Decomposition-Based Memetic Algorithm for Multi-Objective Capacitated Arc Routing Problem

Yi Mei, Ke Tang and Xin Yao

Abstract— **The Capacitated Arc Routing Problem (CARP) is a challenging combinatorial optimization problem with many real world applications, e.g., salting route optimization and fleet management. Despite the fact that most of previous research formulate CARP as a single-objective problem, one often has to consider more than one objective simultaneously in practice. In this paper, we investigate the more realistic multi-objective CARP (MO-CARP). By combining the advanced features from both the MAENS approach for single-objective CARP and the strategies in evolutionary multi-objective optimization, a new memetic algorithm called Decomposition-based Memetic Algorithm with Extended Neighborhood Search (D-MAENS) is proposed. The experimental studies have shown that such combination outperforms significantly a well known existing multi-objective evolutionary algorithm (NSGA-II) and the state-of-the-art approach for MO-CARP (LMOGA), in terms of both the convergence to the Pareto front and the distribution of the nondominated solutions. In addition, the efficacy of employing local search in solving MO-CARP has been demonstrated.**

I. Introduction

The Capacitated Arc Routing Problem (CARP) is a well known combinatorial optimization problem which has wide applications in the real world, including winter gritting [1] [2], urban waste collection, post delivery, etc [3]. For this reason, it has attracted much interest in the last few decades. The problem can be briefly described as follows: given a graph and some of its edges and arcs (directed edges) that are required to be served (we call each of them a task), a number of vehicles with limited capacity are located in a special vertex called the depot to serve the tasks. The problem is to seek an optimal routing plan for the vehicles so that the following constraints are satisfied:

- Each vehicle starts and ends at the depot;
- Each task is served by exactly one vehicle;
- The total demand of the tasks served by each vehicle does not exceed its capacity.

For a better understanding, we take the salting route optimization problem (SRO) as an example. SRO is a routing problem that seeks an optimal plan for spreading salt on the streets in winter nights to prevent them from freezing, and further avoid the possibly caused accidents. In the case of SRO, the vertices correspond to the crossroads while the edges and arcs correspond to the two-way and one-way streets, respectively. For each street required to be served, the demand is the amount of salt needed to spread on it. The depot indicates the location of the salt warehouse, where the vehicles load the salt to spread.

CARP is NP-hard [4], and exact methods are only applicable to the small and medium-size instances. However, most of real world applications have large problem sizes. Besides, the plans often have to be made within a restricted time budget. Under such a situation, heuristics and meta-heuristics are promising approaches in order to provide acceptable solutions in time. For example, the Augment-Merge heuristic proposed by Golden and Wong [4], the path scanning heuristic proposed by Golden et al. [5] and the Ulusoy's splitting heuristic [6] are three typical heuristics for CARP. The first meta-heuristic approach is the tabu search proposed by Hertz et al. [7], and the two state-of-the-art meta-heuristics at present are the memetic algorithm (MA) proposed by Lacomme et al. [8] and the tabu search proposed by Brandao et al. [9]. We have also conducted intensive investigations on CARP in our previous work. According to the observation of the characteristic of CARP, we proposed a Global Repair Operator (GRO) that can be embedded in any search-based approach for CARP [10]. More importantly, we proposed a Memetic Algorithm with Extended Neighborhood Search (MAENS) for CARP [12]. MAENS was demonstrated to be better than other existing approaches in the qualities of final solutions, though more computational time was required.

In most of previous research, CARP was formulated as a single-objective problem. However, in many real-world applications, more than one objective should be taken into account. For example, in SRO, the financial cost of the plan depends on both the total cost and the cost of the longest route (which is also called the makespan). The former determines the expense of fuel, while the latter determines the payment of the drivers that is related to their working time. Therefore, the multi-objective CARP (MO-CARP) is closer to reality than traditional CARP and deserves much more research interest. In our study, we investigate MO-CARP and aim at minimizing the total cost and makespan simultaneously. To distinguish from MO-CARP, we refer to the traditional CARP only considering the total cost as the single-objective CARP (SO-CARP) hereafter. In a multi-objective optimization problem (MOP), instead of finding a single global optimum, the goal is often maintaining a set of solutions that are good "trade-offs" or good compromises among the objectives [13]. Evolutionary Algorithms (EAs) are commonly used in solving MOPs, since they can deal

The authors are with the Nature Inspired Computation and Applications Laboratory, the Department of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China. Xin Yao is also with CERCIA, the School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, U.K. (emails: meiyi@mail.ustc.edu.cn, ketang@ustc.edu.cn, x.yao@cs.bham.ac.uk).

Corresponding author: Ke Tang (Phone: +86–551–3600754).

with a set of solutions (also called population) simultaneously and thus can find multiple such good trade-off solutions in a single run. Therefore, in our study, we consider solving MO-CARP with EAs.

MO-CARP is a challenging problem because of the difficulties induced by the characteristics of both CARP and MOP. In contrast to SO-CARP, investigation of MO-CARP is still in its infancy, and there has been only one published approach proposed for it [14]. In the paper, the authors proposed a hybrid algorithm by combining the SO-CARP approach proposed by them earlier [8] and the nondominated sorting genetic algorithm II (NSGA-II) [15], which is one of the most commonly used EA framework for MOP.

