Computer Science Journal of Moldova, vol.**27**, no.3(81), 2019

A new multi-offspring crossover operator for genetic algorithm to facilitate the traveling salesman problem

Abid Hussain, Salman A. Cheema

Abstract

This research work provides a detailed working principle and official analysis of a multi-offspring crossover operator. The proposed operator explains the true theory of survival-of-fittest using the foundation of evolutionary theories of biology and ecological theories of mathematics. We found a considerable improvement because the proposed operator enhances the opportunity of having better offspring, which results in highly competitive population. Simulation results of this operator with other competitor crossover operators for one of the combinatorial optimization problems, i.e. traveling salesman problem, are obviously showing its pros at better accuracy level. Moreover, the t-test and performance index (PI) establishes the improved significance and accuracy levels of the proposed operator. Preferable results of this operator not only confirm its advantages over the others, but also show long run survival of a generation having a number of offspring more than the number of parents with the help of mathematical ecology theory.

Keywords: Traveling salesman problem, genetic algorithms, crossover operators, multi-offspring, performance index.

1 Introduction

The traveling salesman problem (TSP) is one of the most famous benchmark, significant, historic and very important combinatorial optimization problem [1]. TSP was documented by Euler in 1759, his interest

©2019 by CSJM; Abid Hussain, Salman A. Cheema

was how to get rid of the knight's tour problem [2]. It is the fundamental problem in the fields of computer science, engineering, operations research, discrete mathematics and graph theory etc. To find the shortest tour that visits each city by a salesman in a given list exactly once and then returns to the starting city is the required goal. TSP has extremely large search spaces and is very difficult to solve, so it is called a typical non-deterministic polynomial (NP-hard) problem [3]–[5]. The given n cities, a distance matrix $C = [c_{ij}]_{n\times n}$ is searched for a permutation $\lambda : \{0, ..., n-1\} \longrightarrow \{0, ..., n-1\}$, where c_{ij} is the distance from city i to city j, which minimizes the traveled distance, $f(\lambda, C)$:

$$f(\lambda, C) = \sum_{i=0}^{n-1} d(c_{\lambda(i)}, c_{\lambda(i+1)}) + d(c_{\lambda(n)}, c_{\lambda(1)}), \tag{1}$$

where $\lambda(i)$ represents the location of city i in each tour, $d(c_i, c_j)$ is the distance between city i and city j and (x_i, x_j) is a specified position of each city in a tour in the plane, and the Euclidean distances of the distance matrix C between the city i and the city j are expressed as:

$$c_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \tag{2}$$

For n cities, there are $(n-1)!$ possible ways to find the tour after fixing the starting city for asymmetric distances, i.e. $c_{ij} \neq c_{ji}$ and its half for symmetric ones, i.e. $c_{ij} = c_{ji}$. This type of problems cannot be solved using traditional optimization approaches like gradient-based methods. To achieve the optimal solution within a reasonable amount of time, heuristic approaches are efficient at handling the NP-hard problems [6]. Apart from its theoretical approach, TSP is widely used as a model in many fields such as vehicle routing problem [7], physical mapping problems [8], constructing phylogenetic trees [9], machine flow shop scheduling [10] and so on.

TSP is a very carefully studied problem and received considerable attention over the last few decades. A lot of algorithms have been presented to solve this problem by researchers. These algorithms are generally divided into two classes: exact and approximate algorithms. The

branch-and-bound [11] and cutting planes [12] are the two examples of exact algorithms which are excessively time consuming especially in large scale problems. Approximate algorithms are further classified into heuristic and meta-heuristic algorithms. There are three classes of the heuristic algorithm: tour construction, tour improvement and composite methods. To add an unvisited city to the solution at each step and try to shorten the initial solution are tour construction and tour improvement methods respectively. Finally, the composite method is the combination of these two algorithms. In the last three decades, it has started a new stage of study about optimization problems with the appearance of meta-heuristic algorithms. These search algorithms have also been applied on TSP; 2-opt [6], particle swarm optimization [13], simulated annealing [14], ant colony optimization [15], neural network [16], tabu search [17], and genetic algorithms (GAs) [18]–[21].

