IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 5, 2020, accepted August 31, 2020, date of publication September 10, 2020, date of current version September 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023090

Dynamic Memory Memetic Algorithm for VRPPD With Multiple Arrival Time and Traffic Congestion Constraints

HONGGUANG ZHANG, ZAN WANG, MENGZHEN TANG, XIUSHA LV, HAN LUO, AND YUANAN LIU, (Member, IEEE)

Beijing Key Laboratory of Work Safety Intelligent Monitoring, School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Hongguang Zhang (hongguang-zhang@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61876199.

ABSTRACT With the new distribution demands emerging continuously in the last decade, the distribution mode is changing gradually in various applications, such as e-commerce, emergency relief supplies distribution, the last-mile delivery, and so on. We formulate vehicle routing problem with pickup and delivery (VRPPD), while simultaneously considering multiple arrival time and traffic congestion constraints. Our model focuses on the clear changes of the distribution mode to meet the fast-delivery requirements in the last decade, which is characterized by the multi-batch arrival of goods in 24 hours and time-varying congestion in various time windows. Besides, we propose dynamic memory memetic algorithm, which updates its dynamic memory by whether to promote populations to find new better solutions or not. This is an effective acceleration mechanism to promote the population progress. Meanwhile, dynamic memory memetic algorithm determines the serious congestion tasks in the delivery route and transforms them into normal congestion or even non-congestion tasks. Test sets with 30 test problems are constructed by using real distribution data from Alibaba Cloud and traffic congestion data from Baidu Map in Shanghai. By comparing with four compared algorithms, the effectiveness, efficiency, and robustness of our proposed algorithm in non-congestion and congestion tests are simultaneously demonstrated.

INDEX TERMS VRPPD, pickup and delivery problem, dynamic memory, distribution mode, the multi-batch arrival of goods, traffic congestion.

I. INTRODUCTION

With the new distribution demands emerging continuously in the last decade, the distribution mode is changing gradually in various applications, such as e-commerce, emergency relief supplies distribution, the last-mile delivery, and so on. The modern distribution systems are characterized by their fast-delivery requirements. In the last decade, there are some clear changes of the distribution mode, such as large-scale pickup & delivery tasks, the multi-batch arrival of goods, traffic congestion, and so on. For instance, large-scale pickup & delivery tasks in Double 11 event of Alibaba is about 200 million packages. Moreover, these tasks are constrained by the multi-batch arrival of goods in 24 hours. Meanwhile, traffic congestion in large cities is daily occurring and producing more pollutants. These requirements should be formulated to express new distribution mechanisms.

We explain our research problem as follows. Vehicles pickup goods from depots and deliver them to customers. This is defined as vehicle routing problem with pickup and delivery (VRPPD). Fig. 1 (a) is able to demonstrate VRPPD, its new variants with various constraints [1]–[5], and its research methods [6], [7]. Different from these researches, we focus on the multi-batch arrival of goods and serious traffic congestion. We formulate VRPPD with multiple arrival time and traffic congestion constraints, as shown in Fig. 1 (b). Our model is able to meet the fast-delivery requirements of the modern distribution systems. The differences are as follows. (i) One is that goods arrive at different times, such as 6:00, 7:00, and 8:00. Thus, one vehicle must take a good from a

The associate editor coordinating the review of this manuscript and approving it for publication was Christian Pilato.

depot, after this good arrives at this depot. This constraint about the multi-batch arrival of goods is also very common in many other fields, such as disaster relief. (ii) Another is that time-varying congestion leads to the obvious differences of vehicle speed in various time windows. Although the distances among depots and customers are fixed, this results in the remarkable differences of delivery time.

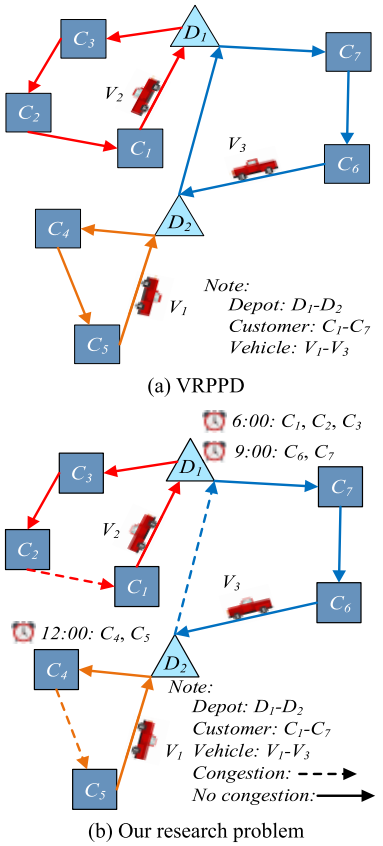


FIGURE 1. The differences between VRPPD and our research problem. Note that, the congestion conditions are time-dependent.

Inspired by memetic algorithms, we propose dynamic memory memetic algorithm (DMMA) with local individual and global population searches to address our research problem. Our contributions are as follows. (i) From a local search perspective, single-vehicle and multi-vehicle Hungarian searches are proposed to determine serious congestion tasks and address traffic congestion, which are effective to respond to the challenge of time-varying traffic congestion. Meanwhile, we introduce the individual coding and its repair strategies to deal with the challenge of the multi-batch arrival of goods. (ii) From a global search perspective, a new kind of dynamic memory is proposed to find new better solutions. DMMA updates its dynamic memory by whether to promote populations to find new better solutions or not. This is an effective acceleration mechanism to promote the population progress. (iii) Besides, we use real logistics data from Alibaba Cloud and traffic congestion data from Baidu Map in Shanghai to construct our test sets. Non-congestion results indicate that the effectiveness and efficiency of DMMA are satisfactory. Meanwhile, various-congestion-situation results mean that DMMA is effective to address VRPPD with multiple arrival time and various real-congestion constraints.

The rest of this paper is organized as follows. In Section II, we review related works, summarize our research background, and discuss the real requirements of our model. In Section III, we formulate VRPPD with multiple arrival time and traffic congestion constraints. In Section IV, we introduce DMMA. In Section V, we provide non-congestion and various-congestion-situation results. In Section VI, we present concluding remarks and future work.

II. RELATED WORK

A. VRPPD BACKGROUND AND ITS NEW VARIANTS

VRPPD consists of three parts: vehicles, depots, and customers. Vehicles load goods from depots and then deliver goods to customers. Generally, VRPPDs are classified as one-to-one VRPPDs (OTO-VRPPDs), many-to-many VRPPDs (MTM-VRPPDs), and one-to-many-to-one VRPPDs (OTMTO-VRPPDs). In OTO-VRPPD, there is a one-to-one relationship between customers and depots. For instance, one depot can only supply one customer and one customer can only receive goods from one depot. In MTM-VRPPD, depots supply multiple customers and customers receive goods from multiple depots. In principle, our model belongs to MTM-VRPPD. In OTMTO-VRPPD, vehicles not only need to transport goods to customers, but also need to collect goods from customers and return these goods to depots [8]. For example, breweries provide beer to a group of customers, while simultaneously collecting empty bottles from customers. Our model is different from OTMTO-VRPPD. We review some new variants of OTO-VRPPD and MTM-VRPPD as follows.

In the OTO-VRPPD researches, Treleaven *et al.* propose SPLICE for a one-to-one pickup and delivery problem, which is usually referred to as stacker crane problem (SCP) [9]. Besides, their research objective is to demonstrate the existence of simple polynomial-time algorithms for SCP with probabilistic optimality guarantees. Sahin *et al.* consider multi-vehicle one-to-one pickup and delivery problem with split loads (MPDPSL) [10]. Meanwhile, they propose an efficient heuristic that combines tabu search with simulated annealing. Besides, Haddad *et al.* also focus on MPDPSL and propose an iterated local search metaheuristic and a branch-and-price algorithm, which produces high-quality solutions [11]. Naccache *et al.* consider a multi-pickup and delivery problem with time windows, in which a set of requests is satisfied by a fleet of vehicles [12]. In each request, items are required to be picked up from different locations to be shipped and unloaded at one common delivery location. This problem is characterized by multi-pickup and one common delivery. Besides, they solve this problem by using a hybrid large neighborhood search heuristic. Bartolini *et al.* present traveling salesman problem with pickup, delivery, and ride-time

constraints (TSPPD-RT) [13]. Besides, they analyze TSPPD-RT with and without capacity constraints. TSPPD-RT is characterized by that each request's ride-time does not exceed its maximum ride-time.

In the MTM-VRPPD researches, Ahkamiraad and Wang consider the mixed-integer linear programming model of a special type of capacitated and multiple cross-docked vehicle routing problem with pickup, delivery, and time windows [14]. This model is characterized by multiple cross docks and each cross dock has a limited maximum capacity. Besides, they propose a hybrid of genetic algorithm and particle swarm optimization. Chen *et al.* formulate the paired many-to-many pickup and delivery problem, which is characterized by its paired demands between customers [15]. Besides, they use library vehicle routing as an application and test their two-stage solution algorithm by using real data of San Francisco library system. Ghaffarinasab *et al.* formulate planar hub location-routing problem by using a continuous approximation technique [16]. The research objective is to decide on the locations of hubs & service regions and minimize the approximate total transportation cost, including local pickup and delivery costs. Besides, an iterative Weiszfeld-type algorithm and a particle swarm optimization metaheuristic are proposed. Liu *et al.* propose a fast decomposition and reconstruction framework to solve the pickup and delivery problem with time windows and last-in-first-out loading [17]. Moreover, they use the adaptive memory mechanism to provide a better start when populations fall into the local optima. Zhou *et al.* develop a two-stage heuristic algorithm for green vehicle routing problem which focus on dual services with stochastic travel times [18]. Experimental results indicate that the proposed algorithm is suitable to address this problem, especially for providing sustainable and diversified delivery services. Wang investigates and proposes two heuristic algorithms based on tabu search and simulated annealing algorithm to address the heterogeneous fleet location routing problem with simultaneous pickup and delivery [19]. Focusing the generation of a priori routes for a fleet of vehicles, Zhu and Sheu propose a failure-specific cooperative recourse strategy to explore a risk pooling mechanism for routing in the context of simultaneous pickup and delivery with stochastic demands [20]. Marinaki and Marinakis propose combinatorial-neighborhood-topology glowworm swarm optimization for successfully solving the vehicle routing problem with stochastic demands [21]. Moreover, they use capacitated vehicle routing problem and vehicle routing problem with stochastic demands to demonstrate the effectiveness of the proposed algorithm. Ruiz *et al.* consider the open vehicle routing problem with split deliveries, propose a new two mixed-integer formulations of the problem, and design a cutting-plane method to improve the optimization performance [22].

B. VRPPD APPLICATIONS WITH VARIOUS CONSTRAINTS

In the last decade, emergency relief supplies distribution, e-commerce, and last-mile logistics develop rapidly, such as Amazon. These VRPPD applications consider many real constraints, such as load, fuel consumption, dangerous goods transportation route, delivery time, pollution emissions, time window, and so on.

We review some existing applications of VRPPDs as follows. Yanik *et al.* focus on capacitated vehicle routing problem with multiple pickups, single delivery, and time windows [23]. Moreover, they propose a hybrid metaheuristic approach and a modified savings algorithm. Besides, they test their method by using a real dataset in Istanbul. Guo and Liu address time-dependent vehicle routing of free pickup and delivery service to deliver a set of airline passengers, while considering traffic congestion, time window, and maximum ride time constraints [24]. Focusing on on-demand transportation systems and ride-sharing services, Mahmoudi and Zhou propose a new time-discretized multi-commodity network flow model based on the integration of vehicles' carrying states [25]. This model is characterized by three-dimensional state-space-time network construction. Besides, their dynamic programming approach is tested by using the Chicago sketch and Phoenix regional transportation networks. Zheng *et al.* focus on the influence of time-dependent velocity and propose a realistic model to solve urban pickup and delivery problem with considering the time-dependent fuzzy velocity of vehicles [26]. Besides, they introduce a real pickup and delivery problem with 28 customers in a logistics company. Xiao and Konak propose a new mixed-integer linear programming model for green vehicle routing and scheduling problems [27]. This model considers heterogeneous vehicles, time-varying traffic congestion, customer/vehicle time window constraints, the impact of vehicle load on emissions, and vehicle capacity/mileage constraints. Sun *et al.* introduce a new time-varying pickup and delivery problem, which considers the time window constraints of pickup and delivery points and traffic congestion [28]. In addition, a restricted dynamic programming heuristic algorithm is proposed that finds the best solution for 32 out of 34 instances. Zhang *et al.* equivalently transform the collection vehicle routing problem of the garbage facilities to the vehicle routing problem with simultaneous pickup-delivery and time windows by introducing dummy customers [29]. Besides, they propose a parallel simulated annealing algorithm to address the real application problem in the Xuanwu District of Beijing. Considering the vehicle routing problem with simultaneous pickup and delivery and handling costs, Hornstra *et al.* propose an adaptive large neighborhood search metaheuristic and improve 39 out of 54 best known solutions for this problem [30]. Peng *et al.* study the multiple vehicle pickup and delivery problem that considers the constraints of vehicle route time, vehicle capacity, and vehicle last-in-first-out [31]. Besides, they propose a learning-based memetic algorithm and improve the previous best known results for 132 out of 158 problem instances. Anwar and Younas formulate the pickup and delivery problem as a many-objective pickup and delivery problem with delay time of vehicle having six criteria to be optimized [32]. Moreover, they propose

TABLE 1. Summary of our research background.

