This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Decomposition-Based Memetic Algorithm for Multiobjective Capacitated Arc Routing Problem

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

***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. There have been many attempts at solving CARP using heuristic and meta-heuristic approaches, including evolutionary algorithms. However, almost all such attempts formulate CARP as a single-objective problem although it usually has more than one objective, especially considering its real-world applications. This paper studies multiobjective CARP (MO-CARP). A new memetic algorithm (MA) called decomposition-based MA with extended neighborhood search (D-MAENS) is proposed. The new algorithm combines the advanced features from both the MAENS approach for single-objective CARP and multiobjective evolutionary optimization. Our experimental studies have shown that such combination outperforms significantly an off-the-shelf multiobjective evolutionary algorithm, namely nondominated sorting genetic algorithm II, and the state-of-the-art multiobjective algorithm for MO-CARP (LMOGA). Our work has also shown that a specifically designed multiobjective algorithm by combining its single-objective version and multiobjective features may lead to competitive multiobjective algorithms for multiobjective combinatorial optimization problems.**

***Index Terms*—Capacitated arc routing problem (CARP), local search, memetic algorithms (MA), meta-heuristics, multiobjective optimization.**

I. Introduction

THE CAPACITATED arc routing problem (CARP) [1] is a well-known combinatorial optimization problem. Due to its wide applications in the real world, including winter gritting [2]–[6], urban waste collection [7], [8], and snow removal [9], [10], CARP has been intensively investigated in the past few decades. Given a graph with some edges and arcs required to be served (called tasks) and a number of vehicles with limited capacity, a CARP is defined as seeking an optimal routing plan for the vehicles under the following conditions.

1) Each vehicle starts and ends at a predefined vertex, namely depot.
2) Each task is served by exactly one vehicle.
3) The total demand of the tasks served by each vehicle does not exceed its capacity.

Since CARP is NP-hard [11], exact methods are only applicable to the instances with small problem sizes. However, many real-world applications involve large-size CARPs, and routing plans must be made within a restricted time budget. Therefore, heuristics and meta-heuristics are promising approaches in such a situation in order to obtain acceptable solutions in time. During the last century, constructive heuristics were often adopted because of their ability to generate relatively good solutions in a very short time period. To name a few, the Augment-Merge heuristic proposed by Golden and Wong [11], the path scanning heuristic proposed by Golden *et al.* [12], and Ulusoy's splitting heuristic proposed in [13] are three typical heuristics for CARP. More recently, researchers shifted their attentions to meta-heuristics, which can provide much better solutions. Although meta-heuristics usually induce higher computational cost, this additional cost is now affordable due to the rapid development of the computational power of computers. The first meta-heuristic approach to CARP is the tabu search proposed by Hertz *et al.* [14]. After that, the variable neighborhood descent algorithm [15], the guided local search [16], the tabu scatter search [17], the memetic algorithm (MA) [18], and another tabu search algorithm [19] have been proposed. A comprehensive survey of the recent results on various arc routing problems is presented in [20]. We have also conducted intensive investigations on CARP in our previous work. A global repair operator that can be embedded in any search-based approach was proposed in [21]. More importantly, we proposed a MA with extended neighborhood search (MAENS) [22], which has been shown to outperform most existing approaches in terms of solution quality.

So far, CARP has been predominantly formulated as a single-objective problem with the only objective of minimizing the total cost of the service. However, there is a huge gap between such a formulation and reality. Contributions are now needed to fill this gap in literature. For this purpose, Lacomme *et al.* [23] considered minimizing total cost and makespan (i.e., the cost of the longest route) simultaneously. Specifically,

Manuscript received October 6, 2009; revised February 7, 2010 and April 14, 2010. This work was partially supported by the National Natural Science Foundation of China, under Grants 60802036, U0835002, and 61028009, by the Fund for Foreign Scholars in University Research and Teaching Programs, under Grant B07033, and by the ESPRC, under Grant EP/E058884/1, on "Evolutionary Algorithms for Dynamic Optimization Problems: Design, Analysis, and Applications."

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 and also with the Center of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, U.K. (e-mail: x.yao@cs.bham.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2010.2051446

1089-778X/$26.00 © 2010 IEEE

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

they formulated a multiobjective CARP (MO-CARP) and developed a hybrid algorithm for it by combining an approach for single-objective CARP (SO-CARP) [18] and a commonly used multiobjective evolutionary algorithm (MOEA), namely Nondominated Sorting Genetic Algorithm II (NSGA-II) [24].

The two objectives considered by Lacomme *et al.* in [23] are conflicting with each other. Thus, no unique global optimal solution exists in this case. Instead, Lacomme *et al.* proposed a multiobjective genetic algorithm (referred to as LMOGA in this paper) to maintain a set of solutions, which are good "tradeoffs" between the two objectives. Essentially, the MO-CARP lies in the reign of multiobjective optimization. Numerous previous publications have shown that MOEAs are good approaches to this kind of problem. Nevertheless, how to make the best use of MOEAs in the context of MO-CARP has not been fully investigated. Motivated by this, this paper aims to contribute from two aspects. First, a number of important issues for evolutionary multiobjective optimization (EMO) are discussed, and the utility of existing EMO strategies in the context of MO-CARP is examined. Second, an algorithm named decomposition-based MAENS (D-MAENS) is proposed. The D-MAENS employs the framework of the MOEA based on decomposition (MOEA/D), with MAENS embedded in it. In addition, it adopts proper EMO strategies based on domain-specific considerations of MO-CARP. Comparative studies are also presented to evaluate the efficacy of D-MAENS.

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

II. Background

A. Multiobjective CARP

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. For each edge $(v_i, v_j) \in E$ and arc $\langle v_i, v_j\rangle \in A$, three nonnegative features are associated, i.e., the traversal cost $c^{trav}(v_i, v_j)$, the serving cost $c^{serv}(v_i, v_j)$, and the demand $d(v_i, v_j)$. An edge or arc with a positive demand is called a task, and is required to be served by vehicles at the cost of its serving cost. We denote the edge task set as $E_R = \{(v_i, v_j) \in E | d(v_i, v_j) > 0\}$ and the arc task set as $A_R = \{(v_i, v_j) \in A | d(v_i, v_j) > 0\}$. Then, the task set is $T = E_R \cup A_R$. Note that the serving cost is only induced by serving a task, we have $c^{serv}(v_i, v_j) > 0 \iff d(v_i, v_j) > 0$ and $c^{serv}(v_i, v_j) = 0 \iff d(v_i, v_j) = 0$. m vehicles with an identical capacity Q are based at the depot $v_s \in V$ to serve the tasks. For an edge task (v_i, v_j), service in either direction is acceptable. In order to facilitate the problem definition, each edge task is assigned two IDs (say t_1 and t_2), one for each direction, and each arc task is assigned one ID. All the IDs are unique positive integers. For an ID $t \in \mathbb{N}^+$, the following six features are associated: 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 ID $inv(t)$. For an edge task (v_i, v_j), these features are defined as follows:

1) $hv(t_1) = tv(t_2) = v_i$;
2) $tv(t_1) = hv(t_2) = v_j$;
3) $c^{trav}(t_1) = c^{trav}(t_2) = c^{trav}(v_i, v_j)$;
4) $c^{serv}(t_1) = c^{serv}(t_2) = c^{serv}(v_i, v_j)$;
5) $d(t_1) = d(t_2) = d(v_i, v_j)$;
6) $inv(t_1) = t_2$, $inv(t_2) = t_1$.

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

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

Since all the IDs are positive, $inv(t) = -1$ indicates the inverse ID of t does not exist. In addition, zero is defined as the ID of the depot loop with the following definitions:

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

Using the above notations, a CARP solution can be represented as a set of routes $S = (R_1, R_2, \ldots, R_m)$. Each route R_k is a sequence of the IDs, i.e., R_k=$(t_1^k, t_2^k, \ldots, t_{l_k}^k)$, where $t_p^k (1 \leqslant p \leqslant l_k)$ are the IDs. 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, the edge task set $E_R = \{(v_1, v_5), (v_2, v_6), (v_3, v_7), (v_4, v_8)\}$, and the depot is v_0. There is no arc task in this case. The task IDs 1, 2, 3, and 4 are assigned to $\langle v_1, v_5\rangle$, $\langle v_2, v_6\rangle$, $\langle v_3, v_7\rangle$, and $\langle v_4, v_8\rangle$, respectively, while 5, 6, 7, and 8 are assigned to their inversions. There are two numbers associated with each edge task, the one out of the parenthesis denotes the task ID of the direction traversed by the route, while the other one denotes its inversion. The dashed arrows between adjacent task IDs (e.g., $\langle v_0, v_1\rangle$ and $\langle v_7, v_6\rangle$ in Fig. 1) stand for the intermediate paths.

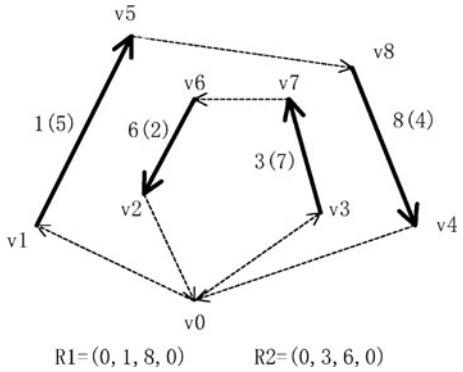


Fig. 1. Example of a CARP solution

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

For each route $R_k = (t_1^k, t_2^k, \ldots, t_{l_k}^k)$, its total cost $c^{tot}(R_k)$ and total demand $d(R_k)$ can be calculated as

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

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

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

Under such a solution representation scheme, the MO-CARP can be represented as follows:

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

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

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

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

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

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

where the inequation $(k_1, p_1) \neq (k_2, p_2)$ 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. Equation (1) is the total cost of all the routes and (2) is the makespan. Constraints (3)–(5) guarantee that each task is served exactly once by one vehicle. Constraints (6), which are also called the capacity constraints, indicate that the total demand served by each vehicle does not exceed its capacity.

