Egyptian Informatics Journal 24 (2023) 100420

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full Length Article

A hybridization of granular adaptive tabu search with path relinking for the multi-depot open vehicle routing problem

Wenhan Shao, Tuanyue Xiao, Zhouxing Su, Junwen Ding *, Zhipeng Lü

School of Computer Science and Technology, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan, 430074, China

ARTICLE INFO

Keywords:
Multi-depot open vehicle routing problem
Granular neighborhoods
Solution-based tabu search
Adaptive neighborhood selection
Path relinking

ABSTRACT

The multi-depot open vehicle routing problem (MDOVRP) differs from the classical VRP in that there is more than one depot and the vehicle does not need to return to a depot after serving the last customer. For solving this challenging problem, we propose a hybrid metaheuristic algorithm (GATS-PR) which integrates the granular adaptive tabu search with path relinking. The main contributions of this work consist of introducing a solution-based tabu search technique in granular tabu search, designing an adaptive neighborhood selection method for the large neighborhoods with 22 kinds of move types, and adopting path relinking with a new similarity definition to the MDOVRP for the first time. Computational results on 24 public instances demonstrate that GATS-PR outperforms the previous state-of-the-art algorithms in the literature. Specifically, GATS-PR improved and matched the previous best known results on 4 and 19 instances, respectively.

1. Introduction

Since the vehicle routing problem (VRP) was proposed in [1], its variants have been introduced to model various practical applications, including but not limited to city logistics [2], online food ordering delivery [3], multi-robot exploration [4], hazardous waste collection [5] and telecommunications [6]. As a VRP variant, the multi-depot open vehicle routing problem (MDOVRP) was first formulated and solved to distribute fresh meat in an area of the city of Athens [7]. However, it received little attention until the public MDOVRP instances originating from the multi-depot VRP benchmarks were proposed [8]. Because the VRP and its variants are NP-hard [9], it is difficult for an exact algorithm to obtain optimal solutions in a reasonable time on large-scale instances of the MDOVRP [10–12]. In order to solve the MDOVRP effectively and efficiently, we propose the hybridization of granular adaptive tabu search with path relinking (GATS-PR). The main contributions of this study include:

1. The solution-based tabu search instead of attribute-based tabu search is devised in our granular tabu search.
2. The adaptive neighborhood selection method is developed for the large composite neighborhoods with 22 kinds of move types.
3. Path relinking is introduced to solve the MDOVRP for the first time and a new similarity definition is designed for the MDOVRP to guide the path generation.
4. We validate the performance of GATS-PR on 24 public instances. Computational results indicate that GATS-PR outperforms the previous state-of-the-art algorithms.

The remaining of the paper is organized as follows. In Section 2, related work about the MDOVRP is presented. The definition of the MDOVRP is given in Section 3. Section 4 describes the proposed GATS-PR in detail. In Section 5, parameter tuning, computational results and importance analysis of the proposed algorithmic components are reported. Conclusion is given in Section 6.

2. Related work

In this section, we first review the state-of-the-art algorithms for the MDOVRP. Then, we introduce the application of granular tabu search and path relinking in the VRP related field, which are important components of our proposed algorithm.

It is common to solve a problem by proposing its mixed-integer programming model and solving the model with commercial solvers [13–15]. This method is also widely used to solve the MDOVRP. In [8],

* Corresponding author.
E-mail addresses: shaowenhan95@hust.edu.cn (W. Shao), ty_xiaohust@hust.edu.cn (T. Xiao), suzhouxing@hust.edu.cn (Z. Su), junwending@hust.edu.cn (J. Ding), zhipeng.lv@hust.edu.cn (Z. Lü).

https://doi.org/10.1016/j.eij.2023.100420
Received 31 August 2023; Received in revised form 8 November 2023; Accepted 14 November 2023

Available online 24 November 2023
1110-8665/

a mixed integer programming (MIP) formulation of the MDOVRP was proposed and solved with CPLEX. In [11] and [16], subtour elimination constraints of [8] were improved and constraints related to the minimum vehicle number and the first customer to visit were added. In [10], a two-index vehicle flow formulation was proposed to solve the open location routing problem and the MDOVRP. In [12], the MIP model designed for the multi-depot open location routing problem with a heterogeneous fixed-charge fleet was adapted for the MDOVRP and solved with CPLEX.

In order to efficiently obtain high-quality solutions for combinatorial optimization problems, numerous metaheuristic algorithms have been proposed, such as tabu search [17], evolutionary algorithms [18–20], hybrid genetic algorithms [21–24]. They are also applied for the MDOVRP. A hybrid genetic algorithm was proposed in [8], where a solution is encoded as a giant tour of all routes without trip delimiters, the classic order crossover [25] is used to generate a new child solution from the chosen parent solutions, and three move types including 1-0 Exchange, 1-1 Exchange and 2-Opt are applied to improve the child solution. Tabu search with multiple neighborhood search were hybridized to solve the MDOVRP in [26], where multiple neighborhoods are generated by path moves and ejection chains. A memory-based iterated local search algorithm was presented for the MDOVRP in [27], which records moves performed during the local search phase and uses historical search information to guide the perturbation procedure. A variable tabu neighborhood search algorithm was proposed in [28], which applies granular local search in the intensification phase and the tabu shaking mechanism in the diversification phase. A multi-start metaheuristic algorithm was designed in [12], which consists of a constructive heuristic and an iterated local search algorithm to solve the MDOVRP. In [29], a tabu search algorithm was designed for the simultaneous scheduling of multi-project construction and vehicle routing, and it was adapted to the MDOVRP.

Granular neighborhoods [30] and tabu search [31] are common in the field of the VRP, which are often adopted together to efficiently obtain high-quality solutions [32–34]. By excluding unsuitable moves in the current search phase, granular neighborhoods can improve the search efficiency without sacrificing the solution quality. To leverage the characteristic of granular neighborhoods, we design different granular neighborhoods for each move type. Although the review shows that many studies have used tabu search to solve the MDOVRP and other VRP-related problems, all these studies choose the attribute-based tabu search strategy. Because it is difficult to rapidly calculate hash values of solutions and quickly identify whether two solutions are identical, the solution-based tabu search strategy is rarely applied to the VRP and its variants. These challenges are addressed for the MDOVRP by designing the suitable hash function, adopting open addressing hashing with double hashing to handle hash collisions, and recording the characteristics of the visited solution rather than the solution itself.

Due to the difficulty in defining the similarity between solutions, there are few studies applying path relinking in the field of the VRP. In [35], the number of the same customers that appear in matched routes is defined as the similarity of two solutions for the VRP. Because the sequence of nodes on a route is ignored, the solution at the end of the path may not be the same as the guiding solution. Hence, the child solution may not be able to inherit good attributes of the guiding solution. In [36], the minimum number of relocating moves needed to transform one solution to another is defined as their similarity for the VRP. Therefore, the final solution is the same as the guiding solution after path relinking. However, the diversity of the child solution is limited because only one move type is used. To overcome these disadvantages and take advantage of the characteristic that there is more than one depot in the MDOVRP, we design a new similarity definition and consider all the proposed move types at each step of the path generation.

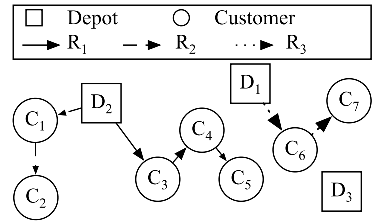

Fig. 1. An example of the solution of an MDOVRP instance.

3. Problem definition

The MDOVRP is defined on an undirected complete graph $G = (V, E)$, where V is the set of nodes and E denotes the set of edges. V consists of two exclusive subsets V^D and V^C which include $|V^D|$ depots and $|V^C|$ customers, respectively. A fleet with an unlimited number of homogeneous vehicles with capacity Q is based on each depot. Each customer $v_i \in V^C$ is characterized by a positive demand q_i ($q_i \leq Q$). Each edge $e_{ij} \in E, i \neq j$ is associated with a positive cost c_{ij} representing the traveling time from node v_i to node v_j. The costs are symmetric and satisfy the triangular inequality.

A route $R = \{v_R^0, v_R^1, \ldots, v_R^{m_R}\}$ consists of m_R different customers and depot v_R^0, where m_R is the number of customers in R. The traveling time T_R of R is equal to $\sum_{i=0}^{m_R-1} c_{v_R^i, v_R^{i+1}}$. The load L_R is the total demand of all customers of R, i.e., $L_R = \sum_{i=1}^{m_R} q_{v_R^i}$. If $L_R \leq Q$, R is a feasible route. The goal of the MDOVRP is to determine a set of feasible routes to serve each customer exactly once with the minimum total traveling time of all routes.

Fig. 1 depicts an example of the solution of an MDOVRP instance, where seven customers are served by three routes beginning from depots D_1 and D_2, while depot D_3 is not used.

4. The hybridization of granular adaptive tabu search with path relinking for the MDOVRP

To explore promising solution space systematically, we propose the GATS-PR algorithm which hybridizes granular adaptive tabu search (GATS) with path relinking. GATS-PR manages population P with μ feasible solutions as shown in Algorithm 1. At each iteration, a new solution $child$ is generated by path relinking from two parent solutions randomly chosen from P (lines 4-5) and is improved by the local search procedure GATS (line 6). If $child$ is feasible and does not belong to P, its objective function value $f(child)$ is compared with that of the worst parent solution p_{worst} (lines 7-8). If $child$ is better than p_{worst}, it replaces p_{worst} (lines 9-10). If $child$ is better than the best solution s^* found so far, it replaces s^* (lines 11-12).

Algorithm 1 General framework of GATS-PR.

