Int. J. Production Economics 141 (2013) 4–13

Contents lists available at SciVerse ScienceDirect

Int. J. Production Economics

journal homepage: www.elsevier.com/locate/ijpe

A coevolutionary algorithm for the flexible delivery and pickup problem with time windows

Hsiao-Fan Wang *, Ying-Yen Chen

Department of Industrial Engineering and Engineering Management, National Tsing Hua University, No. 101, Section 2, Kuang Fu Road, Hsinchu, 30013 Taiwan, ROC

ARTICLE INFO

Article history:
Received 19 September 2011
Accepted 20 April 2012
Available online 9 May 2012

Keywords:
Bi-directional logistics
Delivery and pickup
Vehicle routing problem with backhaul
Coevolutionary algorithm

ABSTRACT

This paper addresses a flexible delivery and pickup problem with time windows (FDPPTW) and formulates the problem into a mixed binary integer programming model in order to minimize the number of vehicles and to minimize the total traveling distance. This problem is shown to be NP-hard. In this study, therefore, a coevolutionary algorithm incorporated with a variant of the cheapest insertion method is developed to speed up the solution procedure. The FDPPTW scheme overcomes the shortcomings of the existing schemes for the delivery and pickup problems. By testing with some revised Solomon's benchmark problems, the computational results have shown the efficiency and the effectiveness of the developed algorithm.

1. Introduction

Due to the awareness of environmental protection, many companies have introduced remanufacturing, recovering, and recycling operations for transforming trash into money and saving the environment from damage. Therefore the reverse logistics dealing with the returned flows have drawn much attention of enterprises and researchers. Recently, many enterprises have incorporated the reverse logistics into the regular forward logistics to form a closed-loop supply chain (Wang and Hsu, 2010). A state of the art survey of reverse and close-supply chains can be found in Ilgin and Gupta (2010). Within such a loop, the logistics between the distribution/collection center and the customers is the most complicated part because it is related to the bi-directional logistics regarding delivery and pickup activities.

In the literatures, such problems have been referred to as the delivery and pickup problems (DPP). DPP applications are frequently encountered, for example, in the distribution system of grocery store chains. Each grocery store may have a demand for both delivery (cf. fresh food or soft drinks) and pickup (cf. outdated items or empty bottles). The foundry industry is another example in Dethloff (2001). The collection of the used sand and the delivery of the purified reusable sand at the same customer location are carried out.

In order to achieve low carbon emission and high resource productivity, enterprises need to incorporate the reverse logistics into the regular forward logistics to perform both delivery and pickup. One way to reduce both the carbon emissions toward the environment and the operational cost of an enterprise is to lower the total traveling distance and the number of vehicles. This win–win situation benefits the enterprises, the governments, and the human beings; and consequently, more efficient and effective bi-directional logistics are desired. Based on this requirement, a flexible delivery and pickup problem with time windows (FDPPTW) is proposed in this study to facilitate the operations and management.

This paper is organized as follows. In Section 2, the literatures related to the issues in interest are reviewed. In Section 3, after formally defining the FDPPTW, a mathematical model for the FDPPTW is proposed. To facilitate effective applications, Section 4 gives a detailed description of the developed coevolutionary genetic algorithm. Section 5 provides the computational results with the evaluation on the accuracy and efficiency of the developed algorithm. Finally, the conclusions are drawn in Section 6.

2. Related work

Delivery and pickup problems (DPP) are extensions to the vehicle routing problem (VRP) where the vehicles are not only required to deliver goods to customers but also to pick some goods up at customer locations. In the general DPP, two types of customers are served from a single depot by a fleet of vehicles. The first type of customers is known as "linehaul" customers, who require deliveries of their goods to the specific locations. The second type is known as "backhaul" customers, who require

* Corresponding author. Tel.: +886 3 5742654; fax: +886 3 5722204.
E-mail addresses: hfwang@ie.nthu.edu.tw, hfwang82@hotmail.com (H.-F. Wang).

0925-5273/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ijpe.2012.04.011

pickups from their specific locations. A survey of the DPP can be referred to Parragh et al. (2008).

To carry out the jobs of DPP, there are three main strategies which have been developed into three categories of the problems: (1) delivery-first, pickup-second for the Vehicle Routing Problem with Backhauls (VRPB); (2) mixed deliveries and pickups for the Mixed Vehicle Routing Problem with Backhaul (MVRPB); and (3) simultaneous deliveries and pickups for the Simultaneous Delivery and Pickup Problem (SDPP). They are explained briefly below:

(1) Delivery-first, pickup-second strategy: vehicles can only pick up goods after they have finished delivering their entire loads (e.g. Ropke and Pisinger, 2006). This strategy facilitates the implementation because accepting pickups before finishing all deliveries may cause the vehicle to be overloaded during its trip (even if the total delivery and the total pickup loads are not above the vehicle capacity), resulting in an infeasible vehicle tour. However, visiting a customer twice would increase the traveling and operation costs.
(2) Mixed deliveries and pickups strategy: linehauls and backhauls can occur in any sequence on a vehicle route (e.g. Wade and Salhi (2002), Nagy and Salhi (2005), Crispim and Brandao (2005) and Tütüncüa et al. (2009)). This strategy releases the constraints that pickups are only accepted after finishing all deliveries and makes the operation more efficient. However, if a customer requires both delivery and pickup services, this strategy may cause twice accesses and increase the traveling and operation costs.
(3) Simultaneous pickups and deliveries: to facilitate both delivery and pickup services required by a customer, single access to the customer is performed. This is a further improvement from the previous strategies, yet the simultaneous pickups and deliveries is the only choice. In some literature, the SDPP was called the Vehicle Routing Problem with Simultaneous Delivery and Pickup (VRPSDP). Because of its efficiency in handling both services, many researchers have devoted to this problems. For instance, Min (1989), Dethloff (2001), Nagy and Salhi (2005), Crispim and Brandao (2005), Chen and Wu (2006), Dell'Amico et al. (2006), Montané and Galvao (2006), Berbeglia et al. (2007), Bianchessi and Righini (2007) and Ai and Kachitvichyanukul (2009).

In order to provide more satisfactory services, nowadays, enterprises have allowed customers to request their goods being delivered or picked up within specific time windows. Such consideration extends the problems above into VRPB and Time Windows (VRPBTW), MVRPB and Time Windows (MVRPBTW) and SDPP with Time Windows (SDPPTW), respectively. Since such extension increases the complexit of the problems, therefore researchers have devoted to developing efficient algorithms for finding good feasible solutions. For instance, Kontoravdis and Bard (1995) developed a greedy randomized adaptive search procedure to solve the MVRPBTW. Zhong and Cole (2005) developed a guided local search heuristic to solve both the VRPBTW and the MVRPBTW. Angelelli and Mansini (2002) developed a branch and price algorithm for the small-scale SDPPTW. Wang and Chen (2012) developed a coevolutionary algorithm for the SDPPTW.