In this paper, by investigating MO-CARP from the viewpoint of both CARP and MOP, we propose an algorithm named the Decomposition-based Memetic Algorithm with Extended Neighborhood Search (D-MAENS). It is the hybrid algorithm that combines the MAENS approach [12] with a decomposition-based framework. Besides, proper strategies are employed to address the EMO issues in the context of MO-CARP, considering both the performance of the algorithm and the ease of implementation. D-MAENS was compared with the genetic algorithm proposed by Lacomme et al. (LMOGA) [14] and NSGA-II [15] on the gdb test set in terms of several performance metrics. The empirical results show that D-MAENS performed better than the other compared algorithms with respect to both convergence to the Pareto front and the distribution of the nondominated solutions. Furthermore, observing that D-MAENS and LMOGA, both of which employ local search, performed much better than NSGA-II that does not employ local search, the efficiency of employing local search in solving MO-CARP is verified.

The rest of the paper is organized as follows: Section II gives the background of our work, including the detailed introduction of MO-CARP and related work on EMO. Section III discusses the important issues in solving MO-CARP with EAs and evaluate the existing strategies for addressing them. Section IV describes the proposed D-MAENS. Afterwards, experimental studies are carried out in Section V. Finally, the paper is concluded in Section VI.

II. Background

A. Multi-Objective Capacitated Arc Routing Problem

CARP is defined on a graph $G(V, E, A)$, where V, E and A stand for the set of vertices, edges and arcs (directed edges), respectively. Each edge $(v_i, v_j) \in E$ and arc $\langle v_i, v_j \rangle \in A$ is associated with three nonnegative features : the traversal cost $c^{trav}(v_i, v_j)$, the serving cost $c^{serv}(v_i, v_j)$ and the demand $d(v_i, v_j)$. There are some certain edges and arcs that are required to be served, and their demands are greater than zero. Such edges and arcs are called tasks. The edge task set is denoted as $E_R = \{(v_i, v_j) \in E | d(v_i, v_j) > 0\}$ and the arc task set is denoted as $A_R = \{(v_i, v_j) \in A | d(v_i, v_j) > 0\}$. Finally, the task set is denoted as $T = E_R \cup A_R$. The tasks are to be served by a set of m vehicles with an identical capacity Q that are based at the depot $v_s \in V$. An edge task (v_i, v_j) can be serve from both the positive direction (from v_i to v_j) and the negative direction (from v_j to v_i), while an arc task $\langle v_i, v_j \rangle$ can only be served from v_i to v_j. In order to facilitate the problem definition, each edge task is assigned two identities, one for each direction, and each arc task is assigned one identity. All the assigned identities are unique positive integers. An identity $t \in \mathbb{N}^+$ is associated with the following six features: the tail vertex $tv(t)$, the head vertex $hv(t)$, the traversal cost $c^{trav}(t)$, the serving cost $c^{serv}(t)$, the demand $d(t)$, and the inverse identity $inv(t)$. For an edge task (v_i, v_j) and the two identities t_1 assigned to its positive direction $\langle v_i, v_j \rangle$ and t_2 assigned to its negative position $\langle v_j, v_i \rangle$, the features are defined as follows:

- $hv(t_1) = tv(t_2) = v_i$;
- $tv(t_1) = hv(t_2) = v_j$;
- $c^{trav}(t_1) = c^{trav}(t_2) = c^{trav}(v_i, v_j)$;
- $c^{serv}(t_1) = c^{serv}(t_2) = c^{serv}(v_i, v_j)$;
- $d(t_1) = d(t_2) = d(v_i, v_j)$;
- $inv(t_1) = t_2$, $inv(t_2) = t_1$.

For an arc task $\langle v_i, v_j \rangle$ and its corresponding identity t, the features are defined as:

- $hv(t) = v_i$, $tv(t) = v_j$;
- $c^{trav}(t) = c^{trav}(v_i, v_j)$;
- $c^{serv}(t) = c^{serv}(v_i, v_j)$;
- $d(t) = d(v_i, v_j)$;
- $inv(t) = -1$.

Since all of the identities are positive, $inv(t) = -1$ indicates the inverse identity of t does not exist. In addition, a depot loop is assigned the identity 0 and its features are defined as follows:

- $tv(0) = hv(0) = v_s$;
- $c^{trav}(0) = c^{serv}(0) = d(0) = 0$;
- $inv(0) = 0$.

Using the above notations, a CARP solution can be represented as a set of routes $S = (R_1, R_2, ..., R_m)$. Each route R_k is a sequence of the identities $R_k = (t_1^k, t_2^k, ..., t_{l_k}^k)$. In order to ensure that each route starts and ends at the depot, R_k starts and ends at the depot loop 0, i.e., $t_1^k = t_{l_k}^k = 0$. An example is illustrated in Fig. 1. In the graph, $E_R = \{(v_1, v_5), (v_2, v_6), (v_3, v_7), (v_4, v_8)\}$, $A_R = \emptyset$ and the depot is v_0. The numbers presented close to each edges are the identities assigned to them. The number out of the parenthesis is the one assigned to the current direction while the number in the parenthesis indicates the one assigned to the inverse direction. The dashed arrows between adjacent task identities (e.g., $\langle v_0, v_1 \rangle$ and $\langle v_7, v_6 \rangle$ in Fig. 1) stand for the shortest path from the former vertex to the latter vertex, which can be obtained by Dijkstra's algorithm [16].

