Research Article

Minimization of the Total Traveling Distance and Maximum Distance by Using a Transformed-Based Encoding EDA to Solve the Multiple Traveling Salesmen Problem

S. H. Chen

Department of Information Management, Cheng Shiu University, No. 840, Chengcing Road, Niaosong District, Kaohsiung 83347, Taiwan

Correspondence should be addressed to S. H. Chen; shchen@csu.edu.tw

Estimation of distribution algorithms (EDAs) have been used to solve numerous hard problems. However, their use with in-group optimization problems has not been discussed extensively in the literature. A well-known in-group optimization problem is the multiple traveling salesmen problem (mTSP), which involves simultaneous assignment and sequencing procedures and are shown in different forms. This paper presents a new algorithm, named EDA_{MLA}, which is based on self-guided genetic algorithm with a minimum loading assignment (MLA) rule. This strategy uses the transformed-based encoding approach instead of direct encoding. The solution space of the proposed method is only $n!$. We compare the proposed algorithm against the optimal direct encoding technique, the two-part encoding genetic algorithm, and, in experiments on 34 TSP instances drawn from the TSPLIB, find that its solution space is $n! \binom{n-1}{m-1}$. The scale of the experiments exceeded that presented in prior studies. The results show that the proposed algorithm was superior to the two-part encoding genetic algorithm in terms of minimizing the total traveling distance. Notably, the proposed algorithm did not cause a longer traveling distance when the number of salesmen was increased from 3 to 10. The results suggest that EDA researchers should employ the MLA rule instead of direct encoding in their proposed algorithms.

1. Introduction

Estimation of distribution algorithms (EDAs) use the learning while optimizing principle [1]. Two review articles have suggested that EDAs have emerged as a prominent alternative to evolutionary algorithms [2, 3]. In contrast to genetic algorithms (GAs), which employ crossover and mutation operators to generate solutions, EDAs explicitly extract global statistical information from the previous search to build a posterior probability model of promising solutions from which new solutions are sampled [4, 5]. This crucial characteristic distinguishes EDAs from GAs [6, 7].

Numerous studies aimed at using EDAs to solve nondeterministic polynomial-time hard (NP-hard) scheduling problems have shown that EDAs are able to perform effectively in terms of the solution quality [2, 8, 9]. Ceberio et al. [2], in particular, extensively tested 13 famous permutation-based EDAs on four combinatorial optimization problems,

including the quadratic assignment problem, traveling salesman problem (TSP), permutation flowshop scheduling problems (PFSPs), and linear ordering problem. Their paper provides a good basis for comparison.

Although EDAs are effective in solving various hard problems, EDA studies seldom extensively discuss a problem. To our knowledge, only one EDA, namely, that is proposed by Shim et al. [10], can solve in-group optimization problems such as the multiple traveling salesmen problem (mTSP) and parallel machine scheduling problems (PMSPs) [11]. In-group optimization problems involve assigning and sequencing procedures simultaneously. Take the mTSP, for example: a number of n cities are assigned to m salesmen and these n cities are visited only once by a salesman, where $n > m$. Thus, this appears to be an NP-hard problem.

Because only one EDA could solve in-group optimization problems, there is much room for additional research. In-group optimization problems are relevant in industry, such

as in the application of the mTSP. This research developed a new EDA, named EDA_{MLA}, dealt with by using a self-guided genetic algorithm (SGGA) [12] with the minimum loading assignment (MLA) rule to solve the mTSP. As opposed to direct encoding, the proposed strategy is called the transformed-based encoding approach. The solution space of the MLA is only $n!$. We compare the proposed algorithm against the optimal direct encoding technique, the two-part encoding genetic algorithm (TPGA) [13]. Notably, the solution space of the two-part encoding approach is $n! \binom{n-1}{m-1}$. The proposed MLA method, consequently, is superior to the two-part encoding technique, and an improved solution quality is expected when the SGGA works with the MLA method.

This paper is organized as follows: Section 2 primarily reviews the literature on in-group optimization problems, encoding techniques, and EDAs. In Section 3, the core MLA method is presented to dispatch n cities to m salesmen. This assignment rule is further employed by the SGGA in Section 4. Section 5 reveals the effectiveness of the proposed algorithm, which is compared with the existing famous direct encoding methods, including the one-chromosome and two-part chromosome encoding. Finally, we draw conclusions in Section 6.

2. Background Information

The mTSP is a well-known in-group optimization problem. We review mTSP studies and their variants in Section 2.1. To solve in-group optimization problems, numerous encoding techniques could be applied in evolutionary algorithms. Solution representations fall into two classes: direct and indirect encoding methods [11], relevant studies about which are presented in Sections 2.2 and 2.3, respectively. The final section illustrates combinatorial-based EDAs.

2.1. In-Group Optimization Problems. Bektas [11] reviewed the seven types of in-group optimization problems, which we detail in Table 1. Among the variants of in-group optimization problems, the most well-known form is the mTSP because it models daily activities and exists in every enterprise [13]. The problem properties of the mTSP include assignment and sequence optimization procedures. For instance, we must optimize the traveling sequence for the route of each salesman. Both procedures directly lead to the traveling cost and time of the trip after assigning m salesmen to visit n places every day. A detailed definition of the mTSP can be found in [11].

Although the mTSP could be solved using exact algorithms [14–16], large-sized problems are not solved efficiently. To deal with large-size instances, evolutionary algorithms (EAs) are a commonly used approach. The first crucial step of using EAs is selecting the appropriate encodings. Encoding approaches are presented in the next section.

2.2. Direct Encoding Methods. There are five major direct encodings of EAs: one-chromosome [17], two-chromosome

TABLE 1: Application contexts for the in-group optimization problems.

| Application context	Type of application
Routing	mTSP [13, 24, 46–48]
Print scheduling	Print press scheduling [49]
	Preprint advertisement scheduling [50]
Workforce planning	Bank crew scheduling [51]
	Technical crew scheduling [52]
	Photographer team scheduling [53]
	Interview scheduling [54]
	Workload balancing [55]
	Security service scheduling [56]
Transportation planning	School bus routing [57]
	Crane scheduling [58]
	Local truckload pickup and delivery [59]
	Vehicle routing problem [60, 61]
Mission planning	Planning of autonomous mobile robots [62–65]
	Planning of unmanned air vehicles [66]
Production planning	Hot rolling scheduling [17]
	Parallel machine scheduling with setup [29]
Satellite systems	Designing satellite surveying systems [67]
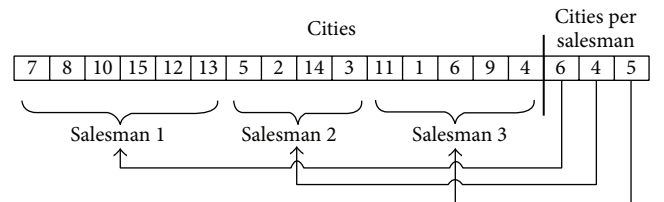
FIGURE 1: A representation of the two-part chromosome encoding for 15 cities and three salesmen.

[18, 19], two-part chromosome [13], grouping genetic algorithms (GGAs) [20–22], and matrix representation [23]. Two-part chromosome encoding, which is superior to one- and two-chromosome encoding [13] because of its smaller solution space, is depicted in Figure 1.

We assume this encoding with $n = 15$ and $m = 3$. There are two distinct parts. The first part of the chromosome represents the permutation of n cities. The second part of the chromosome shows the number of cities assigned to each m salesman so that its chromosome length is m. The total sum of m genes is equal to the number of n cities. In [24], they examined an improved combination of crossover and mutation operators for the two-part chromosome encoding method and suggested appropriate genetic operators that could be applied in GAs.

GGAs commonly use an array of jobs for each parallel machine, and the processing order of the jobs assigned to that machine is shown [25]. Kashan et al. [26] extended the GGAs into the grouping version of the particle swarm optimization algorithm. Later, Arnaout et al. [23] proposed a matrix representation of the N jobs on M machines, whose

size is $M \times N$. Each row indicates the parallel machines and the processing sequence of the jobs on it. When there are no jobs to be processed on a machine, number 0 is inserted into the blank spaces. As a result, it became apparent that GGAs memory usage was inefficient, though Liao et al. [27] found that this approach was better than the other four variants of hybrid ant colony optimization. Thus, $M \times N - N$ spaces are unused if we apply this encoding technique.