Due to the advantage of flexible delivery and pickup with MVRPBTW, further improvement on reducing the operation cost has been carried out. One issue is how to reduce the accessing time when a simultaneous delivery and pickup at the same customer location occurs. Dethloff (2001), Chen and Wu (2006), Montané and Galvao (2006) have suggested that the accessing time can be reduced by performing a simultaneous delivery and pickup. This possibility is adopted and evaluated by this study, of which a new model will be developed to remain the flexibility of mixing pickup and delivery operations while time saving from simultaneously performing delivery and pickup is realized. We call this kind of problems to be the Flexible Delivery and Pickup Problem with Time Windows (FDPPTW).

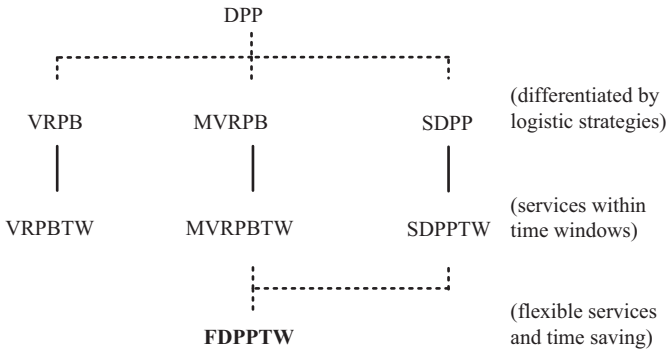
FDPPTW will be discussed in Section 3, and the relation of the reviewed problems above are summarized in Fig. 1.

It is known that the vehicle routing problem with time windows (VRPTW) is NP-hard (Solomon, 1987). The VRPTW is polynomial time reducible from the FDPPTW by setting all pickup demands equal to zero; therefore the FDPPTW is also NP-hard. As a result, an efficient and effective solution procedure is needed for the FDPPTW. The methods dedicated to the VRPBTW, the MVRPBTW, and the SDPPTW are the most relevant. However, most of these methods were only useful to find practicable solutions in a short time. When the computational time lasts longer, they did not show themselves to have significant improvement for better solutions. Furthmore, it is noticed that various VRPs reviewed in this section were solved by heuristics or evolutionary algorithms since large-scale VRPs cannot be solved by exact algorithms. Evolutionary algorithms were showed to perform well in VRPs and the coevolutionary algorithm performed even better, see Wang and Chen (2012). Therefore, the coevolutionary algorithm used in Wang and Chen (2012) is revised to solve the FDPPTW in this study. It will be explained in Section 4 in details.

3. Problem formulation

The flexible delivery and pickup problem with time windows (FDPPTW) can be stated as below:

> A set of customers who each require a delivery and/or a pickup of certain quantities within specific time windows, must be served by a fleet of capacitated vehicles which are stationed at a distribution center (DC) and ready to serve within a certain time horizon. The FDPPTW is thus to search for the most economic route for each vehicle with the minimum operational cost.



DPP: Delivery and Pickup Problem
VRPB: Vehicle Routing Problem with Backhauls
MVRPB: Mixed Vehicle Routing Problem with Backhauls
SDPP: Simultaneous Delivery and Pickup Problem
VRPBTW: Vehicle Routing Problem with Backhauls and Time Windows
MVRPBTW: Mixed Vehicle Routing Problem with Backhauls and Time Windows
SDPPTW: Simultaneous Delivery and Pickup Problem with Time Windows
FDPPTW: Flexible Delivery and Pickup Problem with Time Windows

Fig. 1. Related delivery and pickup problems.

For each service (either delivery or pickup) required by any customer, one vehicle will be assigned exactly once and certain service time will be consumed. If both services are required by one customer, he/she can request a delivery time window and a pickup time window.

In a common application of the FDPPTW to a recycling network, for illustration, all vehicles may return to a collection center (CC) to unload the recycled stuff. The infrastructure of the system can be seen in Fig. 2. The black and the white squares indicate the distribution center and the collection center, respectively. The white circles and black triangles indicate linehaul and backhaul customers correspondingly. The solid arrows indicate the movements. A driver would not need to re-access to a customer if he/she picks up stuff right after delivers goods. Therefore, we use a dot arrow to describe that the pickup service for a customer is performed right after the delivery service. Fig. 2 shows that there are five customers (2, 3, 6, 7, and 8) who are served delivery and pickup simultaneously; and the other four customers are served delivery earlier but pickup later.

The FDPPTW has two objectives: minimizing the number of vehicles and minimizing the total traveling distance. Trade-offs between these two kinds of costs are needed to be considered.

Based on the principle of a VRP problem, one customer is visited exactly once by one vehicle for one service. A pseudo customer should be introduced for separating two services required by one customer. Assume there are n customers, each is indicated by customer i, $i=1,\ldots, n$. When modeling, $2n$ customers are generated with n new customer i, $i=1,\ldots, n$, each demanding only a delivery service, and n new customer $n+i$, $i=1,\ldots, n$, each demanding only a pickup service. Assume there are m vehicles. The flexible delivery and pickup problem with time windows is then formulated into a mixed binary integer programming model denoted by Model FDPPTW as below.

3.1. Notations

Sets

J	Set of all customers, $J=j\|j=1,\ldots,2n$
J_D	Set of all delivery customers, $J_D=j\|j=1,\ldots,n$
J_F	Set of all customers plus D.C., $J_F=0\cup J$
J_R	Set of all customers plus C.C., $J_R=J\cup 2n+1$
J_C	Set of all nodes, $J_C=0\cup J\cup 2n+1$
V	Set of all vehicles, $V=v\|v=v_1,\ldots,v_m$

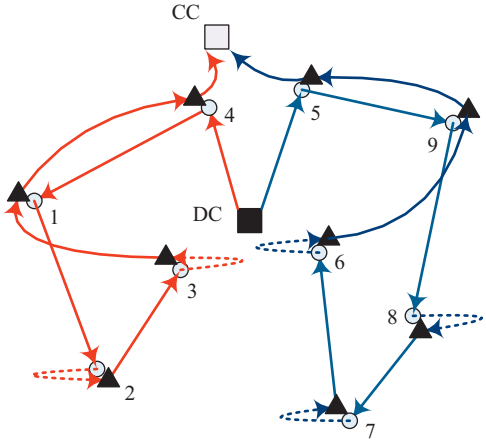


Fig. 2. Infrastructure of the delivery and pickup network.