We used genetic algorithm (GA) in this research to solve the TSP. In the literature of GAs, the pair of parents generated the pair of offspring. For a long run survival and diversity of species, it should be desirable to create more offspring than parents. In this research, we provide a multi-offspring crossover operator to handle this problem and give a detail discussion how this can be done. The simulation results with benchmarks show that the proposed operator indeed generates improved performance as compared to other traditional operators in the TSP application.

The rest of this article is presented as follows: in Section 2, we present the overview of GAs for TSP. Theoretical background with implementation of the proposed operator is presented in Section 3. The performance evaluation based on simulation study and conclusion are in Sections 4 and 5 respectively.

2 Genetic algorithms for TSP

Genetic algorithms (GAs) are derivative free stochastic approaches which are based on biological evolutionary processes proposed by John Holland [22]. GA is used to search the optimal or near to optimal solutions of various optimization problems. It is one of the main branches of

evolutionary algorithms, where the natural process from biology discipline is mimicked which was originally described by Darwin [23]. GAs are iterative processes and iterations are called generations [24]. In 1985, GAs were first time used to solve TSP by Goldberg [25]. A simple and pure GA for TSP can be defined in the following subsections.

2.1 Chromosome representation

In literature, there have been various representations to encode feasible solutions to solve the TSP using the GAs. Some of them are binary, path, adjacency, real and matrix representations of the chromosomes. A detailed study of these approaches is presented by Larranaga et al. [2]. Other than the path, all techniques have complex nature to complete a legal tour for the next generation. A path representation is desired because it is the most natural and elaborated way to represent a tour. For example, a nine-city tour $7 \longrightarrow 2 \longrightarrow 6 \longrightarrow 1 \longrightarrow 4 \longrightarrow 8 \longrightarrow 3 \longrightarrow 9 \longrightarrow 5$ can be represented simply as (7 2 6 1 4 8 3 9 5), and the total distance of this path is

$$D_{72} + D_{26} + D_{61} + D_{14} + D_{48} + D_{83} + D_{39} + D_{95} + D_{57}$$

2.2 Initial population

A traveling track in the population can be represented by the sequence of all cities, and an array of integers is used to indicate the sequence of cities. For example, there is a TSP with nine cities so all 9! tracks are non-repeating integers from 1 to 9 (both included), such as 7 2 6 1 4 8 3 9 5 is one of them. More generally, if a TSP having 'm' cities and 'n' is the population size, it means we generate 'n' randomly chosen sequences with each of them containing 'm' non-repeating integers.

2.3 Fitness evaluation

The evaluation of individuals in a population is based upon some specific fitness criteria. It turns from simple to complex if more and more parameters are added to evaluate an individual. The easiest way to

evaluate the fitness of an individual is summing up the lengths of its track [26],[27]. This kind of fitness evaluator is used if all solutions are feasible and no other constraints are given (like the number of routes, the maximal length of the route and similar).

2.4 Selection operator

Selection scheme is a procedure in which individuals are selected on the base of their fitness for mating. There are several methods usually used for the TSP and these are equally useful for their respective problems. The commonly used methods are tournament [28], roulette wheel [22] and rank based [29] methods. Among the above methods, the most popular one is tournament, where randomly selected individuals compete with one another and the best one is picked for the next phase. Generally there are two types of tournament methods, i.e. binary and more than two individuals competing at the same time [30]. We have used binary tournament selection (BTS) in this research, where two candidates are selected at random and the better of them is nominated as a parent. For t-tournament selection, the probability after rank 1 to the best individual can be defined as:

$$P_i = N^{-t}((N - i + 1)^t - (N - i)^t), \tag{3}$$

where $i \in \{1, 2, ...N\}$, N and t are the population and the tournament sizes respectively. A roulette wheel selection (RWS) is also used in this study to choose the individuals for mating process. For this purpose, first we calculate the fitness of all individuals using the following rule:

$$f_i = \beta(1 - \beta)^i; \quad i = 1, 2, ..., N,$$

where f_i is the i^{th} individual of sorted population in ascending order, and $\beta \in (0, 1)$ and generally suitable within the range of 0.01 to 0.3 [31]. After this, we assign the selection probability of i^{th} individual to be calculated as follows:

