A Hybrid Particle Swarm Optimization Algorithm for Multi-Objective Pickup and Delivery Problem with Time Windows

Shuilong Zou
Nanchang Institute of Science & Technology, Nanchang, P.R.China
Email: zsl69088@163.com

*Jin Li
School of Computer Science and Information Engineering, Zhejiang Gongshang University, Hangzhou, P.R.China;
Contemporary Business and Trade Research Center of Zhejiang Gongshang University, Hangzhou, P.R.China
*Corresponding author: lijinkingdom@163.com

Xueqian Li
Business school, University of Shanghai for Science and Technology, Shanghai, P.R.China
Email: lxq4238@163.com

***Abstract*—This paper studies the multi-objective pickup and delivery problem with time windows (PDPTW), in which a fleet of homogeneous vehicles with the same capacities located in a depot serve a collection of given transportation requests. Each request is composed of a pickup location, a delivery location and a given load. The PDPTW is to determine a vehicle scheduling strategy with the objectives of minimizing the number of vehicles utilized, the total travel distances and the total waiting times. A mixed integer programming model is built to formulate this multi-objective PDPTW. Then a novel hybrid particle swarm optimization (HPSO) is proposed to solve this problem. This algorithm adds particles neighbor information to diversify the particle swarm and use the variable neighborhood search (VNS) to enhance the convergence speed. Finally, some numerical experiments based on existing benchmark instances are given to show the effectiveness and feasibility of the algorithm.**

***Index Terms*—vehicle routing problem with time windows, pickup and delivery, transportation, particle swarm optimization, algorithm**

I. INTRODUCTION

Since the Vehicle Routing Problem (VRP) was proposed by Dantzig and Ramser [1], it has been a hot issue in the field of management science and operational research. Due to the wide applications in the logistics management and transportation management [2-5], a lot of new constraints is added to the classical VRP and generated new problems, e.g. Vehicle Routing Problem with Backhauls (VRPB), Vehicle Routing Problem with Time Windows (VRPTW) [6], Vehicle Routing Problem with Simultaneous Pickup and Delivery (VRPSPD) [7] and Pickup and Delivery Problem with Time Windows (PDPTW), etc. Pickup and delivery problem with time windows is a generalization of well-known VRPTW.

PDPTW can be described as: A fleet of homogeneous vehicles located in a depot serve a collection of given transportation requests. Each request is composed of a pickup location, a delivery location and a given load. Each location has a time window, that is the earliest start time and latest start time for loading and unloading. The vehicle can't arrive at a location after its latest start time, but allows to arrive at the location before the earliest start time. The vehicle must be wait until the time window is open if it reach early. These vehicles must be routed to serve all requests to minimize the number of vehicles used, the total distance traveled, and total waiting time, etc., while satisfying the time windows and vehicle capacity constraints.

PDPTW can be used to model many core problems arising in production and daily life practice, e.g. logistics and public transit. Searching for good solutions to these problems is very important, because it can provide decision-makers to utilize the existing fleet in the most cost-effective way to meet customer demands. Rekiek, Delehambre and Saleh [8] used the PDPTW model to solve the vehicle routing problem of handicapped person transportation. Fagerholt and Christiansen [9] applied this model in dealing with combined ship scheduling and allocation problem.

Compared with VRPTW, existing literatures reporting on PDPTW are relatively less. Since VRPTW was a well-known NP-hard problem [10], PDPTW was also an NP-hard problem. Most work focused on different kinds of intelligent heuristic algorithms to solve PDPTW. Hoong and Zhe [11] present a two-phase method to solve the PDPTW. In the first phase, they applied a novel construct heuristics to generate an initial solution. In the second phase, a tabu search method is proposed to improve the solution. Wang and Chen [12] developed a mixed binary integer programming model for the simultaneous delivery and pickup problem with time windows and proposed a co-evolution genetic algorithm with variants of the cheapest insertion method. Lu and Dessouky [13] presented a new insertion-based construction heuristic to solve the multi-vehicle pickup and delivery problem with

doi:10.4304/jcp.8.10.2583-2589

time windows. Li and Lim [14] proposed a metaheuristic algorithm combining tabu search and simulated annealing, and solve a benchmark problem with 100 nodes. Pankratz [15] introduced a group genetic algorithms, realized the vehicles-based genetic coding, crossover and mutation methods by using special coding and decoding technology. Wang and Chen [16] addressed a flexible delivery and pickup problem with time windows and formulated the problem into a mixed binary integer programming model. Then, a coevolutionary algorithm incorporated with a variant of the cheapest insertion method is developed to speed up the solution procedure.