B. Evolutionary Multiobjective Optimization Revisited

A multiobjective optimization problem can be briefly stated as follows:

$$\min \ \boldsymbol{F}(x) = (f_1(x), \ldots, f_n(x))$$
$$\text{s.t.} : \ x \in \Omega$$

where Ω is the decision variable space. $\boldsymbol{F}{:}\Omega \rightarrow R^n$ consists of n objective functions that are conflicting with each other. For a multiobjective optimization problem, one aims to seek a set of solutions that have good tradeoffs among the objectives. In order to make a clear notion of optimality in this scenario, Pareto defined domination relationship and Pareto optimality [25]. Let $\boldsymbol{u}, \boldsymbol{v} \in R^n$, $\boldsymbol{u}$ *dominates* $\boldsymbol{v}$ if and only if $u_i \leqslant v_i$ for each $i \in \{1, \ldots, n\}$ and $u_j < v_j$ for at least one $j \in \{1, \ldots, n\}$. Then, a decision variable $x^* \in \Omega$ is said to be *Pareto optimal* if there is no other $x \in \Omega$ so that $\boldsymbol{F}(x)$ dominates $\boldsymbol{F}(x^*)$. With the above definitions, a multiobjective optimization problem requires finding or approximating the set of Pareto optimal solutions and their corresponding objective vectors (called *Pareto front*). Hence, a MOEA should return a set of nondominated solutions that can well approximate the Pareto optimal solutions [26].

There are three important issues that must be addressed in EMO, i.e., fitness assignment, diversity preservation and elitism. Unlike in single-objective optimization problems, the fitness of a solution needs to be assigned according to multiple criteria in a multiobjective optimization problem. Diversity preservation is important for MOEAs to obtain solutions that are uniformly distributed on the Pareto front. Elitism is implemented in MOEAs to keep the nondominated solutions in the population during the search. These three issues have been addressed in various ways, and thereby numerous MOEAs have been proposed (see [24]–[29]).

Despite the lack of research on MO-CARP, evolutionary multiobjective combinatorial optimization has attracted a lot of interest. Ehrgott *et al.* gave a survey of multiobjective combinatorial optimization problems [30], and introduced the characteristics of the problems and nominated MOEAs as one available methodology. Some examples of the approximative solution methods in multiobjective combinatorial optimization problems were presented in [31], including various MOEAs, simulated annealing and tabu search. In addition to directly using the traditional MOEAs, some researchers considered combining the MOEA framework with local search to pursue enhanced performance. For example, Ishibuchi *et al.* proposed a genetic local search for solving the flowshop scheduling problem [32]. Jaszkiewicz proposed a genetic local search framework for multiobjective combinatorial optimization problems to determine the weight vectors used in the weighted sum approach for aggregating the objective functions during the local search, and successfully applied it to the traveling salesman problem [33] and the 0/1 knapsack problem [34]. Tan *et al.* developed a MOEA for solving a multiobjective vehicle routing problem in [35]. The algorithm incorporates two problem-specific heuristics for local exploitation. However, due to different structures of combinatorial optimization problems, it is often difficult to directly apply a MOEA developed for one problem to another. For the same reason, although traditional MOEAs have shown satisfactory performance on numerical optimization benchmark test functions, they do not necessarily guarantee good performance on MO-CARP. First, our preliminary studies showed that the problem natures of SO-CARP such as the discrete search space, the lack of a natural definition of neighborhood and various constraints made successful algorithms for numerical optimization problems failed on SO-CARP. This phenomenon may also occur in the multiobjective case. Second, the shape of the Pareto front can directly influence the performance of MOEAs [36]. Therefore, the difference between the shape of the Pareto fronts of MO-CARP and the numerical test functions makes the performance of an existing MOEA in the case of MO-CARP unpredictable.

In general, MO-CARP can be solved from two different directions. One is to extend an approach for SO-CARP to a multiobjective one, and the other is to directly use an existing MOEA by employing the problem-specific solution representation and operators. Lacomme *et al.* followed the former direction in [23], while the latter direction has been overlooked

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

so far. Both methods have their advantages and disadvantages, and are actually complementary to each other. When extending a SO-CARP approach, the strength of searching in the complicated solution space can be inherited. However, the EMO issues need to be addressed appropriately. On the other hand, when applying an existing MOEA, it is difficult to search effectively in the solution space of MO-CARP, although the EMO issues are deeply considered. Therefore, it is reasonable to increase the synergy between the two directions so that both of their drawbacks can be overcome. In this paper, we consider incorporating an existing SO-CARP approach and various strategies proposed for EMO issues. By this means, the hybridized algorithm will be strong in both searching within the solution space and addressing the EMO issues. In order to accomplish this, it is necessary to evaluate the EMO strategies in the context of MO-CARP, as will be presented in the next section.

III. EMO Issues in MO-CARP

MO-CARP is a combinatorial problem that tries to find a set of Pareto optimal feasible solutions in a discrete and finite solution space subject to a number of constraints. The hybridization of EA with local search has been reported to be quite efficient for solving combinatorial problems including SO-CARP (see [18], [22]). This hybridized approach is also called MA. When combining a MOEA with local search, a new important issue arises. That is, how to identify a solution in the neighborhood to replace the current solution. Usually, the best solution in the neighborhood is selected, and thus the issue can be seen as identifying the best solution in the neighborhood. Therefore, when solving MO-CARP with evolutionary algorithms (EAs), one may have to consider the following four important issues: 1) fitness assignment; 2) diversity preservation; 3) elitism; and 4) identifying the best neighboring solution during local search. The first three issues are commonly considered in EMO, while the last one must be addressed when local search is employed. The existing strategies for addressing these issues are evaluated in the case of MO-CARP one by one.

A. Fitness Assignment in MO-CARP

The existing strategies for fitness assignment in EMO can be categorized into three types: 1) the criterion-based (see [27]); 2) domination-based (see [24]); and 3) decomposition-based (see [29]) methods. Previous studies on numerical test functions showed that the criterion-based methods will overlook the intermediate regions of the Pareto front, while the domination-based methods will not. Since in criterion-based and domination-based methods, the fitness of a solution only depends on the values of objective functions, the conclusions drawn from numerical test functions should still hold in the case of MO-CARP. Hence, domination-based methods are more appropriate than criterion-based methods in this context. On the other hand, the decomposition-based methods are based on the assumption that each Pareto optimal solution can be seen as the optimal solution to a scalar optimization subproblem. However, this is not true in the case of MO-CARP. In fact, there usually exist solutions which are not optimal for any weighted sum of the objectives in MO-CARP [37]. Furthermore, due to the discreteness of the Pareto front in MO-CARP, one Pareto optimal solution may be the optimal solution of multiple decomposed subproblems. The search process may be hindered since most of computing resources may be wasted to find the same Pareto optimal solution. Therefore, the decomposition-based methods may not perform well in MO-CARP.

B. Diversity Preservation in MO-CARP

The niching technique, cell-based methods and crowding distance method are three typical existing strategies for diversity preservation. They are mainly based on preventing solutions close to each other from appearing simultaneously in the population. Therefore, they are expected to be able to maintain diversity in MO-CARP as well. Among them, the performance of the niching technique and cell-based methods are parameter-dependent, i.e., they largely depend on the parameters such as the sharing parameter in the niching technique and the cell size in the cell-based methods. The performance of the crowding distance method is expected to have a small variance since it has no user-defined parameter. In addition to the above three strategies, algorithms like MOEA/D utilize an implicit strategy to maintain diversity. That is, the diversity is naturally preserved by the "diversity" among subproblems [29]. However, this is based on the assumption that different subproblems can reach different optimal solutions. In MO-CARP, one Pareto optimal solution can be the optimal solutions to multiple subproblems. As a result, the diversity can no longer be maintained in this way.

C. Elitism in MO-CARP

The elitism mechanism can be implemented by either storing the nondominated solutions in an archive or combining the parents and offsprings for selection. The archive strategy can be further divided into two types: the solutions stored in the archive do or do not influence the search process. There is no big difference when they are adopted in the test functions and MO-CARP. Therefore, they can be applied to MO-CARP in exactly the same way as to the test functions.

D. Evaluating Solutions During Local Search in MO-CARP

Identifying the best neighboring solution is essentially equivalent to assigning fitness to each solution and then selecting the one with the best fitness. Therefore, this issue can be examined from the perspective of fitness assignment. The criterion-based and domination-based strategies divide the solutions into different fronts, each of which consists of solutions with the same fitness. In this way, one can hardly tell which solution in each front is the best one. The only available strategy is to use decomposition-based methods. When solving each decomposed scalar subproblem, the best solution can be easily identified during local search. In addition to the decomposition-based methods, aggregating objective functions into a single one has been a commonly used idea in the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

literature (see [32], [33]). This method is usually faster than the domination-based methods, but its performance largely depends on the weight vector.

Based on the evaluations of the existing strategies, one can either select an existing MOEA for solving MO-CARP or design a specialized algorithm according to practical requirements. Based on the previous discussions, it can be seen that the existing MOEAs other than MOEA/D are able to address the first three issues well. MOEA/D is the only algorithm that can address the last issue in MO-CARP because its distinctive decomposition-based framework provides a natural way to employ local search. Therefore, we propose a multiobjective MA, named D-MAENS, to address all four issues properly. In order to keep the algorithm simple, among the available strategies, the ones with the least parameters are employed. Specifically, D-MAENS employs the fast nondominated sorting procedure of NSGA-II for fitness assignment, the crowding distance method of NSGA-II for diversity preservation, both of the two existing strategies for elitism and the decomposition strategy of MOEA/D for identifying the best solutions during local search. Next section describes the full details of D-MAENS.

IV. D-MAENS

D-MAENS adopts a decomposition-based framework which is analogous with that of MOEA/D. It decomposes the original MO-CARP into a number of scalar subproblems using the weighted sum approach with a set of uniformly distributed weight vectors. A population of individuals (solutions) whose size equals the number of the subproblems is maintained. Each individual in the population corresponds to a unique subproblem. When solving each subproblem, the evolutionary operators and local search are applied to the individuals corresponding to the neighboring subproblems of the current one. The crossover operator and local search process are exactly those employed in MAENS. We will describe the decomposition-based framework of D-MAENS in Section IV-A. Then, the MAENS ingredients will be briefly introduced in Section IV-B.