$$P_i = \frac{f_i}{\sum_{i=1}^{N} f_i}. \tag{4}$$

The next selection approach which we have used in this study is the linear ranked selection (LRS). This LRS is an alternative approach of RWS, when fitness values are far away from each other in a population. The selection probability is linearly assigned to the individuals after rank 1 to the worst individual as:

$$p_i = \frac{1}{N}(\eta^- + (\eta^+ - \eta^-)\frac{i-1}{N-1}); \qquad i \in \{1, 2, ..., N\}. \tag{5}$$

Here $\frac{\eta^-}{N}$ and $\frac{\eta^+}{N}$ are the probabilities of the worst and the best chromosomes to be selected respectively. All the individuals get a different rank even if they have same fitness value. The conditions $\eta^+ = 2 - \eta^-$ and $\eta^- \geq 0$ must be fulfilled. All the above mentioned selection procedures have been used in various applications, see for example [32].

2.5 Crossover operator

Crossover is the process of mating selected candidates for the purpose of producing further offspring. Since these offspring share the characteristics of their parents, hence there must be a rule to make a combination of these characteristics. The type of crossover is strongly dependent on the genotype representation of the population. In TSP, the most used method is path-based crossover technique. To generate offspring from a combination of the pair of paths, any random point is selected in such a way that it takes the first portion from one parent and the second portion from the other parent. This approach is called a 1-point crossover. As a result, there may emerge some infeasible solutions which can be avoided by putting a constraint that only feasible solutions can enter into coming generation [33]. Whereas in 2-point crossover (2PX), the expansion proceeds with a random selection of two points. Since the middle portion is exchanged here [34],[35], hence it may also become infeasible. To overcome this problem, Choi et al. [26] randomly insert missing bits in the middle portion. Another approach to counter this problem is a Partially-mapped crossover (PMX), where the indexes are exchanged within the cut-points portion and missing bits are replaced with their mapping index in the other parent [25]. The order crossover

(OX) is the next more commonly used operator of 2PX, in which the central part (from point a to point b, where $a < b$) is taken from one parent and other parent is selected from $b+1$ point circularly onward to complete the legal offspring [36]. The existing crossover approaches are less useful because they don't take the important nodes into account. It means that some tracks may be divided in half and others may not even be copied. This can dramatically affect in the new generation by generating new random tracks irrespective of the good ones. Next crossover operation is named cycle crossover(CX), which is unusual in its nature and detects a set of cycles between two selected candidates in their chromosomes and later it duplicates these cycles one by one to offspring [37].

The multi-offspring genetic algorithm (MO-GA) was proposed by Wang et al. [31]. It builds offspring preserving the relative order of bits of one parent by choosing a sub-tour of other parent. First of all the parent's track is divided into three regions after applying two random cut points. This scheme produces four offspring in two stages. In the first stage, its work is similar as OX, where the middle portion (from point a to point b, where $a < b$) is taken from one parent and the other parent is selected from $b + 1$ point circularly onward to complete the legal offspring. In the second stage, exchange regions 1 and 2 or simply de-track the original tracks of parents. Find region 3 elements from the first new parent, delete the corresponding ones in the second new parent, and vice versa. After this the regions 3 of both parents are replaced with each other for producing offspring. We also proposed a new multi-offspring crossover operator in this study which is presented in Section 4.

2.6 Mutation operator

Mutation operator expands the search space by a small degree. It is rare happening in nature and similar as that in GAs, where the mutation probability factor is predefined by the researchers. Since mutation causes extraordinary results to genetic principle, hence its factor must be small. It is the simplest operator in GAs which can be used in

various ways. A fractional change in genotype may result in a reasonable change in the next generation. Dealing with TSP, the simplest mutation can be completed by random exchange of any two cities in a route as an exchange mutation (EM) [38]. The other approach is insertion mutation (IM) which shifts the selected city from its original place and inserts it on another random place [39],[40]. Another popular approach is inversion mutation (INM), where a specific sub-track of a chromosome is inverted [39],[41]. If only two cities in a chromosome are inverted, then it is called the swap mutation (SM) [42].