Hosny and Mumford [17] compared several construction algorithms that generated initial feasible solution to the multiple vehicle pickup and delivery problem with time windows, and suggested a simple routing heuristic to create individual vehicle routes. Qu and Bard [18] developed a greedy randomized adaptive search procedure (GRASP) with several novel features. In the construction phase, shipment requests were inserted into routes until all demand was satisfied or no feasible insertion exists. In the improvement phase, an adaptive large neighborhood search algorithm was used to modify portions of the feasible routes. Specialized removal and insertion heuristics were designed for this purpose. Nanry and Barnes [19] presented a reactive tabu search approach to solve the pickup and delivery problem with time windows using three distinct move neighborhoods that capitalized on the dominance of the precedence and coupling constraints. A hierarchical search methodology was used to dynamically alternate between neighborhoods in order to negotiate different regions of the solution space and adjust search trajectories.

Most previous work focused on the single objective pickup and delivery problem with time windows. The traditional objective only concerned with minimizing the number of vehicles or the total distance, and paid less attention to minimize the total waiting time. In reality, the vehicle's waiting time is also an important measure to the efficiency and costs of the vehicle's distribution. Moreover, less research work uses the particle swarm optimization method to solve PDPTW. The PSO is an effective heuristic algorithm, and may produce better solutions for PDPTW. Therefore, this paper will study the multi-objective PDPTW, and design an effective and fast improved PSO to solve this problem.

This paper is organized as follows. The problem formulation for tourist trip design problem in time-dependent network is described in Section 2. In Section 3, the label correcting algorithm is given. Moreover, Section 4 validates the proposed label correcting algorithm by numerical example. Finally, the concluding remarks and further research are included in Section 5.

II. PROBLEM DESCRIPTION

A. Prolbem Assumptions

In the pickup and delivery problem with time windows, a fleet of vehicles serve a group of customers' requests. Each request specifies a pickup and delivery location. Vehicles must be routed to serve all requests.

We assume that there is unlimited number of vehicles and all vehicles are the same type with the same capacity Q. Given a transportation network $G=(V,E)$, where V is the set of transportation requests, and E is the edge set. Each pickup or delivery is considered as a job. For each request $i \in V$, q_i is a load transported from an origin node V_i^1 to a destination node V_i^2. q_i^1 and q_i^2 are pickup and delivery load respectively. Then, $V^1 = \bigcup_{i \in V} V_i^1$ and $V^2 = \bigcup_{i \in V} V_i^2$ are represented as the sets of all origins and destinations nodes respectively. To simplify the problem, it is assumed that V_i^1 and V_i^2 is disjoint. Let $V = V_i^1 \cup V_i^2$, $n = |V|$. Let M denote the set of vehicles, and $m = |M|$ represents the number of vehicles. Each vehicle starts and ends with depot 0 with no cargo. For each $(i,j) \in E$, d_{ij} is the travel distance and t_{ij} denotes the travel time. Let $[e_i, l_i]$ be the time window at location i. Since the service durations time at the origins and destinations could be included in the travel times, we will not considered them explicitly.

For a pickup and delivery route R_k of vehicle k, the following conditions must be satisfied. R_k starts and ends in the depot 0; Both or neither V_i^1 and V_i^2 belongs to R_k; If both V_i^1 and V_i^2 belong to R_k, V_i^1 is visited before V_i^2; Vehicle k visits each location once and only once; The vehicle load never exceeds the capacity Q. Let a_i and d_i denote the arrival time and departure time of location i. The vehicle can arrive at the location before e_i, but can't reach it after l_i. If the vehicle arrives early, it must wait until the time window open. Here, $d_i = \max\{a_i, e_i\}$.

Figure 1 shows a PDPTW example. In Figure 1, there are three vehicles and five requests (ten pickup or delivery jobs). It gives a good description for PDPTW.

