A Hybrid Heuristic Based on a Particle Swarm Algorithm to Solve the Capacitated Location-Routing Problem With Fuzzy Demands

HUIZHEN ZHANG1, FAN LIU1, LIANG MA1, AND ZIYING ZHANG2

1School of Management, University of Shanghai for Science and Technology, Shanghai 200093, China
2School of Materials Engineering, Shanghai University of Engineering Science, Shanghai 201620, China

Corresponding author: Huizhen Zhang (zhzzywz@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 71401106, in part by the Shanghai Natural Science Foundation under Grant 14ZR1418700, in part by the Humanities and Social Sciences Foundation from the Ministry of Education of China under Grant 16YJA630037 and Grant 19YJAZH064, and in part by the Hujiang Foundation of China under Grant A14006.

ABSTRACT In this paper, the capacitated location-routing problem with fuzzy demands (CLRP-FD) is considered, which simultaneously solves two problems: locating the facilities and designing the vehicle routes among the established facilities and customers. In the CLRP-FD, the capacities of the employed vehicles and established facilities cannot exceed, and the demands of the customers are assumed to be triangular fuzzy variables. To model the CLRP-FD, a fuzzy chance constrained programming approach is designed using fuzzy credibility theory. To solve this problem, a hybrid particle swarm optimization (HPSO) algorithm, which includes a stochastic simulation and a local search strategy based on the variable neighborhood search algorithm, is proposed. Finally, the influence of the value of the dispatcher preference index (DPI) on the total distribution cost is analyzed by conducting numerical experiments. To evaluate the efficiency of the proposed HPSO and the performance of the CLRP-FD model, the results obtained using the HPSO were compared with their corresponding lower bound provided using the CPLEX solver. Moreover, we also evaluated the performance of HPSO through computational experiments on some well-known benchmark CLRP instances. The numerical results show that the proposed HPSO is competitive, and can give a satisfactory solution in a reasonable amount of time.

INDEX TERMS Capacitated location-routing problem, credibility theory, fuzzy demand, particle swarm optimization, stochastic simulation.

I. INTRODUCTION

In modern logistics decisions, facility locations and vehicle routing designs are vitally important to supply chain management because the dependable, efficient, and flexible decisions on the depots and the delivery routes can not only save distribution cost and time but also enhance the company’s ability to compete. Thus, a large number of researchers have been devoted to the facility location problem (FLP) [1] and vehicle routing problem (VRP) [2]–[4]. FLP is an example of a strategic level decision in which a long-term decision is made to select the locations of the facilities. VRP is an example of a tactical level decision in which vehicle routes are designed according to the demands of the customers and other factors. However, many researchers have indicated that solving of the FLP without considering the VRP could be immoderate and lead to a suboptimal solution [5], [6], and vice versa. As a combination of FLP and VRP, the location routing problem (LRP) simultaneously includes the strategic and tactical levels to overcome this drawback [7] and is conducive to building a flexible and efficient logistics system.

Since both FLP and VRP are well-known NP-hard combinatorial problems, the LRP must necessarily be an NP-hard problem [8], [9]. Overall, when the LRP instance size increases, the computing time increases exponentially. Thus, exact methods, such as the Branch-and-Cut method [5], [9] and Branch-and-Bound algorithm, cannot optimally solve large-size LRP instances in a reasonable amount of computing time. On the other hand, the wide range of applications of the LRP, such as waste collection [9], [10], newspaper
delivery, bill delivery [11], blood shipments [12] and perishable food distribution [13], often leads to large-size instances. Under such conditions, many researchers focus their attention on heuristics. Heuristics have been applied to many fields. Approaches such as simulated annealing heuristic [16]–[18], particle swarm optimization [19], genetic algorithm [20], ant colony algorithm [21], and hybrid artificial intelligence and robust optimization [22] have been used to solve the LRP and other supply chain management problems [14], [15], and they can provide suboptimal solutions.

The capacitated location-routing problem (CLRP), an important variant of the standard LRP, is constrained by the capacities of the vehicles and depots. CLRP is much closer to reality than the standard LRP and has been attracting more and more attention worldwide from researchers. The aim of the CLRP is to optimally determine the locations of the depots, assign customers to each established depot as well as detect a set of distribution routes for a set of identical vehicles in such a way that all customers with known demands can be serviced by one vehicle exactly once. Moreover, every vehicle should start and end its route at the same given depot. Due to the difficulty of the CLRP, exact methods are usually not good at solving intermediate and large instances. Therefore, heuristic and meta-heuristic approaches, which can find approximate solutions in a reasonable amount of computing time, are suitable methods for solving intermediate and large CLRP instances [3]. In addition to the basic characteristics of each solution method, the corresponding experimental results are also illustrated and studied in the literature. Nagy and Salhi surveyed the meta-heuristics and categorized them into four main groups, namely, sequential, clustering, iterative and hierarchical methods [3], [23]. Interested researchers can refer to the review paper [23] for more details about the mathematical formulations and solution methods for the LRP.

The CLRP requires that the demand of each customer should be known clearly before distribution. However, in most of the real-world situations, the demands of the customers can be vague or uncertain. These uncertain customers’ demands are considered in the capacitated location routing problem with fuzzy demands (CLRP-FD) [24]. In CLRP-FD, the demand of each customer is assumed to be imprecise, but according to experience, it can be approximately a given amount or within a given range, for example, approximately 40 units, or between 30 and 80 units. For these cases, fuzzy logic can be worthwhile, and a fuzzy chance constrained programming model can be formulated based on fuzzy credibility theory. The CLRP-FD was first modeled and solved by Mehrjerdi and Nadizadeh in 2013 [24]. In [24], Mehrjerdi and Nadizadeh presented a fuzzy chance constrained programming approach with a credibility measurement to model the CLRP-FD, which was then solved by a greedy clustering method with a stochastic simulation. Furthermore, Nadizadeh and Nasab considered a dynamic CLRP-FD, in which the facility location problem and vehicle routing problem are solved on a time horizon [21]. To solve the dynamic CLRP-FD formulation, they developed a hybrid heuristic algorithm that had four phases, including the stochastic simulation and a local search method [21]. In addition, Nadizadeh and Kafash also studied the fuzzy capacitated location-routing problem with simultaneous pickup and delivery (FCLRP-SPD), in which both pickup and delivery demands of customers were considered as fuzzy variables, and developed a greedy clustering method with four phases to solve this problem [25]. Without considering the capacities of the depots, Ghaffari-Nasab et al. considered an LRP with fuzzy demands and proposed a fuzzy chance-constrained programming model based on fuzzy credibility theory [17]. They used a hybrid simulated annealing-based heuristic to solve this problem, in which stochastic simulation was employed to assess the believability of a solution. Zarandi et al. proposed a CLRP with time windows under uncertainty, in which the demands of the customers and the travel time were assumed to be fuzzy variables [26]. They also developed a simulation-embedded simulated annealing algorithm [26] to solve this problem. Fazayeli et al. developed a mixed-integer nonlinear mathematical fuzzy model for the LRP in a multi-modal transportation network with time windows and fuzzy demands, in which the capacities of the depots were not considered [27]. Moreover, they proposed a genetic algorithm with two parts and a new chromosomes structure [27] to solve this model.

To the authors’ knowledge, the CLRP-FD has significantly fewer investigations, and there is still plentiful space for the improvement of the corresponding model and solution methods. In this paper, the CLRP-FD is also considered, which is formulated using the approach of fuzzy chance constrained programming with credibility measurement. In this problem, the actual demand of a customer can be known only when the vehicle arrives at the customer [24], [28]. Thus, an additional distance arises when the vehicles implement preplanned distribution routes, which are designed using a given solution approach. Owing to the uncertain demands of the customers, a customer might not be served by a vehicle that arrives there and does not have sufficient capacity. In such a situation, the vehicle is assumed to return to the depot for loading, then go back to the customer and resume its service along the remainder of the preplanned route [24]. This strategy results in an additional distance, which also greatly impacts the distribution cost and efficiency. Thus, the additional distance is also minimized in this work when minimizing the locating and routing costs.

To solve the proposed CLRP-FD model, a hybrid particle swarm optimization (HPSO) algorithm with three phases, including a local search method and stochastic simulation, is presented. The motivation of using HPSO in this work is threefold. First, the PSO, a swarm-based meta-heuristic algorithm, has provided good solutions for many combinatorial optimization problems (e.g., multidimensional knapsack problem [29], data allocation problem [30] and personnel assignment problem [31]) within a reasonable amount of computing time, especially many different variants of the FLP, LRP and VRP have been solved...
using PSO-based approaches in the literature (e.g. VRP with stochastic demands [32], VRP with dynamic requests [2], multi-workshop facility layout problem [33] and LRP with stochastic demands [19]). Second, a meta-heuristic is a general algorithmic framework for finding solutions to optimization problems within which local heuristics can be guided and adapted to effectively explore the solution space [3]. Third, to obtain better results, there should still be some opportunities to improve the PSO by introducing some heuristic techniques into it. The authors are intensively motivated by these factors to use the hybrid PSO to solve the proposed CLRP-FD model.

This paper makes the following contributions. First, a chance-constrained programming model of CLRP with fuzzy demands is proposed. Second, a hybrid particle swarm optimization algorithm with a stochastic simulation is presented for solving the CLRP-FD, in which an effective initialization method based on the greedy cluster algorithm and a local search method based on the path-relinking algorithm and variable neighborhood search algorithm [34] are employed to provide good initial solutions to the algorithm and strengthen the exploitation in the promising solution space, respectively. Finally, some experiments are conducted to evaluate the operational gains obtained using the proposed CLRP-FD model and to analyze the performance of the proposed HPSO algorithm. As shown later, the proposed algorithm can effectively produce good solutions to the CLRP-FD.

The rest of this paper is organized as follows: Fuzzy credibility theory and some assumptions are introduced in the section II. The details of CLRP model with fuzzy demands are stated in the section III. The proposed HPSO algorithm to solve the CLRP-FD model is discussed in the section IV. Then, computational results and analysis are given in the section V. Finally, conclusions and some future researches are presented in the section VI.

II. Fuzzy Credibility Theory

The concept of a fuzzy set was first proposed by Zadeh [35]. To measure a fuzzy event, Kaufmann [36] and Zadeh [37] successively proposed the term fuzzy variable and the corresponding possibility measure theory, respectively. Although possibility measure theory has been widely used, it does not obey the law of truth conservation and is inconsistent with the laws of excluded middle and contradiction [38]. Because a self-dual measure is very necessary and important to the possibility measure, Liu [39] proposed the concept of a credibility measure for defining self-duality. In this study, the CLRP with fuzzy demand is formulated based on a credibility measure. The related concepts and definitions used in this paper are briefly introduced as follows:

Let \( \Theta \) and \( \phi \) be non-empty and empty set, respectively. \( P \) is the power set of \( \Theta \) (\( P(\Theta) \)), and each element in \( P \), denoted as \( A \), is called an event. \( \text{Pos}[A] \) is the occurring possibility of event \( A \). The following four axioms are the basic of credibility theory, and all the concepts of credibility theory can be obtained from them [39].

