Journal of Systems Engineering and Electronics
Vol. 33, No. 4, August 2022, pp.997 – 1009

Solving open vehicle problem with time window by hybrid column generation algorithm

YU Naikang[1,2], QIAN Bin[2,3,*], HU Rong[2,3], CHEN Yuwang[4], and WANG Ling[5]

1. School of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming 650500, China;
2. School of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China;
3. Yunnan Key Laboratory of Artificial Intelligence, Kunming University of Science and Technology, Kunming 650500, China;
4. Alliance Manchester Business School, University of Manchester, Manchester M139SS, U.K.;
5. Department of Automation, Tsinghua University, Beijing 100084, China

Abstract: This paper addresses the open vehicle routing problem with time window (OVRPTW), where each vehicle does not need to return to the depot after completing the delivery task. The optimization objective is to minimize the total distance. This problem exists widely in real-life logistics distribution process. We propose a hybrid column generation algorithm (HCGA) for the OVRPTW, embedding both exact algorithm and metaheuristic. In HCGA, a label setting algorithm and an intelligent algorithm are designed to select columns from small and large subproblems, respectively. Moreover, a branch strategy is devised to generate the final feasible solution for the OVRPTW. The computational results show that the proposed algorithm has faster speed and can obtain the approximate optimal solution of the problem with 100 customers in a reasonable time.

Keywords: open vehicle routing problem with time window (OVRPTW), hybrid column generation algorithm (HCGA), mixed integer programming, label setting algorithm.

DOI: 10.23919/JSEE.2022.000096

1. Introduction

As an indispensable part of modern enterprises, the logistics system directly affects the profits of enterprises. Companies not only have to spend on employee and vehicle arrangements but also consider the various costs of vehicle distribution. Therefore, the research on the vehicle routing problem (VRP) in the logistics distribution system has received extensive attention from the academic community. Since Dantzig et al. [1] proposed this problem, the vehicle routing problem has been a key research direction in the field of combinatorial optimization, and its related research has always been paid attention to by relevant researchers. VRP is usually described as a group of depot points and customer points, and the arrangement of group of fleets to provide services to all current customers in a given order under certain constraints such as the vehicle load and the time window. The optimization goal of VRP is usually traveling cost or time. In practical applications, there are various types of VRP problems, including the capacity vehicle routing problem (CVRP), the vehicle routing problem with times windows (VRPTW), and the vehicle routing problem with pick-up and deliveries (VRPPD). In the CVRP, the fleet needs to return to the depot node after serving the last customer. For the variant of the open vehicle routing problem (OVRP), the vehicle does not need to return to the depot node after serving the last customer.

Open vehicle routing problem with time windows (OVRPTW) is an important variant type that can be formally expressed as follows: Given a central depot and a group of customers, they can be represented by a completely directed graph $G=(V,E)$, where $V=\{0,1,\cdots,n\}$, node 0 represents the central depot, $V\backslash\{0\}$ represents customers, and each edge has a cost $c_e>0$; each customer point has a demand $q_v>0$ ($q_0=0$), a time window $[e_i,l_i]$ represents the time range allowed by the customer to serve and s_i represents the time for serving the customer. There are k homogeneous vehicles in the central depot, and each vehicle has its own capacity Q. All customers $V\backslash\{0\}$ must be served once by one of the m vehicles. The OVRPTW is a strong non-deterministic polynomial (NP)-hard problem because it generalizes the Hamiltonian path and knapsack problem [2,3], which is known as an NP-hard problem.

To solve the OVRPTW, we propose a hybrid column generation algorithm (HCGA), which is a hybrid frame-

Manuscript received November 30, 2020.
*Corresponding author.
This work was supported by the National Natural Science Foundation of China (61963022; 51665025; 61873328) .

work that combines the column generation method, a rule-based label setting algorithm, a particle swarm optimization (PSO) algorithm and a branch strategy. The column generation method is utilized to solve the linear programming relaxation (LPR) of a set partitioning model of the considered problem. Accordingly, the arising subproblem of LPR consists of a so-called resource-constrained shortest path problem (RCSPP), whose objective is to minimize the reduced cost. Since the VRP is already NP-hard and it reduces to the RCSPP, the latter is also NP-hard. In order to reasonably determine columns from the RCSPP, a label setting algorithm based on the dynamic programming program scheme [4] and a PSO algorithm are designed to solve the small and large RCSPPs, respectively. Then, a branch strategy is specially devised to further transform the LPR's solution obtained by the column generation method into a feasible and satisfactory solution of the OVRPTW.

The rest of this paper is organized as follows: In Section 2, the relevant literature is briefly reviewed. In Section 3, the definition of the OVRPTW is introduced and a corresponding integer programming model is also proposed. In Section 4, the set partitioning model of the OVRPTW is established, and the HCGA is proposed and described in detail. In Section 5, experimental results and comparisons are presented and discussed. Finally, we end the paper with some conclusions and future work in Section 6.

2. Literature review

The OVRP was first described by Schrage [5], but no corresponding algorithm was given. Sariklis et al. [6] used a two-stage heuristic algorithm to solve OVRP. In the first stage, customers are divided into groups, and the rules are used to allocate customers in the customer group again. The second stage is to transform the route of each customer group into an open one and optimize the route with the minimum spanning tree algorithm. Brandao [2] used the basic tabu search (TS) algorithm to solve the OVRP and explores the structure of the solution of the OVRP, so that the algorithm can improve the quality of the solution in a short time, which provides a benchmark for other heuristic algorithms. However, the research on OVRPTW is limited. Schopka et al. [7] adopted the improved adaptive large neighborhood search (ALNS) algorithm to solve the OVRP problem with time window constraints. Ge et al. [8] proposed an improved TS algorithm to solve the OVRP with soft time windows, and Repoussis et al. [9] proposed a greedy per production path construction heuristic algorithm to solve OVRPTW. At present, most of the solutions for OVRPTW are focused on the heuristic algorithm, and the exact algorithm for solving this problem has not attracted the attention of relevant scholars.

As an exact algorithm, the column generation algorithm (CGA) has been widely used to accurately solve large-scale combinatorial optimization problems. Since the CGA was first applied by Ford et al. [10], it has been widely used to solve various combinatorial optimization problems, such as the production scheduling problem [11], the surgical scheduling problem [12], cutting stock problems [13], and so on. CGA is also one of the most frequently used exact algorithms for solving vehicle routing problems. Cacchiani et al. [14] proposed a set partitioning formula for the periodic vehicle routing problem (PVRP), and solved the linear relaxation model by the CGA. In the subproblem, columns are generated by the iterative local search algorithm, and the tabu list is used to avoid loop. Aiming at the vehicle routing and scheduling problem with semi-soft time windows (VRPSSTW), a basic shortest path problem with resource constraints and delay penalty conditions was proposed [15]. The modified forward insertion heuristic algorithm was used to solve the subproblem, which provided better columns for the main problem, so as to accelerate the convergence speed of the algorithm. However, due to the particularity of the vehicle routing problem, the sub-problem is also described as a mixed integer programming, so in each iteration of the algorithm, it takes too much time using commercial solvers such as CPLEX and GUROBI. Yuan et al. [16] proposed a set partitioning model for the generalized vehicle routing problem, and proposed a heuristic algorithm based on column generation. This mathematical method relies on constructing heuristics, path optimization process, local search operators and heuristic process to provide negative cost reducing paths. Behnke et al. [17] proposed a heuristic column generation method and an exact branching and pricing method for the emission oriented VRP (EVRP) in the green logistics, and designed a label correction algorithm for the problem. In order to speed up the pricing, an improved double programming formulation is proposed, which reduces the number of iterations and the number of generated columns. It is a mainstream research direction to redescribe the subproblem as the shortest path problem with limited resources, and to use label setting algorithm [4,18] and label correction algorithm [19]. Ozbaygin et al. [20] proposed an improved labeling algorithm to solve the subproblem for the vehicle routing problem with roaming delivery locations. Li et al. [21] proposed a branch pricing cutting algorithm based on a strong set partition model for the vehicle routing problem with pick-up and delivery, in which an ad-hoc label setting algorithm was designed to solve the pricing problem. While

Timo et al. [22−24] studied the two-way label setting algorithm. To solve the subproblem, Ke et al. [25] used the variable neighborhood search (VNS) to provide high-quality columns and Roman et al. [26] used the method of machine learning to solve the pricing subproblem.

