A Memetic Algorithm for Periodic Capacitated Arc Routing Problem

Yi Mei, *Student Member, IEEE*, Ke Tang, *Member, IEEE*, and Xin Yao, *Fellow, IEEE*

***Abstract*—This paper investigates the Periodic Capacitated Arc Routing Problem (PCARP), which is often encountered in the waste collection application. PCARP is an extension of the well-known Capacitated Arc Routing Problem (CARP) from a single period to a multi-period horizon. PCARP is a hierarchical optimization problem which has a primary objective (minimizing the number of vehicles mnv) and a secondary objective (minimizing the total cost tc). An important factor that makes PCARP challenging is that its primary objective mnv is little affected by existing operators and thus difficult to improve. We propose a new Memetic Algorithm (MA) for solving PCARP. The MA adopts a new solution representation scheme and a novel crossover operator. Most importantly, a Route-Merging (RM) procedure is devised and embedded in the algorithm to tackle the insensitive objective mnv. The MA with RM (MARM) has been compared with existing meta-heuristic approaches on two PCARP benchmark sets and a real-world data set. The experimental results show that MARM obtained better solutions than the compared algorithms in much less time, and even updated the best known solutions of all the benchmark instances. Further study reveals that the RM procedure plays a key role in the superior performance of MARM.**

***Index Terms*—Capacitated arc routing problems, combinatorial optimization, evolutionary algorithms, memetic algorithms.**

I. Introduction

As a CLASSICAL combinatorial optimization problem, Capacitated Arc Routing Problem (CARP) [13] has attracted much research interest from various scientific and industrial fields. It has many real-world applications in the fields of logistics and transportation management, such as waste collection, post delivery, winter gritting and snow removal. CARP can be described as follows: Given a connected graph, some edges (called the *tasks*) of the graph are required to be served by a vehicle fleet located at the *depot* vertex. CARP aims to determine a least-cost plan subject to the following constraints:

1) Each vehicle must start and end at the depot vertex;
2) Each task is served exactly once by one vehicle;
3) The total demand of the tasks served by each route cannot exceed its capacity.

CARP has been proven to be NP-hard in [19]. Thus, exact methods are only available for small and medium instances. When solving real-world instances which often have large problem size, heuristics and meta-heuristics are promising approaches for finding near-optimal solutions in the given time budget. So far, numerous constructive heuristics (e.g., [18], [19], [32], [33], [39]) and meta-heuristics (e.g., the guided local search [6], the tabu search [7], [20], [22], [30], the variable neighborhood search [23], [34] and the memetic algorithm [21], [26], [37]) have been proposed for CARP, and typically been evaluated by comparing the obtained solutions to the tight lower bounds calculated with the relaxation methods proposed in [2], [4].

Despite the intensive investigations conducted on CARP, there is still a wide gap between this simple model and real-world situations. Hence, for many applications in reality, the CARP approaches can hardly be directly applied. To model the real-world applications more closely, an extended CARP, called the Periodic CARP (PCARP), is studied in this paper. PCARP is often encountered in the waste collection application, in which the department in charge aims to design a plan to collect the daily waste produced by each street in the city. In practice, there may be a large difference in the amount of daily waste production and the tolerance of the waste duration between the streets, depending on the region, population size, the type of buildings around, etc. To be more specific, the apartment blocks and heavy industrial regions produce a large amount of daily waste that cannot be stored for a long time. It is necessary to collect waste for such streets everyday. On the other hand, the rural areas could keep the produced waste for a relatively long time. These streets can be treated less frequently (e.g., every two days or twice a week). Therefore, it is more proper to make a schedule for a multi-period horizon (e.g., a week or month) rather than a single day. PCARP is thus introduced to model such a situation.

As first introduced by Lacomme *et al.* [25], PCARP can be described as follows: Given a connected graph and a multi-period horizon, the tasks need to be served for a certain number of times (say their *service frequencies*) over the horizon. PCARP is to make an optimal service plan so that the following

Manuscript received March 6, 2011; accepted May 11, 2011. Date of publication July 14, 2011; date of current version November 18, 2011. This work was partially supported by an EPSRC Grant (No. EP/E058884/1) on "Evolutionary Algorithms for Dynamic Optimisation Problems: Design, Analysis and Applications," the National Natural Science Foundation of China under Grants U0835002 and 61028009, and the grant for Distinguished Young Scholars of Anhui Province. This paper was recommended by Associate Editor H. Takagi.

Y. Mei and K. Tang are with the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: meiyi@mail.ustc.edu.cn; ketang@ustc.edu.cn).

X. Yao is with the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China. He is also with CERCIA, the School of Computer Science, University of Birmingham, B15 2TT Birmingham, U.K. (e-mail: x.yao@cs.bham.ac.uk).

Digital Object Identifier 10.1109/TSMCB.2011.2158307

constraints are satisfied:

1) In each period, each vehicle must start and end at the depot vertex;
2) In each period, each task is served no more than once;
3) The number of services of each task over the horizon equals its service frequency;
4) The *period combination* of each task must be in the *allowed period combination set*;
5) In each period, the total *accumulated demand* of the tasks served by each vehicle cannot exceed its capacity.

In Constraint 4) of PCARP, the period combination of a task means the periods in which the task is served. The allowed period combination set is predefined by the problem nature. For instance, consecutive services may be forbidden for the tasks with low service frequencies and thus not in the allowed period combination set. In Constraint 5), the demand of tasks are replaced by the accumulated demand. This is derived from the fact that in the waste collection application, the untreated waste after one service will be retained until the next service reaches. Therefore, a service for a task in PCARP is to serve the accumulated demand since its previous service.

When there is only one period and all the tasks are to be served in this period, PCARP is reduced to CARP. In this sense, PCARP can be seen as a generalization of CARP from single-period to multi-period, and CARP can be seen as a single-period special case of PCARP.

Besides the constraints, PCARP is different from CARP in the objectives as well. As mentioned in [25], waste management departments usually consider minimizing the investment cost (depending on the fleet size) prior to minimizing the operating costs (i.e., the total cost). Hence, PCARP has a primary objective of minimizing the number of vehicles used over the horizon (denoted as mnv) and a secondary objective of minimizing the total cost (denoted as tc).

Being a relatively new topic, research on PCARP is still in its infancy and there have been only a few literatures available so far. It was first defined by Lacomme *et al.* [25]. Then, Chu *et al.* built a linear programming model and proposed several constructive heuristics [8], and proposed lower bounds for it [9]. Lacomme *et al.* proposed a Memetic Algorithm (MA) [27] (referred to as LMA hereafter) and Chu *et al.* proposed a Scatter Search (SS) [10]. LMA and SS extend a solution representation scheme for CARP and the corresponding search operators to the multi-period case, and include an elite solution generated by a greedy constructive heuristic in the initial population for performance enhancement. Recently, Kansou and Yassine [24] proposed an efficient constructive heuristic and an ant colony system.

Compared with PCARP, the Periodic Vehicle Routing Problem (PVRP), which can be viewed as the vertex routing counterpart of PCARP, arose earlier and received more research interests (e.g., [1], [3], [11], [14], [15], [17]). Nevertheless, previous work on PVRP ignored the primary objective mnv. Besides, CARP has more complicated problem characteristics than VRP, e.g., a CARP with n tasks can be transformed to a VRP with $2n+1$ vertices [29]. Thus, solving PCARP will be more complicated than solving PVRP.

In this paper, it is found that the primary objective mnv can hardly be improved by existing search operators. To address this issue, a specific Route-Merging (RM) procedure is developed. Then, the RM procedure is embedded in the MA framework and the MA with RM (MARM) is thereby proposed. In MARM, the RM procedure is located before the local search process. Thus, a solution is first improved in mnv by the RM procedure, and then improved in tc by the subsequent local search. The experimental results on two benchmark sets and a real-world data set demonstrate the superior performance of MARM.

The rest of the paper is organized as follows: First, Section II gives the notations and problem definition. Afterwards, the newly introduced primary objective mnv is investigated and the RM procedure is introduced in Section III. Then, Section IV describes MARM in detail. After that, experimental studies are carried out in Section V. Finally, Section VI concludes this paper.

II. Notations and Problem Definition

In PCARP, a connected graph $G(V,E)$ and a p-period horizon are given, where V and E are the vertex and edge sets. Each edge $(u,v)\in E$ is associated with a serving cost $sc(u,v)$, a deadheading cost $dc(u,v)$, a demand vector $\boldsymbol{d}(u,v)=(d_1(u,v),\ldots,d_p(u,v))$ and a service frequency $freq(u,v)$. The serving cost indicates the cost induced by serving (u,v), while the deadheading cost is the cost induced by traversing (u,v) without service. $d_i(u,v)$ of $\boldsymbol{d}(u,v)$ is the demand of (u,v) increased in period i. The task set T consists of the edges with positive demand in at least one period, and the number of tasks is $n=|T|$. Note that the serving cost is only induced by serving a task, we have $sc(u,v)>0, \forall (u,v)\in T$ and $sc(u,v)=0, \forall (u,v)\notin T$. Besides, each task $(u,v)\in T$ is associated with an allowed period combination set $APC(u,v)$, which consists of a number of period combinations being represented as a p-dimensional 0–1 vector. The ith component of the vector takes 1 if (u,v) is served in period i, and 0 otherwise. For example, (1, 0, 0, 1, 0, 0, 0) indicates that (u,v) is served on Monday and Thursday of the week. Without loss of generality, the depot vertex is denoted as $v_0\in V$. These services are to be served by m vehicles with capacity of Q.

Under the above notations, the route X_{ij} traversed by vehicle j in period i can be denoted as a sequence of vertices (with length of l_{ij}), i.e., $X_{ij}=(x_{ij1},\ldots,x_{ijl_{ij}})$, where $x_{ijk}\in V$, $k=1,\ldots,l_{ij}$. Besides, a 0–1 vector $Y_{ij}=(y_{ij1},\ldots,y_{ij(l_{ij}-1)})$ is used to indicate the positions of the task services in X_{ij}. Concretely, $y_{ijk}=1$ if $(x_{ijk},x_{ij(k+1)})$ is a service, and $y_{ijk}=0$ otherwise. Then, a PCARP solution S can be denoted as the combination of the X_{ij}'s and Y_{ij}'s, i.e., $S=\{(X_{ij},Y_{ij})|i=1,\ldots,p; j=1,\ldots,m\}$.

Given a solution S, the period combination $PC(u,v,S)$ of each task (u,v) is represented as $PC(u,v,S)=(PC_1(u,v,S),\ldots,PC_p(u,v,S))$, where $PC_i(u,v,S)$ takes 1 if (u,v) is served in period i of S, and 0 otherwise. It can be obtained as follows: For period i, if there exists $j\in\{1,\ldots,m\}$ and $k\in\{1,\ldots,l_{ij}\}$ so that $y_{ijk}=1$ and $(x_{ijk},x_{ij(k+1)})=(u,v)$ or $(x_{ijk},x_{ij(k+1)})=(v,u)$, then $PC_i(u,v,S)=1$. Otherwise, $PC_i(u,v,S)=0$.