A. Decomposition-Based Framework

In the decomposition framework, the original MO-CARP is first decomposed into a number of SO-CARPs by the weighted sum approach. To be specific, given the objective vector $\boldsymbol{F}(x) = (f_1(x), \ldots, f_n(x))$ and a weight vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)$, the objective function of a subproblem is stated as

$$g^{ws}(x|\boldsymbol{\lambda}) = \sum_{i=1}^{n} \lambda_i f_i(x).$$

Suppose there are N weight vectors $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^N$, the original MO-CARP is thus decomposed into N SO-CARPs. The objective function of the ith subproblem is $g^{ws}(x|\boldsymbol{\lambda}^i)$. D-MAENS maintains a population $X = \{x_1, \ldots, x_N\}$ throughout the optimization process. At each generation, the population is evolved in the following steps. First, each subproblem is assigned a unique solution $x \in X$, which is called its representative. Then, N subpopulations are constructed, each for a subproblem. The subpopulation of a subproblem associated with weight vector $\boldsymbol{\lambda}^i$ is composed of the representatives of the T subproblems whose associated weight vectors are the T closest (in term of Euclidean distance) weight vectors to $\boldsymbol{\lambda}^i$, where T is the size of subpopulation. As stated in [29], the optimal solution of the ith subproblem should be close to that of the jth subproblem if $\boldsymbol{\lambda}^i$ is close to $\boldsymbol{\lambda}^j$. Thus, the information of the subproblems whose weight vectors are close to that of the current subproblem should be helpful for solving the current subproblem. Second, one new solution is generated for each subproblem. For the ith subproblem, two parents are selected from the subpopulation associated with it, and then crossover and local search of MAENS are applied to the parents to generate an offspring y_i. By repeating this procedure for all subproblems, an offspring population $Y = \{y_1, \ldots, y_N\}$ is generated. Finally, the solutions in both X and Y are combined together and then sorted by the fast nondominated sorting procedure and the crowding distance method. The best N solutions are kept to form the population X in the next generation. The detailed steps of the decomposition-based framework are given as below.

Input:

1) a MO-CARP instance P;
2) a stopping criterion;
3) the number of decomposed subproblems, denoted as N;
4) a number of uniformly distributed weight vectors $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^N$;
5) the size of the neighborhood of each subproblem, denoted as T.

Output: A set of nondominated solutions X^*.

Step 1: Initialization.

a) Set $X^* = \emptyset$.
b) Decompose the original MO-CARP P into a set of SO-CARPs $\{P_1, P_2, \ldots, P_N\}$ with $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^N$.
c) Initialize a population $X = \{x_1, \ldots, x_N\}$ randomly or by problem-specific methods.
d) Compute the Euclidean distance between each pair of weight vectors. Then, get the neighborhood $B(i) = \{i_1, \ldots, i_T\}$ for each P_i, so that $\boldsymbol{\lambda}^{i_1}, \ldots, \boldsymbol{\lambda}^{i_T}$ are the T closest weight vectors to $\boldsymbol{\lambda}^i$ (including $\boldsymbol{\lambda}^i$ itself).

Step 2: Search for new solutions.

a) Assign each subproblem a unique representative $x_i^r \in X$.
b) Construct a subpopulation $X_i = \{x_{i_1}^r, \ldots, x_{i_T}^r\}$ for each subproblem.
c) Set $i = 1$.
d) Randomly select two solutions x_k^r and x_l^r from X_i.
e) Apply the crossover and local search operators of MAENS to x_k^r and x_l^r to generate y_i for P_i.
f) Remove from X^* all the vectors dominated by $\boldsymbol{F}(y_i)$. Insert $\boldsymbol{F}(y_i)$ in X^* if no vector in X^* dominates it.
g) Set $i \rightarrow i + 1$. If $i \leqslant N$, go back to Step 2d.
h) Sort the solutions in the set $Z = X \cup Y$ by the fast nondominated sorting procedure and crowding distance approach of NSGA-II [24]. Then, let X be the set of the best N solutions in the sorted Z.

Step 3: Termination. If stopping criteria are satisfied, terminate the algorithm. Otherwise, go to Step 2.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 1: The assignment of representatives

Input: A population $X = \{x_1, \ldots, x_N\}$;
Output: A representative set $\{x_1^r, \ldots, x_N^r\}$;
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**

In Step 2a, a representative x_i^r needs to be assigned to subproblem P_i. It is natural to assign the solution which is the best one for P_i as the corresponding representative. However, it may occur that some solutions are assigned to multiple subproblems while some others are never selected. To make each subproblem be assigned with a unique solution, a representative assignment scheme has been further developed. In the case of the MO-CARP, two objectives are to be minimized and the weight vectors $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^N$ are defined as

$$\boldsymbol{\lambda}^i = \left(\frac{i-1}{N-1}, \frac{N-i}{N-1} \right).$$

As i increases, the importance in the aggregated objective function decreases on f_1 and increases on f_2. Therefore, we designed Algorithm 1 to sort the population based on the two objective functions. Then, the ith solution in the sorted population is assigned to P_i.

Since the decomposition-based framework of D-MAENS is analogous with that of MOEA/D, it is necessary to make comparison between them. D-MAENS differs from MOEA/D in the following aspects. First, in MOEA/D, the solution replacement is immediately done once an offspring is generated at each generation, while in D-MAENS, it is called after all subproblems have been solved. In this way, changing the order of solving the subproblems will make no difference in the framework of D-MAENS. Second, in our framework, representatives of the subproblems are re-assigned at each generation, while MOEA/D carries out the assignment at the initialization phase only. In this way, each subproblem can be assigned a more appropriate representative according to the information of the current population during the search process. Finally, in contrast to the solution replacement of MOEA/D, in which a solution of a subproblem can only be replaced by a solution generated for the same subproblem, D-MAENS combines all the solutions together and compares them regardless of which subproblem they belong to. By combining all the solutions together, the subproblems can help each other by sharing their representatives. In addition, the crowding distance approach of NSGA-II prevents the diversity loss caused by the local search operator of MAENS. Moreover, elitism is implemented in D-MAENS by both using an archive and maintaining elite solutions during the search in D-MAENS. This makes it more likely to capture the whole Pareto front when the number of Pareto optimal solutions is larger than the population size.

B. MAENS Component

The previous section describes a general framework that solves a MO-CARP by decomposing it into a number of single-objective subproblems. In general, any approach to SO-CARP can be embedded in this framework. We propose employing MAENS due to its appealing performance. As a MA, MAENS is characterized by five issues: 1) the solution representation; 2) the evolutionary operator; 3) the local search operator; 4) the evaluation schemes in the evolutionary phase; and 5) the evaluation schemes in the local search. D-MAENS adopts the exactly same solution representation, crossover and local search operators as MAENS. Hence, we refer interested readers to the original publication [22] for the full details of them. Since MAENS needs to solve a single-objective problem that is a bit different from the traditional CARP, the evaluation schemes in both evolutionary and local search phases have been modified accordingly, as described below.

The SO-CARP to be solved by MAENS in the framework of D-MAENS takes the following form:

$$\min \quad g(S) = \lambda_1 f_1(S) + \lambda_2 f_2(S) \tag{7}$$

$$\text{s.t.} : \quad \sum_{k=1}^{m} (l_k - 2) = |T| \tag{8}$$

$$t_{p_1}^{k_1} \neq t_{p_2}^{k_2}, \qquad \forall (k_1, p_1) \neq (k_2, p_2) \tag{9}$$

$$t_{p_1}^{k_1} \neq inv(t_{p_2}^{k_2}), \quad \forall (k_1, p_1) \neq (k_2, p_2) \tag{10}$$

$$d(R_k) \leqslant Q, \qquad \forall 1 \leqslant k \leqslant m. \tag{11}$$

In the above definition, $f_1(S)$ and $f_2(S)$ can be directly set to the two objective functions (i.e., $c^{tot}(S)$ and $c^{\max}(S)$) of the MO-CARP. However, $c^{tot}(S)$ and $c^{\max}(S)$ are of different scales, and the direct use of them will make MAENS bias more to $c^{tot}(S)$. Therefore, normalization is required. Ideally, the normalized objective functions should be

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

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

where c_*^{tot} and c_{**}^{tot} are the minimal and maximal total costs of all possible solutions, while $c_*^{\max}$ and $c_{**}^{\max}$ stand for the minimal and maximal makespan. However, in practice, we cannot exhaustively enumerate all possible solutions to get these values. Hence, one has to replace them with approximated values. Here, we set them as the minimal (maximal) values among the total cost (makespan) of all feasible solutions that have been found so far. Concretely, they are firstly set to

$$c_*^{tot} = c_*^{\max} = \alpha$$

$$c_{**}^{tot} = c_{**}^{\max} = 0$$

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

where α is a sufficiently large number. Then, they are updated during the search process. In summary, solutions are evaluated with the weighted sum of the normalized objective functions during the evolutionary phase of MAENS. In case some infeasible solutions are generated, both (7) and total violation to the capacity constraints (denoted as $tvl(S)$) will be considered. Stochastic ranking [38] will be used in the same way as the original MAENS.

As for local search, (7) is used as the objective function, while infeasible solutions are handled in a way more efficient than stochastic ranking. When comparing two infeasible solutions, they are first compared in terms of $tvl(S)$, and then in terms of $g(S)$ if they are equal in $tvl(S)$. Concretely, given two candidate solutions S_1 and S_2, S_1 is said to be better than S_2 if $tvl(S_1) < tvl(S_2)$ or $tvl(S_1) = tvl(S_2)$ and $g(S_1) < g(S_2)$. In case of a draw, no replacement will occur.

C. Comparisons Between D-MAENS and LMOGA

Comparing D-MAENS with the only existing approach to MO-CARP, LMOGA [23], it can be seen that they have the same selection operator, which is the combination of the fast nondominated sorting procedure and the crowding distance approach. However, they are totally different in the ways of generating offspring. At each generation, D-MAENS decomposes the original MO-CARP into multiple SO-CARPs, and then generates one offspring for each subproblem. LMOGA, on the other hand, solves the MO-CARP as a whole. Concretely, their differences in generating offspring lie in the following two aspects. First, in D-MAENS, the parents for a certain subproblem are selected from the predefined neighborhood of the corresponding representative solution, while in LMOGA, the parents are consistently selected from the whole population. Second, although LMOGA also transforms the MO-CARP into a SO-CARP by weighted sum when carrying out local search, the weight vector is determined based on the location of the objective vector of the offspring in the objective space. In contrast, the weight vectors are set to fixed values in the initialization phase in D-MAENS and kept unchanged throughout the search process.

Moreover, D-MAENS and LMOGA employ different solution representations and evaluation schemes. LMOGA utilizes an implicit solution representation, in which a solution is represented as a single task sequence and the capacity constraint is temporarily neglected. When a solution is evaluated, it is first split into a set of feasible routes so that the additional cutting cost induced is minimized. Since such a decoding procedure does not take the makespan into account, LMOGA might be strong in seeking solutions with a low total cost, but weak in finding low-makespan solutions. The explicit solution representation employed in D-MAENS, on the other hand, addresses both total cost and makespan since a solution is directly evaluated without being transformed by any decoding procedure.