Category	Reference/Year	Characteristics for some new variants of VRPPDs
VRPPD new variants in Section II. A	Treleaven et al. (2013)	consider OTO-VRPPD as stacker crane problem with probabilistic optimality guarantees and derive stability conditions for its dynamic counterpart
	Sahin et al. (2013)	focus on split loads from a VRPPD perspective
	Haddad et al. (2018)	focus on split loads from a VRPPD perspective
	Naccache et al. (2018)	consider multi-pickup and one common delivery
	Bartolini et al. (2016)	consider pickup, delivery, and ride-time constraints, especially for that each request's ride-time does not exceed its maximum ride-time
	Ahkamiraad and Wang (2018)	focus on multiple cross docks and each cross dock has a limited maximum capacity
	Chen et al. (2015)	focus on the paired demands between customer nodes
	Ghaffarinasab et al. (2018)	consider the locations of hubs and service regions to minimize the approximate total transportation cost
	Liu et al. (2019)	consider pickup and delivery problem with time windows and last-in-first-out loading
	Zhou et al. (2019)	consider green vehicle routing problem considering dual services with stochastic travel times
	Wang (2019)	focus on the heterogeneous fleet location routing problem with simultaneous pickup and delivery and overloads
	Zhu and Sheu (2018)	consider simultaneous pickup and delivery problem with stochastic demands
	Marinaki and Marinakis (2016)	focus on capacitated vehicle routing problem and vehicle routing problem with stochastic demands
	Ruiz et al. (2020)	consider the open vehicle routing problem with split deliveries, which allows open routes and partitioned deliveries for customers
Category	Reference/Year	Application researches under various distribution constraints
VRPPD applications with various constraints in Section II. B	Yanik et al. (2014)	focus on vehicle routing problem with multiple pickup, single delivery, and time windows. They test their method by using real dataset in Istanbul for e-tailing
	Guo and Liu (2017)	consider time-dependent vehicle routing of free pickup and delivery service for airline passengers in flight ticket sales companies
	Mahmoudi and Zhou (2016)	focus on on-demand transportation systems & ride-sharing services and use the Chicago sketch and Phoenix regional transportation networks
	Zheng et al. (2011)	test their method by using a real pickup and delivery problem with 28 customers in a logistics company
	Xiao and Konak (2016)	consider green vehicle routing and scheduling problem with heterogeneous vehicles, time-varying traffic congestion, time window, vehicle load, and vehicle capacity/mileage constraints
	Sun et al. (2018)	focus on VRPPD with time window and traffic congestion and test their algorithm by using 34 instances
	Zhang et al. (2020)	focus on vehicle routing problem with simultaneous pickup-delivery and time windows. They apply their method to the real vehicle path planning in the Xuanwu District of Beijing
	Hornstra et al. (2020)	consider vehicle routing problem with simultaneous pickup and delivery and handling costs
	Peng et al. (2020)	focus on multi-vehicle pickup and distribution problem with constraints on vehicle route time, vehicle capacity, and last-in-first-out
	Anwar and Younas (2020)	consider a many-objective pickup and delivery problem with delay time of vehicle having six criteria to be optimized

a memetic improved decomposition based evolutionary algorithm and use a variety of small, medium, and large-scale problems to test the proposed algorithm.

C. SUMMARY AND DISCUSSIONS

These previous research lines are summarized in Table 1. The research background discussions of our model are as follows. From a new variant perspective, our model focuses on MTM-VRPPD with multiple arrival time and traffic congestion constraints. This means that goods arrive at depots in various time windows. Besides, our model focuses on large-scale & urban real applications, especially for considering time-varying traffic congestion constraints.

III. PROBLEM STATEMENT AND FORMULATION

We formulate VRPPD with multiple arrival time and traffic congestion constraints as follows.

First, we explain the relationship among depots, customers, congestion coefficients, and vehicle speeds as follows. (i) Depot and customer set DC is $\{DC_1, DC_2, \ldots DC_m, \ldots DC_{NDC}\}$, DC_m represents a depot or a customer, and $NDC = |DC|$. Depot set D is $\{D_1, D_2, \ldots D_p, \ldots D_{ND}\}$, D_p is the pth depot, and $ND = |D|$. Customer set C is $\{C_1, C_2, \ldots C_q, \ldots C_{NC}\}$, C_q is the qth customer, and $NC = |C|$. In addition, $DC = D \cup C = \{DC_1, DC_2, \ldots DC_m, \ldots DC_{NDC}\} = \{D_1, D_2, \ldots D_p, \ldots D_{ND}, C_1, C_2, \ldots C_q, \ldots C_{NC}\}$, and $NDC = ND + NC$. (ii) The distances among depots and customers are expressed by using distance matrix DIS in (1). $DIS(DC_l, DC_m)$ is the element of the lth row and mth column, which is the distance between DC_l and DC_m. (iii) Congestion coefficient set J is a 3-D matrix (i.e., $NDC \times NDC \times NTW$) to express congestion situations in various time windows. NTW is the number of time windows and $J(:, :, TW_k)$ is a 2-D matrix to define the congestion situation in the kth time window TW_k. For example, $J(DC_l, DC_m, TW_k)$ is the congestion coefficient of $DIS(DC_l, DC_m)$ in TW_k, which is from 1 to $+\infty$. The greater congestion coefficient, the more serious congestion. (iv) We assume that all vehicles have the same non-congestion speed S, which is equal to 60km/h. The real speed of one vehicle from DC_l to DC_m in TW_k is given in (2) by using the definition of congestion coefficient [33]. Note that, (2) means the more serious congestion, the lower real speed. The definition of congestion coefficient $J(DC_l, DC_m, TW_k)$ reflects its average condition in TW_k. Moreover, TW_k is the time window when the vehicle leaves DC_l.

$$DIS = \begin{array}{c} \\ DC_1 \\ \cdots \\ DC_l \\ \cdots \\ DC_{NDC} \end{array} \begin{array}{c} \begin{array}{cccccc} DC_1 & \cdots & DC_m & \cdots & DC_{NDC} \end{array} \\ \begin{bmatrix} DIS_{1-1} & \cdots & DIS_{1-m} & \cdots & DIS_{1-NDC} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ DIS_{l-1} & \cdots & DIS_{l-m} & \cdots & DIS_{l-NDC} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ DIS_{NDC-1} & \cdots & DIS_{NDC-m} & \cdots & DIS_{NDC-NDC} \end{bmatrix} \end{array} \tag{1}$$

$$RS\left(DC_l, DC_m, TW_k\right) = \frac{S}{J\left(DC_l, DC_m, TW_k\right)} \tag{2}$$

$$TA = \begin{array}{c} \\ V_1 \\ \cdots \\ V_i \\ \cdots \\ V_{NV} \end{array} \begin{array}{c} \begin{array}{ccccc} T_1 & \cdots & T_j & \cdots & T_{NT} \end{array} \\ \begin{bmatrix} TA_{1-1} & \cdots & TA_{1-j} & \cdots & TA_{1-NT} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ TA_{i-1} & \cdots & TA_{i-j} & \cdots & TA_{i-NT} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ TA_{NV-1} & \cdots & TA_{NV-j} & \cdots & TA_{NV-NT} \end{bmatrix} \end{array} \tag{3}$$

$$\sum_{i=1}^{NV} TA_{i-j} = 1 \tag{4}$$

$$\sum_{i=1}^{NV} \sum_{j=1}^{NT} TA_{i-j} = NT \tag{5}$$

Secondly, we define the task allocation relationship by using task allocation matrix TA. Task set T is $\{T_1, T_2, \ldots T_j, \ldots T_{NT}\}$ and $NT = |T|$. T_j is the jth task, which is equal to $\{D_p, C_q, TW_k\}$. This indicates that this good arrives at D_p in TW_k and should deliver to C_q. Besides, vehicle set V is $\{V_1, V_2, \ldots V_i, \ldots V_{NV}\}$, V_i is the ith vehicle, and $NV = |V|$. Then, TA is given in (3) and each $TA_{i-j} \in \{0, 1\}$. If $TA_{i-j} = 1$, this means that T_j is allocated to V_i. Otherwise, this means that T_j is not allocated to V_i. In addition, (4) expresses that each task can be executed once. (4) and (5) mean that all tasks must be allocated.

Thirdly, we give the feasible task allocation solution COD as follows. The feasible task allocation solution COD is $\{COD_1, COD_2, \ldots, COD_i, \ldots, COD_{NV}\}$. As the task allocation solution of V_i, COD_i is $\{COD_{i-1}, COD_{i-2}, \ldots, COD_{i-n}, \ldots, COD_{i-NCODi}\}$ and $NCODi = |COD_i|$. COD_{i-n} is the nth depot or customer that V_i should arrive at. For example by using Fig. 2, COD_1 is the V_1's task allocation solution, which is equal to $\{D_1, C_1, D_2, C_2\}$. That is to say, $COD_{1-1} = D_1$, $COD_{1-2} = C_1$, $COD_{1-3} = D_2$, $COD_{1-4} = C_2$, and $NCOD1 = |COD_1| = 4$. This means that V_1 sequentially reaches to D_1, C_1, D_2, and C_2. Note that, each task in Fig.2 has its own arrival time, which is only able to be picked up after its arrival time.

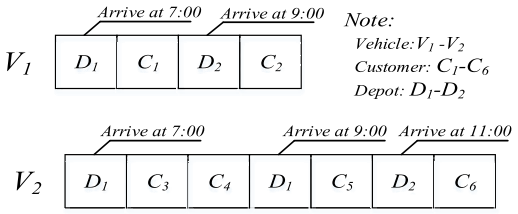


FIGURE 2. An example of one feasible task allocation solution, which includes V_1's tasks (i.e., $T_1 = \{D_1, C_1, TW_1\}$ and $T_2 = \{D_2, C_2, TW_2\}$) and V_2's tasks (i.e., $T_3 = \{D_1, C_3, TW_1\}$, $T_4 = \{D_1, C_4, TW_1\}$, $T_5 = \{D_1, C_5, TW_2\}$, and $T_6 = \{D_2, C_6, TW_3\}$).

Lastly, our objective function (i.e., fitness function in DMMA) is to minimize the delivery time of all vehicles for all tasks, which is given by

$$minimize(\sum_{i=1}^{NV} DT_i) \tag{6}$$

$$DT_i = \sum_{n=2}^{NCODi} DT_{i-n} = \sum_{n=2}^{NCODi} \frac{DIS\left(COD_{i-(n-1)}, COD_{i-n}\right)}{RS\left(COD_{i-(n-1)}, COD_{i-n}, TW_k\right)} \tag{7}$$

where DT_i is the delivery time of all V_i's tasks. Besides, DT_{i-n} is the delivery time from $COD_{i-(n-1)}$ to COD_{i-n}.

IV. DYNAMIC MEMORY MEMETIC ALGORITHM

A. MOTIVATION

Our research objective is to demonstrate that DMMA is effective to address VRPPD with multiple arrival time and traffic congestion constraints. Besides, another objective is to demonstrate the effectiveness and efficiency of DMMA, even in VRPPD with multiple arrival time and non-congestion constraints.