Coefficients

q_v	Capacity of vehicle v, $q_v\in \boldsymbol{R}^+$
g_v	Dispatching cost of vehicle v, $g_v\in \boldsymbol{R}^+$
c_{ij}	Distance between nodes $i\in J_F, j\in J_R; i\neq j, c_{ij}\in \boldsymbol{R}^+$
t_{ij}	Traveling time between nodes $i\in J_F, j\in J_R; i\neq j, t_{ij}\in \boldsymbol{R}^+$
d_j	Delivery demand of customer $j\in J, d_j\in \boldsymbol{Z}^+$
p_j	Pickup demand of customer $j\in J, p_j\in \boldsymbol{Z}^+$
s_j	Service time of customer $j\in J, s_j\in \boldsymbol{R}^+$
r_j	Reduced accessing time if the delivery and pickup services of customer j are performed simultaneously, $j\in J_D$
a_j	Earliest service time of customer $j\in J, a_j\in \boldsymbol{R}^+$
b_j	Latest service time of customer $j\in J, b_j\in \boldsymbol{R}^+$
a_0	Earliest departure time of any vehicle from D.C., $a_0\in \boldsymbol{R}^+$
b_{2n+1}	Latest arrival time that a vehicle must return C.C., $b_{2n+1}\in \boldsymbol{R}^+$
M	An arbitrary large constant
α	A parameter indicating the trade-off between dispatching cost and traveling cost, $\alpha\in[0,1]$

Decision variables

L_{0v}	Load of vehicle $v\in V$ when leaving D.C., $L_{0v}\in \boldsymbol{Z}^+$
L_j	Remaining load of a vehicle after having served customer $j\in J, L_j\in \boldsymbol{Z}^+$
x_{ijv}	Traveling variable of a vehicle $v\in V, x_{ijv}\in 0,1$; if it travels directly from node $i\in J_F$ to node $j\in J_R, x_{ijv}=1$; otherwise $x_{ijv}=0$
T_j	Time to begin service at customer $j\in J, T_j\in \boldsymbol{R}^+$
T_{0v}	Departure time of vehicle $v\in V$ at D.C., $T_{0v}\in \boldsymbol{R}^+$
$T_{(2n+1)v}$	Arrival time of vehicle $v\in V$ at C.C., $T_{(2n+1)v}\in \boldsymbol{R}^+$

3.2. Model FDPPTW

$$\text{Minimize } z=\alpha\sum_{v\in V}\sum_{j\in J}g_v x_{ojv}+(1-\alpha)\sum_{i\in J_F}\sum_{j\in J_R}\sum_{v\in V}c_{ij}x_{ijv} \tag{1}$$

subject to

$$\sum_{i\in J_F}\sum_{v\in V}x_{ijv}=1 \quad \forall j\in J \tag{2}$$

$$\sum_{i\in J_F}x_{ihv}=\sum_{i\in J_R}x_{hjv} \quad \forall h\in J, \forall v\in V \tag{3}$$

$$\sum_{j\in J}x_{0jv}=\sum_{i\in J}x_{i(2n+1)v} \quad \forall h\in J, \forall v\in V \tag{4}$$

$$L_{0v}=\sum_{i\in J_F}\sum_{j\in J}d_j x_{ijv} \quad \forall v\in V \tag{5}$$

$$L_j\geq L_{0v}-d_j+p_j-M(1-x_{0jv}) \quad \forall j\in J, \forall v\in V \tag{6}$$

$$L_j\geq L_i-d_j+p_j-M\left(1-\sum_{v\in V}x_{ijv}\right) \quad \forall i\in J, \forall j\in J \tag{7}$$

$$L_{0v}\leq q_v \quad \forall v\in V \tag{8}$$

$$L_j\leq q_v+M\left(1-\sum_{i\in J_F}x_{ijv}\right) \quad \forall j\in J, \forall v\in V \tag{9}$$

$$T_j\geq T_{0v}+t_{0j}-M(1-x_{0jv}) \quad \forall j\in J, \forall v\in V \tag{10}$$

$$T_j \geq T_i + s_i + t_{ij} - M\left(1 - \sum_{v \in V} x_{ijv}\right) \quad \forall i \in J, \forall j \in J - n + i \tag{11}$$

$$T_{n+i} \geq T_i + s_i - r_i + t_{i(n+i)} - M\left(1 - \sum_{v \in V} x_{i(n+i)v}\right) \quad \forall i \in J_D \tag{12}$$

$$T_{(2n+1)v} \geq T_i + s_i + t_{i(2n+1)} - M(1 - x_{i(2n+1)v}) \quad \forall i \in J, \forall v \in V \tag{13}$$

$$a_0 \leq T_{0v} \quad \forall v \in V \tag{14}$$

$$a_j \leq T_j \leq b_j \quad \forall j \in J \tag{15}$$

$$T_{(2n+1)v} \leq b_{2n+1} \quad \forall v \in V \tag{16}$$

(Ensure feasibility of the time schedule)

$$x_{ijv} \in 0,1 \quad \forall i \in J_F, \forall j \in J_R, \forall v \in V. \tag{17}$$

Objective function (1) is to minimize the total cost, which includes the total dispatching cost and the total traveling cost. Since these costs are compensated to each other, the trade-off parameter, $\alpha \in [0,1]$, is employed to adjust for different decision criteria. This trade-off parameter α is determined by the decision maker. The most commonly considered objective functions are to minimize the number of vehicles, and to minimize the total distance. In general, minimizing the number of vehicles is the primary objective, whereas minimizing the total distance is the secondary. This can be achieved by setting α close to 1, i.e. $\alpha \to 1$. Constraint (2) ensures that each customer will be visited exactly once by a vehicle. Constraints (3) and (4) ensure the flow conservation 'for each customer h' and 'between the distribution center and the collection center,' respectively.

Constraints (5)–(9) describe the vehicle loading along a route. While Eq. (5) shows the initial load of each vehicle, constraint (6) calculates the vehicle load of each vehicle after finishing the service to its first customer. If the first customer of vehicle v is customer j, which denotes by $x_{0jv} = 1$, then

$$L_j = L_{0v} - d_j + p_j. \tag{18}$$

This implication can be stated as below:

$$x_{0jv} = 1 \Rightarrow L_j = L_{0v} - d_j + p_j \tag{19}$$

Implication (19) can be remodeled as the following constraint:

$$L_j = L_{0v} - d_j + p_j + y_{0jv}(1 - x_{0jv}) \tag{20}$$

where y_{0jv} is a auxiliary variable, $y_{0jv} \in \boldsymbol{R}$. However, this constraint is not linear. In order to get a linear constraint, implication (19) is revised as below:

$$x_{0jv} = 1 \Rightarrow L_j \geq L_{0v} - d_j + p_j \tag{21}$$

Although the consequence in implication (21) is an inequality not Eq. (18), it still preserves the meaning of capacity constraint. Thus, implication (21) can be remodeled into constraint (6) which is linear. This modeling technique also applies to the formation of constraints (7) and (9)–(13).

Constraint (7) calculates the 'en route' vehicle loads. If any vehicle delivers the commodity from customer i to customer j, which denotes by $\sum_{v \in V} x_{ijv} = 1$, then

$$L_j = L_i - d_j + p_j \tag{22}$$

By the technique mentioned above, the implication with Eq. (21) can be rewritten as constraint (7). Constraint (8) ensures that the initial load of each vehicle is below the vehicle capacity, so does the constraint (9) for the 'en route' vehicle loads.