In these direct encoding techniques, the optimal approach could be the two-part chromosome technique, according to Carter and Ragsdale [13]. When we have n items and m groups, the solution space of one-chromosome encoding requires $(n + m - 1)!$. Two-chromosome encoding takes $n!m^n$ and the size of the two-part chromosome is $n! \binom{n-1}{m-1}$.

2.3. Indirect Encoding Methods. The transformed-based encoding type separates sequencing and assignment decisions because the complex encoding may yield poor results [28]. Its encoding strategy first utilizes permutation encoding and then assigns the items into groups at every stage. Although this approach could be used to solve the PMSP [29], the separated method is also applicable in complex flowshop problems involving numerous parallel machines in the flowshop. Ruiz and Maroto [28] referred to this application as the priority rules for hybrid flowshops. Wang et al. [30] called it the earliest completion factory rule for solving the distributed permutation flowshop scheduling problem. Salhi et al. [31] selected the index of the machine that allows a job that has the shortest completion time for solving complex flowshop scheduling problems.

To achieve optimal efficiency, this study adopts the transformed-based encoding method instead of direct encoding. In addition, several EAs could apply the assignment rule and then solve the in-group optimization problem. To evaluate the performance of the algorithms examined in this study, we select the mTSP for an extensive comparison.

In presenting the latest development in EDAs, it is clear that only a few can solve in-group assignment problems. Thus, this study is relevant to the investigation of in-group assignment problems.

2.4. Recently Developed Combinatorial-Based EDAs. Unlike the implicit processing of building blocks in GAs, EDAs explicitly rely on the used probability model. The building blocks are based on selection and crossover operators that do not preserve essential patterns [32]. The probability model is the core factor in affecting the performance of EDAs. The more accurate the probability model is, the more effective the algorithm will be in preventing the disruption of essential building blocks [33]. In general, a distinguishing characteristic of EDAs is their application of the probabilistic model, which is not used by GAs.

Numerous attempts at using EDAs to solve sequencing or combinatorial optimization problems have been made. For example, Chang et al. [34] proposed a hybrid framework to alternate between EDAs and genetic operators for solving the single machine scheduling problem. A position-based

univariate probability model was used in the proposed algorithm. The hybrid framework is beneficial, because though EDAs efficiently improve solution quality in the first few runs, the loss of diversity rapidly increases as additional iterations are executed [7, 35, 36].

Jarboui et al. [37] proposed a hybrid approach, named EDA-VNS, that combined EDAs with the variable neighborhood search (VNS) [38] to solve PFSPs by using the minimization of the total flowtime. Their probabilistic model considered the order of the job queue and the building blocks of the jobs. This was the first attempt to take into account both first- and second-order statistical information. In addition, VNS improved as the EDA was run. Jarboui et al. [37] found that EDA-VNS was effective in small benchmarks; however, for larger problems, VNS was superior to EDA-VNS in terms of objective values and computational time. It was unclear why EDA-VNS did not outperform the VNS in large benchmarks. A new EDA in [4] also employed job permutation and similar blocks of jobs to solve lot-streaming flowshop problems. In this EDA, the definitions of job permutation and similar blocks differed from those of [37]; it also introduced a diversity measure to restart evolutionary progress when the population diversity decreased to a certain level.

In contrast to traditional EDAs, an SGGA uses a probabilistic model as the fitness function surrogate [39]. The probabilistic model guides the evolutionary direction in selecting candidate solutions for crossover and mutation operators. An SGGA could solve PFSPs. It could also be integrated with dominance properties to solve single machine scheduling problems [40]. An eSGGA was proposed for problems involving variable interactions [9].

To the best of our knowledge, the first EDA for the mTSP involved applying the one-chromosome representation [10]. Because there are $m - 1$ pseudo cities introduced in the chromosome, every chromosome comprises $n + m - 1$ genes. As a result, the dimension of their probability model $P_r(x)$, by computing the marginal probability of each city, is $N \times N$ where N is $n + m - 1$. This might be a drawback of the first EDAs, which were inherited from one-chromosome encoding, even though their performance was superior to three state-of-the-art multiobjective evolutionary algorithms. Consequently, the proposed algorithm EDA_{MLA}, together with the MLA rule, may be the second EDA to solve the mTSP; it is a promising algorithm that does not use the larger probability model of EDAs.

3. Minimum Loading Assignment Rule in the mTSP

Suppose that there is a set of n cities, sequenced $\pi_1, \pi_2, \ldots, \pi_n$ in π, that could be assigned to m salesmen. These cities are not yet assigned to any salesmen. The sequence π could be decoded by using an assignment rule to assign the cities to each salesman. After the assignment rule is executed, we can calculate the fitness function of each chromosome. We propose an MLA rule to perform the assignment work. The following pseudocode in Algorithm 1 illustrates this rule.

Require:
    i: The position of a city in the sequence π
    $k[i]$: The current number of cities assigned to a salesman i
    $\Omega_{k[i]}^{i}$: The visiting sequence of n cities
(1) $i \leftarrow 1$
(2) **while** $i \leq m$ **do**
(3) $k[i] \leftarrow 1$
(4) $\Omega_{k[i]}^{i} \leftarrow \pi_i$
(5) $i \leftarrow i + 1$
(6) $k[i] \leftarrow k[i] + 1$
(7) **end while**
(8) **while** $i \leq m$ **do**
(9)   Select a salesman j who could process the π_i with the minimum objective value
(10) $\Omega_{k[j]}^{j} \leftarrow \pi_i$
(11) $i \leftarrow i + 1$
(12) $k[i] \leftarrow k[i] + 1$
(13) **end whie**

ALGORITHM 1: Minimum loading assignment rule.

Population: A set of solutions
Generations: The maximum number of generations
$P(t)$: Probabilistic model
t: Generation index
(1) Initialize *Population*
(2) $t \leftarrow 0$
(3) Initialize $P(t)$
(5) **while** $t < generations$ **do**
(6)   EvaluateFitness(*Population*) with minimum loading assignment rule
(7)   Selection/Elitism(*Population*)
(8) $P(t + 1) \leftarrow$ BuildingProbabilityModel(*Selected Chromosomes*)
(9)   Self-Guided Crossover( )
(10)   Self-Guided Mutation( )
(11) $t \leftarrow t + 1$
(12) **end while**

ALGORITHM 2: Main procedures of EDA$_{\text{MLA}}$.

In the beginning, the first m cities are assigned to m salesmen and the objective values of each salesman are calculated. The objective function of the mTSP would be the total traveling distance or maximum traveling distance among salesman. The MLA rule is then applied iteratively for unassigned cities. The MLA rule assigns the first unassigned city in the sequence π to a salesman when it results in the minimum objective value. This assigned city is removed from π. This process continues until no cities are left in π.

By using the MLA rule, a city could be assigned to a salesman who has less loading. It also ensures that this assigned city is close to the last city visited by the salesman; a faraway city would not be considered. Through the MLA rule, mTSP can be extended to the PMSP with a setup consideration or the distributed flowshop scheduling problem.

4. Proposed Algorithm: EDA$_{\text{MLA}}$

This section explains the procedures of the EDA with the MLA rule. The advantages of the proposed method include preserving the salient genes of the chromosomes and exploring and exploiting optimal searching directions for genetic operators [40, 41]. The major difference between this proposed algorithm and other works is the problem type; other studies have been aimed at solving the sequencing problem, whereas we addressed the grouping and sequencing problems simultaneously. The major procedures of EDA$_{\text{MLA}}$ are shown in Algorithm 2.

In Step (1), the population is initialized and the sequence of each chromosome is generated randomly. Step (3) builds the probability matrix $P(t)$ with a matrix dimension of n by

n, where n is the problem size. Each $P_{ij}(t)$ is initialized to be $1/n$, where n is the total number of cities in |Parentset|. This initialization means that all solutions have the same likelihood of being an optimal solution. The reason for such an initialization is that we have no information about the location of promising solutions.