V. Experimental Studies

In order to evaluate the efficacy of incorporating the strength of SO-CARP approaches and EMO strategies, we compared D-MAENS with LMOGA and NSGA-II on three benchmark test instances. Here, LMOGA was chosen since it was the only published algorithm proposed for MO-CARP and represented the way of extending an approach of SO-CARP to solve MO-CARP. NSGA-II served as a representative of the traditional MOEAs without local search and represented the way of directly using an existing MOEA for MO-CARP.

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	-	-

n is the number of tasks in the problem.

A. Experimental Setup

The experiments were carried out on three well-known CARP benchmark sets, i.e., the *gdb* set [39], the *val* set [40] and the *egl* set [41]–[43]. The *gdb* set was generated by DeArmon in [39] and consists of 23 instances, most of which are small-size instances. The *val* set was generated by Benavent *et al.* [40]. It contains 34 instances based on ten different graphs. Different instances based on each graph were generated by changing the capacity of the vehicles. The *val* instances have larger problem sizes than the *gdb* instances. The *egl* set was generated by Eglese based on data from a winter gritting application in Lancashire [41]–[43]. It consists of 24 instances based on two graphs, each with distinct set of required edges and capacity constraints. They have the largest problem sizes among the three benchmark sets. In total, 81 instances were used in our studies, with their complexities from easy to hard.

In our experiments, the MO9 version of LMOGA proposed in [23] was selected, for it showed the best performance on the test instances among all the nine LMOGA versions. NSGA-II was originally proposed for numerical optimization problems. In order to apply it to MO-CARP, problem-specific solution representation and operators need to be employed. In our experiments, the solution representation and crossover operator of D-MAENS were directly employed in NSGA-II, and 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. We set the parameters in such a way that the three algorithms shared the same key parameters, such as population size, crossover rate, and mutation rate. Since NSGA-II does not employ a local search process, it is assigned a larger generation number. For each test instance, 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 a MOEA is usually evaluated from two aspects. First, the obtained nondominated set should be as

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

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. The two aspects can hardly be reflected by a single metric, and a number of metrics have been suggested in [44]. In this paper, the following three metrics are used.

1) *Distance From Reference Set (I_D):* This metric was suggested by Czyzzak *et al.* [45]. It is defined as follows:

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

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. If the reference set R is defined as a set of Pareto optimal solutions whose objective vectors are uniformly distributed on the Pareto front, I_D will indicate the closeness of the set A to the Pareto front and the distribution of the solutions in A. However, it is difficult to obtain the Pareto optimal solutions in a MO-CARP instance. Furthermore, the Pareto optimal solutions themselves may even be distributed non-uniformly. Alternatively, the nondominated solutions obtained by all the three algorithms in 30 runs on a test instance were combined, and those solutions remained nondominated in this set were used as the reference set in our experiment.

2) *Spread (Δ):* This metric was suggested by Deb *et al.* [24]. It can be stated as follows:

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

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 ith left and the $(i+1)$th left solutions in A. $\bar{d}$ stands for the average over all d_i's. Δ is an indicator of the distribution of solutions. A smaller Δ indicates that the solutions are distributed more uniformly and have 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.

3) *Hypervolume (I_H):* This metric was suggested by Zitzler *et al.* [28] to indicate the area in the objective space that is dominated by at least one solution of the nondominated set. In practice, I_H of a given nondominated set A is calculated as follows:

$$I_H(A) = \int_{z \in \cup_{x \in A} HV(\boldsymbol{f}(x), \boldsymbol{f}^*)} \cdots \int 1 \cdot dz$$

where $HV(\boldsymbol{f}(x), \boldsymbol{f}^*) = [f_1(x), f_1^*] \times \cdots \times [f_m(x), f_m^*]$ is the Cartesian product of the closed intervals $[f_i(x), f_i^*]$, $i = 1, \ldots, m$.

Here we assume the objectives are to be minimized and the reference point $\boldsymbol{f}^* = (f_1^*, \ldots, f_m^*)$ is the ideal worst point, i.e., $f_i^* = \max_{x \in \Omega} f_i(x), \forall i = 1, \ldots, m$. An example with two objectives is given in Fig. 2, where the objective vectors of the solutions in the set A are $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$. The area in shadow indicates the hypervolume of A.

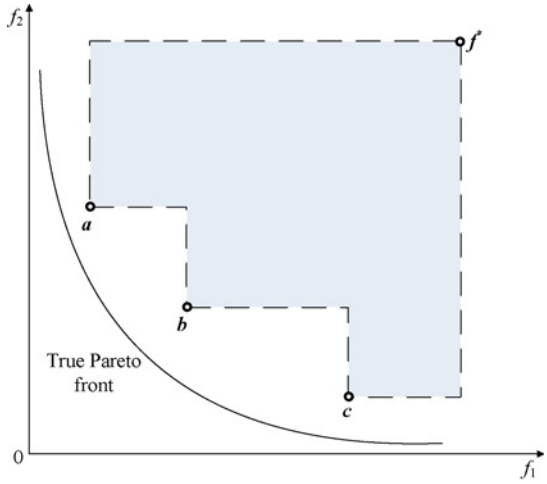


Fig. 2. Hypervolume of a set of nondominated solutions

It is obvious that if solution x_1 dominates solution x_2, then $HV(\boldsymbol{f}(x_1), \boldsymbol{f}^*) \supseteq HV(\boldsymbol{f}(x_2), \boldsymbol{f}^*)$. Hence, I_H reflects the closeness of the nondominated set to the Pareto front. The larger the I_H, the closer the corresponding nondominated set is to the Pareto front. In addition, from the fact that for two sets A_1 and A_2, $A_1 \supseteq A_2 \Rightarrow I_H(A_1) \geqslant I_H(A_2)$, it is deduced that a larger I_H implies that the nondominated set covers the Pareto front more completely. Furthermore, I_H is the only unary measure which is consistent with the Pareto dominance relationship, i.e., if a set dominates another one, it always has a better I_H [46]. For this reason, I_H is one of the most commonly used measures for evaluating MOEAs nowadays.

In our study, all the metrics were computed based on the normalized objective vectors of the nondominated solutions, which were obtained by

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

where f_1 and f_2 stand for the total cost and makespan. $f_i^{\max}$ and $f_i^{\min}$ are the maximal and minimal values of f_i among all the results obtained over the 30 runs of the three compared algorithms. Since the elements of the normalized objective vectors always lie in the interval $[0, 1]$, the point $(1, 1)$ was used as the reference point in our experiment.

C. Experimental Results

Tables II-IV present the average value of I_D, Δ and I_H over the 30 independent runs of the compared algorithms on the three test sets, respectively. The characteristics of the instances such as the number of vertices and edges are also provided. In the tables, the columns headed "$|V|$" and "$|E|$" stand for the number of vertices and edges in the graph of the instance, while the column headed "$|R|$" represents the number of tasks. In the *gdb* and *val* sets, all edges in the graph need to be served. Therefore, the column headed "$|R|$" is omitted in Tables II and III. The column headed "τ" indicates the minimal number

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE II

AVERAGE VALUE OF I_D, Δ AND I_H OVER 30 INDEPENDENT RUNS OF THE COMPARED ALGORITHMS ON THE *gdb* SET

Problem	$\|V\|$	$\|E\|$	τ	I_D			Δ			I_H		
				D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II
*gdb*1	12	22	5	0.000000	0.000000	0.141144	0.737152	0.737152	0.738243	0.933333	0.933333	0.801793
*gdb*2	12	26	6	0.047577	0.055494	0.201959	0.744072	0.775904	0.846040	0.968374	0.968089	0.733586
*gdb*3	12	22	5	**0.008124**	0.049914	0.190699	0.835997	0.879512	0.820943	**0.960873**	0.950715	0.821352
*gdb*4	11	19	4	**0.001766**	0.019672	0.149119	0.780746	0.737634	0.809841	**0.928217**	0.923841	0.782717
*gdb*5	13	26	6	**0.013742**	0.081535	0.191624	0.798773	0.869112	0.824886	**0.925012**	0.881022	0.665339
*gdb*6	12	22	5	0.022758	**0.010959**	0.166298	0.820776	0.844730	**0.777610**	0.927929	**0.930271**	0.744261
*gdb*7	12	22	5	**0.001905**	0.206082	0.259892	**0.727180**	0.787064	0.779443	**0.779109**	0.632002	0.532821
*gdb*8	27	46	10	**0.065808**	0.097087	0.214706	0.812013	0.859538	0.823990	**0.924160**	0.872941	0.724383
*gdb*9	27	51	10	**0.038974**	0.051164	0.274357	0.874343	0.857652	0.881908	**0.930479**	0.917804	0.633869
*gdb*10	12	25	4	**0.017936**	0.258262	0.351409	**0.709984**	0.879912	0.853631	**0.823408**	0.630947	0.465408
*gdb*11	22	45	5	**0.039002**	0.277287	0.315678	**0.785507**	0.918200	0.900487	**0.881007**	0.708175	0.512891
*gdb*12	13	23	7	**0.005164**	0.016110	0.104416	0.850628	0.832020	0.853378	**0.980243**	0.979106	0.809011
*gdb*13	10	28	6	0.170071	0.180372	0.273860	0.988458	1.000000	**0.874523**	0.865900	0.862069	0.707721
*gdb*14	7	21	5	**0.033982**	0.261274	0.352942	**0.769615**	0.863232	0.854531	**0.872280**	0.794213	0.589352
*gdb*15	7	21	4	**0.050147**	0.349475	0.227593	**0.695741**	0.869075	0.776009	**0.735185**	0.562500	0.556944
*gdb*16	8	28	5	**0.068613**	0.303918	0.331089	**0.731969**	0.890029	0.844825	**0.797619**	0.608333	0.467989
*gdb*17	8	28	5	**0.180030**	0.550350	0.380069	**0.700056**	0.931299	0.756089	**0.825556**	0.621111	0.550556
*gdb*18	9	36	5	**0.064065**	0.314809	0.339070	**0.705372**	0.875640	0.880833	**0.842605**	0.737725	0.533990
*gdb*19	8	11	3	0.000000	0.000000	0.217262	0.689926	0.689926	**0.673179**	0.714286	0.714286	0.504762
*gdb*20	11	22	4	**0.125009**	0.195592	0.182633	**0.768393**	0.823900	0.823426	**0.819643**	0.776587	0.735615
*gdb*21	11	33	6	**0.066627**	0.252814	0.237787	**0.777325**	0.914989	0.888137	**0.850412**	0.713228	0.634039
*gdb*22	11	44	8	**0.066028**	0.143205	0.243994	0.823458	0.911066	0.813657	**0.869808**	0.740383	0.606475
*gdb*23	11	55	10	**0.104860**	0.144992	0.386921	0.875470	0.881174	0.869281	**0.803514**	0.719563	0.389269

For each instance and each metric, the result that is significantly better than others is in boldface (with smallest I_D and Δ, while with greatest I_H).