The travel and service time along a route is described in constraints (10)–(16). Constraints (10)–(13) establishes the relationship between the vehicle arrival time to a customer and its immediate predecessor. Constraints (10)–(13) are remodeled from the following implications:

$$x_{0jv} = 1 \Rightarrow T_j \geq T_{0v} + t_{0j} \tag{23}$$

$$\sum_{v \in V} x_{ijv} = 1, \quad j \neq n+i \Rightarrow T_j \geq T_i + s_i + t_{ij} \tag{24}$$

$$\sum_{v \in V} x_{i(n+i)v} = 1 \Rightarrow T_{n+i} \geq T_i + s_i - r_i + t_{i(n+i)} \tag{25}$$

$$x_{i(2n+1)v} = 1 \Rightarrow T_{(2n+1)v} \geq T_i + s_i + t_{i(2n+1)} \tag{26}$$

Constraint (14) ensures that each vehicle never departs from the distribution center before it opens. Constraint (15) is the time window constraint. Constraint (16) ensures that each vehicle never enters the collection center after it closes. Constraint (17) is the binary constraint.

This model contains $4n^2m + 4n + 3m$ variables ($4n^2m$ are binary) and $8n^2 + 8nm + 4n + 5m$ constraints. If the problem scale is not very large, one can solve the problem by implementing this model with Cplex or other software package to get the optimal solution. If $n = 100$ and $m = 20$, it will have 800,000 binary variables and about 100,000 constraints. When the scale of the problem is up to such large, Cplex is difficult to reach an optimal solution. Sometimes it even could not get a feasible solution in a reasonable time. Consequently, a efficent algorithm is needed to produce qualified solutions.

4. Coevolutionary algorithm (CEA)

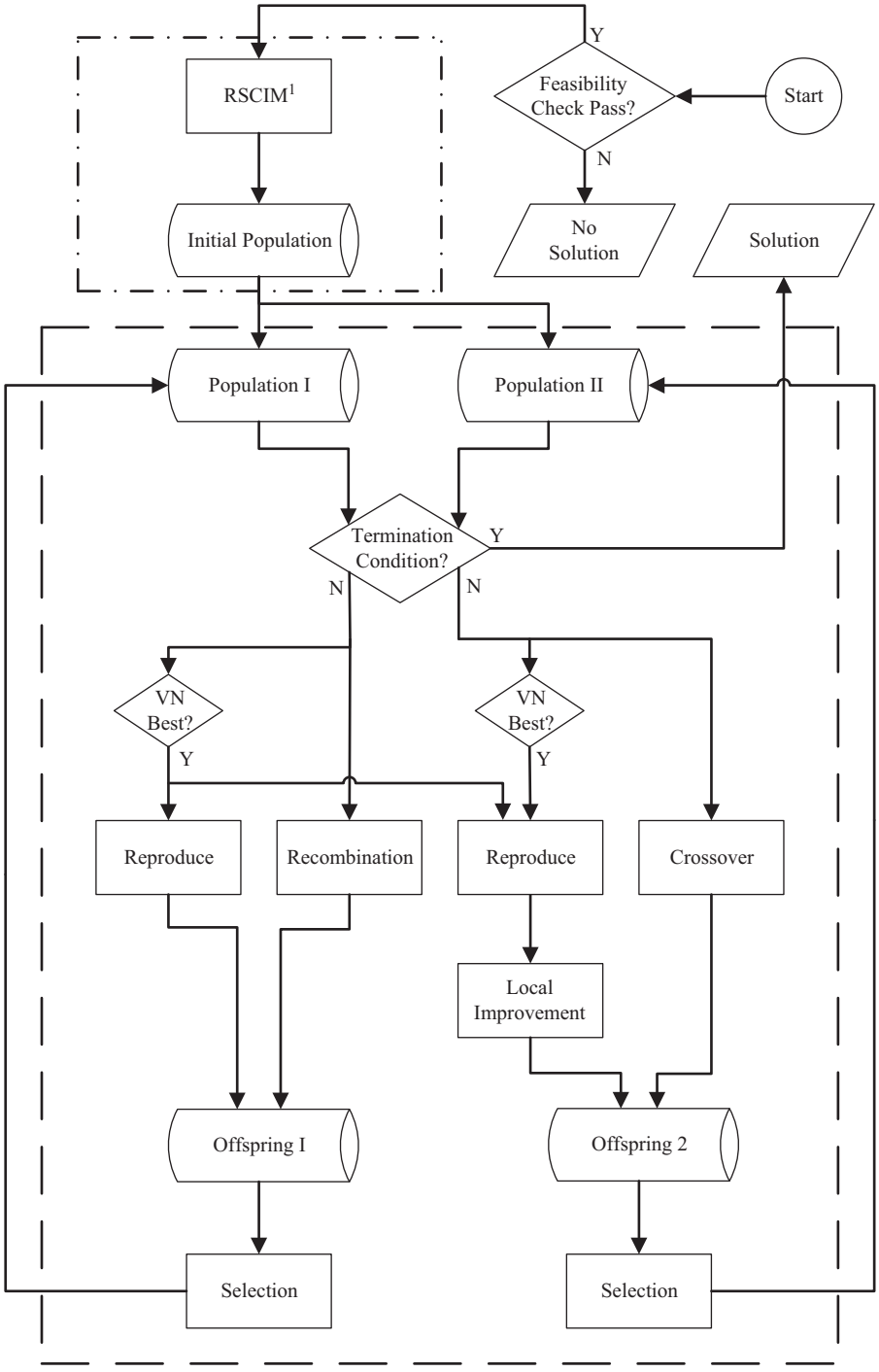
The genetic algorithm (GA) was first proposed by Holland (1975). Due to its global search mechanism, GA has shown its capability to find good solutions for complex mathematical problems, like the VRP and other NP-hard problems, in a reasonable amount of time. The traditional design of a GA faced the dilemma of 'converging too quickly to non-acceptable local optima' or 'converging too slowly and resulting in exhaustive time consumption for deriving an acceptable solution.' The coevolutionary algorithm (CEA) developed in this study could avoid any one of the above situations. It is done by carrying out two separate evolutions simultaneously: Population I is employed for the diversification purpose while Population II is employed for the evolutionary intensification. The framework of this algorithm is shown in Fig. 3.

4.1. Initial population

The first step of a GA is the generation of the initial population. If a fast and simple heuristic procedure can be found to distribute all customers to the vehicles as the first generation of the GA, it can significantly reduce the GA's computational time required to reach the reasonable local minima. In other words, the method used to create initial solutions for a GA should compute as quickly as possible and reveal as many good properties as possible. With this purpose, the cheapest insertion heuristic was frequently used by many researchers. The cheapest insertion heuristic is developed from the savings procedure of Clarke and Wright (1964). For the descriptions of some variants of the cheapest insertion method (CIM), one can refer to Mester et al. (2007) and Osman (1993). In this study, a revised CIM is used to generate the initial population and is named the random seeds cheapest insertion method (RSCIM). The CIM and the RSCIM are introduced in the following subsections.

4.1.1. CIM

The CIM begins with an initial solution in which each customer is served individually by a vehicle, i.e. the number of vehicles = the



1: RSCIM is the method used to generate initial solutions; it will be introduced later.

Fig. 3. Framework of the coevolutionary genetic algorithm.

number of routes = the number of customers. Insertions of single route customers to alternative positions in the solution vector are then attempted in a loop. For a single route customer k, the method considers alternative positions between adjacent customers l and m in other routes. The insertions are evaluated using the cost saving criterion of Osman (1993), $(c_{0k}+c_{k(n+1)}+c_{lm})-(c_{lk}+c_{km})$ where c_{ij} refers to the cost of the associated arc (i, j). Each insertion examines all single route customers; the insertion trial with the maximum cost savings is executed. The insertion procedure is stopped if the algorithm cannot reduce the number of single customer routes.