In Step (5), we evaluate the objective value of each solution. In addition, the MLA rule is used here (see Algorithm 1). After all n cities are assigned to m salesmen, the algorithm evaluates the total distance of all salesmen and the maximum distance among the m salesmen. In Step (6), a binary tournament selection is used to select good solutions from the population.

Step (7) forms the probability model $P(t)$ after the selection procedure. The calculation details are outlined in Section 4.1. In Steps (8) and (9), $P(t)$ is employed in the self-guided crossover and mutation operators. The probability model is used as a fitness surrogate, which is shown in Sections 4.2 to 4.4. We use the two-point partial mapping crossover and swap mutation in the crossover and mutation procedures for solving the mTSP.

The proposed algorithm is explained in the following sections. We first describe the probability model of the EDA and then explain how the probabilistic model guides the crossover and mutation operators.

4.1. Formulation of the Probabilistic Model.
The probability model $P(t)$ of the EDA is defined as

$$P(t) = \begin{pmatrix} P_{11}(t) & \cdots & P_{1n}(t) \\ \vdots & \ddots & \vdots \\ P_{n1}(t) & \cdots & P_{nn}(t) \end{pmatrix}, \qquad (1)$$

where $P_{ij}(t)$ is the probability of city i being in position j in a promising solution. $P(t)$ summarizes the global statistical information about promising solutions obtained from the previous search.

Let ϕ_{ij} be the number of solutions in Parentset, in which city i is in position j and |Parentset| is the size of Parentset. $P_{ij}(t+1)$ in Line (7) is updated as follows:

$$P_{ij}(t+1) = (1-\lambda)P_{ij}(t) + \lambda \frac{\phi_{ij}+1}{|\text{Parentset}|+n}, \qquad (2)$$

where $\phi_{ij}/|\text{Parentset}|$ is the percentage of solutions in which city i is in position j. It represents the knowledge of promising solutions learned from the current generation. We use $(\phi_{ij}+1)/(|\text{Parentset}|+n)$, the Laplace correction of $\phi_{ij}/|\text{Parentset}|$ in (2), to prevent P_{ij} from becoming very small [42–44]. $P_{ij}(t)$ is the historical knowledge of promising solutions. We update $P(t+1)$ in an incremental manner, as suggested by [45]. $\lambda \in (0,1)$ balances the contribution from historical knowledge with that of the knowledge learned from the current generation.

4.2. Probabilistic Model as the Fitness Surrogate.
With the probabilistic model $P(t+1)$, we define the following function to predict the quality of solution X:

$$Q_{t+1}(X) = \prod_{k=1}^{n} P_{k[k]}(t+1), \qquad (3)$$

where $[k]$ is the position of city k in X. The following should be noted regarding this function:

(i) $P_{k[k]}(t+1)$ is the probability that city k in position $[k]$ is a promising solution. Therefore, $Q_{t+1}(X)$ can measure how promising X is.

(ii) In general, $Q_{t+1}(X)$ is not an exact probability measure of the set of all the solutions of X because

$$\sum_X Q_{t+1}(X) \neq 1. \qquad (4)$$

$Q_{t+1}(X)$ is only an estimation value of the probability that X is promising. This estimation is more effective and much easier to compute compared with other probabilistic models in the literature. Thus, this method is effective and reduces computational time.

$Q_{t+1}(X)$ is applied to select good candidate solutions during the crossover and mutation operation. In the following subsection, we drop $t+1$ in P and Q for simplicity.

4.3. Crossover Operator with Probabilistic Model.
With the surrogate function in (3), we preevaluate the solution quality of the new solutions generated by the crossover and mutation operators. In the normal two-point crossover procedure, two random cut-points, K and L, are set in the beginning, where K is less than L. Then, a parent solution X mates with the other parent solution to yield a new offspring. However, a difference exists in the proposed algorithm.

Because of the difference, we let a parent solution X mate with a set of randomly selected solutions Y. The size of Y ranges from 2 to TC, where TC is the number of tournaments. These crossover steps produce a set of offspring Z. The quality difference between offspring Z_i and parent solution X is denoted as Δ_i. Δ_i is given as follows:

$$\begin{aligned} \Delta_1 &= Q(Z) - Q(X) \\ &= \left[\prod_{K \le k \le L} P_{y_i i} - \prod_{K \le k \le L} P_{x_i i} \right] \prod_{1 \le i < K} P_{x_i i} \prod_{L < i \le n} P_{x_i i}. \end{aligned} \qquad (5)$$

The larger Δ_i is, the more likely that Z_i is superior to other offspring when a set of parent solutions Y mate with a solution X. Hence, Z_i is added to the offspring population. We repeat the crossover steps to generate offspring until the offspring population is full. Both the concepts of self-guided crossover and self-guided mutation employ the same idea. The mutation procedure is shown in the next section.

4.4. Mutation Operator with Probabilistic Model.
Suppose that two cities i and j are randomly selected and they are located in position a and position b, respectively. p_{ia} and

p_{jb} denote city i in position a and city j in position b. After these two cities are swapped, the new probabilities of the two cities become p_{ib} and p_{ja}. The probability difference Δ_{ij} is calculated as (6), which is a partial evaluation of the probability difference because the probability sum of the other cities remains the same:

$$\Delta_{ij} = Q\left(X'\right) - Q\left(X\right)$$
$$\approx \prod_{p \notin (a \text{ or } b), g=[p]}^{n} P_{t+1}\left(X_{gp}\right)\left[\left(p_{ib}p_{ja}\right) - \left(p_{ia}p_{jb}\right)\right]. \quad (6)$$

Now because the part of $\prod_{p \notin (a \text{ or } b), g=[p]}^{n} P_{t+1}(X_{gp})$ is always ≥ 0, it can be subtracted, and (6) is simplified as follows:

$$\Delta_{ij} = \left(p_{ib}p_{ja}\right) - \left(p_{ia}p_{jb}\right),$$
$$\Delta_{ij} = \left(p_{ib} + p_{ja}\right) - \left(p_{ia} + p_{jb}\right). \quad (7)$$

If Δ_{ij} is positive, it implies that one gene or both genes might move to a promising area. On the other hand, when Δ_{ij} is negative, the implication is that at least one gene moves to an inferior position.

On the basis of the probabilistic differences, it is natural to consider different choices of swapping points during the mutation procedure. A parameter TM is introduced for the self-guided mutation operator, which denotes the number of tournaments in comparing the probability differences among the TM choices in swap mutation. Basically, TM ≥ 2 while TM = 1 implies that the mutation operator mutates the genes directly without comparing the probability differences among the different TM choices.

When TM = 2, suppose the other alternative is that two cities m and n are located in position c and position d, respectively. The probability difference of exchanging cities m and n is

$$\Delta_{mn} = \left(p_{md} + p_{nc}\right) - \left(p_{mc} + p_{nd}\right). \quad (8)$$

After Δ_{ij} and Δ_{mn} are obtained, the difference between the two alternatives is as follows:

$$\Delta = \Delta_{ij} - \Delta_{mn}. \quad (9)$$

If $\Delta < 0$, the contribution of swapping cities m and n is better, so we swap cities m and n. Otherwise, cities i and j are swapped. Consequently, the option of a larger probability difference is selected and the corresponding two cities are swapped. By observing the probability difference Δ, the self-guided mutation operator exploits the solution space to enhance the solution quality and prevent destroying some dominant genes in a chromosome. Moreover, the main procedure of the self-guided mutation is (9), where the time-complexity is only a constant after the probabilistic model is employed. This approach proves to work efficiently.

To conclude, the EDA$_{MLA}$ is obviously different from the previous EDAs. Firstly, the algorithm utilizes the transformed-based encoding instead of using the direct encoding used by Shim et al. [10]. Secondly, the proposed algorithm explicitly embeds the probabilistic model in the crossover and mutation operators to explore and exploit the solution space. Most important of all, the algorithm works more efficiently than previous EDAs [10] in solving the mTSP because the time-complexity is $O(n)$ whereas the previous EDAs need $O(n^2)$ time. They are the major differences to other existing EDAs.