At present, the research on the OVRP is mostly seen in heuristic algorithms [27,28]. The research on the CGA has not attracted the attention of related scholars. In the existing research, Majid et al. [29] proposed a compound called SISEC heuristic algorithms, including scanning algorithms, insertion, exchange, and 2-opt movement, which improved elite ant colony system (EAS) and the CGA, cannot guarantee the quality of accurate solutions due to heuristic operations. Faiz et al. [30] proposed two different models for the open vehicle pickup and delivery problem. The first model was solved by the accurate method, while the second model was solved by the fast vehicle path generation algorithm combined with the column generation algorithm. The research on the precise algorithm of OVRPTW has not attracted the attention of related scholars.

3. Problem description

3.1 Description of OVRPTW

The OVRPTW can be described as a homogeneous fleet located in the central depot and a group of customers with deterministic demand and service time in a predetermined geographical location. The goal of OVRPTW is to build a route from the central depot to the customer point to meet customer needs. Each vehicle only visits one customer at a time, and each customer is allowed to be served by one vehicle only once. Each feasible path requires the vehicle to visit the customer within the time window specified by the customer. The total customer demand within the path does not exceed the vehicle's capacity. As shown in Fig. 1, there are three homogeneous fleets in the depot with a capacity of 200. Each customer point has three labels, which are demand, time window, and service time. t is the cumulative total time of vehicles at the customer point, and it is also the earliest time to visit the next customer point. T represents the total time at the customer point. Assume that the speed is constant. The route distance is also the time spent. Path 1 is feasible, and its customer's total demand is 115, with 40+10+60+5<200, and the vehicle access service within the time window of the customer point. Path 2 total demand is 230, with 90+50+90>200, violating the weight constraint. Path 3 does not violate the weight constraint. However, when visiting the last customer point, T = 150 is not within the time window of the last customer. Therefore, Path 2 and Path 3 are illegal paths.

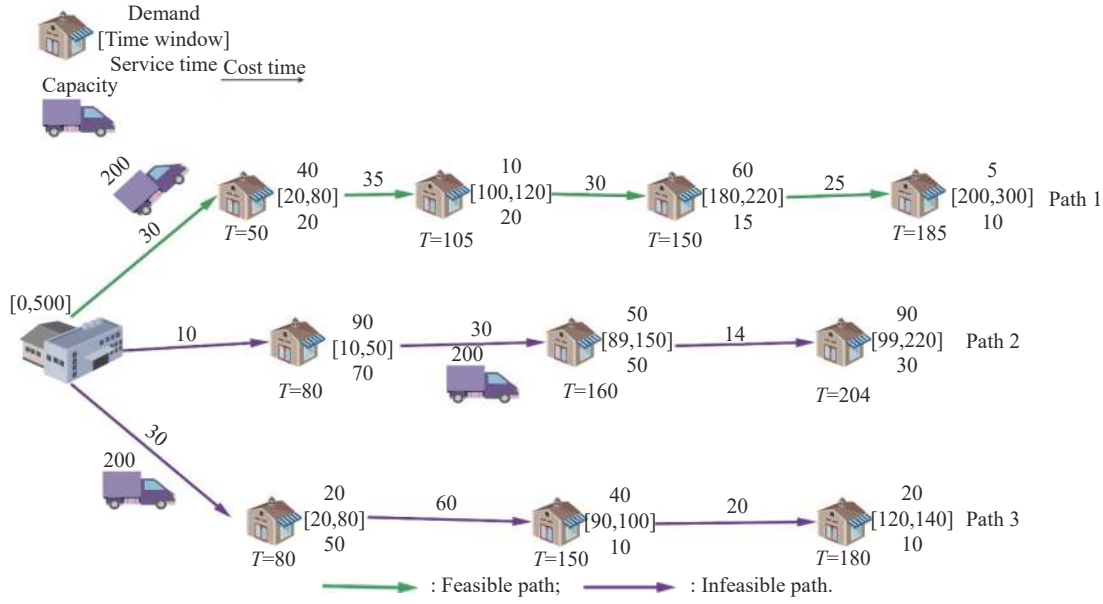


Fig. 1 Illustrative examples associated with the OVRPTW

3.2 Notation and mathematical formulation

Let $G=(V,A)$ be a complete directed graph with $V=\{0,1,2,\cdots,n\}$. Node 0 denotes the distribution center, and other numbers represent the customers. For customer i, the quality d_i of demand is to be satisfied by the vehicle within the time window $[e_i,l_i]$. Furthermore, the service time s_i is required. $A=\{(i,j)|i,j\in V\}$ signifies the group of arcs. For each$(i,j)\in A$, let c_{ij} be the distance from

node i to node j. A homogeneous fleet is located in the depot, and each vehicle has a capacity Q. The objective of the OVRPTW is to minimize the total distance, while considering the following assumptions:

(i) The vehicle must provide services for all customers, and the vehicle can only visit each customer once;

(ii) Each feasible route must start from the depot;

(iii) The total demand of customers on the path served by each vehicle does not exceed the capacity Q;

(iv) The service of each customer i must start within the customer time window $[e_i, l_i]$, and if the vehicle arrives early, the waiting time w_i must be spent.

The complete variable parameter definition of the OVRPTW is shown in Table 1.

The OVRPTW'S mixed integer programming model is described in the following equations:

$$\min \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=1}^{K} c_{ij}x_{ij}^{k} \tag{1}$$

s.t.

$$\sum_{k=1}^{K}\sum_{i=0}^{n} x_{ij}^{k} = 1,\ \ \forall j = 1,2,\cdots,n; i \neq j \tag{2}$$

$$\sum_{k=1}^{K}\sum_{j=0}^{n} x_{ji}^{k} = 1,\ \ \forall i = 1,2,\cdots,n; i \neq j \tag{3}$$

$$\sum_{i=1}^{n} d_i y_i^k \leqslant Q,\ \ \forall k=1,2,\cdots,K \tag{4}$$

$$\sum_{j=1}^{n} x_{0j}^{k} = 1,\ \ \forall k = 1,2,\cdots,K \tag{5}$$

$$\sum_{i=1}^{n} x_{i0}^{k} = 0,\ \ \forall k = 1,2,\cdots,K \tag{6}$$

$$\sum_{k=1}^{K}\sum_{i=0}^{n} x_{ij}^{k}(t_i + t_{ij} + w_i + s_i) = t_j,\ \ \forall j = 1,2,\cdots,n \tag{7}$$

$$e_i \leqslant t_i + w_i \leqslant l_i,\ \ \forall i = 1,2,\cdots,n \tag{8}$$

$$x_{ij}^{k} \in \{0,1\},\ \ \forall k = 1,2,\cdots,K;$$
$$\forall i = 1,2,\cdots,n;\ \ \forall j = 1,2,\cdots,n \tag{9}$$

$$y_i^k \in \{0,1\},\ \ \forall i = 1,2,\cdots,n;\ \forall k = 1,2,\cdots,K \tag{10}$$

The optimization objective (1) minimizes the distance traveled by the vehicles. Constraints (2) and (3) ensure that each customer is served by a vehicle. Constraint (4) represents the vehicle capacity. Constraints (5) and (6) indicate that all vehicles should leave the central depot to serve a group of customers without returning to the depot. Constraints (7) and (8) indicate that each customer has a time window. Constraints (9) and (10) indicate that the decision variables are binary decision integer variables. Moreover, we assume that the distance between customers satisfies the constraint of the triangle inequality. Obviously, the above model can be transformed to the CVRP model by setting $c_{i0} \neq 0$ and the right-hand side of constraint (7) is 1.