Input: Instance
Output: Best solution s^* found so far
1: Initialize population P with μ feasible solutions (Section 4.2)
2: $s^* \leftarrow \text{argmin}_{s \in P} f(s)$
3: **while** the time limit is not reached **do**
4: $p_1, p_2 \leftarrow$ Randomly choose two solutions from P
5: $child \leftarrow$ Apply path relinking to p_1, p_2 (Section 4.3)
6: $child \leftarrow$ Apply GATS to $child$ (Section 4.1)
7: **if** $child$ is feasible and $child \notin P$ **then**
8: $p_{worst} \leftarrow \text{argmax}_{s \in \{p_1, p_2\}} f(s)$
9: **if** $f(child) < f(p_{worst})$ **then**
10: $p_{worst} \leftarrow child$
11: **if** $f(child) < f(s^*)$ **then**
12: $s^* \leftarrow child$
13: **end if**
14: **end if**
15: **end if**
16: **end while**

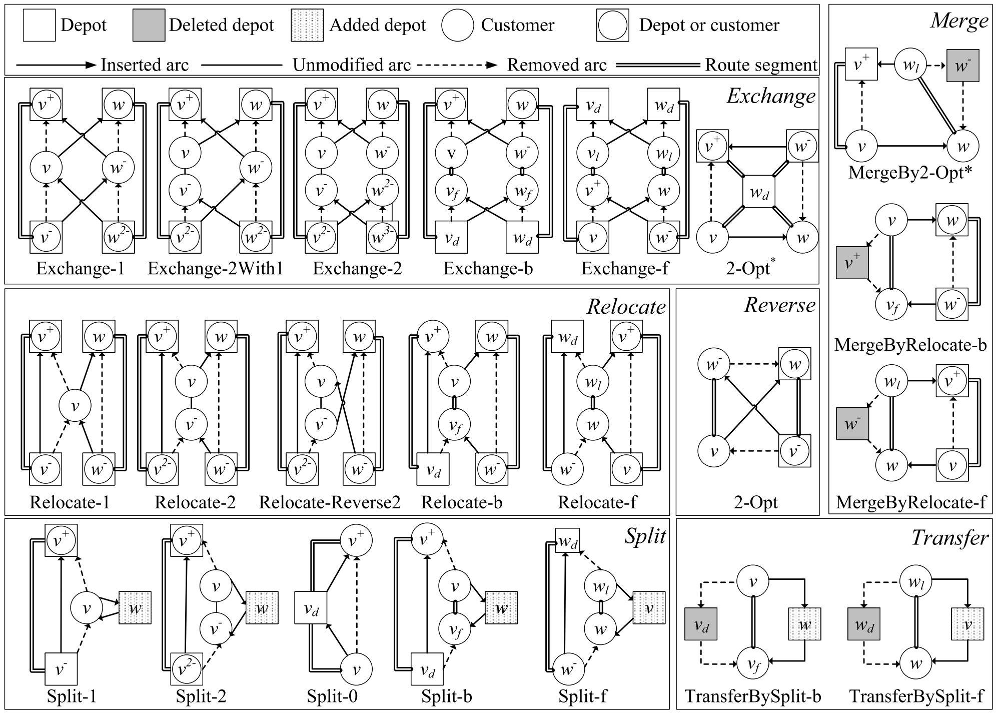

Fig. 2. Move types used in GATS-PR.

4.1. Granular adaptive tabu search

Algorithm 2 presents the main framework of granular adaptive tabu search (GATS). GATS iteratively improves the initial feasible solution s until the termination condition is satisfied (line 2), where ite_{max} is the maximum consecutive iterations without improvement to terminate GATS. At each iteration, δ customers are randomly chosen from all customers (line 3) and a set M of move types is determined by the adaptive neighborhood selection method (line 4). After evaluating all neighborhood solutions of s generated according to M, Cus and granular neighborhoods of Cus, we set s to the best unvisited neighborhood solution in terms of the generalized objective function value (line 5), record s as a tabu solution (line 6), and update the best solution s^* found so far (lines 7-11).

Algorithm 2 General framework of GATS.

Input: Initial feasible solution s
Output: Best solution s^* found so far
1: $ite_{unImp} \leftarrow 0, s^* \leftarrow s$
2: **while** $ite_{unImp} \le ite_{max}$ **do**
3: $Cus \leftarrow$ Randomly choose δ customers
4: $M \leftarrow$ Choose move types with the adaptive neighborhood selection mechanism
5: $s \leftarrow$ Choose the best unvisited neighborhood solution of s
6: Record s as a tabu solution in hash tables
7: **if** s is feasible and $f(s) < f(s^*)$ **then**
8: $ite_{unImp} \leftarrow 0, s^* \leftarrow s$
9: **else**
10: $ite_{unImp} \leftarrow ite_{unImp} + 1$
11: **end if**
12: **end while**

4.1.1. Generalized objective function

In GATS, it is allowed to visit infeasible solutions which violate vehicle capacity constraints to explore wider search space. In order to compare the quality of feasible solutions with that of infeasible ones, we design a generalized objective function f' for the MDOVRP as shown in Eq. (1), where $f(s)$ denotes the objective function value of solution s, $\rho(s)$ defined in Eq. (2) is a normalization factor for relating the order of magnitude of the objective function value and the vehicle capacity penalty, p_{cap} is the factor for penalizing the violation of vehicle capacity constraints, and $v_{cap}(s)$ is the cumulative violation of vehicle capacity constraints of all the tours as shown in Eq. (3). To control the degree of constraint violations timely, p_{cap} is updated at each iteration in the same way as in [37].

$$f'(s) = f(s) + \rho(s) \times p_{cap} \times v_{cap}(s) \tag{1}$$

$$\rho(s) = \frac{|V^C| \times f(s)}{\sum_{i \in V^C} q_i} \tag{2}$$

$$v_{cap}(s) = \sum_{R \in s} max(L_R - Q, 0) \tag{3}$$

4.1.2. Move types

Let v and w represent two nodes, v^{k-} and v^{k+} stand for the kth predecessor and successor nodes of v (k is omitted when $k = 1$). When v is a customer, v_f and v_l denote the first and the last customers of the tour including v, and v_d represents the depot serving v. A total of 22 move types in 6 categories are depicted in Fig. 2 and described as follows:

1. *Exchange* interchanges distinct customers. Exchanged customers are served by the same depot but belong to different tours in 2-Opt* and by different depots in Exchange-b and Exchange-f. For Exchange-1, Exchange-2 and Exchange-2With1, there is no limitation for exchanged customers.
2. *Relocate* moves one or several consecutive customers into a new position. For Relocate-Reverse2, the order of relocated customers is reversed and it is unchanged for other *Relocate* move types. After *Relocate*, there must be at least one customer left on the affected tours.
3. *Reverse* reverses a route segment and it includes 2-Opt.
4. *Split* extracts a route segment of a tour to construct a new tour served by the same or a different depot. After the operation, there

is at least one customer left on the affected tours. Hence, the tour number is increased by one.

5. $Merge$ combines two tours into one, which reduces the tour number by one.
6. $Transfer$ changes the depot of a tour to another.

Although a vehicle does not return to a depot in the MDOVRP, we add an auxiliary edge linking the last customer with the departing depot to simplify the design of move types. The edge cost from a customer to a depot is set to zero.

These six categories of move types differ in how they modify a solution. $Exchange$ swaps customers, $Relocate$ changes the position of customers, $Reverse$ changes the direction of a route segment, $Split$ extracts a route segment from the original tour and serves it with a depot, $Merge$ combines two tours into one, and $Transfer$ severs the tour with another depot. Additionally, the impact degree on the solution varies for distinct move types within each category. For example, relocating one customer has a smaller impact on the solution than relocating a lot of customers. By designing large composite neighborhoods with different functions and impact degrees, the algorithm can select appropriate move types in different search phases and achieve the balance between intensification and diversification of the search.

4.1.3. Adaptive neighborhood selection

If evaluating neighborhood solutions generated by all the move types at each iteration, it will be too time-consuming. Hence, we develop an adaptive neighborhood selection (ANS) strategy to choose suitable move types and group all the move types into four neighborhoods $N = \{N_1, N_2, N_3, N_4\}$ in ascending order of the degree of change to a solution as follows:

- $N_1 = \{$Exchange-1, Relocate-1, Relocate-2, RelocateWithReverse-2, 2-Opt$\}$
- $N_2 = \{$2-Opt*, Exchange-2, Exchange2With1$\}$
- $N_3 = \{$Exchange-b, Exchange-f, Relocate-b, Relocate-f, TransferBySplit-b, TransferBySplit-f$\}$
- $N_4 = \{$MergeBy2-Opt*, MergeByRelocate-b, MergeByRelocate-f, Split-0, Split-1, Split-2, Split-b, Split-f$\}$

At each iteration, ANS selects neighborhood $nei \in N$ with probability p_{nei} as shown in Eq. (4), where w_{nei} is the weight of selecting nei. To avoid too large weight difference among neighborhoods, weights are limited to the interval $\left[w_{min}, w_{max}\right]$ and their initial values are w_{min}. Let s denote the current solution, s' stand for the solution after the move, and sel represent the selected neighborhood. After the move, ANS updates weight w_{sel} as follows:

1. The accumulative generalized objective function value improvement Δ^*_{sel} of sel is updated as shown in Eq. (5), where discount factor α is introduced to balance historical and current contributions of neighborhoods.
2. ANS adjusts w_{sel} by comparing Δ^*_{sel} with the median value $med(\Delta^*)$ of Δ^* as shown in Eq. (6), where factor λ is used to control the convergence speed of weights. If the former is less than the latter, it indicates that sel is more likely to improve the solution. Therefore, w_{sel} is increased. Otherwise, w_{sel} is reduced.

$$p_{nei} = \frac{w_{nei}}{\sum_{i \in N} w_i} \tag{4}$$

$$\Delta^*_{sel} = \alpha \times \Delta^*_{sel} + (1-\alpha) \times (f'(s') - f'(s)) \tag{5}$$

$$w_{sel} = \begin{cases} w_{sel} + \lambda \times (w_{max} - w_{sel}) & \Delta^*_{sel} < med(\Delta^*) \\ w_{sel} + \lambda \times (w_{min} - w_{sel}) & \Delta^*_{sel} \geq med(\Delta^*) \end{cases} \tag{6}$$

If we define a neighborhood for each move type, there will be two disadvantages: 1) Weights of many move types are comparable and the neighborhood selection is random for these move types. This reduces intensification of the search because the neighborhoods whose generalized objective function value improvements are sub-optimal are likely to be frequently selected. 2) If a move type is chosen at an unsuitable time, its weight will be set to a small value and it is hard to select it again when there are too many neighborhoods. This reduces diversification brought by various move types.

Due to the fact that the probabilities of improving the solution are high for move types in N_1 and N_2 in the early stages of the search, these two neighborhoods have high weights and are frequently selected. Because a lot of move types among these neighborhoods are compared to obtain the best improvement move, it guarantees intensification of the search. When the search falls into a local optimum, weights of selecting N_1 and N_2 are decreased because move types in N_1 and N_2 are unlikely to improve the solution quality. This increases the probabilities of choosing N_3 and N_4, and diversifies the search at suitable time.