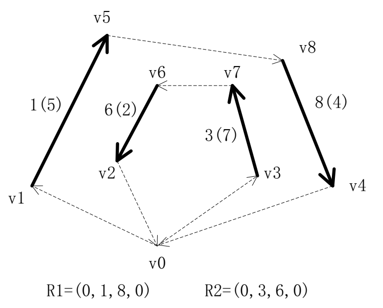

Fig. 1. An example of a CARP solution

Under such solution representation scheme, the problem can be represented as follows:

$$\min\ c^{tot}(S) = \sum_{k=1}^{m} c^{tot}(R_k) \tag{1}$$

$$\min\ c^{max}(S) = \max_{k} c^{tot}(R_k) \tag{2}$$

$$s.t.:\ \sum_{k=1}^{m}(l_k - 2) = |T| \tag{3}$$

$$t_{p_1}^{k_1} \neq t_{p_2}^{k_2}, \forall (k_1, p_1) \neq (k_2, p_2) \tag{4}$$

$$t_{p_1}^{k_1} \neq inv(t_{p_2}^{k_2}), \forall (k_1, p_1) \neq (k_2, p_2) \tag{5}$$

$$d(R_k) \leqslant Q, \forall 1 \leqslant k \leqslant m \tag{6}$$

The inequality $(k_1, p_1) \neq (k_2, p_2)$ between the two pairs (k_1, p_1) and (k_2, p_2) indicates that at least one of the two inequations $k_1 \neq k_2$ and $p_1 \neq p_2$ is satisfied. $c^{tot}(R_k)$ and $d(R_k)$ stand for the total cost induced and the total demand served in route $R_k = (t_1^k, t_2^k, ..., t_{l_k}^k)$. They can be calculated as follows:

$$c^{tot}(R_k) = \sum_{p=1}^{l_k - 1}[c^{serv}(t_p^k) + dist(tv(t_p^k), hv(t_{p+1}^k))]$$

$$d(R_k) = \sum_{p=1}^{l_k} d(t_p^k)$$

where $dist(v_1, v_2)$ indicates the distance from vertex v_1 to vertex v_2, which is defined as the length of the shortest path from v_1 to v_2.

Eq. (1) is the total cost while Eq. (2) is the makespan. Constraints (3)-(5) guarantee that each task is served exactly once by one vehicle. Constraint (6), which is also called the capacity constraint, indicates that the total demand served by each vehicle does not exceed its capacity. Note that each route naturally starts and ends at the depot since the corresponding task identity sequence starts and ends at the depot loop.

B. Related Work on Evolutionary Multi-objective Optimization

A MOP can be briefly described as follows:

$$\max\ F(x) = (f_1(x), ..., f_n(x))$$
$$s.t.:\ x \in \Omega$$

where Ω is the decision variable space, and $F : \Omega \to R^n$ is an objective vector that consists of n contradicted objectives. Given two objective vectors u and v, u is said to dominate v if an only if all the objective values of u are no smaller than that of v and at least one objective value of u is greater than v. u and v are said to be nondominated when neither u nor v dominate the other. Then, a solution x^* is said to be Pareto optimal if there is no other $x \in \Omega$ so that $F(x)$ dominates $F(x^*)$. When solving a MOP, the goal is to find or approximate the set of Pareto optimal solutions (called Pareto optimal set or Pareto set for short) and their corresponding objective vectors (called Pareto front).

When solving MOPs with EAs, there are several important issues that have not been encountered when solving single-objective problems. First, unlike in single-objective optimization problems, the fitness of a solution should be assigned according to multiple criteria (the multiple objective values) in MOP. Second, when solving single-objective problems, the goal is to obtain a single global optimum. However, in MOP, a set of nondominated solutions should be maintained. Therefore, the distribution of the nondominated solutions has to be considered. In general, the nondominated solutions should be distributed as uniformly as possible. Besides, they should be able to cover the whole Pareto front completely. Third, like in single-objective cases, elitism should be implemented to keep the nondominated solutions in the population during the search. There have been various MOEAs proposed to address the above three issues in different ways (e.g., [17] [18] [19] [15] [20] [21]).

As a more general research field, the evolutionary multi-objective combinatorial optimization has attracted much more interest than MO-CARP. A survey of multi-objective combinatorial optimization problem was given in [22]. Examples of the search-based methods for solving multi-objective combinatorial optimization problems were provided in [23], including traditional MOEAs, simulated annealing, tabu search, etc. In addition to directly using the traditional MOEAs, some researchers considered combining the MOEA framework with local search to enhance the performance of the algorithm (e.g., [24] [25] [26]). However, it is difficult to apply the same algorithm to different combinatorial problems because of their different problem structures. For example, when applied to the Traveling Salesman Problem [25] and 0/1 knapsack problem [26], although the same framework was adopted, the representations and operators were different. Similarly, the traditional MOEAs cannot be expected to show the same performance on MO-CARP as they do on the test functions like the ZDT [27] and DTLZ [28] test sets. One has to design a specific algorithm according to problem characteristics of MO-CARP.

When solving MO-CARP, one can either extend an approach of its single-objective counterpart SO-CARP or adapt an existing MOEA to the case of MO-CARP by employing the problem specific solution representation and operators. The only published approach for MO-CARP was extended from an SO-CARP approach [14]. The above two strategies have complementary advantages and disadvantages. When extending an SO-CARP approach, the algorithm can search effectively in the complicated solution space, but the EMO issues are not appropriately addressed. On the other hand, when using an existing MOEA framework, the EMO issues are considered carefully while the search may be ineffective in the solution space. Therefore, it is reasonable to combine the two strategies together to overcome the weaknesses of both of them. Concretely, we consider combining an existing SO-CARP approach and proper strategies to address the EMO issues. To this end, the existing strategies have to be evaluated in the context of MO-CARP. This will be done in the next section.