**Axiom 1:** \( \text{Pos}[\emptyset] = 1 \).

**Axiom 2:** \( \text{Pos}[\phi] = 0 \).

**Axiom 3:** For each \( A_i \in P(\Theta) \),

\[
\text{Pos}\left(\bigcup_{i} A_i\right) = \sup \text{Pos}[A_i].
\]

**Axiom 4:** If \( \Theta_i \) is a non-empty set, and the set function \( \text{Pos}_i[A_i](i = 1, 2, \cdots, n) \) satisfies above three axioms, and \( \Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_n \), then for each \( A \in P(\Theta) \),

\[
\text{Pos}(A) = \sup_{(\Theta_1 \times \Theta_2 \times \cdots \times \Theta_n) \in A} \text{Pos}_1[\Theta_1] \land \cdots \land \text{Pos}_n[\Theta_n].
\]

Note that \( \text{Pos} = \text{Pos}_1 \land \text{Pos}_2 \land \cdots \land \text{Pos}_n \).

If \( \text{Pos} \) satisfies the first three axioms, it is named possibility measure. The triple set \( (\Theta, P(\Theta), \text{Pos}) \) is a possibility space. With the axiom 4, the basic of credibility theory was built.

**Definition 1:** Let \( (\Theta, P(\Theta), \text{Pos}) \) be a possibility space, and \( A \) is a set in \( P(\Theta) \), then the necessity measure and credibility measure of \( A \) are defined by \( \text{Nec}[A] = 1 - \text{Pos}[A^c] \) (\( A^c \) is the complement of set \( A \)) and \( \text{Cr}[A] = \frac{1}{2}(\text{Pos}(A) + \text{Nec}(A)) \), respectively.

Definition 1 shows that the credibility of a fuzzy event is the average of its possibility and necessity [24]. The credibility measure is self-dual [24]. A fuzzy event may fail even though its possibility is equal to 1, and occur even though its necessity is 0 [24]. However, the fuzzy event must occur if its credibility is equal to 1, and fail if its credibility is 0 [39].

Now, a triangular fuzzy variable \( \tilde{d} = (d_1, d_2, d_3) \) is considered as the demand of a customer where \( \tilde{d} \) is described by its left boundary \( d_1 \) and its right boundary \( d_3 \). The dispatchers can subjectively estimate the value of \( d_1 \) and \( d_3 \) based on their experience and available information, and the actual demand of each customer will be between \( d_1 \) and \( d_3 \). The value of \( d_2 \) corresponding to a grade of membership of 1 can also be subjectively estimated. The membership function is shown as Figure 1. Let \( r \) be the actual demand of a customer, the possibility \( \text{Pos}(\tilde{d} \geq r) \), necessity \( \text{Nec}(\tilde{d} \geq r) \) and credibility \( \text{Cr}(\tilde{d} \geq r) \) are easy to obtain as follows [40]:

\[
\text{Pos}(\tilde{d} \geq r) = \begin{cases} 
1, & \text{if } r \leq d_2 \\
\frac{d_3 - r}{d_3 - d_2}, & \text{if } d_2 \leq r \leq d_3 \\
0, & \text{if } r \geq d_3 
\end{cases}
\]

(1)

\[
\text{Nec}(\tilde{d} \geq r) = \begin{cases} 
1, & \text{if } r \leq d_1 \\
\frac{d_2 - r}{d_2 - d_1}, & \text{if } d_1 \leq r \leq d_2 \\
0, & \text{if } r \geq d_2 
\end{cases}
\]

(2)

\[
\text{Cr}(\tilde{d} \geq r) = \begin{cases} 
1, & \text{if } r \leq d_1 \\
\frac{2d_2 - d_1 - r}{2(d_2 - d_1)}, & \text{if } d_1 \leq r \leq d_2 \\
\frac{d_3 - r}{2(d_3 - d_2)}, & \text{if } d_2 \leq r \leq d_3 \\
0, & \text{if } r \geq d_3 
\end{cases}
\]

(3)
Let $P$ be the capacity of a depot. Similarly, after allocating $h$ customers to a depot, its remaining capacity equals $P_h = P - \sum_{i=1}^{h} d_i = (P - \sum_{i=1}^{h} d_3i, P - \sum_{i=1}^{h} d_2i, P - \sum_{i=1}^{h} d_1i) = (p_1h, p_2h, p_3h)$, which is also a triangular fuzzy number. Then, the credibility that the demand of the $h + 1$st customer ($d_{h+1}$) is not greater than $P_h$ can be given as follows:

$$Cr = Cr(d_{h+1} \leq P_h)$$

$$= Cr([d_{1,h+1} - p_{3,h}, d_{2,h+1} - p_{2,h}, d_{3,h+1} - p_{1,h}] \leq 0)$$

$$= \begin{cases} 
0, & \text{if } d_{1,h+1} \geq p_{3,h} \\
2*(p_{3,h} - d_{1,h+1} + d_{2,h+1} - p_{2,h}), & \text{if } d_{1,h+1} \leq p_{3,h}, d_{2,h+1} \geq p_{2,h} \\
2*(p_{2,h} - d_{2,h+1} + d_{3,h+1} - p_{1,h}), & \text{if } d_{2,h+1} \leq p_{2,h}, d_{3,h+1} \geq p_{1,h} \\
1, & \text{if } d_{3,h+1} \leq p_{1,h} 
\end{cases}$$

(6)

Obviously, the possibility of serving the next customer is greater if the remaining capacity of the vehicle is high and the demand of the next customer is low [21]. The value of $Cr$ ($Cr \in [0, 1]$), calculated using formula (5), shows the level of certainty about the ability of a vehicle to serve a new customer [26]. When $Cr = 0$, the vehicle cannot serve the next customer and should return to the depot [26]. When $Cr = 1$, the dispatcher will definitely make sure that the remaining vehicle capacity can serve the next customer. In this study, a threshold is defined as $API$, where $API \in [0, 1]$ is the dispatcher preference index, which can affect the solution considerably [26]. We decide whether the vehicle serves the next customer by comparing the value of $API$ with the credibility obtained using the equation (5). The vehicle should serve the $h + 1$th customer if $Cr \geq API$; otherwise, it should return to the depot to load sufficient goods and, then, return to the customer. The process does not terminate until all of the customers are serviced.

Similarly, the value of $Cr$ ($Cr \in [0, 1]$), calculated using formula (7), shows the level of certainty with which a depot is able to serve a new customer [26]. If the remaining capacity of the depot is high and the demand of the next customer is low [21], the possibility of allocating the new customer to the depot is greater. When $Cr = 1$, the manager of the depot is absolutely certain that the remaining capacity of the depot can provide service to the next customer. When $Cr = 0$, the manager of the depot will definitely ensure that the next customer cannot be allocated to the depot due to insufficient available capacity. In this study, another threshold is defined as $API$, where $API \in [0, 1]$ is the manager preference index, which can also affect the solution considerably. The manager decides whether the depot serves the next customer by comparing the value of $Cr$ with the value of $API$. The $h + 1$th customer will be allocated to the current depot if $Cr \geq API$; otherwise, he/she will be assigned to another
open depot. This process does not terminate until all of the customers are allocated [21].

In this paper, an additional distance is considered, which is produced by the “failure” of the planned route and has non-neglected impact on the routing cost. As stated above, the parameters DPI and API can considerably affect the length of the planned routes and the additional distance. For example, a lower value of DPI, which shows that the dispatcher wants to use the the capacity of the vehicle the best he can, increases the possibility of failure that a vehicle arrives at a customer and is unable to provide service for it. Thus, the lower value of DPI results in shorter planned routes and longer additional distances. In contrast, a higher value of DPI shows that the dispatcher desires to serve every customer along the planned route. Hence, a higher value of the DPI results in higher planned routes and lower additional distances. In addition, the higher value of DPI requires more vehicles, and it would lose some of customers because there are not enough vehicles to serve all the customers. Therefore, the cost of lost opportunity due to lack of service to customers is also considered in this paper. As a result, the sensitive parameters DPI and API significantly influence the sum lengths of the planned routes and additional distance and should be determined properly [24].

In this work, the additional distance caused by service failures is estimated using a stochastic simulation.

The CLRP-FD can be modeled as a directed complete graph \( G(V, E) \), with a vertex set \( V \) (Set of all candidate depots and customers) and an edge set \( E = \{(i, j) | i, j \in V \ and \ i \neq j\} \). Before presenting the mathematical formulation of the problem, the following notations are defined.

**Set and parameters:**

- \( J \): Set of customers.
- \( I \): Set of candidate depots.
- \( V \): Set of all candidate depots and customers, \( V = I \cup J \).
- \( K \): Set of vehicles.
- \( d_i \): demand of customer \( i \).
- \( Q \): Capacity of each vehicle. It is assumed that all of the vehicles are homogeneous.
- \( P_i \): Capacity of depot \( i \).
- \( F_k \): Fixed cost of using vehicle \( k \).
- \( O_i \): Fixed cost of opening depot \( i \).
- \( C_{ij} \): Cost per unit distance from vertex \( i \) to vertex \( j \), \((i,j) \in E \).
- \( A_k \): Maximum driving distance of vehicle \( k \).
- \( D_{ij} \): Distance from vertex \( i \) to vertex \( j \).
- \( B_i \): Cost of the lost opportunity due to the lack of providing service to customer \( j \).

**Decision Variables:**

- \( x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ goes directly from node } i \text{ to node } j \\ 0, & \text{otherwise} \end{cases} \)
- \( w_{ijk} \): The sequence number of customer \( j \) in route \( k \), which are auxiliary variables for eliminating sub-tour in route \( k \) [21].
- \( AD_k \): Additional distance traveled by vehicle \( k \).