TABLE III

AVERAGE VALUE OF I_D, Δ AND I_H OVER 30 INDEPENDENT RUNS OF THE COMPARED ALGORITHMS ON THE *val* SET

Problem	$\|V\|$	$\|E\|$	τ	I_D			Δ			I_H		
				D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II
*val*1A	24	39	2	**0.021581**	0.288084	0.277642	**0.826860**	1.000000	0.872144	**0.943678**	0.684483	0.598121
*val*1B	24	39	3	**0.044519**	0.115148	0.244812	**0.836487**	0.884870	0.870302	**0.872647**	0.830208	0.544473
*val*1C	24	39	8	0.026728	0.032653	0.814618	0.900184	0.896428	1.000000	0.958554	0.941799	0.117460
*val*2A	24	34	2	**0.015511**	0.206106	0.218118	**0.778396**	0.870051	0.879490	**0.794212**	0.664432	0.528108
*val*2B	24	34	3	**0.029323**	0.122290	0.204035	**0.795504**	0.878269	0.861804	**0.804240**	0.748594	0.563582
*val*2C	24	34	8	**0.075862**	0.125862	0.605525	0.996913	0.991727	**0.882948**	**0.925115**	0.875517	0.404713
*val*3A	24	35	2	**0.009113**	0.130534	0.244528	**0.829782**	0.933323	0.879654	**0.855250**	0.758879	0.535624
*val*3B	24	35	3	**0.003884**	0.009137	0.226411	0.880052	0.879665	**0.797203**	**0.901889**	0.894556	0.659611
*val*3C	24	35	7	**0.007143**	0.050000	0.726190	1.000000	0.979689	0.989825	**0.992857**	0.950000	0.273810
*val*4A	41	69	3	**0.036222**	0.097135	0.517681	**0.845928**	0.928517	0.877692	**0.881834**	0.793611	0.257703
*val*4B	41	69	4	**0.040741**	0.055040	0.640778	**0.829101**	0.861189	0.884425	**0.895275**	0.835651	0.179179
*val*4C	41	69	5	0.044510	**0.039666**	0.521099	**0.854632**	0.888505	0.883814	**0.908314**	0.900294	0.319008
*val*4D	41	69	9	**0.042473**	0.066038	0.650771	0.948836	0.954178	**0.861916**	**0.968994**	0.938014	0.378267
*val*5A	34	65	3	**0.043365**	0.211394	0.365025	**0.855040**	0.918426	0.917828	**0.859080**	0.701103	0.424836
*val*5B	34	65	4	**0.046929**	0.152943	0.499159	**0.823922**	0.900973	0.939342	**0.829466**	0.692898	0.222838
*val*5C	34	65	5	**0.053699**	0.155414	0.528122	**0.800515**	0.861334	0.884102	**0.830813**	0.702173	0.237407
*val*5D	34	65	9	**0.055831**	0.074474	0.465619	0.876871	0.911589	**0.844131**	**0.928692**	0.891527	0.376311
*val*6A	31	50	3	**0.030430**	0.212964	0.241787	**0.779257**	0.863983	0.839278	**0.849628**	0.678763	0.539604
*val*6B	31	50	4	**0.047001**	0.196831	0.312510	**0.789869**	0.906557	0.871675	**0.749397**	0.694004	0.424838
*val*6C	31	50	10	0.203895	0.235074	0.735624	**0.840530**	0.901210	0.887079	0.867320	0.855501	0.249619
*val*7A	40	66	3	**0.050913**	0.361388	0.292486	0.848629	0.871379	0.862394	**0.869234**	0.619957	0.520646
*val*7B	40	66	4	**0.053772**	0.311073	0.526678	**0.793655**	0.910238	0.875159	**0.785541**	0.560383	0.212282
*val*7C	40	66	9	**0.038781**	0.106679	0.295054	**0.807624**	0.877243	0.854016	**0.899756**	0.865095	0.480320
*val*8A	30	63	3	**0.032564**	0.197845	0.385505	**0.846041**	0.946727	0.924361	**0.877391**	0.749813	0.418221
*val*8B	30	63	4	**0.033601**	0.130713	0.426237	**0.814564**	0.893369	0.894872	**0.874730**	0.775106	0.357385
*val*8C	30	63	9	**0.074759**	0.117168	0.320418	0.836183	0.847926	0.832353	**0.922176**	0.822137	0.535190
*val*9A	50	92	3	**0.060197**	0.301938	0.536941	**0.822405**	0.935880	0.871752	**0.830741**	0.708264	0.257671
*val*9B	50	92	4	**0.069932**	0.254137	0.662066	**0.805901**	0.925571	0.909500	**0.773374**	0.653468	0.115218
*val*9C	50	92	5	**0.064638**	0.189373	0.411132	**0.816305**	0.906102	0.875979	**0.848982**	0.804664	0.358905
*val*9D	50	92	10	**0.061239**	0.083620	0.648318	0.835957	0.861696	0.831780	**0.912214**	0.838818	0.216695
*val*10A	50	97	3	**0.057796**	0.342043	0.506623	**0.838297**	0.955142	0.929302	**0.818545**	0.656889	0.318174
*val*10B	50	97	4	**0.058944**	0.274119	0.381459	**0.841877**	0.951383	0.942558	**0.862753**	0.791315	0.443111
*val*10C	50	97	5	**0.069181**	0.222700	0.652653	**0.837638**	0.932813	0.940012	**0.803281**	0.695455	0.145332
*val*10D	50	97	10	**0.093867**	0.153825	0.546695	**0.832806**	0.886949	0.862742	**0.812797**	0.740643	0.224720

For each instance and each metric, the result that is significantly better than others is in boldface (with smallest I_D and Δ, while with greatest I_H).

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE IV
AVERAGE VALUE OF I_D, Δ AND I_H OVER 30 INDEPENDENT RUNS OF THE COMPARED ALGORITHMS ON THE *egl* SET

Problem	$\|V\|$	$\|E\|$	$\|R\|$	τ	I_D			Δ			I_H		
					D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II	D-MAENS	LMOGA	NSGA-II
E1-A	77	98	51	5	0.036912	0.041494	0.267209	**0.837173**	0.871417	0.886003	0.961134	0.953894	0.673405
E1-B	77	98	51	7	**0.022972**	0.027591	0.393926	0.928263	0.901692	**0.811100**	**0.985737**	0.973730	0.620783
E1-C	77	98	51	10	0.103739	0.115848	0.783209	0.806003	0.810029	**0.710195**	0.842120	0.830386	0.154459
E2-A	77	98	72	7	**0.034070**	0.155534	0.228773	0.871803	0.852683	0.856005	**0.929793**	0.895206	0.624767
E2-B	77	98	72	10	**0.030468**	0.048951	0.319239	0.865474	0.832002	0.830119	**0.949172**	0.917306	0.605356
E2-C	77	98	72	14	**0.052499**	0.108517	0.394959	0.896010	0.860160	0.841612	**0.912012**	0.853518	0.547432
E3-A	77	98	87	8	**0.039940**	0.067949	0.295953	**0.861154**	0.890025	0.890814	**0.929949**	0.856558	0.532301
E3-B	77	98	87	12	**0.049111**	0.079399	0.385193	0.886609	**0.860360**	0.906476	**0.920454**	0.871371	0.544479
E3-C	77	98	87	17	**0.055030**	0.128920	0.754184	0.919013	0.897894	**0.861464**	**0.934317**	0.845319	0.234861
E4-A	77	98	98	9	**0.052780**	0.075407	0.361713	0.884950	0.894944	0.904382	**0.926844**	0.872198	0.461175
E4-B	77	98	98	14	**0.132789**	0.208926	0.403383	0.922241	0.906295	0.924598	**0.949293**	0.905957	0.649063
E4-C	77	98	98	19	**0.055220**	0.121632	0.736181	0.928553	0.951792	**0.845886**	**0.956615**	0.884639	0.314961
E1-A	140	190	75	7	**0.030514**	0.080384	0.177150	0.904305	0.921717	0.884422	**0.890462**	0.842490	0.649117
E1-B	140	190	75	10	**0.029941**	0.055984	0.214091	0.884088	0.903788	**0.851167**	**0.932369**	0.891414	0.707422
S1-C	140	190	75	14	**0.074585**	0.167701	0.552732	**0.822246**	0.859363	0.856784	**0.906602**	0.806034	0.333658
S2-A	140	190	147	14	**0.034959**	0.049498	0.353840	**0.869846**	0.907437	0.937896	**0.928025**	0.862601	0.486828
S2-B	140	190	147	20	**0.044781**	0.079291	0.374232	0.897434	0.919009	0.881360	**0.948473**	0.883102	0.582001
S2-C	140	190	147	27	**0.071934**	0.111792	0.673141	0.930810	0.924213	0.919789	**0.932086**	0.885749	0.288865
S3-A	140	190	159	15	**0.046492**	0.059549	0.328910	**0.859125**	0.910351	0.883791	**0.923804**	0.868949	0.516624
S3-B	140	190	159	22	**0.055405**	0.096119	0.479496	0.848864	0.883151	0.868471	**0.924604**	0.862733	0.450213
S3-C	140	190	159	29	**0.084777**	0.132210	0.699735	0.889979	0.921042	0.900136	**0.916414**	0.858534	0.257848
S4-A	140	190	190	19	**0.076616**	0.160206	0.420624	0.910406	0.942660	0.895271	**0.958476**	0.916048	0.517106
S4-B	140	190	190	27	**0.094041**	0.212546	0.831535	0.949837	0.930120	0.958830	**0.917572**	0.806978	0.171004
S4-C	140	190	190	35	**0.064972**	0.103996	0.842116	0.986036	0.973376	0.940114	**0.939239**	0.904580	0.163917

For each instance and each metric, the result that is significantly better than others is in boldface (with smallest I_D and Δ, while with greatest I_H).