III. EMO Issues in MO-CARP

It has been demonstrated that hybridizing EAs with local search shows good performance in solving SO-CARP (e.g., [8] and [12]). When combining an MOEA with local search, a new important issue arises. That is, how to identify a neighboring solution to replace the current solution. In general, the best solution in the neighborhood is selected to replace the current solution, and thus the issue can be transformed to identifying the best solution in the neighborhood. Together with the aforementioned three issues, there are four important issues in solving MO-CARP with EAs: (1) fitness assignment, (2) diversity preservation, (3) elitism and (4) identifying the best neighboring solution during local search.

Fitness Assignment: The existing strategies for fitness assignment in EMO can be divided into three types: the criterion-based (e.g., [17]), domination-based (e.g., [15]) and decomposition-based (e.g., [21]) methods. The first two strategies are solely based on the objective values, and thus is expected to perform similarly when employed in various problems. In other words, when using the criterion-based methods, the intermediate regions of the Pareto front will be overlooked, while the domination-based methods can effectively force the population towards the Pareto front. When using the decomposition-based methods, it is assumed that each Pareto optimal solution can be seen as the optimal solution of a corresponding scalar optimization sub-problem. However, there usually exist solutions which are not optimal for any weighted sum of the objectives in combinatorial MOPs such as MO-CARP [29]. Therefore, the decomposition-based method is expected not to perform well for fitness assignment in MO-CARP.

Diversity Preservation: The niching technique, cell-based methods and crowding distance method are the three typical existing strategies for diversity preservation. The basic idea behind them are preventing the occurrence of multiple solutions that are close to each other in the population. Therefore, they are expected to be able to maintain diversity in MO-CARP as well. Among them, the performances of niching technique and cell-based methods are parameter dependent, i.e., their performance depend largely on the parameters such as the sharing parameter δ_{share} in the niching technique and the cell size in the cell-based methods. On the other hand, the performance of the crowding distance method is expected to perform consistently since it has no user-defined parameter. There is an additional implicit strategy to maintain diversity associated with the decomposition-based method. That is, the diversity can be naturally preserved by the "diversity" among sub-problems [21]. However, this is based on the assumption that different sub-problems can reach different optimal solutions, which is not true in MO-CARP.

Elitism: The elitism mechanism can be implemented by either storing the nondominated solutions in an archive or combining the parents and offsprings for selection. Both of them perform independently of the problem structure.

Evaluating Solutions During Local Search: The most commonly used strategy is to aggregate the objectives into a single objective during the local search (e.g., [25] [30]) by the weighted sum approach. Indeed, their performances depend largely on the selected weight vector. Besides, one can also employ a decomposition-base framework so that the local search is only applied in each single-objective sub-problem. In this way, there is no user-defined parameter.

According to the above discussions, it can be seen that the existing MOEAs employing criterion-based and decomposition-based methods for fitness assignment may perform unsatisfactorily in the case of MO-CARP. Besides, when using the niching technique and cell-based method for diversity preservation, and using aggregation method for evaluating solution during local search, the corresponding parameters are often difficult to tune to guarantee a good performance. Therefore, we propose the algorithm D-MAENS by employing the following EMO strategies: (1) the fast nondominated sorting procedure of NSGA-II for fitness assignment; (2) the crowding distance approach for diversity preservation; (3) both of the two existing elitism strategies for elitism and (4) the decomposition-based strategy of MOEA/D for identifying best solutions during local search.

IV. Decomposition-based Memetic Algorithm with Extended Neighborhood Search

D-MAENS employs a decomposition-based framework and the evolutionary operators of MAENS. The EMO strategies mentioned above are also included to address the EMO issues. The framework can be described as follows:

Input:

- A MO-CARP instance P;
- A stopping criterion;
- The number of decomposed sub-problems: N;
- N uniformly distributed weight vectors: $\vec{\lambda}^1, ..., \vec{\lambda}^N$;
- Neighborhood size of each sub-problem: T.

Output: A set of nondominated solutions X^*.

Step 1: Initialization:

a) Set $X^* = \emptyset$;

b) Generate N single-objective sub-problems $\{P_1, P_2, ..., P_N\}$, where the objective of P_i is $g^{ws}(x|\vec{\lambda}^i) = \vec{\lambda}^i \cdot \vec{f}(x)$;

c) For each $\vec{\lambda}^i$, obtain the T closest weight vectors $\vec{\lambda}^{i_1}, ..., \vec{\lambda}^{i_T}$ according to the Euclidean distance.

Then, set the neighborhood $B(i) = \{i_1, ..., i_T\}$.

d) Randomly initialize a population $X = \{x_1, ..., x_N\}$;

Step 2: Update:

For $i = 1$ to N, do

Step 2.1: Assign each sub-problem P_i a unique representative $x_i^r \in X$;

Step 2.2: Construct a sub-population $X_i = \{x_k^r | k \in B(i)\}$;

Step 2.3: Randomly select two solutions x_k^r and x_l^r from X_i;

Step 2.4: Apply the crossover and local search of MAENS to x_k^r and x_l^r and the objective function $g^{ws}(x|\vec{\lambda}^i)$ to generate an offspring y_i;

Step 2.5: Update X^* by y_i.