2.7 Termination criteria

Generally, the termination criteria vary with the nature of research. It may be time dependent process as done with evolving the limited time only 1800 seconds [43]. Another option is the limitation of generations as done [27] with maximum of 1000 while [44] used up to 50,000 generations. The other significant process of evaluation is terminating when candidates stay the same for 30% of the generation limit [45]. Researchers [44] stopped the evolution process when almost 95% of the population is converged to one unique solution. This research uses two criteria to stop the process, i.e. a tour shorter than the current optimum trip has not been found during 300 consecutive iterations otherwise to complete the process with a maximum number of generations, i.e. 5000.

3 Theoretical background of MO-OX

3.1 Biological theory foundation

In the literature, the GA for TSP explains a pair of parents producing a pair of offspring [21],[46]. However the biological evolution proceeds in such a way that it generates equal or less than a pair of parents, which is uncommon. Since having the risk of extinction the generation or being less competitive, it is desirable to have the number of offspring more than two in genetic. When the number of offspring is larger than

2 from two better parents, there is more competition among them to survive, with increases chances of better offspring.

3.2 Mathematical ecological theory foundation

The probability of extinction of species can be illustrated by supposing that a pair of species has only two offspring at first. The probability of population size that leads to "0" at a certain time "t" can be defined as:

$$P_0(t|i=1) = \frac{\eta\ exp((\theta-\eta)t) - \eta}{\theta\ exp((\theta-\eta)t) - \eta}, \tag{6}$$

where i denotes the size of the initial population, η and θ are the mortality and fertility rates respectively. This population can be extinct with time elapsing by the following probability:

$$P_0(t) = [P_0(t|i=1)]^i = [\frac{\eta\ exp((\theta-\eta)t) - \eta}{\theta\ exp((\theta-\eta)t) - \eta}]^i. \tag{7}$$

As t$\longrightarrow \infty$, three scenarios of equation (7) can be observed:

(1) When the mortality rate equals the fertility rate, i.e. $\eta = \theta$, equation (7) is expressed as a series of exponential terms. Letting $\theta - \eta = r$, then $P_0(t)$ as t$\longrightarrow \infty$ should be

$$P_0(t) = [\frac{\eta(1 + rt + r^2t^2/2! + ...) - \eta}{\theta(1 + rt + r^2t^2/2! + ...) - \eta}]^i. \tag{8}$$

If η equals to θ, then r$\longrightarrow$0, so we are eliminating r^2 and higher terms and get the following expression:

$$P_0(t) \longrightarrow [\frac{\eta rt}{(\theta + \theta rt) - \eta}]^i \longrightarrow [\frac{\eta rt}{(\theta - \eta) + \theta rt}]^i \longrightarrow [\frac{\eta rt}{(r + \theta rt}]^i. \tag{9}$$

Hence,

$$\lim_{t\to\infty} [\frac{\theta t}{1 + \theta t}]^i = 1. \tag{10}$$

It means that the species should become extinct with 100% probability (shown in equation (10)), even when the mortality rate equals to the fertility rate.

(2) If the mortality rate is greater than fertility rate in each generation, i.e. $\eta > \theta$. Letting $\theta - \eta = -r$, then the exponential terms of equation (7) would be "0" when t$\longrightarrow \infty$ and we get:

$$P_0(t) \longrightarrow [\frac{\eta\ exp(-rt) - \eta}{\theta\ exp(-rt) - \eta}]^i, \tag{11}$$

and finally it leads to

$$\lim_{t\to\infty}(P_0(t)) = 1.$$

Hence, in this scenario with growing the time, species should become extinct.

(3) If the mortality rate is less than fertility rate in each generation, i.e. $\eta < \theta$, the equation (7) as t$\longrightarrow \infty$ can be expressed as follows:

$$P_0(t) \longrightarrow [\frac{\eta\ exp(t) - \eta}{\theta\ exp(t) - \eta}]^i. \tag{12}$$

Hence,

$$\lim_{t\to\infty}(P_0(t)) = [\frac{\eta}{\theta}]^i.$$

According to equation (12), there is no guarantee that such a population will never become extinct because a finite probability exists for the extinction. However, if the mortality rate is much lower than the fertility rate, then we have the least probability of biological extinction.