5. Experimental Results

We conducted extensive computational experiments to evaluate the performance of EDA$_{MLA}$ in solving the mTSP. There were 34 instances selected from the well-known traveling salesman problem library, TSPLIB, and the size of these instances ranged from 48 to 400. This paper assumed that the first city of each instance was the home depot. The number of salesmen used was 3, 5, 10, and 20. Hence, there were 136 instances in the experiments. Across all the experiments, we replicated each instance 30 times.

The proposed algorithm was compared with the benchmark encoding algorithm and a classic encoding, which are the TPGA [13] and one-chromosome genetic algorithm [17], respectively. We employed the genetic operators and parameter settings of the TPGA suggested by S.-H. Chen and M.-C. Chen [24], because they used the design-of-experiments (DOE) to select significant parameters; the genetic operators are the two-point partial mapping crossover operator and swap mutation operator. This ensures a fair comparison between the proposed algorithm and benchmark encoding algorithm. One-chromosome GA utilizes the same operators of TPGA and also employs the DOE to tune the parameters as well. The crossover and mutation rate of the one-chromosome GA are 0.5 and 0.1, respectively. Finally, a standard genetic algorithm also applies the MLA rule, which is named GA$_{MLA}$. GA$_{MLA}$ could show whether the performance is enhanced by the assignment rule proposed by this research.

We implemented the algorithms in Java 2 on an Amazon EC2 with a Windows 2012 server (32-core CPU). The stopping criterion is the number of examined solutions which is up to 100,000. The objective functions include minimizing the total traveling distance and maximizing the traveling distance, which are detailed in Sections 5.1 and 5.2, respectively.

5.1. Total Traveling Distance Results. This objective evaluates the total distance traveled by the m salesmen. It reflects the total cost of the assignment. Figure 2 shows the main effects plot of the method comparison and the differences in the number of salesmen assigned. This figure clearly illustrates that the EDA$_{MLA}$ and GA$_{MLA}$ are superior to the one-chromosome GA and TPGA. This indicates that the MLA rule, that is, the transformed-based method, could be a more promising approach than the direct encoding methods. The total distance increased greatly with the number of salesmen, particularly when 20 salesmen could be assigned. This implies that the request of too many salesmen would be inefficient from a managerial perspective.

Figure 3 depicts the interaction plot between the factor method and number of salesmen. Notably, the EDA$_{MLA}$
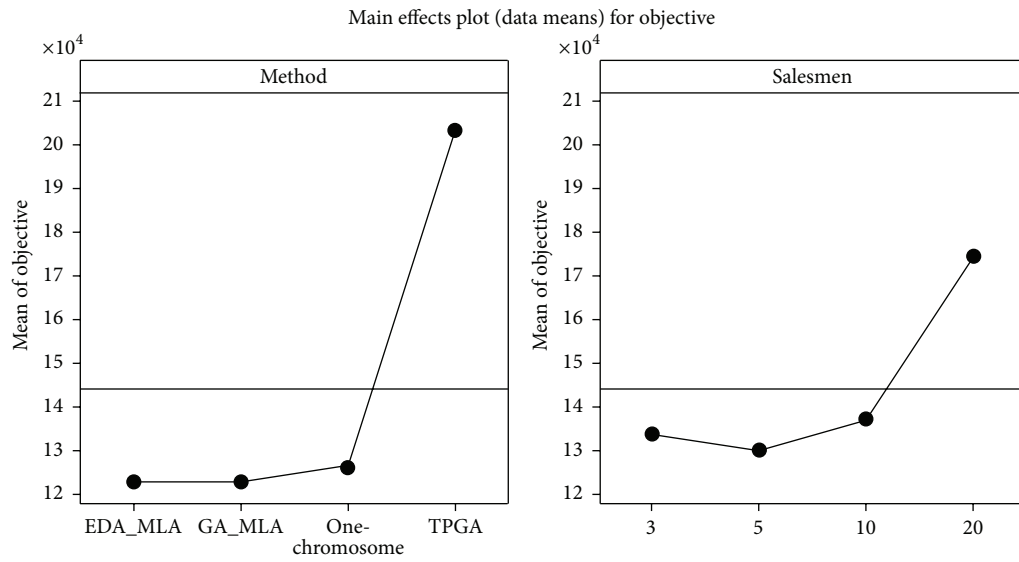
FIGURE 2: Main effects plot on the total travelling distance of the compared algorithms.
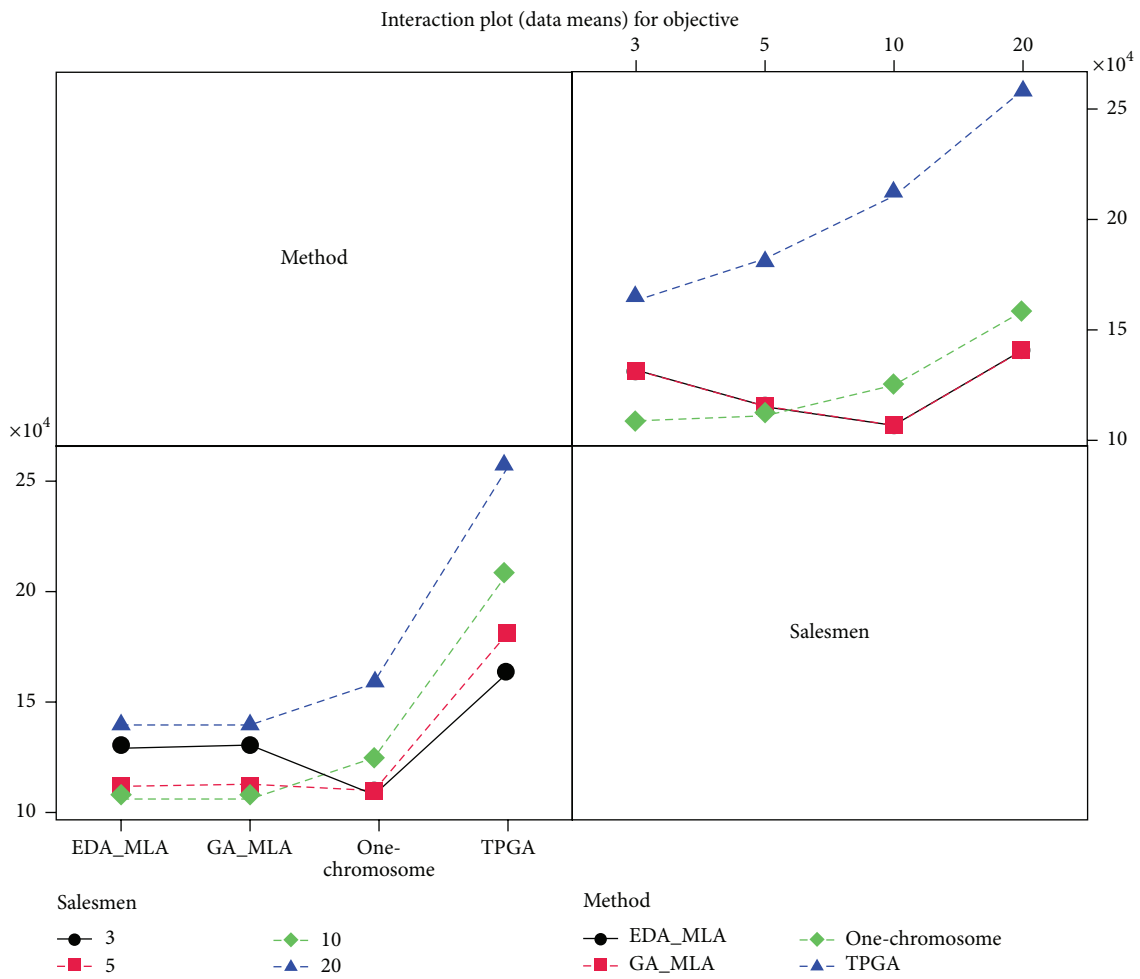

