A Hybrid Estimation of Distribution Algorithm with Decomposition for Solving the Multiobjective Multiple Traveling Salesman Problem

V. A. Shim, K. C. Tan, and C. Y. Cheong

***Abstract*—Evolutionary multiobjective optimization with decomposition, in which the algorithm is not required to differentiate between the dominated and nondominated solutions, is one of the promising approaches in dealing with multiple conflicting objectives. In this paper, the estimation of distribution algorithm (EDA) is integrated into the decomposition framework. The search behavior of the algorithm is further enhanced by hybridizing local search metaheuristic approaches with the decomposition EDA. Three local search techniques, including hill climbing, simulated annealing, and evolutionary gradient search, are considered. A novel multiobjective formulation of the multiple traveling salesman problem is proposed. The hybrid algorithms are used to solve the formulated problem with different number of objective functions, salesmen, and problem sizes. The effectiveness and efficiency of the algorithms are tested and benchmarked against several state-of-the-art multiobjective evolutionary paradigms.**

***Index Terms*—Estimation of distribution algorithms (EDAs), evolutionary gradient search (EGS), evolutionary multiobjective optimization, hill climbing (HC), multiple traveling salesman problem (mTSP), simulated annealing (SA) , univariate modeling (UM).**

I. Introduction

THE multiple traveling salesman problem (mTSP), where multiple salesmen are involved in the routing in order to achieve a common goal, is a generalization of the classical TSP. In the mTSP, m salesmen are instructed to visit n cities ($m < n$), whereby all the salesmen will start from and end at the single depot (may be multiple depots) after visiting the ordered cities. Each city can only be visited once, and the total cost for all salesmen is required to be minimized. The cost can be defined as distance, time, expense, risk, etc. When $m = 1$, the problem simplifies to the classical TSP. The mTSP is more complex than the TSP since it is required to allot a set of cities to each salesman in an optimal ordering, while minimizing the total cost for all salesmen. However, the mTSP is more appropriate for real-life problems where more than one salesman is usually involved. The problem is closely related to the school bus routing problem [1], design of global navigation satellite system [2], interview scheduling [3], hot rolling scheduling [4], mission planning [5], etc. Over the past few decades, research on the TSP has attracted a great deal of attention. However, the mTSP has not received the same amount of research effort compared with the TSP. Due to the high complexity of the mTSP (NP-hard problem [6]), exact algorithms are unsuitable to solve the problem even for a moderate number of cities. Even though heuristic approaches [7]–[9] are unable to guarantee optimal solutions, they are still able to obtain approximate optimal solutions within specific time or computational constraints.

In addition, many real-life scheduling problems also involve several conflicting objectives that have to be simultaneously optimized. In the evolutionary multiobjective framework [10], no single point is an optimal solution. Instead, the optimal solution is a set of nondominated solutions, which represents the tradeoff between the multiple objectives. In this case, fitness assignment to each solution in the evolutionary framework is considered to be an important feature for the assurance of the survival of fitter and less crowded solutions to the next generation. Much research has been carried out over the past few decades to address this issue, and fitness assignment based on domination is one of the most popular approaches [11]. However, this fitness assignment approach is less efficient in solving many objective problems. This is because the strength of the domination is weakened when there are many objectives, which in turn results in poor decision making in the selection of promising solutions. Recently, the classical approach for multiobjective optimization based on aggregation has been reformularized into a population-based approach [12], [13], whereby a set of nondominated solutions is obtained from a single simulation run. In the decomposition approach, it is not required to differentiate the domination behavior of the solutions. Instead, the solutions are aggregated according to any classical aggregation approach.

Estimation of distribution algorithm (EDA) [14]–[17] is a computing paradigm in the field of evolutionary computation. This algorithm estimates the overall distribution of the parent solutions in the decision space so as to predict the distribution of the unknown solutions. In EDA, a probabilistic model is constructed from the parent solutions by any modeling paradigm including statistical methods [18], probability mechanisms [19], [20], or machine learning approaches [21]–[23]. Subsequently, offspring is generated by sampling according to the constructed model.

Manuscript received January 12, 2011; revised June 10, 2011 and January 18, 2012; accepted January 27, 2012. Date of publication April 3, 2012; date of current version August 15, 2012. This paper was recommended by Associate Editor M.-H. Lim.

V. A. Shim and K. C. Tan are with the Department of Electrical and Computer Engineering, National University of Singapore, 117576 Singapore (e-mail: g0800438@nus.edu.sg; eletankc@nus.edu.sg).

C. Y. Cheong is with the Computing Science Department, Institute of High Performance Computing, 138632 Singapore (e-mail: cheongcy@ihpc.a-star.edu.sg).

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/TSMCC.2012.2188285

This paper studies the hybridization of an evolutionary optimizer with several local search techniques in a multiobjective decomposition framework to deal with the multiobjective mTSP (MmTSP). EDA that is based on univariate modeling (UM) [14], [19] is used in this paper due to its simplicity, efficiency, and convenience of not requiring any internal parameter tuning. First, EDA in a decomposition framework is being developed. The performance of the algorithm is, then, enhanced by hybridizing EDA with local search. Three local search metaheuristics, including multipoint hill climbing (HC), multipoint simulated annealing (SA), and multipoint evolutionary gradient search (EGS), are explored in this paper. A new formulation of the objective functions for the mTSP is proposed and extended to the multiobjective framework. The proposed algorithms are, then, used to solve the formulated problem and simulation studies are carried out on various instances of the problems with different number of objective functions, salesmen, and cities. The proposed algorithms are, then, rigorously compared with several state-of-the-art evolutionary multiobjective optimizers.

The remaining parts of this paper are organized as follows. The following section presents a literature review on the application of EC in mTSP, as well as a brief introduction of EDA and the local search metaheuristics that will be studied in this paper. Section III describes the problem formulation of the MmTSP, while the proposed algorithm is highlighted in Section IV. Section V presents the experimental results and discussions. Conclusions are drawn in Section VI.

II. Background Information

In this section, a literature review, focusing on the application of evolutionary approaches to mTSP, is provided. See [24] for a more rigorous review of the works that involved nonevolutionary techniques, which is out of scope of this paper. A brief introduction on EDAs and local search metaheuristics will also be presented.

A. Multiple Traveling Salesman Problems

The first implementation of genetic algorithm (GA) to solve the mTSP was carried out in [25]. Zhang *et al.* used a simple GA with natural representation to schedule the multiple teams of photographers to visit a large number of elementary and secondary schools. The objective is to minimize both the total distance traveled and the time consumed, such that the time constraints are satisfied and each team must be able to visit at least two schools daily. Unfortunately, the authors did not elaborate on how they manipulated multiple teams in their chromosome representation.

Another application of the mTSP, involving a hot rolling scheduling problem in Shanghai Baoshan Iron and Steel Complex, was studied by Tang *et al.* [4]. The problem considers actual production constraints and aims to schedule multiple turns within the same shift. The authors first modeled the hot rolling scheduling problem as an mTSP. Next, the mTSP was converted into a classical TSP through the proposal of a one-chromosome representation. The selection operation was modified such that the best solutions obtained so far were always selected to be one of the parent chromosomes to undergo crossover operation.

In [26], Park studied the vehicle scheduling problem using GA. Since there are multiple vehicles involve in the routing, the problem can, basically, be classified as an mTSP. In the paper, a two-chromosome representation was proposed. The first chromosome locates the cities, while the second chromosome indicates which vehicle is to be assigned to visit the city specified in the first chromosome. In [27], a two-part chromosome representation was proposed for the mTSP. Under this scheme, the chromosome for a gene is divided into two distinct parts. The first part of the chromosome allots the permutation of the cities, while the second part of the chromosome determines the number of cities to be visited by each salesman. This means that there will be an additional of m genes in the chromosome for m salesmen. In this two-part chromosome representation, the solution space is also smaller than the two-chromosome representation. The results also demonstrated that the proposed representation is able to produce better results than the one-chromosome representation under most of the test instances.