Based on the period combination, the accumulated demand $ad(u,v,i,S)$ of a task (u,v) in period i can be further obtained

TABLE I
NOTATIONS USED IN THE PCARP DEFINITION

Notation	Description
G	The given graph
V	The vertex set
E	The edge set
T	The task set
v_0	The depot vertex
$(u,v)\in E$	An edge
$sc(u,v)$	Serving cost of (u,v)
$dc(u,v)$	Deadheading cost of (u,v)
$\boldsymbol{d}(u,v)$	Demand vector of (u,v)
$freq(u,v)$	Service frequency of (u,v)
$PC(u,v,S)$	The period combination of (u,v) in S
$APC(u,v)$	Allowed period combination set of (u,v)
m	The number of vehicles available
n	The number of tasks
p	The number of periods
Q	The capacity of vehicles
S	A PCARP solution
X_{ij}	The vertex sequence traversed by vehicle j in period i
Y_{ij}	The 0-1 vector corresponding to X_{ij}
x_{ijk}	The k^{th} vertex of X_{ij}
y_{ijk}	The k^{th} 0-1 indicator of Y_{ij}
$ad(u,v,i,S)$	The accumulated demand of (u,v) in period i of S
$nv(i,S)$	The number of vehicles used in period i of S
$mnv(S)$	The number of vehicles of S used over the horizon
$tc(S)$	The total cost of S
α	The weight of $mnv(S)$ in the objective function

by the following steps:

i) Set $l=i$, $ad(u,v,i,S)=d_l(u,v)$;
ii) $l\leftarrow l-1$. If $l=0$, set $l=p$;
iii) If $PC_l(u,v,S)=1$, return $ad(u,v,i,S)$;
iv) $ad(u,v,i,S)\leftarrow ad(u,v,i,S)+d_l(u,v)$. go back to ii).

Then, the total cost and total accumulated demand of the tasks served by each route can be obtained as:

$$tc(X_{ij},Y_{ij})=\sum_{k=1}^{l_{ij}-1} sc\left(x_{ijk},x_{ij(k+1)}\right)\cdot y_{ijk} + \sum_{k=1}^{l_{ij}-1} dc\left(x_{ijk},x_{ij(k+1)}\right)\cdot(1-y_{ijk}) \quad (1)$$

$$d(X_{ij},Y_{ij})=\sum_{k=1}^{l_{ij}-1} ad\left(x_{ijk},x_{ij(k+1)},i,S\right)\cdot[y_{ijk}=1] \quad (2)$$

where the indicator function $[A]$ of an event A is defined as:

$$[A]=\begin{cases}1, & A \text{ is true;}\\ 0, & \text{otherwise.}\end{cases} \quad (3)$$

Obviously, if vehicle j is not used in period i, then both X_{ij} and Y_{ij} are empty vectors and $l_{ij}=0$. Therefore, the number of vehicles $nv(i,S)$ in period i can be obtained as:

$$nv(i,S)=\sum_{j=1}^{m}[l_{ij}>0]. \quad (4)$$

Finally, the primary objective $mnv(S)$ and the secondary objective $tc(S)$ of solution S can be computed as:

$$mnv(S)=\max_{i\in\{1,\ldots,p\}} nv(i,S) \quad (5)$$

$$tc(S)=\sum_{i=1}^{p}\sum_{j=1}^{m} tc(X_{ij},Y_{ij}). \quad (6)$$

For clarity, all the notations used in the problem definition are presented in Table I.

Then, PCARP can be defined as follows:

$$\min \quad f(S)=\alpha\cdot mnv(S)+tc(S) \quad (7)$$

$$s.t.: \; x_{ij1}=x_{ijl_{ij}}=v_0, \qquad \forall i=1,\ldots,p,\; j=1,\ldots,m; \quad (8)$$

$$\left(x_{ijk_1},x_{ij(k_1+1)}\right)\neq\left(x_{ijk_2},x_{ij(k_2+1)}\right), \; \forall y_{ijk_1}=1, y_{ijk_2}=1, i=1,\ldots,p,\; j=1,\ldots,m; \quad (9)$$

$$\left(x_{ijk_1},x_{ij(k_1+1)}\right)\neq\left(x_{ij(k_2+1)},x_{ijk_2}\right), \; \forall y_{ijk_1}=1, y_{ijk_2}=1, i=1,\ldots,p,\; j=1,\ldots,m; \quad (10)$$

$$PC(u,v,S)\in APC(u,v),\; \forall (u,v)\in T \quad (11)$$

$$d(X_{ij},Y_{ij})\leqslant Q, \qquad \forall i=1,\ldots,p, j=1,\ldots,m. \quad (12)$$

In the objective function (7), the primary objective $mnv(S)$ and the secondary objective $tc(S)$ are combined into a single objective function $f(S)$ by the weighted sum approach. α is a sufficiently large constant that can guarantee the priority of mnv (e.g., $\alpha=100\,000$ in the subsequent experiments). Constraint (8) means that each route starts and ends at the depot. Constraints (9) and (10) indicate that each task is served no more than once in each period. Constraint (11) restricts the period combinations of the tasks in their allowed period combination sets. Constraint (12) indicates the capacity constraint. Note that the equality between the number of services of each task and its service frequency is implicitly ensured by Constraints (8)–(11).

III. IMPROVING THE mnv WITH ROUTE-MERGING

PCARP is different from CARP in various problem characteristics. First, PCARP contains multiple CARPs, each for a period. Second, the restriction imposed on the period combination of the tasks makes the feasible regions in the solution space smaller and more separated. Finally, and most importantly, PCARP introduces a primary objective mnv whereas the tc, which is the only objective of CARP, becomes a secondary objective. This makes PCARP a hierarchical optimization problem rather than a simple single-objective optimization problem.

Since PCARP is an extension of CARP, it is reasonable to consider extending CARP approaches to solve PCARP. This can be accomplished by modifying the solution representation scheme and corresponding search operators. Both LMA and SS adopted this idea, and extended the solution representation and search operators of a competitive MA for CARP [26] to solve PCARP. However, merely extending the search operators can by no means lead to high-quality PCARP solutions. The reason lies in that CARP only considers minimizing tc while PCARP considers minimizing mnv prior to minimizing tc. The search operators extended from those for minimizing tc will also keep their eyes on tc and thus will be unable to attain high-quality solutions in PCARP, which must have small mnv's as well as small tc's.

LMA and SS have already made some effort to deal with mnv. LMA maintains a small-mnv solution in the population throughout the search process so that mnv of the final solution can be guaranteed. SS adopts elite solution scheme as well, and modifies the evaluation phase to focus more on mnv. Nevertheless, generating good solutions and evaluating them are both crucial to the success of an algorithm. Although SS makes the latter more effective, nothing has been done to the former.

Compared with tc, mnv is much less sensitive to the search operators, including the crossover and local search operators. When applying a crossover operator, the offspring will inherit the characteristics of its parents. Thus, there is no trend that crossover operators can generate solutions with smaller mnv's than their parents. During the local search, the existing operators generally define a small neighborhood around the current solution by moving only one or two tasks. However, reducing mnv by one requires eliminating a route in all the periods that have the most routes over the horizon, which means selecting one route in each such period and move all the involved tasks to other routes. This requires many steps of the local search. Thus, it is difficult for the existing local search operators to improve mnv.

Based on the above discussions, it is necessary to design a specific operator to deal with mnv directly. To this end, the RM procedure is developed. It is described in Algorithm 1.

Algorithm 1: The Route-Merging (RM) procedure

```
Input: A solution $S$, $minmnv$;
Output: A solution $S'$;
1: Set $S' = S$;
2: for $i = 1 \rightarrow p$ do
3:    if $nv(i, S') > minmnv$ then
4:       for $j = 1 \rightarrow nv(i, S') - minmnv$ do
5:          Call $(R_1, R_2) = \text{MinAccDemands}(S', i)$;
6:          Call $\text{Merge}(R_1, R_2)$;
7:       end for
8:    end if
9: end for
10: return $S'$
```

The function $(R_1, R_2) = \text{MinAccDemands}(S', i)$ returns the two routes R_1 and R_2 whose total accumulated demands are minimized among all the route couples in S'_i. The function $\text{Merge}(R_1, R_2)$ merges the two routes by appending the latter route to the end of the former.

It is obvious that the performance of the RM procedure depends on the user-defined parameter $minmnv$. To keep the balance between mnv and other objectives such as tc and the violation to the capacity constraints, $minmnv$ should be set to an appropriate value. Here, we set $minmnv$ to the lower bound of mnv, which can be obtained as follows:

$$minmnv = \left\lceil \frac{\sum_{(u,v)\in T} \sum_{i=1}^{p} d_i(u,v)}{p_w Q} \right\rceil \tag{13}$$

where p_w indicates the number of working periods (the periods that can be used to allocate services) over the horizon. Since the above value is the theoretical minimal mnv value that can be achieved in the solution space, it should be a proper setting for $minmnv$.

The computational complexity of the RM procedure is $\sum_{i=1}^{p} \max\{nv(i,S) - minmnv, 0\} = O(pm)$.

The RM procedure can keep the mnv no larger than $minmnv$. However, it may increase the violation to the capacity constraints since the the tasks that are originally served in two routes will be served in the single merged route. Therefore, the RM procedure can be seen as reducing mnv at the cost of increasing the violation to the capacity constraints, and efforts are still to be made before successfully combining the RM procedure and existing algorithms. In our study, the RM procedure is carefully embedded in the framework of MA and the resultant MARM is thus proposed. Next, we will introduce MARM in detail.

IV. Memetic Algorithm With Route-Merging

As a recently growing area in evolutionary computation, MA was firstly introduced by Moscato in [31]. It can be viewed as a class of population-based meta-heuristic approaches that incorporates local search procedures with the traditional genetic algorithms, and has been successfully applied to many real-world problems (e.g., [16], [28], [38]) with better solutions achieved and the ability of exploring the solution spaces more efficiently than traditional genetic algorithms. In the field of PCARP, the only two meta-heuristic approaches LMA and SS can both be viewed as adopting the framework of MA by combining global search operators with local search process. Without loss of generality, the framework of MA can be presented in Algorithm 2.

Algorithm 2: The framework of MA

```
1: Initialization: Generate an initial population;
2: while stopping criteria are not met do
3:    Evaluate all individuals in the population;
4:    Evolve a new population using evolutionary operators;
5:    for each individual in the new population do
6:       Perform local search with probability $P$;
7:    end for
8: end while
```

The framework of MARM is derived from Algorithm 2, but with certain problem-specific modifications and extensions so as to solve PCARP more effectively.