FIGURE 3: Intreaction plot on the total travelling distance.
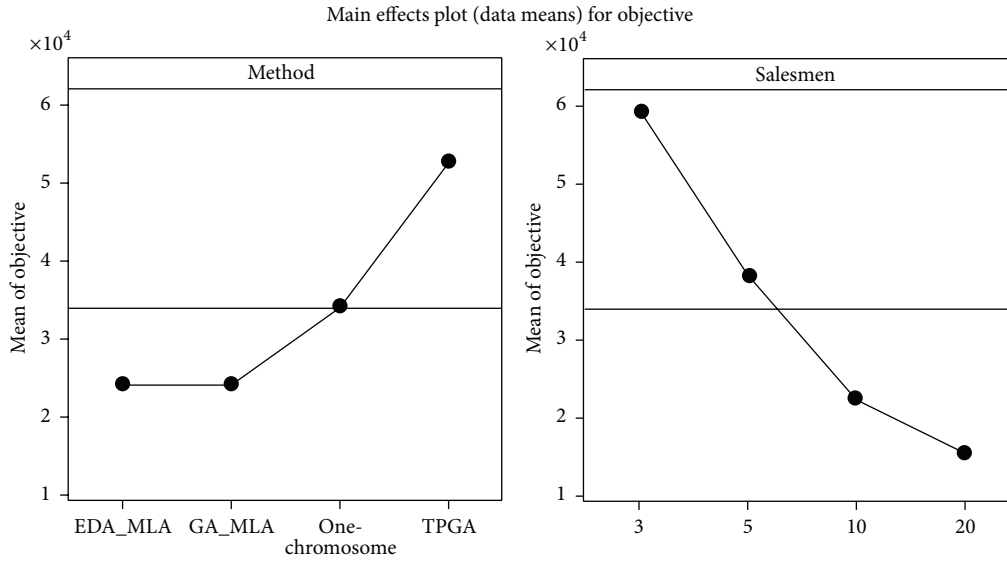
Main effects plot (data means) for objective

FIGURE 4: Main effects plot of the maximum traveling distance for the compared algorithms.

and GA_{MLA} did not yield a longer total traveling distance when the number of salesmen increased from two to 10. However, the TPGA suffered when the number of salesmen was increased. Thus, this figure reveals the effectiveness of the transform-based rule compared with the direct encoding method. According to this interaction plot, if a manager wants to determine how many salesmen are required, the lowest total traveling distance would be 5.

Table 2 presents the results of the total traveling distance for the four algorithms. This table shows the minimum, mean, maximum, and the standard deviation (StDev). Among these 34 instances, EDA_{MLA} is better than one-chromosome GA, TPGA, and GA_{MLA} out of the 17 cases when it comes to the average of the total distance. In addition, the standard deviation of one-chromosome GA, TPGA, GA_{MLA}, and EDA_{MLA} is 21187, 33230, 19785, and 20041, respectively. It implies that the EDA_{MLA} yields less variance than one-chromosome GA and TPGA. EDA_{MLA} might be more robust in terms of the average performance and the variance.

5.2. Maximum Traveling Distance Results. The maximum traveling distance was used as the second objective tested by the three algorithms. Thus, the algorithms minimized the loading of the salesman with the highest loading. As a result, this objective balanced the loading among the salesmen. As shown in Figure 4, both the EDA_{MLA} and GA_{MLA} remained superior to the one-chromosome GA and TPGA. A primary reason for these results could be the selection of a suitable salesman during the assignment phase according to the MLA rule. Hence, following this rule reduced the maximum traveling distance.

The assignment of 20 salesmen (see Figure 4) caused the lowest maximum loading on a salesman. This is a reasonable result because the loading is distributed over many salesmen. However, the assignment of 20 salesmen also resulted in the longest total traveling distance (see Section 5.1). Hence,

the two objectives present a trade-off and should be considered simultaneously. In Figure 5, it shows the interaction plot between the method and the number of salesmen. This plot indicates EDA_{MLA} and GA_{MLA} perform well no matter how many salesmen are assigned. In addition, the number of salesmen yields the lower maximum traveling distance solved by the four algorithms.

Table 3 shows the complete results for the four algorithms. The EDA_{MLA} and GA_{MLA} are evidently superior to the one-chromosome GA and TPGA. The GA_{MLA} and EDA_{MLA} have 20 and 14 lower mean values, respectively. This phenomenon indicates that the indirect encoding is better than the direct coding approach. The standard deviation of one-chromosome GA, TPGA, GA_{MLA}, and the EDA_{MLA} is 14944, 21728, 13037, and 12940, respectively. StDev indicates that the EDA_{MLA} has less variation than GA_{MLA} and TPGA. The EDA_{MLA} might perform well in the minimization of the maximum traveling distance.

6. Conclusions

This study solves an in-group optimization problem that is rarely solved by EDAs. A new EDA EDA_{MLA}, an EDA combined with the MLA rule, was proposed. Because the MLA rule was classified as transform-based encoding, the proposed algorithm was compared with the TPGA, the most favorable direct encoding strategy thus far. We evaluated these algorithms by solving the mTSP problem for 33 instances drawn from the TSPLIB. The scale of the experiments was larger than those of other mTSP studies. Our experimental results showed that the EDA_{MLA} with the MLA rule outperformed the TPGA for both the objectives of total traveling and maximum traveling distance. Thus, the proposed algorithm is capable of efficiently solving the mTSP problem. In addition, the MLA rule was effective and could be applied with some GAs originally designed for

Table 2: The total average distance of the three algorithms.

Instance	One-chromosome				TPGA				GA$_{MLA}$				EDA$_{MLA}$			
	Min	Mean	Max	StDev	Min	Mean	Max	StDev	Min	Mean	Max	StDev	Min	Mean	Max	StDev
att48	39195	**64157**	104400	20065	47960	88032	142234	29831	42765	73320	119821	26312	42446	73416	121381	26676
berlin52	9355	**12574**	18389	2534	10693	16034	23586	3884	9654	14087	20647	3756	9410	14173	21140	3856
bier127	208466	**253086**	330415	32864	261954	332874	433197	47074	220310	253968	328622	18768	214928	255589	311903	18317
ch130	13488	17246	23859	3019	17602	24176	32313	3785	12781	15893	19262	1644	12020	**15813**	18983	1582
eil101	1021.6	1343.3	1832.4	233.2	1328.1	1829.7	2441.7	326.1	1024.4	1270.4	1625.4	183.7	943.9	**1256.9**	1624.1	197.1
eil51	511.2	**738.2**	1147.4	190.7	614	963.5	1450.8	260.7	497.2	771.2	1195.1	243.6	491.9	772.6	1186	245.7
eil76	721.8	**1036.2**	1457.1	226.1	1035.6	1460.2	2151.2	331.4	771.2	1082.3	1640.5	280.4	760	1087	1610.3	285.4
gr96	862	**1355.4**	2026.5	348.6	1291.9	1972.6	2968.2	519.3	911.3	1396.9	2260.3	405.3	900.2	1396.4	2257.7	421.3
kroa150	70880	89392	121337	14485	91575	124442	164472	18229	62801	78955	104886	8744	60914	**77017**	93631	7330
kroa200	99818	121945	164443	16620	136530	172730	223643	23919	90234	110421	145046	13832	86536	**109347**	141175	12517
kroB100	40570	57373	82441	11490	56782	83671	118140	17817	41476	54609	74861	10195	40070	**54504**	78913	10802
kroB150	69561	90636	122612	15676	98376	128704	171691	20984	62532	79664	93700	8023	63013	**79136**	97548	8422
kroB200	95106	119595	156774	15559	129137	170888	220721	25313	85875	111429	146582	13879	88753	**111071**	142722	12021
kroC100	40573	59920	90359	13754	61044	85942	128239	17762	37895	54879	77495	12239	40453	**54072**	78388	12654
kroD100	36251	57801	85497	12768	55413	85475	129610	19920	40864	56408	82342	12689	40074	**56230**	81408	12490
kroE100	42230	59632	84089	11448	59108	87720	128825	19932	41706	57851	80784	11665	39912	**57714**	85800	12747
lin105	30104	**45726**	71980	12090	41504	70560	107157	19321	30822	48377	79901	14521	30116	48700	77804	15158
lin318	212259	244247	307006	22524	271844	341953	447096	43659	182437	**230828**	311132	38487	183700	231180	318757	39493
mtsp100	42551	58361	87622	12581	57443	85849	123128	19325	42190	56268	81858	12583	39444	**56187**	83089	13184
mtsp150	84018	105130	139427	15595	110196	144405	184927	20654	76189	94674	115786	9230	76317	**93618**	118046	8784
mtsp51	524.7	**744.9**	1107.2	192.9	600.5	960.9	1436.8	265	514.1	771.5	1188	239.1	506.4	776.3	1198.2	241.1
pr124	150254	235652	346625	51913	244964	349702	507277	74210	133373	202203	298646	49945	134543	**200514**	296597	50491
pr136	225315	304074	428602	58108	300700	446351	634173	89691	223163	**286062**	383137	50634	200711	287362	390805	52720
pr144	205204	278907	414337	56813	289835	425899	587282	85838	194333	251995	327206	39986	198892	**250551**	323671	39774
pr152	279221	375123	530527	76952	380235	559885	790537	115996	207562	304803	409260	58957	217451	**302093**	403995	58259
pr226	465922	598166	796733	87652	645486	890751	1223229	146404	329781	**488266**	687780	83403	343841	489202	698623	85134
pr264	285262	373461	479017	37923	385908	527207	698832	78944	172047	230943	311573	33215	173869	**230313**	314209	36013
pr299	241958	295390	381982	33177	343357	432610	585709	60203	218292	272548	337654	28996	220328	**272507**	341877	27987
pr76	164907	**249381**	389365	73419	231932	363181	560341	107747	175830	268514	434977	84973	168656	269631	440088	88178
rat195	6740	**8679**	12038	1491	9573	12711	16873	2107	7200.4	8721.7	10212.1	884.8	7059.3	8739.1	10285.1	887.8
rat99	2149.2	**3295.3**	5008.3	913.2	3148	4989	7707	1417	2270	3672	5940	1198	2279	3651	5924	1221
rd400	82572	91903	104639	4786	101925	114910	135460	9463	53411	**80713**	117869	20490	53204	82324	119892	21221
st70	1023.9	**1487.8**	2264.9	372	1309.4	2163.9	3315.9	609.2	939.3	1594.3	2594	530.9	972.7	1606.4	2641.4	537
tsp225	13022	16756	22477	2559	18365	24630	32360	4086	13615	16592	20431	1548	13417	16775	20124	1553
Average	95930	126303	173878	21187	131434	182519	252133	33230	82825	112163	154056	19785	82557	112010	154332	20041