According to mathematical ecological theory in biology, if $\theta = \eta$ for a certain population, the probability of population extinction is 1. Thus, the probability distribution of biological population size depends on the product of fertility rate and time when the biological initial population is known. Hence, it is important to improve the fertility rate of species to get better individuals in possible shortest time.

3.3 Multi-offspring crossover procedure

The new extension involves producing four offspring, two from each technique of direct ordering, as well as the other one being from revers ordering of the parents tracks, so we suggested multi-offspring order crossover (MO-OX). Consider an example of the two parents tours (with randomly two cut points marked by "|"):

$$P_1 = (9 \quad 4 \quad 5 \mid 2 \quad 8 \quad 1 \mid 6 \quad 7 \quad 3) \quad \textit{and}$$

$$P_2 = (3 \quad 6 \quad 1 \mid 9 \quad 7 \quad 8 \mid 2 \quad 4 \quad 5).$$

The MO-OX is utilized in two stages to generate four offspring. In Stage 1, the crossover operation is as follows:

Stage 1: The offspring are produced in the following way. First, the bits are copied down between the cuts with similar way into the offspring, which gives:

$$O_1 = (\times \quad \times \quad \times \mid 2 \quad 8 \quad 1 \mid \times \quad \times \quad \times) \quad \textit{and}$$

$$O_2 = (\times \quad \times \quad \times \mid 9 \quad 7 \quad 8 \mid \times \quad \times \quad \times).$$

After this, starting from the second cut point of one parent, the bits from the other parent are copied in the same order omitting existing bits. As the sequence of the bits in the second parent from the second cut point is:

$$2 - 4 - 5 - 3 - 6 - 1 - 9 - 7 - 8$$

after removal of bits 2, 8 and 1, which already exist in the first offspring, the new sequence is:

$$4 - 5 - 3 - 6 - 9 - 7.$$

To complete the first offspring, this new sequence is placed starting from the second cut point:

$$O_1 = (6 \quad 9 \quad 7 \mid 2 \quad 8 \quad 1 \mid 4 \quad 5 \quad 3).$$

Analogously, we complete the second offspring as well:

$$O_2 = (5 \quad 2 \quad 1 \mid 9 \quad 7 \quad 8 \mid 6 \quad 3 \quad 4).$$

Stage 2: In Stage 2, the offspring are produced in the following way. First, the bits are copied down between the cuts with similar way into the offspring, which gives:

$$O_1 = (\times \ \times \ \times \mid 2 \ \ 8 \ \ 1 \mid \times \ \times \ \times) \quad \textit{and}$$

$$O_2 = (\times \ \times \ \times \mid 9 \ \ 7 \ \ 8 \mid \times \ \times \ \times).$$

After this, starting from the first cut point of one parent, the bits from the other parent are copied in the reverse order omitting existing bits. The reverse sequence of the bits in the second parent from the first cut point is as follows:

$$1 - 6 - 3 - 5 - 4 - 2 - 8 - 7 - 9.$$

After removal of bits 2, 8 and 1, which are already in the first offspring, the new sequence is:

$$6 - 3 - 5 - 4 - 7 - 9.$$

This sequence is placed in the first offspring starting from the first cut point with reverse sequence as:

$$O_1 = (5 \ \ 3 \ \ 6 \mid 2 \ \ 8 \ \ 1 \mid 9 \ \ 7 \ \ 4).$$

Analogously, we complete the second offspring as well:

$$O_2 = (3 \ \ 4 \ \ 5 \mid 9 \ \ 7 \ \ 8 \mid 2 \ \ 1 \ \ 6).$$

The newer scheme, by combining both OX and ROX can be termed as a multi-offspring order crossover (MO-OX). Hence, we differentiate MO-OX in the Algorithm 1.