Table 1 Symbol description

Symbol	Description of symbol
x_{ij}^{k}	Decision variable: if the vehicle k visits customer i after visiting customer point j, then the variable value is 1; otherwise, the value is 0
y_i^k	Decision variable: if customer i is visited by the vehicle k, the variable value is 1; otherwise, the value is 0
t_i	The time of vehicle arrives at customer i
w_i	The waiting time of vehicle to arrive at customer i
i, j	Customer node $i, j \in V$
k	Vehicle node $k = 1,2,\cdots,K$
K	Total number of vehicles
n	Total number of customers
Q	Maximum vehicle capacity
c_{ij}	Distance from customer i to customer j
t_{ij}	Time from customer i to customer j
d_i	Demand of customer i
$[e_i, l_i]$	Time window of customer i
e_i	The earliest time that customer i can be visited
l_i	The latest time that customer i can be visited
s_i	Service time of customer i

4. HCGA for the OVRPTW

In this section, we will detail the proposed HCGA for the OVRPTW via explaining its key components, including the framework of HCGA, the set partitioning model, the generation of feasible path, the label setting algorithm for the small-scale subproblem, the PSO algorithm for the large-scale subproblem, and the branch strategy.

4.1 HCGA's framework

The OVRPTW model in Subsection 3.2 is not suitable for addressing medium and large-scale problems. Obviously, with the increase of the scale of the OVRPTW, its decision variables increase rapidly, which makes the solution space very huge. It is difficult to find good enough solutions in such a huge solution space. We consider combin-

ing the decomposition method and the intelligent algorithm to solve the OVRPTW. The existing decomposition techniques include Lagrange relaxation [31], column generation, and row generation [32]. Since the column generation has been successfully applied to various combinatorial optimization problems, we utilize it as the decomposition method and the search framework in our HCGA. When the mixed integer programming model of the OVRPTW is transformed into the set partitioning model, the column generation framework is used to deal with many variables. The process of generating other effective columns is realized by solving a so-called pricing problem, in which one or more dual variables with negative cost are determined. After each execution of the pricing procedure, we calculate the optimal value of the linear programming (LP) relaxation of the restricted master problem (RMP), which is the upper bound of the optimal value based on the set partitioning model, and this value itself is the lower bound of the mixed integer programming model of the OVRPTW. When using the solver to solve the RMP, we obtain the optimal value of the dual variable corresponding to the constraint. These values are used for pricing issues to check whether we can generate new columns with negative costs. If we find such columns, we add them to relax the main problem and re-optimize them. In the framework of the HCGA, the way of adding columns adopts the PSO algorithm. This hybrid mechanism can provide diverse columns and reduce the running time of the algorithm. A detailed description of the CGA can be found in [30].

The HCGA can be considered as an improvement of the basic CGA based on the characteristics of the OVRPTW. The main process of the HCGA is provided as follows: Firstly, the LP relaxation is applied to the master problem (MP) of the OVRPTW, and the original RMP is obtained. Then the RMP is solved via a commercial solver (e.g., GUROBI) with the dual variables obtained and its final solution set as the initial solution. Secondly, the goal of the subproblem is to generate the shortest path to visit customers. The subproblem is considered as a resource constrained shortest path problem, and the subproblem is solved by using the label setting algorithm to accurately consider the resource constraints in the OVRPTW. In the algorithm, the depot node is set as the initial node, and a label is set for each customer node to indicate the current state of time and cost resource constraints, and the reduced cost of the node is calculated by the dual variables. In each iteration of the algorithm, these nodes are extended and the hopeless nodes are filtered out through the dominating rules until no new labels are generated. The label setting algorithm essentially depends on the mathematical programming model, and the solution time of the mathematical programming model is relatively large. Therefore, when the solution time of the label setting algorithm exceeds the set value, the PSO is used to further explore the sub problem space, which can not only provide a better combination of existing columns, but also introduce new columns. As a result, the resulting columns are more diverse and may also accelerate pricing. Finally, the HCGA uses a concise and effective branching strategy to generate feasible solutions. The complete HCGA framework can be seen in Fig. 2.

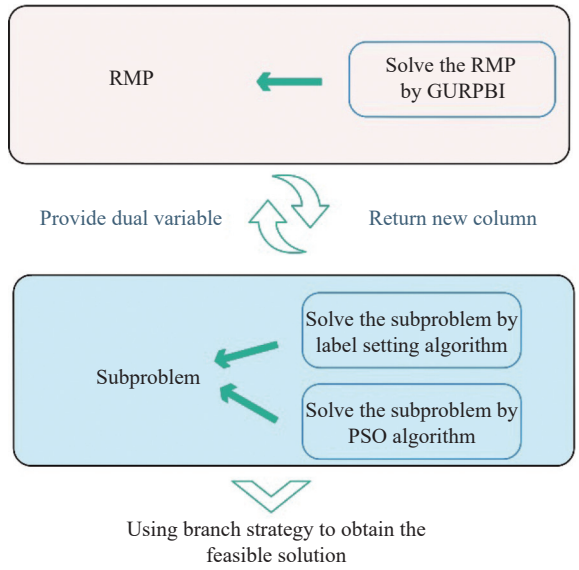


Fig. 2 Complete HCGA framework

4.2 Set partitioning model

Balinski et al. [33] first proposed a classical method for solving CVRP. This method represents the model as a set of division forms and lists all feasible routes. The start and end positions of each vehicle's feasible route are at the central depot, and the total demand of all customers served by the route does not exceed the total vehicle load. In the OVRPTW, since the vehicle does not return to the depot, we assume a virtual depot node connected to the last customer in the path, but its cost is 0.

Next, we describe the set partitioning model, let the set of all feasible paths be $\mathcal{R}=\{1,2,\cdots,R\}$, and let c_r be the cost of route r. Define δ_{ir} as the customer selection parameter in the path

$$\delta_{ir}=\begin{cases}1, & \text{route } r \text{ through customer } i;\ r\in\mathcal{R}\\ 0, & \text{otherwise}\end{cases}$$

for customer $i\in V$ and path $r\in\mathcal{R}$. Let x_r be a binary decision variable

$$x_r = \begin{cases} 1, & \text{route } r \text{ is adopted} \\ 0, & \text{otherwise} \end{cases}.$$

The goal is to select a set of feasible paths with the minimum cost so that each customer is included in a route. It is

$$\min \sum_{r\in\mathcal{R}} \mathrm{c}_r x_r \tag{11}$$

$$\overset{\text{s.t.}}{\sum_{r\in\mathcal{R}}} \delta_{ir} x_r = 1, \quad \forall i \in V\backslash\{0\}, \tag{12}$$

$$\sum_{r\in\mathcal{R}} x_r \leqslant K, \tag{13}$$

$$x_r = \{0,1\}, \ \forall r \in \mathcal{R}. \tag{14}$$

This model is the MP of the OVRPTW, the objective function (11) determines a group of travel expenses with the smallest feasible path; constraint (12) ensures that each customer point is covered by a feasible path in the selected scheme; constraint (13) means that the number of feasible routes cannot exceed the number of vehicles. Constraint (14) is a variable value constraint. $\mathcal{R}$ is the set of all feasible routes that satisfy the OVRPTW constraints. The difficulty of this model is that the value of $|R|$ is usually very large, and the number of feasible routes $r \in \mathcal{R}$ increases exponentially with the number of customers. There is no way to find all the columns initially. In fact, we can find the set $\widehat{\mathcal{R}} \in \mathcal{R}$ composed of some initial paths, and relax the constraint (14) as follows:

$$0 \leqslant x_r \leqslant 1, \quad \forall r \in \widehat{\mathcal{R}}. \tag{15}$$

Then the RMP is obtained. Obviously, the above problem is an LP problem, and the optimal solution of the problem can be obtained in a short time by commercial software. And this relaxation does not lose the optimal solution of the original problem.