TABLE 3: Maximum distance of the three compared algorithms.

| Instance	One-chromosome				TPGA				GA_MLA				EDA_MLA			
	Minimum	Mean	Maximum	StDev	Min	Mean	Max	StDev	Min	Mean	Max	StDev	Min	Mean	Max	StDev
att48	13668	16178	23013	2972	13678	18440	31915	4870	13668	**14650**	18185	1489	13668	14687	19252	1570
berlin52	2440	3267.4	5506	913.3	2440	3671	6885	1253	2440	2850.3	4357.3	566.9	2440	**2838.5**	4020.1	543.1
bier127	28890	68194	127487	32040	34697	88175	160235	42537	24007	50498	105182	29541	24007	**50441**	107861	29014
ch130	2119	4685	8530	2088	2472	6288	11699	2968	1177	3113	6777	2002	1177	**3112**	6718	1962
eil101	156	349.3	621	145.4	178	438.4	811	201.2	107.2	**234.3**	504.2	132.3	108	236.2	505	130.5
eil51	109	173.23	301	54.08	109	195.7	352	76.63	108.89	137.98	245.88	39.32	108.89	**137.24**	238.07	38.33
eli76	140	261.84	451	99.51	141	318.8	576	141	124.28	**189.96**	369.26	79.58	124.42	190.91	350.25	79.71
gr96	181	341.1	572	120.4	204	434.3	834	182.4	169.74	**249.33**	479.74	92.83	169.74	253.24	461.39	97.9
kroa150	10877	24701	43153	11241	12890	33802	65152	16075	5395	**15378**	34924	10129	5395	15389	34281	9933
kroa200	13968	34060	63613	16286	18234	47219	85907	22693	6221	**21580**	48028	15050	6271	21647	48262	15043
kroB100	7492	15510	27325	6333	8623	19895	35162	8710	6699	10503	20761	4546	6699	**10480**	19748	4563
kroB150	11170	24615	43288	11044	13835	33480	66963	15396	5750	15271	36168	9535	5750	**15220**	32456	9289
kroB200	14455	33215	60208	15652	17701	46330	89048	22545	6698	21760	50222	15170	6697	**21740**	51004	14895
kroC100	7686	15738	27000	6322	8456	20534	39549	9018	5750	10031	20252	4830	5750	**9858**	19638	4549
kroD100	8035	15495	26648	6166	8676	20215	37334	8865	6357	**10374**	20636	4679	6357	10395	19507	4606
kroE100	8321	15860	27766	6208	8896	20891	36552	8846	7038	**10946**	20938	4631	7038	11058	21363	4766
lin105	6687	11982	21810	4386	7334	15608	27895	6384	6375	**8802**	15930	3060	6375	8816	15539	3017
lin318	26565	65155	117892	32195	34679	91301	170309	45253	10175	**44867**	106246	33540	10266	45772	107495	34536
mtsp100	7805	15625	28178	6230	8670	20291	36571	8799	6357	10519	22419	4920	6357	**10380**	20549	4554
mtsp150	12427	28944	51754	13383	14710	38711	71653	18645	5352	18495	42674	12571	5306	**18188**	39861	12141
mtsp51	110	171.37	278	53.43	109	194.65	367	73.71	108.89	**136.71**	214.08	36.13	108.89	138.23	238.87	38.52
pr124	31977	65199	119940	26367	38626	89080	179180	40189	22594	**38131**	77415	19053	22594	38281	75941	18940
pr136	38778	81809	146217	34977	47037	110777	209589	51156	25731	53626	113915	30275	25731	**53285**	120554	29799
pr144	37745	79364	147984	33520	46297	108344	196336	49131	24313	49939	115229	29812	24313	**49548**	107708	28633
pr152	48531	101819	178569	41889	59013	142063	271591	63667	31727	**60579**	131923	34854	31727	61139	133314	34801
pr226	75797	169009	297280	76223	91773	243451	454610	114227	34845	101101	248381	73633	34845	**100943**	250000	71767
pr264	42743	101884	188189	46735	57515	149315	278195	69900	16339	**46435**	114410	32724	16524	46883	117359	33250
pr299	33418	78941	144137	38053	42242	113477	208744	53805	14293	**51391**	114725	36170	14344	51988	118493	36699
pr76	37970	61320	102253	20609	38692	77148	135311	30936	37971	**48125**	82033	13526	37971	48825	83361	14041
rat195	1032	2319.6	4174	1040.4	1271	3201	6023	1518	604.2	1595.2	3553	990.7	605	**1593.6**	3554.1	984.7
rat99	466	838.9	1491	299.2	503	1069.9	2007	442	432.4	**625.3**	1215.9	230.8	432.9	625.8	1120.4	227.1
rd400	9050	23770	43172	12268	12312	33286	62312	17169	2970	**16509**	39917	13220	2903	16603	39024	13265
st70	209	368	624	120.3	221	448.8	817	178.6	206.4	**275.81**	481.14	86.41	207.4	276.8	492.14	88.51
tsp225	1971	4479	7780	2066	2521	6188	10987	2891	998	3068	6843	2045	998	3130	7060	2091
Average	15970	34284	61388	14944	19258	47185	87984	21728	9797	21823	47810	13037	9805	21885	47857	12940
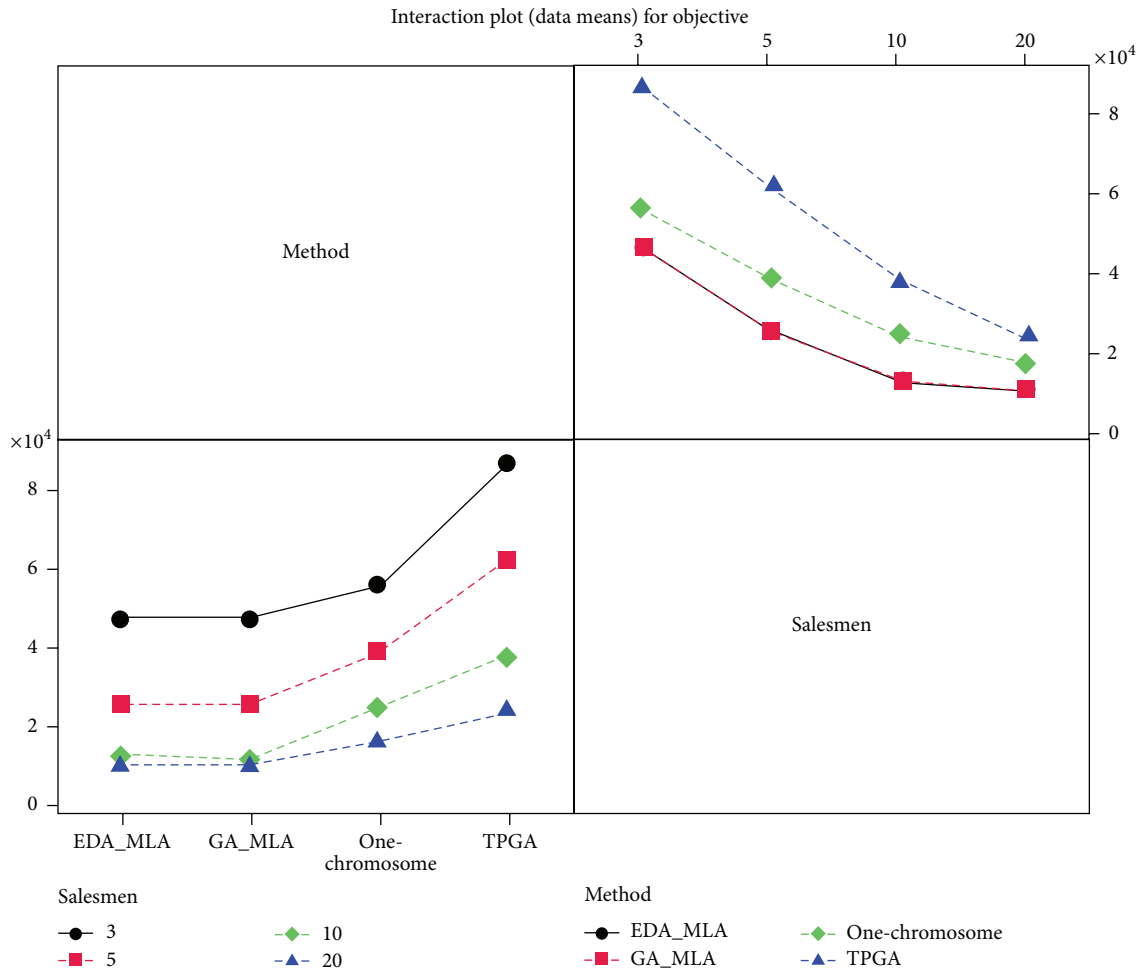
FIGURE 5: Interaction plot of the maximum distance for the compared algorithms.