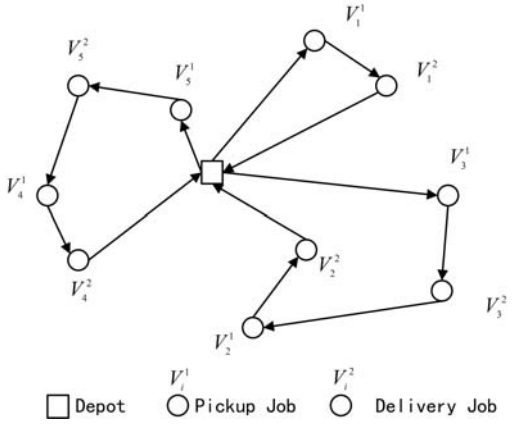

Figure 1. An example for PDPTW.

There are two binary variables. Let x_{ijk} be 1 if only if vehicle k is traveling from node i to node j

($i \in V, j \in V, k \in M$), y_{ik} is 1 if only if request i is assigned to vehicle k ($i \in V, k \in M$), and Z_j denote an intermediate variable storing the total load of the vehicle visiting job j. There are multi-objective functions for PDPTW described in the following:

- minimize the number of the vehicles, which is always the most dominant part of the cost;
- minimize the total travel distance, which is the sum of lengths of all the routes;
- minimize the total waiting time of the vehicles, which also leads to cost and reduce the distribution efficiency.

B. Mathmatical Model

According to the above problem assumptions, we model the PDPTW as an integer programming in the following:

$$\min \quad m \tag{1}$$

$$\min \quad \sum_{k\in M}\sum_{i\in V}\sum_{j\in V} d_{ij}x_{ijk} \tag{2}$$

$$\min \quad \sum_{k\in M}\sum_{i\in V} \max\{e_i - a_i, 0\} y_{ik} \tag{3}$$

$$s.t. \quad \sum_{k\in M} y_{ik} = 1, \ \forall i \in V \tag{4}$$

$$\sum_{k\in M}\sum_{j\in V} x_{ijk} = 1, \ \forall i \in V \tag{5}$$

$$\sum_{i\in V} x_{i0k} = 1, \ \forall k \in M \tag{6}$$

$$\sum_{j\in V} x_{0jk} = 1, \ \forall k \in M \tag{7}$$

$$\sum_{i\in V} x_{ihk} - \sum_{j\in V} x_{hjk} = 0, \ \forall h \in V, k \in M \tag{8}$$

$$\sum_{i\in V} x_{ijk} Q \geq z_j, \ \forall j \in V, k \in M \tag{9}$$

$$(y_i - y_j + q_i)x_{ijk} = 0, \ \forall i, j \in V \bigcup \{0\}, k \in M \tag{10}$$

$$(d_i + t_{ij} - d_j)x_{ijk} \leq 0, \ \forall i, j \in V \bigcup \{0\}, k \in M \tag{11}$$

$$d_p \leq d_q, \ \forall i \in V, p = V_i^1, q = V_i^2 \tag{12}$$

$$z_0 = 0, \ d_0 = 0 \tag{13}$$

$$x_{ijk} \in \{0,1\}, y_{ik} \in \{0,1\}, z_i \geq 0, \forall i, j \in V \bigcup \{0\}, k \in M \tag{14}$$

Where (1)-(3) are three objective functions. The objective function (1) minimizes the total number of vehicles, the objective function (2) minimizes the total travel distances, and the objective function (3) minimizes the total waiting time of the vehicles. Constraints (4) represent that each request is assigned to only one vehicle. Constraints (5) guarantee that each job is visited exactly once. Constraints (6) and (7) each vehicle starts from and ends at the same depot. Constraints (8) ensure the flow balance. Constraints (9)-(10) are the capacity constraints. Constraints (11)-(12) represent the time windows and precedence constraints. Constraints (13) are used to initialize the problem. Constraints (14) are decision variables constraints.

III. HYBRID PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) algorithm is a kind of intelligent optimization algorithm based on group evolution technology. It is proposed in 1995 by Kennedy and Eberhart [20]. This algorithm simulates the cluster collaborative behavior of birds, fish and other organisms to achieve global optimization. It is simple, efficient and robust to solve a lot of practical problems. To solve the vehicle routing problem, traditional PSO determine the search direction using the individual and group information. Due to less social information available, the algorithm is easy to fall into local optimum.