4.1.4. Granular neighborhoods

As shown in Fig. 2, a neighborhood solution is determined when choosing a move type, a customer cus, and a neighboring node of cus after the move. In order to further speed up the search without deteriorating solution quality, we define granular neighborhoods of cus after the move by considering only proximate nodes of cus to limit the number of neighborhood solutions as follows:

- All depots.
- The first $\lfloor gt \times |V^C| \rfloor$ closest customers of cus, where gt is the granularity threshold and its value is between gt_{min} and gt_{max}.
- Neighboring nodes of cus in the best solution found so far.

We update gt in the same way as mentioned in [37].

4.1.5. Solution-based tabu search

For a population-based hybrid evolutionary algorithm, when the attribute-based tabu search strategy is used, a solution may be visited many times in the local search even if initial solutions are different. Thus, we design a solution-based tabu search strategy which records all visited solutions and avoids revisiting them to further enhance diversification of the search.

Let l represent the list with $|V^C|+|V^D|$ random numbers and l_i the value of the ith element of l. The hash value h of solution s is calculated by the hash function H as shown in Algorithm 3. For each edge of s, corresponding values of its two ends in l are multiplied to XOR with h (line 5), where h is initially set to zero (line 1). The hash value of s after a move is calculated as shown in Algorithm 4. For each deleted and inserted edge of the move, the corresponding values in l of its two ends are multiplied to XOR with h to obtain the hash value of s after the move (lines 3 and 7). Due to the characteristics of the XOR operator, the time complexity of Algorithm 4 is $O(1)$, which ensures the efficiency of the neighborhood evaluation.

Algorithm 3 Hash value of a solution.

Input: Solution s
Output: Hash value h of s
1: $h \leftarrow 0$
2: **for** $tour \in s$ **do**
3: **for** $edge \in tour$ **do**
4: $node_1, node_2 \leftarrow$ Two nodes of $edge$
5: $h \leftarrow (l_{node_1} \times l_{node_2}) \oplus h$
6: **end for**
7: **end for**

To handle hash collisions, we adopt classic open addressing hashing with double hashing. In open addressing hashing, probing refers to obtaining offsets of the hash value calculated by the hash function H when a collision occurs. Due to probing, open addressing hashing is able to place items with the same hash value in different positions. Double hashing is one of the probing methods, which uses

Algorithm 4 Hash value of a solution after a move.

Input: Hash value h of the current solution, all deleted and inserted edges E_d and E_i of the move, respectively

for $edge \in E_d$ **do**
  $node_1, node_2 \leftarrow$ Two nodes of $edge$
  $h \leftarrow h \oplus (l_{node_1} \times l_{node_2})$
end for
for $edge \in E_i$ **do**
  $node_1, node_2 \leftarrow$ Two nodes of $edge$
  $h \leftarrow h \oplus (l_{node_1} \times l_{node_2})$
end for

the second hash function SH to calculate offsets. In our experiment, $SH(s) = H(s) \bmod 97 + 1$.

Let $len = 999983$ represent the size of hash table HT, m stand for the number of solutions in HT, and $lf = m/len$ denote the load factor of HT. If solution s is recorded in the ith row of HT, $HT_i = (f(s), v_{cap}(s))$, where $f(s)$ is the objective function value of solution s and $v_{cap}(s)$ is the cumulative violation of vehicle capacity constraints of s. The initial mapped position $IP(s)$ of solution s and the ith mapped position $P_i(s)$ after double hashing are shown in Eq. (7) and Eq. (8), respectively. When searching for an empty position to record solution s or determining whether s has existed in HT, we check values recorded in rows $R = \{IP(s), P_1(s), P_2(s), \dots\}$ of HT in turn until finding a row $r \in R$, where HT_r is empty or equal to $((f(s), v_{cap}(s))$.

$$IP(s) = H(s) \bmod len \tag{7}$$

$$P_i(s) = (H(s) - i \times SH(s)) \bmod len, i \in \mathbb{N}^+ \tag{8}$$

For the current row $r \in R$:

- If HT_r is equal to $(f(s), v_{cap}(s))$, we consider that solution s has been visited;
- If HT_r is empty, it indicates that s is an unvisited solution;
- Otherwise, the value in the next row of HT is checked.

When load factor lf is large, the search efficiency of the hash table will significantly decrease. Hence, when $lf > 0.75$: An unvisited solution s will be recorded in $HT_{IP(s)}$ if its objective function value $f(s)$ is better than that recorded in $HT_{IP(s)}$; Otherwise, s is discarded.

To avoid identifying an unvisited solution as a visited one under acceptable computational cost, we use three hash tables TB^1, TB^2 and TB^3 with different values in l to record solutions. Only when the pair of $f(s)$ and $v_{cap}(s)$ exists in the mapped positions of all the tables, s is regarded as a tabu solution. Fig. 3 illustrates an example of determining whether a solution is in tabu state. There are two depots 0 and 1 and three customers 2, 3 and 4. Solution s includes one tour departing from depot 0, visiting customers 2, 3 and 4 in turn. The pair p of $f(s)$ and $v_{cap}(s)$ is (9, 2) for solution s. For HT^1, $IP(s) = 22$ and HT^1_{22} is equal to p. Therefore, solution s is considered as a visited solution in HT^1. The pairs of HT^2_5 and HT^3_{40} are different from p, so we check the first mapped positions after double hashing in these tables, which are both 999982. Because HT^2_{999982} and HT^3_{999982} are both empty, solution s is regarded as an unvisited solution in HT^2 and HT^3. Hence, s is an unvisited solution.

4.2. Population initialization

Population P is initialized with μ feasible solutions, each of which is generated after the following two steps and is improved by the local search procedure GATS.

1. Customer assignment: Each customer cus is assigned to its nearest depot for $0.7 \times \mu$ solutions and randomly assigned to one of its two nearest depots according to the probability $1 - \frac{c_{d,cus}}{\sum_{i=1}^{2} c_{d^i_{cus},cus}}$ for the remaining $0.3 \times \mu$ solutions, where d^i_{cus} is the ith nearest depot of cus.
2. Solution construction: For each depot d, customers assigned to d are randomly selected and inserted into a tour served by d one at a time according to the following rules:
   - The customer is inserted into a tour with the minimum cost if it does not violate vehicle capacity constraints;
   - Otherwise, the customer is served by a new tour.

4.3. Path relinking for the MDOVRP

Let $SD_{i,j}$ ($SN_{i,j}$) represent the number of customers served by the same depot (with the same neighbors) in solutions s_i and s_j. The similarity $sim_{i,j}$ between s_i and s_j is defined as the weighted sum of $SD_{i,j}$ and $SN_{i,j}$ as shown in Eq. (9), where w_{SD} is the weight factor of $SD_{i,j}$.

$$sim_{i,j} = w_{SD} \times SD_{i,j} + (1 - w_{SD}) \times SN_{i,j} \tag{9}$$

Algorithm 5 presents the pseudo-code of our path relinking. For initial solution p_1 and guiding solution p_2, the similarity sim_{cur} between them is calculated (line 1). A high-quality solution is more likely to be obtained from solutions in the middle part of the path after applying the GATS procedure because it inherits good attributes of parent solutions.

Consequently, parameters $ratio_{beg}$ and $ratio_{end}$ are used to limit the interval of the path where the final result is selected (line 2). In the path construction (lines 3-12), to avoid the final solution from violating vehicle capacity constraints too much and push p_1 toward p_2, only feasible neighborhood solutions whose similarities to p_2 are larger than sim_{cur} are considered (lines 4-5). We sort solutions of Nei in an ascending order of the sum of their rankings in terms of similarity and objective function value improvements (line 6). After choosing a random solution from the first γ ones of Nei as new p_1 (line 7), sim_{cur} and the reference solution are updated (lines 8-11).

Algorithm 5 Path relinking for the MDOVRP.

Input: Initial solution p_1, guiding solution p_2
Output: Reference solution s_{ref}
1: $sim_{cur} \leftarrow$ Calculate similarity between p_1 and p_2
2: $sim_{beg} \leftarrow sim_{cur} \times ratio_{beg}, sim_{end} \leftarrow sim_{cur} \times ratio_{end}, f(s_{ref}) \leftarrow +\infty$
3: **while** $sim_{cur} \le sim_{end}$ **do**
4:   $Nei \leftarrow$ Generate all the feasible neighborhood solutions of p_1 with all the move types defined in Fig. 2
5:   $Nei \leftarrow$ Retain $s \in Nei$ whose similarity to p_2 is greater than sim_{tmp}
6:   $Nei \leftarrow$ Sort Nei according to the sum of rankings of s in Nei in terms of similarity and objective function value improvements
7:   $p_1 \leftarrow$ Randomly pick a solution from the first γ ones of Nei
8:   $sim_{cur} \leftarrow$ Calculate similarity between p_1 and p_2
9:   **if** $sim_{cur} \ge sim_{beg}$ and $f(s_{ref}) > f(p_1)$ **then**
10:     $s_{ref} \leftarrow p_1$
11:   **end if**
12: **end while**

5. Experimental results

We tested GATS-PR on the public MDOVRP instances introduced in [8].[1] Table 1 presents the notations used in this section. In the following subsections, we first present the parameter tuning process of GATS-PR. Then, the results obtained by GATS-PR are compared with those obtained by the state-of-the-art reference algorithms in the literature. Finally, the importance of the proposed algorithmic components is analyzed.

5.1. Parameter tuning

For instances which are easy to solve, it is not sensitive to parameter settings. Hence, instances p09, p10 and pr05 which are hard to solve

[1] http://neo.lcc.uma.es/vrp/vrp-instances/multiple-depot-vrp-instances/.

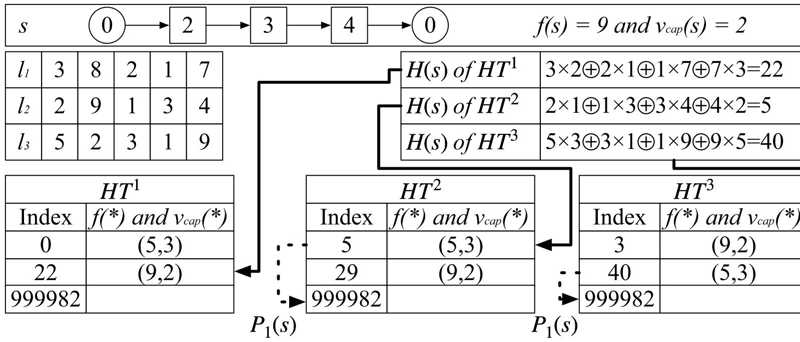

Fig. 3. An example of determining whether a solution is in tabu state.

Table 1
Notations.