TABLE V
NONDOMINATED SOLUTIONS WITH THE LEAST TOTAL COST OBTAINED BY D-MAENS AND THE BEST SOLUTIONS OBTAINED BY MAENS OVER 30 RUNS ON THE *gdb* SET

Problem	D-MAENS		MAENS	
	Total Cost	Makespan	Total Cost	Makespan
*gdb*1	**316**	**74**	316	76
*gdb*2	**339**	**69**	339	73
*gdb*3	275	65	275	65
*gdb*4	**287**	**74**	287	78
*gdb*5	**377**	**78**	377	79
*gdb*6	**298**	**75**	298	75
*gdb*7	**325**	**68**	325	73
*gdb*8	348	48	**348**	**44**
*gdb*9	304	50	**303**	**49**
*gdb*10	**275**	**70**	275	73
*gdb*11	**395**	**80**	395	90
*gdb*12	458	97	458	97
*gdb*13	**536**	**151**	536	154
*gdb*14	**100**	**21**	100	23
*gdb*15	**58**	**15**	58	17
*gdb*16	**127**	**26**	127	30
*gdb*17	**91**	**13**	91	22
*gdb*18	**164**	**33**	164	37
*gdb*19	**55**	**21**	55	22
*gdb*20	**121**	**36**	121	37
*gdb*21	**156**	**27**	156	30
*gdb*22	**200**	**26**	200	29
*gdb*23	**233**	**28**	233	36

For each instance, the solution dominating the other is in boldface.

of vehicles required subject to the capacity constraints. It was obtained by dividing the total demand of the tasks by the capacity of vehicles. 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 on the results obtained by 30 runs of the three compared algorithms, and the one that is significantly better than that of the other two (with the significance level of 5%) is in boldface.

TABLE VI
NONDOMINATED SOLUTIONS WITH THE LEAST TOTAL COST OBTAINED BY D-MAENS AND THE BEST SOLUTIONS OBTAINED BY MAENS OVER 30 RUNS ON THE *val* SET

Problem	D-MAENS		MAENS	
	Total Cost	Makespan	Total Cost	Makespan
*val*1A	**173**	**58**	173	74
*val*1B	**173**	**60**	173	62
*val*1C	245	41	245	41
*val*2A	**227**	**114**	227	115
*val*2B	259	108	259	108
*val*2C	457	71	457	71
*val*3A	81	41	81	41
*val*3B	87	32	87	32
*val*3C	138	27	138	27
*val*4A	**400**	**134**	400	142
*val*4B	**412**	**106**	412	116
*val*4C	430	100	428	115
*val*4D	536	82	530	85
*val*5A	**423**	**141**	423	143
*val*5B	**446**	**112**	446	115
*val*5C	**474**	**97**	474	100
*val*5D	595	79	584	88
*val*6A	**223**	**75**	223	77
*val*6B	233	68	233	68
*val*6C	317	54	317	54
*val*7A	**279**	**85**	279	91
*val*7B	**283**	**58**	283	68
*val*7C	334	50	334	50
*val*8A	**386**	**129**	386	133
*val*8B	**395**	**99**	395	105
*val*8C	532	83	526	90
*val*9A	324	109	323	110
*val*9B	**326**	**83**	326	86
*val*9C	**332**	**68**	332	69
*val*9D	392	51	391	55
*val*10A	**428**	**143**	428	178
*val*10B	436	117	**436**	**112**
*val*10C	**446**	**92**	446	94
*val*10D	533	61	530	68

For each instance, the solution dominating the other is in boldface.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE VII
NONDOMINATED SOLUTIONS WITH THE LEAST TOTAL COST OBTAINED BY D-MAENS AND THE BEST SOLUTIONS OBTAINED BY MAENS OVER 30 RUNS ON THE *egl* SET

Problem	D-MAENS		MAENS	
	Total Cost	Makespan	Total Cost	Makespan
E1-A	3548	943	3548	943
E1-B	4525	839	4498	899
E1-C	5595	836	5595	836
E2-A	5018	953	5018	953
E2-B	6347	871	**6321**	**870**
E2-C	8339	854	**8335**	**854**
E3-A	5926	942	**5898**	**929**
E3-B	7801	872	**7779**	**872**
E3-C	10 340	827	10 305	875
E4-A	6476	953	**6476**	**930**
E4-B	9069	926	**9016**	**914**
E4-C	11 774	822	11 628	872
E1-A	5068	1068	**5018**	**1023**
E1-B	6435	984	6394	1050
S1-C	8518	1018	8518	1018
S2-A	10 117	1087	9981	1109
S2-B	13 459	1040	**13 297**	**1040**
S2-C	16 832	1040	**16 552**	**1040**
S3-A	10 469	1099	**10 355**	**1099**
S3-B	14 082	1040	**13 877**	**1040**
S3-C	17 650	1040	17 362	1061
S4-A	12 602	1092	12 470	1108
S4-B	16 686	1027	16 528	1103
S4-C	21 213	1027	20 874	1067

For each instance, the solution dominating the other is in boldface.

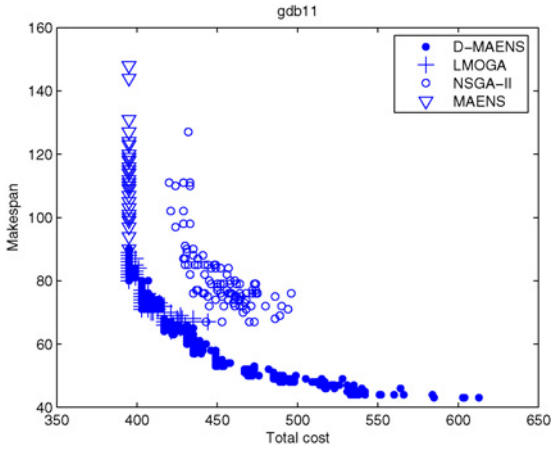


Fig. 3. Nondominated solutions obtained by all 30 runs of the compared algorithms for the *gdb*11 instance

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 better than the others on 71 out of the total 81 instances, including 18 out of the 23 *gdb* instances, 31 out of the 34 *val* instances, and 22 out of the 24 *egl* instances. LMOGA was significantly better on two instances, including one *gdb* instance and *val* instance. NSGA-II failed to outperform the other algorithms on any instance. Note that on *gdb*1 and *gdb*19, D-MAENS and LMOGA both reached the minimal value 0 of I_D. This shows that for these two instances, D-MAENS and LMOGA both consistently found all the best nondominated solutions in all 30 runs. D-MAENS also obtained the smallest average I_D on six other instances (*gdb*2, *gdb*13, *val*1C, *val*6C, E1-A, and E1-C), although the results were not statistically significant.

TABLE VIII
NUMBER OF NONDOMINATED SOLUTIONS ON THE TEST SETS AFTER COMBINING THE NONDOMINATED SETS OBTAINED BY 30 RUNS OF THE COMPARED ALGORITHMS TOGETHER

Problem	No.NS	Problem	No.NS	Problem	No.NS
*gdb*1	3	*val*1A	7	E1-A	9
*gdb*2	4	*val*1B	7	E1-B	3
*gdb*3	4	*val*1C	2	E1-C	2
*gdb*4	3	*val*2A	13	E2-A	7
*gdb*5	6	*val*2B	12	E2-B	7
*gdb*6	5	*val*2C	1	E2-C	5
*gdb*7	5	*val*3A	6	E3-A	16
*gdb*8	6	*val*3B	4	E3-B	5
*gdb*9	4	*val*3C	1	E3-C	4
*gdb*10	10	*val*4A	16	E4-A	12
*gdb*11	17	*val*4B	11	E4-B	6
*gdb*12	5	*val*4C	10	E4-C	2
*gdb*13	2	*val*4D	2	S1-A	19
*gdb*14	5	*val*5A	19	S1-B	11
*gdb*15	6	*val*5B	20	S1-C	7
*gdb*16	7	*val*5C	13	S2-A	27
*gdb*17	4	*val*5D	9	S2-B	9
*gdb*18	7	*val*6A	11	S2-C	6
*gdb*19	2	*val*6B	11	S3-A	16
*gdb*20	6	*val*6C	4	S3-B	8
*gdb*21	9	*val*7A	11	S3-C	7
*gdb*22	7	*val*7B	10	S4-A	8
*gdb*23	7	*val*7C	5	S4-B	1
		*val*8A	19	S4-C	1
		*val*8B	19		
		*val*8C	8		
		*val*9A	17		
		*val*9B	17		
		*val*9C	16		
		*val*9D	8		
		*val*10A	23		
		*val*10B	15		
		*val*10C	19		
		*val*10D	13		

"Problem" indicates the test instance, and "No.NS" stands for the number of nondominated solutions.

As for the Δ metric, it can be observed that D-MAENS performed significantly better than the others on ten *gdb* instances, 25 *val* instances, and five *egl* instances. LMOGA performed the best on one *egl* instance, and NSGA-II performed the best on three *gdb* instances, four *val* instances, and five *egl* instances. There are ten *gdb* instances, five *val* instances, and 13 *egl* instances on which none of the algorithms showed significantly better performance than the other two. Among them, D-MAENS provided the smallest Δ on four *gdb* instances, one *val* instance, and three *egl* instances. LMOGA obtained the smallest Δ on four *gdb* instances, two *val* instances, and three *egl* instances. NSGA-II got the smallest Δ on three *gdb* instances, two *val* instances, and seven *egl* instances. Note that on *gdb*1 and *gdb*19, although D-MAENS and LMOGA consistently found all the best nondominated solutions in the 30 runs, their Δ's were either statistically comparable with or significantly worse than that of NSGA-II. This showed that the best nondominated solutions were not uniformly distributed due to the discreteness of the solution space.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