4.1.2. RSCIM

The initial population of GA needs to spread widely to avoid falling into local optimum in the later evolution. For this reason, the RSCIM was developed with the concept of the random seed customers for route initialization. Instead of beginning with an initial solution in which each customer is supplied individually by a separate route, the RSCIM generates a random order of customers for route growing. Top k customers of the order are the seeds for route growing where k=(Total Demand/Average Vehicle Capacity). By this random order, iteratively, one of the remainder customers will be added to the partial solution vector to form

another single customer route, and then the insertion trail with the maximum cost savings is executed. The addition and insertion procedure is stopped if the algorithm cannot reduce the number of single customer routes. This method can search a wider space for the initial population.

4.2. Co-evolution

The developed CEA has two populations. Population I is employed for diversification purposes, while Population II is employed for evolutionary intensification. The coevolutionary structure of these two populations is illustrated in Fig. 4. Population I aims to retain the wide searching ability through three operators: Reproduction, Recombination and Selection. Population II aims to reach high quality solutions rapidly and improve them constantly through four operators: Reproduction, Local Improvement, Crossover, and Selection. Besides, the best chromosome of Population I is added into Population II at each generation. This provides endless possibilities that Population II can get rid of local optimal solutions. The "simultaneous evolution of diversification and intensification" and the "continuous forwarding of innovativeness" are the key factors to avoid converging to non-acceptable local optima or resulting in exhaustive time consumption.

If the population size N is used, the N initial solutions that RSCIM generates are copied directly into Population I and Population II. Evolution of each population is described in the following sections. In both populations, N parents generate $2N$ offspring and then these $2N$ offspring compete with each other for only N to survive as the parents of the next generation. In Population I, the Reproduction and Recombination operators are used to generate $2N$ offspring from N parents. In Population II, the Reproduction, Local Improvement and Crossover operators are used to generate $2N$ offspring from N parents. In both populations, the Selection operator is used to select N parents of the next generation from $2N$ offspring.

4.2.1. Reproduction

In both populations, the reproduction operator copies the best parents to form the first offspring. The best individual is the one with minimal objective value. Reproducing to keep the best individual is also known as the Elitism strategy, which guarantees that CEA never retreats from a high quality solution.

4.2.2. Recombination

The recombination operator is a *remove–insert mechanism* which preserves the wide searching ability of the developed CEA. In the first step, it randomly removes $1/2 \sim 1/10$ of customers from their routes. Then, the reinsertion of isolated customers is done by RSCIM. The existing routes are regarded as seed routes.

4.2.3. Local Improvement

Two types of Local Improvement are used in this work: Reinsertion Improvement and Swap Improvement. Either one of these two kinds of improvements can be used to improve the offspring prototypes which the Reproduction operator generates.

Reinsertion Improvement: This operator reinserts one customer into an alternative position in the solution vector by Osman's (1993) cost savings criterion. For a customer k currently serviced between customers i and j, the operator considers all alternative positions in the solution vector. Consider the position between the adjacent customers l and m, the cost savings is evaluated as: $(c_{ik}+c_{kj}+c_{lm})-(c_{lk}+c_{km}+c_{ij})$. The improvement operator applied here uses the best-move strategy, i.e. all possible moves in the current neighborhood are evaluated and the best improvement move is selected.

Swap Improvement: This operator swaps the position of two customers simultaneously in the solution vector by Osman's (1993) cost savings criterion. For customers k and h currently serviced between customers i and j, and l and m, respectively, the swap possibility is evaluated on cost savings as: $(c_{ik}+c_{kj}+c_{lh}+c_{hm})-(c_{lk}+c_{km}+c_{ih}+c_{hj})$.

4.2.4. Crossover

In the developed CEA, the search space is confined to the feasible region; therefore, every individual is feasible. Consequently, caution should be also taken on the crossover operator because a simple exchange between two customers can violate time or capacity constraints. This study employs a revised crossover algorithm which does not entail bias in any particular direction but makes offspring inherit good properties from parents. This crossover operator has the offspring inherit as many routs as possible from parents. Once inherited routes are chosen, they can be regarded as seed routes and all other un-routed customers can be inserted into seed routes or other single customer routes.

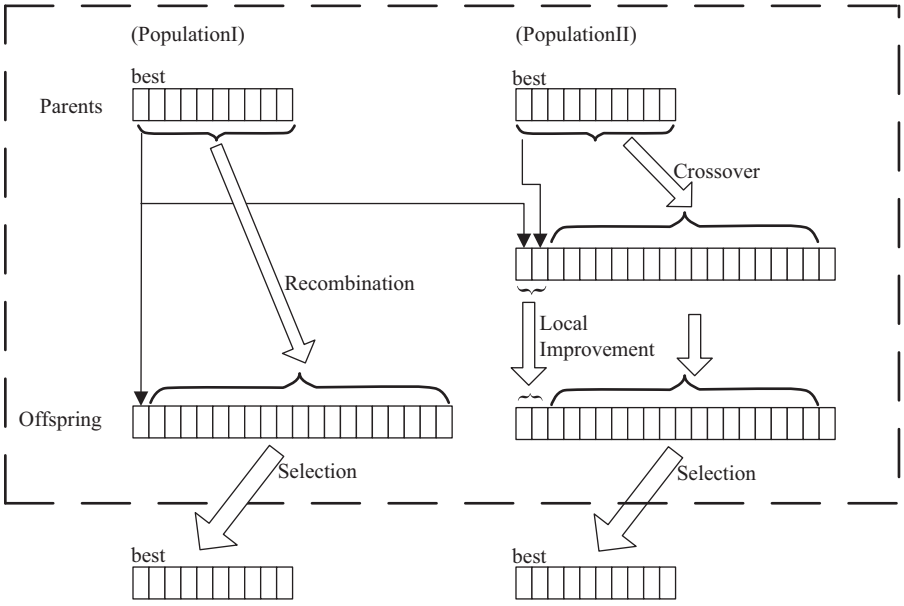


Fig. 4. Coevolutionary structure.