Under the basic framework of PSO, this paper presents a Hybrid PSO (HPSO) to solve the multi-objective PDPTW. This algorithm will add particles neighbor information to diversify the particle swarm and adopt the Variable Neighborhood Search (VNS) to enhance the convergence speed.

A. The Algorithm Steps of HPSO

We assume that the dimension of the objective search space is D, each particle i consists of four D dimensional vectors: position vector $\vec{X}_i = (X_{i1}, X_{i2}, \cdots, X_{iD})$, speed vector $\vec{V}_i = (V_{i1}, V_{i2}, \cdots, V_{iD})$, best position vector of the particle iterative search $\vec{P}_i = (P_{i1}, P_{i2}, \cdots, P_{iD})$, and position vector of the optimal nearest neighbor particles $\vec{P}'_i = (P'_{i1}, P'_{i2}, \cdots, P'_{iD})$. The position of the optimal nearest neighbor particles is obtained by calculating the maximum Fitness Distance Ratio (FDR). FDR is calculated in the following:

$$FDR(i, j, d) = \frac{Z(\vec{X}_i) - Z(\vec{P}_j)}{\left| X_{id} - P_{jd} \right|} \tag{15}$$

where $Z(X)$ is the fitness of vector X.

Let the position vector P_g of the $g-th$ particle be optimal position of all P_i ($i = 1,2,\cdots,S$) which corresponds to global optimal solution of the entire particle swarm. The movement of each particle is determined by its current information, historical experiences, the entire group and social learning (i.e. nearest neighbor particles information).

Particle velocity and position update is calculated as follows:

$$\begin{aligned} V_{id}(t+1) = {} & w(t)V_{id} + c_1 r_{1d}(t)[P_{id}(t) - X_{id}(t)] \\ & + c_2 r_{2d}(t)[P_{gd}(t) - X_{id}(t)] \\ & + c_3 r_{3d}(t)[P'_{id}(t) - X_{id}(t)] \end{aligned} \tag{16}$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \tag{17}$$

$$i = 1,2,\cdots,S, \ d = 1,2,\cdots,D$$

where $w(t)$ is inertia factor at iteration t, $c_1, c_2, c_3 > 0$ denote the acceleration factors, $r_{1d}(t), r_{2d}(t), r_{3d}(t)$ are random numbers in [0, 1] at iteration t.

The detailed steps of the HPSO for solving PDPTW are in the following.

Step 1: Initialize the particle swarm

Construct a swarm consisting of S particles. Generate the initial population of the particles using MPNS-

GRASP [21]. Let velocity $\vec{V}_i = 0$, best position $\vec{P}_i = \vec{X}_i$, iteration counter t =1, the counter of solution not improvement t_0 =0.

Step 2: Decoding and evaluation of the particle

For each particle $i = 1,2,\cdots,S$, decode $\vec{X}_i$ into the set of routes R_i. And calculate the fitness $Z(\vec{X}_i)$ of $\vec{X}_i$ to evaluate each particle.

Step 3: Improvement of the particle solution

Let $rnd \sim (0,1)$ for each particle $i = 1,2,\cdots,S$. If $rnd < p_{VNS}$ (The probability of random selection), variable neighborhood search (VNS) algorithm is used to improve the routes in R_i. If the solution is improved, update the individual historical optimal solution.

Step 4: Update statistic information of each particle

4.1. Update the particle historical optimal solution, if $Z(\vec{X}_i) < Z(\vec{P}_i)$, then $\vec{P}_i = \vec{X}_i$;

4.2. Update global optimal solution of the swarm, if $Z(\vec{P}_i) < Z(\vec{P}_g)$, then $\vec{P}_g = \vec{P}_i$, t_0 =1, else $t_0 = t_0 + 1$;

Step 5: For each particle $i = 1,2,\cdots,S$, according to equations (15), search the position $\vec{P}_i'$ of the optimal nearest neighbor particles.

Step 6: Update the movement velocity and position of each particle

Movement velocity and position of each particle is calculated by equations (16) and (17). When their values exceed the limit, we take the boundary values.

Step 7: The algorithm termination condition and results output

If $t < T$ and $t_0 < T_0$ (T_0 is the maximum number of solution not improvement), then $t = t + 1$, go to step 2, else the termination condition is reached, output global optimal solution and objective values.