Combine the parent and offspring populations together and sort them by the fast nondominated sorting procedure and crowding distance approach of NSGA-II [15]. Then, pick the first N solutions into the next generation.

Step 3: **Stopping Criteria**: If stopping criteria is satisfied, stop the algorithm and return X^*. Otherwise, go to **Step 2**.

In MO-CARP which has two objectives, the weight vectors are uniformly set as follows:

$$\vec{\lambda}^i = (\frac{i-1}{N-1}, \frac{N-i}{N-1})$$

Accordingly, at Step 2.1, the representatives of the decomposed sub-problems are assigned in Algorithm 1. In this way, it is expected that each sub-problem can be assigned a relatively good solutions according to its own aggregated objective.

```
Input: A population $X = \{x_1, ..., x_N\}$;
Output: A representative set $\{x_1^r, ..., x_N^r\}$;
1:  for $i$ = 1 to $N-1$ do
2:    for $j = i+1$ to $N$ do
3:      if $f_2(x_j) < f_2(x_i)$ or
        $(f_2(x_j) = f_2(x_i)$ and $f_1(x_j) > f_1(x_i))$ then
4:        swap $x_i$ and $x_j$;
5:      end if
6:    end for
7:  end for
8:  for $i$ = 1 to $N$ do
9:    set $x_i^r = x_i$;
10: end for
```

Algorithm 1: The assignment of representatives

In D-MAENS, the solution representation scheme, crossover operator and local search process of MAENS [12] are directly employed, while the solution evaluation is adapted to the multi-objective case. For this reason, we only describe the solution evaluation of D-MAENS here. For full details of MAENS, readers can refer to [12].

As mentioned in Sub-Section II-A, in MO-CARP, the total cost $c^{tot}(S)$ and the makespan $c^{max}(S)$ are to be minimized. Besides, to compare between feasible and infeasible solutions, the total violated load $tvl(S)$ is also taken into account. Since $c^{tot}(S)$ and $c^{max}(S)$ are not of the same scale, normalization should be carried out between them. To this end, the two fitness functions are defined as follows:

$$f_1(S) = \hat{c}^{tot}(S) = (c^{tot}(S) - c_*^{tot})/(c_{**}^{tot} - c_*^{tot})$$

$$f_2(S) = \hat{c}^{max}(S) = (c^{max}(S) - c_*^{max})/(c_{**}^{max} - c_*^{max})$$

where c_*^{tot} and c_{**}^{tot} indicate the minimal and maximal values among the total cost of all possible solutions, while c_*^{max} and c_{**}^{max} stand for the minimal and maximal possible makespan values. However, in practice, we cannot obtain the exact values of them due to the large problem size. Therefore, we replace them with the approximated values. Concretely, they are set to the minimal (maximal) value among the total cost (makespan) of the feasible solutions found during the search process. Finally, the objective function of each sub-problem can be stated as follow:

$$g^{ws}(S|\vec{\lambda}) = \lambda_1 f_1(S) + \lambda_2 f_2(S)$$

During the local search, the feasible and infeasible solutions are compared according to the following simple but effective rule: A solution S_1 is said to be better than another solution S_2 if $tvl(S_1) < tvl(S_2)$ or $tvl(S_1) = tvl(S_2)$ and $g^{ws}(S_1|\vec{\lambda}) < g^{ws}(S_2|\vec{\lambda})$.

V. Experimental Studies

In order to evalute D-MAENS, we compared it with LMOGA and NSGA-II on the gdb test set [31]. Here, LMOGA is selected to be compared since it is the only published algorithm proposed for MO-CARP and represents the way of extending an approach of SO-CARP to solve MO-CARP. NSGA-II stands for the traditional MOEAs without local search and represents the way of directly using an existing MOEA for MO-CARP.

A. Experimental Setup

The experiments were carried out on the gdb test set [31], which was generated by DeArmon in [31] and consists of 23 instances. In our experiments, the MO9 version of LMOGA proposed in the original paper [14] was selected, for it showed the best performance over the test instances among all the 9 LMOGA versions. For the sake of convenience, the MO9 version of LMOGA is called LMOGA hereafter without loss of clarity. Note that NSGA-II was proposed for numerical optimization problems. In order to directly use it to solve MO-CARP, problem specific solution representation and operators should be employed. In our experiments, the solution representation and crossover operator of D-MAENS were directly employed in NSGA-II, while the mutation operator was a random implementation of the single insertion operator. That is, a task is randomly selected and moved to another randomly selected position. The parameter settings of the compared algorithms are listed in Table I. In this way,

TABLE I
PARAMETER SETTINGS OF THE COMPARED ALGORITHMS

Parameter	D-MAENS	LMOGA	NSGA-II
Population size	60	60	60
Crossover rate	1.0	1.0	1.0
Mutation/LS rate	0.1	Every 10 generations	0.1
Max. generations	200	200	$200n/\log n$
Neighborhood size	9	-	-

the compared algorithms have their parameters as consistent with each other as possible. Note that NSGA-II spends much less time at each generation since it does not employ the local search process, which is much more time consuming than the traditional mutation operation. Hence, we set the maximal number of generations of NSGA-II to $200n/\log n$, where n is the number of tasks. In this way, the overall computational time of NSGA-II will be comparable with that of D-MAENS. All the algorithms were independently run 30 times on a computer with Intel(R) Xeon(R) E5335 2.00GHz CPU.