Algorithm 1 The Pseudo-code of MO-OX

```
N ⟵ no. of cities
i ⟵ 1
while (i <= N) do
    P1 ⟵ a sequence of the first parent
    P2 ⟵ a sequence of the second parent
    cut1 ⟵ random point at the same location on parents
    cut2 ⟵ another random point at the same location on parents
    offspring1 ⟵ P1(cut1 to cut2)          %child by OX
    offspring2 ⟵ P2(cut1 to cut2)          %child by ROX
    offspring3 ⟵ P1(cut1 to cut2)          %child by RX
    offspring4 ⟵ P2(cut1 to cut2)          %child by ROX
 % For offspring1 in clockwise direction
    i ⟵ cut2, j ⟵ i
    while (length of offspring1 ! = length of P1) do
        if find((offspring1 == P1(i)))     %if value already exists in offspring1
            i = i + 1
        else
            offspring1(j) = P1(i)
            i ⟵ i + 1; j ⟵ j + 1;
        end if
        if (i > length of P1)
            i ⟵ 1
        end if
        if (j > length of P1)
            j ⟵ 1
        end if
    end while
    %Similarly to offspring3 from P2%
 % For offspring2 in anticlockwise direction
    i ⟵ cut1, j ⟵ i
    while (length of offspring2 ! = length of P1) do
        if find ((offspring2 == P1(i)))    %if value already exists in offspring2
            i = i - 1
        else
            offspring2(j) = P1(i)
            i ⟵ i - 1; j ⟵ j - 1;
        end if
        if(i==0)
            i ⟵ length of P1
        end if
        if(j==0)
            j ⟵ length of P1
        end if
    end while
    %Similarly to offspring4 from P2%
end while
```

4 Performance evaluation

4.1 Computational testing methodology

To evaluate the performances of the proposed MO-OX, computational experiments have been performed by using six benchmark instances which are taken from the traveling salesman problem library (TSPLIB) [47]. The test benchmarks are Euclidean, two-dimensional symmetric and asymmetric problems with 42, 53, 171, 180, 443, and 532 cities. In our simulation experiments, all GA programs were implemented in MATLAB version R2015a. Table 1 shows the state-of-the-art parametric configuration of the GA, and complete methodology for all the experiments performed in this research is shown in Figure 1. Moreover, we used two stopping criteria for our simulation experiments, i.e. attaining the maximum number of generations and if the tour, shorter than the current optimal tour, is not being found during the last 300 consecutive generations.

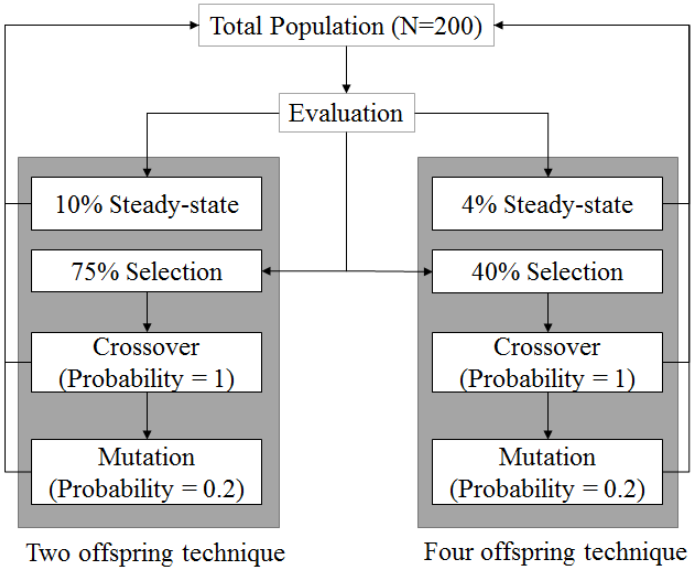

Figure 1. The methodology flow chart for simulation study

Table 1. Parametric configuration for GA

Parameter	Value
Population size	200
Selection criteria	BTS, RWS, RBS
Crossover rate	100%
Mutation method	EM, INM
Mutation rate	20%
Maximum generation	5000
Replacement in GA	Steady-state GA

Since GAs belong to the class of stochastic search algorithms, we employ statistical hypothesis testing using the two independent samples t-test. The equation (13) presents the general expression for the applied t-test such as:

$$t = \frac{\bar{x_1} - \bar{x_2}}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}, \tag{13}$$

where,

$$s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2},$$

and $\bar{x_1}$ and $\bar{x_2}$ are the averages of 30 trails for the proposed and rival operators respectively. Similarly, s_1^2 and s_2^2 are standard deviations (S.D) of the proposed and contemporary method, respectively. A statistical significant difference is then quantified at $\alpha = 0.05$ (95% confidence) level of significance, under the null hypothesis stating that "MO-OX and the selected operator converge equally fast under given settings".