Given the above defined assumptions, parameters, and decision variables, the corresponding model of the CLRP-FD based upon the credibility theory is formulated as follows:

\[
\begin{align*}
\text{min} & \sum_{i \in I} O_i z_i + \sum_{i \in I} \sum_{J \in J} \sum_{k \in K} F_k x_{ijk} \\
&+ \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} C_{ij} D_{ij} x_{ijk} \\
&+ \sum_{j \in V} B_i (1 - \sum_{i \in V} \sum_{k \in K} x_{ijk}) \\
\text{Subject to:} & \sum_{i \in V} \sum_{j \in J} d_{ij} x_{ijk} \leq Q \ \forall k \in K \quad (10) \\
& \sum_{i \in V} \sum_{j \in I} x_{ijk} \leq 1 \ \forall j \in J \quad (12) \\
& \sum_{j \in V} \sum_{k \in K} x_{ijk} \leq 1 \ \forall k \in K \quad (13) \\
& \sum_{i \in I} \sum_{j \in V} x_{ijk} - \sum_{i \in V} x_{ijk} = 0 \ \forall j \in V, \forall k \in K \quad (15) \\
& \sum_{i \in V} \sum_{j \in I} x_{ijk} + \sum_{(i,j) \in E \setminus I \cup J} x_{ijk} \leq 1 + y_j \quad (16) \\
& \sum_{i \in I} \sum_{j \in J} x_{ijk} - z_i \leq 0 \ \forall i \in I, \forall k \in K \quad (19) \\
& z_i - \sum_{j \in J} \sum_{k \in K} x_{ijk} \leq 0 \ \forall i \in I \quad (20) \\
& x_{ijk} \in \{0, 1\} \ \forall i \in V, \forall j \in V, \forall k \in K \quad (21) \\
& y_{ij} \in \{0, 1\} \ \forall i \in I, \forall j \in J \quad (22) \\
& z_i \in \{0, 1\} \ \forall i \in I \quad (23) \\
& w_{ijk} \in \{0, 1, \cdots, N\} \ \forall j \in J, \forall k \in K \quad (24) \\
\end{align*}
\]
\[ AD_k \geq 0 \quad \forall k \in K \] (25)

The objective function (8) is to minimize the total costs which consists of sum of the fixed costs of opening depots, the fixed costs of using vehicles, the travel costs, the costs of lost opportunities, respectively [21]. The objective function (9) is to minimize the total additional distances. \( AD_k \) is only related to the designed route of vehicle \( k \), and the objective function (9) is calculated by the simulation procedure presented in subsection IV-D. Constraints (10) and (11) assure that all of the customers are visited within vehicle capacity and are allocated within depot capacity with a confidence level, respectively [21]. Constraints (12) ensure that each customer is served by one vehicle at most and that each customer should have one predecessor at most. Constraints (13) guarantee that each vehicle can be used only once at most, where \( N \) is the number of customers served by vehicle \( k \). Constraints (14) state the elimination of the sub-tour for each vehicle route. Constraints (15) ensure that every vehicle will exit the customer after visiting it, and that each vehicle starts from and ends at a single depot. Constraints (16) represent that a customer must be allocated to a depot if they are connected by a route. Constraints (17) ensure the limitation on the driving distance for each employed vehicle. Constraints (18) illustrate that each customer must be allocated to exact one depot. Constraints (19) and (20) show two decision variables’ relation. Constraints (21)-(23) and (24) specify the binary variables and auxiliary variables, respectively. Constraints (25) specify the continuous variables.

IV. THE HPSO FOR THE CLRP-FD

Based on the PSO and stochastic simulation, the hybrid heuristic algorithm for the CLRP-FD is proposed in this section. In a PSO-based algorithm, initially, a group of particles are initialized and every particle represents a possible solution. In the solution space, every particle moves with a given velocity from the current position to the next possible position. A successful designation of a PSO-based algorithm for the CLRP-FD depends mainly on three key issues, which are (1) creating the initial set of particles; (2) designing a desirable mapping between the particles in the PSO and the CLRP-FD solutions; and (3) moving every particle with a given velocity. In this study, a greedy cluster algorithm is used to design an efficient initialization approach and enhance the efficiency of the proposed algorithm. In addition, a path-relinking algorithm and variable neighborhood search (VNS) algorithm are implemented to update the positions of the particles, which not only effectively avoids transforming the solutions between continuous space and discrete space [32], but also strengthens the exploitation on the promising search space.

The basic steps of the HPSO are addressed in the following subsections. Its pseudocode is depicted in Algorithm 1. It takes as parameters the maximum number of iterations, \( G \), the number of the particles, \( H \), and the range of the velocity, \([V_{\text{min}}, V_{\text{max}}]\). After initializing the population in the first step (line 1), the fitness of each individual that is included in the initial population is evaluated (line 2), and the best individual in the current population is also found (line 3). The best solution found for the CLRP-FD, \( g_{\text{best}} \), and the velocity of each particle, \( v_i \), are initialized in line 4 and lines 5-7, respectively. The iterations of the HPSO are executed in lines 8-27. Each HPSO iteration includes the calculation of the average velocity of each particle (line 10), the updating of the thresholds \( (R_1 \text{ and } R_2) \) of the average velocity of particle \( p_i \) (line 10), a position updating strategy (lines 11-19) based on the mixed path-relinking algorithm (see subsection IV-C1 for the details) and the variable neighborhood search (see subsection IV-C2 for the details), resetting the best individual (lines 20-21) and saving the best so far solution (lines 22-23). During the optimization processes, the mixed path-relinking algorithm and the variable neighborhood search strengthen the exploitation in the neighborhood of each particle.

A. SOLUTION REPRESENTATION

A solution representation should determine the assigned customers to each vehicle, the depots to be established and the sequence of customers to be served by a specific vehicle starting from and ending at a depot [26]. Given \( m \) candidate depots and \( n \) customers, a solution representation is presented using all of the elements within the sets \( I = \{1, 2, \cdots , m\} \) and \( J = \{m+1, m+2, \cdots , m+n\} \), where the \( m \) depots and \( n \) customers are numbered with the elements in sets \( I \) and \( J \), respectively. The first number in a solution is always in the set \( I \), which indicates the first depot under consideration [17]. In the solution representation, the customers assigned to each depot are between it and the next different depot. If there are no customers after a depot, it indicates that this candidate depot is not open. Each route of a given depot starts by serving the first customer, and the other customers allocated to that depot are assigned to the current route from left to right [17], [41]. If the credibility of having sufficient capacity for serving the next customer falls below the dispatcher preference index [17], the current route is terminated, and the next number in the solution representation will be a depot if there are unsigned customers. If the next number in the solution representation is still this depot, a new route will start from it to service the remaining customers allocated to it; otherwise, a new route will start from a new different depot. It can be verified that this solution representation always gives a CLRP-FD solution without violating the capacity constraints of the vehicles and depots [17], [41].

To further clarify the above description, a feasible solution of a CLRP-FD instance with \( m = 5 \) and \( n = 20 \) is shown in Figure 2. It can be seen that the candidate depot, depot 1, is not open, and the 20 customers are assigned to 4 open depots (2, 3, 4 and 5). The delivery routes related to the open depots are shown in Figure 3.

B. INITIAL SOLUTION GENERATION

Initialization of a solution plays a significant role for the performance of a meta-heuristic because the quality of the
To establish the depots among the candidate sites, the gravity of the vehicle can be fully utilized using this procedure. Until all of the customers are assigned. Obviously, the capacitated customers will be assigned to the new cluster for another different customer, which is closest to the last selected customer of the current cluster is selected. Otherwise, it will be excluded from the cluster, and the greedy search method will search for another different customer, which is closest to the last added member of the cluster among the non-clustered customers. If the remaining capacity of the vehicle cannot serve a new customer, a new cluster will be launched, and the non-clustered customers will be assigned to the new cluster until all of the customers are assigned. Obviously, the capacity of the vehicle can be fully utilized using this procedure.

**Step 1 (Clustering the Customers):** Based on the consideration of the fuzzy demands and intra-distances of the customers and the capacities of the vehicles, the customers are clustered using a greedy search method in this step. First, to generate a cluster, a customer from the non-clustered customers is randomly selected. Then, the nearest customer to the last selected customer of the current cluster is selected. Considering the DPI value and the credibility of the customer, if the closest customer has not been clustered and its fuzzy demand satisfies the remaining capacity of the vehicle, it will be assigned to the cluster. Otherwise, it will be excluded from the cluster, and the greedy search method will search for another different customer, which is closest to the last added member of the cluster among the non-clustered customers. If the remaining capacity of the vehicle cannot serve a new customer, a new cluster will be launched, and the non-clustered customers will be assigned to the new cluster until all of the customers are assigned. Obviously, the capacity of the vehicle can be fully utilized using this procedure.

**Step 2 (Calculation of the Gravity Center of Each Cluster):** To assign the clusters generated in the first step to the depots, the gravity center of each cluster is calculated using equation (26),

\[
(X_c, Y_c) = \left(\frac{\sum_{j \in C} x_j}{n_c}, \frac{\sum_{j \in C} y_j}{n_c}\right)
\]  

where \( n_c \) is equal to the number of customers assigned to cluster \( C \), and \((X_c, Y_c)\) and \((x_j, y_j)\) represent the coordinates of the gravity center of cluster \( C \) and customer \( j \), respectively.

Second, the sum of the Euclidean distances between \( n \) customers and each candidate depot is calculated using equation (27). In (27), \( U_i \) is the sum of the distances from depot \( i \) to all of the customers, and \((x_i, y_i)\) and \((x_j, y_j)\) represent the coordinates of depot \( i \) and customer \( j \), respectively. Since most of the state-of-the-art heuristic algorithms for solving CLRP instances in the existing literature used Euclidean distance, this study also employed Euclidean distance to conveniently compare the computational results obtained by HPSO and other algorithms. If the actual traffic distance is used, it will be closer to the realistic situation, but the type of distance and calculation method cannot affect the optimization performance of the algorithm.

\[
U_i = \sum_{j=1}^{n} \left[ (x_i - x_j)^2 + (y_i - y_j)^2 \right]^{\frac{1}{2}},
\]  

where \( j = 1, 2, \cdots, m \)

Third, the candidate depots are sorted in descending order according to the amount of factor \( W_i \), which is calculated by equation (28).

\[
W_i = \frac{P_i}{O_i \times U_i}
\]  

**Step 3 (Allocating the Customers to the Depots):** To assign the clusters to the ranked depots, the Euclidean distance between the gravity center of each cluster and the top-ranked depot is calculated first in this step. Afterward, the unassigned clusters are ranked in ascending order according to the distance between their gravity centers and the top-ranked depot.
The top-ranked cluster is assigned to the top-ranked depot if $Cr \geq API$. The clusters in the sorted list should be allocated one by one to the top-ranked depot considering the above relation. The process of allocating clusters to a depot will be terminated when the depot does not have enough capacity to serve a new cluster. In this situation, the allocating procedure will be repeated for the next-ranked depot until all of the clusters are allocated [21].