The criterion of the insertion is the same as the one used in RSCIM. The procedure to construct a solution after inherited routes are chosen is called Fixed Seeds Cheapest Insertion Method (FSCIM). The crossover operator generates one offspring each time. The crossover algorithm is shown in Algorithm 1:

Algorithm 1. Crossover algorithm

```
function Crossover;
begin
   repeat
        Copy Random Route from Parent 1 to the offspring;
        Copy Random Route from Parent 2 to the offspring;
until (no more inherited routes are feasible)
All un-routed customers form single customer routes;
Reduce all single customer routes by FSCIM;
end;
```

4.2.5. Selection

Instead of using the objective function as a fitness function, the fitness is related to the total travel distance (TD). This is motivated by the fact that better individuals more likely evolve from the ones with the lower TD than the ones with the lower number of vehicles (NV). Recall that the population size is N and the number of offspring is $2N$. The fitness values of the individuals with the minimal TD and the maximal TD are set to be $4N$ and $2N+1$, respectively. This implies that the individual with the minimal TD has about double the probability to be reproduced compared to the one with the maximal TD. The fitness is defined as below:

$$fitness = 4N+1-(ranking\ \ of\ \mathrm{TD}). \tag{27}$$

All offspring are evaluated twice by the fitness function and objective function. In both populations, the individual with the best objective function is kept directly by the Elitism strategy. The remaining parents of the next generation are reproduced by the Roulette wheel selection rule, related to the fitness values. In other words, these individuals are reproduced from all offspring where the individual k has the probability of reproduction as:

$$\Pr(individual\ k\ to\ be\ reproduced) = \frac{fitness\ (k)}{total\ fitness}. \tag{28}$$

4.3. Termination condition

There are two termination conditions in the developed CEA: convergence and time limit. At the beginning of each generation, updating the best individual of these two populations is based on the comparison between the best individuals of two consecutive generations. If the improvement between two consecutive generations is zero, the evolution is under a 'stagnancy' state. The developed CEA converges when the evolution has been under 'stagnancy' for consecutive 500 generations. Another termination condition is when the computational time reaches half an hour. Once one of the termination conditions is satisfied, the evolution process will be terminated and the best individual will be reported as the best solution.

5. Computational experiments

Since there have not been any studies with testing problems which were dedicated to the FDPPTW, for evaluation, this study generates some FDPPTW test problems which are revised from Wang and Chen's SDPPTW test problems (refer to Wang and Chen (2012)). Wang and Chen's SDPPTW test problems were revised from Solomon's VRPTW benchmarks (refer to Solomon (1987)). The set of Solomon's test problems is composed of six different problem types (C1, C2, R1, R2, RC1, and RC2). Each data set contains between eight to twelve 100-customer problems. The categories of the six problem types refer to:

C: with clustered customers whose time windows were generated based on a known solution;
R: with customer locations generated uniformly randomly over a square;
RC: with a combination of randomly placed and clustered customers.
where
Type 1 has narrow time windows and small vehicle capacity, and
Type 2 has large time windows and large vehicle capacity.

By revising from Wang and Chen's test problems, this study generates fifty-six 100-customer problems. For the small-scale test problems, this study also generates three 5-customer problems, three 10-customer problems, three 25-customer problems, and three 50-customer problems. Due to different objective functions used in the literatures, this analysis employs the trade-off parameter α to adjust for different decision criteria, in particular, by setting $\alpha \to 1$, to reveal the primary concern of minimizing the number of vehicles, than the minimization of the total distance. All experiments were executed on an Intel Core2 Quad 2.4 G computer with 1 G memory.

5.1. Definition of the parameters

The developed CEA does not employ mutation operators for the reason that there is no significant improvement. Those additional tests with mutation operators and other adjustments will be described in the latter subsections. Therefore, there are a few parameters to be determined. The parameters used in the developed CEA are listed in Table 1 and explained below.

SIZE_POP1 and SIZE_POP2 represent the population sizes of Population I and Population II, respectively, in which values of (50, 50) are empirically chosen. Since larger population size increases the computational time and smaller population size decreases the quality of solutions, 50 is an appropriate population size for the FDPPTW. Convergence is defined as the evolution appearing in a given amount of (CONV_COUNT = 500) consecutive stagnancy generations and is used as a stopping condition.

Table 1
Parameters used in the CEA.

CEA parameter (description)	Value
SIZE_POP1 (population size of population 1)	50
SIZE_POP2 (population size of population 2)	50
CONV_COUNT (definition for convergence in terms of number of consecutive stagnancy generations)	500

5.2. Factorial experiments

In order to increase the quality of solutions and the speed of convergence, some experiments are conducted: with or without local improvement operators, with or without mutation operators, one or two classes, and one or two populations.

Eleven mutation operators used in Wang and Chen (2012) with the concept of customer reinsertion, customer migration, customer exchange, route partitioning, and route reduction are employed to half these experiments. The other half are not employed any mutation operator. The mutation rate is temporarily set to 0.5. The mechanism of two-class is design to increase the speed of convergence. The chromosomes are separated into two classes: high class and massive class. High-class chromosomes are guaranteed to have their offspring while massive-class chromosomes are not. The mutation operators and the mechanism of two-class is employed in Population II and depicted in Fig. 5. Furthermore, for those experiments that only single population is employed, Population II is kept as that single population but Population I is removed.

The results of these experiments are illustrated in Table 2. The best setting of the developed CEA is two-population, one-class, with-local-improvement, and no-mutation. The reason that the employment of mutation does not contribute significant improvements may be that small changes of chromosomes cannot stop chromosomes from falling into local optimums. About the other factors, the employment of the local-improvement operators successfully guides chromosomes to reach better solutions. However, the mechanism of two-class neither increased the speed of convergences as expected nor increased the quality of solutions. On the contrary, the mechanism of two-population significantly increases both the speed of convergences and the quality of solutions. It successfully broadens the searching space and helps chromosomes jump out from local optimums as expected.

The above experiments show that there is no significant improvement when the mutation rate is set to be 0.5. For examining more mutation rates, two set of experiments with mutation rates 0.1 and 0.05 are further conducted. The results in Table 3 have confirmed the non-necessity of the mutation operators.

5.3. Accuracy of the developed CEA

The commercial linear programming software, like ILOG Cplex, could find the optimal solution for the small-scale FDPPTW. Hence it is adopted to evaluate the accuracy of the developed algorithm. For the none-flexible case of the FDPPTW, the SDPPTW, Wang and Chen (2012) generated some small-scale problems: three 10-customer problems, three 25-customer problems, and

Table 2
Factorial experiments.

Factors				NV	TD	Comp. time
Population	**Class**	**LI**	**Mutation**			**(s)**
One	One	None	None	105	11555.06	8910
One	One	None	0.5	108	12267.47	13749
One	One	With	None	107	11825.56	11349
One	One	With	0.5	103	11414.11	10691
One	Two	None	None	107	11952.98	14286
One	two	None	0.5	105	12250.97	14412
One	Two	With	None	104	11437.36	4548
One	Two	With	0.5	102	11367.85	7701
Two	One	None	None	103	11238.03	3198
Two	One	None	0.5	104	11279.52	4206
Two	**One**	**With**	**None**	**101**	**11247.80**	**4424**
Two	One	With	0.5	103	11319.24	5710
Two	Two	None	None	102	11153.05	3171
Two	Two	None	0.5	104	11200.60	2711
Two	Two	With	None	105	11325.45	9485
Two	Two	With	0.5	104	11290.88	11067

LI: local improvement operator; NV: number of vehicles; Comp. time: computational time.

Table 3
Experiments of the mutation rates.