4.3 Generation of feasible path

The construction of an initial feasible solution has an essential impact on the running speed of the whole algorithm. First of all, it is impossible to give all the columns at the beginning of the CGA. Similar to the simplex method, if the initial columns are nonlinear, they form a basis. Therefore, we cannot select more than the number of vehicles to form the initial column. Each column represents the customer point to be visited, and only one customer point is visited at a time. Without violating other constraints, the initial column formed by these paths is obviously a basic feasible basis. Second, in the initial path, we do not want to visit too many customers, but to visit promising clients and to obtain dual variables quickly. The generation rules of the initial feasible basis are shown in Algorithm 1, and the route division is based on the vehicle weight q_i and time window $[e_i, l_i]$ required by customers.

Algorithm 1 Initialization procedure

1:INPUT: Number of customer points n;
Customer demand q_i;
Distance between customers ;
Number of vehicles k ;
Vehicle load limit Q;
Time window restrictions $[e_i, l_i]$;
Candidate partial path set $C = \varnothing$;
Path set $\mathcal{R} = \varnothing$
2: **for all** $i \in V - \{0\}$ **do**
3: **if** $q_i < Q$ and $i < k$ **then**
4: **set** $C = C \cup \{i\}$
5: **end if**
6: $\mathcal{R}+ = C$
7: **Return** k paths in $\mathcal{R}$ set

4.4 Pricing subproblem

In order to solve the RMP, the method of dynamically adding columns is adopted, because it is impossible to find all feasible paths $r \in \mathcal{R}$. Define

$$\text{reduced cost} = c_r - \sum_{\forall i\in V\backslash\{0\}} (\pi_i - \pi_0)$$

where π_i and π_0 represent dual variables related to constraint (12) and constraint (13) respectively. According to the duality theory, if there is a path with a negative reduced cost, there are solutions that can improve the current target value, and this process is called pricing. The subproblem is equivalent to a elementary shortest path problem with resource constraints (ESPPRC). The ESPPRC is a strong NP-hard problem [33]. The precise algorithms for solving the shortest path problem include Dijkstra's algorithm, Floyd's algorithm and so on. The Dijkstra algorithm is a single-source shortest path algorithm. It cannot only find the shortest path from the source node to the sink node, but also the shortest path from any point in the graph to the rest of the nodes. The time complexity of the algorithm is $O(n^2)$ but the Dijkstra's algorithm cannot solve the shortest path problem with negative weights. The Floyd algorithm is a multi-source shortest path algorithm that calculates the shortest distance between any two nodes. However, the complexity of Floyd algorithm is $O(n^3)$, which is not suitable for large-scale data calculation. Therefore, the choice of subproblem solving algorithm is very important to the speed of the whole algorithm. In the following, we will introduce

two different types of solving algorithms, namely, the dynamic programming algorithm that uses the label setting algorithm and the intelligent algorithm to solve the pricing subproblem.

4.5 Label setting algorithm for the small-scale pricing subproblem

According to the model in the previous section, it can be concluded that the aim of the pricing subproblem is to find one or more paths with negative reduced cost. So far, an effective and exact method reduces the problem to a resource constrained shortest path problem and solve it with dynamic programming. We refer to the dynamic programming based label setting algorithm described in [4] and [34], and apply it to OVRPTW. This subsection gives the procedure and properties of the label setting algorithm. In the label setting algorithm, the structure of the OVRPTW label is given first, and the label i of node $L_i = \{c_i^r, q_i, \alpha_i, y_i\}$ is given, where node c_i^r represents related reduced cost; q_i represents demand accumulation related to the path; α_i represents accumulative total times related to node i; y_i represents the visited node associated with the label.

Given the label $L_i = \{c_i^r, q_i, \alpha_i, y_i\}$, L_i can extend $(i, j) \in V$, which needs to meet the following requirements:

$$q_i + d_j < Q, \tag{16}$$

$$\alpha_i + s_i + \mathrm{t}_{ij} < e_j. \tag{17}$$

Constraint (16) ensures that when the label is expanded to form a path, the customer's demand will not exceed the vehicle's load, and it guarantees the related constraints of the time window. We have the following claims about label setting algorithm:

Claim 1 In the label setting algorithm of dynamic programming, if the label $L_i' = \{c_i'^r, q'_i, \alpha_i', y_i'\}$ is extended to j, then $(i, j) \in V$ and subsequent paths do not need to be extended if the above resource constraints are not satisfied. This path must not be feasible.

Proof If there is a path r in directed $G = (V, A)$, and r contains a subtour $i \to k \to i$, then there must be $\alpha_i + t_{ik} + s_k > e_i$ at node k. This is in contradiction with the constraint $\alpha_k + s_k + \mathrm{t}_{ki} < e_i$ that can be extended. The assumption does not hold. □

From the above label definition and extension function, we can know that as long as the extension conditions are met, each node can expand a considerable number of labels. In the process of finding the shortest path, it needs to continuously traverse all the labels, which will increase the solving time of the subproblem. Still not all the labels are effective in the process of solving the subproblem.

Suppose both $L_i' = \{c_i'^r, q'_i, \alpha_i', y_i'\}$ and $L_i'' = \{c_i''^r, q_i'', \alpha_i'', y_i''\}$ are labels associated with customer point i if there are

$$c_i'^r \leqslant c_i''^r, \tag{18}$$

$$q' \leqslant q'', \tag{19}$$

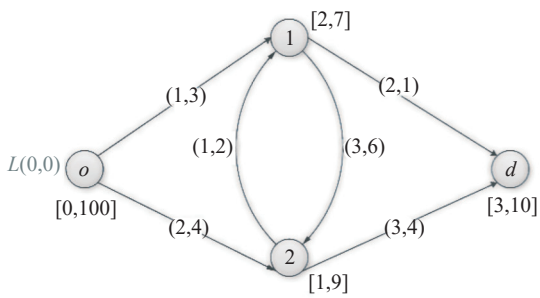
$$\alpha_i' \leqslant \alpha''. \tag{20}$$

As long as one of the inequalities (18), (19), and (20) is strictly true, the label L_i' dominates L_i''.

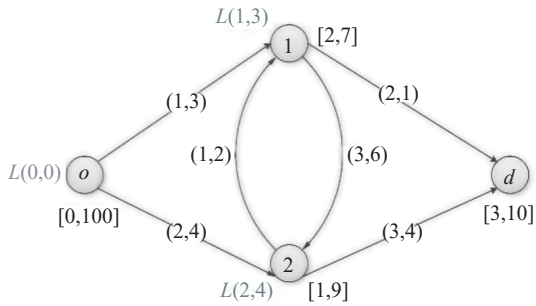
Claim 2 In the label setting algorithm of dynamic programming, if the label $L_i' = \{c_i'^r, q_i', \alpha_i', y_i'\}$ dominates $L_i'' = \{c_i''^r, q_i'', \alpha_i'', y_i''\}$, the label $L_i'' = \{c_i''^r, q_i'', \alpha_i'', y_i''\}$ and other labels controlled by it can be discarded.