A. Framework of the Algorithm

Algorithm 3 gives the framework of MARM. At first, the population *pop* is set empty. Then, initial solutions are generated and inserted into *pop* one by one. To keep the diversity of the population, identical solutions, also called *clones*, are not allowed in the population throughout the search process. Once an initial solution has been generated, it is compared with all the solutions in *pop*. If it is not a clone of any solution in *pop*, then it is accepted and inserted into *pop*. Otherwise, it is abandoned and a new trial starts. The initialization phase terminates when *pop* is full (its size equals $popsize$) or no eligible solution has been generated after $Mtrial$ consecutive trials.

The population initialization is followed by the search process, which consists of a number of generations. At each generation, two solutions S_1 and S_2 are randomly selected from pop. Then, the crossover operator is applied to S_1 and S_2 to generate an offspring S_x. After that, the RM procedure and local search are applied to S_x in turn. Finally, the offspring is compared with the solutions in pop. If it is not a clone, it is inserted into pop and then the worst solution in pop is removed to keep the size of pop. For the removal, the solutions in pop is sorted by the stochastic ranking and the last solution is removed. The search process stops after G_{max} generations.

Algorithm 3: The framework of MARM

```
Input: A PCARP instance, $popsize, Mtrial, P_{ls}, G_{max}$;
Output: A PCARP solution $S_{bf}$;
      // Initialization:
1: Set $pop = \emptyset$;
2: while $|pop| < popsize$ do
3:    Set $ntrial = 0$;
4:    repeat
5:       Generate a solution $S_0$;
6:       $ntrial \leftarrow ntrial + 1$;
7:    until ($S_0$ is not a clone) or $(ntrial = Mtrial)$
8:    if $S_0$ is a clone then
9:       break;
10:   end if
11:   $pop \leftarrow pop \cup \{S_0\}$;
12: end while
      // Main Loop:
13: Set $ngen = 0$;
14: while $ngen < G_{max}$ do
15:   $ngen \leftarrow ngen + 1$;
16:   Call $(S_1, S_2) =$ RandSelect($pop$);
17:   Call $S_x =$ Crossover($S_1, S_2$);
18:   Call $S_x =$ RM($S_x, minmnv$);
19:   Sample $r \in U(0, 1)$;
            // $U(0, 1)$ is an uniform distribution between 0 and 1
20:   if $r < P_{ls}$ then
21:      Call $S_{ls} =$ LocalSearch($S_x$);
22:      if $S_{ls}$ is not a clone then
23:         $pop \leftarrow pop \cup \{S_{ls}\}$;
24:      else if $S_x$ is not a clone then
25:         $pop \leftarrow pop \cup \{S_x\}$;
26:      end if
27:   else if $S_x$ is not a clone then
28:      $pop \leftarrow pop \cup \{S_x\}$;
29:   end if
30:   if $S_{ls}$ or $S_x$ has been inserted in $pop$ then
31:      Call $pop =$ StochasticRanking($pop$);
32:      $pop = pop(1 : |pop| - 1)$;
33:   end if
34: end while
35: return the best feasible solution $S_{bf}$ in $pop$;
```

In PCARP, the complicated solution structure makes an exact clone examination between PCARP solutions quite time-consuming. In this situation, the following approximated scheme is used instead: Solution S_1 is considered as a clone of solution S_2 if they have the same values of the mnv, tc and violation to the capacity constraints.

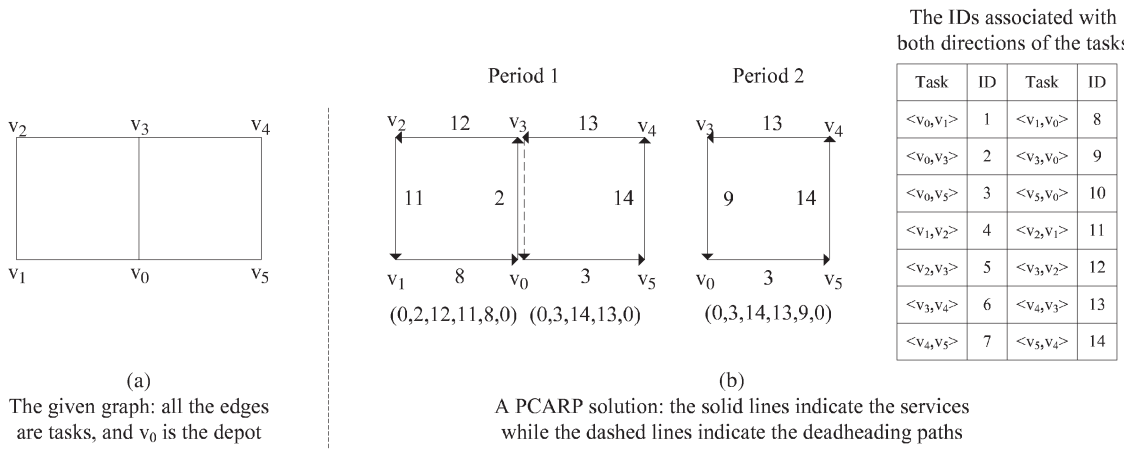
Next, we will describe the details of MARM, including the solution representation and evaluation, solution initialization, crossover operator and local search process.

B. Solution Representation and Evaluation

How to represent a solution is a fundamental issue in a stochastic search algorithm such as MA. Different representation schemes will build different fitness landscapes in the solution space, and thus lead to different difficulties to search for the global optimum. LMA and SS employ the same solution representation scheme, which can be called the implicit task encoding scheme. In the implicit task encoding scheme, a PCARP solution is represented as a number of task sequences, each for a period. During the evaluation phase, each task sequence undergoes a two-phase decoding procedure. First, it is split into a set of feasible routes with respect to the capacity constraints so that the additional cutting cost is minimized. Then, the adjacent tasks of the routes are connected with the shortest path between them. After the above decoding procedure, the two objectives mnv and tc can be directly calculated. The implicit task encoding scheme simplifies the design of crossover operator. However, it makes solution evaluation relatively time-consuming due to the computational complexity of the decoding procedure. Besides, a slight modification of the encoded solution may result in a severe change after decoding. This characteristic makes the encoded solution space more rugged and hinders local search. To overcome these drawbacks, an explicit task encoding scheme is developed and adopted by MARM.

In the explicit task encoding scheme, each edge task is first associated with two IDs, each for a direction. The IDs are different from each other, and chosen from the positive integer set $\mathcal{N}^+$. For an edge task, the two corresponding IDs have the same serving costs, deadheading costs, demand vectors, service frequencies and allowed period combination sets, which are exactly those of the edge task itself. The only differences between them are their head and tail vertices. To be specific, assuming a task (u, v) is associated with IDs t_1 and t_2, then $tv(t_1) = hv(t_2) = v$ and $hv(t_1) = tv(t_2) = u$, where $tv(t)$ and $hv(t)$ indicate the tail and head vertices of ID t. Besides, a depot loop 0 is defined to act as a delimiter between different routes. Its head and tail vertices are both v_0, and its serving cost, deadheading cost, demand vector and service frequency are 0 ($\mathbf{0}$).

Using the above IDs, a solution S can be represented as a set of task sequences $S = \{S_{ij} | i = 1, \ldots, p; j = 1, \ldots, m\}$, where $S_{ij} = (0, t_{ij1}, \ldots, t_{ijl_{ij}}, 0)$ presents the route j in period i. The depot loop 0 is inserted at the beginning and end of each task sequence to ensure that it starts and ends at the depot. l_{ij} is the number of tasks served in S_{ij}, and $l_{ij} = 0$ (i.e., $S_{ij} = (0, 0)$) indicates that vehicle j is not used in period i. Fig. 1 gives a simple illustration of the explicit task encoding scheme.


v2
v3
v4
v1
v0
v5
(a)
The given graph: all the edges are tasks, and v0 is the depot
Period 1
Period 2
12
13
11
2
14
8
3
9
(0,2,12,11,8,0) (0,3,14,13,0)
(0,3,14,13,9,0)
(b)
A PCARP solution: the solid lines indicate the services while the dashed lines indicate the deadheading paths
The IDs associated with both directions of the tasks

Task	ID	Task	ID
$\langle v_0,v_1\rangle$	1	$\langle v_1,v_0\rangle$	8
$\langle v_0,v_3\rangle$	2	$\langle v_3,v_0\rangle$	9
$\langle v_0,v_5\rangle$	3	$\langle v_5,v_0\rangle$	10
$\langle v_1,v_2\rangle$	4	$\langle v_2,v_1\rangle$	11
$\langle v_2,v_3\rangle$	5	$\langle v_3,v_2\rangle$	12
$\langle v_3,v_4\rangle$	6	$\langle v_4,v_3\rangle$	13
$\langle v_4,v_5\rangle$	7	$\langle v_5,v_4\rangle$	14

Fig. 1. Example of the explicit task encoding scheme. (a) The given graph: all the edges are tasks, and v_0 is the depot; (b) a PCARP solution: the solid lines indicate the services while the dashed lines indicate the deadheading paths.

Under the explicit task encoding scheme, the objectives mnv and tc can be directly calculated as:

$$mnv(S) = \max_{i\in\{1,\ldots,p\}} \left(\sum_{j=1}^{m} [l_{ij} > 0] \right) \tag{14}$$

$$tc(S) = \sum_{i=1}^{p}\sum_{j=1}^{m}\sum_{k=1}^{l_{ij}-1} \left(sc(t_{ijk}) + dist\left(t_{ijk}, t_{ij(k+1)}\right)\right) \tag{15}$$

where $dist(t_1, t_2)$ indicates the distance from t_1 to t_2, which is defined as the total cost of the shortest path from the tail vertex of t_1 to the head vertex of t_2. It can be obtained by Dijkstra's algorithm [12].

During the search process, the fitness function $fit(S)$ is defined as the weighted sum of mnv and tc, i.e.,

$$fit(S) = \alpha \cdot mnv(S) + tc(S). \tag{16}$$

In this minimization problem, the solution with smaller $fit(S)$ is considered to be better. To guarantee the priority of mnv, the weight α is set to a large constant throughout the search process. In the subsequent experiments, α is set to 100 000 so that a solution with a smaller mnv always has a smaller fit.

One advantage of the explicit task encoding scheme over the implicit task encoding scheme used in previous work is that it makes the fitness evaluation much less time-consuming. Concretely, in the explicit task encoding scheme, the worst-case computational complexity of the fitness evaluation is $O(pn)$, whereas evaluating a solution represented by the implicit task encoding scheme needs $O(pn^2)$ in the worst case. Besides, the explicit task encoding scheme also facilitates local search since no decoding procedure (e.g., the split procedure in the implicit task encoding scheme) is required.

It should be noted that under the explicit task encoding scheme, the capacity constraints may be violated and infeasible solutions may appear during the search. To compare between infeasible and feasible solutions, the total violated demand $tvd(S)$ is defined to measure the violation to the capacity constraints. It is defined as the summation of excessive demand of the infeasible routes, and can be calculated as:

$$tvd(S) = \sum_{i=1}^{p}\sum_{j=1}^{m} \max\{d(S_{ij}) - Q, 0\} \tag{17}$$

where $d(S_{ij})$ is the total demand of the tasks served by S_{ij}. $tvd(S)$ is to be minimized and a solution S is feasible if and only if $tvd(S) = 0$.