B. Performance Measures

The performance of an MOEA includes two aspects. First, the obtained nondominated set should be as close to the true Pareto front as possible. Second, the solutions in the obtained nondominated set should be distributed as diversely and uniformly as possible. It is clear that the two aspects cannot be fully reflected by a single metric, and a number of metrics have been suggested in the previous research works [32]. In this study, the following three metrics are used.

Distance from reference set (I_D): This metric was suggested by Czyzzak et al. [33]. It is defined as follow:

$$I_D(A) = \frac{\sum_{y\in R}\{\min_{x\in A}\{d(x,y)\}\}}{|R|}$$

Given a set A, $I_D(A)$ provides information about the average distance from a solution in the reference set R to the closest solution in A. A smaller value of $I_D(A)$ indicates that A is closer to R, and $I_D(A) = 0$ means that A covers R. In general, the reference set R is defined as a set of Pareto optimal solutions so that their objective vectors are uniformly distributed in the Pareto front. In this way, I_D indicates the closeness of the nondominated set to the Pareto front and the uniformity of the distribution of the nondominated solutions. Concretely, a smaller I_D indicates that the nondominated set is closer to the Pareto front and the nondominated solutions are distributed more uniformly. However, in practice, the Pareto optimal solutions are difficult to be obtained in a MO-CARP instance. Furthermore, the Pareto optimal solutions themselves may even be distributed non-uniformly. In our experiments, as an approximation of the Pareto set, R is defined as the set of solutions that remain nondominated when the nondominated solutions obtained by all the 30 runs of the compared algorithms are combined together. Note that I_D can no longer reflect the uniformity that the nondominated solutions are distributed due to the non-uniformity of the distribution of solution in R.

Spread (Δ): This metric was suggested by Deb et al. [15]. It can be stated as follow:

$$\Delta(A) = \frac{d_f + d_l + \sum_{i=1}^{n-1}|d_i - \bar{d}|}{d_f + d_l + (n-1)\times \bar{d}}$$

where d_f and d_l are the Euclidean distances between the leftmost and rightmost solutions of the Pareto front and the extreme solutions in A. n is the number of solutions in A and d_i is the Euclidean distance between the i^{th} left and the $(i+1)^{th}$ left solutions in A. $\bar{d}$ stands for the mean value of all d_i's. Δ implies the uniformity of distribution of the nondominated solutions and the extent of the nondominated set. A smaller Δ indicates that the nondominated solutions are distributed more uniformly and the nondominated set has a better extent. In practice, the leftmost and rightmost solutions of the Pareto front are not available. Therefore, in our experiments, they are defined as the leftmost and rightmost solutions among the nondominated solutions obtained by all the 30 runs of the compared algorithms.

Hypervolume (I_H): This metric was suggested by Zitzler et al. [20] to indicate the volume covered by the nondominated solutions. Given a nondominated set A and a reference point in the objective space f^*, $I_H(A)$ is defined as the area of the union of all the hypercubes constructed by the objective vectors of the solutions in A and f^*. I_H reflects the closeness of the nondominated set to the Pareto front. In addition, it implies the extent that the nondominated solutions cover the Pareto front. A greater I_H indicates that the nondominated set is closer to the Pareto front and covers the Pareto front more completely.

In our study, all of the metrics are computed based on the normalized objective vectors of the nondominated solutions, which are obtained as follows:

$$\hat{f}_i = (f_i - f_i^{\min})/(f_i^{\max} - f_i^{\min}), i = 1, 2$$

where f_1 and f_2 stand for the total cost and makespan, respectively. $f_i^{\max}$ and $f_i^{\min}$ are the maximal and minimal value of f_i among all the results obtained over the 30 runs of the three compared algorithms.

C. Experimental Results

Table II presents the average value of I_D, Δ and I_H over the 30 independent runs of the compared algorithms. In addition, the characteristics of the instances such as the number of vertices and edges are provided. In the tables, the columns headed "$|V|$" and "$|E|$" stand for the number of vertices and edges in the graph of the instance. Since in the gdb instances, all of the edges are required to be served, i.e., the number of tasks is equal to that of the edges. Thus, the column "$|E|$" also presents the number of tasks. The column headed "τ" indicates the minimal number of vehicles required subject to the capacity constraint, which can be obtained by dividing the total demand of the tasks by the capacity of vehicles. In general, given the same number of tasks, a greater value of τ indicates a higher complexity of the instance. For each instance and each performance metric, the Wilcoxon rank sum test was further carried out

TABLE II

THE AVERAGE VALUE OF I_D, Δ AND I_H OVER THE 30 INDEPENDENT RUNS OF THE COMPARED ALGORITHMS ON THE GDB SET. THE SIGNIFICANTLY BEST RESULTS ARE IN BOLD (WITH SMALLEST I_D AND Δ, WHILE WITH GREATEST I_H).