Proof When the label $L_i' = \{c_i'^r, q_i', \alpha_i', y_i'\}$ dominates $L_i'' = \{c_i''^r, q_i'', \alpha_i'', y_i''\}$, it indicates that the reduced cost of L_i' must not be greater than that of L_i'', then the reduced cost generated by the extended label of L_i'' must not be smaller than that of L_i', then the new path generated based on $L_i'' = \{c_i''^r, q_i'', \alpha_i'', y_i''\}$ cannot be the optimal solution of the subproblem, so we can discard the label dominated by $L_i'' = \{c_i''^r, q_i'', \alpha_i'', y_i''\}$. □

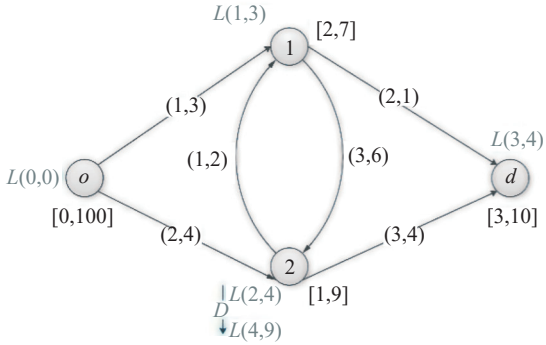
As shown in Fig. 3, a simple example of the label setting algorithm solving the shortest path problem is shown. In the directed graph, the data on each arc corresponds to two kinds of resources, namely cost resource and time resource. Each point has its own service time window. It is assumed that the goal of optimization is to minimize the path cost.



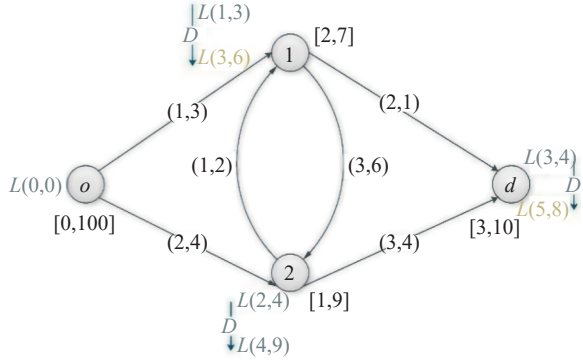
(a) Initialization



(b) The first round of label expansion



(c) The second round of label expansion



(d) The third round of label expansion

Fig. 3 Illustrative example associated with label setting

In the first round of label expansion: Set the initial label of Node o to $L(0,0)$, representing the initial accumulation of two resources. First, add the label of source Node o to the label set to be expanded; secondly, the customers Node 1, Node 2 are connected to Node o . Use the label expansion function to gradually mark the Node 1, Node 2 labels connected to it and set them to $L(1,3)$ and $L(2,4)$. At this time, the extension of the customer point label connected to Node o is completed. And there is no dominance relationship between the labels of the customers.

In the second round of label expansion: Firstly, put the customer Node 1 into the label set to be expanded, and delete the label of Node o from the label set to be expanded. Secondly, the label of customer Node 1 is $L(1,3)$, the expandable node is Node 2, Node d, and the labels are set to $L(4,9)$ and $L(3,4)$ respectively. The label connected to customer Node 1 is expanded. At this time, the label set of customer Node 2 is $L(2,4)$ and $L(4,9)$, and the two labels have a dominant relationship, so the label $L(4,9)$ of customer Node 2 does not need to be expanded.

In the third round of label expansion: Firstly, put customer Node 2 into the label set to be expanded, and delete the label of customer Node 1 from the label set to be expanded. Secondly, the label of customer Node 2 is $L(2,4)$, the expandable node is Node 1, Node d, and its label is set to $L(3,6)$ and $L(5,8)$. The label connected to customer Node 2 is expanded. At this time, the label set of customer Node 1 is $L(1,3)$ and $L(3,6)$, and the two labels have a dominant relationship, so the label $L(3,6)$of customer Node 1 does not need to be expanded. The label set of Node d is $L(3,4)$ and $L(5,8)$, and there is a dominant relationship between the two labels, so the label of Node d only retains $L(3,4)$.

Through the three rounds of expansion, the labels of all customer points in Fig. 4 are expanded. At this time, it can be seen that the optimal shortest path is $o \to 1 \to d$, the smallest cost is 3, and the arrival time is 4.

Next, a label setting algorithm based on the OVRPTW is given (see Algorithm 2), which can be described as follows:

(i) ND is the set of customer nodes to be processed;

(ii) WL is the label set waiting to be processed;

(iii) TL is the processed label set;

(iv) Extend_l (L_i, j) is label extension function: input the label L_i and Node j to be expanded, and compare the load resources and times windows resources from i to j. If the constraints (16) and (17) are met, it can be extended. The function returns the new label and updates the resource. Otherwise, it does not return.

Algorithm 2 Label setting algorithm

Initialization: set a label $L_0 = \{0,0,0,0\}$; set WL = $\{L_0\}$;TL = $\varnothing$
for all $i \in$ ND **do**
set WL = $\varnothing$ **;**TL = $\varnothing$
repeat
  select $L_i \in$ WT
    for all $j \in$ ND
      if $L_j \leftarrow$ Extend_$l(L_i, j)$;
        if L_j is not dominated by label in L_i
          set WL = WL + $\{L_j\}$;
            filter labels dominated by L_i
        end if
      else
        set WL = WL + $\{L_j\}$;
      end if
  end for
  set WL = WL − $\{L_i\}$;TL = TL + $\{L_i\}$
until WL = $\varnothing$
return the set of paths with reduced cost

4.6 PSO algorithm for the large-scale pricing subproblem

Since the label setting algorithm belongs to dynamic programming in essence, its running time is too long for large-scale subproblem. Fortunately, intelligent algorithms can get better solutions for such problems in a

short time. Thus, when the current results are still unsatisfactory after the label setting algorithm running a certain time, the HCGA applies PSO algorithm with a certain probability to see whether it can further decrease the current reduced cost. That is, PSO is used to solve the large-scale pricing subproblem. PSO was proposed by Kennedy et al. [35] in 1995. It originated from the research on the predatory behavior of birds. Its basic core is to make use of the information shared by the individuals in the group. The movement of the whole group will evolve from disorder to order in the problem solving space, to obtain the optimal solution of the problem.

In PSO, the set of current solutions is called population. Each member of the population (called a particle) is a feasible solution. The main components of PSO algorithm include particle individual coding (integer coding is adopted in this paper), fitness function (the algorithm's fitness function is related to reduced cost), speed update and position update. PSO starts with randomly generated individual particles, which have their positions and velocities. Then these groups of particles pass through a population. The particles in the population update towards the optimal particle position and repeat the process until the maximum value is reached. Fig. 4 presents a flow diagram of the developed PSO algorithm.

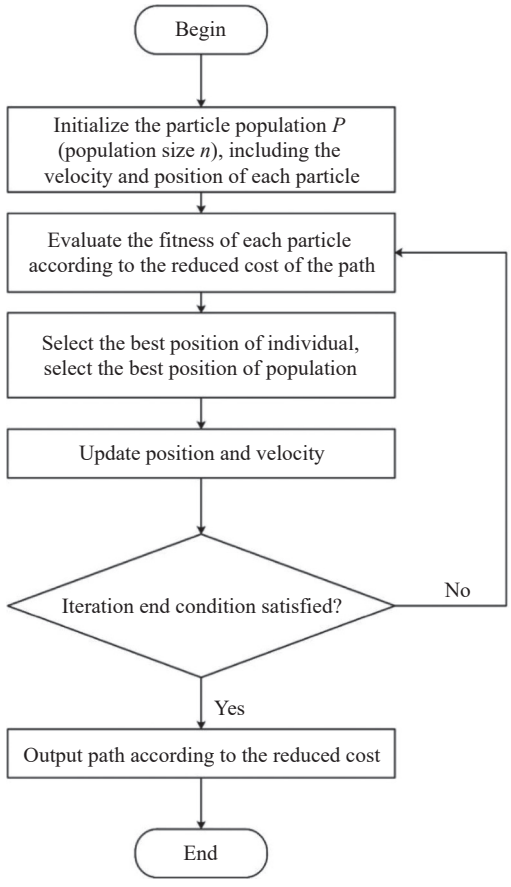


Fig. 4 A flow diagram of the developed PSO algorithm

4.7 Branch strategy

At the end of HCGA, if the current RMP solution is an integer, this solution is the optimal solution or approximate optimal solution of the set partitioning model. If the generated solution contains fractions, then the solution is a lower bound of the set partition model. The integer solution of the current scheme is obtained by embedding the branch and bound algorithm. In the process of building a branch and bound tree, each child node of the tree is the lower bound of the set partition model. The integer solution found in each iteration is updated with the current upper bound. A simple and effective branching strategy is to branch the number of scheduling schemes (vehicles). In other words, if $\sum_{r\in\widehat{\mathcal{R}}} x_r$ is a fraction, it branches by adding constraints $\sum_{r\in\widehat{\mathcal{R}}} x_r \geqslant s$ or $\sum_{r\in\widehat{\mathcal{R}}} x_r \leqslant s$, where s is the sum of scheduling schemes, and branches continuously until a feasible scheduling scheme is generated.