Zhao *et al.* [28] proposed a GA, which utilized the one-chromosome representation, to solve the mTSP. The algorithm employed a pheromone-based crossover operator that utilized information of edge lengths, adjacency relations, and pheromone levels to construct new solutions. Several local search strategies (relocation, exchange, and 2-opt) were also used to facilitate the search. The grouping GA was also used by Singh and Baghel [6] who proposed a replacement policy to reduce problem redundancy. In their work, two different objective functions were considered—minimizing total distance traveled by all salesmen, and minimizing the maximum distance traveled by any salesman.

Recently, multichromosome representation of mTSP solutions has been proposed in [29] where the route assigned to each salesman was represented in a chromosome. Therefore, each solution has m associated chromosomes. The crossover and mutation operators designed to deal with the representation were also proposed.

B. Estimation of Distribution Algorithms

EDAs are a class of evolutionary computing techniques. EDAs imitate all properties of GA including survival of the fittest, elitism, and generation of new solutions from parent solutions. The only difference is that the reproduction process in GA is based on biological recombination, while that in EDA is carried out by constructing a probabilistic model from the parent solutions, and the offspring is produced by sampling the corresponding model [14], [15], [30]. One of the simplest, yet effective, EDAs is based on UM [19]. The pseudocode of UM can be found in Fig. 1. First, S initial solutions are randomly generated with a marginal probability of 0.5. After evaluating the solutions, $S1$, where $S1 \leq S$, promising candidate solutions are selected using any selection mechanism. A probabilistic model is, then, constructed from the selected solutions. Considering the binary case, UM constructs the probability distribution of the solutions by calculating the frequency of existence of all

Begin

1. Initialization: Generate S solutions with a marginal probability of 0.5. Store the solutions in Pop.

Do while ("Stopping criterion is not met")

2. Evaluation: Get the fitness $F(x)$ of all solutions in Pop.
3. Selection: Select $S1 \leq S$ solutions from Pop using any selection strategy.
4. Probabilistic model: Estimate the probability distribution of the selected solutions. In binary representation, the marginal probability distribution of x_i is constructed as follows.

$$p_g(x_i) = \sum_{j=1}^{S1} \delta_j(x_i)/S1$$
$$\delta_j(x_i) = \begin{cases} 1 \; if \; x_i = 1 \\ 0 \; otherwise \end{cases} \tag{1}$$

In other words, the delta function is the value of variable x_i in the j^{th} individual.

5. Sampling: Sample S offspring from $p_g(x)$ using simple sampling technique as follows.

$$x_i = \begin{cases} 1 \; if \; random\,(0,1) \leq p_g(x_i) \\ 0 \; otherwise \end{cases} \tag{2}$$

End Do

End

Fig. 1. Pseudocode of EDA based on UM.

cardinalities in the genes according to (1). Offspring is created by sampling the constructed model according to (2). The same procedure is repeated until the stopping criterion is met.

C. Local Search Metaheuristics

Local search, which has been proven to be able to improve the performance of global search, especially EC [31]–[38], is used to exploit the local optimal in a specific region. In this paper, three general metaheuristics with multiple searches are studied and their brief descriptions are as follows (consider the minimization case).

1) Hill Climbing: The basic procedure behind HC [39] lies in the random perturbation of a solution so as to search for a local neighbor which is fitter than the original one. First, K solutions (K directional searches) are identified to undergo local search. For each selected solution, a bit string of the solution is flipped to produce a new solution (binary case). The fitness of the generated solution is evaluated and, then, compared with the original solution. The generated solution replaces the original solution as the current solution if it is fitter than the original one. This procedure of generating a new solution, comparing it with the current one, and replacing the current solution if it is better, is performed L times, where L is the predetermined number of local neighbors. The whole procedure described earlier is repeated until the stopping criterion is met. The pseudocode of the multipoint HC can be found in Fig. 2.

2) Simulated Annealing: SA [40] is similar to HC except that SA allows the acceptance of inferior solutions. A probability that is based on Boltzmann criterion is used to determine the chance of survival of inferior solutions. The basic principle is inspired from the cooling process in metallurgy. In this process, a metal is heated to a high temperature, which melts

Begin

Do while ("Stopping criterion is not met")

For $j = 1$: K(Number of solutions undergoing local search)

1. Initial solution: Select a solution (x_o^j) from selection pool. Set current solution $x^j = x_o^j$ and evaluate x^j.

For $i = 1$: L(Number of local neighbors)

2. Reproduction: Create local neighbor (y^i) by flipping a single bit of x_o^j.
3. Evaluation: Calculate the fitness value of y^i, $F(y^i)$.
4. Update solution: if $F(y^i) < F(x^j)$ then $x^j = y^i$

End for i

5. Output: Output x^j with the best fitness.

End for j

End do

End

Fig. 2. Pseudocode for multipoint HC algorithm.

Begin

Do while ("Stopping criterion is not met")

1. Input parameter: Initialize temperature (Te).

For $j = 1$: K(Number of solutions undergoing local search)

2. Initial solution: Select a solution (x_o^j) from selection pool. Set current solution $x^j = x_o^j$ and evaluate x^j.

For $i = 1$: L (Number of local neighbors)

3. Reproduction: Create local neighbor (y^i) by flipping a single bit of x_o^j.
4. Evaluation: Calculate the fitness value of y^i, $F(y^i)$.
5. Update solution:
   if $F(y^i) < F(x^j)$
   then $x^j = y^i$
   else if $random(0,1) < exp(\frac{F(x^j)-F(y^i)}{Te})$
   then $x^j = y^i$

End for i

6. Output: Output x^j.

End for j

$Te = \alpha \times Te$; $0 < \alpha < 1$

End do

End

Fig. 3. Pseudocode for multipoint SA algorithm.

the metal, leading to a random rearrangement of the positions of the particles in the metal. The metal is, subsequently, cooled until thermal equilibrium of the solid metal is reached. The pseudocode of the multipoint SA can be found in Fig. 3. Initial temperature *Te* and annealing schedule α should be predetermined before the process begins. The difference between SA and HC is in step 5 of the algorithm whereby the Boltzmann criterion $random(0,1) < \exp(F(x^j) - (y^i)/Te)$ gives a change of accepting inferior solutions. *random*(0,1) is a randomly generated floating number between 0 and 1. *Te* is initially set to a high value and it is gradually reduced according to the factor of α where $\alpha \in (0,1)$. With such an acceptance criterion, there would be a higher chance of inferior solutions surviving during early iterations and this probability is reduced as the search progresses. The acceptance of inferior solutions in SA may allow the search to escape from local optimal regions.

3) Evolutionary Gradient Search: Gradient search is one of the classical continuous optimization approaches. The direction, in terms of gradient, is captured and is used to guide the search. Every simulation run will generate a single solution.

Begin
1. Input: Define initial step size σ_0. $t = 1$.
Do while ("Stopping criterion is not met")
For $j = 1$: K(Number of solutions undergoing local search)
2. Initial solution: Select a solution x^j from selection pool.
3. Reproduction: Create L local neighbors y^i, $i \in (1,2,\ldots,L)$ by perturbing x^j using normal mutation $N(0,\sigma_t^2)$.
4. Evaluation: Calculate the fitness value of y^i, $F(y^i)$.
5. Direction: Estimate the global gradient direction as follows.

$$\hat{v} = \frac{\sum_{i=1}^{L}[F(y^i)-F(x^j)](y^i-x^j)}{\left\|\sum_{i=1}^{L}[F(y^i)-F(x^j)](y^i-x^j)\right\|} \tag{3}$$

6. Create offspring:

$$g = x^j - \sigma_t \hat{v}$$

7. Update mutation step size σ_{t+1}:

$$\sigma_{t+1} = \begin{cases} \sigma_t \varepsilon \ if\ F(g) < F(x^j) \\ \sigma_t/\varepsilon \quad otherwise \end{cases}, \varepsilon = 1.8$$

8. Update solution: if $F(g) < F(x^j)$ then $x^j = g$
9. Output: Output x^j.
End for j
$t = t + 1$.
End do
End

Fig. 4. Pseudocode for multipoint EGS algorithm.