In MARM, the evaluation of solutions occurs in two different cases, one in the population update and the other during the local search. In the former case, when an offspring has been inserted in the population, the worst solution in the population is to be determined and removed from the population. To this end, we employ the stochastic ranking procedure [36], which has been demonstrated to be an efficient evaluation scheme for constrained optimization problems. Algorithm 4 gives the pseudo code of the stochastic ranking procedure in the case of a minimization problem, where N is the population size and P_f is a user-defined parameter. In MARM, we simply set P_f to 0.45, following the recommendation in [36]. After applying stochastic ranking procedure to *pop*, the last solution $pop_{(N)}$ is removed from *pop*.

Algorithm 4: Stochastic ranking procedure for minimization problem

```
1: for $i = 1$ to $N - 1$ do
2:   for $j = 1$ to $N - 1$ do
3:     Sample $r \in U(0, 1)$;
4:     if $tvd(pop_{(j)}) = tvd(pop_{(j+1)}) = 0$ or $r < P_f$ then
5:       if $fit(pop_{(j)}) > fit(pop_{(j+1)})$ then
6:         Call Swap$(pop)(j), pop_{(j+1)}$;
7:       end if
8:     else
9:       if $tvd(pop_{(j)}) > tvd(pop_{(j+1)})$ then
10:        Call Swap$(pop)(j), pop_{(j+1)}$;
11:      end if
12:    end if
13:  end for
14: end for
```

In the latter case, at each iteration of the local search, each neighboring solution is compared with the current solution and takes its place if better. In other words, a single solution in the neighborhood is to be selected. Thus, it is no longer appropriate to use stochastic ranking procedure since it can only maintain a set of relatively good solutions, but cannot tell which solution is the best. Here, a more efficient strategy is adopted. During the local search, solution S_1 is said to be better than solution S_2 if

- $tvd(S_1) < tvd(S_2)$ or
- $tvd(S_1) = tvd(S_2)$ and $fit(S_1) < fit(S_2)$.

The strategy has the following properties:

1) Feasible solutions are always better than infeasible solutions;
2) For infeasible solutions, the one with less constraint violation is better;
3) For feasible solutions or infeasible solutions with the same constraint violation, the one with smaller fitness value is better.

C. Solution Initialization

In the initialization phase, each initial solution is generated by the following three steps:

Step 1) Randomly choose a period combination for each task from its allowed period combination set;
Step 2) For each period, apply the path scanning heuristic [18] to all the task that should be served in that period to generate a corresponding single-period sub-solution;
Step 3) Combine all the single-period sub-solutions to form a complete PCARP solution.

The path scanning heuristic was proposed by Golden *et al.* [18] in 1983 for the Vehicle Routing Problem (VRP), which is the node routing counterpart of CARP, and was extended by Lacomme *et al.* in 2002 to solve CARP [26]. It has been demonstrated to be able to construct good CARP solutions in a very short time. Therefore, employing the path scanning heuristic in the solution initialization can generate good PCARP initial solutions and accelerate the convergence of MARM.

The solutions generated by the initialization procedure are feasible, since each sub-solution generated by the path scanning heuristic is feasible. Therefore, the feasibility of the final solution can be guaranteed, and the final solution should be no worse than the best initial solution.

D. Crossover Operator

Since a new solution representation scheme is employed in MARM, a corresponding crossover operator should be designed. To this end, we extend the Route-Based Crossover (RBX) operator [35] from the case of CARP to PCARP. Given two solutions S_1 and S_2, the extended operator, which is called the Periodic RBX (PRBX) operator, works as follows:

Step 1) Randomly sample a number $r \in \{1, \ldots, P\}$;
Step 2) Randomly select two routes R_1 and R_2, one from period r of S_1 and the other from period r of S_2;
Step 3) Set $S_x = S_1$;
Step 4) Replace R_1 of S_x by R_2, and remove the redundant tasks in other routes in period r of S_x;
Step 5) For each task served in R_1 and R_2, replace its period combination in S_x with that in S_2;
Step 6) For each period of S_x, remove the tasks whose period combinations have been changed and should no longer be served in that period;
Step 7) Insert the missed tasks into the periods of S_x according to their new period combinations. Note that the insertion of a task may increase the tc and tvd. Thus, each missed task is inserted in such a position that there is no other position that can induce both smaller tc and tvd. If more than one such position exists, one of them is chosen arbitrarily;
Step 8) Return S_x.

An example of the PRBX operator is given in Fig. 2.

From the above description, it can be seen that the computational complexity of the PRBX operator is dominated by Step 7), which is the most time-consuming step. In the worst case, the selected routes R_1 and R_2 involve all the n tasks and the period combinations of the tasks in S_1 and S_2 are totally different. That is, $\forall t \in T$ and $i \in \{1, \ldots, p\}$, $PC_i(t, S_1) \neq PC_i(t, S_2)$, where $PC_i(t, S)$ takes 1 if task t is served in period i of solution S, and 0 otherwise. In such situation, the total number of task services to be inserted in Step 7) is:

$$NS = \sum_{i=1}^{n} freq(t) = O(np).$$

For each task service, there are at most n candidate positions to insert. Therefore, the computational complexity of the PRBX operator is $O(NS \times n) = O(pn^2)$.

E. Local Search

After S_x has been generated by the PRBX operator, the RM procedure is applied to it to reduce its mnv. Then, S_x undergoes the local search process with a predefined probability P_{ls}. At each step of the local search process, the best solution S' in the neighborhood $\mathcal{N}(S)$ of the current solution S (initially set to S_x) is selected and compared with S. If it is better, then it becomes the current solution in the next step. Otherwise, the local search stops. The neighborhood $\mathcal{N}(S)$ is generated by the single-insertion, double-insertion and swap operators. Specifically, $\mathcal{N}(S)$ is defined as the set of the solutions that can be obtained by applying the single-insertion, double-insertion or swap operator to S. The above three operators are described as follows:

Single-insertion: move a task service from its original position to another;
Double-insertion: move two adjacent task services from their original positions to another;
Swap: exchange the positions of two task services.

In all the three operators, both directions of the involved tasks are considered. In the swap operator, the two task services must belong to different tasks. Note that the period combinations of the involved tasks may change due to the movements. Therefore, a task can only be moved to such positions that the new period combination is still in its allowed period combination set.

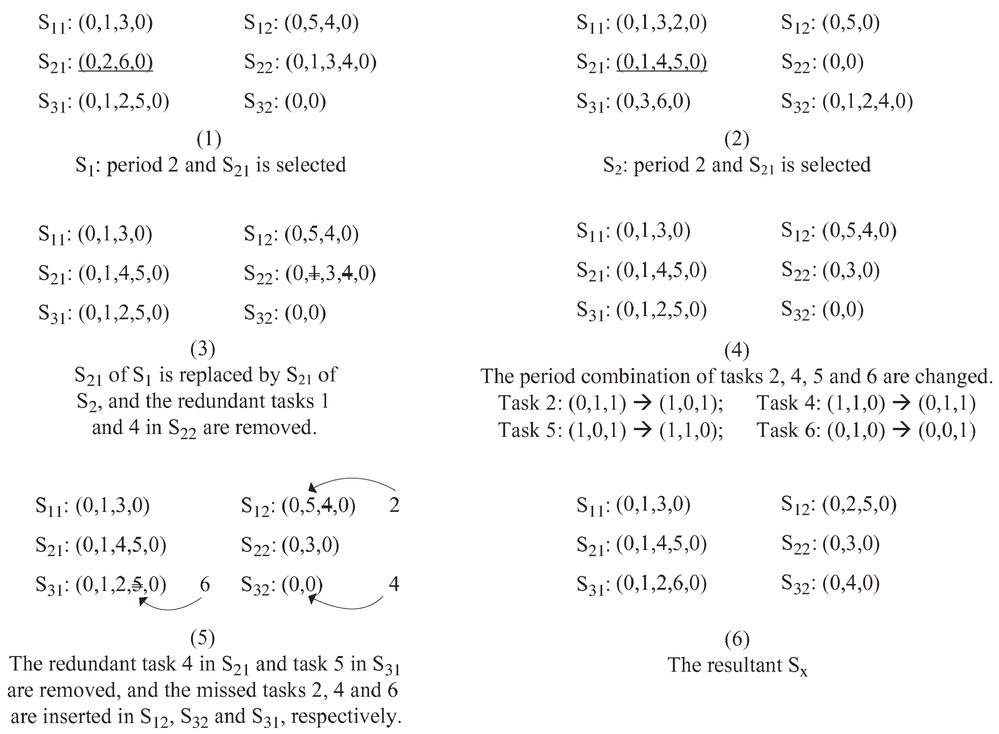

S11: (0,1,3,0) S12: (0,5,4,0)
S21: (0,2,6,0) S22: (0,1,3,4,0)
S31: (0,1,2,5,0) S32: (0,0)
(1)
S1: period 2 and S21 is selected
S11: (0,1,3,2,0) S12: (0,5,0)
S21: (0,1,4,5,0) S22: (0,0)
S31: (0,3,6,0) S32: (0,1,2,4,0)
(2)
S2: period 2 and S21 is selected
S11: (0,1,3,0) S12: (0,5,4,0)
S21: (0,1,4,5,0) S22: (0,~~1~~,3,~~4~~,0)
S31: (0,1,2,5,0) S32: (0,0)
(3)
S21 of S1 is replaced by S21 of S2, and the redundant tasks 1 and 4 in S22 are removed.
S11: (0,1,3,0) S12: (0,5,4,0)
S21: (0,1,4,5,0) S22: (0,3,0)
S31: (0,1,2,5,0) S32: (0,0)
(4)
The period combination of tasks 2, 4, 5 and 6 are changed.
Task 2: (0,1,1) → (1,0,1); Task 4: (1,1,0) → (0,1,1)
Task 5: (1,0,1) → (1,1,0); Task 6: (0,1,0) → (0,0,1)
S11: (0,1,3,0) S12: (0,5,~~4~~,0) 2
S21: (0,1,4,5,0) S22: (0,3,0)
S31: (0,1,2,~~5~~,0) 6 S32: (0,0) 4
(5)
The redundant task 4 in S21 and task 5 in S31 are removed, and the missed tasks 2, 4 and 6 are inserted in S12, S32 and S31, respectively.
S11: (0,1,3,0) S12: (0,2,5,0)
S21: (0,1,4,5,0) S22: (0,3,0)
S31: (0,1,2,6,0) S32: (0,4,0)
(6)
The resultant Sx

Fig. 2. Example of the PRBX operator.