B. The Evaluation of the Particle

In this paper, PDPTW is an multi-objective problem. According to the importance of the objective functions, the objectives of the PDPTW are to minimize the number of vehicles, total travel distances and total waiting time of the vehicles. The integrated objective function is in the following:

$$\min Z(x) = \alpha_1 N(x) + \alpha_2 D(x) + \alpha_3 W(x) \tag{18}$$

where $N(x), D(x), W(x)$ denote the number of vehicles, total distances and total waiting time of the particle x; $\alpha_1, \alpha_2, \alpha_3$ are coefficients, and $\alpha_1 >> \alpha_2 >> \alpha_3$. The fitness of the evaluation function is:

$$Fit(x) = Z(x) / Z_{\min} \tag{19}$$

where $Z(x)$ is the objective function of the particle x, $Z_{\min}$ is the minimum value of the objective functions in the current particles swarm.

C. Variable Neighborhood Search

To improve the convergence speed, the variable neighborhood search is used to improve the solution. Recently, some scholars have applied the VNS to solve many combinatorial problems and achieved good performance [22][23]. Preliminary experiments show that if the individual optimal solution of each particle is improved by VNS, the diversity of the group particles will be reduced. So, the proposed VNS is applied at each particle of the current swarm with a certain probability p_{VNS} in the implementation process.

Here, we use six neighborhood operators [24]. They are Shift 1-0, Shift 1-1, Cross, 2-opt*, Reverse and Exchange. Shift 1-0, Shift 1-1, Cross and 2-opt* are called inter-route neighborhood operators. Reverse and Exchange are intra-route neighborhood operators. The depots are denoted as 0 and 1, 2… n represent customer nodes. Implementation details concerning the six neighborhood operators are provided as follows.

(1) Shift 1-0, this is inter-route customer movement. It also called Swap/Shift movements. Shift 1-0 represents that a client from one route is inserted into another route. For example, the customer node 3 is moved from a route (0123450) to the other route (067890) before customer 8, which generates the route (01245067890).

(2) Shift 1-1, that is, permutation between a customer from one route and a customer from another route, e.g. client 3 from the route (0123450) and customer 8 from the route (067890) are swapped to produce the obtained route (0128450673 90).

(3) Cross. The arc of both adjacent customers from one route and the one from another route are both removed, and then the two routes are cross-connected. For example, customers 2, 3 from route (0123450) are cross-connected with customers 7, 8 from route (067890) and we obtain the route (012890673450).

(4) 2-opt*. It is the extension of Shift 1-1. This operator can realize permutation between tail clients from one route and tail customers from another route. For example, some customers 3, 4, 5 from route (0123450) are exchanged with customer 9 from route (067890) and the resulted route is (012906783450).

(5) Reverse. This also is one of the intra-route neighborhoods. This movement reverses the sub-route direction. For example, the customers 2, 3, 4 from route (012345670) had their direction reversed and obtain a route (014325670).

(6) Exchange. This neighborhood belongs to intra-route neighborhoods. It's an intra-route version of shift 1-1. Two customers from one route are permutated. For example, the customers 2 and 6 from a route (012345670) are swapped and the new generated route is (016345270).

For these six neighborhood operators, the vehicle load and time windows must be verified by its capacity and time constraints. The computational complexity of the neighborhood operators Shift 1-0, Shift 1-1, Cross, 2-opt*, and Exchange is $O(n^3)$, while the complexity of the neighborhood operator Reverse is $O(n^2)$.

The implementation of the VNS is to find the local optimal solution of the current solution. So, further exploration should be carried out by integrating these neighborhood operators. The inter-route operators could find new improvement to current solution, and intra-route operators optimize further the solution. The execution sequence of the neighborhood operators will affect the performance of the algorithm. After some preliminary experiments, the execution sequence of the neighborhood operators is Shift 1-0, Shift 1-1, Cross, 2-opt*, Exchange and Reverse.

Our proposed VNS algorithm is shown in the following.

Step 1: Initialize the counter $k = 1$

Step 2: If $k \le 4$, then repeat the following steps

2.1. Apply k inter-route neighborhood operators to current solution.

2.2. If the current solution is improved, then perform the intra-route operators Exchange and Reverse in sequence, and update the best solution, let $k = 1$, go to Step 2.1.