Name	$\|V\|$	$\|E\|$	τ	I_D			Δ			I_H		
				D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II
gdb1	12	22	5	**0.000000**	**0.000000**	0.141144	0.737152	0.737152	0.738243	**0.933333**	**0.933333**	0.801793
gdb2	12	26	6	0.047577	0.055494	0.201959	0.744072	0.775904	0.846040	0.968374	0.968089	0.733586
gdb3	12	22	5	**0.008124**	0.049914	0.190699	0.835997	0.879512	0.820943	**0.960873**	0.950715	0.821352
gdb4	11	19	4	**0.001766**	0.019672	0.149119	0.780746	0.737634	0.809841	**0.928217**	0.923841	0.782717
gdb5	13	26	6	**0.013742**	0.081535	0.191624	0.798773	0.869112	0.824886	**0.925012**	0.881022	0.665339
gdb6	12	22	5	0.022758	**0.010959**	0.166298	0.820776	0.844730	**0.777610**	0.927929	**0.930271**	0.744261
gdb7	12	22	5	**0.001905**	0.206082	0.259892	**0.727180**	0.787064	0.779443	**0.779109**	0.632002	0.532821
gdb8	27	46	10	**0.065808**	0.097087	0.214706	0.812013	0.859538	0.823990	**0.924160**	0.872941	0.724383
gdb9	27	51	10	**0.038974**	0.051164	0.274357	0.874343	0.857652	0.881908	**0.930479**	0.917804	0.633869
gdb10	12	25	4	**0.017936**	0.258262	0.351409	**0.709984**	0.879912	0.853631	**0.823408**	0.630947	0.465408
gdb11	22	45	5	**0.039002**	0.277287	0.315678	**0.785507**	0.918200	0.900487	**0.881007**	0.708175	0.512891
gdb12	13	23	7	**0.005164**	0.016110	0.104416	0.850628	0.832020	0.853378	**0.980243**	0.979106	0.809011
gdb13	10	28	6	0.170071	0.180372	0.273860	0.988458	1.000000	**0.874523**	0.865900	0.862069	0.707721
gdb14	7	21	5	**0.033982**	0.261274	0.352942	**0.769615**	0.863232	0.854531	**0.872280**	0.794213	0.589352
gdb15	7	21	4	**0.050147**	0.349475	0.227593	**0.695741**	0.869075	0.776009	**0.735185**	0.562500	0.556944
gdb16	8	28	5	**0.068613**	0.303918	0.331089	**0.731969**	0.890029	0.844825	**0.797619**	0.608333	0.467989
gdb17	8	28	5	**0.180030**	0.550350	0.380069	**0.700056**	0.931299	0.756089	**0.825556**	0.621111	0.550556
gdb18	9	36	5	**0.064065**	0.314809	0.339070	**0.705372**	0.875640	0.880833	**0.842605**	0.737725	0.533990
gdb19	8	11	3	**0.000000**	**0.000000**	0.217262	0.689926	0.689926	**0.673179**	**0.714286**	**0.714286**	0.504762
gdb20	11	22	4	**0.125009**	0.195592	0.182633	**0.768393**	0.823900	0.823426	**0.819643**	0.776587	0.735615
gdb21	11	33	6	**0.066627**	0.252814	0.237787	**0.777325**	0.914989	0.888137	**0.850412**	0.713228	0.634039
gdb22	11	44	8	**0.066028**	0.143205	0.243994	0.823458	0.911066	0.813657	**0.869808**	0.740383	0.606475
gdb23	11	55	10	**0.104860**	0.144992	0.386921	0.875470	0.881174	0.869281	**0.803514**	0.719563	0.389269

on the results obtained by the 30 runs of the three compared algorithms, and the one that is significantly better than that of the other two (under the significance level of 95%) is marked in bold. Note that I_D and Δ are to be minimized while I_H is to be maximized, the boldfaced I_D and Δ are the smallest while I_H is the greatest among that of the compared algorithms.

First, we focus on the metric I_D. It is shown from the tables that D-MAENS performed significantly the best 20 out of the total 23 instances. LMOGA showed significantly the best performance for 3 instances. NSGA-II was unable to significantly outperform the other two algorithms for any of the instances. Note that for gdb1 and gdb19, D-MAENS and LMOGA both reached the minimal value 0 of I_D. This shows that for this two instances, D-MAENS and LMOGA both consistently found all the Pareto-optimal solutions in all of their 30 runs. For the rest two instances (gdb2, gdb13), D-MAENS still provided the smallest average I_D, although they cannot be statistically distinguished from that of LMOGA.

Then, we consider Δ. From the tables, it is seen that D-MAENS performed significantly better than others for 10 instances. LMOGA performed significantly the best for none of the instance, while NSGA-II performed significantly the best for 3 instances. There are 10 instances on which none of the algorithms showed significantly the best performance. Among them, D-MAENS provided the best Δ for 4 instances. LMOGA obtained the best Δ for 4 instances. NSGA-II got the best Δ for 3 instances. Note that for gdb1 and gdb19, although D-MAENS and LMOGA consistently found all of the Pareto optimal solutions in the 30 runs, their Δ's were either statistically comparable with or significantly outperformed by that of NSGA-II. This shows that the Pareto optimal solutions of these instances are not uniformly distributed.

Finally, the algorithms are compared in terms of I_H. It is observed that D-MAENS provided significantly the greatest I_H for 20 instances. LMOGA performed significantly better than others on 3 instances, while NSGA-II was unable to significantly outperform others for any instance. As mentioned before, D-MAENS and LMOGA both consistently found all the Pareto-optimal solutions in all of their 30 runs for gdb1 and gdb19. Therefore, for this two instances, D-MAENS and LMOGA obtained the same I_H's, which are significantly greater than that obtained by NSGA-II.

VI. CONCLUSION

Being a well known combinatorial optimization problem, CARP has been formulated as a single-objective problem by most of researchers. However, in many real world applications, there usually exist more than one objective, e.g., the total cost and the balance of routes in SRO. In this paper, we investigated the more realistic MO-CARP and attempted to minimize the total cost and makespan simultaneously.