We also used the performance index (PI) criteria which is widely used to compare the newly proposed techniques for population-based heuristic algorithms [48]–[51]. The relative performances using PI are calculable as:

$$PI = \frac{1}{Np} \sum_{i=1}^{Np} (k_1 \alpha_1^i + k_2 \alpha_2^i + k_3 \alpha_3^i),$$

where,

$$\alpha_1^i = \frac{A^i}{MA^i}$$
$$\alpha_2^i = \frac{S^i}{MS^i}$$
$$\alpha_3^i = \frac{R^i}{MR^i}; \qquad for\ i = 1, 2, ..., N_p$$

and

N_p**:** Total number of problems analyzed

A^i**:** Average values of objective function of i^{th} problem for all competing operators

MA^i**:** Minimum of average value of objective function of i^{th} problem obtained through all competing operators

S^i**:** Standard deviations of objective function of i^{th} problem for all competing operators

MS^i**:** Minimum of Standard deviation of objective function of i^{th} problem obtained through all competing operators

R^i**:** Relative errors of objective function of i^{th} problem for all competing operators

MR^i**:** Minimum of relative error of objective function of i^{th} problem obtained through all competing operators.

Where k_1, k_2 and k_3 are the weights of aforementioned criteria, such that $k_1 + k_2 + k_3 = 1$ for all $0 \leq k_1, k_2, k_3 \leq 1$. Following the [49], the weights are assigned as:

$$case(a): \quad k_1 = w,\ k_2 = k_3 = (1-w)/2,\ 0 \leq w \leq 1$$
$$case(b): \quad k_2 = w,\ k_1 = k_3 = (1-w)/2,\ 0 \leq w \leq 1$$
$$case(c): \quad k_3 = w,\ k_2 = k_3 = (1-w)/2,\ 0 \leq w \leq 1,$$

such that two criteria are given equal weights. For demonstration purposes, we explain in all the PI's Figures with respect to three cases: case (a), the S.D and R.E are given the same weights, whereas in case (b), mean and R.E have the same weights and in the last case (c), same weights have been assigned to mean and S.D.

4.2 Simulation-based results and discussions

This section examines the performance comparisons between the proposed MO-OX and the other most used approaches for permutation based crossover operators. For a better evaluation, we compared the proposed MO-OX to MO-GA, OX, PMX and CX in parallel with three selection and two mutation operators. The selection operators which are used in this study are binary tournament selection (BTS), roulette wheel selection (RWS) and linear rank selection (LRS). The exchange mutation (EM) and inversion mutation (INM) are used for random changes in each generation. So there are six groups of experiments conducted as five crossovers examined with each pair of three selection and two mutation operators.

Table 2 summarizes the results of five competing crossover operators with BTS and EM. The results are compared on the basis of average, S.D and R.E in percentage (%) values. The significant improvements in the results of MO-OX with respect to each other approach are indicated through t-values. The proposed operator indicated less than average values for all six benchmarks with low S.D as well. According to the critical value (t = - 2.00), all computed t-values are less than -2.00 for all six benchmark instances except of one operator for rbg443 instance and one for att532 instance, which is MO-GA. The bold t-test values have shown the significantly improved performance by the proposed operator. The other two values of t-statistic which are not bold (non-significant), but are negative, indicate a better average performance by the proposed operator. In other words, the simulation results found by the MO-OX are statistically significantly better than the solution quality by the other four crossover approaches (MO-GA, OX, PMX and CX). For the given setting of experiment, the relevant graphs of PI for each case are shown in Figure 2, where weights are presented on horizontal axis and PI stays at vertical axis. A vividly superior

performance of the proposed MO-OX can be observed as compared to the existing alternatives in all three cases.