2.3. If the current solution is not improved, then let $k = k + 1$; if $k > 4$, then go to Step 3, else go to Step 2.1.

Step 3: Update individual historical optimal solution of the particle using the best solution produced by VNS.

IV. Numerical Experiments

The efficiency and feasibility of the proposed HPSO algorithm would be demonstrated by the following numerical example in this section. The HPSO algorithm described in the previous sections is coded in Matlab 7.0 and executed on a PC equipped with 1.92G SDRAM and a P8400 processor running at 1.58GHz.

We select a PDPTW benchmark instances as testing data. The experimental problem has 100 customers with 56 instances from the website http://www.sintef.no/Projectweb/ToP/Problems/PDPTW. The performance of the proposed HPSO is affected by some parameters, which decide the solution quality of our algorithm. Using preliminary experiments, the related parameters are set in the following: T =500, T_0 =30, S =50, c_1 =0.7, c_2 =0.3, c_3 =1.5, p_{VNS} =0.3.

In previous literatures, to the best of our knowledge, only the tabu-embedded simulated annealing algorithm (denoted by TSAA) proposed by Li and Lim [14] has the similar objectives with our research. And the objectives of other algorithms only considered minimizing the number of vehicles or total travel distances. They are not directly comparable. Therefore, the results of HPSO will compare with TSAA.

Table 1 and table 2 show the results comparisons of our proposed HPSO and TSAA. The average results of these two algorithms are compared in table 1 for each instance, and the average improvement values of our proposed HPSO compared with TSAA on various objectives are calculated. In table 1, ANV, ATD, ATW, and ACPU are average values of number of vehicles, total travel distances, total waiting time and computation time (second) of the algorithms respectively. The bold figures represent the results of HPSO are better than that of TSAA. The TSAA is computed on the computer with Linux Kernel 2.2.15-5.0smp on i686.

From table 1, we can see that the solution quality of HPSO and TSAA is similar only for instances LC2 and LR1. Except these two instances, the solution quality of HPSO is better than the one of TSAA for other instances. Especially, the average total waiting time is improved significantly using the proposed HSPO.

In table 2, NV, TD, TW, and CPU denote the number of vehicles utilized, total travel distances, total waiting time and computation time (second) respectively. From table 2, it shows that by using HPSO there are seven instances generating better solutions over TSSA (the better solutions are bolded). For the first objective, there are three instances obtaining better solutions. For the second objective, there are four instances. And for the third objective, there are six instances. For other instances, our proposed HPSO mostly generates the same results compared with the algorithm of TSSA. In a word, the proposed HPSO can find new best-so-far solutions for nearly all instances except LC2.

TABLE I.
Comprehensive comparisons of solution results of TSAA and HPSO

Instances	TSAA				HPSO				Improvement(%)		
	ANV	ATD	ATW	ACPU	ANV	ATD	ATW	ACPU	INV	ITD	ITW
LC1	9.89	832.09	42.12	225.56	**9.75**	850.88	**27.35**	55.21	-1.44	2.21	-54.00
LC2	3.00	9609.74	31.77	196.25	3.00	9610.56	**20.78**	120.66	0	0.01	-52.89
LR1	11.92	1222.20	245.54	371.08	11.92	1222.20	245.54	37.25	0	0	0
LR2	2.73	973.87	478.60	1876.36	2.73	**970.79**	**467.72**	314.17	0	-0.32	-2.32
LRC1	11.63	1387.60	149.62	261.25	**11.47**	**1358.26**	**137.26**	35.65	-1.39	-2.16	-9.00
LRC2	3.25	1187.82	549.15	1536.13	3.25	**1137.94**	567.16	210.73	0	-4.38	3.18