Recently, this approach has been adapted into evolutionary mechanism through the introduction of population-based and survival of the fittest concepts into the algorithm so as to produce EGS [41]–[43]. In this approach, multidirectional searches are carried out. The gradient is the direction calculated from the evolutionary movement instead of the single movement of a solution. The pseudocode of the multipoint EGS can be found in Fig. 4. First, initial step size σ_0 is predetermined. Step size σ is used to govern the degree of mutation applied to generate local neighbors and offspring. After selecting an individual as the initial solution, L local neighbors are generated by perturbing the initial solution using mutation with normal distribution of zero mean and σ^2 variance. The global gradient direction is, subsequently, estimated from the local neighbors according to (3). It is to be noted that the gradient offspring is generated during step 6, and a factor ε is used to control the mutation step size as shown in step 7. The solution is updated in step 8 and the process is repeated until the stopping criterion is met.

III. Problem Formulation

In the literature, the aim of the mTSP is specified to be either minimizing the total traveling cost of all salesmen or the highest traveling cost incurred by any single salesman [44]. In this paper, the focus is tailored, specifically, for the mTSP with single depot, considering the minimization of the total traveling cost and the balancing of the workload among all salesmen. This is achieved by formularizing the objective function to be the weighted sum of the total traveling cost of all salesmen and the highest traveling cost by any single salesman. In the context of multiobjective optimization, more than one objective is subject to be minimized, which can be formulated as follows.

Fig. 5. One-chromosome representation.

Minimize:

$$F(x) = (F_1(x), \ldots, F_p(x))$$
$$F_1(x) = w_1 * \mathrm{TC}^1(x) + w_2 * \mathrm{MC}^1(x)$$
$$\vdots$$
$$F_P(x) = w_1 * \mathrm{TC}^P(x) + w_2 * \mathrm{MC}^P(x)$$

where

$$\mathrm{TC}^k(x) = \sum_{j=1}^{m} \mathrm{IC}_j^k(x)$$
$$\mathrm{MC}^k(x) = \max_{1\le j\le m} (\mathrm{IC}_j^k(x))$$
$$\mathrm{IC}_j^k(x) = \sum_{i=1}^{n_{j-1}} D_j^k(a_{i,j}, a_{i+1,j}) + D_j^k(a_{n_{j,j}}, a_{1,j}).$$

In the aforementioned formulation, $x \in \phi$, ϕ is the decision space, a_i, j is the ith visiting city by salesman j, P is the number of objective functions, w_1 and w_2 are the weights to balance between total cost and highest cost ($w_1 + w_1 = 1.0$), TC is the total traveling cost of all salesmen, MC is the highest traveling cost of any single salesman, IC is the individual traveling cost, m is the number of salesmen, n_j is the number of cities traveled by salesman j, and $D_j^k(a_{i,j}, a_{i+1,j})$ is the traveling cost (for the kth objective function) between cities at locations i and $i + 1$ for salesman j. The constraint in the problem is that all the cities must be visited exactly once and each salesman must be assigned at least one city in his traveling route.

IV. Algorithms

The proposed algorithm, consisting of four main mechanisms (chromosome representation, decomposition, modeling, and local search metaheuristics), is presented in this section. The proposed decomposition framework is a modified version of the multiobjective evolutionary algorithm with decomposition (MOEAD) [13]. In this implementation, one-chromosome representation [4] is utilized to represent the order of the cities to be traveled by m salesmen. This scheme introduces $m - 1$ pseudocities (integer values ≤ 1) to the chromosome. These pseudocities represent the same initial city where all the salesmen will start their routes. Therefore, each chromosome may consist of $n + m - 1$ genes.

The representation with nine cities and three salesmen is illustrated in Fig. 5. The sequence of travel is as follows. The first salesman starts from the initial city 1 then visits cities 5, 7, and 9 in that order. The second salesman again starts from the initial city then visits cities 4 and 3 in that order. The third salesman visits cities 2, 6, and 8 in that order.

In the decomposition framework, the fitness assigned to each solution can be based on any classical aggregation approach.

For the implementation, the Tchebycheff approach [45] is used and the decomposition framework is described according to this approach. A set of evenly distributed weight vectors $\lambda^1, \ldots, \lambda^S$ and the reference point z^* are generated, where S is the number of subproblems. The algorithm decomposes the population into S scalar optimization subproblems according to the Tchebycheff formulation and the fitness value of the jth subproblem is defined as

$$gt(x|\lambda^j, z^*) = \max_{1\le i\le P}\{\lambda_i^j |F_i(x) - z_i^*|\}. \tag{4}$$

The pseudocode of the proposed algorithm can be found in Fig. 6. In step 1, the Q neighbors that are nearest, in terms of Euclidean distance, to each weight vector are determined. Therefore, subproblem i has Q neighbors denoted as $B(i) = \{i_1, \ldots, i_Q\}$. Then, initial chromosomes in the form as indicated in Fig. 5 are, randomly, generated. Objective values $F_1(x), \ldots, F_p(x)$ are calculated based on the formulation in Section III, and the reference point z^* is set to the minimum objective value of the initial population.

Step 2 performs the probabilistic modeling according to univariate marginal distribution by calculating the probability of existence of each city in each permutation location in the chromosome. In the model, a $N \times N$ probabilistic matrix $\mathrm{Pr}_g(x_i)$, where $N = n + m - 1$, is constructed according to (5). The offspring is, subsequently, generated by sampling the constructed probabilistic model according to (6). Since the sampling is carried out marginally, the existing cities in a chromosome are not taken into consideration by the sampling mechanism. Therefore, some cities may appear more than once, while some cities may not even be included in a chromosome. The chromosome is repaired according to the heuristic approach proposed in [46] to ensure that there is no repetition of any city. In this approach, those repeated and unallocated cities are determined. The unallocated city is inserted to the location of the repeated city if the unallocated city has the smallest traveling cost (e.g., distance, times, charge, risk, etc.) to the adjacent cities in the location of the repeated city. For example, let us assume that a salesman is instructed to visit five cities in an order of 1, 3, 2, 3, 5. In this case, city 3 is repeated, while city 4 is not included in the order. The traveling costs of visiting cities in an order of 1, 4, 2 and 2, 4, 5 are calculated. If it is in the condition that the traveling cost of visiting cities in the order of 1, 4, 2 is smaller than the cost of visiting cities in the order of 2, 4, 5, then the former order is applied. Thus, the final sequence of travel is 1, 4, 2, 3, 5.

In the event that some salesmen are not being assigned any city in their routes, there will be a penalty added to the objective values of the solutions by multiplying the original values with a constant value $\mathcal{L}$. This is done to weaken the solutions that do not satisfy all the constraints. In this implementation, $\mathcal{L} = 10$ is applied. Step 3 updates z^* followed by FV. For z^*, it is the reference point that is used in the Tchebycheff approach and is updated by taking the minimum value of the objective functions. It is emphasized that step 3(b) is one of the important features in decomposition framework whereby the fitness of the solutions is assigned according to (4). All solutions sampled from the probabilistic model will also be updated one by one to all neighboring solutions, and the superior solutions will replace the inferior ones.