5. Computational experiments

Our numerical simulation experiment consists of two parts. The first part is from Solomon's instance, and the second part is from a real logistics transportation case. All experiments are run on Windows 10 platform, which has 2.5 GHz CPU and 8 GB RAM, and is on a single thread. The algorithm in this paper is implemented by Python 3.6. The MP is solved by solver Gurobi9.0.2.

5.1 Soloman's instances

The Solomon instance covers all aspects of VRP with time windows. Therefore, in the first part of the experiment, this test set is used to test algorithm's performance. The Solomon benchmark test set divides the test data into three categories according to the position of nodes in Classes C, R, and RC, the client nodes of Class C data are centralized, and the nodes are distributed around several central locations. In Class R data, the nodes are randomly distributed, while the RC data are between them. Some nodes are randomly distributed and some nodes are centralized. According to the different test data, the above three categories can be further divided into six sub categories, C1, C2, R1, R2, RC1, and RC2. The same sub category test data has the same node coordinates, different time windows, and different vehicle volumes. Since the OVRPTW and the CVRPTW can be transformed into each other (see Section 3), the Solomon benchmark test set is used to illustrate the effectiveness of the proposed algorithm to a large extent.

In order to verify the performance of HCGA, this section compares the HCGA with the simulated annealing

(SA) [36] and the memetic approach (MA) [37] in international journals to solve similar problems in Soloman's instances. Among them, SA and MA run for 200 generations, and the optimal value is taken as the ub value which is the optimal values of SA, MA, and HCGA. Table 2 shows that C, R, and RC of Soloman's instances select different scale customer points for comparison respectively. opt is the known optimal solution, where

$$\text{Gap} = \frac{\text{ub}-\text{opt}}{\text{opt}} \times 100\%.$$

Table 2 Comparison of HCGA with SA and MA

Instance	n	opt	SA			MA			HCGA			
			ub	Gap/%	Time/s	ub	Gap/%	Time/s	ub	Gap/%	addcolumn	Time/s
C101	25	191.3	194.18	1.51	1.00	198.9	3.97	6.55	**191.4**	**0.05**	1 351	1.31
	50	362.4	489.08	34.96	1.50	448.9	23.87	14.77	**363.24**	**0.23**	3 280	8.85
	100	827.3	866.16	4.70	1.11	1 104.8	33.54	27.82	**828.94**	**0.20**	11 188	78.35
C102	25	190.3	192.36	1.08	1.05	256.1	34.58	6.78	**190.3**	**0.00**	5 516	22.82
	50	361.4	539.43	49.26	1.06	574.8	59.05	14.35	**362.17**	**0.21**	14 705	24.27
	100	827.3	994.56	20.22	1.11	1 317.4	59.24	28.73	**828.93**	**0.20**	90 644	4 138
C105	25	191.3	196.81	2.88	1.06	202.4	5.80	6.53	**191.3**	**0.00**	1835	1.47
	50	362.4	472.3	30.33	2.13	435.6	20.20	12.22	**363.24**	**0.23**	5 291	9.37
	100	827.3	830.67	0.41	3.13	1 288.5	55.75	28.83	**828.93**	**0.20**	15 398	19.59
C106	25	191.3	195.71	2.31	1.06	214.4	12.08	6.4	**191.8**	**0.26**	1 542	1.32
	50	362.4	497.13	37.18	2.17	450.6	24.34	12.33	**363.24**	**0.23**	4 184	11.27
	100	827.3	945.36	14.27	2.12	1 194.2	44.35	27.96	**828.93**	**0.20**	49 817	210.7
C201	25	214.7	219.82	2.38	1.05	255.7	19.10	6.42	**215.54**	**0.39**	4 303	3.64
	50	360.2	598.24	66.09	1.09	413.7	14.85	13.52	**361.79**	**0.44**	18 633	13.27
	100	589.1	762.6	29.45	2.26	1 038.9	76.35	27.87	**591.56**	**0.42**	122 465	268.01
R101	25	617.1	772.38	25.16	0.10	656.9	6.45	6.86	**618.32**	**0.20**	171	0.10
	50	1 044	1 044	**0.00**	1.10	1 344.5	28.78	13.52	**1 046.70**	**0.26**	839	0.50
R102	25	547.1	764.68	39.77	1.24	590.1	7.86	6.98	**547.40**	**0.05**	556	0.45
	50	909	972	6.93	2.87	936.72	3.05	15.25	**911.44**	**0.27**	2 930	3.98
R103	25	454.6	454.6	**0.00**	1.45	470.8	3.56	7.28	**455.69**	**0.24**	1 500	1.60
	50	772.9	795.02	2.86	2.81	832.5	7.71	13.14	**771.9**	*	8 591	17.82
R104	25	416.9	417.89	0.24	0.49	490.3	17.61	7.57	**417.96**	**0.25**	1 797	4.69
	50	625.4	752.32	**20.29**	1.04	1 091.4	**74.51**	16.79	–	–	–	–
R105	25	530.5	533.72	0.61	0.54	534.6	0.77	7.02	**531.53**	**0.19**	407	0.13
	50	899.3	901.57	**0.25**	1.77	901.07	**0.20**	13.17	–	–	–	–
RC101	25	461.1	494.01	7.14	1.59	461.1	**0.00**	6.57	**409.24**	*	621	0.67
	50	944	944	**0.00**	2.63	1 078.4	**14.24**	12.79	–	–	–	–
RC103	25	332.8	337.41	1.39	1.43	543.7	63.37	6.97	**333.91**	**0.33**	6 933	25.52
	50	822.5	822.5	**0.00**	2.05	1 168.9	42.12	13.22	**647.36**	*	24 232	572.67
RC105	25	411.3	419.56	2.01	1.49	417.1	1.41	6.63	**412.37**	**0.26**	1 611	1.88
	50	855.3	915.12	6.99	2.87	994.97	16.32	12.59	**763.42**	*	5 301	8.3

The "addcolumn" column in the table is the total number of columns added by the HCGA. The optimal ub value and Gap value corresponding to each example in Table 2 are shown in bold. In Table 2, "–" means that the optimal solution is not found within one hour. "*" means a better solution than the known optimal solution obtained by HCGA.

As can be seen from Table 2, the test results of SA and MA on most problems run faster than the HCGA. However, for small-scale instances, the optimal value obtained by the HCGA in the same order of magnitude is better than the other two algorithms. On large-scale instances, the HCGA runs for a long time. However, the ub value obtained by the HCGA is better, and the maximum Gap is no more than 1%, which verifies that the HCGA has good performance.

5.2 Case analysis of enterprise logistics scheduling

In this part, we apply the HCGA to a real logistics distribution case. In small and medium-sized cities, many enterprises cannot afford to build a complete logistics department. The solution usually adopts outsourcing logistics or only relies on its own distribution vehicles. Usually, the enterprise only pays the cost from the depot to the last customer point. This is a real application case of the OVRPTW.