Algorithm 1 Dynamic Memory Memetic Algorithm

Input: population size *PopuSize*, dynamic memory size *DMSize*, the maximum distance of the memetic search K_{max}, the minimum number of shaking S_{min}, the maximum number of shaking S_{max}, the mutation probability p_m, the initial vitality $V_{initial}$, the vitality limit V_{limit}, the congestion-search threshold *CSThr*.
Initialize the population by using Algorithm 2 and evaluate the initial population.
Initialize dynamic memory by using the best individuals of the population.
Define *TempInd* as the temporary-individual variable.
Define k as the current search distance.
REPEAT the following steps, until the termination condition is satisfied.
  FOR each individual in the population
    TempInd $\leftarrow$ the selected individual *T1* from the population.
    FOR $k \leftarrow 1$ *to* K_{max}
      Select an individual from dynamic memory randomly, which is named as *D1*.
      Shaking in Algorithm 3 for *TempInd* and obtain the individual *T2*.
      Local search in Algorithm 5 for *T2* and obtain the individual *T3*.
      Update *D1* in dynamic memory by using Algorithm 9.
      If *T3* is better than *TempInd*, then *TempInd* $\leftarrow$ *T3* and break for-loop. Otherwise, $k \leftarrow k + 1$.
    END
  END
  Except for the best individual, perform mutation in Algorithm 4 according to the mutation probability p_m.
  Update the number of using the fitness functions, and check the termination condition.
END
Output the best individual as the best solution.

Algorithm 2 The Population Initialization

Input: population size *PopuSize*
Step 1: Obtain all tasks in the first batch and randomly allocate these tasks to each vehicle.
Step 2: Repeat *Step 1* for the following batches, sequentially. Then, obtain the coding of one individual.
Step 3: Execute 10 times of swap operators in Fig. 4. If found a better one, update this individual immediately.
Step 4: Repeat *Steps 1-4* until all individuals are generated.

B. FRAMEWORK

The framework of DMMA is given in Algorithm 1. Comparing with memetic algorithms [31], [32], [34], the main differences are as follows. (i) We design dynamic memory to update the population and this is effective to prompt the global-population-search progress. (ii) Our local search is to address congestion by using the local-individual-search method.

First, one feasible task allocation solution is also the coding of one individual in the population, as shown in Fig. 2. In this paper, one coding corresponds to one individual in the population. Moreover, various operator processes may lead to infeasible solutions. There are three kinds of infeasible solutions and their repair strategies, as shown in Fig. 3. These cases are only related to the single-vehicle task sequences and details are as follows. Fig. 3 (a) means that V_i must firstly pick up and then deliver. Fig. 3 (b) indicates that D_2 is not reasonable, because V_i is not allocated to accomplish T_2. Fig. 3 (c) indicates that V_i is not necessary to arrive at D_1 twice only for delivering C_1. Besides, if one infeasible solution is found, we should use the corresponding repair strategies to obtain a feasible solution.

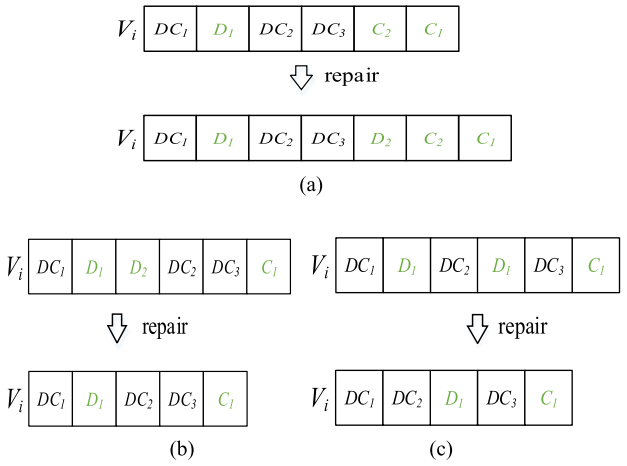


FIGURE 3. Three kinds of infeasible solutions and their repair strategies. Note that, there are two tasks (i.e., $T_1 = \{D_1, C_1, TW_1\}$ and $T_2 = \{D_2, C_2, TW_1\}$). Besides, V_i is the ith vehicle and DC_i may be a depot or a customer.

Secondly, the population initialization is given in Algorithm 2, which randomly allocates tasks to vehicles and uses swap operator in Fig. 4 to optimize a single-vehicle task sequence.

Thirdly, shaking in Algorithm 3 is used to inject the continuously-updated genes of dynamic memory into the population. In each time of shaking, a k-length slice is chosen from a randomly selected individual of dynamic memory. Besides, the shaking illustrations are given in Fig. 5. Note that, k is equal to the current search distance.

Lastly, mutation in Algorithm 4 uses swap operator to improve the population diversity.

C. LOCAL SEARCH TO ADDRESS CONGESTION

Local search in Algorithm 5 consists of single-vehicle and multi-vehicle Hungarian searches. Single-vehicle and

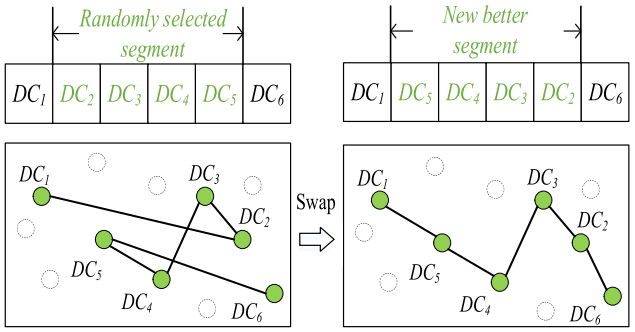


FIGURE 4. The process of swap operator, which randomly selects a segment (i.e., from DC_2 to DC_5) from a single-vehicle sequence. If the swapped segment (i.e., from DC_5 to DC_2) is better than before, then replace the randomly selected segment with the swapped segment.

Algorithm 3 Shaking

Input: the randomly selected individual *D1* from dynamic memory, the current search distance k, the temporary-individual variable *TempInd*, the minimum number of shaking S_{min}, the maximum number of shaking S_{max}.

Step 1: Define a temporary vector as *TV* and $TV \leftarrow TempInd$. Then, randomly generate an integer number S_n, which is from S_{min} to S_{max}.

Step 2: Repeat the next steps, until the S_n times of shaking are accomplished.

Step 3: Randomly selected a single-vehicle sequence with more than k genes from *D1*, as shown from Fig. 5 (a) to Fig. 5 (b). Then, randomly selected a segment with k genes as *Seg*, as shown from Fig. 5 (b) to Fig. 5 (c).

Step 4: Delete each task of *Seg* from *TV* and randomly insert *Seg*, as shown from Fig. 5 (d) to Fig. 5 (e). Then, obtain *TV2* and $TV \leftarrow TV2$.

Algorithm 4 Mutation

Input: population size *PopuSize*, mutation probability p_m.

Step 1: Except for the best individual, randomly select the $INT(PopuSize \times p_m)$ individuals from the population and *INT*() returns an integer.

Step 2: Perform swap operator once for each selected individual. Then, replace the old individual with this new individual.

multi-vehicle Hungarian searches are used to optimize the single-vehicle task sequences and multi-vehicle task sequences, respectively. Besides, Fig. 6 explains how to determine a serious congestion task. This is frequently used in local search. As shown in Fig. 6, we select the most serious congestion paths from a single-vehicle task sequence according to congestion coefficient set J, obtain the corresponding serious congestion tasks, and randomly choose one as the selected serious congestion task.

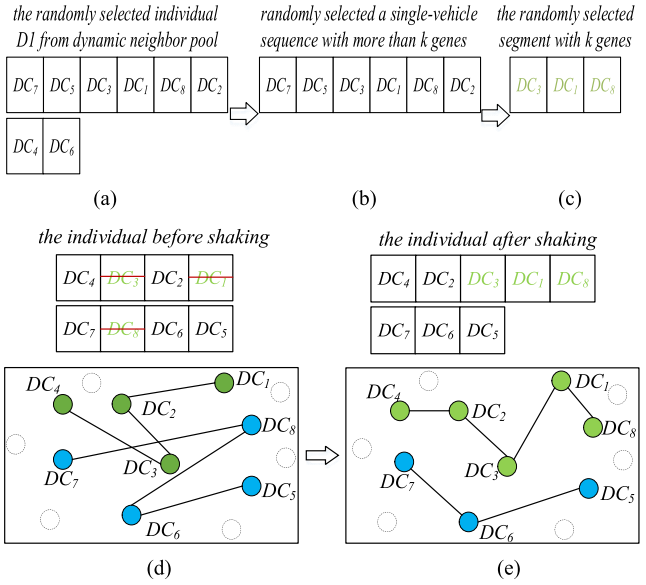


FIGURE 5. The shaking illustrations when the current search distance k is equal to 3.

Algorithm 5 Local Search

Input: the individual *T2*

Step 1: Perform single-vehicle search in Algorithm 6 for *T2* and obtain the individual *T2-A*.

Step 2: Define a temporary individual as *TI*. If *T2-A* is better than *T2*, then $TI \leftarrow T2\text{-}A$. Otherwise, $TI \leftarrow T2$.

Step 3: Perform multi-vehicle Hungarian search in Algorithm 7 for *TI* and then obtain the individual *T2-B*.

Step 4: If *T2-B* is better than *TI*, use *T2-B* as output. Otherwise, use *TI* as output.

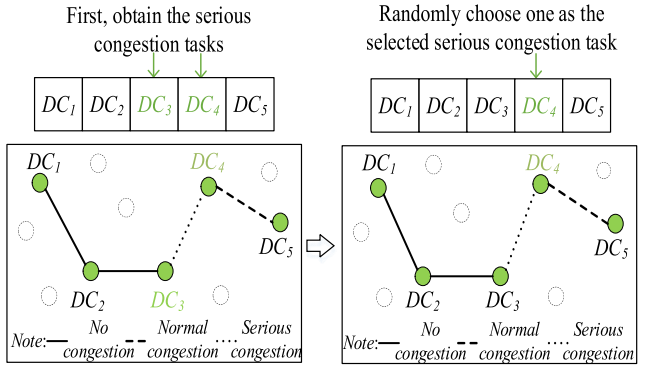


FIGURE 6. An example of determining a serious congestion task for a single-vehicle task sequence.

1) SINGLE-VEHICLE SEARCH

Single-vehicle search in Algorithm 6 uses swap and congestion shift operators. In Algorithm 6, *SVN* and *PON* are defined only to adjust the search efforts, which are equal to 10 and 4, respectively. Besides, congestion shift operator is used to transform the serious congestion tasks into normal-congestion or even non-congestion tasks, as shown in Fig. 7.

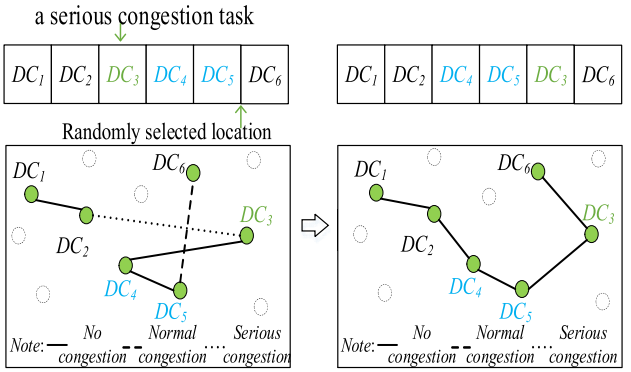


FIGURE 7. The process of congestion shift operator, which randomly selects a serious congestion task (i.e., DC_3) and randomly moves this task. If found a better single-vehicle task sequence, then update.

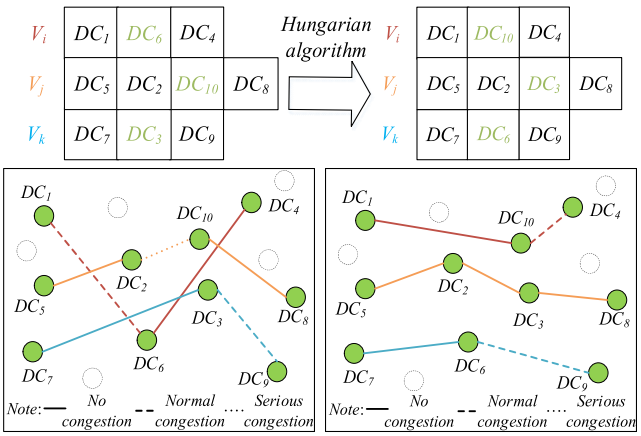


FIGURE 8. The illustration of multi-vehicle Hungarian search to demonstrate the matching function of Hungarian algorithm between vehicles and depot & customer positions.