Local search is performed if it is activated and it is applied every generation thereafter. The same procedure is carried out until the stopping criterion is met. Three local search metaheuristics are considered. Since the evolutionary paradigm consists of multiple solutions in a population, multipoint searches are used. This increases the exploration in the search space, which will promote the diversity of the solutions. For HC and SA, each subproblem will undergo local search by generating L local neighbors, which are created by simply swapping two genes in a chromosome. For each local solution, step 3 is carried

Step 1: Initialization:

a) Compute the Euclidean distance between all the weight vectors and then group the Q closest weight vectors $B(i) = \{i_1, .., i_Q\}$, $i \in [1, S]$, to each weight vector.

b) Randomly generate the initial population $x^1, \ldots, x^s$ in integer number from $[2 - m, n]$. No integer number is repeated. Set $FV^i = F(x^i)$.

c) Initialize $z^* = (z_1^*, \ldots, z_P^*)$ by setting z^* according to the minimum objective value of the initial population.

Step 2: Reproduction based on EDA:

a) Construct the probabilistic model $Pr_g(x)$ by computing the marginal probability of each city $(c_1, .., c_N)$, where $N = n + m - 1$, in each permutation location as follows.

$$Pr_g(x) = \begin{bmatrix} Pr_g(x_1 = c_1) & \ldots & Pr_g(x_N = c_1) \\ \vdots & \ddots & \vdots \\ Pr_g(x_1 = c_N) & \ldots & Pr_g(x_N = c_N) \end{bmatrix}$$

$$Pr_g(x_i = c_j) = \frac{\sum_{k=1}^{S} \delta_k(x_i = c_j) + 1/N}{S + S/N} \tag{5}$$

$$\delta_k(x_i = c_j) = \begin{cases} 1 \ \ if\ x_i = c_j \\ 0 \ \ otherwise \end{cases}$$

where $Pr_g(x_i)$ is the probability distribution of the cities at generation g, $Pr_g(x_i = c_j)$ is the probability of city j to be located at the i^{th} position of the chromosome, and c_j is the city j $(c_1 = 2 - m, \ldots, c_N = n)$.

b) Sample $Pr_g(x)$ to generate S offspring as follows.

$$y_i = \begin{cases} C_1 \ if\ random\ (0,1) \le Pr_g(x_i = c_1) \\ C_2 \ if\ Pr_g(x_i = c_1) < random\ (0,1) \le \sum_{j=1}^{2} Pr_g(x_i = c_j) \\ \vdots \\ C_N \ if\ \sum_{j=1}^{N-1} Pr_g(x_i = c_j) < random\ (0,1) \le \sum_{j=1}^{N} Pr_g(x_i = c_j) \end{cases} \tag{6}$$

where y_i is a newly generated city at i^{th} position of a chromosome.

c) Improvement: Apply specific heuristic approach to repair the chromosomes to ensure that the constraints are satisfied. Penalize the solution if any salesman is not assigned any city by multiplying the objective value with a constant $\mathcal{L}$, $F(y) = \mathcal{L} \times F(y)$.

Step 3: Update solution:

For $i = 1, \ldots, S$, do

a) Update of z^*: For $j = 1, \ldots, P$, if $z_j^* > F_j(y^i)$, then set $z_j^* = F_j(y^i)$

b) Update of neighboring solutions: For $j \in B(i)$, if $gt(y^i|\lambda^j, z^*) \ge gt(x^j|\lambda^j, z^*)$, then set $x^j = y^i$ and $FV^j = F(y^i)$.

End do

Step 4: Local search:

a) Perform local search if local search is activated. Next, apply step 3 to update the solutions.

Step 5: Stopping criterion: If stopping criterion is met, then stop. Else, go to Step 2.

Fig. 6. Pseudocode of the proposed hybrid EDA with decomposition.

out to update the z^* and *FV*. It is also to be noted that the standard HC and SA can be implemented directly to the algorithm. HC and SA are performed according to the pseudocodes shown in Figs. 2 and 3, respectively.

The pseudocode of standard multipoint EGS can be found in Fig. 4. For EGS, the adaptation is not as direct as HC and SA since the gradient information in permutation arrangement cannot be directly accessed. However, EGS is, naturally, suitable to be implemented in the decomposition framework because the aggregation function, which is required to determine the gradient, is available from the decomposition multiobjective evolutionary algorithm (MOEA). Some modifications are required to adapt the EGS for the permutation-based problem. First and foremost, the neighboring solutions are generated in a manner similar to that in HC and SA. The fitness of the solutions is aggregated according to the weighted sum approach by using available $\lambda^1, \ldots, \lambda^S$ values. This is different from the previous implementation [41]–[43] where the weights are, randomly, generated and all solutions share a common weight value. The global gradient direction is estimated according to step 5 in Fig. 4. In step 6, we have x^j as a floating value. However, in MmTSP, x^j is the city which may not be suitable to create an offspring. As such, the average cost between the local neighbors to the original chromosome is calculated and, then, assigned to x^j. Following which, g is updated. However, g is a floating value calculated from the gradient information of the solutions and it cannot be used to represent a city. Thus, we have to determine the candidate city to be the one that has the closest g value to the original city. For example, let $g = 100$, city 1 as the original city, and a salesman is instructed to visit two other cities (2 and 3). Then, calculate the traveling cost between original city (city 1) and cities 2 and 3. If the traveling cost between cities 1 and 2 is 200 and the traveling cost between cities 1 and 3 is 300, then city 2 is the candidate city because the traveling cost (200) is closer to the g value (100). The mutation step size is updated according to step 7 in Fig. 4, and the gradient solution is updated to the population according to step 3 in Fig. 6. A shared mutation step size σ_t is used to generate the gradient offspring and is adaptively tuned based on the quality of the estimated gradient. $\varepsilon = 1.8$ is used as proposed in the previous work [42], [43].

V. Experimental Studies and Discussions

All the algorithms in this study were implemented in C++.

The experimental settings are shown in Table I. MmTSP with two and five objectives are studied. An $n \times n$ cost matrix is randomly generated for each objective in the range of [0,1000] [46]–[48]. For experimental studies, the results are compared based on the performance metrics of inverted generational distance (IGD) [49] and Pareto front. IGD measures the average Euclidean distance between each solution on the Pareto optimal front to the nearest solution on an evolved front. A smaller IGD implies better proximity and spread. Since the optimal solution set to the problem is unknown, the estimated optimal front is formed using the nondominated solutions found from all algorithms and all simulation runs.

TABLE I
Parameter Settings for Experiments

Population size or number of subproblem, S	Number of cities, n [44]
Number of cities, n	100, 300, 500
Number of salesmen, m	2,5,10 for 100 cities; 2,5,10,30 for 300 cities; 2,5,10,20,50 for 500 cities
Number of objective functions, P	2 and 5
Q	Population size, S
Number of local neighbors, L	10
Computing budget in fitness evalutions	200 000 for 100 cities; 600 000 for 300 cities; 1 000 000 for 500 cities
Crossover and mutation rates in NSGAII and MOEAD	0.8 and 0.005 [44]
Initial temperature (Te) and scheduling factor (α) in SA	100 and 0.99
Initial step size (σ_0) and ε in EGS	300 and 1.8 [41]
Local search activation (in terms of number of fitness evaluations)	After 100,000 for 100 cities; 300,000 for 300 cities; 500,000 for 500 cities

Seven algorithms are put to comparison. UMEGS is the hybrid algorithm between decomposition EDA and EGS. UMSA is the hybrid algorithm between decomposition EDA and SA, and UMHC hybridizes EDA and HC. UMDAD is a pure decomposition EDA using UM. MOEAD is a pure decomposition GA proposed in [13] and used in [47]. UMGA is a synthesizing algorithm between EDA and nondominated sorting genetic algorithm-II (NSGA-II) proposed in [46]. Finally, NSGA-II is one of the most popular MOEAs based on the concept of domination proposed in [11] and used in [46]. The results that are presented have been averaged over ten independent runs with different random number seeds. The test instances consist of 24 MmTSP with different number of objective functions (P), salesmen (m), and problem size (n). The problem is denoted in the form of P2m5n100, which refers to an MmTSP with two objective functions, five salesmen, and 100 cities. Since the aims of this study are to obtain the smallest traveling cost and balancing the workload among the salesmen, the results, in terms of total traveling cost of all salesmen and highest traveling cost of any single salesman, are presented.