F. Summary of MARM

Finally, MARM is summarized by being compared with LMA and SS. Briefly speaking, MARM is different from LMA and SS in the solution representation, crossover and the employment of the RM procedure. First, MARM adopts the explicit task encoding scheme while LMA and SS adopts the implicit task encoding scheme for solution representation. As a result, a solution in MARM can be directly evaluated within the computational complexity of $O(pn)$, while LMA and SS require a decoding process whose computational complexity is $O(pn^2)$. Second, for crossover, MARM employs the PRBX operator, while LMA and SS employ another operator called the PLOX operator [10], [27], which is only applicable for implicitly encoded solutions. According to the discussions in Section IV-D and the descriptions of the PLOX operator in [10], [27], the computational complexities of the PRBX and PLOX operator are $O(pn^2)$ and $O(pn)$, respectively. The complexity $O(pm)$ of the RM operator employed in MARM can be neglected because $m \ll n$ in practice. Recalling that a generation of MA is composed of a crossover, a local search process and an evaluation phase, and the local search process of all the compared algorithms are the same, the overall computational complexity of one generation of the compared algorithms are comparable. However, compared with the implicit task encoding scheme, the explicit task encoding scheme facilitates the local search since there is no solution transformation needed during the local search process. Finally, and the most importantly, MARM employs the newly developed RM procedure to deal with the primary objective mnv, while LMA and SS do not. The employment of the RM procedure is expected to make MARM able to obtain solutions with smaller mnv's than LMA and SS.

V. Experimental Studies

Two sets of experiments have been carried out to evaluate the performance of MARM. The first experiment is carried out on two relatively simple test sets, i.e., the $pgdb$ and $pval$ test sets [10] generated from the corresponding gdb and val CARP benchmark sets. MARM is applied to them and the results are compared with that obtained by LMA (provided in [26]) and SS (provided in [10]). Besides, to observe the influence of the RM procedure on the performance of MARM, we remove it from the framework of MARM and compare the resultant algorithm with MARM on the $pgdb$ and $pval$ test sets.

The second experiment is carried out on a problem set generated from real-world data to evaluate the performance of MARM on real-world applications. The real-world data set was first generated by Brandão and Eglese [7] and consists of 10 large CARP instances defined on a road network in Lancashire, U.K. Here, these single-period instances are extended to multi-period ones in the same way as the gdb and val test sets are extended to the $pgdb$ and $pval$ test sets. Then, LMA, SS and MARM are applied to the problem set and the results are compared.

A. Experimental Setup

The $pgdb$ and $pval$ test sets used in the first experiment were generated by Chu *et al.* in [10] by extending two well-known CARP benchmark sets, i.e., the gdb and val sets from single-period case to multi-period case. The gdb set was proposed by Golden *et al.* in 1983 [18]. It consists of 23 small and medium undirected CARP instances. The val set was proposed by Benavent *et al.* in 1992 [5] and contains 34 larger undirected CARP instances based on 10 different graphs. Different

TABLE II
PARAMETER SETTINGS OF MARM

Name	Description	Value
α	The weight of the mnv in the fitness function	100000
$popsize$	Population size	30
$Mtrial$	Maximum trials for generating non-clone initial solutions	10
P_{ls}	Probability of carrying out local search (mutation)	0.1
G_{max}	Maximum number of generations	10000 (in the first experiment) 500 (in the second experiment)

instances based on each graph were generated by changing the capacity of vehicles.

The periodic extension of these instances were generated to simulate a weekly waste collection. Therefore, the horizon was defined as a week, with Saturday and Sunday as idle days. For each task (u, v), the service frequency is defined as $freq(u, v) = 1 + (u + v)$ mod 5, and the demand vector is defined as $\boldsymbol{d}(u, v) = (d_1(u, v), \ldots, d_7(u, v))$, where $d_i(u, v) = d(u, v)$ is the demand of (u, v) in the original CARP instance. If the service frequency of (u, v) is 2 or 3, then consecutive services over the horizon (e.g., (Monday, Tuesday) and (Monday, Tuesday, Wednesday)) are forbidden. Otherwise, all possible period combinations are allowed. After the extension, the accumulated demand of a single task may exceed the capacity. To avoid such situation, the capacities were duplicated (by 2, 3 or 4 depending on the instance).

In addition to the $pgdb$ and $pval$ test sets, Lacomme *et al.* extended the egl CARP test set to $pegl$ PCARP test set [27]. Unfortunately, this set is unavailable online, nor could we implement it due to the ambiguous description provided in the original literature. Thus, we were unable to conduct experimental studies on this set.

The real-world data set used in the second experiment, called the pG set, is extended from the G set generated by Brandão and Eglese in [7], which consists of 10 large CARP instances based on a road network in Lancashire, U.K. The G set consists of two groups, i.e., the $G1$ and $G2$ groups, each with 5 instances (denoted as 1-A $\sim$ 1-E and 2-A $\sim$ 2-E). The instances in the same group have the same task set, but different capacities. When extending the G instances to pG instances, the service frequencies, demand vectors and allowed period combinations of the tasks are defined in the same way as in the $pgdb$ and $pval$ test sets. Besides, the capacity of each instance is multiplied by a constant to prevent the accumulated demand from exceeding the capacity (3 for 1-A, 1-B, 2-A and 2-B, 4 for 1-C and 2-C, and 5 for 1-D, 1-E, 2-D and 2-E). After the duplication, the capacities of 1-D, 1-E, 2-D and 2-E are quite close to that of 1-A, 1-B, 2-A and 2-B. Thus, experiments on them are omitted. All the test sets can be downloaded from http://nical.ustc.edu.cn/ymei/.

Table II summarizes the parameter settings of MARM used in the experiments. They are set in such a way that a relatively good compromise between solution quality and runtime can be obtained. To make a fairer comparison, in the second experiment, the G_{max} of LMA and SS have been shortened to make their runtime comparable with that of MARM. All the compared algorithms were implemented for 30 independent runs and the best and average results obtained are reported.

B. Experimental Results on the Benchmark Sets

In the first experiment, MARM is compared with LMA and SS on the $pgdb$ and $pval$ test sets. As mentioned in [27], LMA has three different versions namely PMA, DMA and IPMA. In PMA, the proposed MA ingredients are directly applied on the given instance. In DMA, the services of each task are first heuristically assigned over the horizon. Then, the MA is applied on each period for solving the corresponding single-period subproblem. IPMA can be seen as the combination of PMA and DMA. It conducts PMA and DMA sequentially, i.e., it takes the final population of PMA as the initial population of DMA. In our study, MARM is compared with all the above three LMA versions and SS. Unfortunately, LMA was not tested on the $pval$ set. Therefore, we take SS as the only compared approach for the $pval$ set. All the experimental results can be downloaded from nical.ustc.edu.cn/ymei/publications.htm.

The experimental results are shown in Tables III and IV. The columns headed "$|V|$," "$|E|$" and "$|S|$" indicate the number of vertices, edges and the services required over the horizon. For all the test instances, all the edges are tasks and thus the number of task equals the number of edges. The columns headed "LBV" and "LB" stand for the lower bound of the mnv and tc. LBV can be obtained by (13), while LB was provided by Chu *et al.* in [10]. For PMA, DMA, IPMA and SS, the results are directly collected from the original literatures, as shown besides the algorithm names in the tables. For MARM, the columns headed "Average" and "Best" stand for the average performance and best performance. The average performance is presented in the form of "mean $\pm$ standard deviation" of the results obtained by the 30 runs, while the best performance is shown by the best result obtained by the 30 runs. The column headed "BK" indicates the best known results for each instance, which have been obtained by sophisticated settings of LMA, SS and the ant colony system, and are collected from [10], [24], [27]. Three additional rows are included at the bottom of the tables. The "Mean" row presents the mean values of the results obtained by the compared algorithms over all the instances in the test set. The mean LBV and LB are also calculated for reference. The "No.Best" row counts for each algorithm and the best known result the number of instances on which the algorithm (best known result) has achieved the best result among all the compared results. Finally, the "APD" row indicates the Average Percentage Deviation (APD) from the lower bounds (LBV for the mnv and LB for the tc) for each algorithm. The optimal results are with "$*$," and the new best mnv's and tc's obtained by MARM are marked in bold.

The efficacy of MARM can be assessed from two aspects, i.e., the best and average performance it achieved over the

TABLE III
EXPERIMENTAL RESULTS OF THE COMPARED ALGORITHMS ON THE *pgdb* TEST SET. THE OPTIMAL RESULTS ARE WITH "∗," AND THE NEW BEST RESULTS FOUND BY MARM ARE IN BOLD

Name	$\|V\|$	$\|E\|$	$\|S\|$	LBV	LB	LMA ([27])				SS ([10])		MARM				BK ([10] [24] [27])	
							PMA	DMA	IPMA			Average		Best			
						mnv	tc	tc	tc	mnv	tc	mnv	tc	mnv	tc	mnv	tc
1	12	44	65	3	771	3*	894	890	890	3*	869	3.0 ± 0.0	827.4 ± 8.6	3*	**810**	3*	845
2	12	52	76	3	887	3*	1040	1030	1030	3*	997	3.0 ± 0.0	934.6 ± 12.7	3*	**917**	3*	973
3	12	44	61	3	673	3*	794	803	794	3*	788	3.0 ± 0.0	710.4 ± 9.9	3*	**691**	3*	750
4	11	38	52	2	673	2*	858	858	858	2*	795	2.0 ± 0.0	762.1 ± 12.7	2*	**740**	2*	795
5	13	52	75	3	965	3*	1123	1123	1120	3*	1098	3.0 ± 0.0	1032.4 ± 14.2	3*	**1004**	3*	1078
6	12	44	67	3	861	3*	1009	1012	1008	3*	951	3.0 ± 0.0	912.2 ± 7.9	3*	**900**	3*	926
7	12	44	65	3	801	3*	941	950	935	3*	882	3.0 ± 0.0	836.0 ± 8.1	3*	**819**	3*	866
8	27	92	143	5	814	5*	1021	1041	1000	6	1014	5.0 ± 0.0	986.0 ± 14.7	5*	**953**	5*	1000
9	27	102	155	5	772	5*	972	939	934	6	966	5.0 ± 0.0	917.3 ± 11.5	5*	**892**	5*	934
10	12	50	65	2	669	2*	765	743	744	2*	761	2.0 ± 0.0	696.3 ± 9.1	2*	**677**	2*	720
11	22	90	133	3	1043	3*	1204	1174	1172	3*	1193	3.0 ± 0.0	1113.4 ± 11.0	3*	**1089**	3*	1154
12	13	46	67	3	1021	3*	1218	1221	1218	3*	1245	3.0 ± 0.0	1149.8 ± 14.5	3*	**1118**	3*	1179
13	10	56	81	3	1541	3*	1618	1606	1597	3*	1593	3.0 ± 0.0	1564.9 ± 5.3	3*	**1555**	3*	1574
14	7	42	64	2	278	3	300	307	300	3	285	2.5 ± 0.5	291.0 ± 8.7	**2***	290	3	282
15	7	42	64	2	174	2*	182	186	182	2*	180	2.0 ± 0.0	176.2 ± 1.5	2*	**174**	2*	176
16	8	56	85	3	352	3*	379	374	374	3*	372	3.0 ± 0.0	364.2 ± 2.1	3*	**360**	3*	364
17	8	56	85	2	255	3	267	273	267	3	265	2.0 ± 0.0	266.1 ± 2.0	**2***	261	3	261
18	9	72	106	2	484	3	506	507	506	3	506	2.0 ± 0.0	494.4 ± 3.3	**2***	**487**	3	494
19	8	22	30	2	155	2*	181	179	181	2*	173	2.0 ± 0.0	172.7 ± 0.8	2*	**171**	2*	173
20	11	44	63	2	339	3	375	375	375	3	356	2.0 ± 0.0	357.8 ± 4.1	**2***	**348**	3	352
21	11	66	101	3	488	3*	533	526	525	4	509	3.0 ± 0.0	504.9 ± 3.8	3*	**498**	3*	514
22	11	88	129	4	578	4*	608	598	600	5	597	4.0 ± 0.0	593.0 ± 2.7	4*	**589**	4*	595
23	11	110	165	5	671	5*	709	690	691	6	702	5.0 ± 0.0	691.4 ± 3.1	5*	**686**	5*	690
Mean	-	-	-	2.96	663.70	3.13	760.74	756.74	752.22	3.35	743.35	2.98	711.06	2.96	696.91	3.13	725.87
No.Best	-	-	-	-	-	19	0	0	0	14	0	22	0	23	22	19	2
APD	-	-	-	-	-	8.70	13.57	13.19	12.48	13.84	10.73	1.16	6.77	0.00	4.75	8.70	8.53