Notation	Meaning
Ins	Instance name
BKS	Objective function value of the best known solution
Ref	Reference to the algorithm which obtained BKS for the first time
f_{best}	Objective function value of the best solution
f_{avg}	Average f_{best} among multiple experiments
Gap_{best}	(f_{best}-BKS)/BKS reported as ‰ values
Gap_{avg}	(f_{avg}-BKS)/BKS reported as ‰ values
t_{avg}	Average run time of obtaining the best solutions in seconds
Ave	Average Gap_{best}, Gap_{avg} or t_{avg}
-	Corresponding value is not given in the literature

were selected to tune parameters. Note that similar statistics can be observed on the remaining instances. Because the local search procedure GATS and the path relinking operator are relatively independent, their parameters were tuned by applying GATS and GATS-PR to these instances, respectively. For each parameter, a default value and possible values were given. When tuning a parameter, other parameters were fixed to default values to avoid explosions in the number of parameter combinations. We first tuned GATS-related parameters and set their default values to the selected values when tuning path relinking-related parameters. Three independent runs were performed for the selected instances. In order to normalize the results of different instances, the sum SG_{best} of Gap_{best} of each selected instance is introduced. The parameter value whose corresponding SG_{best} is the minimum was selected as the final value of the parameter and it is indicated in bold in Tables 2 and 3.

Five parameters are critical to the performance of GATS:

- The number δ of customers considered at each iteration. Its default value is $0.3 \times |V^C|$.
- The combination of the lower and upper bounds w_{min} and w_{max} of neighborhood weights. Its default value is (1, 100).
- The discount factor Θ of accumulative generalized objective function value improvement. Its default value is 0.7.
- The convergence speed factor λ of weights. Its default value is 0.05.
- The combination of the minimum and maximum granularity thresholds gt_{min} and gt_{max}. Its default value is (0.05, 0.2).

The termination condition of the GATS procedure is that the number ite_{unImp} of the consecutive iterations without improvement reached $100 \times |V^C|$. The results of tuning GATS-related parameters are shown in Table 2. According to SG_{best}, we set δ to $0.5 \times |V^C|$, w_{min} to 1, w_{max} to 50, Θ to 0.9, λ to 0.1, gt_{min} to 0.05 and gt_{max} to 0.2.

There are four key parameters for the path relinking operator:

- The number γ of elite solutions from which the neighborhood move is picked at each step of path generation. Its default value is 3.
- The weight factor w_{SD} of $SD_{i,j}$. Its default value is 0.7.
- The combination of the population size μ and ite_{max}. Its default value is (10, 2).
- The combination of $ratio_{beg}$ and $ratio_{end}$ determining the path segment of interest. Its default value is (0.6, 0.4).

The maximal run time was 600 seconds for tuning path relinking-related parameters. Based on SG_{best} presented in Table 3, γ was set to 5, w_{SD} to 0.7, μ to 10, ite_{max} to 1, $ratio_{beg}$ to 0.5 and $ratio_{end}$ to 0.2.

5.2. Comparison of GATS-PR with the state-of-the-art algorithms

In Tables 4 and 5, the detailed experimental environment of the reference exact and metaheuristic algorithms are given, where column Name represents the name of the reference algorithms, column Num lists the number of independent runs, column CPU gives the computational configuration, and column Score is the single-thread rating[2] of each computational configuration. GATS-PR was implemented in C++11, built and executed on an Intel Core i3-10100 @ 3.60 GHz processor whose single-thread rating s_{basic} is 2635 MOps/Sec. All CPU times reported in this section are multiplied by s_{alg}/s_{basic}, where s_{alg} is the single-thread rating of the CPU used by the algorithm alg. For GVTS-PR, each instance was independently tested three times with the time limit of one hour.

In terms of t_{avg}, the result of an algorithm is better than/worse than/equal to that of GATS-PR when:

- Its quality is not worse than/not better than/equal to that of GATS-PR. Because HGA, VTNS and MILS were run multiple times, we use f_{avg} as the evaluation criterion of the solution quality. For other reference algorithms except for TS, we use f_{best} because they were run only once. The computational configuration of TS is not given in the literature, so we do not compare its t_{avg} with ours.
- Its t_{avg} is smaller than/larger than/equal to that of GATS-PR.

In Tables 6 and 7, results in column BKS with an asterisk indicate that they are the optimal solutions proved by [11] and [12]. Results with an underline indicate that they are new best known results obtained by GATS-PR. If f_{best} or f_{avg} obtained by an algorithm is equal to BKS, it is indicated in bold. If f_{best}, f_{avg} or t_{avg} obtained by an algorithm is better than that of GATS-PR, it is shown in italics. Rows *Better*, *Equal* and *Worse* represent the number of results whose f_{best}, f_{avg} or t_{avg} is better than, equal to or worse than that of GATS-PR, respectively.

Table 6 presents f_{best} and t_{avg} of GATS-PR and the state-of-the-art exact algorithms. Except for controversial results produced by IMIP on instance p15 and Model on instance pr02, f_{best} obtained by GATS-PR

[2] https://www.passmark.com/.

Table 2
The results of the selected instances to tune GATS-related parameters.

parameter	value	p09			p10			pr05			SG_{best}
		f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}	
δ	$0.1 \times \|V^C\|$	2579.74	2591.39	55.33	2485.79	2490.23	40.00	1704.66	1706.62	14.00	5.81
	$0.3 \times \|V^C\|$	2587.74	2591.75	427.00	2484.93	2487.04	157.33	1695.04	1698.22	114.00	2.90
	$0.5 \times \|V^C\|$	**2577.27**	**2582.25**	**827.67**	**2485.09**	**2489.36**	**501.67**	**1692.98**	**1696.92**	**241.67**	**-2.31**
	$0.7 \times \|V^C\|$	2585.91	2589.18	606.00	2481.73	2488.88	726.67	1700.20	1702.05	169.33	3.94
	$\|V^C\|$	2587.62	2591.90	747.67	2491.18	2495.77	489.33	1692.73	1698.32	387.33	4.01
(w_{min}, w_{max})	**(1, 50)**	**2574.00**	**2585.48**	**228.67**	**2486.33**	**2492.42**	**118.33**	**1692.22**	**1698.91**	**88.67**	**-3.52**
	(1, 100)	2584.44	2589.45	306.33	2485.77	2488.66	82.67	1694.62	1697.85	130.67	1.71
	(1, 500)	2589.24	2593.68	193.00	2480.30	2483.79	161.00	1693.54	1699.36	52.33	0.73
	(1, 1000)	2579.98	2587.00	309.00	2484.05	2485.55	92.67	1696.58	1699.78	89.67	0.44
	(1, 2000)	2572.92	2579.93	174.00	2485.59	2486.36	130.33	1697.49	1700.24	94.00	-1.14
Θ	0.1	2581.14	2584.84	282.00	2481.59	2485.83	92.67	1692.57	1700.48	135.00	-2.46
	0.3	2577.54	2590.23	155.33	2485.71	2486.86	260.67	1700.62	1701.36	84.00	2.55
	0.5	2585.67	2594.99	205.33	2480.07	2484.65	175.33	1699.64	1702.73	82.33	2.85
	0.7	2583.40	2593.53	211.33	2485.71	2486.42	209.67	1699.35	1701.62	92.33	4.07
	0.9	**2579.22**	**2585.14**	**228.00**	**2478.68**	**2482.94**	**235.00**	**1694.97**	**1698.00**	**93.67**	**-2.96**
λ	0.01	2585.30	2591.45	214.00	2482.98	2494.14	161.67	1696.39	1698.22	131.33	1.96
	0.03	2583.66	2591.28	116.00	2475.63	2488.02	172.67	1702.56	1703.97	75.00	2.00
	0.05	2593.09	2598.03	191.00	2491.08	2491.67	228.33	1692.98	1697.63	94.00	6.24
	0.07	2588.27	2592.70	336.00	2484.60	2487.13	153.00	1695.49	1698.13	72.00	3.24
	0.1	**2575.44**	**2585.95**	**217.67**	**2476.81**	**2484.31**	**275.67**	**1698.63**	**1702.92**	**91.67**	**-3.03**
(gt_{min}, gt_{max})	(0.05, 0.1)	2579.22	2580.79	92.00	2491.76	2497.15	74.00	1693.80	1698.65	71.67	1.62
	(0.05, 0.2)	**2573.64**	**2577.69**	**232.33**	**2485.40**	**2486.73**	**84.00**	**1696.05**	**1701.09**	**110.67**	**-1.78**
	(0.05, 0.4)	2587.21	2590.70	266.33	2479.50	2486.30	188.33	1694.70	1698.28	102.33	0.31
	(0.1, 0.2)	2578.74	2593.27	142.67	2485.81	2491.79	94.67	1699.12	1700.07	108.00	2.17
	(0.1, 0.4)	2579.88	2583.94	388.00	2481.96	2483.76	141.33	1695.04	1697.58	151.33	-1.34

Table 3
The results of the selected instances to tune path relinking-related parameters.

parameter	value	p09			p10			pr05			SG_{best}
		f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}	f_{best}	f_{avg}	t_{avg}	
γ	1	2577.71	2580.60	550.67	2482.98	2483.93	492.00	1695.20	1696.65	349.00	-1.68
	2	2582.50	2585.38	616.33	2476.12	2481.61	418.67	1700.72	1701.92	389.33	0.67
	3	2577.40	2579.83	611.67	2478.86	2482.98	553.67	1695.04	1698.00	390.33	-3.55
	5	**2580.61**	**2584.18**	**560.67**	**2474.82**	**2476.47**	**432.33**	**1693.52**	**1695.42**	**552.33**	**-4.83**
	10	2578.84	2583.65	583.00	2476.85	2479.00	414.33	1696.54	1699.14	517.00	-2.92
w_{SD}	0.1	2579.20	2585.21	569.33	2474.82	2481.41	519.00	1695.04	1696.32	401.33	-4.48
	0.3	2575.36	2577.63	537.67	2477.38	2482.45	558.67	1694.97	1697.53	456.00	-4.98
	0.5	2578.41	2584.23	517.67	2476.49	2480.26	484.33	1696.26	1696.30	454.33	-3.40
	0.7	**2575.99**	**2583.97**	**555.67**	**2474.82**	**2479.97**	**482.33**	**1694.62**	**1697.48**	**538.00**	**-5.98**
	0.9	2578.38	2591.81	615.33	2476.49	2479.31	494.33	1697.68	1699.41	618.33	-2.57
(μ, ite_{max})	**(10, 1)**	**2577.20**	**2578.70**	**459.67**	**2476.12**	**2477.29**	**430.33**	**1692.22**	**1696.63**	**300.00**	**-6.40**
	(10, 2)	2585.22	2589.46	519.67	2478.85	2482.84	578.33	1693.26	1698.00	557.33	-1.57
	(20, 1)	2578.01	2580.88	692.67	2476.12	2480.06	635.33	1693.24	1695.31	493.33	-5.48
	(20, 2)	2580.01	2586.31	559.00	2483.26	2483.89	620.67	1696.13	1697.34	537.00	-0.13
	(40, 1)	2579.17	2584.75	787.33	2476.12	2477.76	843.67	1696.55	1698.78	717.00	-3.08
$(ratio_{beg}, ratio_{end})$	(0.7, 0.3)	2580.23	2585.00	605.67	2476.12	2478.92	599.33	1694.30	1698.29	414.00	-4.00
	(0.6, 0.4)	2577.40	2582.39	549.33	2479.24	2484.17	471.67	1696.39	1698.49	422.33	-2.61
	(0.6, 0.3)	2582.37	2588.81	553.33	2475.20	2479.26	588.00	1697.92	1701.39	513.33	-1.40
	(0.6, 0.2)	2577.22	2584.49	499.33	2475.20	2479.40	442.00	1696.11	1699.52	287.00	-4.47
	(0.5, 0.2)	**2578.48**	**2582.43**	**601.00**	**2475.20**	**2480.04**	**624.67**	**1691.94**	**1695.51**	**385.33**	**-6.44**

