ORSA Journal on Computing*, 1989, 1(3): 190-206.

[35] Huang W Q, Zhang D F, Wang H X. An algorithm based on tabu search for satisfiability problem. *Journal of Computer Science and Technology*, 2002, 17(3): 340-346.

[36] Lai X J, Lv Z P. Multistart iterated tabu search for bandwidth coloring problem. *Computers & Operations Research*, 2013, 40(5): 1401-1409.

[37] Wu J, Rosin P L, Sun X F, Martin R R. Improving shape from shading with interactive tabu search. *Journal of Computer Science and Technology*, 2016, 31(3): 450-462.

[38] Glover F. Tabu search and adaptive memory programming—Advances, applications and challenges. In *Interfaces in Computer Science and Operations Research. Operations Research/Computer Science Interfaces Series*, Barr R S, Helgason R V, Kennington J L (eds.), Springer 1997, pp.1-75.

[39] Glover F. Multi-start and strategic oscillation methods—Principles to exploit adaptive memory. In *Computing Tools for Modeling Optimization and Simulation Operations Research/Computer Science Interfaces Series*, Laguna M, Velarde J L G (eds.), Springer, 2000 pp.1-23.

[40] Glover F, Laguna M, Martí R. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 2000, 29(3): 653-684.

[41] Peng B, Lv Z P, Cheng T C E. A tabu search/path relinking algorithm to solve the job shop scheduling problem. *Computers & Operations Research*, 2015, 53: 154-164.

[42] Reinelt G. TSPLIB—A traveling salesman problem library. *ORSA Journal on Computing*, 1991, 3(4): 376-384.

[43] Floyd R W. Algorithm 97: Shortest path. *Communications of the ACM*, 1962, 5(6): Article No. 345.

[44] Lourenço H R, Martin O C, Stützle T. Iterated local search. In *Handbook of Metaheuristics*, Glover F, Kochenberger G A (eds.), Springer, 2003, pp.320-353.

[45] Boese K D. Cost versus distance in the traveling salesman problem. Technical report TR-950018 UCLA CS Department, 1995.

[46] Stützle T, Dorigo M. ACO algorithms for the quadratic assignment problem. In *New Ideas in Optimization*, Corne D, Dorigo M, Glover F *et al.* (eds.), McGraw-Hill Ltd., 1999, pp.33-50.

[47] Merz P, Freisleben B. Fitness landscapes, memetic algorithms, and greedy operators for graph bipartitioning. *Evolutionary Computation*, 2000, 8(1): 61-91.

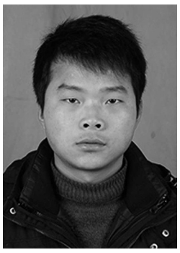
[48] Misevicius A. An improved hybrid genetic algorithm: New results for the quadratic assignment problem. *Knowledge-Based Systems*, 2004, 17(2/3/4): 65-73.

[49] Benlic U, Hao J K. A multilevel memetic approach for improving graph k-partitions. *IEEE Trans. Evolutionary Computation*, 2011, 15(5): 624-642.

[50] Geiger M J. On operators and search space topology in multi-objective flow shop scheduling. *European Journal of Operational Research*, 2007, 181(1): 195-206.

Ai-Hua Yin received his Ph.D. degree in computer science and technology from Huazhong University of Science and Technology, Wuhan, in 2003. Currently, he is a senior researcher at Jiangxi University of Finance and Economics, Nanchang. His research interests include job shop scheduling problem, rectangular packing problem and p-center problem. Dr. Yin is a member of CCF.

Tao-Qing Zhou received his Master's degree in computer application technology from South-Central University for Nationalities, Wuhan, in 2003. Currently, he is a doctorial candidate at Huazhong University of Science and Technology, Wuhan, and a senior lecturer at Zhejiang Agriculture and Forestry University (ZAFU), Hangzhou. His research interests include computational intelligence, solving large-scale combinatorial search problems, designing heuristic algorithms for practical applications, traveling salesman problem and vehicle routing problem.



Jun-Wen Ding received his B.S. degree in computer science from Wuhan Textile University, Wuhan, in 2011, and his M.S. degree in computer software and theory from Huazhong University of Science and Technology, Wuhan, in 2014. He is currently pursuing his Ph.D. degree with the Laboratory of Smart Computing and Optimization, and School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan. His current research interests include artificial intelligence, operations research, and metaheuristic for machine scheduling, job shop scheduling, and vehicle routing problems.

Qing-Jie Zhao received his B.S. degree in computer science from China University of Geosciences, Wuhan, in 2013, and his M.S. degree in computer technology from Huazhong University of Science and Technology, Wuhan, in 2015. His research interests include combinatorial optimization and evolutionary computation.

Zhi-Peng Lv received his B.S. degree in applied mathematics from Jilin University, Changchun, in 2001, and his Ph.D. degree in computer software and theory from Huazhong University of Science and Technology, Wuhan, in 2007. He was a Postdoctoral Research fellow at Laboratoire d'Etude et de Recherche en Informatique d'Angers (LERIA), Department of Computer Science, University of Angers, France, from 2007 to 2011. He is a professor of School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, and the director of the Laboratory of Smart Computing and Optimization (SMART), Wuhan. His research is in the area of artificial intelligence, computational intelligence, operations research and adaptive metaheuristics for solving large-scale real-world and theoretical combinatorial optimization and constrained satisfaction problems.