Mutation rate	NV	TD	Comp. time (s)
0	**101**	**11247.80**	**4424**
0.05	103	11175.57	5268
0.1	101	11363.30	4436
0.5	103	11319.24	5710

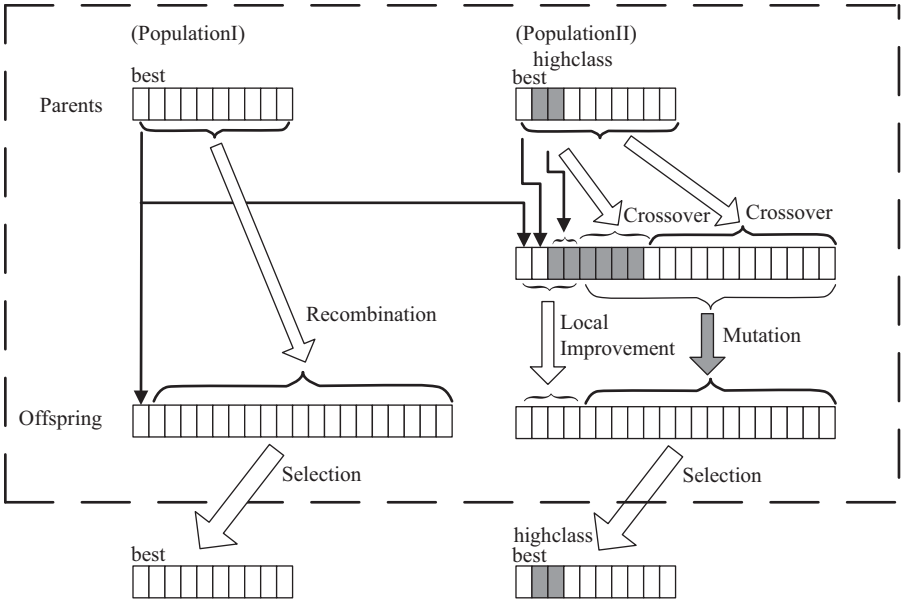


Fig. 5. Employment of mutation and two-class.

three 50-customer problems. In this study, these nine small-scale SDPPTWs are further revised to form nine small-scale FDPPTWs. They are named as RCf10101, RCf10104, RCf10107, RCf25101, RCf25104, RCf25107, RCf50101, RCf50104, and RCf50107. However, Cplex cannott solve the 50-customer problems while the developed CEA easily finds solutions for them. When the number of customer nodes is up to 50, Cplex solver incorrectly shows an error message "presolve determines problem is infeasible or unbounded" due to the truncation errors. In order to provide more evidences about the accuracy of the developed CEA, three 5-customer problems, named RCf05101, RCf05104, and RCF05107, are generated in this study. The comparison between solutions of Cplex and the developed CEA to the small-scale FDPPTW is listed in Table 4. One can see that Cplex is only able to find the optimal solutions of RCf05101, RCf05104, RCf05107, and RCf10101 by 5∼1169 s. However, the developed CEA can get their optimal solution by only 1 s. For the rest of the test problems, Cplex gives the "out of memory" best values for five problems, but cannot find feasible solutions for all of three 50-customer problems due to the truncation errors.

Table 4
Comparison between the solutions of Cplex and the developed CEA to the small-scale SDPPTW.

Problem	Cplex			CEA		
	NV	TD	Com. time	NV	TD	Com. time
RCf05101	3	220.15	5	3	220.15	1
RCf05104	2	214.57	1169	2	214.57	1
RCf05107	2	211.83	473	2	211.83	1
RCf10101	3	347.38	897	3	347.38	1
RCf10104	2[a]	316.99[a]	9893	2	216.69	1
RCf10107	3[a]	264.96[a]	18266	2	310.81	4
RCf25101	10[a]	1137.98[a]	66123	5	529.13	6
RCf25104	12[a]	1777.95[a]	87606	4	473.46	5
RCf25107	11[a]	1526.01[a]	117612	4	567.15	10
RCf50101	[b]	[b]	[b]	9	928.50	60
RCf50104	[b]	[b]	[b]	6	738.30	69
RCf50107	[b]	[b]	[b]	7	807.86	57

[a] The "out of memory" values.
[b] Cplex cannot solve problems due to truncation errors.

5.4. Computational results for the FDPPTW

Wang and Chen (2012) generated fifty-six 100-customer SDPPTW test problems by revising Solomon benchmarks. For evaluating the performance of the developed CEA to the FDPPTW, fifty-six 100-customer FDPPTW test problems are generated by revising those SDPPTWs in this study. In each problem, a longer pickup time window is created by extending the original time window. The CEA results for the FDPPTWs and Wang and Chen's results for the SDPPTWs are compared in Table 5. The average computational times of the SDPPTW and the FDPPTW are 10.7 and 8.7 min. A concise comparison is listed in Table 6. The FDPPTW scheme uses total 7 fewer vehicles than the SDPPTW scheme dose. This also matches the declaration that the FDPPTW scheme is more flexible and more economical than the SDPPTW scheme.

6. Conclusions

Based on green issues, the closed-loop logistics have become more and more important in recent years. Since the bi-directional logistics is the most complicated part in a closed-loop supply chain, the delivery and pickup problems have also drawn much

Table 5
Comparison between the SDPPTW and the FDPPTW.

Problem	SDPPTW		FDPPTW		Gap		Problem	SDPPTW		FDPPTW		Gap	
	NV	TD	NV	TD	NV	TD (%)		NV	TD	NV	TD	NV	TD (%)
C*101	11	1001.97	10	858.24	**−1**	**−14.34**	C*201	3	591.56	3	591.56	0	0.00
C*102	10	961.38	10	897.23	0	**−6.67**	C*202	3	591.56	3	591.56	0	0.00
C*103	10	897.65	10	850.10	0	**−5.30**	C*203	3	591.17	3	591.17	0	0.00
C*104	10	878.93	10	898.04	0	2.17	C*204	3	590.6	3	590.6	0	0.00
C*105	11	983.10	10	922.05	**−1**	**−6.21**	C*205	3	588.88	3	588.88	0	0.00
C*106	11	878.29	10	862.08	**−1**	**−1.85**	C*206	3	588.49	3	588.49	0	0.00
C*107	11	913.81	10	854.69	**−1**	**−6.47**	C*207	3	588.29	3	588.29	0	0.00
C*108	10	951.24	10	840.51	0	**−11.64**	C*208	3	588.32	3	588.32	0	0.00
C*109	10	940.49	10	928.47	0	**−1.28**							
R*101	19	1653.53	18	1606.16	**−1**	**−2.86**	R*201	4	1280.44	4	1289.15	0	0.68
R*102	17	1488.04	16	1443.61	**−1**	**−2.99**	R*202	4	1100.92	4	1133.73	0	2.98
R*103	14	1216.16	14	1227.14	0	0.90	R*203	3	950.79	3	974.37	0	2.48
R*104	10	1015.41	10	1022.58	0	0.71	R*204	3	775.23	3	775.67	0	0.06
R*105	15	1375.31	14	1391.09	**−1**	1.15	R*205	3	1064.43	3	1101.47	0	3.48
R*106	13	1255.48	13	1291.21	0	2.85	R*206	3	961.32	3	1011.14	0	5.18
R*107	11	1087.95	11	1104.33	0	1.51	R*207	3	835.01	3	839.75	0	0.57
R*108	10	967.49	10	974.99	0	0.78	R*208	3	718.51	3	728.18	0	1.35
R*109	12	1160.00	12	1201.52	0	3.58	R*209	3	930.26	3	1008.49	0	8.41
R*110	12	1116.99	12	1136.52	0	1.75	R*210	3	983.75	3	1076.93	0	9.47
R*111	11	1065.27	11	1119.90	0	5.13	R*211	3	839.61	3	826.04	0	**−1.62**
R*112	10	974.03	10	976.69	0	0.27							
RC*101	15	1652.9	14	1728.07	**−1**	4.55	RC*201	4	1587.92	4	1504.38	0	**−5.26**
RC*102	14	1497.05	13	1493.89	**−1**	**−0.21**	RC*202	4	1211.12	4	1269.46	0	4.82
RC*103	12	1338.76	12	1334.29	0	**−0.33**	RC*203	4	964.65	4	1054.28	0	9.29
RC*104	11	1188.49	12	1254.67	1	5.57	RC*204	3	822.02	3	817.01	0	**−0.61**
RC*105	14	1581.26	14	1538.77	0	**−2.69**	RC*205	4	1410.18	4	1417.18	0	0.50
RC*106	13	1422.87	13	1444.44	0	1.52	RC*206	3	1176.85	4	1128.03	1	**−4.15**
RC*107	12	1282.10	12	1292.51	0	0.81	RC*207	4	1036.59	4	1053.44	0	1.63
RC*108	11	1175.04	11	1161.16	0	**−1.18**	RC*208	3	878.57	3	1020.95	0	16.21