TABLE Ⅱ.
EXPERIMENTAL RESULTS OF TSAA AND HPSO

No.	TSAA				HPSO				No.	TSAA				HPSO			
	NV	TD	TW	CPU	NV	TD	TW	CPU		NV	TD	TW	CPU	NV	TD	TW	CPU
Lc101	10	828.94	0	33	10	828.94	0	15	Lr201	4	1263.84	1231.97	193	4	**1237.64**	**1185.73**	137
Lc102	10	828.94	0	71	10	828.94	0	32	Lr202	3	1197.67	551.73	885	3	1197.67	551.73	287
Lc103	10	827.86	230.17	191	**9**	1013.25	**90.31**	38	Lr203	3	949.40	813.40	1950	3	949.40	813.40	272
Lc104	9	861.95	144.95	1254	9	863.87	**138.75**	70	Lr204	2	849.05	139.52	2655	2	849.05	139.52	588
Lc105	10	828.94	0	47	10	828.94	0	25	Lr205	3	1054.02	496.11	585	3	1054.02	496.11	230
Lc201	3	591.56	0	27	3	591.56	0	18	Lrc101	14	1708.80	247.52	119	14	1708.80	247.52	23
Lc202	3	591.56	0	94	3	591.56	0	73	Lrc102	13	1563.55	200.59	152	**12**	**1558.07**	**145.85**	39
Lc203	3	585.56	26.09	145	3	**573.29**	**18.50**	124	Lrc103	11	1258.74	184.95	175	11	1258.74	184.95	35
Lc204	3	591.17	0	746	3	591.17	0	231	Lrc104	10	1128.40	109.77	202	10	1128.40	109.77	51
Lc205	3	588.88	0	190	3	588.88	0	76	Lrc105	13	1637.62	192.54	179	13	1637.62	192.54	22
Lr101	19	1650.78	948.65	87	19	1650.78	948.65	58	Lrc201	4	1468.96	889.45	266	4	**1389.57**	927.95	98
Lr102	17	1487.10	714.89	1168	**15**	1530.58	**679.21**	41	Lrc202	3	1374.27	282.87	987	3	1374.27	282.87	176
Lr103	13	1292.68	436.48	169	13	1292.68	436.48	36	Lrc203	3	1089.07	611.23	1605	3	1089.07	611.23	188
Lr104	9	1013.39	37.46	459	9	1013.39	37.46	66	Lrc204	3	827.78	710.05	3634	3	827.78	710.05	457
Lr105	14	1377.11	254.45	69	14	1377.11	254.45	23	Lrc205	4	1302.20	1162.41	639	4	1302.20	1162.41	176

For the computational speed of the algorithm, the average computational time of HPSO and TSSA solving all the instances are listed in table 1 and table 2. However, due to the different testing environment of the two algorithms, the computational speed of the two algorithms can not be directly comparable. By overall analysis, the computational time of the proposed HPSO is acceptable.

From the above implementation of the algorithm, it is clear that the proposed HPSO is effective and efficient to decide on optimal pickup and delivery scheduling problem, and provide a best solution to meet the multi-objectives demands.

V. CONCLUSION

PDPTW is an famous problem which can model a lot of practical application in logistics and transportation. Nowadays, decision-makers tend to consider multi-objectives in real applications with PDPTW. In this paper, we describe the multi-objective pickup and delivery problem with time windows. The previous researches ignored the objective considering total waiting time. So, this paper establishes a mixed integer programming model for multi-objective PDPTW. Then, a hybrid PSO is proposed to solve this problem. In the algorithm, particles neighbor information is used to diversify the particle swarm, and the variable neighborhood search is adopted to enhance the convergence speed. At last, the algorithm is implemented and programmed with Matlab 7.0. The numerical experiments using benchmark instances show the proposed HPSO has higher solutions quality.

ACKNOWLEDGMENT

The authors wish to thank the reviewers for their valuable comments. This work was supported in part by the Contemporary Business and Trade Research Center of Zhejiang Gongshang University which is the Key Research Institute of Social Sciences and Humanities Ministry of Education (Grant No.12JDSM16YB), Humanities and Social Sciences Foundation of Ministry of Education of China (Grant No. 12YJC630091), Zhejiang Provincial Natural Science Foundation of China (Grant No. LQ12G02007), and Zhejiang Provincial Commonweal Technology Applied Research Projects of China (Grant No. 2013C33030).

REFERENCES

[1] G.B. Dantzig, J.H. Ramser, “The truck dispatching problem,” *Management Science*, vol. 6, no.1, pp.80-91, 1959.

[2] J. Li, “Research on team orienteering problem with dynamic travel times,” *Journal of Software*, vol.7, no. 2, pp.249-255, 2012.