**Step 4 (Constructing the Routes):** In this step, the routing problem is solved using the greedy search method for each cluster and its relevant depot. First, the nearest customer to each depot selected in step 3 is routed. Then, a process similar to that employed in step 1 is used to select the closest customer to the last added customer of the current route. If the remaining capacity of the current vehicle is not sufficient to serve a new customer, a new route will be added, and the non-routed customers will be assigned to the new route until all of the customer allocated to the given depots are routed.

### C. Position Updating

In the PSO, the position of each individual particle is adjusted according to its velocity, which is updated iteratively through its personal best position (i.e., the best position found so far by the particle) and the best position found by the particles in its neighborhoods. As designed before, the position of each individual particle in this study is represented by a $d$-dimensional vector in problem space $x_i = (x_{i1}, x_{i2}, \cdots, x_{id})$, $i = 1, 2, \cdots, H$ ($H$ is the population size, and $d$ is the dimension of the vector) [32]. In a classic PSO, the new position and velocity of each particle can be calculated by Equations (29) and (30) [42], respectively.

$$
x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{29}
$$

$$
v_{ij}(t+1) = \chi(v_{ij}(t)) + c_1r_1(p_{best_i} - x_{ij}(t)) + c_2r_2(g_{best_j} - x_{ij}(t)) \tag{30}
$$

where

$$
\chi = \frac{2}{|2 - c - \sqrt{c^2 - 4c}|} \quad \text{and} \quad c = c_1 + c_2, \quad c > 4 \tag{31}
$$

where $t$ is the loop counter, $c_1$ and $c_2$ are the acceleration coefficients, and $r_1$ and $r_2$ are random values generated in the interval (0,1). If Equations (29) and (30) are employed in this study, the solutions of the proposed model (8)-(25) should be transformed to continuous space. To avoid this transformation, Equation (29) is not used at all. As mentioned before, in this study, the position of each particle is updated using a Path Relinking Algorithm [19], which is an intensification strategy and is usually used to explore trajectories between solutions. Therefore, the Path Relinking algorithm follows the idea of the movement of an individual particle in a PSO; that is, when the path relinking algorithm is used to adjust the position of a particle, the particle can either move along its own path or move toward its previous optimal solution, or move toward the global optimal solution, where the roles of the starting solution and target solution are played by the particle (current solution) and the global best particle of the swarm or the current best solution of the particle.

In this study, the average velocity of each particle is used to decide which target solution it follows. The average velocity of each particle, $Average_{Vi} (i = 1, 2, \cdots, H)$, is calculated using Equation (32). If $Average_{Vi} < R_1$ ($R_1$ is a real number), then path relinking will not perform, and the particle will follow its own path, which is implemented using a variable neighborhood search algorithm (which will be described later); otherwise, the particle will perform a path relinking with its current best solution if $R_1 \leq Average_{Vi} < R_2$ ($R_2$ is also a real number) or with the global best particle if $Average_{Vi} \geq R_2$. The two real numbers $R_1$ and $R_2$ are

### Algorithm 1 Pseudocode of the Hybrid Particle Swarm Optimization (HPSO)

**Input:** $G = $ number of iterations, $H = $ size of the population, $V_{\text{max}} = $ maximum velocity of each particle, $V_{\text{min}} = $ minimum velocity of each particle, 

**Output:** $gbest = $ Best solution found for the CLRP-FD.

1. Initialize a population $\mathcal{H}$ by the proposed greedy cluster algorithm;
2. Evaluate each individual $p_{i} \in \mathcal{H} (i = 1, 2, \cdots, H)$;
3. $p_{best} := $ the best particle in the current population;
4. $g_{best} := p_{best}$;
5. for $i := 1$ to $H$ do
6. \quad $v_{i} := \text{rand}(V_{\text{min}}, V_{\text{max}})$;
7. end for
8. for $g := 1$ to $G$ do
9. \quad for $i := 1$ to $H$ do
10. \quad \quad Calculate $Average_{Vi}$, $R_1$ and $R_2$. $Average_{Vi}$ is the average velocity of particle $p_{i}$. $R_1$ and $R_2$ are the thresholds of average velocity of particle $p_{i}$.
11. \quad \quad if $R_1 \leq Average_{Vi}$ & $Average_{Vi} < R_2$ then
12. \quad \quad \quad $p_{i} := \text{Path Relinking}(p_{i}, p_{best})$;
13. \quad \quad else
14. \quad \quad \quad if $Average_{Vi} \geq R_2$ then
15. \quad \quad \quad \quad $p_{i} := \text{Path Relinking}(p_{i}, g_{best})$;
16. \quad \quad \quad else
17. \quad \quad \quad \quad $p_{i} := \text{BVNS}(p_{i})$;
18. \quad \quad end if
19. \quad \quad end if
20. \quad \quad if $f(p_{i}) < f(p_{best})$ then
21. \quad \quad \quad $p_{best} := p_{i}$;
22. \quad \quad \quad if $f(p_{best}) < f(g_{best})$ then
23. \quad \quad \quad \quad $g_{best} := p_{i}$;
24. \quad \quad end if
25. \quad \quad end if
26. \quad end for
27. \quad end for

return $g_{best}$
calculated using the following Equations (33) and (34),

\[
\text{Average}_V_i = \frac{\sum_{j=1}^{d} V_i(t + 1)}{d}
\]

\[
R_1 = (V_{\text{max}} - V_{\text{min}}) \times (w_1 - \frac{w_1}{G} \times t) + V_{\text{min}}
\]

\[
R_2 = (V_{\text{max}} - V_{\text{min}}) \times (w_2 - \frac{w_2}{2 \times G} \times t) + V_{\text{min}}
\]

where \( G \) is the maximum iterations, \( t \) is the current iteration, and \( V_{\text{max}} \) and \( V_{\text{min}} \) are the upper and lower bounds for the velocity of each particle. If the value of the velocity of a particle is not between these bounds in some iterations, a value between these bounds will be randomly selected to initialize it. With an increase in the number of iterations, we want to increase the possibility of performing path relinking and decrease the possibility of performing the variable neighborhood search, and therefore, \( R_1 \) and \( R_2 \) have large values at the beginning of the algorithm, and the values are reduced during the iterations.

1) PATH RELINKING ALGORITHM
Path-relinking is a search intensification strategy proposed by Glover [43], [44]. As a major enhancement to heuristic search methods for solving combinatorial optimization problems, its hybridization with other meta-heuristics has led to significant improvements in both the solution quality and running time [43]. Because the ideas behind the path-relinking algorithm and the movement of an individual particle are very similar [32], the path-relinking algorithm is employed in this paper to improve the current solution and make it close to the target solution. In the path relinking strategy, the current solution plays the role of the starting solution, and the best particle of the swarm or the current best solution of the particle plays the role of the target solution [32]. If the Hamming distance (the Hamming distance between two solutions is the number of positions at which the corresponding elements are different), \( HD \), between the starting solution and the target solution is not equal to 0, then the trajectories between the two solutions are explored by a cycling exchange operator until the value of \( HD \) is decreased to 0.

Algorithm 2 shows the pseudocode of the path-relinking algorithm employed in this study. First, the best solution produced by this optimization procedure is initialized in line 2, and the Hamming distance between the starting solution \( x \) and the target solution \( x' \) is calculated in line 3. Then, the trajectories between the two solutions are explored by a cycling exchange operator (lines 5-31) if \( HD > 0 \). If the variable \( Temp \) is equal to 1, the exchanging procedure is implemented from the left-hand side of the solutions (lines 6-14); otherwise, it is implemented from the right-hand side of the solutions (lines 15-24). The solution \( x' \) obtained by the exchange operator replaces the best solution \( x^* \) if its fitness is better than the one of \( x^* \) (lines 25-27). The starting solution \( x \) is also updated using the solution \( x' \) in line 28. To find solutions with better quality, the roles of the starting and target solutions are interchanged (line 29) at each iteration of the path-relinking algorithm. The Hamming distance is updated in line 30, and the procedure repeats until \( HD = 0 \). Finally, the best solution obtained by this optimization procedure is returned in line 32.

2) VARIABLE NEIGHBORHOOD SEARCH
As mentioned previously, if \( \text{Average}_V_i < R_1 \) for individual particle \( i \), the Variable Neighborhood Search (VNS)
Algorithm 3 Pseudocode of the Variable Neighborhood Search Algorithm

1: function VNS(x, tmax, qmax)
2:     for t := 1 to tmax do
3:         for q := 1 to qmax do
4:             x’ ← Shake(x, q); //Generating a point x’ at random from the kth neighborhood of x, i.e., x’ ∈ Nq(x)
5:             x'' ← FirstImprovement(x’, q); //Local search; Finding a point x” from the qth neighborhood of x’ where f(x’’) ≤ f(x’)
6:             if f(x’’) < f(x) then
7:                 x = x’’;
8:                 q = 1;
9:             else
10:                q = q + 1;
11:         end if
12:     end for
13: end for
14: return x;
15: end function

Algorithm 4 Pseudocode of the First Improvement Heuristic

1: function FIRSTIMPROVEMENT(x, q)
2:     i := 0; x’ := x;
3:     while i < Nq(x) and f(x) ≥ f(x’) do
4:         i ← i + 1;
5:         x ← arg min{f(x), f(x’)}, x’ ∈ Nq(x)
6:     end while
7:     return x;
8: end function

Neighborhood N1. We randomly select two elements, which are located in position i and j (j > i), e.g., 13 located in position 9, and 4 located in position 24. The element located in position i is moved to position j, and the other elements from i+1 to j are separately moved to the position previous to them. The new sequence obtained using this shifting operator is shown in Figure 4.

Neighborhoods N2 and N3. The approach of generating neighborhoods N2 and N3 is similar to the approach of generating N1. The difference is that the sequential elements from position i to i+l are selected and moved to the positions from j−l to j correspondingly (j > i and j ≠ [i, i + 1, · · · , i + l]), and the other elements from i + l + 1 to j are separately moved forward i + l positions. According to the numerical experiments, l is set to 2 and 3 in this study. Here, l = 2 and l = 3 correspond to neighborhoods N2 and N3, respectively. For example, if i = 9, j = 24 and l = 2, then the sequential elements 13, 24 and 23 are moved to the positions j − 2, j − 1 and j, respectively. The result for neighborhood N2 is shown in Figure 5.

Neighborhoods N4 and N5. Neighborhoods N4 and N5 are obtained by randomly exchanging one pair of customers and depots, respectively. The swapping between a customer and a depot is not allowed in N4 and N5. For example, new sequences are produced by exchanging customers 13 and 25 in Figure 6, and depots 2 and 5 in Figure 7.

Neighborhood N6. Neighborhood N6 is obtained using the inverse operator. We randomly select two points, e.g. 11 and 14. A new sequence shown in Figure 8 is obtained by reversing the subsequence between these two points [3].