30 runs. First, we evaluate the best performance of MARM. From the column headed "Best", it can be observed that the best performance of MARM was better than LMA, SS, and even the best known results. The APD values of both the mnv and tc under the "Best" column are less than that of the other compared results on both the $pgdb$ and $pval$ test sets. Furthermore, the APD value of the best mnv obtained by MARM is 0.00 for both $pgdb$ and $pval$ test sets, which indicates that MARM reached optimal mnv's for all the test instances. From the "No.Best" row, it can be seen that MARM managed to obtain solutions better than the best known solutions for all the 57 benchmark instances. To be specific, for the 21 instances where the previous approaches failed to achieve optimal mnv's, MARM managed to find the solutions with optimal mnv's. For the other 36 instances on which the best known solutions are with optimal mnv's, MARM obtained solutions with not only optimal mnv's, but also smaller tc's than the best known tc's.

Then, MARM is compared with other algorithms with respect to the average performance. In this case, the best results of LMA and SS also should not be taken into account because they are obtained by testing various parameter settings and choosing the best one. Hence, the approaches with standard parameter settings that are fixed for all the test instances are adopted. Unfortunately, it was not stated that in the original papers ([27] and [10]) that the reported results were the averaged or the best results over a number of independent runs. However, from the observation of the columns headed "Average," it is found that the average performance of MARM was still much better than the reported results of LMA and SS, in term of both mnv and tc. For the $pgdb$ test set, the APD values of the average mnv and tc obtained by MARM is 1.16 and 6.67, both of which are better than that of the best known solutions (8.70 and 8.53). For the $pval$ test set, MARM obtained average mnv APD value of 1.50, and average tc APD value of 17.72. They are both better than the corresponding APD values of the best known solutions (17.68 and 23.03).

In addition to the solution quality on mnv and tc, it is natural to evaluate the stability of MARM, i.e., the probability that MARM performs well. From the experiments, it is found that for each instance, MARM always obtained solutions with mnv's no worse than the best mnv obtained by the compared algorithms in the 30 runs. Besides, MARM consistently achieved best solutions in terms of fitness for all the 34 $pval$ instances. Therefore, we can conclude that MARM was quite stable to obtain good solutions.

Finally, the computation time is compared. Table V presents the average runtime (in seconds) of each compared algorithm on the two test sets. The results of the LMAs are collected from [27], while those of SS are collected from [10]. Note that the LMAs have not been applied to $pval$ set and then the runtime of PMA, DMA and IPMA on that set are unavailable. In our experiments, MARM was coded in C and run on Intel(R) Xeon(R) E5335 2.00 GHz. Since the compared algorithms were run on different computers, normalization is required to make the comparison fairer. SS was coded by a Delphi 5 and executed on a Pentium 4 2.40 GHz, while PMA, DMA and IMPA were run on a Pentium 4 1.40 GHz. Besides, the runtime reported in the original papers were in minutes. Hence, the runtime of the three LMA versions and SS were multiplied by $1.40/2.00 \times 60 = 42$ and $2.40/2.00 \times 60 = 72$, respectively.

TABLE IV
EXPERIMENTAL RESULTS OF THE COMPARED ALGORITHMS ON THE *pgdb* TEST SET. THE OPTIMAL RESULTS ARE WITH "$*$," AND THE NEW BEST RESULTS FOUND BY MARM ARE IN BOLD

Name	$\|V\|$	$\|E\|$	$\|S\|$	LBV	LB	SS ([10])		MARM				BK ([10])	
								Average		Best			
						mnv	tc	mnv	tc	mnv	tc	mnv	tc
1a	24	78	105	2	434	2*	544	2.0 ± 0.0	488.8 ± 6.6	2*	**470**	2*	518
1b	24	78	105	3	458	3*	585	3.0 ± 0.0	540.0 ± 4.7	3*	**530**	3*	568
1c	24	78	105	4	532	5	701	4.0 ± 0.0	678.5 ± 11.6	**4***	**653**	5	693
2a	24	68	94	2	606	2*	782	2.0 ± 0.0	706.0 ± 5.0	2*	**697**	2*	740
2b	24	68	94	2	695	3	868	2.0 ± 0.0	788.8 ± 6.7	2*	**775**	2*	878
2c	24	68	94	4	1012	5	1259	4.0 ± 0.0	1183.3 ± 16.1	4*	**1149**	4*	1254
3a	24	70	96	2	192	2*	249	2.0 ± 0.0	225.9 ± 2.3	2*	**222**	2*	243
3b	24	70	96	2	212	3	278	2.0 ± 0.0	263.8 ± 5.0	**2***	**255**	3	274
3c	24	70	96	4	289	4*	358	4.0 ± 0.0	347.1 ± 4.5	4*	**336**	4*	350
4a	41	138	205	2	1089	3	1330	2.0 ± 0.0	1262.4 ± 18.6	**2***	**1228**	3	1320
4b	41	138	205	3	1113	4	1471	3.0 ± 0.0	1314.7 ± 14.0	3*	**1288**	3*	1422
4c	41	138	205	4	1205	4*	1583	4.0 ± 0.0	1438.7 ± 13.6	4*	**1409**	4*	1500
4d	41	138	205	6	1502	8	1988	6.1 ± 0.3	1905.6 ± 31.0	**6***	**1858**	7	1930
5a	34	130	194	2	1184	3	1454	2.0 ± 0.0	1353.3 ± 18.8	**2***	**1315**	3	1414
5b	34	130	194	3	1241	4	1528	3.0 ± 0.0	1417.6 ± 15.6	**3***	**1384**	4	1508
5c	34	130	194	4	1348	4*	1655	4.0 ± 0.0	1541.9 ± 9.6	4*	**1522**	4*	1652
5d	34	130	194	6	1638	7	2097	6.0 ± 0.0	2033.3 ± 21.9	**6***	**1991**	7	2058
6a	31	100	150	2	658	3	754	2.0 ± 0.0	741.4 ± 8.5	**2***	**722**	3	754
6b	31	100	150	3	684	4	863	3.0 ± 0.0	786.4 ± 7.5	3*	**774**	3*	836
6c	31	100	150	7	921	8	1183	7.0 ± 0.0	1139.0 ± 11.3	**7***	**1117**	8	1153
7a	40	132	201	2	833	3	1040	2.0 ± 0.2	997.7 ± 24.4	**2***	**966**	3	1008
7b	40	132	201	3	835	4	1046	3.0 ± 0.0	978.5 ± 11.2	**3***	**960**	4	1018
7c	40	132	201	7	949	8	1283	7.0 ± 0.0	1190.8 ± 14.5	7*	**1165**	7*	1271
8a	30	126	194	2	1150	3	1343	2.7 ± 0.5	1282.3 ± 47.1	**2***	**1292**	3	1313
8b	30	126	194	3	1197	4	1453	3.0 ± 0.0	1330.4 ± 14.8	**3***	**1301**	4	1371
8c	30	126	194	7	1567	7*	1970	7.0 ± 0.0	1886.0 ± 14.6	7*	**1853**	7*	1929
9a	50	184	274	2	867	3	1083	2.0 ± 0.2	990.0 ± 13.4	**2***	**966**	3	1034
9b	50	184	274	3	891	4	1087	3.0 ± 0.0	1014.0 ± 9.6	3*	**990**	3*	1103
9c	50	184	274	4	913	4*	1195	4.0 ± 0.0	1050.9 ± 12.3	4*	**1031**	4*	1148
9d	50	184	274	7	1064	8	1418	7.0 ± 0.0	1367.2 ± 23.4	**7***	**1324**	8	1390
10a	50	194	300	2	1247	3	1478	2.3 ± 0.4	1401.8 ± 31.6	**2***	**1385**	3	1452
10b	50	194	300	3	1259	4	1580	3.0 ± 0.0	1415.9 ± 11.2	3*	**1395**	3*	1508
10c	50	194	300	4	1301	4*	1631	4.0 ± 0.0	1477.1 ± 11.0	4*	**1461**	4*	1555
10d	50	194	300	7	1551	9	1964	7.0 ± 0.0	1879.8 ± 21.0	**7***	**1837**	8	1913
Mean	-	-	-	3.62	959.91	4.38	1208.85	3.65	1129.97	3.62	1106.50	4.12	1178.76
No.Best	-	-	-	-	-	10	0	29	0	34	34	17	0
APD	-	-	-	-	-	25.14	26.12	1.50	17.72	0.00	15.14	17.68	23.03

TABLE V
RUNTIME (IN CPU SECONDS) OF THE COMPARED ALGORITHMS ON THE TEST SETS

Test Set	PMA	DMA	IPMA	SS	MARM
pgdb	125.3	197.5	326.7	133.9	12.2
pval	-	-	-	531.5	52.6

It should be noted that the computation time depends on many other factors, such as operating systems, compilers, coding skills of the programmer, etc. Thus, the comparison is meant to be indicative. However, we can still believe that MARM is indeed much faster than the compared algorithms since the difference between MARM and other algorithms is quite significant even if only the CPU clock speed has been taken into account. It is worth noting that MARM stopped much earlier than the compared algorithms (e.g., it stopped after 10 000 crossovers while PMA stopped after 40 000 crossovers), but can still provide much better solutions. Therefore, we can conclude that MARM is much more efficient as better solutions can be obtained much earlier during the search process.