is not worse than that of the state-of-the-art exact algorithms on all the instances. Specifically, GATS-PR improved BKS on four instances p09, p10, p11 and pr05. In terms of average Gap_{best} and t_{avg}, GATS-PR obtains significantly better results than the reference algorithms in less than 30% run time of Model, which is the most efficient exact algorithm among the reference algorithms. Hence, GATS-PR outperforms these state-of-the-art exact algorithms.

Tables 7 and 8 report f_{best}, f_{avg} and t_{avg} of GATS-PR and the state-of-the-art metaheuristic algorithms.

- In terms of f_{best}, MNS-TS obtains 6 worse solutions than GATS-PR and its average Gap_{best} (i.e. 0.43) is worse than that of GATS-PR (i.e. -0.60). Except for controversial results produced by MNS-TS on instances p12, p15 and p18, GATS-PR obtains better f_{best} than

Table 4
Experimental environment of the state-of-the-art exact algorithms.

Name	CPU	Score (MOps/Sec)
IMIP [16]	Intel Dual Core @ 3.50 GHz	1954
TIVF [10]	Intel Dual Core @ 3.50 GHz	1954
TIMIP [11]	Intel Core i7 @ 3.70 GHz	2759
Model [12]	Intel Core i5-6300 @ 2.40 GHz	1676

Table 5
Experimental environment of the state-of-the-art heuristic algorithms.

Name	Num	CPU	Score (MOps/Sec)
HGA [8]	20	Intel Dual Core @ 3.2 GHz	550
MNS-TS [26]	1	Intel Core i7-2600 @ 3.40 GHz	1742
MBILSA [27]	1	Intel Core i7-3820 @ 3.60 GHz	1739
VTNS [28]	25	Intel Core i7-8700 @ 3.20 GHz	2657
MILS [12]	15	Intel Core i5-6300 @ 2.40 GHz	1676
TS [29]	1	-	-

Table 6
The comparison of the solution quality of GATS-PR and the state-of-the-art exact algorithms.

Ins	BKS	Ref	f_{best}					t_{avg}				
			IMIP	TIVF	TIMIP	Model	GATS-PR	IMIP	TIVF	TIMIP	Model	GATS-PR
p01	386.18*	HGA	**386.18**	**386.18**	**386.18**	**386.18**	**386.18**	50.2	20.91	0.95	2.54	0.00
p02	375.93*	HGA	**375.93**	**375.93**	**375.93**	**375.93**	**375.93**	0.74	0.44	0.21	0.64	0.00
p03	474.57*	HGA	**474.57**	**474.57**	**474.57**	**474.57**	**474.57**	50.2	13.53	1.44	3.82	1.00
p04	662.22	IMIP	**662.22**	**662.22**	**662.22**	**662.22**	**662.22**	5339.2	5339.2	7538.82	4579.58	4.00
p05	607.53*	TIVF	608.73	**607.53**	**607.53**	**607.53**	**607.53**	5339.2	33.81	76.38	22.9	8.00
p06	611.99*	TIVR	**611.99**	**611.99**	**611.99**	**611.99**	**611.99**	5339.2	519.09	431.35	321.21	12.00
p07	608.28*	TIVF	613.96	**608.28**	**608.28**	**608.28**	**608.28**	5339.2	509.45	1079.96	494.21	52.67
p08	2776.12	VTNS	3156.89	2898.51	2870.21	2853.57	2776.35	5339.2	5339.2	7538.82	4579.58	2473.00
p09	2578.49	MBILSA	2795.43	2627.28	2660.45	2595.80	2572.46	5339.2	5339.2	7538.82	4579.58	1230.33
p10	2482.32	VTNS	2704.64	2505.37	2528.98	2488.77	2474.82	5339.2	5339.2	7538.82	4579.58	2798.00
p11	2465.28	Model	2844.66	2516.60	2499.25	**2465.28**	2451.33	5339.2	5339.2	7538.82	4579.58	1108.33
p12	953.26*	HGA	**953.26**	**953.26**	**953.26**	**953.26**	**953.26**	0.54	0.56	0.48	1.27	0.00
p15	1885.81*	TIVF	*1883.53*[1]	**1885.81**	**1885.81**	**1885.81**	**1885.81**	31.67	13.39	3.8	11.45	0.33
p18	2818.36*	IMIP	**2818.36**	**2818.36**	**2818.36**	**2818.36**	**2818.36**	278.75	13.73	15.37	41.34	2.33
pr01	647.03*	HGA	**647.03**	**647.03**	**647.03**	**647.03**	**647.03**	0.76	0.37	0.15	0.64	0.00
pr02	979.82*	IMIP	**979.82**	**979.82**	**979.82**	*978.82*[2]	**979.82**	16.26	4.23	2.06	3.82	0.67
pr03	1423.48*	IMIP	**1423.48**	**1423.48**	**1423.48**	**1423.48**	**1423.48**	5339.2	28.7	*17.63*	23.53	21.67
pr04	1514.07*	TIMIP	1522.52	1521.69	**1514.07**	**1514.07**	**1514.07**	5339.2	5339.2	1819.98	866.3	28.00
pr05	1697.99	Model	1802.71	1711.68	1716.02	**1697.99**	1691.94	5339.2	5339.2	7538.82	4579.58	843.67
pr06	1976.47	Model	2221.94	2014.69	1978.46	**1976.47**	**1976.47**	5339.2	5339.2	7538.82	4579.58	2125.00
pr07	821.25*	HGA	**821.25**	**821.25**	**821.25**	**821.25**	**821.25**	2.8	1.89	0.31	0.64	0.00
pr08	1254.45*	IMIP	**1254.45**	**1254.45**	**1254.45**	**1254.45**	**1254.45**	5339.2	37.15	8.85	20.99	6.67
pr09	1591.78*	TIVF	1659.39	**1591.78**	**1591.78**	**1591.78**	**1591.78**	5339.2	5339.2	341.57	526.65	64.33
pr10	1968.67	Model	2295.01	2051.59	1997.96	**1968.67**	**1968.67**	5339.2	5339.2	7538.82	4579.58	579.00
Better			1	0	0	1	0	0	0	1	0	0
Equal			12	16	17	20	24	0	0	0	0	24
Worse			11	8	7	3	0	24	24	23	24	0
Ave			36.44	6.99	5.20	1.51	-0.60	3355.00	2274.55	2671.29	1624.11	473.29

[1] The same author reported that the objective function value of the optimal solution on p15 is 1885.81 in the subsequent paper [11].
[2] The objective function value of the optimal solution on pr02 is proved to be 979.82 in [11] and [16].

HGA, MNS-TS, MBILSA, MILS and TS on all the instances and the values of *Worse* exceed 15 for these algorithms. It indicates that GATS-PR obtains more high-quality solutions than the state-of-the-art metaheuristic algorithms.

- In terms of f_{avg}, GATS-PR obtains significantly better results than HGA and VTNS in terms of average Gap_{avg}. Specifically, not only we improved BKS on instances p09, p10, p11 and pr05, but also our f_{avg} on these instances was better than BKS. Although VTNS obtains a slightly better solution than GATS-PR on instance p08 in terms of f_{best}, its f_{avg} (i.e. 3036.02) is significantly worse than that of GATS-PR (i.e. 2778.26). It indicates that GATS-PR can obtain high-quality solutions frequently and GATS-PR is quite robust.
- In terms of t_{avg}, no reference algorithms can obtain better results in less time than GATS-PR. The values of *Worse* are 16, 8, 15, 16 and 13 for HGA, MNS-TS, MBILSA, VTNS and MILS. It shows that GATS-PR is more efficient than them. Although GATS-PR spends a lot of time on large-scale instances, such as p08 and p10, it indicates that GATS-PR has better search capability than the reference algorithms because GATS-PR can obtain better results than them.

Table 7
The comparison of the results of GATS-PR and the state-of-the-art metaheuristic algorithms.