2) MULTI-VEHICLE HUNGARIAN SEARCH

Multi-vehicle Hungarian search in Algorithm 7 uses Hungarian algorithm [35] in Algorithm 8. Besides, Fig. 8 is used to demonstrate the matching function of Hungarian algorithm between vehicles and depot & customer positions. There are two key points. (i) The meaning of consumption matrix *CM* is as follows. One row of *CM* corresponds to one vehicle, and one column of *CM* corresponds to one depot & customer position. Each element in *CM* is the delivery time of considering *J* for a single-vehicle task sequence. By using V_i in Fig. 8 as an example, CM_{1-1}, CM_{1-2}, and CM_{1-3} are the delivery times of $\{DC_1, DC_6, DC_4\}$, $\{DC_1, DC_{10}, DC_4\}$, and $\{DC_1, DC_3, DC_4\}$, respectively. (ii) With the *CM's* size increasing, the computation cost of Hungarian algorithm increases, significantly. However, we use *CMMaxSize* to restrict its computation cost, which is equal to 4 in this paper.

D. UPDATE DYNAMIC MEMORY

1) DEFINITION AND IMPLEMENTATION

By referencing vitality definition and vitality selection in [36], a new vitality definition and a variant of vitality selection are proposed to update dynamic memory as follows.

The individual vitality in dynamic memory is defined as the cumulative value that records whether to promote populations to find new better solutions or not. As shown in (8) and (9), the individual vitality value varies from ($V_{initial}$ - V_{limit}) to ($V_{initial}$ + V_{limit}). $V_{initial}$ is the initial vitality, V_{limit} is the vitality limit, and they are integers. When an individual in dynamic memory (like *D1* in Algorithm 1) promotes one individual in the population (like *TempInd* and *T3* in Algorithm 1) to find new better solutions, then we increase the individual

Algorithm 6 Single-Vehicle Search

Input: the individual *T2*, the number of the selected vehicles *SVN*, the number of executing operators *PON*, the congestion-search threshold *CSThr*

If the vehicle number of *T2* is greater than *SVN*, then let *SVN* equal to the vehicle number of *T2*.
Define a temporary single-vehicle vector as *TSV*.
REPEAT the following steps for the *SVN* times.
  According to delivery times, use roulette selection and select a single-vehicle task sequence.
  TSV ← the selected single-vehicle task sequence.
  REPEAT the following steps for the *PON* times.
    IF *rand*() < *CSThr* //*rand*() returns a random number ∈ [0,1]
      Execute congestion shift operator for *TSV*.
    ELSE
      Execute swap operator for *TSV*.
    END
    If found a better single-vehicle task sequence, then update *TSV* by
    using this better task sequence.
  END
  If *TSV* is updated in the previous steps, then update *T2* by using *TSV*.
END

Algorithm 7 Multi-Vehicle Hungarian Algorithm

Input: the individual *TI*, the maximum size of consumption matrix *CMMaxSize*, the congestion-search threshold *CSThr*

Step 1: Define the size of the consumption matrix as *CMSize*. If the vehicle number of *TI* is greater than *CMMaxSize*, then *CMSize* ← *CMMaxSize*. Otherwise, *CMSize* ← the vehicle number of *TI*.

Step 2: According to *CMSize*, randomly select the single-vehicle task sequences from *TI* and obtain the selected vehicle set *SelVehicleSet*. Then, use the following process to select the tasks from these sequences and obtain the selected task set *SelTaskSet*. If *rand*() < *CSThr*, select a serious congestion task. Otherwise, select a task randomly.

Step 3: According to these selected sequences and tasks, obtain the consumption matrix *CM*.

Step 4: Execute Hungarian algorithm in Algorithm 8. If found a better solution, update *TI*. Otherwise, don't update.

TABLE 2. Computational complexity analysis.

Operation	Description	Time complexity
Population initialization	In Algorithm 2, initialize *PopuSize* individuals by using swap operator and then evaluate fitness of each individual in population.	$PopuSize \times O(NV) + PopuSize \times O(NDC \times NV)$
Dynamic memory initialization	In Algorithm 1, sort individuals of the population according to their fitness and initialize dynamic memory by using the best individuals.	$O(PopuSize \log PopuSize)$
Shaking	In Algorithm 3, execute up to S_{max} times shaking for the selected individual.	$O(S_{max})$
Local search/Single-vehicle search	In Algorithms 5 and 6, execute congestion shift operator or swap operator *PON* times for *SVN* selected vehicles and evaluate new individuals.	$SVN \times PON \times O(NV) + O(NDC \times NV)$
Local search/Multi-vehicle Hungarian algorithm	In Algorithms 5, 7, and 8, execute Hungarian algorithm for the selected individual. Besides, the size of consumption matrix is restricted as *CMMaxSize* and then evaluate new individuals.	$O(CMMaxSize^3) + O(NDC \times NV)$
Vitality selection	In Algorithm 9, compute and compare the vitality of individuals and then update dynamic memory.	$O(1)$
Population sort	Sort the individuals in population by using quick sort.	$O(PopuSize \log PopuSize)$
Mutation update	In Algorithm 4, execute mutation update for up to *PupoSize* individuals.	$PopuSize \times O(NV)$
Fitness evaluate	By using (6), evaluate each individual in population.	$PopuSize \times O(NDC \times NV)$

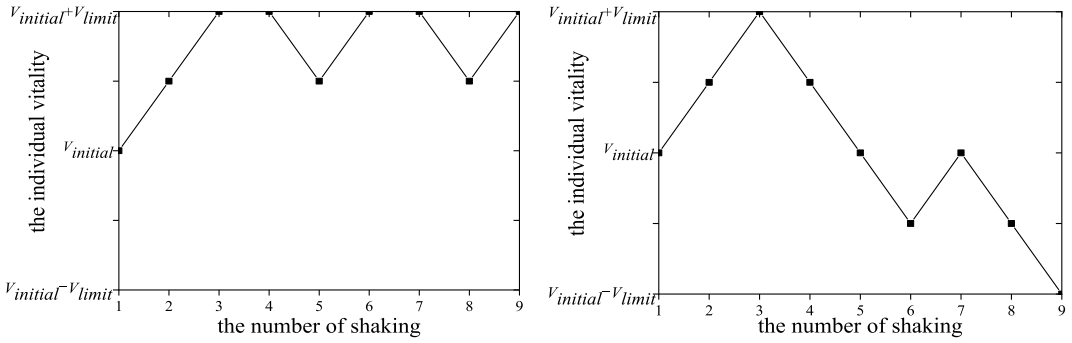


FIGURE 9. The meanings for two kinds of the individual-vitality trend lines. These trend lines are the evidences to prove that one individual in dynamic memory is effective or is not effective to promote the population progress.

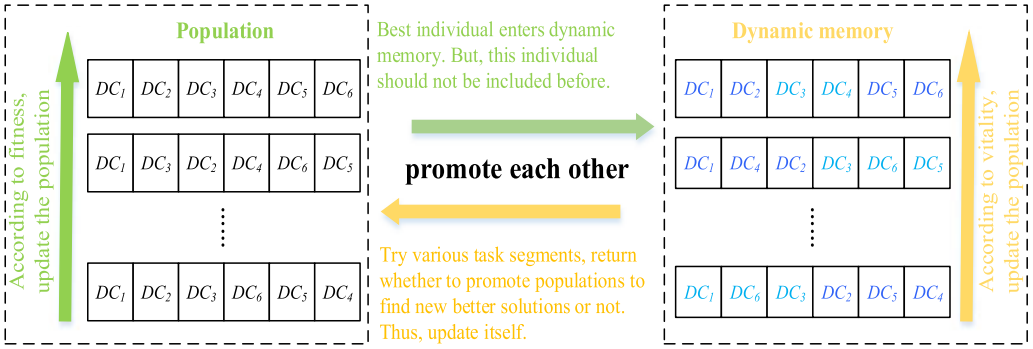


FIGURE 10. An illustrative example to demonstrate the function of dynamic memory in DMMA.

vitality V_d by $+1$. Otherwise, we decrease this individual vitality by -1.

$$V_d = \begin{cases} V_d + 1 & V_d + 1 < (V_{initial} + V_{limit}) \\ V_{initial} + V_{limit} & V_d + 1 \geq (V_{initial} + V_{limit}) \end{cases} \quad (8)$$

$$V_d = \begin{cases} V_d - 1 & V_d - 1 > (V_{initial} - V_{limit}) \\ V_{initial} - V_{limit} & V_d - 1 \leq (V_{initial} - V_{limit}) \end{cases} \quad (9)$$

Vitality selection is used to update dynamic memory in Algorithm 9. Fig. 9 is used to explain the meaning of the individual-vitality trend as follows. (i) In Fig. 9 (a), one individual vitality continuously increases. This kind of

TABLE 3. Test sets used real distribution data from Alibaba Cloud in Shanghai. Besides, *ND* is the number of depots, *NV* is the number of vehicles, and *NT* is the number of tasks. Moreover, the arrival time of the multi-batch goods are 7:00, 9:00, 11:00, 13:00, 15:00, 17:00 and 19:00, respectively.

Small-scale test set					Large-scale test set (Part 1)					Large-scale test set (Part 2)				
Problem	*ND*	*NV*	*NT*	Map size	Problem	*ND*	*NV*	*NT*	Map size	Problem	*ND*	*NV*	*NT*	Map size
P1	3	3	35	38km×25km	P11	12	20	206	46km×51km	P21	24	30	264	48km×44km
P2	3	4	45	36km×42km	P12	12	20	232	42km×49km	P22	24	30	349	57km×45km
P3	3	5	55	39km×33km	P13	12	20	266	45km×53km	P23	24	30	387	49km×54km
P4	3	5	60	34km×41km	P14	12	20	294	41km×55km	P24	24	30	433	54km×60km
P5	3	5	76	45km×43km	P15	12	20	313	54km×43km	P25	24	30	472	58km×62km
P6	7	10	88	39km×48km	P16	18	25	227	45km×47km	P26	30	45	230	53km×58km
P7	7	10	118	43km×34km	P17	18	25	284	52km×44km	P27	30	45	296	50km×47km
P8	7	8	75	31km×24km	P18	18	25	335	53km×42km	P28	30	45	359	53km×61km
P9	7	9	96	38km×35km	P19	18	25	371	46km×53km	P29	30	45	403	51km×48km
P10	7	10	156	36km×49km	P20	18	25	398	49km×57km	P30	30	45	484	59km×62km

TABLE 4. Used parameter setting in this paper.

Items	Parameters
MAPSO	population size *PopuSize* =50, the maximum neighborhood structures l_{max}=5.
LSVNS	population size *PopuSize* =50, *k* for the Nbr1(*x*) branch=5, *k* for the Nbr2(*x*) branch=20.
MTWPS	population size *PopuSize*=50, the approach rate *Step*=5, the local search scale *R*=5, the elitism quantity *N'* = 10, the number of weak wolves *N** = 10.
BRKGA	population size *PopuSize*=50, the elite probability p_e= 0.25, the crossover probability p_c= 0.7, the mutation probability p_m = 0.2.
DMMA	population size *PopuSize*=50, dynamic memory size *DMSize* =10, the maximum distance of the memetic search K_{max}=6, the minimum number of shaking S_{min}=5, the maximum number of shaking S_{max}=10, the mutation probability p_m=0.05, the initial vitality $V_{initial}$= 5, the vitality limit V_{limit}= 5, the congestion-search threshold *CSThr* =0.3.
Termination condition	We used termination generation *TG* =200 (i.e., the number of using the fitness functions = 10000) as termination condition.
Program and computer	All algorithms were programmed in Matlab R2012a. Meanwhile, we executed them on Intel® Xeon® CPU X5670@2.93GHz, 2GB ARM, and Windows 7 in VMware.

TABLE 5. Wilcoxon results for the best-fitness data of 20 runs between one compared algorithm and DMMA.