permutation-type problems. As a result, this study provides insight for researchers investigating scheduling problems and advances the research on in-group optimization problems.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

References

[1] L. Ke, Q. Zhang, and R. Battiti, "MOEA/D-ACO: a multiobjective evolutionary algorithm using decomposition and Ant-Colony," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1845–1859, 2013.

[2] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano, "A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems," *Progress in Artificial Intelligence*, vol. 1, no. 1, pp. 103–117, 2012.

[3] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111–128, 2011.

[4] Q.-K. Pan and R. Ruiz, "An estimation of distribution algorithm for lot-streaming flow shop problems with setup times," *Omega*, vol. 40, no. 2, pp. 166–180, 2012.

[5] A. Zhou, F. Gao, and G. Zhang, "A decomposition based estimation of distribution algorithm for multiobjective traveling salesman problems," *Computers & Mathematics with Applications*, vol. 66, no. 10, pp. 1857–1868, 2013.

[6] Q. Zhang, J. Sun, and E. Tsang, "An evolutionary algorithm with guided mutation for the maximum clique problem," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 2, pp. 192–200, 2005.

[7] S.-H. Chen, M.-C. Chen, P.-C. Chang, Q. Zhang, and Y.-M. Chen, "Guidelines for developing effective Estimation of Distribution Algorithms in solving single machine scheduling problems," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6441–6451, 2010.

[8] Y. Zhang and X. Li, "Estimation of distribution algorithm for permutation flow shops with total flowtime minimization," *Computers & Industrial Engineering*, vol. 60, no. 4, pp. 706–718, 2011.

[9] S.-H. Chen and M.-C. Chen, "Addressing the advantages of using ensemble probabilistic models in Estimation of Distribution Algorithms for scheduling problems," *International Journal of Production Economics*, vol. 141, no. 1, pp. 24–33, 2013.

[10] V. A. Shim, K. C. Tan, and C. Y. Cheong, "A hybrid estimation of distribution algorithm with decomposition for solving the multiobjective multiple traveling salesman problem," *IEEE Transactions on Systems, Man and Cybernetics C: Applications and Reviews*, vol. 42, no. 5, pp. 682–691, 2012.

[11] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

[12] S.-H. Chen, P.-C. Chang, T. C. E. Cheng, and Q. Zhang, "A self-guided genetic algorithm for permutation flowshop scheduling problems," *Computers & Operations Research*, vol. 39, no. 7, pp. 1450–1457, 2012.

[13] A. E. Carter and C. T. Ragsdale, "A new approach to solving the multiple traveling salesperson problem using genetic algorithms," *European Journal of Operational Research*, vol. 175, no. 1, pp. 246–257, 2006.

[14] B. Gavish and K. Srikanth, "An optimal solution method for large-scale multiple traveling salesmen problems," *Operations Research*, vol. 34, no. 5, pp. 698–717, 1986.

[15] G. Laporte and Y. Nobert, "A cutting planes algorithm for the *m*-salesmen problem," *Operational Research Society Journal*, vol. 31, no. 11, pp. 1017–1023, 1980.

[16] D. L. Miller and J. F. Pekny, "Exact solution of large asymmetric traveling salesman problems," *Science*, vol. 251, no. 4995, pp. 754–761, 1991.

[17] L. Tang, J. Liu, A. Rong, and Z. Yang, "A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron & Steel Complex," *European Journal of Operational Research*, vol. 124, no. 2, pp. 267–282, 2000.

[18] C. J. Malmborg, "A genetic algorithm for service level based vehicle scheduling," *European Journal of Operational Research*, vol. 93, no. 1, pp. 121–134, 1996.

[19] Y.-B. Park, "A hybrid genetic algorithm for the vehicle scheduling problem with due times and time deadlines," *International Journal of Production Economics*, vol. 73, no. 2, pp. 175–188, 2001.

[20] E. C. Brown, C. T. Ragsdale, and A. E. Carter, "A grouping genetic algorithm for the multiple traveling salesperson problem," *International Journal of Information Technology and Decision Making*, vol. 6, no. 2, pp. 333–347, 2007.

[21] A. Király and J. Abonyi, "A novel approach to solve multiple traveling sales-men problem by genetic algorithm," in *Computational Intelligence in Engineering*, pp. 141–151, Springer, 2010.

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

[23] J.-P. Arnaout, G. Rabadi, and R. Musa, "A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times," *Journal of Intelligent Manufacturing*, vol. 21, no. 6, pp. 693–701, 2010.

[24] S.-H. Chen and M.-C. Chen, "Operators of the two-part encoding genetic algorithm in solving the multiple traveling salesmen problem," in *Proceedings of the 16th Annual Conference on Technologies and Applications of Artificial Intelligence (TAAI '11)*, pp. 331–336, November 2011.

[25] E. Vallada and R. Ruiz, "A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times," *European Journal of Operational Research*, vol. 211, no. 3, pp. 612–622, 2011.

[26] A. H. Kashan, M. H. Kashan, and S. Karimiyan, "A particle swarm optimizer for grouping problems," *Information Sciences*, vol. 252, pp. 81–95, 2013.

[27] T. W. Liao, P. C. Chang, R. J. Kuo, and C.-J. Liao, "A comparison of five hybrid metaheuristic algorithms for unrelated parallel-machine scheduling and inbound trucks sequencing in multi-door cross docking systems," *Applied Soft Computing*, vol. 21, pp. 180–193, 2014.