Ins	BKS	Ref	f_{best}							f_{avg}		
			HGA	MNS-TS	MBILSA	VTNS	MILS[3]	TS	GATS-PR	HGA	VTNS	GATS-PR
p01	386.18*	HGA	**386.18**	**386.18**	**386.18**	**386.18**	**386.18**	386.69	**386.18**	388.82	386.95	**386.18**
p02	375.93*	HGA	**375.93**	376.78	376.44	**375.93**	**375.93**	377.76	**375.93**	376.90	376.04	**375.93**
p03	474.57*	HGA	**474.57**	**474.57**	475.79	**474.57**	485.20	481.56	**474.57**	477.69	474.73	**474.57**
p04	662.22	IMIP	667.02	670.13	**662.22**	**662.22**	689.14	686.55	**662.22**	669.49	663.83	**662.22**
p05	607.53*	TIVF	612.30	609.81	609.04	**607.53**	623.03	617.87	**607.53**	616.35	608.88	**607.53**
p06	611.99*	TIVR	614.93	616.90	**611.99**	**611.99**	628.72	631.69	**611.99**	621.13	614.60	**611.99**
p07	608.28*	TIVF	615.24	615.98	609.60	**608.28**	631.60	626.11	**608.28**	623.91	610.82	**608.28**
p08	2776.12	VTNS	2,913.63[1]	2852.56	2794.10	***2776.12***	3027.15	-	2776.35	2,911.51[1]	3036.02	2778.26
p09	2578.49	MBILSA	2,738.54	2625.04	2582.65	2587.04	3049.10	-	2572.46	2,755.28	2619.16	2573.43
p10	2482.32	VTNS	2,577.48	2539.60	2500.43	**2482.32**	2730.14	-	2474.82	2,616.02	2500.41	2475.25
p11	2465.28	Model	2,561.85	2508.45	2478.31	2472.85	2707.01	-	2451.33	2,598.56	2487.38	2456.92
p12	953.26*	HGA	**953.26**	*953.25*[2]	**953.26**	**953.26**	1021.16	**953.26**	**953.26**	955.80	**953.26**	**953.26**
p15	1885.81*	TIVF	1,888.67	*1885.80*[2]	**1885.81**	**1885.81**	2044.42	1932.51	**1885.81**	1,899.98	**1885.81**	**1885.81**
p18	2818.36*	IMIP	2,830.78	*2818.34*[2]	**2818.36**	**2818.36**	2939.89	-	**2818.36**	2,854.81	**2818.36**	**2818.36**
pr01	647.03*	HGA	**647.03**	**647.03**	**647.03**	**647.03**	768.43	**647.03**	**647.03**	**647.03**	**647.03**	**647.03**
pr02	979.82*	IMIP	981.63	**979.82**	**979.82**	**979.82**	992.41	1004.84	**979.82**	985.69	**979.82**	**979.82**
pr03	1423.48*	IMIP	1,445.65	1,426.95	1429.38	**1423.48**	1475.74	1451.17	**1423.48**	1,458.75	1426.68	**1423.48**
pr04	1514.07*	TIMIP	1,548.93	1,517.80	1524.18	**1514.07**	1637.30	-	**1514.07**	1,586.67	1519.49	**1514.07**
pr05	1697.99	Model	1,746.57	1,758.51	1700.93	1699.40	1915.10	-	1691.94	1,755.26	1705.98	1691.94
pr06	1976.47	Model	2,069.01	2009.97	1992.26	1982.13	2152.19	-	**1976.47**	2,085.62	1992.32	1976.98
pr07	821.25*	HGA	**821.25**	**821.25**	**821.25**	**821.25**	839.35	830.78	**821.25**	824.09	**821.25**	**821.25**
pr08	1254.45*	IMIP	1,266.87	1,266.17	1258.64	**1254.45**	1325.12	1283.43	**1254.45**	1277.79	1255.40	**1254.45**
pr09	1591.78*	TIVF	1,643.56	1,615.42	1592.77	**1591.78**	1694.34	-	**1591.78**	1666.02	1592.83	**1591.78**
pr10	1968.67	Model	2,037.22	2013.86	1975.36	1969.35	2186.16	-	**1968.67**	2076.08	1979.00	**1968.67**
Better			0	3	0	1	0	0	0	0	0	0
Equal			6	5	9	17	2	2	24	1	6	24
Worse			18	16	15	6	22	22	0	23	18	0
Ave			17.48	9.56	2.38	0.43	67.17	-[4]	-0.60	25.89	6.94	-0.45

[1] These two values are probably recorded in reverse in [8].
[2] The results are better than the optimal ones proved by [10] and [11].
[3] We round the results in [12] because results in other literature are only retained to two decimal places.
[4] The value is missing because TS cannot obtain feasible solutions on a lot of instances.

Table 8
The comparison of t_{avg} of GATS-PR and the state-of-the-art metaheuristic algorithms.

Ins	HGA	MNS-TS	MBILSA	VTNS	MILS	GATS-PR
p01	2.88	0.97	5.02	16.44	2.89	0.00
p02	3.03	0.81	5.74	21.77	1.66	0.00
p03	8.52	3.99	10.69	40.14	4.85	1.00
p04	17.95	3.97	20.19	15.88	67.24	4.00
p05	18.99	9.80	14.65	31.13	7.60	8.00
p06	19.08	17.35	14.52	23.51	13.10	12.00
p07	18.31	14.20	20.19	28.89	29.89	52.67
p08	227.39	36.62	97.74	18.11	78.77	2473.00
p09	230.08	39.54	100.05	99.08	85.86	1230.33
p10	233.53	32.14	76.09	188.99	83.50	2798.00
p11	232.48	56.22	87.97	252.65	85.51	1108.33
p12	10.29	0.80	6.86	42.40	4.16	0.00
p15	70.34	6.07	24.15	159.13	25.37	0.33
p18	216.7	4.40	50.16	360.23	67.42	2.33
pr01	2.73	0.37	3.30	42.84	1.29	0.00
pr02	17.72	6.00	12.61	85.69	6.34	0.67
pr03	63.10	18.74	27.85	152.21	17.72	21.67
pr04	76.81	9.31	39.93	208.32	36.40	28.00
pr05	213.82	63.43	85.73	281.21	64.06	843.67
pr06	368.93	45.01	118.33	311.95	110.47	2125.00
pr07	8.50	0.78	7.79	97.58	3.43	0.00
pr08	52.54	3.72	21.38	188.57	17.74	6.67
pr09	162.50	11.47	56.56	309.50	51.32	64.33
pr10	363.38	116.00	115.16	446.58	112.33	579.80
Better	0	0	0	0	0	0
Equal	0	0	0	0	0	0
Worse	16	8	15	16	13	0

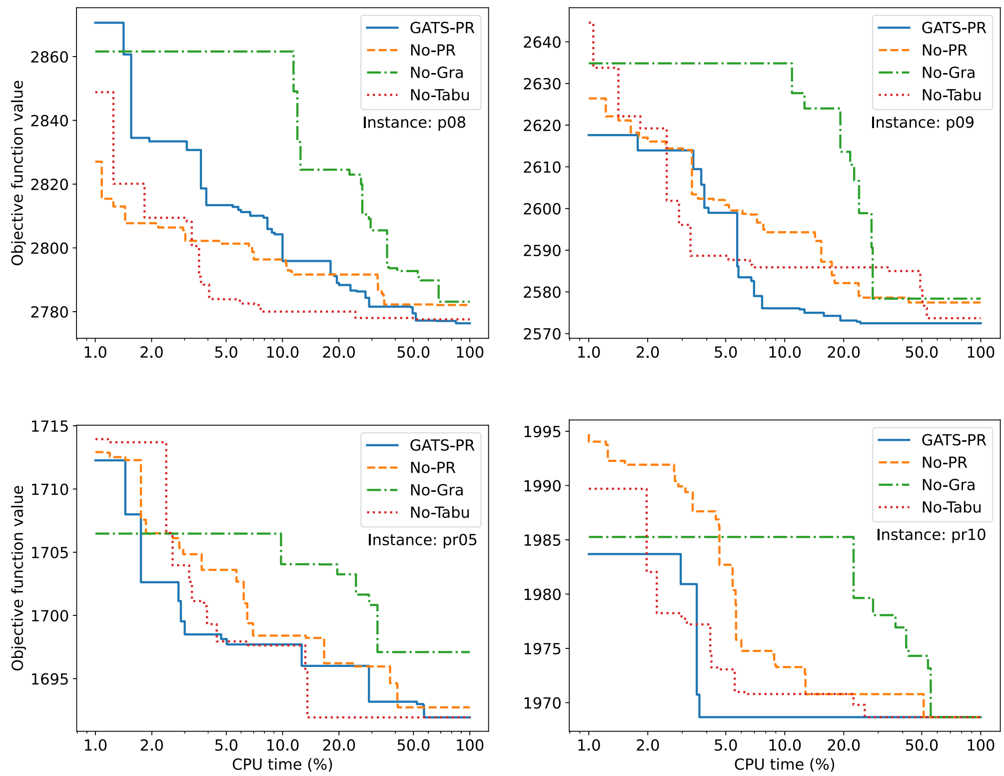

Fig. 4. Evolution process of f_{best} for GATS-PR, No-PR, No-Gra and No-Tabu over time on a logarithmic scale.

These comparisons demonstrate that GATS-PR outperforms these state-of-the-art metaheuristic algorithms for solving the MDOVRP.

5.3. Importance analysis of algorithmic components

In order to analyze the importance of the proposed algorithmic components, we disable one of them from GATS-PR at a time and compare the results obtained by the variant algorithm with those of GATS-PR. The No-PR version only uses the granular adaptive tabu search presented in Algorithm 2 to verify the effectiveness of path relinking. The No-Gra version disables granular neighborhoods, which considers fixed number of neighboring nodes of a given customer by setting both gt_{min} and gt_{max} to 0.2. The No-Tabu version chooses the move with the best improvement of the generalized function value at each iteration to validate the role of the solution-based tabu search. The No-ANS version does not use the adaptive neighborhood selection (ANS) but considers all the move types described in Fig. 2 at each iteration. The No-Group version regards each move type as a neighborhood to verify the effectiveness of grouping move types into four neighborhoods in ANS.

For GATS-PR and its variants, each instance was independently run three times and the time limit was one hour. The results of these algorithms are shown in Table 9. Although No-Tabu converges quickly, its results are significantly worse than those of GATS-PR. It indicates that the solution-based tabu search enables the search to escape from the local optima. GATS-PR obtains better results with less run time than No-PR and No-Gra. It demonstrates that path relinking and granular neighborhoods can improve the search efficiency without sacrificing the solution quality. No-Group obtains better solutions than No-ANS with less run time and GATS-PR obtains better solutions than No-Group with less time. It demonstrates that ANS can improve efficiency and effectiveness of the search and grouping multiple move types into one neighborhood contributes to ANS when the number of move types is large. Therefore, all these algorithmic components are important for GATS-PR.

In order to intuitively show the convergence of GATS-PR and its variants, Figs. 4 and 5 plot the evolution process of f_{best} obtained by each algorithm on four representative instances p08, p09, pr05 and pr10 over time on a logarithmic scale. As shown in Figs. 4 and 5, we observe that:

- Although GATS-PR obtains slightly worse solutions than No-PR and No-Tabu at the early stage of the search, GATS-PR outperforms them on instances p08 and p09, and match them on instances pr05 and pr10 at the end of the search. It indicates that GATS-PR converges slightly slower than No-PR and No-Tabu but it has a better search capability.
- After excluding solutions obtained at the earliest stage of the search which are heavily influenced by initial solutions and with certain randomness, GATS-PR outperforms No-Gra on the selected instances. It shows that the granular neighborhoods can improve the efficiency of the search without sacrificing the solution quality.
- After excluding solutions obtained at the earliest stage of the search, GATS-PR always outperforms No-ANS on these four instances. Although No-Group obtains better solutions than GATS-PR at the middle stage of the search on instance pr05, it obtains worse solutions at the end of the search. On other three instances, No-Group is outperformed by GATS-PR. These observations demonstrate the importance of ANS again.