A. Effects of Weight Setting on Optimization Performance

The formulation of the MmTSP in this paper takes into account the weighted sum of total traveling cost of all salesmen and the highest traveling cost of any single salesman. The weight setting is dependent on the preference of the manager whether he wants to achieve the lowest total traveling cost of all salesmen or he wants to achieve the balancing of workload of all salesmen. If the aim is to obtain the lowest total traveling cost, the weights will be set to $w_1 = 1.0$ and $w_2 = 0.0$. On the other hand, if the final objective is to balance the workload of all salesmen, the weights will, then, be set to $w_1 = 0.0$ and $w_2 = 1.0$. However, if the aim is to achieve tradeoff between the two aims, then different weight settings should be employed.

Simulations were carried out to study the performance of UMDAD under various weight settings. Fig. 7 shows the IGD metric for (a) total traveling cost of all salesmen and (b) highest traveling cost of any single salesman under various weight

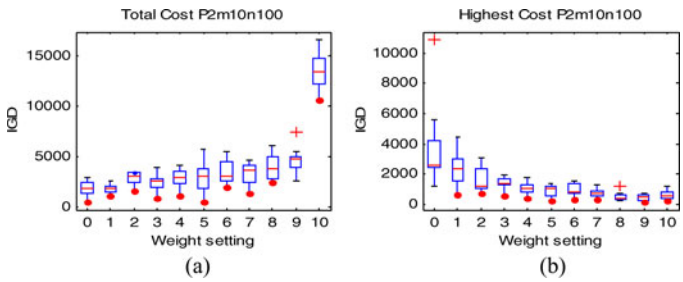


Fig. 7. IGD metric for (a) total traveling cost of all salesmen and (b) highest traveling cost of any single salesman under various weight settings for the MmTSP with two objective functions, 10 salesmen, and 100 cities (P2m10n100).

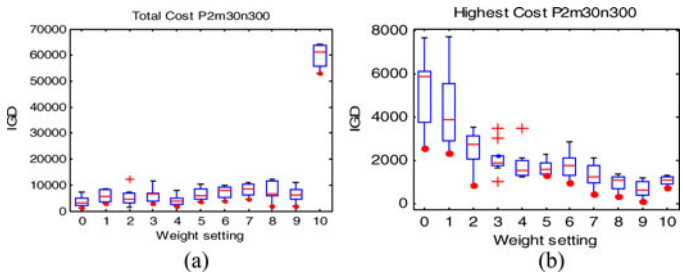


Fig. 8. IGD metric for (a) total traveling cost of all salesmen and (b) highest traveling cost of any single salesman under various weight settings for the MmTSP with two objective functions, 30 salesmen, and 300 cities (P2m30n300).

TABLE II
INDICES OF DIFFERENT WEIGHT SETTINGS

Index	0	1	2	3	4	5	6	7	8	9	10
W1	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1	0.0
W2	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

settings for the MmTSP with two objective functions, 10 salesmen, and 100 cities (P2m10n100). Fig. 8 shows the IGD metric for the MmTSP with two objective functions, 30 salesmen, and 300 cities. The indices of the weight settings are illustrated in Table II.

From Figs. 7(a) and 8(a), it is observed that the algorithm is able to produce solutions with minimum traveling cost of all salesmen under the weight setting of $w_1 = 1.0$ and $w_2 = 0.0$. This is expected since the weight setting deems solutions with smaller total traveling cost as superior. However, Figs. 7(b) and 8(b) show that the weight setting causes imbalance workload for the salesmen. On the other hand, if the focus is to minimize the highest traveling cost of any single salesmen (index 10), it is then observed that the highest cost is far smaller than that of index 0, but this leads to the solutions having the highest total traveling cost. This observation suggests that there is a tradeoff between both aims. Striking a balance between both aims can be achieved by setting the weights to intermediate values, which will lead to the ability to produce routes with smaller total cost of all salesmen and highest cost of any single salesman. For the rest of the experimental studies, the weight setting $w_1 = 0.5$, $w_2 = 0.5$ is used.

TABLE III
IGD METRIC FOR TOTAL TRAVELING COST FOR ALL SALESMEN OF SOLUTIONS OBTAINED BY VARIOUS ALGORITHMS FOR THE MmTSP WITH TWO OBJECTIVES, m SALESMEN, AND n CITIES

	UM EGS	UM SA	UM HC	UMDAD	MOEAD	UMGA	NSGAII
P2m2	**3236**	9316	8847	12820	9402	7627	7521
n100	**±765**	±1230	±1340	±240	±1149	±986	±678
P2m5	**4334**	9449	8682	13064	10214	8895	8049
n100	**±802**	±528	±891	±460	±725	±774	±610
P2m10	9996	10091	**9768**	13234	10729	10729	9228
n100	±929	±796	**±905**	±545	±953	±763	±692
P2m2	**5818**	35133	36030	42398	39448	35216	35837
n300	**±1732**	±1729	±2525	±737	±1318	±1891	±1913
P2m5	**8931**	36614	36747	42830	40278	37322	37038
n300	**±2218**	±2042	±2080	±591	±1021	±1586	±2197
P2m10	**33883**	36235	36942	43189	40423	41038	37025
n300	**±6294**	±1829	±1736	±959	±8689	±1562	±1327
P2m30	41055	**39057**	41040	45728	43346	48740	43945
n300	±2027	**±1470**	±1923	±702	±1610	±1055	±1901
P2m2	**11222**	63807	63547	68795	69515	63406	65982
n500	**±3291**	±1588	±2137	±1008	±1175	±1978	±1786
P2m5	**16426**	64796	64672	69206	69111	64399	67243
n500	**±5531**	±1491	±2396	±1883	±1733	±1719	±19482
P2m10	**33923**	64677	64060	69093	69791	71181	67623
n500	**±16241**	±1482	±1758	±1335	±1598	±2859	±1591
P2m20	64977	**64388**	64553	69196	71185	74967	71566
n500	±1999	**±1465**	±2184	±783	±1625	±2533	±1079
P2m50	69891	**68490**	69165	74446	79054	89038	84874
n500	±1409	**±1348**	±1800	±1910	±1526	±2053	±1312

B. Results for Two Objective Functions

Simulations were carried out to study the performance of the seven algorithms applied on the MmTSP with different number of objective functions, salesmen, and cities. Fig. 9(a) shows the evolved Pareto front of total traveling cost generated by three of the algorithms applied to the MmTSP with two objective functions, five salesmen, and 100 cities. The remaining algorithms show similar distributions as compared with MOEAD and NSGA-II and their Pareto fronts are left out of the plot for ease of visualization. It is observed that UMEGS is able to produce a set of diverse solutions but they are inferior in terms of proximity.

Similar observations can also be made for the MmTSP with two objective functions, five salesmen, and 300 cities [see Fig. 9(b)]. The Pareto front for the MmTSP with 20 salesmen and 500 cities is shown in Fig. 9(c). From Fig. 9(c), it is observed that all decomposition algorithms achieve better Pareto front than the algorithms using the concept of domination (UMGA and NSGA-II). For all decomposition algorithms, hybrid algorithms are, generally, able to generate better solutions than pure global population-based algorithms (UMDAD and MOEAD). Furthermore, UMDAD is able to perform better than MOEAD. From this observation, it can be concluded that decomposition algorithms scale well with the increase in the number of decision variables compared with the algorithms using the concept of domination. EDA uses global distribution of the parent solutions to guide the search process and is shown to have good proximity results, but poor solution diversity. Introducing local information into the evolutionary process, which helps the algorithm to further explore and exploit the search space, rectifies this limitation of EDA. Overall, UMEGS shows superior performance compared with the other algorithms.