C. Investigation of the Influence of the RM Procedure

In contrast with other search operators that are either extended from the single-period situation or just directly employed, the RM procedure is specifically developed for PCARP to deal with mnv that has not been considered in CARP. Thus, it is important to examine its effect on the performance of MARM. For this purpose, we remove the RM procedure from the framework of MARM, and compare the resultant MA with MARM on the $pgdb$ and $pval$ test sets. The mean and standard deviation of the mnv's and tc's of the solutions obtained by the 30 independent runs of the two MAs on the $pgdb$, $pval$ and both test sets (denoted as "Overall") are presented in Table VI. It can be found that with RM embedded, solutions with smaller mnv's can be obtained consistently, while the average tc is comparable.

Then, the Wilcoxon rank sum test is conducted between the results of the two compared algorithms over 30 independent runs to make a more precise comparison. Table VII shows the results of the statistical comparison. In the table, for mnv, tc and fitness, the corresponding "W" and "L" columns indicate the number of instances on which embedding RM leads to

TABLE VI
COMPARISON BETWEEN THE MA WITH AND WITHOUT RM ON THE TEST SETS

Test set	Without RM		With RM	
	mnv	tc	mnv	tc
pgdb	3.2 ± 0.1	712.3 ± 9.2	3.0 ± 0.0	711.1 ± 7.5
pval	4.2 ± 0.3	1127.7 ± 25.6	3.6 ± 0.0	1130.0 ± 14.5
Overall	3.7 ± 0.2	920.0 ± 17.4	3.3 ± 0.0	920.5 ± 11.0

TABLE VII
STATISTICAL COMPARISON BETWEEN THE RESULTS OBTAINED BY 30 INDEPENDENT RUNS OF MAs WITH AND WITHOUT RM

Test set	mnv			tc			fitness		
	W	D	L	W	D	L	W	D	L
pgdb	9	14(14)	0	2	16	5	10	13	0
pval	25	9(6)	0	3	19	12	27	7	0
Overall	34	23(20)	0	5	35	17	37	20	0

significantly (with 95% confidence probability) better and worse results, while "D" indicates the number of instances where there is no statistically significant difference between the MAs with and without RM (the numbers in the parenthesis under the "mnv" column indicate the number of instances on which the two compared algorithms both obtained optimal mnv's consistently in 30 runs).

From Table VII, it can be found that embedding RM led to significant improvement on mnv on 9 $pgdb$ and 25 $pval$ instances, while never obtained significantly worse mnv. There are 14 $pgdb$ and 9 $pval$ instances on which there is no significant difference between the results of the MAs with and without RM. However, most of them are so simple that the MA without RM already reached the optimal mnv's consistently, and thus the RM procedure is not necessary for them. Due to the priority of the mnv, the advantage on the mnv always led to advantage on the fitness. This is consistent with the fact observed from Table VII that embedding RM significantly improved the fitness on 37 out of the 57 instances, and never got significantly worse fitness. In summary, we can conclude that embedding RM can enhance the ability of the algorithm to find better solutions, especially those with smaller mnv's.

D. Results on the Real-World Data Set

For the pG real-world data set, LMA, SS, and MARM were implemented for 30 independent runs, and their best and average performance are shown in Tables VIII and IX. For each compared algorithm, Table VII gives the mnv and tc of the best solution obtained in the 30 independent runs, while Table IX presents the mean and standard deviation of the mnv and tc of the 30 final solutions obtained by the 30 runs (in the same form as in Tables III and IV). Besides, the properties of each instance, including the number of vertices, edges and services over the horizon and LBV are provided. For these newly generated instances, the lower bounds of tc are unavailable. In Table VIII, the optimal mnv's are with "*". In Table IX, the Wilcoxon rank sum test was conducted on the mnv's and tc's, and the ones obtained by MARM that are significantly better (with confidence probability of 95%) than that of LMA and SS are in bold.

First, the best performance of MARM is evaluated. From Table VIII, it is seen that MARM was able to achieve solutions with smaller mnv than that of LMA and SS on all the 6 pG instances. Furthermore, MARM managed to obtain solutions with optimal mnv on 3 pG instances. This verifies the superiority of MARM in optimizing the primary objective mnv. Although the solutions of MARM have higher tc than that of LMA and SS, their fitness are still better due to the advantage on mnv.

Then, we focus on the average performance. From Table IX, it can be observed that for each instance, both the mean mnv and tc of the solutions obtained by MARM are smaller than that of the solutions obtained by LMA and SS. Moreover, MARM obtained solutions with significantly smaller mnv than LMA and SS for all the 6 pG instances.

In summary, the conclusion drawn from the pG real-world data set is consistent with that of the $pgdb$ and $pval$ benchmark sets, which demonstrated the advantage of MARM over LMA and SS in terms of solution quality, especially on the primary objective mnv.

E. Summary and Further Discussion

The experimental studies showed that MARM outperformed the other two compared algorithms significantly, especially in terms of the primary objective mnv. By comparing MARM to the MA without RM, one may find that the appealing performance of MARM should be attributed to the RM procedure, which was specifically designed for reducing mnv. Besides, it was found that the average tc obtained by MARM was also smaller than those of the compared algorithms. The reason is that MARM adopts the explicit task encoding scheme, which facilitates the local search process. Therefore, smaller tc's can be obtained through the more efficient local search process.

Since CARP is a special case of PCARP, one might wonder whether MARM can still perform well on CARP. However, MARM was specifically designed to cope with the new objective introduced by PCARP, and the only objective of CARP, the total cost, is not the major concern of it. Hence, it is unlikely that MARM can compete with those state-of-the-art algorithms on CARP. In [10], empirical study was carried out to verify a similar hypothesis, and showed that SS performed much worse than the state-of-the-art approaches for CARP. This observation indirectly implies that MARM is not an ideal approach to CARP. Thus, we suggest practitioners employ MARM only in the case of PCARP.

VI. CONCLUSION AND FUTURE WORK

In this paper, we investigate PCARP, which is a generalization of CARP over a multi-period horizon introduced very recently in [25] to model the real-world applications such as the urban waste collection more closely. If the horizon consists of only one period, PCARP will reduce to CARP. A major difference between PCARP and CARP is that tc becomes an auxiliary objective in PCARP, while the new primary objective mnv is taken into account. Unlike the properties commonly considered in CARP such as tc and tvd, mnv is not so sensitive to the traditional search operators for solving CARP. As a result, the approaches that employ extended versions of traditional search operators will be ineffective in obtaining solutions with small mnv's.

Based upon the above observations, we propose a new MA, i.e., MARM. It adopts an explicit solution representation

TABLE VIII
BEST PERFORMANCE OF LMA, SS, AND MARM ON THE pG REAL-WORLD DATA SET. THE OPTIMAL mnv'S ARE WITH "*"

Test set	$\|V\|$	$\|E\|$	$\|S\|$	LBV	LMA		SS		MARM	
					mnv	tc	mnv	tc	mnv	tc
G1-A	255	347	1062	10	11	3623482	12	3513307	10*	3678504
G1-B	255	347	1062	12	14	3874996	15	3808003	13	3947359
G1-C	255	347	1062	11	13	3690919	14	3710681	11*	3799614
G2-A	255	375	1138	11	12	3820365	14	3814253	11*	3923292
G2-B	255	375	1138	13	15	4118258	16	4108672	14	4235707
G2-C	255	375	1138	11	14	4019023	15	3949113	13	4103535
Mean	-	-	-	11.3	13.2	3857840.5	14.3	3817338.2	12.0	3948001.8

TABLE IX
AVERAGE PERFORMANCE OF LMA, SS, AND MARM ON THE pG REAL-WORLD DATA SET. THE mnv'S AND tc'S OBTAINED BY MARM THAT ARE SIGNIFICANTLY BETTER THAN THAT OF LMA AND SS (WITH CONFIDENCE PROBABILITY OF 95%) ARE IN BOLD

Test set	$\|V\|$	$\|E\|$	$\|S\|$	LBV	LMA		SS		MARM	
					mnv	tc	mnv	tc	mnv	tc
G1-A	255	347	1062	10	12.1±0.6	3846199.2±204563.0	13.1±0.4	4191076.8±589288.5	**11.0±0.4**	3758286.1±48305.9
G1-B	255	347	1062	12	15.1±0.8	4170835.4±223624.0	16.3±0.6	4319454.0±457226.7	**14.0±0.7**	4093805.0±67843.8
G1-C	255	347	1062	11	13.6±0.6	4034616.1±185334.1	14.8±0.4	4230602.5±593157.0	**12.3±0.7**	3935364.7±66648.1
G2-A	255	375	1138	11	13.5±0.8	4189723.8±233001.0	14.6±0.5	4306027.1±504355.2	**12.3±0.5**	4051017.5±56278.0
G2-B	255	375	1138	13	16.3±0.8	4515713.9±273187.1	17.6±0.6	4584540.6±511354.6	**15.0±0.8**	4411075.3±81881.3
G2-C	255	375	1138	11	14.6±0.6	4362438.9±299696.1	15.8±0.5	4493359.9±586055.1	**13.4±0.5**	4210976.6±55288.2
Mean	-	-	-	11.3	14.2	4186587.9	15.4	4354176.8	13.0	4076754.2

scheme rather than the implicit representation employed by previous approaches, and thus employs a newly designed multi-period crossover operator for such representation. Most importantly, the RM procedure is proposed and embedded between the crossover and local search process of MARM so that the objective mnv can be tackled separately from other properties including tc and tvd. To be specific, a solution is firstly improved in mnv by merging the routes of the periods in which there are more routes than expected, and then improved with respect to tc and tvd by the subsequent local search process. Thanks to the insensitivity of mnv to the local search operators, it is unlikely that mnv deteriorates during the local search process. The efficacy of MARM and the importance of RM in improving mnv have been demonstrated by experimental studies.

In this paper, the efficacy of MARM has been validated on the basic version of PCARP, which is based on an undirected graph. However, according to the representation scheme and the search operators adopted by MARM, it can be extended to mixed PCARP with little effort. As there is no standard test instance for mixed PCARP, any further research on this topic should be done by first generating such test instances.

As most algorithms for other combinatorial optimization problems, MARM is specifically designed for PCARP and requires reasonable modification when applied to other problems, even other variants of CARP. However, the success of MARM can shed a light on solving a class of problems in which some properties are much less sensitive to the search operators than other properties (e.g., mnv v.s. tc and tvd in PCARP). For such problems, it may be more appropriate to deal with the properties separately instead of simultaneously. By means of improving the insensitive properties before improving other sensitive properties, better solutions may be obtained through a more efficient search process.

In our study, the mnv is considered as the primary objective of PCARP. However, in other situations, the primary objective might be different, e.g., minimizing the number of vehicles in surplus when the vehicle fleet has already been purchased. In the future, we will consider including more objectives in the problem formulation to make it even closer to reality.