We investigate the distribution process of a large flooring enterprise in Qingdao. The enterprise has three depots. The customer node is based on real-world longitude and latitude coordinates. According to the different

needs of distribution customers, we abstract three different sizes of distribution customer sets: small (5,10,15), medium (20,25,30,35), and large (40,45,50). Table 3 shows the characteristics of the case. The first column gives the name of the case, and the second and third columns show the number of vehicles available V_{num} and the vehicle's carrying capacity V_{cap}, respectively. The third to fifth columns give the range of customer demand C_{demand}, fixed service time $C_{ServiceTime}$, and time window $C_{TimeWindow}$.

Table 3 Parameters in enterprise logistics scheduling

Instance	V_{num}	V_{cap}	C_{demand}	$C_{ServiceTime}$/min	$C_{TimeWindow}$/min
C501	15	300	[10−50]	45	[50−70]
C502	15	300	[10−110]	30,45	[20−80]
C503	15	300	[30−110]	45	[20−100]

Table 4 shows the distribution optimization results between cities. The MIPsolved column gives the optimal solution (shown in bold) obtained by GUROBI solver based on the mixed integer programming model given in Subsection 3.2. The CGsolved column gives the HCGA proposed in this paper to find the shortest path of distribution. The calculation method of Gap is

$$\text{Gap} = \frac{\text{CGsolved} - \text{MIPsolved}}{\text{MIPsolved}} \times 100\%.$$

Table 4 Results of the algorithm for enterprise logistics scheduling

Instance	Scale	n	MIPsolved	Time/s	CGsolved	Time/s	Gap/%
C501	Small	5	3.24	0.46	**3.24**	0.012	**0.00**
		10	**5.72**	2.95	**5.72**	0.026	**0.00**
		15	14.33	12.58	14.73	0.11	**2.79**
	Medium	20	**21.20**	51.87	**21.20**	0.377	**0.00**
		25	36.46	68.03	**36.46**	1.07	**0.00**
		30	48.17	63.25	49.5	2.16	**2.76**
		35	70.12	59.17	70.655	3.47	**0.76**
	Large	40	81.59	125.49	81.657	5.78	**0.08**
		45	103.98	242.82	104.77	13.21	**0.76**
		50	119.24	644.77	**119.24**	13.8	**0.00**
C502	Small	5	12.8	0.64	**12.8**	0.013	**0.00**
		10	40.19	2.15	**40.19**	0.019	**0.00**
		15	60.52	14.56	**60.52**	0.027	**0.00**
	Medium	20	62.05	30.62	62.93	0.128	**1.42**
		25	69.11	103.79	70.28	0.216	**1.69**
		30	81.02	89.21	81.91	0.562	**1.10**
		35	85.34	144.93	86.22	1.08	**1.03**
	Large	40	89.99	201.01	91.67	2.17	**1.87**
		45	96.8	396.54	101.86	2.48	**5.23**
		50	111.07	365.71	116.41	3.61	**4.81**
C503	Small	5	10.31	0.45	10.68	0.009	**3.59**
		10	18.52	1.7	19.37	0.0169	**4.59**
		15	32.35	8.07	33.3	0.0319	**2.94**
	Medium	20	58.86	49.42	61.08	0.055	**3.77**
		25	106.66	88.58	**107.63**	0.147	**0.91**
		30	142.37	56.83	146.84	0.339	**3.14**
		35	179.33	103.35	189.02	0.97	5.40
	Large	40	209.95	179.48	220.9	1.47	5.22
		45	227.32	282.85	241.25	1.61	6.13
		50	234.75	1228.245	251.38	2.69	7.08

It can be seen from Table 4 that the solving time of the proposed algorithm in three cases is less than 1 min, and the Gap is less than 10%. In the case of intercity and intercity, the optimal solution time is better than that of the GUROBI solver.

Fig. 5 shows the box diagram of Gap between the solutions of the mixed integer programming model and the column generation model under the scale of C501. It can be seen from Fig.5 that the Gap is usually less than 1%, which shows that the HCGA is very effective in solving the problem of this scale. Fig. 6 shows the convergence of the HCGA for the C501_50 case. It proves that the HCGA can converge to the optimal value in a short time, while the mixed integer programming model usually needs a long time to converge.

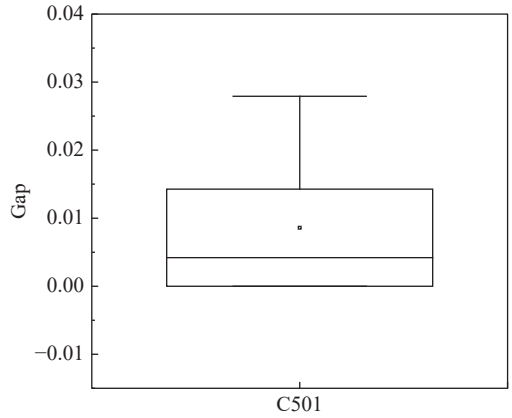


Fig. 5 Box diagram of Gap in C501

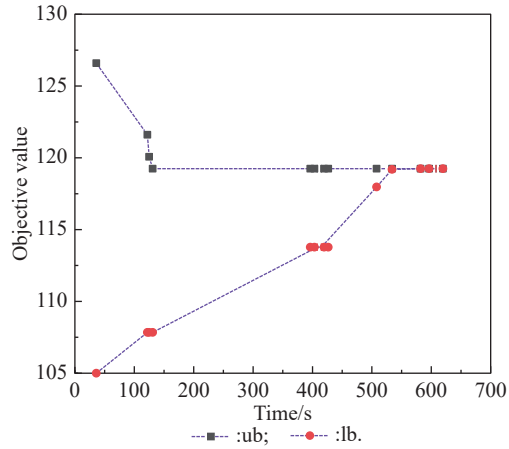


Fig. 6 Convergence of C501-50

6. Conclusions

This paper considers the OVRPTW, which is an NP-hard problem widely existing in the real world. Due to the NP-hardness of the OVRPTW, we propose an HCGA to address it. The proposed HCGA not only utilizes the column generation method to decompose the OVRPTW into a series of subproblems, thereby obviously reducing the scale of the problem and reasonably determining the search regions, but also designs a label setting algorithm and a PSO algorithm to quickly solve the decomposed subproblems and obtain quality-assured columns. Furthermore, a problem-dependent branch strategy is devised and embedded in the HCGA to generate a high-quality feasible solution as the final output. The test results and comparisons based on the benchmark instances and a real-life instance manifest the effectiveness of our HCGA. The future research is to extend the proposed algorithm to other complex logistics scheduling problems.