[3] J. Li, P.H. Fu, “A label correcting algorithm for dynamic tourist trip planning,” *Journal of Software*, vo.7, no. 12, pp.2899-2905, 2012.

[4] Y.Z. Wang, “Constraint cellular ant algorithm for the multi-objective vehicle routing problem,” *Journal of Software*, vol.8, no.6, pp.1339-1345, 2013.

[5] C.Y. Ren, "Research on single and mixed fleet strategy for open vehicle routing problem," *Journal of Software*, vol.6, no.10, pp. 2076-2081, 2011.
[6] J. Li, "Vehicle routing problem with time windows for reducing fuel consumption," *Journal of Computers*, vol. 7, no.12, pp.3020-3027, 2012.
[7] J. Li, "Model and simulation for collaborative VRPSPD," *Journal of Networks*, vol.8, no.2, pp.331-338, 2013.
[8] B. Rekiek, A. Delehambre, H.A. Saleh, "Handicapped person transportation: An application of the grouping genetic algorithm," *Eng Appl Artif Intel*, no.19, pp.511-520, 2006.
[9] K. Fagerholt, M. Christiansen, "A combined ship scheduling and allocation problem," *Journal Operation Research Society*, no.51, pp.834-842, 2000.
[10] A.G. Najera, J.A. Bullinaria. "An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows," *Computers & Operations Research*, no.38, pp.287-300, 2011.
[11] C.L. Hoong, L. Zhe, "Pickup and delivery with time windows : algorithms and test case generation," *International Journal on Artificial Intelligence Tools*, vol.11, no.3, DOI: 10.1142/S0218213002000988, 2002.
[12] H.F. Wang, Y.Y. Chen, "A genetic algorithm for the simultaneous delivery and pickup problems with time window," *Computers & Industrial Engineering*, 2012, vol.62, no.1, pp.84-95.
[13] Q. Lu, M.M. Dessouky, "A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows," European Journal of Operational Research, 2006, vol.175, no.2, pp.672-687.
[14] H. Li, A. Lim, "A metaheuristic for the pickup and delivery problem with time windows," 13th IEEE International Conference on Tools with Artificial Intelligence, IEEE Computer Society, LosAlamitos, CA, 2001, pp. 333-340.
[15] G. Pankratz, "A grouping genetic algorithm for the pickup and delivery problem with time windows," *Operation Research Spectrum*, 2005, vol.27, pp.21-41.
[16] H.F. Wang, Y.Y. Chen, "A coevolutionary algorithm for the flexible delivery and pickup problem with time windows," *International Journal of Production Economics*, 2013, vol. 141, no. 1, pp.4-13.
[17] M.I. Hosny, C.L. Mumford, "Constructing initial solutions for the multiple vehicle pickup and delivery problem with time windows," *Journal of King Saud University - Computer and Information Sciences*, 2012, vol.24, no.1, pp.59-69.
[18] Y. Qu, J.F. Bard, "A GRASP with adaptive large neighborhood search for pickup and delivery problems with transshipment," *Computers & Operations Research*, 2012, vol.29, no.10, pp.2439-2456.
[19] W.P. Nanry, J.W. Barnes, "Solving the pickup and delivery problem with time windows using reactive tabu search," *Transportation Research Part B: Methodological*, 2000, vol.34, no.2, pp.107-121.
[20] J. Kennedy, R.C. Eberhart, "Particle swarm optimization," In Proceeding of the IEEE international conference on neural network, 1995, no.4, pp.1942-1948.
[21] Y. Marinakis, M. Marinaki, G. Dounias, "A hybrid particle swarm optimization algorithm for the vehicle routing problem," Engineering Applications of Artificial Intelligence, 2010, vol.23, pp.463-472.
[22] Q.K. Pan, M.F. Tasgetiren, Y.C. Liang, "A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem," *Computers & Operation Research*, 2008, vol.35, no.9, pp.2807-2839.
[23] Q. Kang, H. He, "A novel discrete particle swarm optimization algorithm for meta-task assignment in heterogeneous computing systems," *Microprocessors and Microsystems*, 2011, no.35, pp.10-17.
[24] A. Subramanian, L.M.A. Drummond, C. Bentes, et al., "A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery," *Computers& Operations Research*, vol.37, pp.1899–1911, 2010.