For ease of visualization, the optimization results (mean $\pm$ standard deviation) for the different problem settings are

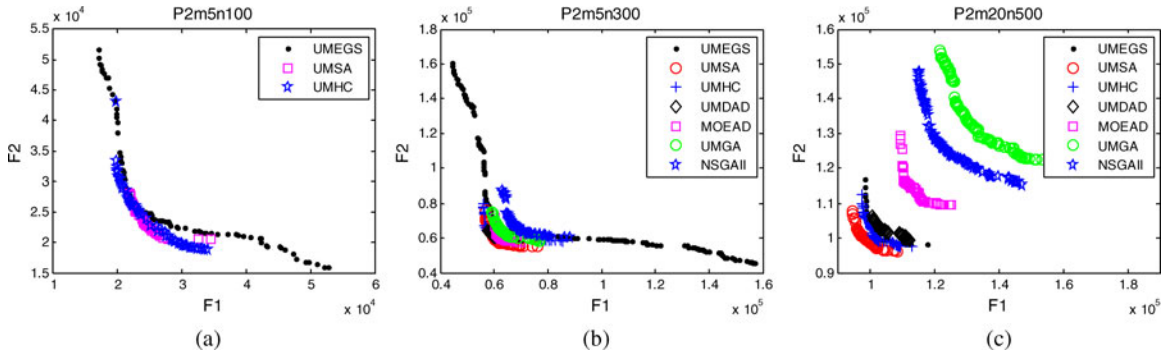


Fig. 9. Evolved Pareto front of total traveling cost generated by the various algorithms applied to the MmTSP with (a) two objective functions, five salesmen, and 100 cities, (b) two objective functions, five salesmen, and 300 cities (c) two objective functions, 20 salesmen, and 500 cities.

TABLE IV
IGD METRIC FOR HIGHEST TRAVELING COST OF ANY SINGLE SALESMAN OF SOLUTIONS OBTAINED BY VARIOUS ALGORITHMS FOR THE MmTSP WITH TWO OBJECTIVES, m SALESMEN, AND n CITIES

	UM EGS	UM SA	UM HC	UMDAD	MOEAD	UMGA	NSGAII
P2m2	**1659**	5116	4960	7091	5114	4307	4190
n100	**±545**	±653	±909	±231	±669	±607	±523
P2m5	**1604**	1942	1806	2861	1976	209	1898
n100	**±554**	±490	±493	±513	±407	9±448	±581
P2m10	997	989	968	1203	**522**	1160	1396
n100	±412	±276	±428	±521	**±385**	±382	±323
P2m2	**3417**	15797	16330	19531	18163	16076	16816
n300	**±1104**	±1013	±1252	±624	±536	±1368	±964
P2m5	**3423**	8716	9068	11083	9754	9102	8523
n300	**±1824**	±722	±899	±359	±444	±567	±920
P2m10	1989	1680	1927	2433	**1027**	3778	3098
n300	±790	±471	±1152	±848	**±663**	±1140	±975
P2m30	852	780	825	975	**627**	2562	2104
n300	±294	±313	±304	±301	**±433**	±509	±499
P2m2	**5567**	39621	39249	42267	41859	37211	37675
n500	**±2039**	±1286	±1533	±1099	±789	±1884	±1751
P2m5	**6185**	15976	15635	17222	15857	16291	17005
n500	**±2521**	±1088	±9402	±9969	±8588	±7313	±1219
P2m10	**3754**	3053	2589	3863	2860	6595	6485
n500	**±1492**	±1031	±1129	±1309	±1127	±1263	±1499
P2m20	1597±	**1184**	1574	1788	1721	4491	3464
n500	1018	**±656**	±995	±936	±1015	±610	±907
P2m50	871	**694**	873	1089	1290	3101	3432
n500	±380	**±477**	±370	±387	±531	±567	±808

presented in table form. In each table, the best result (mean) for each problem setting is bolded. Table III presents the total traveling cost for all salesmen of solutions generated by the seven algorithms for the MmTSP with two objective functions, m salesmen, and n cities. The results show that hybrid algorithms generate the best solutions. UMEGS achieved the best performance in most of the settings, followed by UMSA and UMHC. UMSA, which gives inferior solutions a chance for survival, performed better than UMHC for most of the problem settings. On the other hand, UMEGS, which uses gradient information for local exploitation, was able to search for more diverse solutions and, thus, performed better than the other algorithms. From the table, it is also observed that the total traveling cost increases with the increase in the number of salesmen. This is because when more salesmen are involved, the task gets more difficult since the algorithms need to determine the route for each salesman while maintaining the minimum total traveling cost at the same time. Since all salesmen need to return to the home city and the final assigned city could be far from the depot, additional traveling cost may be incurred. For UMEGS, the gradient information weakens with the increase in the number of salesmen, resulting in the algorithm not being able to exploit the information as effectively. Hence, it is unable to achieve superior results for problems with many salesmen.

In Table IV, the highest traveling cost of any single salesman of solutions generated by the algorithms for the MmTSP with two objective functions, m salesmen, and n cities are presented. Again, hybrid algorithms are able to achieve better performance than the other algorithms. Most of the best solutions are generated by UMEGS. Although MOEAD did not achieve good results in terms of total traveling cost of all salesmen, it is able to achieve the lowest traveling cost for any single salesman for problems with 10 and 30 salesmen in 100 and 300 cities. UMDAD performed better for problems with the largest number of cities (500), while giving inferior results for problems with smaller number of cities (100).

C. Results for Five Objective Functions

In this section, the results of the simulations on the MmTSP with five objective functions are presented. Table V shows the IGD metric for the total traveling cost of all salesmen of solutions obtained by the algorithms for the MmTSP with different number of salesmen and cities. The corresponding IGD metric for the highest traveling cost of any single salesman of solutions is presented in Table VI. Generally, the performances of algorithms using the decomposition framework (UMEGS, UMSA, UMHC, UMDAD, and MOEAD) are superior to those of the algorithms based on the concept of domination (UMGA and NSGA-II) in all problem settings. The superiority of decomposition algorithms is attributed to the aggregation principle used for fitness assignment. The tournament could be carried out by simply comparing the fitness values of solutions. Solutions with higher fitness values will always be selected to survive and reproduce. On the other hand, the concept of domination requires that fitness be assigned to each solution based on their rank of domination. In many objective problems, most of the solutions are nondominated and are given lower ranks. This may prevent the tournament process from selecting promising solutions to survive. Thus, NSGA-II and UMGA performed poorly compared with decomposition algorithms.

Overall, hybrid algorithms are able to produce better solutions than UMDAD and MOEAD. UMEGS achieved good

TABLE V
IGD METRIC FOR TOTAL TRAVELING COST FOR ALL SALESMEN OF SOLUTIONS OBTAINED BY VARIOUS ALGORITHMS FOR THE MmTSP WITH FIVE OBJECTIVES, m SALESMEN, AND n CITIES

	UM EGS	UM SA	UM HC	UMDAD	MOEAD	UMGA	NSGAII
P5m2	**7768**	8163	8169	13502	9510	16094	15760
n100	**±901**	±394	±471	±578	±363	±602	±931
P5m5	9714	9817	**9576**	13870	11820	19477	19257
n100	±1277	±621	**±727**	±374	±838	±923	±946
P5m10	13485	**13136**	13501	15595	15509	25370	24946
n100	±1068	**±627**	±964	±624	±992	±1447	±1265
P5m2	**15944**	34408	33846	55764	44806	56279	53107
n300	**±1647**	±1522	±1621	±1806	±1083	±1748	±1283
P5m5	**18707**	34866	34137	53830	46928	57302	55428
n300	**±2457**	±1043	±1197	±1496	±1914	±2732	±3197
P5m10	35311	35815	**34929**	53425	51187	61313	57994
n300	±1505	±1695	**±823**	±1268	±1450	±3170	±2308
P5m30	47350	**44889**	47137	57206	69850	92126	89524
n300	±1684	**±1580**	±1770	±8737	±3592	±2587	±2995
P5m2	**21022**	54384	53770	84832	93659	97628	92716
n500	**±3792**	±1534	±1393	±2336	±3677	±2484	±2651
P5m5	**26397**	54384	53279	83492	94109	97481	94088
n500	**±3916**	±1594	±1236	±1839	±3567	±3087	±3045
P5m10	**53377**	56191	54891	81343	99396	103870	96638
n500	**±4308**	±1455	±2005	±935	±2578	±3433	±3795
P5m20	57506	57772	**56241**	81606	104300	107900	98870
n500	±1682	±1319	**±1778**	±1148	±1523	±3062	±5070
P5m50	76294	**70526**	75938	90186	138890	141090	138310
n500	±2164	**±1828**	±2070	±1434	±4302	±4156	±4454