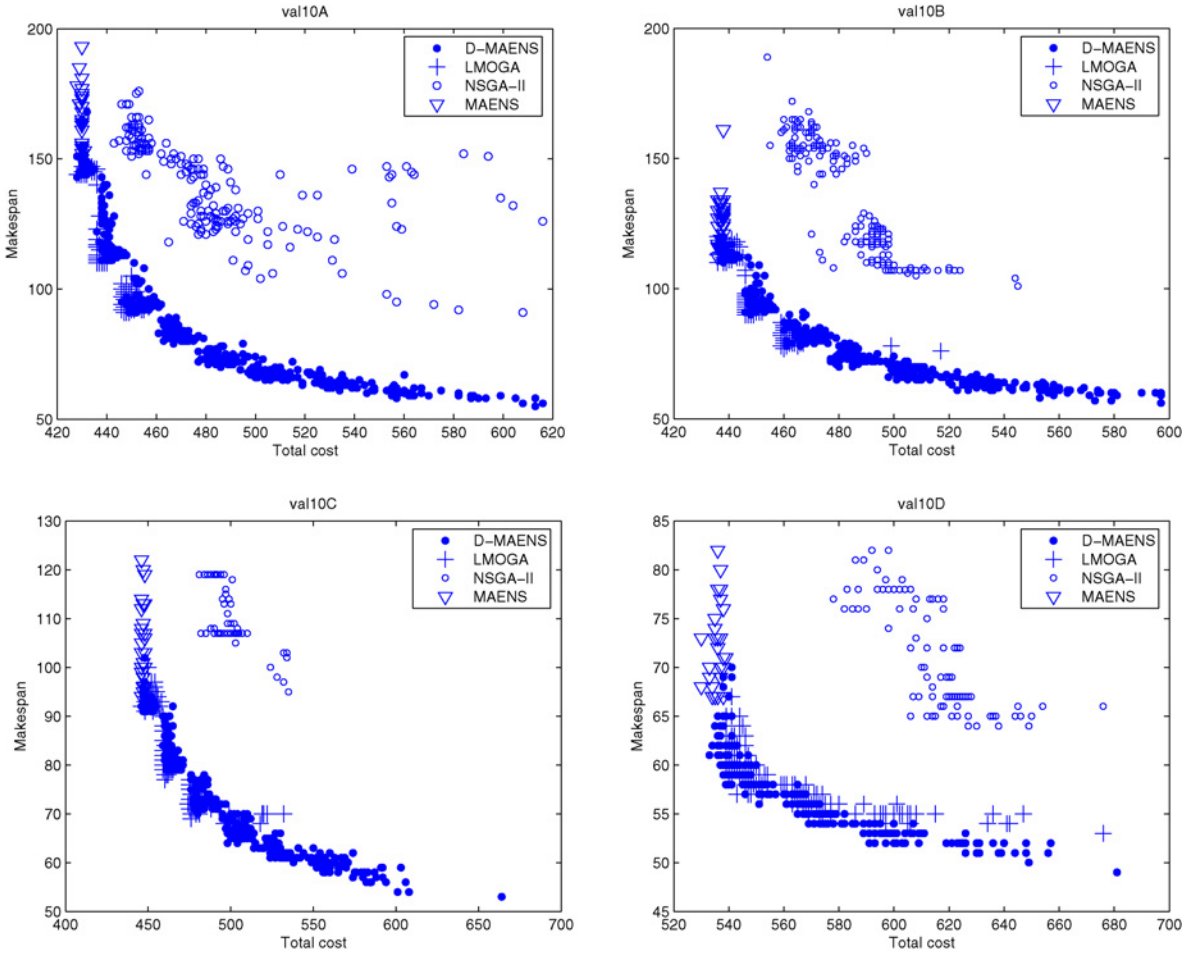


Fig. 4. Nondominated solutions obtained by all 30 runs of the compared algorithms for the *val*10 test instance set

Finally, the algorithms are compared in terms of I_H. It is observed that D-MAENS achieved significantly larger I_H than LMOGA and NSGA-II on 18 instances, 32 *val* instances and 22 *egl* instances. LMOGA was superior on one *gdb* instance, and NSGA-II failed to be the best on any instance.

To check whether a nondominated solution can be obtained by applying MAENS to the original SO-CARP instances, we compared D-MAENS with MAENS as well. To make a fair comparison, the parameters of MAENS were set in such a way that the computational time of the two algorithms were comparable. MAENS was also run 30 times on the same computer as other compared algorithms.

Tables V–VII present the nondominated solutions with the lowest total cost obtained by D-MAENS and the best solutions obtained by MAENS over 30 runs on the three test sets. For each instance, the solution dominating the other is in boldface. From the tables, it can be observed that D-MAENS performed better than MAENS on the *gdb* and *val* sets, while was outperformed by MAENS on the *egl* set. D-MAENS obtained solutions dominating those obtained by MAENS on 19 *gdb* instances, 17 *val* instances, and 0 *egl* instances. The opposite case occurred on two *gdb* instances, one *val* instance, and 11 *egl* instances. Taking a closer look at the results, we found that D-MAENS obtained solutions with the best known total cost on 22 out of the 23 *gdb* instances and 27 out of the 34 *val* instances, while only succeeded on 3 out of the 24 *egl* instances. The reason is that D-MAENS uniformly allocates computational resources to all subproblems to search for the whole Pareto front, while MAENS solely focuses on one problem whose size is equivalent to that of a subproblem in D-MAENS. For the *egl* instances, the computational resources assigned to each subproblem might not be sufficient for D-MAENS. On the other hand, the computational resources were sufficient for relatively simple test instances such as the *gdb* and *val* instances. Hence, D-MAENS was able to obtain good solutions in terms of both the total cost and the makespan.

In order to comprehensively evaluate the performance of the compared algorithms, the nondominated solutions obtained by them on selected instances are also plotted in the objective space. It is more important to visualize the Pareto front for the instances with more Pareto optimal solutions than those with less Pareto optimal solutions. In practice, the number of Pareto optimal solutions of an instance is rarely known. Thus, for each instance, all the solutions obtained throughout the experiment were first combined together. Then, the nondominated solutions were identified. The instances with the most nondominated solutions were chosen for illustration.

Table VIII presents the number of nondominated solutions on the test sets after the combination. It can be seen that the instances *gdb*11, *val*10A–*val*10D and S2-A–S2-C had the most nondominated solutions in their corresponding test sets.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

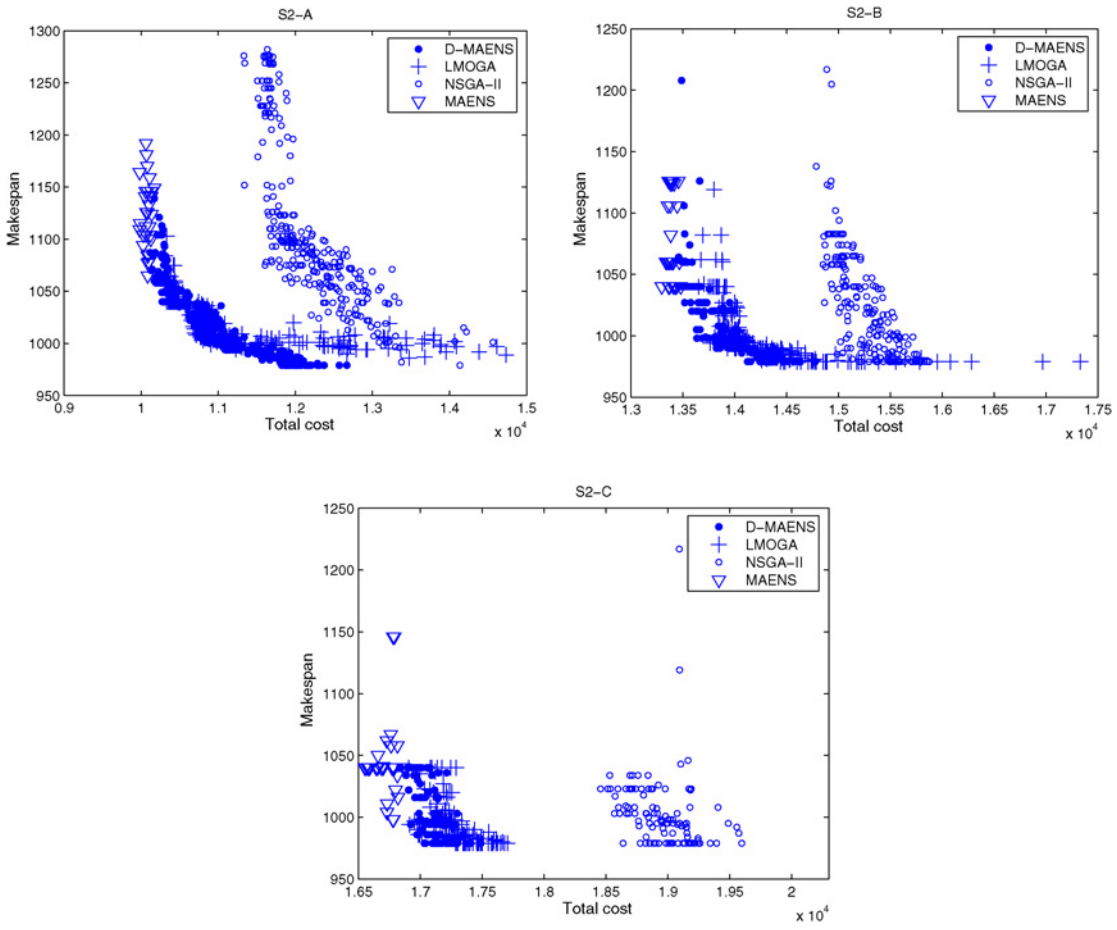


Fig. 5. Nondominated solutions obtained by all 30 runs of the compared algorithms for the *egl* S2 test instance set

Therefore, they are selected as the representative instances to be plotted.

Figs. 3–5 show the nondominated solutions obtained on the selected instances by D-MAENS, LMOGA, NSGA-II, and MAENS in all 30 runs. First, we compare the MOEAs. For *gdb*11, D-MAENS and LMOGA converged better than NSGA-II. Besides, D-MAENS covered the objective space more completely than LMOGA. It reached both the areas with low total cost and low makespan. LMOGA, on the other hand, was confined to the area with low total cost and relatively high makespan. In the intermediate area (that with the value of makespan from 65 to 90), D-MAENS still performed no worse than LMOGA. Similar scenarios can be observed on the four *val* instances. NSGA-II performed the worst. D-MAENS still had a stronger capability of reaching the area with lower makespan than LMOGA, although it was outperformed by LMOGA in the intermediate area on *val*10A∼*val*10C. For the *egl* instances, all the three MOEAs managed to find solutions with low makespan. Meanwhile, D-MAENS obtained solutions with the lowest total cost. Further, it is not surprising to find that MAENS obtained solutions with lower total cost and higher makespan than those obtained by the MOEAs.

The relationship between the above observations and the quantitative results in Tables II-IV deserves more discussion. As mentioned in their definitions, I_D and I_H reflect the extent of the convergence to the true Pareto front. I_D also reflects the distribution of the nondominated solutions if the reference solutions are uniformly distributed, while I_H also reflects how well the nondominated solutions cover the Pareto front. Δ mainly implies the distribution of the nondominated solutions. From Figs. 3–5, it is evident that most reference solutions used in computing I_D, especially those with low makespan, were obtained by D-MAENS. This explains why D-MAENS got smaller I_D's and larger I_H's. The value of Δ depends on two factors: 1) the location of the leftmost and the rightmost nondominated solutions; and 2) the distribution of the nondominated solutions. The advantage of D-MAENS was not so clear with respect to Δ. The figures demonstrated that D-MAENS performed the best while NSGA-II performed the worst with regard to the first factor. Hence, the results on Δ can only be due to the distribution of the solutions, although it is difficult to tell how well the solutions were distributed from the figures.