To further explore the effectiveness of ANS on the neighborhood selection mechanism, we record the frequency of performing each move type when the best solution is obtained by No-ANS, No-Group and GATS-PR, respectively. Let $r_i, i \in N$ represent the ratio of the number of chosen move types that belong to neighborhood i to the total number of iterations. Fig. 6 presents r_i of these three algorithms on all the instances.

Table 9
Results of the GATS-PR and its variants.

Ins	f_{best}						f_{avg}						t_{avg}					
	GATS-PR	No-PR	No-Gra	No-Tabu	No-ANS	No-Group	GATS-PR	No-PR	No-Gra	No-Tabu	No-ANS	No-Group	GATS-PR	No-PR	No-Gra	No-Tabu	No-ANS	No-Group
p01	386.18	386.18	386.18	386.18	386.18	386.18	386.18	386.18	386.18	386.18	386.18	386.18	0.00	2.00	4.33	0.00	7.33	7.33
p02	375.93	375.93	375.93	375.93	375.93	375.93	375.93	375.93	375.93	375.93	375.93	375.93	0.00	1.00	4.33	0.00	1.33	5.33
p03	474.57	474.57	474.57	474.57	474.57	474.57	474.57	474.57	474.57	474.57	474.57	474.57	1.00	4.00	21.33	5.67	24.67	20.67
p04	662.22	662.22	662.22	662.22	662.22	662.22	662.22	662.22	662.43	662.22	662.22	662.22	4.00	8.33	45.33	18.67	41.33	183.67
p05	607.53	607.53	607.53	607.53	607.53	607.53	607.53	607.81	607.53	607.65	608.20	609.39	8.00	12.67	309.67	36.67	56.67	76.67
p06	611.99	611.99	611.99	611.99	611.99	611.99	611.99	611.99	612.74	611.99	613.49	612.65	12.00	15.33	32.67	6.33	24.33	72.00
p07	608.28	608.28	608.28	608.28	608.28	608.28	608.28	608.28	608.95	608.28	609.25	609.13	52.67	18.00	74.00	27.67	51.67	60.00
p08	2776.35	2782.30	2783.11	2777.58	2791.88	2782.31	2778.26	2782.31	2793.90	2781.18	2793.80	2784.67	2473.00	2491.00	3625.00	2122.00	2536.00	3233.00
p09	2572.46	2573.12	2575.27	2572.46	2576.25	2577.56	2573.43	2576.21	2575.33	2573.45	2581.16	2581.84	1230.33	357.67	536.33	876.33	1340.67	2210.33
p10	2474.82	2475.49	2475.95	2474.82	2475.49	2476.12	2475.25	2476.40	2479.54	2475.68	2476.03	2480.04	2798.00	2902.00	2997.33	1629.33	3162.00	2415.33
p11	2451.33	2451.33	2460.50	2464.55	2461.94	2454.45	2456.92	*2452.23*	2462.06	2477.88	2463.30	2457.62	1108.33	2012.67	2695.67	84.67	2869.67	1731.00
p12	953.26	953.26	953.26	953.26	953.26	953.26	953.26	953.26	953.26	953.26	953.26	953.26	0.00	0.00	0.00	0.00	0.00	1.33
p15	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	1885.81	0.33	0.33	3.67	0.67	2.00	58.33
p18	2818.36	2818.36	2818.36	2818.36	2818.36	2818.36	2818.36	2818.36	2818.36	2818.36	2818.36	2821.12	2.33	1.33	14.33	1.67	6.67	44.33
pr01	647.03	647.03	647.03	647.03	647.03	647.03	647.03	647.03	647.03	647.03	647.03	647.03	0.00	0.00	1.00	0.00	1.00	0.00
pr02	979.82	979.82	979.82	979.82	979.82	979.82	979.82	980.36	979.82	979.82	980.34	979.82	0.67	6.67	23.67	8.33	14.33	29.33
pr03	1423.48	1423.48	1423.48	1423.48	1423.48	1423.48	1423.48	1423.48	1423.48	1423.48	1423.48	1424.79	21.67	43.67	189.67	31.67	224.33	77.33
pr04	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	1514.07	28.00	2032.33	2880.67	663.33	1712.00	1096.33
pr05	1691.94	1692.22	1691.94	1691.94	1693.50	1692.44	1691.94	1694.68	1693.54	1693.59	1696.13	1694.05	843.67	1407.67	1750.00	2068.33	1538.33	1466.67
pr06	1976.47	1976.47	1977.41	1976.47	1977.41	1977.07	1976.98	1977.03	1980.62	1977.09	1978.28	1978.54	2125.00	2079.67	2955.67	793.00	3009.33	1697.00
pr07	821.25	821.25	821.25	821.25	821.25	821.25	821.25	821.25	821.25	821.25	821.25	821.25	0.00	0.00	3.67	0.00	1.67	1.33
pr08	1254.45	1254.45	1254.45	1254.45	1254.45	1254.45	1254.45	1254.54	1255.07	1254.45	1255.31	1254.45	6.67	28.00	119.33	13.67	71.00	35.33
pr09	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	1591.78	64.33	10.33	38.33	62.67	345.00	154.67
pr10	1968.67	1968.67	1968.67	1968.67	1969.79	1968.67	1968.67	1968.67	1968.67	1968.67	1970.50	1968.67	579.00	808.67	838.33	527.67	1136.33	223.67
Ave	-0.60	-0.49	-0.26	-0.36	-0.04	-0.33	-0.45	-0.29	0.22	0.01	0.47	0.27	473.29	593.47	798.51	374.10	757.40	620.87

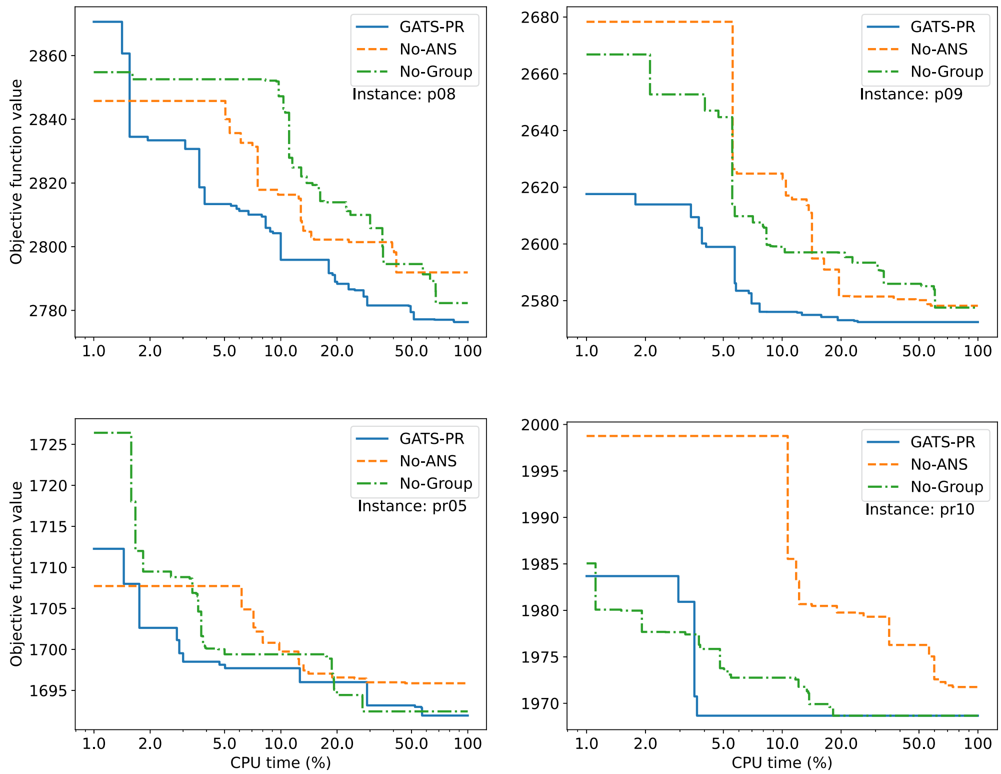

Fig. 5. Evolution process of f_{best} for GATS-PR, No-ANS and No-Group over time on a logarithmic scale.

- For No-ANS, r_{N_1} exceeds 50% on 19 out of 24 instances and it exceeds 70% on 10 ones. For GATS-PR, although N_1 is also the most frequently performed neighborhood except for instance p02, r_{N_1} is only about 45%. Because GATS-PR obtains better solutions than No-ANS, it indicates that ANS diversifies the search at the suitable time by selecting neighborhoods N_3 and N_4 more frequently than No-ANS to help the search to escape from the local optima.
- For No-Group, the sum of r_{N_1} and r_{N_2} exceeds 45% on 12 out of 24 instances and it is below 50% on all the instances. However, it exceeds 50% on 19 out of 24 instances for GATS-PR. Because GATS-PR obtains better solutions than No-Group, it indicates that when the number of move types is large, grouping multiple move types into one neighborhood intensifies the search and thus enhances the search capability.

6. Conclusion and future research

In this paper, we propose the GATS-PR algorithm which hybridizes granular neighborhoods, adaptive neighborhood selection and solution-based tabu search with path relinking to solve the MDOVRP. This is the first time that path relinking is used for the MDOVRP to the best of our knowledge and a new similarity definition is designed for this problem. We develop the adaptive neighborhood selection strategy to select suitable move types from a total of 22 ones and exclude unpromising neighborhood solutions by the granular neighborhoods. By forbidding accessing visited solutions, the solution-based tabu search can further enhance diversification of the search.

GATS-PR improved and matched the previous best known results on 4 and 19 out of 24 instances, respectively. By comparing the results obtained by GATS-PR with those of the state-of-the-art algorithms, we can observe that GATS-PR outperforms them for solving the MDOVRP. Importance analysis of the proposed algorithmic components demonstrates that granular neighborhoods, adaptive neighborhood selection, solution-based tabu search and path relinking all play essential roles in obtaining high-quality solutions efficiently.

In future research, we intend to apply GATS-PR to other VRP-related problems, such as the multi-depot VRP and the multi-depot green VRP. The advanced population management mechanism is another topic that deserves to be investigated. We are also going to design other similarity definitions between solutions and integrate them to improve the search capability of path relinking to solve the MDOVRP.