TABLE VI
IGD METRIC FOR HIGHEST TRAVELING COST OF ANY SINGLE SALESMAN OF SOLUTION OBTAINED BY VARIOUS ALGORITHMS FOR THE MmTSP WITH FIVE OBJECTIVES, m SALESMEN, AND n CITIES

	UM EGS	UM SA	UM HC	UMDAD	MOEAD	UMGA	NSGAII
P5m2	4752	**4273**	4393	6867	5099	9487	9444
n100	±1017	**±190**	±567	±394	±419	±591	±433
P5m5	3513	3589	2855	3463	**2692**	7378	5783
n100	±1350	±1464	±964	±586	**±358**	±1284	±1157
P5m10	1910	1788	1871	2080	**1685**	4777	4624
n100	±405	±521	±381	±321	**±472**	±1003	±752
P5m2	**9121**	17094	16695	26562	25645	31129	30284
n300	**±1003**	±599	±775	±941	±1203	±1338	±702
P5m5	8393	8009	**7699**	11026	9509	18747	16630
n300	±2623	±745	**±1271**	±527	±1296	±4071	±4671
P5m10	3491	**2772**	3077	4178	4204	9350	9527
n300	±1581	**±644**	±1217	±1364	±1069	±1975	±2188
P5m30	**1263**	1800	1268	1732	3102	6237	6659
n300	**±382**	±570	±382	±479	±487	±629	±1092
P5m2	**10949**	32134	32201	47845	48936	52590	49982
n500	**±1677**	±863	±676	±1585	±1684	±1662	±1575
P5m5	9823	11698	**9758**	16412	21898	33181	25245
n500	±2431	±1978	**±1535**	±1065	±5715	±4916	±3937
P5m10	5308	**4960**	5113	5833	11261	16577	15561
n500	±2212	**±1813**	±2219	±1602	±3516	±2113	±1757
P5m20	2596	2661	**2533**	3109	7966	13446	11530
n500	±1169	±913	**±1114**	±1268	±1189	±1622	±2465
P5m50	**1446**	1532	1475	1769	4479	6993	7011
n500	**±607**	±711	±698	±642	±859	±900	±743

performance in problems with a smaller number of salesmen. When the number of salesmen is increased, UMSA or UMHC slightly outperform UMEGS. In Table VI, MOEAD is able to produce solutions with the smallest traveling cost of any salesmen for problems with 100 cities, five and 10 salesmen even though its total traveling cost results are not considered superior.

VI. CONCLUSION

This paper has proposed a hybrid EDA with decomposition to solve TSP with multiple objective functions and salesmen (MmTSP). EDA that is based on UM is adapted to the multi-objective decomposition framework. Three local search metaheuristics have been hybridized with the decomposition EDA to complement the limitation of EDA in maintaining a set of diverse solutions. Both aims of the MmTSP, minimizing the total traveling cost and balancing the workload of the salesman, can be achieved by using different weight settings in the proposed problem formulation. This paper has shown that the decomposition algorithms scale well in both decision space and objective space. Comparative studies were also carried out between the proposed algorithms and three state-of-the-art MOEAs. From the results obtained in the comparative studies, pure EDA was able to achieve better proximity results in larger problems and the hybridization with local search improved the performance of EDA. Finally, the proposed algorithms demonstrated the best performance in most of the problem settings studied.

REFERENCES

[1] R. D. Angle, W. L. Caudle, R. Noonon, and A. Whinston, "Computer assisted school bus scheduling," *Manage. Sci.*, vol. 18, no. 6, pp. 278–288, 1972.

[2] H. A. Saleh and R. Chelouah, "The design of the navigation satellite system surveying networks using genetic algorithms," *Eng. Appl. Artif. Intell.*, vol. 17, no. 1, pp. 111–122, 2004.

[3] K. C. Gilbert and R. B. Hofstra, "A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem," *Decision Sci.*, vol. 23, no. 1, pp. 250–259, 1992.

[4] L. Tang, J. Liu, A. Rong, and Z. Yang, "A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron and Steel Complex," *Eur. J. Oper. Res.*, vol. 124, no. 2, pp. 267–282, 2000.

[5] Y. Zhong, J. Liang, G. Gu, R. Zhang, and H. Yang, "An implementation of evolutionary computation for path planning of cooperative mobile robots," in *Proc. 4th World Congr. Intell. Control Autom.*, 2002, vol. 3, pp. 1798–1802.

[6] A. Singh and A. S. Baghel, "A new grouping genetic algorithm approach to the multiple traveling salesperson problem," *Soft Comput.*, vol. 13, no. 1, pp. 95–101, 2009.

[7] S. Damas, O. Cordón, and J. Santamaría, "Medical image registration using evolutionary computation: An experimental survey," *IEEE Comput. Intell. Mag.*, vol. 6, no. 4, pp. 26–42, Nov. 2011.

[8] P. J. Werbos, "Computational intelligence for the smart grid-history, challenges, and opportunities," *IEEE Comput. Intell. Mag.*, vol. 6, no. 3, pp. 14–21, Aug. 2011.

[9] L. Perlovsky, "Computational intelligence applications for defense [research frontier]," *IEEE Comput. Intell. Mag.*, vol. 6, no. 1, pp. 20–29, Feb. 2011.

[10] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, U.K: Wiley, 2001.

[11] 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. 182–97, Apr. 2002.

[12] H. Ishibuchi, T. Yoshida, and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 28, no. 3, pp. 204–223, Aug. 1998.

[13] 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.

[14] P. Larrañaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Norwell, MA: Kluwer, 2001.

[15] J. A. Lozano, P. Larrañaga, and E. Bengoetxea, *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms (Studies in Fuzziness and Soft Computing)*. New York: Springer-Verlag, 2006.

[16] M. Pelikan, K. Sastry, and D. E. Goldberg, "Multiobjective estimation of distribution algorithm," in *Scalable Optimization via Probabilistic Modeling*. New York: Springer, 2006, pp. 223–248.

[17] R. Santana, P. Larrañaga, and J. A. Lozano, "Research topics in discrete estimation of distribution algorithms based on factorizations," *Memetic Comput.*, vol. 1, no. 1, pp. 35–54, 2009.