Note that the solution obtained by conducting the local search in these above neighborhoods can be infeasible. In this case, a new solution must be encoded again to ensure its feasibility [45]. If the first position is selected, then the other position must be a depot to maintain feasibility [45].

D. STOCHASTIC SIMULATION

As stated previously, each customer’s demand is assumed to be a triangular fuzzy number, and it cannot be directly considered as a deterministic number [21]. To reveal the actual demand of each customer, a stochastic simulation is employed [21], [24]. For each feasible planned route, additional distances due to route failures are also provided using this simulation experiment. Its main steps are summarized as follows:

The variable neighborhood search algorithm shown in Algorithm 3 combines stochastic and deterministic changes in the neighborhoods. The random selection of a point x’ from the qth neighborhood (line 4) represents the stochastic phase. A local search based on the first improvement heuristic represents the deterministic phase. The first improvement heuristic starts at an initial solution x, and then, it makes a move as soon as a direction of decent within the neighborhood Nq(x) is found. This procedure is summarized in Algorithm 4.
Step 1: Generating the actual demand of each customer by using the following processes: (1) randomly generate a real number $r$ in the interval between the left and right bounds of the triangular fuzzy number representing demand of the customer, and compute its membership $\mu$ [21]; (2) randomly generate a real number $\alpha$ in interval $[0, 1]$, if $\alpha \leq \mu$, the actual demand of the customer is adopted as $r$. Otherwise, randomly generate $r$ and $\mu$ again until the relation $\alpha \leq \mu$ is fulfilled; (3) repeat (1) and (2) until every customers have an actual simulation demand [21], [24].

Step 2: Move along each distribution route generated by the proposed HPSO and calculate the additional...
and \( w \), the population size, \( n \), the number of candidate depots (\( m \)), the number of customers (\( n \)), optimal or best-known objective value (\( \text{BOV} \)), how many runs with which the best-known solution was found (\( \text{Found} \)), average percentage deviation from the best-known solution, \( \text{APD} = \frac{\text{BOV} - \text{BKOV}}{\text{BKOV}} \times 100\% \), average CPU time in seconds (\( \text{Time} \)), and average iterations (\( \text{Aiter} \)) in which the best-known solution was found.

From the results shown in Table 1, we can observe that all 5 tested instances obtained their optimal or near optimal solutions within 1500 iterations. Because large instances can require a large number of iterations, we set \( G = 2000 \) in this study. To obtain a high-quality solution without wasting a large amount of computing time, a stopping criterion was also used in the HPSO. If the best solution found thus far could not be improved after \( G/4 \) iterations, the search process of the HPSO was terminated.

### V. NUMERICAL EXPERIMENTS

In order to study the performance of the proposed HPSO for solving the CLRP-FD, a number of computational experiments were carried out. Matlab 2017a was used for the implementation of the algorithm. All the experiments were carried out on a personal laptop equipped with Inter core i5 4200M CPU at 2.50GHz and 8GB of RAM under Windows 10 professional system.

### A. SENSITIVITY ANALYSIS AND PARAMETER SETTINGS IN HPSO

The quality of the solution generated using HPSO can be affected by the parameters, namely, the number of iterations, \( G \), the population size, \( n \), acceleration coefficients, \( c_1 \) and \( c_2 \), and \( w_1 \) and \( w_2 \), which are used to control the searching method. For simplicity, the parameters in HPSO were fixed in the initialization stage and did not change during the running of the algorithm. This method is usually called parameter tuning in the literature. Because we would like to make the computational results as reproducible as possible, the runs in all of the experiments were limited to a single set of parameter settings generated using parameter tuning.

To illustrate how the parameters affect the overall performance of the HPSO, an analysis between the approximate optimal solution and the crucial parameters (\( H, c_1, c_2, w_1 \) and \( w_2 \)) was conducted on 5 benchmark CLRP instances (Gaskell67-21 \( \times \) 5, Gaskell67-32 \( \times \) 5-2, Gaskell67-36 \( \times \) 5, 20-5-1a, 50-5-1a). For the detailed reasons why we used CLRP instances instead of CLRP-FD instances, please refer to Subsection V-C2. When investigating the relationship between one (or one pair) of parameters and the approximate optimal solution, the other parameters were kept unchanged. Each instance was independently tested 10 runs.

#### 1) NUMBER OF ITERATIONS

The number of iterations plays an important role in HPSO. A large number of iterations can waste computing time, while a small number of iterations can stop the search process early and cannot obtain an optimal/near optimal solution. Table 1 presents the results of each tested instance over 10 runs, including the names of the instances, number of candidate depots (\( m \)), number of customers (\( n \)), optimal or best-known objective value (\( \text{BKOV} \)), how many runs with which the best-known solution was found (\( \text{Found} \)), average percentage deviation from the best-known solution, \( \text{APD} = \frac{\text{BOV} - \text{BKOV}}{\text{BKOV}} \times 100\% \), average CPU time in seconds (\( \text{Time} \)), and average iterations (\( \text{Aiter} \)) in which the best-known solution was found.

### TABLE 1. Results of the 5 tested instances over 10 runs.

| Instance   | \( m \) | \( n \) | \( \text{BKOV} \) | \( \text{BOV} \) | \( \text{Found} \) | \( \text{APD} \) % | \( \text{Time} \) (s) | \( \text{Aiter} \) |
|------------|--------|--------|-----------------|-----------------|-----------------|----------------|----------------|----------------|
| Gaskell67-21 \( \times \) 5 | 5      | 21     | 523.9           | 524.9           | 10              | 0.00           | 25.31          | 324.3          |
| Gaskell67-32 \( \times \) 5-2 | 5      | 32     | 504.3           | 504.7           | 7               | 0.08           | 37.21          | 508.9          |
| Gaskell67-36 \( \times \) 5 | 5      | 36     | 460.4           | 467.1           | 8               | 1.45           | 52.92          | 666.2          |
| 20-5-1a    | 5      | 20     | 54793           | 54809.4         | 8               | 0.03           | 31.66          | 284.2          |
| 50-5-1a    | 5      | 50     | 90111           | 90409.4         | 5               | 0.32           | 133.15         | 1348.5         |

#### 2) POPULATION SIZE

For setting the parameter \( H \), we considered 6 different population sizes in the range [30, 50] with a step of 5. Table 2 presents the average results of the 5 tested instances over 10 runs, including the population size \( H \), the average number of runs that produced the best-known objective values, the average CPU time in seconds, and the average value of APD.

Table 2 shows that the average number of runs and the average values of APD are relatively better when \( H \) is equal to or greater than 40. In addition, the larger the value of \( H \) is, the smaller the value of APD, but the more CPU time the HPSO needs. Considering both the solution quality and computing time, we set \( H = 40 \) in this study.

#### 3) PARAMETERS \( w_1 \) AND \( w_2 \)

The parameters \( w_1 \) and \( w_2 \) can not only decide which target solution a particle follows but also determine whether the VNS algorithm or path relinking is performed. According to Formulas (33) and (34), we can determine that the greater the values of \( w_1 \) and \( w_2 \) are, the more the VNS is performed during the first several iterations of the HPSO. To set \( w_1 \) and \( w_2 \), we considered 5 different combinations for their values, following the literature [32].

Table 3 shows that when \( w_1 = 0.8 \) and \( w_2 = 0.9 \), the average value of APD and the average number of runs that produced the best-known objective values are comparatively better, and they are 0.38% and 5, respectively. Hence, we set \( w_1 = 0.8 \) and \( w_2 = 0.9 \) in this study.
TABLE 2. Average results of the 5 tested CLRP instances over 10 runs obtained using different population size.

| Population size | Average number of runs | Average CPU time (s) | Average value of APD (%) |
|-----------------|------------------------|----------------------|--------------------------|
| H=30            | 4                      | 29.89                | 1.51                     |
| H=35            | 4                      | 36.19                | 0.74                     |
| H=40            | 5                      | 37.00                | 0.38                     |
| H=45            | 5                      | 44.03                | 0.38                     |
| H=50            | 5                      | 50.13                | 0.42                     |

TABLE 3. Average results of the 5 tested CLRP instances over 10 runs obtained using different combinations of $w_1$ and $w_2$.

| Parameters $w_1$ and $w_2$ | Average number of runs | Average CPU time (s) | Average value of APD (%) |
|-----------------------------|------------------------|----------------------|--------------------------|
| $w_1 = 0.8$ and $w_2 = 0.5$| 5                      | 37.00                | 0.38                     |
| $w_1 = 0.7$ and $w_2 = 0.8$| 5                      | 39.23                | 0.97                     |
| $w_1 = 0.6$ and $w_2 = 0.9$| 4                      | 33.08                | 2.24                     |
| $w_1 = 0.7$ and $w_2 = 0.8$| 3                      | 35.29                | 2.17                     |
| $w_1 = 0.7$ and $w_2 = 0.8$| 2                      | 37.01                | 2.05                     |

TABLE 4. Average results of the 5 tested CLRP instances over 10 runs obtained using different combinations of $c_1$ and $c_2$.

| Acceleration coefficients $c_1$ and $c_2$ | Average number of runs | Average CPU time (s) | Average value of APD (%) |
|-------------------------------------------|------------------------|----------------------|--------------------------|
| $c_1 = 2.05$ and $c_2 = 2.05$             | 5                      | 39.00                | 0.38                     |
| $c_1 = 2.15$ and $c_2 = 2.15$            | 3                      | 36.99                | 1.57                     |
| $c_1 = 2.25$ and $c_2 = 2.25$            | 3                      | 39.67                | 1.74                     |
| $c_1 = 2.35$ and $c_2 = 2.35$            | 2                      | 38.10                | 1.51                     |
| $c_1 = 2.45$ and $c_2 = 2.45$            | 5                      | 38.60                | 1.75                     |
| $c_1 = 2.05$ and $c_2 = 2.45$            | 4                      | 39.11                | 1.49                     |
| $c_1 = 2.45$ and $c_2 = 2.05$            | 3                      | 38.11                | 1.55                     |

TABLE 5. The relative value of the test problems.

| n   | Vehicle capacity | Depot capacity | Fixed cost of depots | Fixed cost of vehicles | Maximum traveling distance of vehicles | The number of vehicles provided by each depot |
|-----|------------------|----------------|----------------------|------------------------|---------------------------------------|--------------------------------------------|
| 30  | 5                | 300            | 900                  | 50                     | 10                                    | 240                                        |
| 50  | 5                | 500            | 2000                 | 50                     | 10                                    | 240                                        |
| 100 | 7                | 800            | 10000                | 50                     | 10                                    | 240                                        |

4) ACCELERATION COEFFICIENTS $c_1$ AND $c_2$

As shown in Formulas (30) and (31), $c_1$ and $c_2$ are used to compute the velocity of each particle, which has a large influence on the performance of the HPSO. Table 4 presents the average results of the 5 instances over 10 runs that were obtained from the HPSO using different combinations of $c_1$ and $c_2$, where both $c_1$ and $c_2$ were greater than 2, which followed the literature [32].