Small-scale test set												
Algorithm	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	*Sum*(+)	*Sum*(-)
MAPSO	≈	+	+	+	+	+	+	+	+	+	9	0
LSVNS	≈	≈	≈	+	+	+	+	+	+	+	7	0
MTWPS	+	+	+	+	+	+	+	+	+	+	10	0
BRKGA	≈	+	+	+	+	+	+	+	+	+	9	0
Large-scale test set (Part 1)												
Algorithm	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	*Sum*(+)	*Sum*(-)
MAPSO	+	+	+	+	+	+	+	+	+	+	10	0
LSVNS	+	+	+	+	+	+	+	+	+	+	10	0
MTWPS	+	+	+	+	+	+	+	+	+	+	10	0
BRKGA	+	+	+	+	+	+	+	+	+	+	10	0
Large-scale test set (Part 2)												
Algorithm	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30	*Sum*(+)	*Sum*(-)
MAPSO	+	+	+	+	+	+	+	+	+	+	10	0
LSVNS	+	+	+	+	+	+	+	+	+	+	10	0
MTWPS	+	+	+	+	+	+	+	+	+	+	10	0
BRKGA	+	+	+	+	+	+	+	+	+	+	10	0

TABLE 6. Wilcoxon results for the average computation times of 20 runs between one compared algorithm and DMMA.

Small-scale test set												
Algorithm	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	*Sum*(+)	*Sum*(-)
MAPSO	+	+	+	+	+	+	+	+	+	+	10	0
LSVNS	+	+	+	+	+	+	+	+	+	+	10	0
MTWPS	≈	≈	+	+	+	+	+	+	+	+	8	0
BRKGA	-	-	≈	≈	≈	+	+	+	+	+	5	2
Large-scale test set (Part 1)												
Algorithm	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	*Sum*(+)	*Sum*(-)
MAPSO	+	+	+	+	+	+	+	+	+	+	10	0
LSVNS	+	+	+	+	+	+	+	+	+	+	10	0
MTWPS	≈	≈	+	+	+	+	≈	+	+	+	7	0
BRKGA	+	+	+	+	+	+	+	+	+	+	10	0
Large-scale test set (Part 2)												
Algorithm	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30	*Sum*(+)	*Sum*(-)
MAPSO	+	+	+	+	+	+	+	+	+	+	10	0
LSVNS	+	+	+	+	+	+	+	+	≈	+	9	0
MTWPS	+	+	+	+	+	+	+	+	≈	+	9	0
BRKGA	+	+	+	+	+	+	+	+	+	+	10	0

Algorithm 8 Hungarian Algorithm

Input: the consumption matrix *CM*, the selected task set *SelTaskSet*, the selected vehicle set *SelVehicleSet*.

Output: the output result that assigns tasks (row) to vehicles (column).

Step 1: In each row of *CM*, subtract the smallest element of the row.

Step 2: In each column of *CM*, subtract the smallest element of the column.

Step 3: Draw lines through the rows and columns that have the 0 elements by using the fewest lines.

Step 4: If there are *CMSize* lines drawn, then the optimal assignment of 0 elements is the assignment scheme as the output result. The 0 element in column h of line g represents we should allocate the $SelTaskSet_h$ to the $SelVehicleSet_g$. Otherwise, go to Step 5.

Step 5: Find the smallest element in *CM*, which is not covered by any line. Subtract this element from each row that isn't covered and add it to each column that is covered. Then, go to Step 3.

Algorithm 9 Vitality Selection to Update Dynamic Memory

Input: the individual *D1* in dynamic memory, the individual *TempInd* and the individual *T3*, the initial vitality $V_{initial}$, the vitality limit V_{limit}

Step 1: According to *TempInd* and *T3*, update the vitality value of the individual *D1* by using (8) or (9).

Step 2: If the vitality value of the individual *D1* is equal to ($V_{initial}$ - V_{limit}), then replace *D1* with the selected individual from the population. Note that, there are two requirements for this selected individual as follows. One is that dynamic memory does not include this selected individual before replacing *D1*. The other is that, if there are more than one individual that satisfy the previous requirement, then select the best fitness individual from these individuals.

individual-vitality trend lines is the evidence to prove that this individual is effective to promote the population progress. Or to say, this kind of task segments that are inserted into the population is effective to prevent populations from trapping in local optima. Note that, these task segments are inserted as a whole (like Fig. 5). Thus, this changes the arrangement layout of the inserted individuals in the population. (ii) Otherwise, when one individual vitality continuously decreases (like Fig. 9 (b)), this kind of individual-vitality trend lines indicates that this individual is not effective to promote the population progress. Then, when its vitality is equal to ($V_{initial}$ - V_{limit}), this individual is discarded.

2) PRINCIPLE ANALYSIS

Fig. 10 demonstrates the function of dynamic memory and explains how to establish an effective acceleration mechanism for promoting the population search. This acceleration mechanism is goal-oriented by using its two kinds of updating criteria as follows. (i) According to fitness, one search effort updates the population. (ii) According to whether to promote populations to find new better solutions or not, another search effort updates dynamic memory.

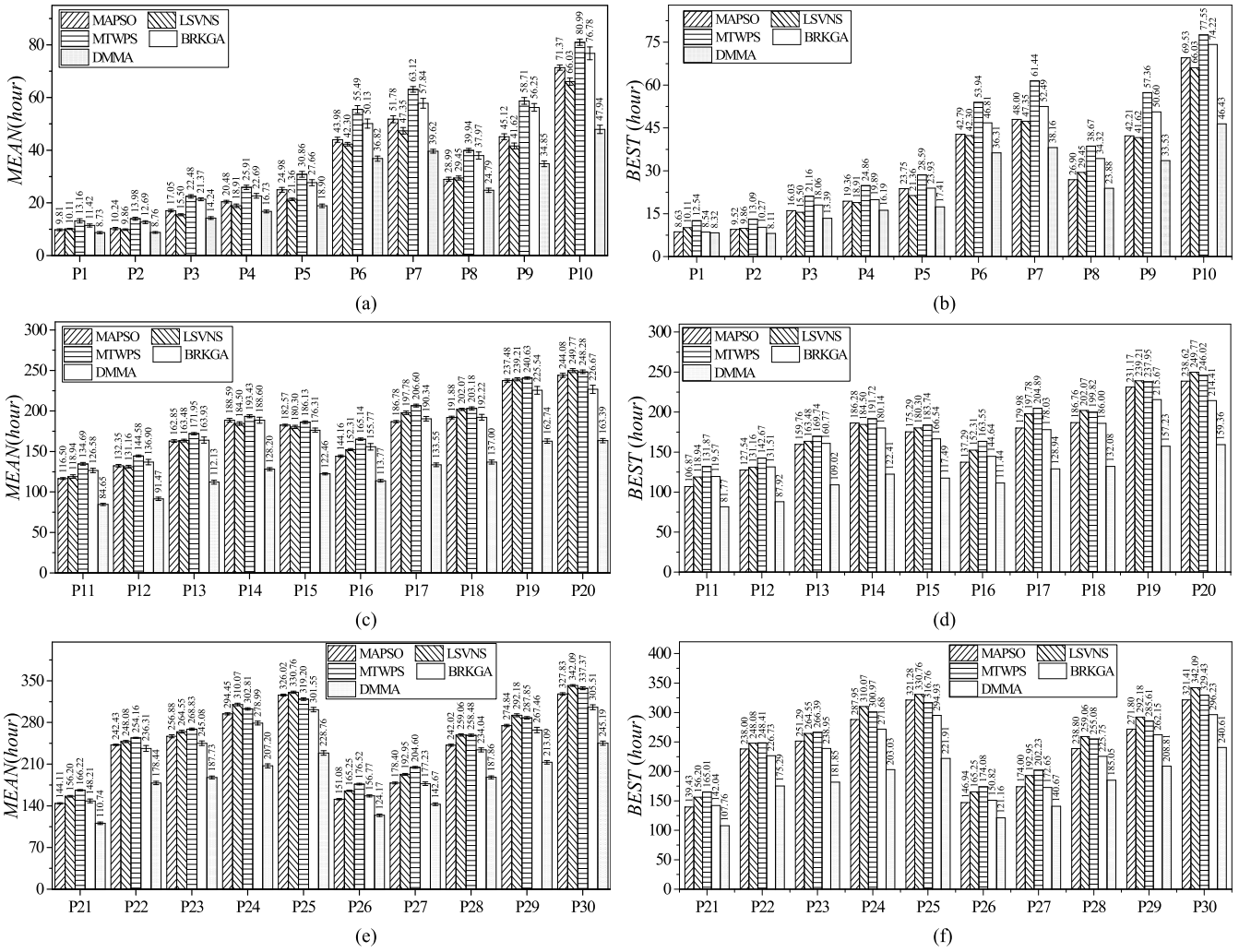


FIGURE 11. Non-congestion *MEAN* & *BEST* results on algorithm effectiveness.

Explanations are as follows. (i) The premature phenomena of populations are one of the major obstacles in evolutionary computation. Different schemes of protecting population diversity are widely used that do not be effective to find new better solutions in all cases. For example, injecting new genes into populations is not able to ensure the population to find new better solutions, definitely. (ii) However, from an updating-criterion perspective, better individuals in populations usually enter dynamic memory in the early stage of population evolution. This is effective to promote individuals to make progress. Then, in the later stage, better individuals do not enter dynamic memory, because dynamic memory has included them. Thus, many individuals with gene diversity and poor fitness could enter dynamic memory. Different from better individuals, they may include various task segments. These task segments could improve the probability of finding new better solutions.

E. COMPUTATIONAL COMPLEXITY

The computational complexity of the proposed model with DMMA in (10) is dependent on the test set size and DMMA's parameters [37], such as the number of depot and customer set *NDC*, the number of vehicle set *NV*, termination generation *TG*, population size *PopuSize*, the maximum distance of the memetic search K_{max}, the maximum number of shaking S_{max}, and the maximum size of consumption matrix *CMMaxSize*. In (10), $T()$ returns the computational complexity of one algorithm's item. Besides, Table 2 gives the computational complexity of each item in (10). Thus, the total time complexity of the proposed model with DMMA could be obtained by using (11).

$$\begin{aligned} &T\ (\text{DMMA}) \\ &= T\ (\text{PopulationInitialization}) \\ &\quad + T\ (\text{DynamicMemoryInitialization}) \\ &\quad + TG \times \left[PopuSize \times K_{max} \times \left[T\ (\text{Shaking})\right.\right. \\ &\quad + T\ (\text{LocalSearch}) + T\ (\text{VitalitySelection})\left.\right] \\ &\quad + T\ (\text{PopulationSort}) + T\ (\text{MutationUpdate}) \\ &\quad \left. + T\ (\text{FitnessEvaluate})\right] \end{aligned} \tag{10}$$

$$\begin{aligned} &T\ (\text{DMMA}) \\ &= PopuSize \times [O\,(NV) + O\,(NDC \times NV)] \\ &\quad + O\,(PopuSize \log PopuSize) + TG \end{aligned}$$