[28] R. Ruiz and C. Maroto, "A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility," *European Journal of Operational Research*, vol. 169, no. 3, pp. 781–800, 2006.

[29] P.-C. Chang and S.-H. Chen, "Integrating dominance properties with genetic algorithms for parallel machine scheduling problems with setup times," *Applied Soft Computing*, vol. 11, no. 1, pp. 1263–1274, 2011.

[30] S.-Y. Wang, L. Wang, M. Liu, and Y. Xu, "An effective estimation of distribution algorithm for solving the distributed permutation flow-shop scheduling problem," *International Journal of Production Economics*, vol. 145, no. 1, pp. 387–396, 2013.

[31] A. Salhi, J. A. V. Rodríguez, and Q. Zhang, "An estimation of distribution algorithm with guided mutation for a complex flow shop scheduling problem," in *Proceedings of the 9th Annual Genetic and Evolutionary Computation Conference (GECCO '07)*, pp. 570–576, ACM, July 2007.

[32] M. Pelikan, D. E. Goldberg, and F. G. Lobo, "A survey of optimization by building and using probabilistic models," *Computational Optimization and Applications*, vol. 21, no. 1, pp. 5–20, 2002.

[33] J. A. Lozano, P. Larranaga, and I. Inza, *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms*, Springer, Berlin, Germany, 2006.

[34] P.-C. Chang, S.-S. Chen, and C.-Y. Fan, "Mining gene structures to inject artificial chromosomes for genetic algorithm in single machine scheduling problems," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 767–777, 2008.

[35] J. L. Shapiro, "Diversity loss in general estimation of distribution algorithms," in *Parallel Problem Solving from Nature—PPSN IX*, vol. 4193 of *Lecture Notes in Computer Science*, pp. 92–101, Springer, Berlin, Germany, 2006.

[36] J. Branke, C. Lode, and J. L. Shapiro, "Addressing sampling errors and diversity loss in UMDA," in *Proceedings of the 9th Annual Genetic and Evolutionary Computation Conference (GECCO '07)*, pp. 508–515, ACM, July 2007.

[37] B. Jarboui, M. Eddaly, and P. Siarry, "An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems," *Computers and Operations Research*, vol. 36, no. 9, pp. 2638–2646, 2009.

[38] P. Hansen and N. Mladenovi, "Variable neighborhood search: principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449–467, 2001.

[39] S. Chen, P. Chang, Q. Zhang, and C. Wang, "A guided memetic algorithm with probabilistic models," *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 12, pp. 4753–4764, 2009.

[40] S. Chen, P. C. Chang, M. C. Chen, and Y. M. Chen, "A self-guided genetic algorithm with dominance properties for single machine scheduling problems," in *Proceedings of the IEEE Symposium on Computational Intelligence in Scheduling (CI-Sched '09)*, pp. 76–83, IEEE, Nashville, Tenn, USA, April 2009.

[41] S.-H. Chen, P.-C. Chang, T. C. E. Cheng, and Q. Zhang, "A self-guided genetic algorithm for permutation flowshop scheduling problems," *Computers and Operations Research*, vol. 39, no. 7, pp. 1450–1457, 2012.

[42] I. Good, *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*, The MIT Press, 2003.

[43] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006.

[44] B. Cestnik, "Estimating probabilities: a crucial task in machine learning," in *Proceedings of the 9th European Conference on Artificial Intelligence*, pp. 147–149, 1990.

[45] S. Baluja and S. Davies, "Fast probabilistic modeling for combinatorial optimization," in *Proceedings of the 15th National Conference on Artificial Intelligence*, pp. 469–476, John Wiley & Sons Inc, July 1998.

[46] Y. Chan and S. F. Baker, "The multiple depot, multiple traveling salesmen facility-location problem: vehicle range, service frequency, and heuristic implementations," *Mathematical and Computer Modelling*, vol. 41, no. 8-9, pp. 1035–1053, 2005.

[47] S. Ghafurian and N. Javadian, "An ant colony algorithm for solving fixed destination multi-depot multiple traveling salesmen problems," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 1256–1262, 2011.

[48] S. Yadlapalli, W. A. Malik, S. Darbha, and M. Pachter, "A Lagrangian-based algorithm for a multiple depot, multiple traveling salesmen problem," *Nonlinear Analysis: Real World Applications*, vol. 10, no. 4, pp. 1990–1999, 2009.

[49] S. Gorenstein, "Printing press scheduling for multi-edition periodicals," *Management Science*, vol. 16, no. 6, pp. 373–383, 1970.

[50] A. E. Carter and C. T. Ragsdale, "Scheduling pre-printed newspaper advertising inserts using genetic algorithms," *Omega*, vol. 30, no. 6, pp. 415–421, 2002.

[51] J. A. Svestka and V. E. Huckfeldt, "Computational experience with an m-salesman traveling salesman algorithm," *Management Science*, vol. 19, no. 7, pp. 790–799, 1973.

[52] J. K. Lenstra and A. H. G. R. Kan, "Some simple applications of the travelling salesman problem," *Operational Research Quarterly*, vol. 26, no. 4, pp. 717–733, 1975.

[53] T. Zhang, W. A. Gruver, and M. H. Smith, "Team scheduling by genetic search," in *Proceedings of the 2nd International Conference on Intelligent Processing and Manufacturing of Materials (IPMM '99)*, vol. 2, pp. 839–844, Honolulu, Hawaii, USA, 1999.

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

[55] C. Okonjo-Adigwe, "An effective method of balancing the workload amongst salesmen," *Omega*, vol. 16, no. 2, pp. 159–163, 1988.

[56] R. Wolfler Calvo and R. Cordone, "A heuristic approach to the overnight security service problem," *Computers & Operations Research*, vol. 30, no. 9, pp. 1269–1287, 2003.

[57] R. D. Angel, W. L. Caudle, R. Noonan, and A. Whinston, "Computer-assisted school bus scheduling," *Management Science*, vol. 18, no. 6, pp. 279–288, 1972.

[58] K. H. Kim and Y.-M. Park, "A crane scheduling method for port container terminals," *European Journal of Operational Research*, vol. 156, no. 3, pp. 752–768, 2004.

[59] X. Wang and A. C. Regan, "Local truckload pickup and delivery with hard time window constraints," *Transportation Research Part B: Methodological*, vol. 36, no. 2, pp. 97–112, 2002.

[60] X. Gan, Y. Wang, S. Li, and B. Niu, "Vehicle routing problem with time windows and simultaneous delivery and pick-up service based on MCPSO," *Mathematical Problems in Engineering*, vol. 2012, Article ID 104279, 11 pages, 2012.

[61] X. Gan, Y. Wang, J. Kuang, Y. Yu, and B. Niu, "Emergency vehicle scheduling problem with time utility in disasters," *Mathematical Problems in Engineering*, vol. 2015, Article ID 164194, 7 pages, 2015.

[62] B. Brumitt and A. Stentz, "Dynamic mission planning for multiple mobile robots," *Intelligent Unmanned Ground Vehicles*, vol. 388, pp. 221–234, 1997.

[63] B. L. Brumitt and A. Stentz, "GRAMMPS: a generalized mission planner for multiple mobile robots in unstructured environments," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1564–1571, IEEE, May 1998.

[64] Z. Yu, L. Jinhai, G. Guochang, Z. Rubo, and Y. Haiyan, "An implementation of evolutionary computation for path planning of cooperative mobile robots," in *Proceedings of the 4th World Congress on Intelligent Control and Automation*, vol. 3, pp. 1798–1802, IEEE, 2002.

[65] A. Basu, A. Elnagar, and R. Al-Hajj, "Efficient coordinated motion," *Mathematical and Computer Modelling*, vol. 31, no. 2-3, pp. 39–53, 2000.

[66] J. L. Ryan, T. G. Bailey, J. T. Moore, and W. B. Carlton, "Reactive Tabu Search in unmanned aerial reconnaissance simulations," in *Proceedings of the 30th Conference on Winter Simulation*, pp. 873–880, IEEE Computer Society Press, December 1998.

[67] H. A. Saleh and R. Chelouah, "The design of the global navigation satellite system surveying networks using genetic algorithms," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 1, pp. 111–122, 2004.