From the results presented in Table 4, we can determine that the first combination is relatively competitive, with which HPSO could produce the best-known solutions in 5 out of 10 runs, and it obtained a lower average percentage deviation within lower computational time.

According to the above discussion and experimental experience, throughout the experiment, the following parameters were used: the maximal number of iterations $G = 2000$, the number of particles $H = 40$, the acceleration coefficients $c_1 = c_2 = 2.05$, and parameters $w_1 = 0.8$, and $w_2 = 0.9$.

B. SENSITIVITY ANALYSIS ON PARAMETER DPI

In these experiments, a set of instances is generated using the approach presented in [21]. A small size instance with 30 customers and 5 candidate depots, a medium size instance with 50 customers and 5 candidate depots, and a large size instance with 100 customers and 7 candidate depots were considered. The coordinates of all of the customers and depots were generated randomly in the interval $[100 \times 100]$ [21]. In addition, the fuzzy demands of the customers, which are triangular fuzzy numbers such as $\tilde{d} = (d_1, d_2, d_3)$, were generated randomly [21]. In this way, $d_1, d_2$ and $d_3$ were generated within the intervals $[10, 35], [36, 60]$ and $[61, 110]$, respectively [21]. The data for the three test instances are shown in Table 5.

The values of DPI and API were also set following the approach presented in the literature [21]. The values of DPI varied from 0.1 to 1 with a step of 0.1. For the sake of simplicity, the value of API was always considered to be 1, which reduces the number of investigation cases. Tables 6-8 give the average computational results of 10 runs for the three test instances, respectively. The columns included in these three tables are the dispatcher preference index (DPI), the planned routes, the additional distances, the routing costs, the lost opportunity costs, the depot costs, the vehicle costs, the total costs (which consists of the routing costs, depot costs, vehicles costs and lost opportunity), and the CPU time obtained by the proposed algorithm. The results of Tables 6-8 are also depicted in Figures 9-11. As shown in Tables 6-8 and Figures 9-11, the total cost reaches its minimum when $DPI=0.6$. This value is the same as the value presented in the literature [17], [21], [24].

Figures 9-11 show the tendencies of the planned routes, additional distances, routing costs and total costs, which vary with $DPI$ from 0.1 to 1. As shown in Figures 9-11, lower values of DPI have shorter planned routes and longer additional distances. This finding indicates that lower values of
TABLE 6. Computational results with different DPI when the number of customers is 30.

| DPI | Planned routes | Additional distances | Routing costs | Lost opportunity | Depot costs | Vehicle costs | Total costs | CPU Time (s) |
|-----|----------------|----------------------|--------------|-----------------|------------|--------------|-------------|-------------|
| 0.1 | 56.13          | 320.66               | 888.79       | 0.00            | 175        | 35           | 1098.79     | 134.22      |
| 0.2 | 591.14         | 295.08               | 796.22       | 0.00            | 190        | 44           | 1030.22     | 140.08      |
| 0.3 | 522.55         | 263.80               | 786.35       | 0.00            | 200        | 47           | 1033.35     | 139.46      |
| 0.4 | 530.79         | 234.48               | 755.27       | 0.00            | 195        | 54           | 1004.27     | 127.71      |
| 0.5 | 570.43         | 163.02               | 733.46       | 0.00            | 195        | 60           | 984.46      | 132.99      |
| 0.6 | 631.23         | 93.50                | 724.74       | 0.00            | 195        | 64           | 983.74      | 128.25      |
| 0.7 | 702.61         | 16.88                | 719.49       | 0.00            | 200        | 72           | 991.49      | 129.12      |
| 0.8 | 743.39         | 4.56                 | 747.95       | 0.00            | 190        | 85           | 1022.99     | 130.44      |
| 0.9 | 774.53         | 0.14                 | 774.77       | 0.00            | 185        | 95           | 1054.77     | 132.26      |
| 1   | 771.01         | 0.00                 | 771.01       | 0.00            | 200        | 101          | 1072.01     | 133.42      |

TABLE 7. Computational results with different DPI when the number of customers is 50.

| DPI | Planned routes | Additional distances | Routing costs | Lost opportunity | Depot costs | Vehicle costs | Total costs | CPU Time (s) |
|-----|----------------|----------------------|--------------|-----------------|------------|--------------|-------------|-------------|
| 0.1 | 729.06         | 243.35               | 972.41       | 0.00            | 150        | 40           | 1162.41     | 164.07      |
| 0.2 | 741.37         | 203.19               | 944.56       | 0.00            | 150        | 44           | 1138.56     | 173.88      |
| 0.3 | 736.31         | 204.07               | 940.38       | 0.00            | 150        | 50           | 1140.38     | 178.55      |
| 0.4 | 753.84         | 172.58               | 926.68       | 0.00            | 150        | 51           | 1127.68     | 171.68      |
| 0.5 | 764.86         | 124.61               | 889.47       | 0.00            | 150        | 57           | 1101.47     | 168.12      |
| 0.6 | 785.17         | 39.54                | 824.71       | 0.00            | 150        | 64           | 1058.71     | 168.15      |
| 0.7 | 825.23         | 6.15                 | 831.38       | 0.00            | 150        | 71           | 1022.38     | 173.14      |
| 0.8 | 834.22         | 0.70                 | 834.92       | 0.00            | 150        | 81           | 1065.92     | 182.63      |
| 0.9 | 837.60         | 0.01                 | 837.61       | 0.00            | 150        | 87           | 1074.61     | 178.96      |
| 1   | 881.56         | 0.00                 | 881.56       | 0.00            | 150        | 99           | 1130.56     | 175.05      |

TABLE 8. Computational results with different DPI when the number of customers is 100.

| DPI | Planned routes | Additional distances | Routing costs | Lost opportunity | Depot costs | Vehicle costs | Total costs | CPU Time (s) |
|-----|----------------|----------------------|--------------|-----------------|------------|--------------|-------------|-------------|
| 0.1 | 1179.91        | 432.65               | 1612.56      | 0.00            | 50         | 60           | 1715.56     | 339.02      |
| 0.2 | 1208.62        | 475.04               | 1683.65      | 0.00            | 50         | 60           | 1793.65     | 354.18      |
| 0.3 | 1223.95        | 457.58               | 1681.53      | 0.00            | 50         | 60           | 1791.53     | 334.58      |
| 0.4 | 1194.29        | 427.75               | 1622.04      | 0.00            | 60         | 62           | 1744.04     | 350.10      |
| 0.5 | 1245.96        | 363.24               | 1609.20      | 0.00            | 60         | 66           | 1735.20     | 361.18      |
| 0.6 | 1270.29        | 97.05                | 1367.34      | 0.00            | 70         | 70           | 1497.34     | 349.22      |
| 0.7 | 1333.42        | 42.88                | 1376.29      | 0.00            | 60         | 80           | 1516.29     | 339.86      |
| 0.8 | 1420.77        | 6.68                 | 1421.45      | 0.00            | 70         | 90           | 1581.45     | 330.52      |
| 0.9 | 1485.40        | 0.00                 | 1485.40      | 0.00            | 50         | 100          | 1635.40     | 341.94      |
| 1   | 1567.08        | 0.00                 | 1567.08      | 0.00            | 60         | 112          | 1739.08     | 340.40      |

DPI have a tendency to use all of the vehicle capacity, which results in increased total additional distances because the vehicles visit the customers but do not have enough capacity to serve them. A higher value of DPI denotes a tendency to use less vehicle capacity, which results in longer planned routes and shorter additional distances due to "route failure". Moreover, Tables 6-8 also show that the cost of employing vehicles is increased with increases in the value of DPI. To ensure higher service to customers at a higher value of the DPI, fewer customers are considered in each vehicle route, and thereby, more vehicles are employed. Although the cost of lost opportunity should also be increased with the increase in the DPI because of the unavailability of vehicles, its values shown in Tables 6-8 are always equal to "0" because enough vehicles can be employed by each candidate depot in this study. Considering the total cost, the proper value of DPI should be approximately 0.6.

C. PERFORMANCE OF THE PROPOSED HPSO

1) EFFECTIVENESS OF THE POSITION UPDATING STRATEGY IN THE HPSO

The main difference between the traditional PSO and HPSO is the position updating strategy employed in this paper, which consists of a path relinking algorithm and VNS. To study the effectiveness of the proposed position updating mechanism, we compared 2 PSOs with the HPSO on six benchmark CLRP instances.

PSO1: the proposed HPSO with the path relinking algorithm only to update the position. This is to evaluate whether the VNS employed in the HPSO enhances the exploration to improve the overall performance of the HPSO.

PSO2: the proposed HPSO with the VNS only to update the position. This is to evaluate the effectiveness of the path relinking algorithm.

The mean and standard deviation (SD) values of the best objective values produced by PSO1, PSO2 and HPSO over 10 runs are presented in Table 9. It can clearly be seen that the HPSO outperforms PSO1 and PSO2 in every instance with regard to the mean values. PSO1 provided the worst mean values, and the difference in HPSO versus PSO1, and PSO2 versus PSO1 is quite significant. This finding indicates that the VNS can effectively improve the performances of HPSO and PSO2. Although the VNS is mainly for local exploitation, 5 different neighborhoods were allowed to explore multiple promising areas in the search space, and this strategy also enhanced the exploration ability. Therefore, HPSO and PSO2 achieve better performance than PSO1. In addition, compared with PSO2, the position
updating mechanism is more effective in guiding the search toward promising areas in the search space because it takes advantage of the best particle in the current population and the global best particle during the evolution.

According to the results shown in Table 9, we can conclude that HPSO is more stable. To support this observation, we present the box plots of the three PSOs on the six selected instances in Figure 12. In Figure 12, the box plots of the HPSO and PSO2 are enlarged for clearly observing the variability of the results. It is no doubt that HPSO has the best stability.