CRediT authorship contribution statement

Wenhan Shao: Investigation, Conceptualization, Methodology, Data collection, Software, Validation, Visualization, Writing – original draft, Writing – review and editing. **Tuanyue Xiao:** Investigation, Methodology, Data collection, Software, Validation, Writing – original draft, Writing – review and editing. **Zhouxing Su:** Conceptualization, Methodology, Writing – review and editing, Funding acquisition. **Junwen Ding:** Conceptualization, Methodology, Writing – review and editing, Supervision, Project administration, Funding acquisition. **Zhipeng Lü:** Conceptualization, Methodology, Writing – review and editing, Funding acquisition.

Declaration of competing interest

The authors declare that there is no competing interest.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants 72101094 and 62202192, and the Special Project for Knowledge Innovation of Hubei Province under Grant 2022013301015175.

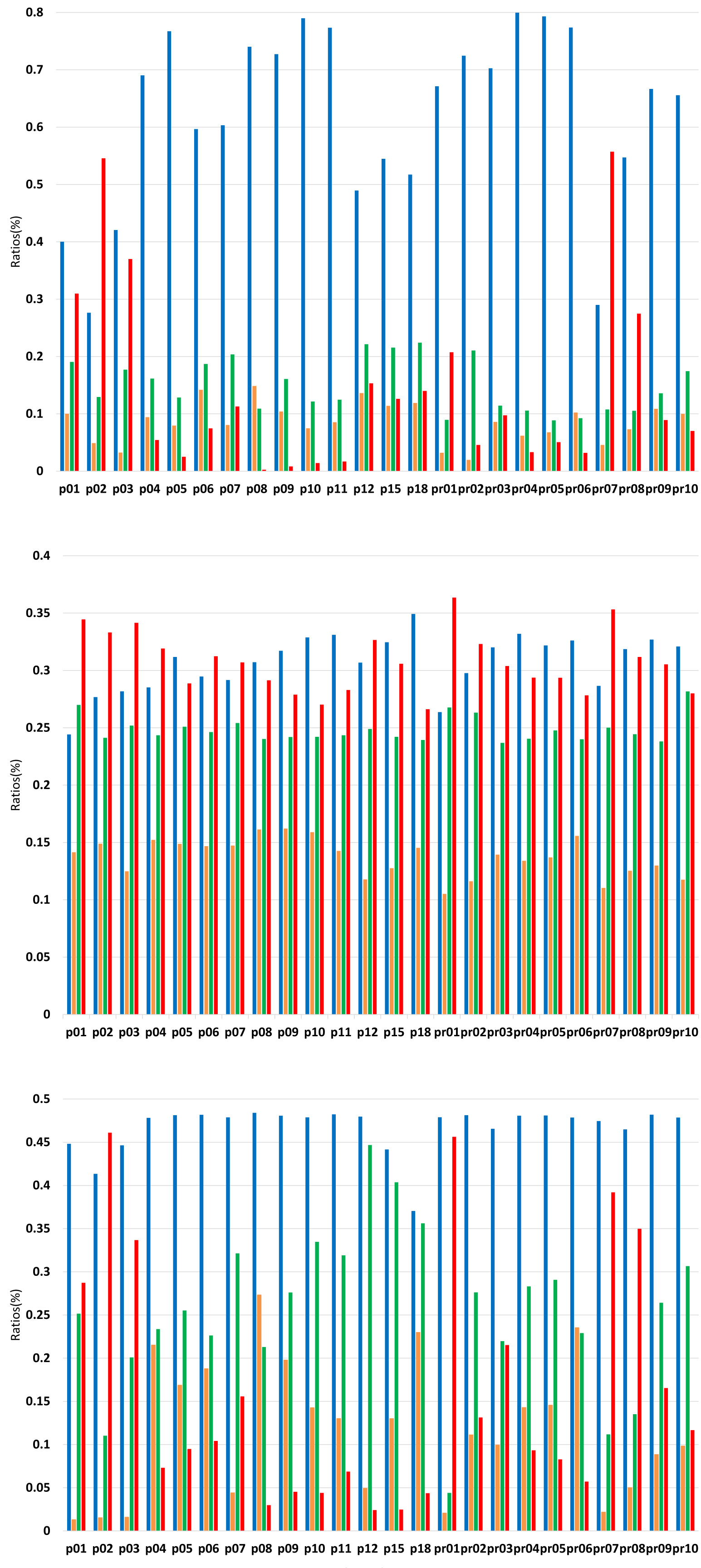

Fig. 6. Statistics on the ratios of the neighborhood selection of No-ANS, No-Group and GATS-PR.

References

[1] Dantzig GB, Ramser JH. The truck dispatching problem. Manag Sci 1959;6(1):80–91.
[2] Xidias E, Zacharia P, Nearchou A. Intelligent fleet management of autonomous vehicles for city logistics. Appl Intell 2022;52(15):18030–48.
[3] Zou G, Tang J, Yilmaz L, Kong X. Online food ordering delivery strategies based on deep reinforcement learning. Appl Intell 2022;52(6):6853–65.
[4] Alitappeh RJ, Jeddisaravi K. Multi-robot exploration in task allocation problem. Appl Intell 2022;52(2):2189–211.
[5] Kaabi J, Harrath Y, Mahjoub A, Hewahi N, Abdulsattar K. A 2-phase approach for planning of hazardous waste collection using an unmanned aerial vehicle. 4OR Nov. 2022.
[6] Vidal T, Crainic TG, Gendreau M, Lahrichi N, Rei W. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. Oper Res 2012;60(3):611–24.
[7] Tarantilis CD, Kiranoudis CT. Distribution of fresh meat. J Food Eng 2002;51(1):85–91.
[8] Liu R, Jiang Z, Geng N. A hybrid genetic algorithm for the multi-depot open vehicle routing problem. OR Spektrum 2014;36(2):401–21.
[9] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. USA: W. H. Freeman & Co.; 1990.
[10] Granada M, Toro EM, Gallego R. An MIP formulation for the open location-routing problem considering the topological characteristic of the solution-paths. Networks 2019;74(4):374–88.
[11] Lalla-Ruiz E, Mes M. Mathematical formulations and improvements for the multi-depot open vehicle routing problem. Optim Lett 2021;15(1):271–86.
[12] Nucamendi-Guillén S, Padilla AG, Olivares-Benitez E, Moreno-Vega JM. The multi-depot open location routing problem with a heterogeneous fixed fleet. Expert Syst Appl 2021;165:113846.
[13] Lambert M, Hassani R. Diesel genset optimization in remote microgrids. Appl Energy 2023;340:121036.
[14] Xiong Z, Zhao M, Tan L, Cai L. Real-time power optimization for application server clusters based on mixed-integer programming. Future Gener Comput Syst 2022;137:260–73.
[15] Vardhan BVS, Khedkar M, Srivastava I. Effective energy management and cost effective day ahead scheduling for distribution system with dynamic market participants. Sustain Energy Grids Netw 2022;31:100706.
[16] Lalla-Ruiz E, Expósito-Izquierdo C, Taheripour S, Voß S. An improved formulation for the multi-depot open vehicle routing problem. OR Spektrum 2016;38(1):175–87.
[17] Wang H, Li R, Gong W. Minimizing tardiness and makespan for distributed heterogeneous unrelated parallel machine scheduling by knowledge and Pareto-based memetic algorithm. Egypt Inform J 2023;24(3):100383.
[18] Cui Z, Li B, Lan Z, Xu Y. Many-objective evolutionary algorithm based on three-way decision. Egypt Inform J 2023;24(3):100388.
[19] Zakaryia SA, Ahmed SA, Hussein MK. Evolutionary offloading in an edge environment. Egypt Inform J 2021;22(3):257–67.
[20] Utamima A. A comparative study of hybrid estimation distribution algorithms in solving the facility layout problem. Egypt Inform J 2021;22(4):505–13.
[21] Lin A, Li S, Liu R. Mutual learning differential particle swarm optimization. Egypt Inform J 2022;23(3):469–81.
[22] Chen W-N, Tan D-Z. Set-based discrete particle swarm optimization and its applications: a survey. Front Comput Sci 2018;12(2):203–16.
[23] Yang W, Ke L. An improved fireworks algorithm for the capacitated vehicle routing problem. Front Comput Sci 2019;13(3):552–64.
[24] Liu G, Guo W, Li R, Niu Y, Chen G. XGRouter: high-quality global router in X-architecture with particle swarm optimization. Front Comput Sci 2015;9(4):576–94.
[25] Liu S, Huang W, Ma H. An effective genetic algorithm for the fleet size and mix vehicle routing problems. Transp Res, Part E, Logist Transp Rev 2009;45(3):434–45.
[26] Soto M, Sevaux M, Rossi A, Reinholz A. Multiple neighborhood search, tabu search and ejection chains for the multi-depot open vehicle routing problem. Comput Ind Eng 2017;107:211–22.
[27] Brandão J. A memory-based iterated local search algorithm for the multi-depot open vehicle routing problem. Eur J Oper Res 2020;284(2):559–71.
[28] Sadati MEH, Çatay B, Aksen D. An efficient variable neighborhood search with tabu shaking for a class of multi-depot vehicle routing problems. Comput Oper Res 2021;133:105269.
[29] Abdzadeh B, Noori S, Ghannadpour SF. Simultaneous scheduling of multiple construction projects considering supplier selection and material transportation routing. Autom Constr 2022;140:104336.
[30] Toth P, Vigo D. The granular tabu search and its application to the vehicle-routing problem. INFORMS J Comput 2003;15(4):333–46.
[31] Glover F. Future paths for integer programming and links to artificial intelligence. Comput Oper Res 1986;13(5):533–49.
[32] Escobar JW, Linfati R, Baldoquin MG, Toth P. A granular variable tabu neighborhood search for the capacitated location-routing problem. Transp Res, Part B, Methodol 2014;67:344–56.
[33] Schneider M, Schwahn F, Vigo D. Designing granular solution methods for routing problems with time windows. Eur J Oper Res 2017;263(2):493–509.
[34] Shahmanzari M, Aksen D. A multi-start granular skewed variable neighborhood tabu search for the roaming salesman problem. Appl Soft Comput 2021;102:107024.
[35] Máximo VR, Nascimento MCV. A hybrid adaptive iterated local search with diversification control to the capacitated vehicle routing problem. Eur J Oper Res 2021;294(3):1108–19.
[36] Sörensen K, Schittekat P. Statistical analysis of distance-based path relinking for the capacitated vehicle routing problem. Comput Oper Res 2013;40(12):3197–205.
[37] Schneider M, Löffler M. Large composite neighborhoods for the capacitated location-routing problem. Transp Sci 2019;53(1):301–18.