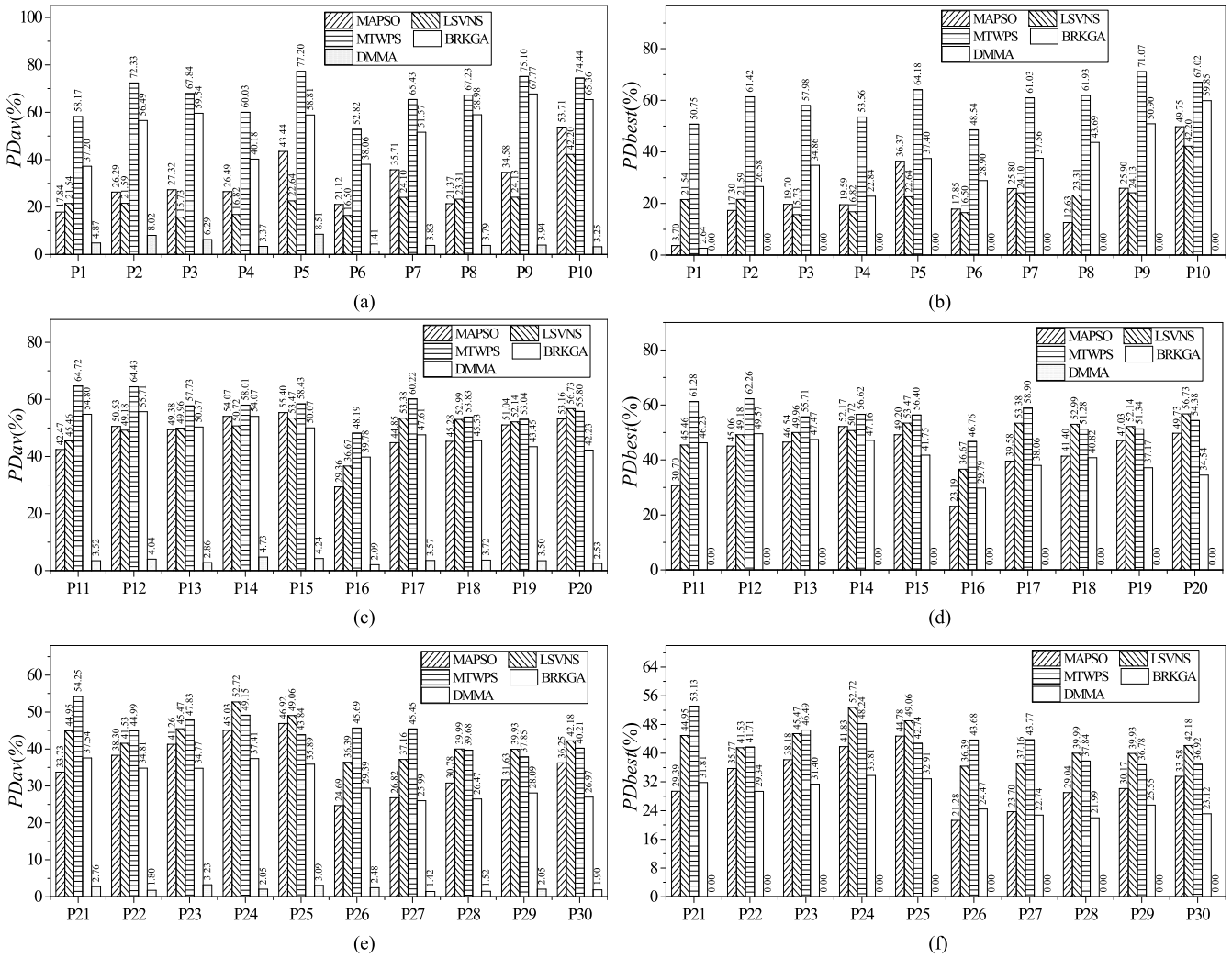


FIGURE 12. Non-congestion *PDbest* & *PDav* results on algorithm effectiveness.

$$
\begin{aligned}
&\times \Big[PopuSize \times K_{max} \times \Big[O\left(S_{max}\right) \\
&+ \Big(SVN \times PON \times O\left(NV\right) + O\left(NDC \times NV\right) \\
&+ O\left(CMMaxSize^3 \right) + O\left(NDC \times NV\right) \Big) \\
&+ O\left(1\right) \Big] + O\left(PopuSize \log PopuSize\right) + PopuSize \\
&\times O\left(NV\right) + PopuSize \times O\left(NDC \times NV\right) \Big] \\
\approx\ & TG \times \Big[PopuSize \times K_{max} \times \Big(O\left(S_{max}\right) + O\left(CMMaxSize^3 \right) \\
&+ O\left(NDC \times NV\right) \Big) \\
&+ O\left(PopuSize \log PopuSize\right) \Big] \qquad (11)
\end{aligned}
$$

V. EXPERIMENTS AND DISCUSSIONS

A. EXPERIMENTAL DESIGN

1) TEST SETS

We used distribution data from Alibaba Cloud [38] in Shanghai and constructed test sets with 30 test problems, as shown in Table 3. Congestion coefficient sets were from Baidu Map [39] on Monday, Wednesday, and Sunday. Test and congestion coefficient sets could be downloaded from [40].

2) COMPARED ALGORITHMS

We used MAPSO [41], LSVNS [42], MTWPS [43], and BRKGA [44] as compared algorithms. Because our model is a new variant of VRPPD, the coding method and repair strategies of infeasible solutions are different from other researches. Therefore, to compare with DMMA objectively, these compared algorithms used the same coding method in Fig. 2, the population initialization in Algorithm 2, and the repair strategies in Fig. 3.

3) PARAMETER SETTING

B. NON-CONGESTION RESULTS

We tested the effectiveness and efficiency of DMMA under the non-congestion condition. Note that, the non-congestion condition means that all elements in congestion coefficient set J were equal to 1.

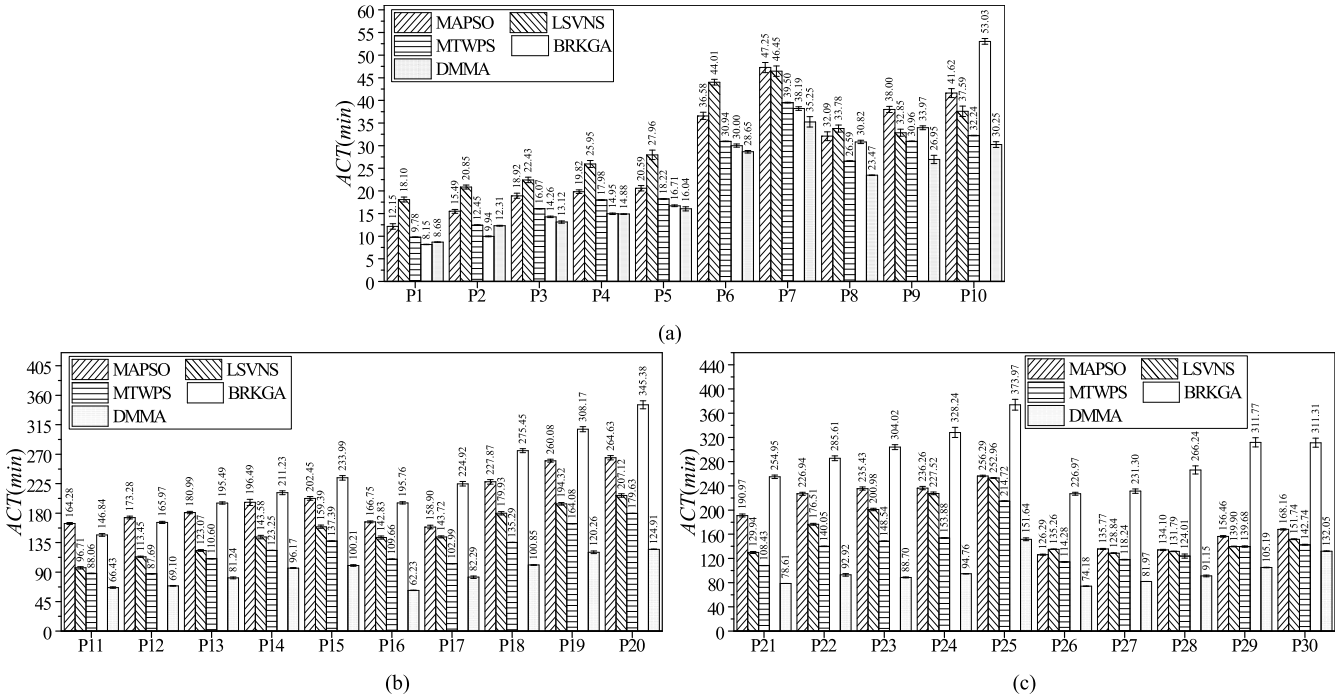


FIGURE 13. Non-congestion *ACT* results on algorithm efficiency.

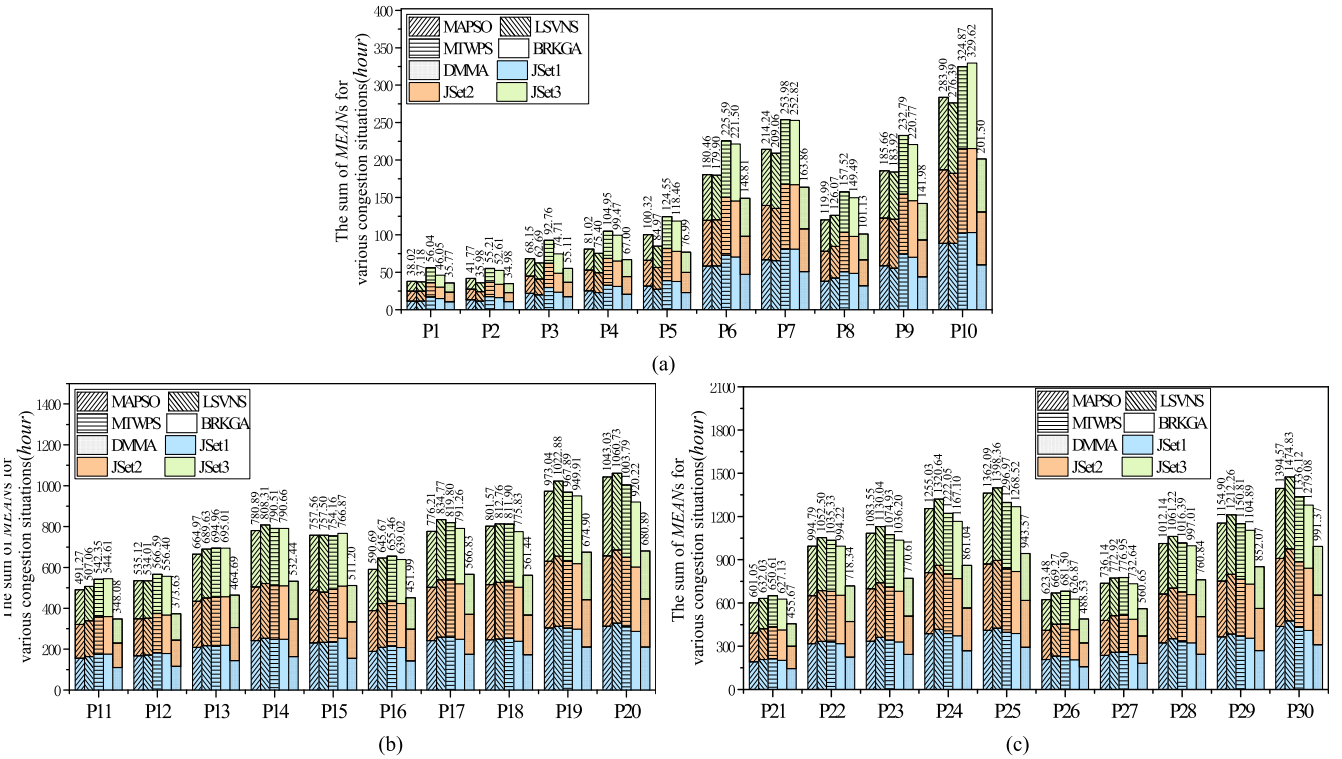
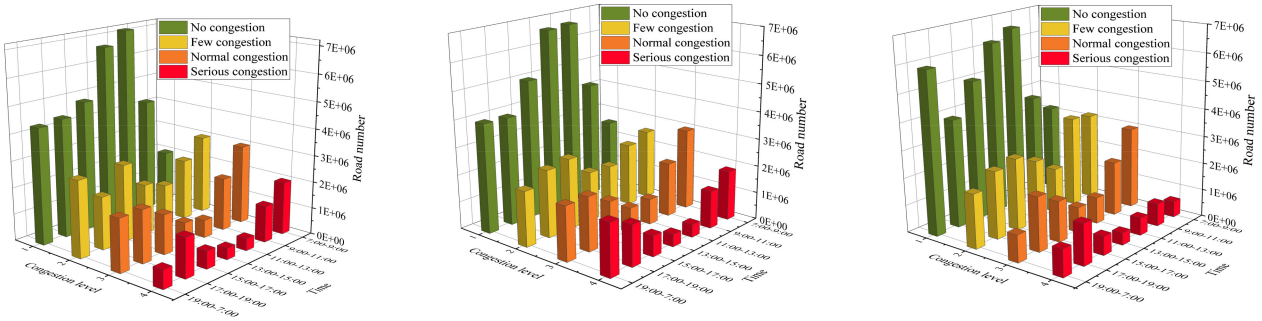


FIGURE 14. Delivery time results to demonstrate the performance of DMMA for various congestion situations.

1) ON ALGORITHM EFFECTIVENESS

Figs. 11 and 12 provide the effectiveness results. *MEAN* is the mean of the best individual fitness for 20 runs and *BEST* is the best solution for 20 runs. Besides, the percentage deviations of the average solution (*PDav*) and the best solution (*PDbest*) are given in (12) and (13). The best known



(a) JSet1 on Monday (b) JSet2 on Wednesday (c) JSet3 on Sunday

FIGURE 15. The histograms of congestion coefficient sets that used real congestion data in 24 hours from Baidu Map in Shanghai.