2) COMPARISON WITH SOFTWARE CPLEX
Because of the fuzzy demands of the customers and the sites of all customers and depots are randomly generated in this study, it is not possible to compare the HPSO results shown in Tables 6-8 with previous work. To evaluate the efficiency of the HPSO, the solutions shown in Tables 6-8 are compared with their lower bounds. In this study, the assumption of each customer’s demand is relaxed for obtaining the lower bound, and each customer’s demand equals its left boundary $d_1$. Thus, the constraints (10) and (11) in the model of CLRP-FD are replaced by the constraints (35) and (36),
respectively.

\[
\sum_{i \in V} \sum_{j \in J} d_{ijk} x_{ijk} \leq Q \quad \forall k \in K \tag{35}
\]

\[
\sum_{j \in J} d_{ij} y_{ij} \leq P_{i} z_{i} \quad \forall i \in I \tag{36}
\]

Table 9. Comparison among HPSO, PSO1 and PSO2.

| Instances     | m  | n  | BKOV | HPSO | PSO1 | PSO2 |
|---------------|----|----|------|------|------|------|
|               | mean | SD | mean | SD   | mean | SD   |
| Gaskell67-21x5 | 5   | 21 | 424.9 | 0.0  | 366.3 | 46.6 |
|                | 425.8 | 1.8 | 506.0 | 3.6  |
| Gaskell67-32x5 | 5   | 32 | 504.3 | 0.6  | 504.7 | 2.1  |
|                | 846.6 | 36.1 | 853.5 | 7.5  |
| Gaskell67-36x5 | 5   | 36 | 460.4 | 4.1  | 683.8 | 48.0 |
|                | 467.7 | 6.2 | 5492.5 | 14.2 |
| Christofides69-75x10 | 10 | 75 | 844.4 | 7.1  | 1208.5 | 130.9 |
|                | 853.5 | 7.5 | 10080 | 10080 |
| 20-5-1a       | 5   | 20 | 54793 | 32.8 | 54809.4 | 32.8 |
|                | 5492.5 | 114.2 | 10080 | 10080 |
| 50-5-1a       | 5   | 50 | 90011 | 319.8 | 90402.9 | 319.8 |
|                | 50523.2 | 318.5 | 10080 | 10080 |

Table 10. The results of two instances with their lower bounds.

| Instances | Quality of solutions | CPU time (s) |
|-----------|---------------------|--------------|
|           | HPSO (DFP = 0.6 and AP = 1) | Lower bound (CPLEX) | Gap (%) | HPSO (DFP = 0.6 and AP = 1) | CPLEX |
| 30 × 5    | 983.74              | 704.59       | 39.62%  | 128.25              | 10080  |
| 50 × 5    | 1038.71             | 821.4        | 24.66%  | 168.15              | 10080  |
| 100 × 7   | 1497.34             | 1008.56      | 4.8%    | 300.17              | 10080  |

Table 11. Performance of the proposed HPSO on some CLRP-FD instances obtained from standard CLRP test instances.

| Instance of CLRP instance | Optimal solution | HPSO solution (DFP = 0.6 and AP = 1) | Gap (%) |
|---------------------------|------------------|--------------------------------------|---------|
| Planned routes            | Additional distances | Depot cost | Vehicle cost | Total cost |
| 20-5-1a                   | 54793            | 42792.16 | 5.38         | 23549       | 17000 | 83346.54 | 55.76% |
| 20-5-1b                   | 39104            | 29919.98 | 14.75        | 15497       | 5000  | 50431.72 | 28.97% |
| 20-5-2a                   | 48908            | 33086.33 | 69.79        | 23769       | 10000 | 65855.33 | 34.65% |
| 20-5-2b                   | 37342            | 36823.43 | 10.29        | 17105       | 6000  | 59938.71 | 59.66% |
| 50-5-1a                   | 87109.64         | 107806.63 | 92.48       | 25422       | 32000 | 163341.11 | 89.81% |
| 50-5-1b                   | 61555.22         | 123676.95 | 33.75       | 35939       | 36000 | 195649.7 | 217.64% |
| 50-5-2a                   | 86055.01         | 77343.13 | 182.2       | 32714       | 25000 | 135239.33 | 57.15% |
| 50-5-2b                   | 65787.75         | 51521.65 | 89.01       | 29319       | 10000 | 90929.66 | 38.22% |

To further evaluate the efficiency of the proposed HPSO and the performance of the model of CLRP-FD, 8 test instances are randomly selected from the standard CLRP library (http://prodhonn.free.fr/Instances/instances_us.htm) for conducting further numerical experiments. The number of customers \(n\) is either 20 or 50, the number of depots \(m\) is 5, and the capacity of vehicle \(Q\) equals 70 or 150 (indicated by the letters “a” and “b” in Table 11). To use the standard CLRP test instances in the case of CLRP-FD, the following steps that were proposed in [21] are implemented on the CLRP instances to change them into CLRP-FD instances:

Step 1: Based on the deterministic demand of each customer, we generate the corresponding triangular fuzzy demand as follows: (1) set the deterministic demand of each customer in the CLRP as the left boundary \((d_1)\) of the fuzzy demand in CLRP-FD; (2) triple the deterministic demand and consider it to be the right boundary \((d_3)\) of the fuzzy demand; (3) randomly generate a number in the interval \([1.5d_1, 2.5d_1]\) and consider it to be the middle number of the triangular fuzzy demand \((d_2)\).

Step 2: Denote the maximum right boundaries of the fuzzy demands of all of the customers and the vehicle capacity of the CLRP instance as \(D_3 = D_3 = \max_{j \in J} \{d_3\}\) and \(Q\), respectively. Consider \(Q\) to be the vehicle capacity of the CLRP-FD instance if \(D_3 \leq Q\); otherwise, consider \(D_3\) to be the vehicle capacity of the CLRP-FD instance.

Step 3: Triple the depot’s capacity of the CLRP instance and consider that to be the depot’s capacity of the corresponding CLRP-FD instance.
Clearly, the optimal solution of the CLRP instance will be a lower bound for the optimal solution of its corresponding CLRP-FD instance, obtained using the above steps. Table 11 summarizes the computational results of 8 standard CLRP instances and their corresponding CLRP-FD instances. The first column of Table 11 gives the names of the test instances. The second column shows the solutions of the standard CLRP instances obtained using the CPLEX solver within 3 hours. The next 5 columns present the solutions of the corresponding CLRP-FD instances obtained using HPSO with DPi=0.6 and API=1. The Gap between the HPSO solution and its lower bound is presented in the last column. The values of gap indicate that the total cost of the CLRP-FD instances is at least 28.97% larger than that of its corresponding CLRP instances. Therefore, the decision makers should cautiously make the decisions in this situation, which lacks sufficient information to make a precise judgment.
H. Zhang et al.: Hybrid Heuristic Based on a Particle Swarm Algorithm to Solve the CLRP-FD

3) COMPARISON WITH THE STATE-OF-THE-ART HEURISTIC ALGORITHMS

To thoroughly evaluate the performance of HPSO, we also compared it with seven state-of-the-art algorithms presented in the literature, which are listed as follows.

- Greedy randomized adaptive search procedure (GRASP) [46]
- Memetic algorithm with population management (MA|PM) [47]
- Lagrangean relaxation and granular tabu search method (LRGTS) [48]
- Two-phase hybrid heuristic algorithm (2-Phase HGTS) [49]
- Multiple ant colony optimization algorithm (MACO) [50]
- Hybrid genetic algorithm (HGA) [51]
- Granular variable tabu neighborhood search (GVTNS) [52]

Table 12 provides a comparison of 27 benchmark CLRP instances between HPSO and the above popular heuristic algorithms. The first 11 instances are from the Barreto dataset, and the remaining 16 instances are from the Prins dataset. Table 12 presents the name of the instance, the number of candidate depots \(m\), the number of customers \(n\), the best known objective value (BOV), the best objective value (BOV) obtained by each algorithm, and the average computational time (Time) of each algorithm for solving each instance. In Table 12, the figures shown in bold indicate the best results among the compared algorithms. From Table 12, it can be seen that GRASP, MA|PM, LRGTS, 2-Phase HGTS, MACO, HGA, GVTNS, and HPSO obtained optimal results in 7, 10, 5, 17, 20, 14, and 20 compared instances, respectively. HPSO did not produce optimal solutions for 9 of the 27 instances compared to the best-known results presented in the literature. It is evident that except for HGA, HPSO produced better solutions than the other six compared algorithms. Therefore, we can conclude that HPSO has better performance than (or similar performance with) the other six popular algorithms in terms of the solution quality.

We should also note that the state-of-the-art hybrid genetic algorithm presented in [51] incorporates additional heuristic components. Apart from the selection and crossover operators, the mutation operator in the HGA is very complicated, which employs the add operator, swap operator and multiple local searches. One of the crucial things in the optimization process of HGA is to determine the proper crossover and mutation rate. Compared with the HGA, the proposed HPSO has a simpler framework from the point of implementation.

In Table 12, it can be seen that HPSO spent much more computational time in solving all of the compared instances. We should note that the different hardware, compilers, and programming languages have a large impact on the heuristic algorithms’ effectiveness. Table 13 presents the implemented platforms and programming languages of the comparison algorithms. It is possible that the slower MATLAB coder and larger number of generations and local search iterations resulted in a larger computational time for the HPSO.

We also conducted statistical tests using the statistical software SPSS to determine whether the above observations between the HPSO and the compared algorithms are statistically significant. The BOVs of each instance presented in Table 12 were considered for the statistical tests.

The results of the nonparametric Friedman test are shown in Table 14, where the significance level \(\alpha = 0.05\). Table 14 presents the average ranks calculated for the compared algorithms, the Friedman \(\chi^2\) statistic values, \(p\)-values, and the decision about acceptance or rejection. As shown in Table 14, for all instances, the Friedman \(\chi^2\) statistic values were greater than the critical value of 7.815, indicating that the differences in the performance of the algorithms were statistically significant. The Friedman test indicated that HPSO had lower \(p\)-values compared to the other algorithms, suggesting better performance. The results of the post-hoc test showed that HPSO had significantly better performance than GRASP, MA|PM, LRGTS, 2-Phase HGTS, MACO, HGA, GVTNS, and HPSO for most of the instances.

In conclusion, the proposed HPSO algorithm showed promising results in solving the CLRP-FD problem. It performed better than or similar to the state-of-the-art algorithms in terms of solution quality and computational time. The results of the statistical tests further supported the findings, indicating that HPSO is a statistically significant improvement over the compared algorithms.
VI. CONCLUSION

In this paper, the CLRP-FD was studied. Based on credibility theory, a fuzzy chance constrained programming model was formulated for this problem. For the purpose of solving this model, a hybrid heuristic algorithm based on the PSO was proposed. In the HPSO, the path relinking algorithm is used to update the position of each particle, and local search is implemented using VNS to improve the quality of the solution. In the solution process of the HPSO, stochastic simulation was employed to estimate the additional distances that resulted from fuzzy demands and route failures for every planned route. To verify the validity of the presented model and evaluate the performance of the proposed HPSO algorithm, three numerical instances with different sizes, which were generated according to the approach shown in the literature, were solved. The experimental results showed that the total distribution cost can be greatly influenced by the value of the dispatcher preference index (DPI). In addition, to investigate the efficiency of the proposed HPSO, the lower bounds of the solutions were computed using the optimization solver, CPLEX 12.8. Results revealed that CPLEX could not optimally solve the tested instances within 3 hours, but the proposed HPSO could obtain satisfactory solutions in a reasonable computation time. To further illustrate the performance of the proposed model, 8 CLRP-FD instances were generated based on the corresponding standard CLRP instances with some changes. The Gap between the HPSO solution with $DPI=0.6$ and $API=1$ and its lower bound is at least 28.97%, which indicates that the decision makers should cautiously make the decisions in the absence of sufficient information. We also compared HPSO with seven state-of-the-art algorithms presented in the literature, and the results showed that HPSO has better or similar performance with these state-of-the-art heuristic algorithms.