As a multi-objective combinatorial optimization problem, MO-CARP contains the difficulties caused by both MOP and combinatorial optimization problem. In our research, we discuss the important issues in solving MO-CARP with EAs and evaluate the existing strategies for addressing these issues in the case of MO-CARP. Guided by the discussions, we incorporate the strength of SO-CARP approaches and EMO strategies by combining the competitive SO-CARP approach MAENS and proper EMO strategies including a decomposition-based framework, and proposed the resultant D-MAENS. Experimental studies were carried out by comparing D-MAENS with LMOGA and NSGA-II on the gdb test set, and the results demonstrate that D-MAENS performed significantly better than the other compared algorithms. This verifies the efficacy of the incorporation of the strength of SO-CARP approaches and EMO strategies and the importance of employing local search in solving MO-CARP.

ACKNOWLEDGEMENT

This work is supported by IEEE Walter Karplus Summer Research Grant, the Fund for Foreign Scholars in University Research and Teaching Programs (Grant No. B07033) and an EPSRC grant (EP/E058884/1) on "Evolutionary Algorithms for Dynamic Optimisation Problems: Design, Analysis and Applications."

REFERENCES

[1] H. Handa, D. Lin, L. Chapman, and X. Yao, "Robust solution of salting route optimisation using evolutionary algorithms," *in Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, pp. 3098–3105, 2006.

[2] H. Handa, L. Chapman, and X. Yao, "Robust route optimization for gritting/salting trucks: a CERCIA experience," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 6–9, 2006.

[3] M. Dror, *Arc routing. Theory, solutions and applications*. Boston: Kluwer Academic Publishers, 2000.

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

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

[6] G. Ulusoy, "The fleet size and mix problem for capacitated arc routing," *European Journal of Operational Research*, vol. 22, no. 3, pp. 329–337, 1985.

[7] A. Hertz, G. Laporte, and M. Mittaz, "A tabu search heuristic for the capacitated arc routing problem," *Operations Research*, vol. 48, no. 1, pp. 129–135, 2000.

[8] P. Lacomme, C. Prins, and W. Ramdane-Chérif, "Competitive memetic algorithms for arc routing problem," *Annals of Operational Research*, vol. 131, no. 1-4, pp. 159–185, 2004.

[9] J. Brandao and R. Eglese, "A deterministic tabu search algorithm for the capacitated arc routing problem," *Computers and Operations Research*, vol. 35, no. 4, pp. 1112–1126, 2008.

[10] Y. Mei, K. Tang, and X. Yao, "A Global Repair Operator for Capacitated Arc Routing Problem," *IEEE Transactions on System, Man and Cybernetics, Part B*, vol. 39, no. 3, pp. 723–734, 2009.

[11] ——, "Improved Memetic Algorithm for Capacitated Arc Routing Problem," *in Proceedings of the 2009 IEEE Congress on Evolutionary Computation (CEC'09)*, pp. 1699–1706, 18-21 May, Trondheim, Norway, 2009.

[12] K. Tang, Y. Mei, and X. Yao, "Memetic Algorithm with Extended Neighborhood Search for Capacitated Arc Routing Problems," *IEEE Transactions on Evolutionary Computation*, to appear (DOI: 10.1109/TEVC.2009.2023449).

[13] C. A. C. Coello, "Evolutionary multi-objective optimization: a historical view of the field," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 28–36, 2006.

[14] P. Lacomme, C. Prins, and M. Sevaux, "A genetic algorithm for a bi-objective capacitated arc routing problem," *Computers and Operations Research*, vol. 33, no. 12, pp. 3473–3493, 2006.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[16] E. W. Dijkstra, "A Note on Two Problems in Connection with Graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[17] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," *in Proceedings of the 1st International Conference on Genetic Algorithms*, pp. 93–100, 1985.

[18] J. D. Knowles and D. W. Corne, "The Pareto archived evolution strategy: a new baseline algorithmfor Pareto multiobjective optimisation," *in Proceedings of the 1999 IEEE International Conference on Evolutionary Computation*, pp. 98–105, 1999.

[19] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[20] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," *in Proceedings of Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pp. 95–100, Athens, Greece, 2002.

[21] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[22] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR Spectrum*, vol. 22, no. 4, pp. 425–460, 2000.

[23] ——, "Approximative solution methods for multiobjective combinatorial optimization," *Top - The Journal of the Spanish Statistical and Operations Research Society*, vol. 12, no. 1, pp. 1–63, 2004.

[24] H. Ishibuchi and T. Murata, "Multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 28, no. 3, pp. 392–403, 1998.

[25] A. Jaszkiewicz, "Genetic local search for multi-objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, no. 1, pp. 50–71, 2002.

[26] ——, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - a comparative experiment," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 402–412, 2002.

[27] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[28] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," *in Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02)*, vol. 1, pp. 825–830, 2002.

[29] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR Spectrum*, vol. 22, no. 4, pp. 425–460, 2000.

[30] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 28, no. 3, pp. 392–403, 1998.

[31] J. S. DeArmon, "A comparison of heuristics for the capacitated Chinese postman problem," Master's thesis, University of Maryland, College Park, MD, 1981.

[32] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[33] P. Czyzzak and A. Jaszkiewicz, "Pareto simulated annealing - a metaheuristic technique for multiple-objective combinatorial optimization," *Journal of Multi-Criteria Decision Analysis*, vol. 7, no. 1, pp. 34–47, 1998.