REFERENCES

[1] F. Alonso, M. Alvarez, and J. Beasley, "A tabu search algorithm for the periodic vehicle routing problem with multiple vehicle trips and accessibility restrictions," *J. Oper. Res. Soc.*, vol. 59, no. 7, pp. 963–976, Jul. 2008.

[2] A. Amberg and S. Voss, "A hierarchical relaxations lower bound for the capacitated arc routing problem," in *Proc. 35th Annu. Hawaii Int. Conf. Syst. Sci.*, 2002, pp. 1415–1424.

[3] E. Angelelli and G. Speranza, "The periodic vehicle routing problem with intermediate facilities," *Eur. J. Oper. Res.*, vol. 137, no. 2, pp. 233–247, Mar. 2002.

[4] J. Belenguer and E. Benavent, "A cutting plane for the capacitated arc routing problem," *Comput. Oper. Res.*, vol. 30, no. 5, pp. 705–728, 2003.

[5] E. Benavent, V. Campos, A. Corberán, and E. Mota, "The capacitated arc routing problem: Lower bounds," *Networks*, vol. 22, no. 7, pp. 669–690, Dec. 1992.

[6] P. Beullens, L. Muyldermans, D. Cattrysse, and D. V. Oudheusden, "A guided local search heuristic for the capacitated arc routing problem," *Eur. J. Oper. Res.*, vol. 147, no. 3, pp. 629–643, Jun. 2003.

[7] J. Brandão and R. Eglese, "A deterministic tabu search algorithm for the capacitated arc routing problem," *Comput. Oper. Res.*, vol. 35, no. 4, pp. 1112–1126, Apr. 2008.

[8] F. Chu, N. Labadi, and C. Prins, "The periodic capacitated arc routing problem: Linear programming model, metaheuristic and lower bounds," *J. Syst. Sci. Syst. Eng.*, vol. 13, no. 4, pp. 423–435, Dec. 2004.

[9] F. Chu, N. Labadi, and C. Prins, "Heuristics for the periodic capacitated arc routing problem," *J. Intell. Manuf.*, vol. 16, no. 2, pp. 243–251, Apr. 2005.

[10] F. Chu, N. Labadi, and C. Prins, "A scatter search for the periodic capacitated arc routing problem," *Eur. J. Oper. Res.*, vol. 169, no. 2, pp. 586–605, Mar. 2006.

[11] J. F. Cordeau, M. Gendreau, and G. Laporte, "A tabu search heuristic for periodic and multi-depot vehicle routing problems," *Networks*, vol. 30, no. 2, pp. 105–119, Sep. 1997.

[12] E. W. Dijkstra, "A note on two problems in connection with graphs," *Numer. Math.*, vol. 1, pp. 269–271, 1959.

[13] M. Dror, *Arc Routing. Theory, Solutions and Applications*. Boston, MA: Kluwer, 2000.

[14] L. Drummond, L. Ochi, and D. Vianna, "An asynchronous parallel metaheuristic for the period vehicle routing problem," *Future Gener. Comput. Syst.*, vol. 17, no. 4, pp. 379–386, Jan. 2001.

[15] P. Francis and K. Smilowitz, "Modeling techniques for periodic vehicle routing problems," *Transp. Res. Part B: Methodol.*, vol. 40, no. 10, pp. 872–884, Dec. 2006.

[16] P. Galinier and J.-K. Hao, "Hybrid evolutionary algorithms for graph coloring," *J. Combinatorial Optim.*, vol. 4, no. 5, pp. 379–397, 1999.

[17] M. Gaudioso and G. Paletta, "A heuristic for the periodic vehicle routing problem," *Transp. Sci.*, vol. 26, no. 2, pp. 86–92, May 1992.

[18] B. Golden, J. DeArmon, and E. Baker, "Computational experiments with algorithms for a class of routing problems," *Comput. Oper. Res.*, vol. 10, no. 1, pp. 47–59, 1983.

[19] B. Golden and R. Wong, "Capacitated arc routing problems," *Networks*, vol. 11, no. 3, pp. 305–315, 1981.

[20] P. Greistorfer, "A tabu scatter search metaheuristic for the arc routing problem," *Comput. Ind. Eng.*, vol. 44, no. 2, pp. 249–266, Feb. 2003.

[21] H. Handa, D. Lin, L. Chapman, and X. Yao, "Robust solution of salting route optimisation using evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 3098–3105.

[22] A. Hertz, G. Laporte, and M. Mittaz, "A tabu search heuristic for the capacitated arc routing problem," *Oper. Res.*, vol. 48, no. 1, pp. 129–135, Jan./Feb. 2000.

[23] A. Hertz and M. Mittaz, "A variable neighborhood descent algorithm for the undirected capacitated arc routing problem," *Transp. Sci.*, vol. 35, no. 4, pp. 425–434, Nov. 2001.

[24] A. Kansou and A. Yassine, "Ant colony system for the periodic capacitated arc routing problem," in *Proc. 2009 International Network Optimization Conference*, 2009, pp. 1–7.

[25] P. Lacomme, C. Prins, and W. Ramdane-Chérif, "Evolutionary algorithms for multiperiod arc routing problems," in *Proc. 9th Int. Conf. Inf. Process. Manage. Uncertainty Knowl.-Based Syst.*, 2002, pp. 845–852.

[26] P. Lacomme, C. Prins, and W. Ramdane-Chérif, "Competitive memetic algorithms for arc routing problem," *Ann. Oper. Res.*, vol. 131, no. 1–4, pp. 159–185, Oct. 2004.

[27] P. Lacomme, C. Prins, and W. Ramdane-Chérif, "Evolutionary algorithms for periodic arc routing problems," *Eur. J. Oper. Res.*, vol. 165, no. 2, pp. 535–553, Sep. 2005.

[28] B. Liu, L. Wang, and Y. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 18–27, Feb. 2007.

[29] H. Longo, M. P. de Aragão, and E. Uchoa, "Solving capacitated arc routing problems using a transformation to the CVRP," *Comput. Oper. Res.*, vol. 33, no. 6, pp. 1823–1837, Jun. 2006.

[30] Y. Mei, K. Tang, and X. Yao, "A global repair operator for capacitated arc routing problem," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 3, pp. 723–734, Jun. 2009.

[31] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," CalTech, Pasadena, CA, CalTech Concurrent Computation Program Rep. 826, 1989.

[32] W. Pearn, "Approximate solutions for the capacitated arc routing problem," *Comput. Oper. Res.*, vol. 16, no. 6, pp. 589–600, 1989.

[33] W. Pearn, "Augment-insert algorithms for the capacitated arc routing problem," *Comput. Oper. Res.*, vol. 18, no. 2, pp. 189–198, 1991.

[34] M. Polacek, K. F. Doerner, R. F. Hartl, and V. Maniezzo, "A variable neighborhood search for the capacitated arc routing problem with intermediate facilities," *J. Heuristics*, vol. 14, no. 5, pp. 405–423, Oct. 2008.

[35] J. Y. Potvin and S. Bengio, "The vehicle routing problem with time windows part II: Genetic search," *INFORMS J. Comput.*, vol. 8, no. 2, pp. 165–172, 1996.

[36] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 284–294, Sep. 2000.

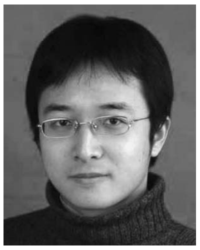
[37] K. Tang, Y. Mei, and X. Yao, "Memetic algorithm with extended neighborhood search for capacitated arc routing problems," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1151–1166, Oct. 2009.

[38] M. Tang and X. Yao, "A memetic algorithm for VLSI floorplanning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 62–69, Feb. 2007.

[39] G. Ulusoy, "The fleet size and mix problem for capacitated arc routing," *Eur. J. Oper. Res.*, vol. 22, no. 3, pp. 329–337, Dec. 1985.

Yi Mei (S'09) received the Bachelor's degree in mathematics from the University of Science and Technology of China (USTC), Hefei, China, in 2005, and the Ph.D. degree in computer science from the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, USTC, in 2010.

He is currently working as a Research Associate at the Chinese University of Hong Kong, Shatin, Hong Kong. His current research interests include evolutionary computation, memetic algorithms, meta-heuristics, multiobjective optimization, and robust design optimization for arc routing problems, financial optimization and other combinatorial optimization problems.



Ke Tang (S'05–M'07) received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2002 and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in 2007.

In 2007, he joined the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, University of Science and Technology of China, Hefei, China, where he was promoted to Professor in 2011. He is the author/coauthor of more than 50 refereed publications. His major research interests include machine learning, evolutionary computation, data mining, meta-heuristic algorithms, and real-world applications.

Dr. Tang is an Associate Editor of *IEEE Computational Intelligence Magazine* and the Chair of the IEEE Task Force on Large Scale Global Optimization. He served as a program co-chair of CEC2010 held in Barcelona.

Xin Yao (M'91–SM'96–F'03) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1982, the M.Sc. degree from the North China Institute of Computing Technology, Beijing, China, in 1985, and the Ph.D. degree from USTC in 1990.

From 1985 to 1990, he was an Associate Lecturer and Lecturer with USTC, while working toward the Ph.D. degree in simulated annealing and evolutionary algorithms. In 1990, he was a Postdoctoral Fellow with the Computer Sciences Laboratory, Australian National University, Canberra, Australia, where he continued his work on simulated annealing and evolutionary algorithms. In 1991, he was with the Knowledge-Based Systems Group, Commonwealth Scientific and Industrial Research Organization, Division of Building, Construction and Engineering, Melbourne, Australia, where he worked primarily on an industrial project on automatic inspection of sewage pipes. In 1992, he returned to Canberra to take up a lectureship in the School of Computer Science, University College, University of New South Wales, Australian Defense Force Academy, Sydney, Australia, where he was later promoted to a Senior Lecturer and Associate Professor. Since April 1999, he has been a Professor (Chair) of computer science in the University of Birmingham, Birmingham, U.K. He is currently the Director of the Center of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Birmingham, U.K. and also a Changjiang (Visiting) Chair Professor (Cheung Kong Scholar) with the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, USTC. He has given more than 50 invited keynote and plenary speeches at conferences and workshops worldwide. He has more than 300 referenced publications. His major research interests include evolutionary artificial neural networks, automatic modularization of machine learning systems, evolutionary optimization, constraint-handling techniques, computational time complexity of evolutionary algorithms, coevolution, iterated prisoner's dilemma, data mining, and real-world applications.

Dr. Yao was the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008, an Associate Editor or editorial board member of 12 other journals, and the Editor of the *World Scientific Book Series on Advances in Natural Computation*. He was the recipient of the President's Award for Outstanding Thesis by the Chinese Academy of Sciences for his Ph.D. work on simulated annealing and evolutionary algorithms in 1989. He was the recipient of the 2001 IEEE Donald G. Fink Prize Paper Award for his work on evolutionary artificial neural networks.