NV—Number of vehicles; TD—Total distance.
Boldface indicates that the solution of the FDPPTW has fewer vehicle or lower total distance.

Table 6
Concise comparison between the SDPPTW and the FDPPTW.

	C1	C2	R1	R2	RC1	RC2	Cumulative	Comp. time
Non-flexible								
NV	94	24	154	35	102	29	438	35981
TD	8406.86	4718.87	14375.66	10440.27	11138.47	9087.90	58168.03	
Flexible								
NV	90	24	151	35	101	30	431	29359
TD	7911.41	4718.87	14495.74	10764.92	11247.80	9264.73	58403.47	

attention in the literatures in the past two decades. The existing operations for three categories of the delivery and pickup problems have some shortages and shortcomings. For the vehicle routing problem with backhaul and time windows and the simultaneous delivery and pickup problem with time windows, they didn't allow the flexible mix of pickup services and delivery services. On the other hand, in the mixed vehicle routing problem with backhauls and time windows, if the delivery and pickup service of a customer was performed simultaneously, the accessing time was not reduced. In this study, the flexible delivery and pickup problem with time windows is considered to overcome the above shortcomings.

The problem is formulated into a mixed binary integer programming model and one can implement this model by Cplex to get the optimal solution if the scale of the problem is small. Since the problem is NP-hard, this study develops a coevolutionary algorithm to get near optimal solutions in an acceptable computational time. The termination conditions are able to prevent exhaustive computations.

This study generates some test problems by revising the well-known Solomon's benchmarks which are oringally used for the vehicle routing problem with time windows. The comparison between the results of Cplex software and the developed algorithm shows the efficiency, but in particular, the accuracy of the developed algorithm. Further comparison with the non-flexible case provides the time-saving evidence of the developed algorithm, and thus the flexible scheme is ont only more flexible but also more economical.

Nowadays, pickup demand in the reverse networks often accompanies uncertainty. The fuzzy mathematics can be a possible tool to enhance the applicability of the model. Hence, how to extend the model to cope with uncertainty is a direction of the further study.

Acknowledgments

The authors acknowledge the financial support from the National Science Council, Taiwan, ROC under Project No. NSC97-2221-E007-095-MY3.

References

Ai, T.J., Kachitvichyanukul, V., 2009. A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. Computers and Operations Research 36, 1693–1702.

Angelelli, E., Mansini, R., 2002. The vehicle routing problem with time windows and simultaneous pickup and delivery. In: Klose., A., Speranza, M.G., Van Wassenhove, L.N. (Eds.), Quantitative Approaches to Distribution Logistics and Supply Chain Management (Lecture Notes in Economics and Mathematical Systems). Springer Press, New York, pp. 249–267.

Berbeglia, G., Cordeau, J.F., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: a classification scheme and survey. TOP 15, 1–31.

Bianchessi, N., Righini, G., 2007. Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. Computers and Operations Research 34, 578–594.

Chen, J.-F., Wu, T.-H., 2006. Vehicle routing problem with simultaneous deliveries and pickups. Journal of the Operational Research Society 57, 579–587.

Clarke, G., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 12, 568–581.

Crispim, J., Brandao, J., 2005. Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. Journal of the Operational Research Society 56, 1296–1302.

Dell'Amico, M., Righini, G., Salani, M., 2006. A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. Transportation Science 40, 235–247.

Dethloff, J., 2001. Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up. OR Spectrum 23, 79–96.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

Ilgin, M.A., Gupta, S.M., 2010. Environmentally conscious manufacturing and product recovery (ECMPRO): a review of the state of the art. Journal of Environmental Management 91, 563–591.

Kontoravdis, G., Bard, J., 1995. A GRASP for the vehicle routing problem with time windows. ORSA Journal on Computing 7 (1), 10–23.

Mester, D., Bräysy, O., Dullaert, W., 2007. A multi-parametric evolution strategies algorithm for vehicle routing problems. Expert Systems with Applications 32, 508–517.

Min, H., 1989. The multiple vehicle routing problem with simultaneous delivery and pick-up points. Transportation Research A 23 (5), 377–386.

Montané, F.A.T., Galvao, R.D., 2006. A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. Computers and Operations Research 33 (3), 595–619.

Nagy, G., Salhi, S., 2005. Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. European Journal of Operational Research 162 (1), 126–141.

Osman, I., 1993. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problems. Annals of Operations Research 41, 421–451.

Parragh, S.N., Doerner, K.F., Hartl, R.F., 2008. A survey on pickup and delivery problems part II: transportation between pickup and delivery locations. Journal of Betriebswirtschaft 58, 81–117.

Ropke, S., Pisinger, D., 2006. A unified heuristic for a large class of vehicle routing problems with backhauls. European Journal of Operational Research 171, 750–775.

Solomon, M.M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35 (2), 254–265.

Tütüncüa, G.Y., Carreto, C.A.C., Baker, B.M., 2009. A visual interactive approach to classical and mixed vehicle routing problems with backhauls. Omega 37, 138–154.

Wade, A.C., Salhi, S., 2002. An investigation into a new class of vehicle routing problem with backhauls. Omega 30, 479–487.

Wang, H.-F., Chen, Y.-Y., 2012. A genetic algorithm for the simultaneous delivery and pickup problems with time window. Computers and Industrial Engineering 62, 84–95.

Wang, H.-F., Hsu, H.-W., 2010. A closed-loop logistic model with a spanning-tree based genetic algorithm. Computers and Operations Research 37 (2), 376–389.

Zhong, Y., Cole, M.H., 2005. A vehicle routing problem with backhauls and time windows: a guided local search solution. Transportation Research Part E 41, 131–144.