TABLE IX
AVERAGE COMPUTATIONAL TIME (IN SECONDS) USED BY THE COMPARED ALGORITHMS

Test set	D-MAENS	LMOGA	NSGA-II
gdb	8.82	1.65	9.66
val	53.98	11.82	78.02
egl	213.50	50.24	95.77
overall	88.42	20.32	63.87

Table IX gives the average runtime of the compared algorithms on each test set. The row headed "overall" shows the

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

average runtime on the total 81 instances. It can be seen that D-MAENS was the most time-consuming among the three algorithms. LMOGA was the fastest algorithm, whose average runtime was much less than that of the others. For each test set, the average runtime of D-MAENS was about 4-5 times as that of LMOGA. Note that D-MAENS and LMOGA both employ local search, and their local search rates were set to the same value in our experiments. However, D-MAENS spent more time than LMOGA. This is due to the high computational cost of the Merge-Split operator employed in the local search phase of MAENS, which also made MAENS much more time-consuming than other approaches for SO-CARP [22]. The average runtime of NSGA-II was comparable to that of D-MAENS on the *gdb* and *val* sets, but D-MAENS was much more time-consuming on the *egl* set. This is because the computational time of the local search of D-MAENS increases with the problem size. With the same parameter settings given in Table I, D-MAENS is computationally more expensive on the *egl* set than NSGA-II. However, when we ran NSGA-II on the *egl* set with more generations, no significant improvement on solution quality was obtained.

VI. Conclusion

In this paper, we investigated a MO-CARP that considers minimizing the total cost and the makespan as two objectives. By revisiting the existing EMO strategies in the context of MO-CARP, a decomposition-based framework for solving MO-CARP was proposed. Integrating a competitive algorithm for SO-CARP into this framework, a novel algorithm called D-MAENS has been developed. Experimental studies on three well-known benchmark sets demonstrated the advantages of D-MAENS over LMOGA and NSGA-II. This verified the efficacy of combining conventional EMO techniques with domain-specific search algorithms.

In comparison with SO-CARP that only requires minimizing the total cost, the MO-CARP investigated in this paper goes one step closer toward reality. However, in most real-world applications such as winter gritting, many other factors need to be considered, e.g., the time window constraints, the intermediate facilities and the time-dependent service costs. Therefore, our future work will focus on incorporating these factors into the CARP model.

References

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

[2] H. Handa, D. Lin, L. Chapman, and X. Yao, "Robust solution of salting route optimization using evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 16–21, 2006, pp. 3098–3105.

[3] H. Handa, L. Chapman, and X. Yao, "Robust route optimization for gritting/salting trucks: A CERCIA experience," *IEEE Comput. Intell. Mag.*, vol. 1, no. 1, pp. 6–9, Feb. 2006.

[4] M. Tagmouti, M. Gendreau, and J. Potvin, "Arc routing problems with time-dependent service costs," *Eur. J. Oper. Res.*, vol. 181, no. 1, pp. 30–39, 2007.

[5] G. Ghiani, F. Guerriero, G. Laporte, and R. Musmanno, "Tabu search heuristics for the arc routing problem with intermediate facilities under capacity and length restrictions," *J. Math. Model. Algorithms*, vol. 3, no. 3, pp. 209–223, 2004.

[6] A. Amberg, W. Domschke, and S. Voß, "Multiple center capacitated arc routing problems: A tabu search algorithm using capacitated trees," *Eur. J. Oper. Res.*, vol. 124, no. 2, pp. 360–376, 2000.

[7] 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, 2006.

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

[9] J. Campbell and A. Langevin, "Roadway snow and ice control," in *Arc Routing: Theory, Solutions, and Applications*. Boston, MA: Kluwer, 2000, pp. 389–418.

[10] M. Polacek, K. Doerner, R. 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, 2008.

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

[12] 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.

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

[14] 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, 2000.

[15] 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, 2001.

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

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

[18] P. Lacomme, C. Prins, and W. Ramdane-Chérif, "Competitive memetic algorithms for arc routing problem," *Ann. Oper. Res.*, vol. 131, nos. 1–4, pp. 159–185, 2004.

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

[20] A. Corberán, C. Prins, and C. Council, "Recent results on arc routing problems: An annotated bibliography," *Networks*, vol. 56, no. 1, pp. 50–69, 2010.

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

[22] 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.

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

[24] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[25] V. Pareto, *Cours D'Economic Politique*, vols. I and II. Lausanne, Switzerland: F. Rouge, 1896.

[26] E. Zitzler, M. Laumanns, and S. Bleuler, "A tutorial on evolutionary multiobjective optimization," *Lecture Notes in Economics and Mathematical Systems*, vol. 535. Berlin, Germany: Springer, 2004, pp. 3–37.

[27] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proc. 1st Int. Conf. Genet. Algorithms*, Jul. 24–26, 1985, pp. 93–100.

[28] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," in *Proc. Evol. Methods Design Optim. Control Applicat. Ind. Problems*, Sep. 19–21, 2002, pp. 95–100.

[29] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

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

[31] M. Ehrgott and X. Gandibleux, "Approximative solution methods for multiobjective combinatorial optimization," *TOP: J. Spanish Statist. Oper. Res. Soc.*, vol. 12, no. 1, pp. 1–63, 2004.

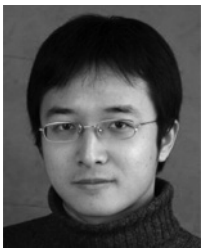
[32] H. Ishibuchi and T. Murata, "Multiobjective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. Syst.,*

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Man Cybernet. C: Applicat. Rev., vol. 28, no. 3, pp. 392–403, Aug. 1998.
[33] A. Jaszkiewicz, "Genetic local search for multiobjective combinatorial optimization," *Eur. J. Oper. Res.*, vol. 137, no. 1, pp. 50–71, 2002.
[34] A. Jaszkiewicz, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem: A comparative experiment," *IEEE Trans. Evol. Comput.*, vol. 6, no. 4, pp. 402–412, Aug. 2002.
[35] K. Tan, C. Cheong, and C. Goh, "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation," *Eur. J. Oper. Res.*, vol. 177, no. 2, pp. 813–839, 2007.
[36] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, Oct. 2006.
[37] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR Spectrum*, vol. 22, no. 4, pp. 425–460, 2000.
[38] 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.
[39] J. S. DeArmon, "A comparison of heuristics for the capacitated Chinese postman problem," Masters thesis, Univ. Maryland, College Park, 1981.
[40] E. Benavent, V. Campos, A. Corberán, and E. Mota, "The capacitated arc routing problem: Lower bounds," *Networks*, vol. 22, no. 7, pp. 669–690, 1992.
[41] R. W. Eglese, "Routing winter gritting vehicles," *Discrete Appl. Math.*, vol. 48, no. 3, pp. 231–244, 1994.
[42] R. W. Eglese and L. Y. O. Li, "A tabu search based heuristic for arc routing with a capacity constraint and time deadline," in *Metaheuristics: Theory and Applications*, I. H. Osman and J. P. Kelly, Eds. Boston: Kluwer, 1996, pp. 633–650.
[43] L. Y. O. Li and R. W. Eglese, "An interactive algorithm for vehicle routing for winter-gritting," *J. Oper. Res. Soc.*, vol. 47, no. 2, pp. 217–228, 1996.
[44] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.
[45] P. Czyzzak and A. Jaszkiewicz, "Pareto simulated annealing: A metaheuristic technique for multiple-objective combinatorial optimization," *J. Multi-Criteria Decision Anal.*, vol. 7, no. 1, pp. 34–47, 1998.
[46] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," Comput. Eng. Netw. Laboratory (TIK), ETH Zurich, Zurich, Switzerland, TIK Rep. 286, Nov. 2008.

Yi Mei (S'09) received the Bachelors degree in mathematics from the University of Science and Technology of China (USTC), Hefei, China, in 2005. He is currently pursuing the Ph.D. degree from the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, USTC.

His current research interests include evolutionary computation, memetic algorithms, meta-heuristics, multiobjective optimization, and robust design optimization for arc routing problems 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.

Since 2007, he has been an Associate Professor with the Nature Inspired Computation and Applications Laboratory, School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. He is the co-author of more than 40 refereed publications. His major research interests include machine learning, pattern analysis, evolutionary computation, data mining, meta-heuristic algorithms, and real-world applications.

Dr. Tang is an associate editor or editorial board member of four international journals and the Chair of the IEEE Task Force on Large Scale Global Optimization.

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

From 1985 to 1990, while working toward the Ph.D. degree, he was an Associate Lecturer and Lecturer at USTC. In 1990, He was a Post-Doctoral Fellow with the Computer Sciences Laboratory, Australian National University, Canberra, A.C.T., Australia, and continued his work on simulated annealing and evolutionary algorithms. In 1991, he joined the Knowledge-Based Systems Group, Commonwealth Scientific and Industrial Research Organization Division of Building, Construction and Engineering, Melbourne, Vic., Australia, where he worked primarily on an industrial project on automatic inspection of sewage pipes. He returned to Canberra in 1992 to take up a lectureship in the School of Computer Science, University College, University of New South Wales, Australian Defense Force Academy, Canberra, Australia, where he was later promoted to a Senior Lecturer and Associate Professor. Attracted by the English weather, he moved to the University of Birmingham, Birmingham, U.K., as Chair of the School of Computer Science in 1999. Currently, he is the Director of the Center of Excellence for Research in Computational Intelligence and Applications, and a Distinguished Visiting Professor (Grand Master Chair Professor) with USTC. He has given more than 60 invited keynote and plenary speeches at conferences and workshops worldwide. He has more than 300 refereed publications in these areas. His current research interests include evolutionary computation, neural network ensembles and real-world applications.

Dr. Yao is the former (2003–2008) Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, an associate editor or editorial board member of ten other journals, and the Editor of the *World Scientific Book Series on Advances in Natural Computation*. He was awarded 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 won the 2001 IEEE Donald G. Fink Prize Paper Award for his work on evolutionary artificial neural networks.