References

[1] DANTZIG G B, RAMSER J H. The truck dispatching problem. Management Science, 1959, 6(1): 80–91.
[2] BRANDAO J. A tabu search algorithm for the open vehicle routing problem. European Journal of Operational Research, 2004, 157(3): 552–564.
[3] FLESZAR K, OSMAN I H, HINDI K S. A variable neighbourhood search algorithm for the open vehicle routing problem. European Journal of Operational Research, 2009, 195(3): 803–809.
[4] LOZANO L, DUQUE D, MEDAGLIA A L. An exact algorithm for the elementary shortest path problem with resource constraints. Transportation Science, 2016, 50(1): 348–357.
[5] SCHRAGE L. Formulation and structure of more complex/realistic routing and scheduling problems. Networks, 2010, 11(2): 229–232.
[6] SARIKLIS D, POWELL S. A heuristic method for the open vehicle routing problem. Journal of the Operational Research Society, 2000, 51(5): 564–573.
[7] SCHOPKA K, KOPFER H. An adaptive large neighborhood search for the reverse open vehicle routing problem with time windows. Logistics Management, 2016, 163(1): 243–257.
[8] GE J H, XIOMG Y, WANG H Z. An improved TS for the open vehicle routing problem with soft time windows. Proc. of the IEEE 5th International Symposium on Computational Intelligence & Design, 2013: 382–385.
[9] REPOUSSIS P P, IOANNOU C D T. The open vehicle routing problem with time windows. Journal of the Operational Research Society, 2007, 58(3): 355–367.
[10] FORD L R, FULKERSON D R. A suggested computation for maximal multi-commodity network flows. Management Science, 1958, 5(1): 97–101.
[11] CHEN Z L, POWELL W B. A column generation based decomposition algorithm for a parallel machine just-in-time scheduling problem. European Journal of Operational Research, 1999, 116(1): 220–232.
[12] FEI H, MESKENS N, CHU C. A planning and scheduling problem for an operating theatre using an open scheduling strategy. Computers & Industrial Engineering, 2010, 58(2): 221–230.
[13] VANCE P H, BARNHART C, JOHNSON E L, et al. Solving binary cutting stock problems by column generation and branch-and-bound. Computational Optimization and Applications, 1994, 3(2): 111–130.
[14] CACCHIANI V, HEMMELMAYR V C, TRICOIRE F. A set-covering based heuristic algorithm for the periodic vehicle routing problem. Discrete Applied Mathematics, 2014, 163: 53–64.
[15] QURESHI A G, TANIGUCHI E, YAMADA T. An exact solution approach for vehicle routing and scheduling problems with soft time windows. Transportation Research Part E: Logistics & Transportation Review, 2009, 45(6): 960–977.
[16] YUAN Y, CATTARUZZA D, OGIER M, et al. A column generation based heuristic for the generalized vehicle routing problem with time windows. Transportation Research Part E: Logistics and Transportation Review, 2021, 152(8): 102391.
[17] BEHNKE M, KIRSCHSTEIN T, BIERWIRTH C. A column generation approach for an emission-oriented vehicle routing problem on a multigraph. European Journal of Operational Research, 2021, 288(3): 794–809.
[18] BOLAND N, DETHRIDGE J, DUMITRESCU I. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. Operations Research Letters, 2006, 34(1): 58–68.
[19] BERTSEKAS D P. A simple and fast label correcting algorithm for shortest paths. Networks, 2010, 23(8): 703–709.
[20] OZBAYGIN G, KARASAN O E, SAVELSBERGH M, et al. A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations. Transportation Research Part B: Methodological, 2017, 100(6): 115–137.
[21] LI C S, GONG L J, LUO Z X, et al. A branch-and-price-and-cut algorithm for a pickup and delivery problem in retailing. Omega, 2019, 89(12): 71–91.
[22] TIMO G, STEFAN I, ANN-KATHRIN R, et al. Bidirectional labeling in column-generation algorithms for pickup-and-delivery problems. European Journal of Operational

Research, 2018, 266(2): 521–530.
[23] RIGHINI G, SALANI M. Symmetry helps: bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Discrete Optimization, 2006, 3(3): 255–273.
[24] RIGHINI G, SALANI M. New dynamic programming algorithms for the resource constrained elementary shortest path problem. Networks, 2010, 51(3): 155–170.
[25] KE L J, GUO H M, ZHANG Q F. A cooperative approach between metaheuristic and branch-and-price for the team orienteering problem with time windows. Proc. of the IEEE Congress on Evolutionary Computation, 2014: 1878–1882.
[26] ROMAN V, ANTONIN N, PREMYSL S, et al. Accelerating the branch-and-price algorithm using machine learning. European Journal of Operational Research, 2018, 271(3): 1055–1069.
[27] SALARI M, TOTH P, TRAMONTANI A. An ILP improvement procedure for the open vehicle routing problem. Computers & Operations Research, 2010, 37(12): 2106–2120.
[28] SHEN Y D, PENG L W, LI J P. An improved estimation of distribution algorithm for multi-compartment electric vehicle routing problem. Journal of Systems Engineering and Electronics, 2021, 32(2): 365–379.
[29] MAJID Y, AZAM D, FARZAD D, et al. A modified column generation to solve the heterogeneous fixed fleet open vehicle routing problem. Journal of Engineering, 2016. DOI: 10.1155/2016/5692792.
[30] FAIZ T I, VOGIATZIS C, NOOR-E-ALAM M. A column generation algorithm for vehicle scheduling and routing problems. Computers & Industrial Engineering, 2019, 130: 222–236.
[31] CUI H J, LUO X C, WANG Y. Scheduling of steelmaking-continuous casting process using deflected surrogate Lagrangian relaxation approach and DC algorithm. Computers & Industrial Engineering, 2020, 140(2): 106271.
[32] BINATO S, PEREIRA M V F, GRANVILLE S. A new benders decomposition approach to solve power transmission network design problems. IEEE Trans. on Power Systems, 2001, 16(2): 235–240.
[33] BALINSKI M L, QUANDT R E. On an integer program for a delivery problem. Operations Research, 1964, 12(2): 187–376.
[34] DROR M. Note on the complexity of the shortest path models for column generation in VRPTW. Operations Research, 1994, 42(5): 977–978.
[35] KENNEDY J, EBERHART R. Particle swarm optimization. Proc. of the IEEE International Conference on Neural Networks, 1995: 1942–1948.
[36] TAVAKKOLI-MOGHADDAM R, GAZANFARI M, ALINAGHIAN M, et al. A new mathematical model for a competitive vehicle routing problem with time windows solved by simulated annealing. Journal of Manufacturing Systems, 2011, 30(2): 83–92.
[37] SABAR N R, BHASKAR A, CHUNG E, et al. An adaptive memetic approach for heterogeneous vehicle routing problems with two-dimensional loading constraints. Swarm and Evolutionary Computation, 2020, 58: 100730.

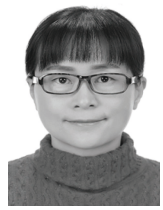
Biographies

YU Naikang was born in 1993. He received his B.S. degree in automation from Qufu Normal University, China, in 2016. He is currently pursuing his Ph.D. degree at Kunming University of Science and Technology, Kunming, China. His current research interests include operation research and mathematical programming and scheduling.

E-mail: nk_yu2020@foxmail.com

QIAN Bin was born in 1976. He received his B.S. degree in automation from Donghua University, Shanghai, China, in 1998, M.S. degree in control theory and its applications from Kunming University of Science and Technology, Kunming, China, in 2004, and Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 2009. He is currently a professor at Kunming University of Science and Technology, China. He has published over 90 referred papers. His current research interests include intelligent optimization and scheduling.

E-mail: bin.qian@vip.163.com



HU Rong was born in 1974. She received her B.S. degree in thermal energy and dynamic engineering from Southeast University, Nanjing, China, in 1997, and M.S. degree in control science and engineering from Tsinghua University, Beijing, China, in 2004. She is currently an associate professor at Kuming University of Science and Technology, China. She has published over 50 referred papers. Her current research interests include intelligent optimization and machine learning.

E-mail: ronghu@vip.163.com

CHEN Yuwang was born in 1982. He received his Ph.D. degree in control theory and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2008. He is currently a senior lecturer in decision sciences in the Alliance Manchester Business School, University of Manchester, Manchester, U.K.. Prior to his current appointment, he was a postdoctoral research associate and then appointed as a lecturer in 2011 at the Decision and Cognitive Sciences (DCS) Research Centre at the University of Manchester, U.K.. He has published over 110 referred papers. His current research interests include decision and risk analysis under uncertainties, modeling and optimization of complex systems, and operational research and data analytics.

E-mail: Yu-wang.chen@mbs.ac.uk

WANG Ling was born in 1972. He received his B.S. degree in automation and Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 1995 and 1999, respectively. Since 1999, he has been with Department of Automation, Tsinghua University, where he became a full professor in 2008. He has authored five academic books and more than 230 refereed papers. His current research interests include intelligent optimization theory, algorithms, and applications.

E-mail: wangling@tsinghua.edu.cn