[18] X. Zhong and W. Li, "A decision-tree-based multi-objective estimation of distribution algorithm," in *Proc. Int. Conf. Comput. Intell. Security*, 2007, pp. 114–118.
[19] H. Muhlenbein and G. Paass, "From recombination of genes to the estimation of distributions I. Binary parameters," in *Proc. 4th Int. Conf. Parallel Problem Solving Nature*, 1996, pp. 178–187.
[20] P. A. N. Bosman and D. Thierens, "Continuous iterated density estimation evolutionary algorithm within the IDEA Framework," in *Proc. OBUPM Workshop Genetic Evol. Comput. Conf.*, 2000, pp. 197–200.
[21] H. J. Tang, V. A. Shim, K. C. Tan, and J. Y. Chia, "Restricted Boltzmann machine based algorithm for multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–8.
[22] L. Marti, J. Garcia, A. Berlanga, and J. M. Molina, "Solving complex high-dimensional problems with the multi-objective neural estimation of distribution algorithm," in *Proc. 11th Annu. Genetic Evol. Comput. Conf.*, 2009, pp. 619–626.
[23] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 41–63, Feb. 2008.
[24] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solution procedures," *Int. J. Manage. Sci.*, vol. 34, no. 3, pp. 209–219, 2005.
[25] T. Zhang, W. A. Gruver, and M. H. Smith, "Team scheduling by genetic search," in *Proc. 2nd Int. Conf. Intell. Process. Manuf. Mater.*, 1999, pp. 829–844.
[26] Y. B. Park, "A hybrid genetic algorithm for the vehicle scheduling problem with due times and times deadlines," *Int. J. Prod. Econ.*, vol. 73, no. 2, pp. 175–188, 2001.
[27] A. E. Carter and C. T. Ragsdale, "A new approach to solving the multiple traveling salesperson problem using genetic algorithms," *Eur. J. Oper. Res.*, vol. 175, no. 1, pp. 246–257, 2006.
[28] F. Zhao, J. Dong, S. Li, and X. Yang, "An improved genetic algorithm for multiple traveling salesman problem," in *Proc. 2nd Int. Asia Conf. Informat. Control, Autom. Robot.*, 2008, pp. 493–495.
[29] A. Kiraly and J. Abonyi, "Optimization of multiple traveling salesman problem by a novel representation based genetic algorithm," in *Proc. 10th Int. Symp. Hungarian Res. Comput. Intell. Informat.*, 2010, pp. 315–326.
[30] J. Zhang, Z. Zhan, Y. Lin, N. Chen, Y. Gong, J. Zhong, H. Chung, Y. Li, and Y. Shi, "Evolutinary computation meets machine learning: A survey," *IEEE Comput. Intell. Mag.*, vol. 6, no. 4, pp. 68–75, Oct. 2011.
[31] Y. S. Ong, M. Lim, and X. S. Chen, "Memetic computation—past, present & future," *IEEE Comput. Intell. Mag.*, vol. 5, no. 2, pp. 24–31, May 2010.
[32] K. Tang, K. C. Tan, and H. Ishibuchi, "Guest editorial: Memetic algorithms for evolutionary multi-objective optimization," *Memetic Comput.*, vol. 2, no. 1, pp. 1–2, 2010.
[33] J. Y. Chia, C. K. Goh, K. C. Tan, and V. A. Shim, "Memetic informed evolutionary optimization via data mining," *Memetic Comput.*, vol. 3, no. 2, pp. 73–88, 2011.
[34] J. Tang, M. H. Lim, and Y. S. Ong, "Adaptation for parallel memetic algorithm based on population entropy," in *Proc. 8th Annu. Conf. Genetic Evol. Comput.*, 2006, pp. 575–582.
[35] Y. S. Ong, M. H. Lim, F. Neri, and H. Ishibuchi, "Special issue on emerging trends in soft computing: Memetic algorithms," *Soft Comput.*, vol. 13, no. 8–9, pp. 739–740, 2009.
[36] F. Neri and E. Mininno, "Memetic compact differential evolution for cartesian robot control," *IEEE Comput. Intell. Mag.*, vol. 5, no. 2, pp. 54–65, May 2010.
[37] K. K. Lim, Y. S. Ong, M. H. Lim, X. Chen, and A. Agarwal, "Hybrid ant colony algorithms for path planning in sparse graphs," *Soft Comput.*, vol. 12, no. 10, pp. 981–994, 2008.
[38] Y. Wang, T. H. Cheng, and M. H. Lim, "A tabu search algorithm for static routing and wavelength assignment problem," *IEEE Commun. Lett.*, vol. 9, no. 9, pp. 841–843, Sep. 2005.
[39] J. M. Renders and H. Bersini, "Hybridizing genetic algorithms with hill-climbing methods for global optimization: Two possible ways," in *Proc. 1st IEEE Congr. Evol. Comput.*, Jun. 1994, vol. 1, pp. 312–317.
[40] F. Rojas, C. G. Puntonet, M. Rodriquez, I. Rojas, and R. Martin, "Blind source separation in post-nonlinear mixtures using competitive learning, simulated annealing, and a genetic algorithm," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 34, no. 4, pp. 407–416, Nov. 2004.
[41] R. Solomon, "Evolutionary algorithms and gradient search: Similarities and differences," *IEEE Trans. Evol. Comput.*, vol. 2, no. 2, pp. 45–55, Jul. 1998.
[42] C. K. Goh, Y. S. Ong, K. C. Tan, and E. J. Teoh, "An investigation on evolutionary gradient search for multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2008, pp. 3741–3746.
[43] W. T. Koo, C. K. Goh, and K. C. Tan, "A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment," *Memetic Comput.*, vol. 2, no. 2, pp. 87–110, 2010.
[44] C. Okonjo, "An effective method of balancing the workload amongst salesman," *Omega*, vol. 16, no. 2, pp. 159–163, 1998.
[45] K. Miettinen, *Nonlinear Multiobjective Optimization*. Norwell, MA: Kluwer, 1999.
[46] V. A. Shim, K. C. Tan, and J. Y. Chia, "Probabilistic based evolutionary optimizers in bi-objective traveling salesman problem," in *Proc. 8th Int. Conf. Simulated Evol. Learning*, 2010, pp. 588–592.
[47] W. Peng, Q. Zhang, and H. Li, "Comparison between MOEA/D and NSGAII on the multi-objective traveling salesman problem," in *Multi-Objective Memetic Algorithm*, vol. 171, C.-K. Goh, K. C. Tan, and Y.-S. Ong, Eds. New York: Springer-Verlag, 2009, pp. 309–324.
[48] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications," Ph.D. dissertation, Comput. Eng. Netw. Lab., Swiss Federal Institute of Technology, Zurich, Switzerland, 1999.
[49] M. A. Villalobos-Arias, G. T. Pulido, and C. A. C. Coello, "A proposal to use stripes to maintain diversity in a multi-objective particle swarm optimizer," in *Proc. 2005 IEEE Swarm Intell. Symp.*, 2005, pp. 23–30.

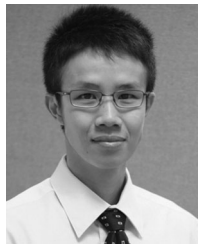
V. A. Shim received the B. Eng. degree (Hons.) in electrical and electronic engineering from the Universiti Teknologi Malaysia, Johor, Malaysia, in 2008. He is currently working toward the Ph.D. degree with the Centre for Intelligent Control, National University of Singapore, Singapore.

His current research interests include evolutionary computation and neural networks, specifically in the application of neural-network-based estimation of distribution algorithm for multiobjective optimization.

K. C. Tan received the B. Eng. degree with First Class Honors in electronics and electrical engineering, and the Ph.D. degree from the University of Glasgow, U.K., in 1994 and 1997, respectively.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, National University of Singapore (NUS). He has published more than 200 journal and conference papers and coauthored five books. He has been invited to be a Keynote/Invited Speaker for 21 international conferences, and served in the international program committee for more than 100 conferences and involved in the organizing committee for more than 30 international conferences.

Dr. Tan is currently a Distinguished Lecturer of the IEEE Computational Intelligence Society and the Editor-in-Chief of the IEEE Computational Intelligence Magazine. He also serves as an Associate Editor/Editorial Board member of 15 international journals, such as the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES, *Evolutionary Computation* (MIT Press), the *European Journal of Operational Research*, the *Journal of Scheduling*, and the *International Journal of Systems Science*. He is currently a Fellow of the NUS Teaching Academic.



C. Y. Cheong received the Ph.D. degree in electrical engineering from the National University of Singapore, Singapore, in 2010.

Since then, he has been a Scientist in the Computing Science Department, Institute of High Performance Computing, A∗STAR, Singapore. His current research interests include high performance computing, evolutionary computation, and multiobjective optimization.