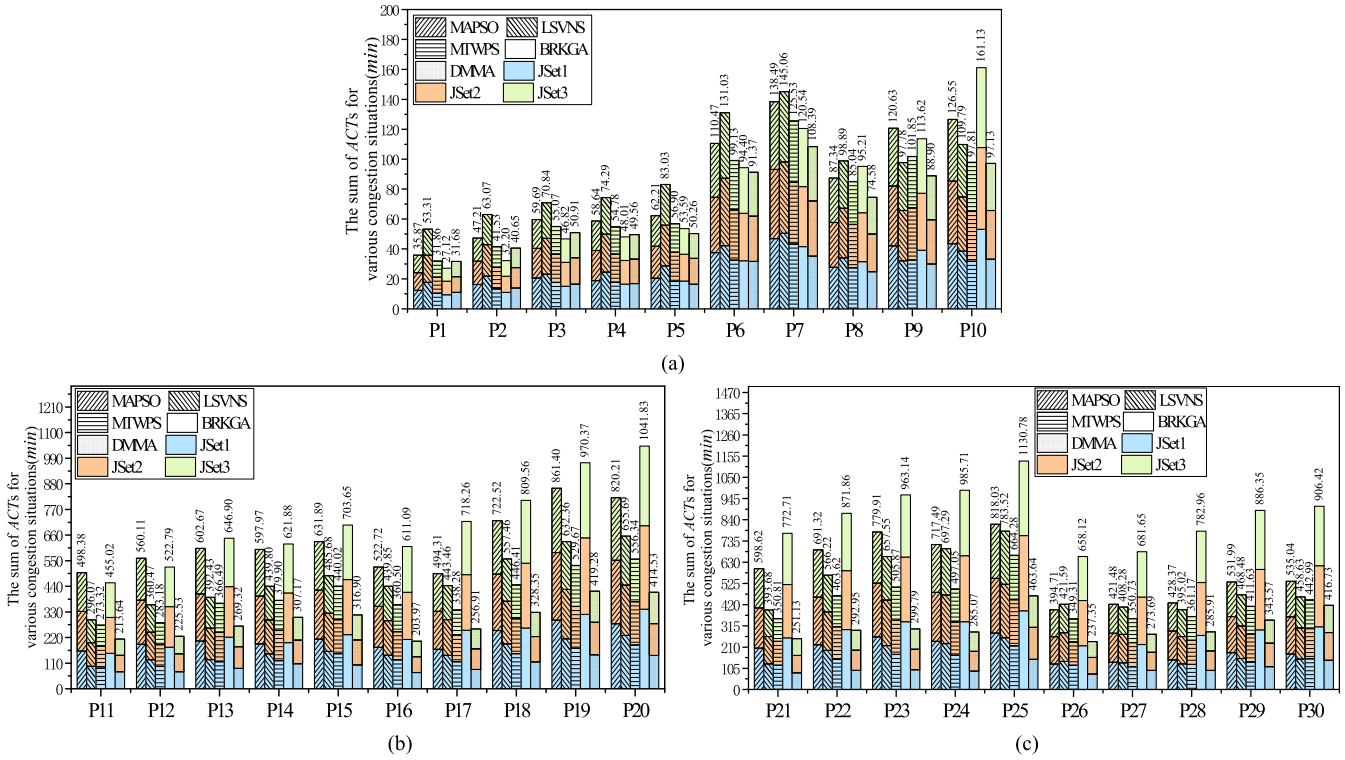


FIGURE 16. The *ACT* results to demonstrate the performance of DMMA for various congestion situations.

solution (*BKS*) was obtained according to the results of all algorithms. In principle, *PDav* and *PDbest* express the algorithm's capability to obtain *BKS*. The smaller *MEAN* (*BEST*, *PDav*, or *PDbest*), the better algorithm effectiveness. These results in Figs. 11 and 12 indicate that DMMA is effective to obtain the best solutions, even for VRPPD with multiple arrival time and non-congestion constraints.

$$PDav = \frac{MEAN - BKS}{BKS} \times 100 \tag{12}$$

$$PDbest = \frac{BEST - BKS}{BKS} \times 100 \tag{13}$$

To further demonstrate the effectiveness of DMMA, we used Wilcoxon test at 0.05 level of significance for the best-fitness data of 20 runs between one compared algorithm and DMMA, as shown in Table 5. The symbols of (+), (≈), and (−) mean that DMMA is better than, approximately equal to, and worse than the compared algorithm, respectively. DMMA obtained the best performance in all cases. The reason is that DMMA uses the effective acceleration mechanism for promoting the population search, as shown in Fig. 10.

2) ON ALGORITHM EFFICIENCY

Fig. 13 presents the efficiency results of the average computation times (*ACT*s) for 20 runs. The smaller *ACT*, the better algorithm efficiency. These results in Fig. 13 indicate that

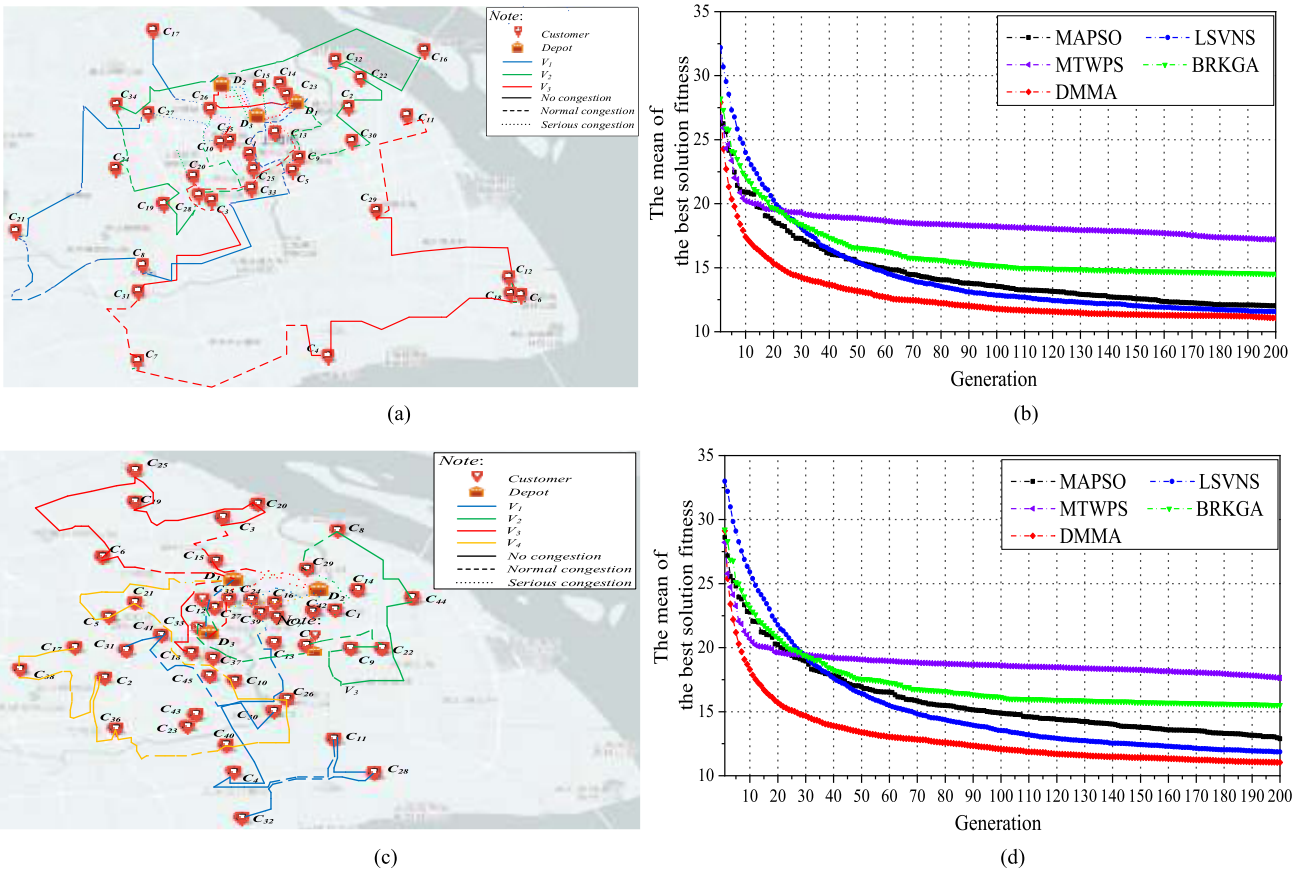


FIGURE 17. The real distribution cases and their convergence curves of DMMA for P1 and P2 test problems. (a) The map route of the best solution of DMMA for P1 test problem. (b) The convergence curves for P1 test problem. (c) The map route of the best solution of DMMA for P1 test problem. (d) The convergence curves for P1 test problem.

DMMA has lower computation costs, especially for large scale test set.

To further demonstrate the efficiency of DMMA, we also used Wilcoxon test at 0.05 level of significance, as shown in Table 6. DMMA obtained the best performance in most cases. The reason is that the computation costs of single-vehicle search, multi-vehicle Hungarian search, and vitality selection are lower, such as the shift operator in Algorithm 6, the swap operator in Algorithm 6, and the Hungarian algorithm only with a short task sequence (i.e., *CMSize*) in Algorithm 8, updating the individual vitality (i.e., (8) and (9)).

C. VARIOUS-CONGESTION-SITUATION RESULTS

We used various congestion coefficient sets (i.e., JSet1, JSet2, AND JSet3) to test DMMA as follows.

1) DELIVERY TIME AND SENSITIVITY ANALYSIS

The best fitness in (6) is the delivery time, which is the most important index to express the pickup & delivery efficiency of real distribution systems. The shorter delivery time, the better pickup & delivery efficiency. Fig. 14 provides the delivery time results. In Fig.14, the sum of *MEAN*s for various congestion situations in 20 runs is used as the evaluation criterion, while each kind of *MEAN*s for each congestion coefficient set is also given. DMMA obtained the best performance in all cases. The reason is that DMMA uses effective schemes to determine serious congestion tasks and address traffic congestion, as shown in Figs. 6, 7, and 8. These results in Fig. 14 indicate that DMMA is robust and effective for VRPPD with multiple arrival time and various congestion constraints.

To demonstrate the robustness of DMMA for various distribution scenarios and various congestion situations, the sensitivity-analysis explanations were as follows. (i) From a distribution perspective, there were various kinds of depot's numbers (i.e., 3, 7, 12, 18, 24, 30), vehicle's numbers (i.e., 3, 4, 5, 8, 9, 10, 20, 25, 30, 45), and task's numbers (i.e., 35, 45, 55, ..., 484) in 30 test problems, as shown in Table 3. (ii) From a congestion perspective, we used various congestion coefficient sets, as shown in Fig. 15. Three kinds of congestion situations were the representatives of our daily traffic congestion in one week, because they corresponded to the traffic congestion situations of Monday, Wednesday, and Sunday. These indicate that DMMA is robust, even for the changes of distribution scenarios and congestion situations.

2) COMPUTATION TIME AND ALGORITHM USABILITY

Fig. 16 gives the sum of *ACT*s for various congestion situations in 20 runs, while each kind of *ACT*s for each congestion coefficient set was also given. DMMA obtained the best results in most cases. Generally, *ACT*s of DMMA for various congestion situations were similar to non-congestion *ACT*s. For example, *ACT*s of DMMA for JSet1 were roughly the same with non-congestion *ACT*s in Fig. 13.

On the other hand, we discussed the usability of DMMA as follows. The maximum value of all *ACT*s in Fig. 16 was 158.88min for P25. In P25, there are 24 depots, 30 vehicles, and 472 tasks. From a computation efficiency perspective, this *ACT*'s maximum value is small enough for the size of this test problem. From an application perspective, this means that DMMA is able to complete task allocations within acceptable time, because real distribution scenarios are able to obtain all pickup & delivery tasks' information at least one day in advance.

To further demonstrate the usability of DMMA in real applications, Fig. 17 shows the real cases by using map route and their convergence curves. These maps were all based on real pickup & delivery data and real congestion data. In Fig. 17 (a) and (c), each kind of the color paths corresponded to one vehicle and each kind of line types corresponded to one kind of congestion degree. Meanwhile, Fig. 17 (b) and (d) indicate that the convergence performance of DMMA is satisfactory. Generally speaking, Fig. 17 displays a fleet pickup & delivery task allocation and this implies that DMMA is able to be used in the real distribution cases.