To extend this work, more future work can be done as follows: (1) considering other parameters of the CLRP as fuzzy variables, such as the travel time and time windows, (2) developing other variants of the CLRP-FD, such as the CLRP-FD with open routing tours and the CLRP-FD with time windows, (3) designing other meta-heuristic algorithms to solve the CLRP-FD and its variants, and (4) integrating exact technique/optimizer like CPLEX into the HPSO to solve subproblems in an optimal way.
REFERENCES

[1] R. K. Ahuja, J. B. Orlin, S. Pallottino, M. P. Scaparra, and M. G. Scutellà, “A multi-exchange heuristic for the single-source capacitated facility location problem,” Manage. Sci., vol. 50, no. 6, pp. 749–760, Jun. 2004.

[2] M. R. Khouadjia, B. Sarasola, E. Alba, L. Jourdan, and E.-G. Talbi, “A comparative study between dynamic adapted PSO and VNS for the vehicle routing problem with dynamic requests,” Appl. Soft Comput., vol. 12, no. 4, pp. 1439–1449, Apr. 2012.

[3] H. Zhang, Q. Zhang, L. Ma, Z. Zhang, and Y. Liu, “A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows,” Inf. Sci., vol. 490, pp. 166–190, Jul. 2019.

[4] E. B. Tirkolaee, A. Goli, M. Pahlevan, and R. M. Kordestanizadeh, “A robust bi-objective multi-trip periodic capacitated arc routing problem for urban waste collection using a multi-objective invasive weed optimization,” Soft Comput., vol. 23, no. 11, pp. 1089–1101, Aug. 2019.

[5] I. Kaoaglou, F. Aliparvank, I. Kaka, and B. Dengiz, “A branch and cut algorithm for the location-routing problem with simultaneous pickup and delivery,” Eur. J. Oper. Res., vol. 211, no. 2, pp. 318–332, Jun. 2011.

[6] C. Prodhon, “A hybrid evolutionary algorithm for the periodic location-routing problem,” Eur. J. Oper. Res., vol. 210, no. 2, pp. 204–212, Apr. 2011.

[7] S. H. Hashemi Doulabi and A. Seifi, “Lower and upper bounds for location-arc routing problems with vehicle capacity constraints,” Eur. J. Oper. Res., vol. 224, no. 1, pp. 189–208, Jan. 2013.

[8] J.-M. Belenguer, E. Benavent, C. Prins, C. Prodhon, and R. W. Calvo, “A branch-and-cut method for the capacitated location-routing problem,” Comput. Oper. Res., vol. 38, no. 6, pp. 931–941, Jun. 2011.

[9] F. Samaniiouglou, “A multi-objective mathematical model for the industrial hazardous waste location-routing problem,” Eur. J. Oper. Res., vol. 226, no. 2, pp. 332–340, Apr. 2013.

[10] M. Rabbani, H. Farrokhi-Asl, and B. Agharian, “Solving the bi-objective location routing problem by a NSGA-II combined with clustering approach: Application in waste collection problem,” J. Ind. Eng. Int., vol. 13, no. 1, pp. 13–27, Oct. 2016.

[11] M. V. Reddy, V. V. Panicker, and R. Sridharan, “Ant colony optimisation for location routing problem and its application to bill delivery services,” Int. J. Logistics Syst. Manage., vol. 14, no. 1, pp. 1–37, Jan. 2013.

[12] I. Or and W. P.ierskalla, “A transportation location-allocation model for regional blood banking,” IIE Trans., vol. 11, no. 2, pp. 86–95, Mar. 1979.

[13] K. Govindan, A. Jafarian, R. Khodaverdi, and K. Devika, “Two-echelon multiple-vehicle location–routing problem with time windows for optimization of sustainable supply chain network of perishable food,” Int. J. Prod. Econ., vol. 152, no. 2, pp. 9–28, Jun. 2014.

[14] A. Golì and S. M. R. Davoodi, “Coordination policy for production and delivery scheduling in the closed loop supply chain,” Prod. Eng., vol. 12, no. 5, pp. 621–631, Jun. 2018.

[15] A. Goli, H. K. Zare, R. Tavakkoli-Moghaddam, and A. Sadegheh, “Multiobjective fuzzy mathematical model for a financially constrained closed-loop supply chain with labor employment,” Comput. Intell., vol. 36, no. 1, pp. 4–34, Jul. 2020.

[16] K. M. Ferreira and T. A. de Queiroz, “Two effective simulated annealing algorithms for the location-routing problem,” Appl. Soft Comput., vol. 70, pp. 389–422, Sep. 2018.

[17] N. Ghaffarif-Nasab, S. G. Ahar, and M. Ghazanfari, “A hybrid simulated annealing based heuristic for solving the location-routing problem with fuzzy demands,” Scientia Iranica, vol. 20, no. 3, pp. 919–930, Jun. 2013.

[18] V. F. Yu and S.-W. Lin, “Multi-start simulated annealing heuristic for the location-routing problem with simultaneous pickup and delivery,” Appl. Soft Comput., vol. 24, pp. 284–290, Nov. 2014.

[19] Y. Marinakis, “An improved particle swarm optimization algorithm for the capacitated location routing problem and for the location routing problem with stochastic demands,” Appl. Soft Comput., vol. 37, pp. 680–701, Dec. 2015.

[20] X. Yu, Y. Zhou, and X.-F. Liu, “A novel hybrid genetic algorithm for the location routing problem with tight capacity constraints,” Appl. Soft Comput., vol. 85, Dec. 2019, Art. no. 105760.

[21] A. Nadizadeh and H. Hosseini Nasab, “Solving the dynamic capacitated location-routing problem with fuzzy demands by hybrid heuristic algorithm,” Eur. J. Oper. Res., vol. 238, no. 2, pp. 458–470, Oct. 2014.

[22] A. Goli, H. Khadem-Zare, R. Tavakkoli-Moghaddam, and A. Sadegheh, “Hybrid artificial intelligence and robust optimization for a multi-objective product portfolio problem case study: The dairy products industry,” Comput. Ind. Eng., vol. 137, Nov. 2019, Art. no. 106090.
[48] C. Prins, C. Prodhon, A. Ruiz, P. Soriano, and R. W. Calvo, “Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation-granular tabu search heuristic,” Transp. Sci., vol. 41, no. 4, pp. 470–483, Nov. 2007.

[49] J. W. Escobar, R. Linfati, and P. Toth, “A two-phase hybrid heuristic algorithm for the capacitated location-routing problem,” Comput. Oper. Res., vol. 40, no. 1, pp. 70–79, Jan. 2013.

[50] C.-J. Ting and C.-H. Chen, “A multiple ant colony optimization algorithm for the capacitated location routing problem,” Int. J. Prod. Econ., vol. 141, no. 1, pp. 34–44, Jan. 2013.

[51] R. B. Lopes, C. Ferreira, and B. S. Santos, “A simple and effective evolutionary algorithm for the capacitated location-routing problem,” Comput. Oper. Res., vol. 70, pp. 155–162, Jan. 2016.

[52] J. W. Escobar, R. Linfati, M. G. Baldoquin, and P. Toth, “A granular variable tabu neighborhood search for the capacitated location-routing problem,” Transp. Res. B, Methodol., vol. 67, pp. 344–356, Sep. 2014.

HUIZHEN ZHANG was born in Shanxi, China, in 1979. She received the B.S. degree in economics from the Shanxi University of Finance and Economics, Shanxi, in 2002, the M.S. degree in earth exploration and information technology from the Chengdu University of Technology, Sichuan, China, in 2005, and the Ph.D. degree in management science and engineering from the University of Shanghai for Science and Technology, Shanghai, China, in 2009.

From October 2009 to September 2011, she was a Postdoctoral Stage with the Department of Statistics and Operations Research, Rey Juan Carlos University, Madrid, Spain, supported by the grant from the Agencia Espanola de Cooperacion Internacional para el Desarrollo (AECID). Since November 2011, she has been an Associate Professor of management science and engineering with the University of Shanghai for Science and Technology. She is the author of four books and more than 60 articles. Her research interests include mixed integer programming, combinatorial optimization, and meta-heuristic algorithms.

FAN LIU was born in Ziyang, Sichuan, China, in 1995. He received the B.S. degree in material forming and controlling engineering from the Southwest University of Science and Technology, Mianyang, China, in 2017. He is currently pursuing the M.S. degree in management science and engineering with the University of Shanghai for Science and Technology, Shanghai, China.

His research interests include combinatorial optimization and meta-heuristic algorithms.

LIANG MA was born in Shanghai, China, in 1964. He received the B.S. degree in mathematics from Fudan University, Shanghai, in 1985, the M.S. degree in system engineering from the Shanghai Institute of Mechanical Engineering, China, in 1988, and the Ph.D. degree in management science and engineering from Shanghai Jiao Tong University, Shanghai, in 2000. From July 1988 to May 1994, he was an Assistant Professor with the Department of Management Engineering, Shanghai Institute of Mechanical Technology. Since June 1994, he has been a Professor of management science and engineering with the University of Shanghai for Science and Technology. He is the author of eight books and more than 300 articles. His research interests include operations research, management system engineering, and meta-heuristic algorithms.

Prof. Ma’s awards and honors mainly include the Shanghai Shuguang Scholar Award and the Shanghai Yucai Award from the Shanghai Municipal Education Commission.

ZIYING ZHANG was born in Hubei, China, in 1978. He received the Ph.D. degree in materials science from Fudan University, Shanghai, China, in 2013. Since September 2013, he has been an Associate Professor with the School of Materials Engineering, Shanghai University of Engineering Science. He has participated more than ten academic research projects as a Principal Investigator/Director or a Key Participant. He is a Special Expert on Development in Northern Jiangsu Province of China. His recent research interests include the corrosion and protection of metallic materials, the theoretical research and computational simulation of the structure and properties of IOL materials, and battery electrochemistry.

VOLUME 8, 2020