VI. CONCLUSION

We formulate VRPPD with multiple arrival time and traffic congestion constraints. Moreover, our model is able to express the fast-delivery requirements of the modern distribution systems in our production and life, such as large-scale emergency relief supplies distribution. As the extensive results shown, DMMA is able to effectively address the real pickup & delivery application problems with large-scale tasks, the multi-batch arrival of goods, and time-dependent traffic congestion. The robustness and usability of DMMA are proved by non-congestion and congestion tests, which are based on the real distribution data and traffic congestion data in Shanghai.

Recently, the deployment of unmanned pickup & delivery platforms gradually increases in the VRPPD applications. Our future researches focus on online pickup-delivery task allocations, which are able to respond to the newly added tasks and real-time traffic congestion.

REFERENCES

[1] W. P. Nanry and J. Wesley Barnes, "Solving the pickup and delivery problem with time windows using reactive tabu search," *Transp. Res. B, Methodol.*, vol. 34, no. 2, pp. 107–121, Feb. 2000.

[2] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transp. Sci.*, vol. 40, no. 4, pp. 455–472, Nov. 2006.

[3] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte, "Static pickup and delivery problems: A classification scheme and survey," *TOP*, vol. 15, no. 1, pp. 1–31, May 2007.

[4] S. Karakatiä and V. Podgorelec, "A survey of genetic algorithms for solving multi depot vehicle routing problem," *Appl. Soft Comput.*, vol. 27, pp. 519–532, Feb. 2015.

[5] A. L. Kok, E. W. Hans, and J. M. J. Schutten, "Vehicle routing under time-dependent travel times: The impact of congestion avoidance," *Comput. Oper. Res.*, vol. 39, no. 5, pp. 910–918, May 2012.

[6] B. Soylu, "A general variable neighborhood search heuristic for multiple traveling salesmen problem," *Comput. Ind. Eng.*, vol. 90, pp. 390–401, Dec. 2015.

[7] X. Xu, H. Yuan, M. Liptrott, and M. Trovati, "Two phase heuristic algorithm for the multiple-travelling salesman problem," *Soft Comput.*, vol. 22, no. 19, pp. 6567–6581, Oct. 2018.

[8] A. Hoff, I. Gribkovskaia, G. Laporte, and A. Løkketangen, "Lasso solution strategies for the vehicle routing problem with pickups and deliveries," *Eur. J. Oper. Res.*, vol. 192, no. 3, pp. 755–766, Feb. 2009.

[9] K. Treleaven, M. Pavone, and E. Frazzoli, "Asymptotically optimal algorithms for One-to-One pickup and delivery problems with applications to transportation systems," *IEEE Trans. Autom. Control*, vol. 58, no. 9, pp. 2261–2276, Sep. 2013.

[10] M. Sahin, G. Cavuslar, T. Oncan, G. Sahin, and D. Tuzun Aksu, "An efficient heuristic for the multi-vehicle one-to-one pickup and delivery problem with split loads," *Transp. Res. C, Emerg. Technol.*, vol. 27, pp. 169–188, Feb. 2013.

[11] M. N. Haddad, R. Martinelli, T. Vidal, S. Martins, L. S. Ochi, M. J. F. Souza, and R. Hartl, "Large neighborhood-based metaheuristic and branch-and-price for the pickup and delivery problem with split loads," *Eur. J. Oper. Res.*, vol. 270, no. 3, pp. 1014–1027, Nov. 2018.

[12] S. Naccache, J.-F. Cáté, and L. C. Coelho, "The multi-pickup and delivery problem with time windows," *Eur. J. Oper. Res.*, vol. 269, no. 1, pp. 353–362, Aug. 2018.

[13] E. Bartolini, L. Bodin, and A. Mingozzi, "The traveling salesman problem with pickup, delivery, and ride-time constraints," *Networks*, vol. 67, no. 2, pp. 95–110, Mar. 2016.

[14] A. Ahkamiraad and Y. Wang, "Capacitated and multiple cross-docked vehicle routing problem with pickup, delivery, and time windows," *Comput. Ind. Eng.*, vol. 119, pp. 76–84, May 2018.

[15] H. K. Chen, H. W. Chou, C. F. Hsueh, and Y. J. Yu, "The paired many-to-many pickup and delivery problem: An application," *Top*, vol. 23, no. 1, pp. 220–243, Apr. 2015.

[16] N. Ghaffarinasab, T. Van Woensel, and S. Minner, "A continuous approximation approach to the planar hub location-routing problem: Modeling and solution algorithms," *Comput. Oper. Res.*, vol. 100, pp. 140–154, Dec. 2018.

[17] F. Liu, M. Gui, C. Yi, and Y. Lan, "A fast decomposition and reconstruction framework for the pickup and delivery problem with time windows and LIFO loading," *IEEE Access*, vol. 7, pp. 71813–71826, 2019.

[18] F. Zhou, Y. He, and L. Zhou, "Last mile delivery with stochastic travel times considering dual services," *IEEE Access*, vol. 7, pp. 159013–159021, 2019.

[19] X. Wang, "The heterogeneous fleet location routing problem with simultaneous pickup and delivery and overloads," *Discrete Continuous Dyn. Syst.*, vol. 12, nos. 4–5, pp. 1147–1166, 2019.

[20] L. Zhu and J.-B. Sheu, "Failure-specific cooperative recourse strategy for simultaneous pickup and delivery problem with stochastic demands," *Eur. J. Oper. Res.*, vol. 271, no. 3, pp. 896–912, Dec. 2018.

[21] M. Marinaki and Y. Marinakis, "A glowworm swarm optimization algorithm for the vehicle routing problem with stochastic demands," *Expert Syst. Appl.*, vol. 46, pp. 145–163, Mar. 2016.

[22] E. Ruiz Y Ruiz, I. García-Calvillo, and S. Nucamendi-Guillén, "Open vehicle routing problem with split deliveries: Mathematical formulations and a cutting-plane method," *Oper. Res.*, vol. 5, pp. 1–21, Jun. 2020.

[23] S. Yanik, B. Bozkaya, and R. deKervenoael, "A new VRPPD model and a hybrid heuristic solution approach for e-tailing," *Eur. J. Oper. Res.*, vol. 236, no. 3, pp. 879–890, Aug. 2014.

[24] J. Guo and C. Liu, "Time-dependent vehicle routing of free pickup and delivery service in flight ticket sales companies based on carbon emissions," *J. Adv. Transp.*, vol. 2017, pp. 1–14, Dec. 2017.

[25] M. Mahmoudi and X. Zhou, "Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations," *Transp. Res. B, Methodol.*, vol. 89, pp. 19–42, Jul. 2016.

[26] Z. Sifa, C. Jiandong, L. Xiaomin, and L. Keqiang, "Urban pickup and delivery problem considering time-dependent fuzzy velocity," *Comput. Ind. Eng.*, vol. 60, no. 4, pp. 821–829, May 2011.
[27] Y. Xiao and A. Konak, "The heterogeneous green vehicle routing and scheduling problem with time-varying traffic congestion," *Transp. Res. E, Logistics Transp. Rev.*, vol. 88, pp. 146–166, Apr. 2016.
[28] P. Sun, L. P. Veelenturf, S. Dabia, and T. Van Woensel, "The time-dependent capacitated profitable tour problem with time windows and precedence constraints," *Eur. J. Oper. Res.*, vol. 264, no. 3, pp. 1058–1073, Feb. 2018.
[29] S. Zhang, D. Mu, and C. Wang, "A solution for the full-load collection vehicle routing problem with multiple trips and demands: An application in beijing," *IEEE Access*, vol. 8, pp. 89381–89394, 2020.
[30] R. P. Hornstra, A. Silva, K. J. Roodbergen, and L. C. Coelho, "The vehicle routing problem with simultaneous pickup and delivery and handling costs," *Comput. Oper. Res.*, vol. 115, Mar. 2020, Art. no. 104858.
[31] B. Peng, Y. Zhang, Z. Lü, T. C. E. Cheng, and F. Glover, "A learning-based memetic algorithm for the multiple vehicle pickup and delivery problem with LIFO loading," *Comput. Ind. Eng.*, vol. 142, Apr. 2020, Art. no. 106241.
[32] A. A. Anwar and I. Younas, "Optimization of many objective pickup and delivery problem with delay time of vehicle using memetic decomposition based evolutionary algorithm," *Int. J. Artif. Intell. Tools*, vol. 29, no. 01, Feb. 2020, Art. no. 2050003.
[33] *Baidu*. [Online]. Available: http://jiaotong.baidu.com/top/report/?citycode=289
[34] S. K. Baliarsingh, W. Ding, S. Vipsita, and S. Bakshi, "A memetic algorithm using emperor penguin and social engineering optimization for medical data classification," *Appl. Soft Comput.*, vol. 85, Dec. 2019, Art. no. 105773.
[35] J. Gil-Aluja, "The Hungarian assignment algorithm," in *The Interaction Management Human Resources Uncertainty*, J. Gil-Aluja, Ed. Boston, MA, USA: Springer, 1998, pp. 148–158.
[36] H. Zhang and J. Zhou, "Dynamic multiscale region search algorithm using vitality selection for traveling salesman problem," *Expert Syst. Appl.*, vol. 60, pp. 81–95, Oct. 2016.
[37] S. K. Baliarsingh, S. Vipsita, K. Muhammad, B. Dash, and S. Bakshi, "Analysis of high-dimensional genomic data employing a novel bio-inspired algorithm," *Appl. Soft Comput.*, vol. 77, pp. 520–532, 2019.
[38] *Alibaba Cloud*. [Online]. Available: https://tianchi.aliyun.com/competition/entrance/231581/information
[39] *Baidu*. [Online]. Available: http://lbsyun.baidu.com/index.php?title=jspopular
[40] *Github*. [Online]. Available: https://github.com/1654402787/VRPPD-with-multiple-arrival-time-and-traffic-congestion
[41] Y. Marinakis, M. Marinaki, and A. Migdalas, "A multi-adaptive particle swarm optimization for the vehicle routing problem with time windows," *Inf. Sci.*, vol. 481, pp. 311–329, May 2019.
[42] X. Wang, B. Golden, and E. Wasil, "The min-max multi-depot vehicle routing problem: Heuristics and computational results," *J. Oper. Res. Soc.*, vol. 66, no. 9, pp. 1430–1441, Sep. 2015.
[43] Y. Chen, Z. Jia, X. Ai, D. Yang, and J. Yu, "A modified two-part wolf pack search algorithm for the multiple traveling salesmen problem," *Appl. Soft Comput.*, vol. 61, pp. 714–725, Dec. 2017.
[44] E. Ruiz, V. Soto-Mendoza, A. E. Ruiz Barbosa, and R. Reyes, "Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm," *Comput. Ind. Eng.*, vol. 133, pp. 207–219, Jul. 2019.

HONGGUANG ZHANG received the B.S., M.S., and Ph.D. degrees from the Harbin Institute of Technology. From 2006 to 2010, he was a Test Engineer with China Academy of Space Technology. He was an Associate Professor and a Ph.D. Supervisor with the Beijing University of Posts and Telecommunications, where he currently works with the School of Electronic Engineering. He won the First Prize Award of science and technology of electronic information in the Chinese Institute of Electronics. His research interests include TSP, fire rescue optimization, real optimization algorithm, online optimization systems, and intelligent communication.

ZAN WANG is currently pursuing the M.S. degree with the Beijing University of Posts and Telecommunications. His research interests include intelligent logistics dispatching systems and task assignment.

MENGZHEN TANG is currently pursuing the M.S. degree with the Beijing University of Posts and Telecommunications. Her research interest includes intelligent bus dispatching systems.

XIUSHA LV is currently pursuing the M.S. degree with the Beijing University of Posts and Telecommunications. Her research interest includes intelligent road traffic management.

HAN LUO is currently pursuing the M.S. degree with the Beijing University of Posts and Telecommunications. His research interest includes task assignment for mobile robot swarms.

YUANAN LIU (Member, IEEE) is currently a Professor with the Beijing University of Posts and Telecommunications. His research interests include 5G systems and electromagnetic interference smart antennas.