As it was mentioned previously, we also have tested the performance of the MO-OX with the same selection method BTS and new mutation approach INM in Table 3. These results are comparable with the Table 2 results based on average, S.D and R.E performance for all used benchmarks. Based on our simulation results, we can say that BTS has given better results with EM other than with INM. MO-OX statistically outperforms (bold t-test values) than the other four crossover approaches for five out of six benchmark instances ($t \leq$ - 2.00). The only two operators (MO-GA and OX) are non-significant with the proposed operator for the benchmark rbg443. For this experiment, the PI for each case are shown in Figure 3, where we also observed a superior performance by the newly proposed operator for all the given benchmarks.

We continue our simulation study to check the performance of the proposed operator along with other crossover methods and different techniques of selection and mutation. Likewise, in Table 4, we tested the performance of the proposed operator MO-OX with the pair of RWS (selection operator) and EM (mutation operator). The simulation results summarize the lower average, S.D and R.E values for all benchmarks. Based on statistical perspective, the MO-OX outperforms (bold t-test values) all the other crossover methods for all six benchmark instances ($t \leq$ - 2.00) except in one case when compared with MO-GA for the problem rbg443. Through the indicator PI, the MO-OX performs better than all other competing operators (see Figure 4).

Table 5 shows the simulation results of the crossover operators but with another mutation approach, i.e. INM, which only differs from Table 4 approaches. The conclusion based on our results indicated that RWS is better performing with EM other than INM. The MO-GA operator is insignificant with the proposed one only for the instance rbg443. Table 6 summarizes the results of all six used crossover operators along with LRS (selection operator) and EM (mutation operator). The experiment results demonstrated that the proposed approach has a significant improvement as compared to all other approaches. The

significance of such improvement is validated through t-test values (t $\leq$ - 2.00). We presented our more simulation results with LRS and INM in Table 7. The results are clearly in the favor of MO-OX in all measurement aspects. MO-GA is insignificant with the proposed one only for the instance ftv170. Finally, with all these changes in the GA's experiments, the PI performs outstanding in the favor of proposed crossover operator (see Figures 5 to 7).

In summary, the simulation results with the six TSP instances show that MO-OX is effective and gave significantly better results throughout the Tables 2 to 7 along with Figures 2 to 7. The main and first purpose to conduct the series of experiments in this study is to measure the performance of the proposed operator with all other competing crossover operators. The second purpose of the simulation is to detect the best against of selection and mutation operators which work better with these crossover operators. The binary tournament (BTS) and exchange mutation (EM) is the best pair to perform better with these crossover operators (Table 2). Finally, we can conclude our final remarks based on all simulation results of Tables 2 to 7 for minimizing the total travel distance with less variations and relative measures, and more performance indexes. These results suggest that the proposed MO-OX achieve significantly better solution quality for all used benchmark instances when compared to the other four alternative crossover approaches.

5 Conclusion

Unlike other meta-heuristic methods, GA exploits natural rules of selection, crossover and mutation, in search of optimal solutions. Inspired by the key role of crossover operator in the scheme of GA, this article suggested a new crossover operator, namely multi-offspring order crossover (MO-OX) operator. This new approach generates four offspring which shows that it can effectively enhance the search ability of the algorithm by producing more possible solutions. The performance of MO-OX is studied by comparing with four highly debated and commonly used existing alternatives (MO-GA, OX, PMX and CX), through rigorous simulation study. Based on various performance evaluation criteria,

i.e. average performance in 30 trails, S.D, R.E, two-tailed t-test and performance index, a superior performance of the proposed operator is observed.

Table 2. Comparison Results of Crossover Operators with BTS and EM

Instance	Optimal	Crossover	Average	R.E	S.D	t-test
		MO-OX	707	1.14	06	-
		MO-GA	721	3.15	15	**-4.67**
dantzig42	699	OX	722	3.29	23	**-3.30**
		PMX	739	5.72	33	**-5.14**
		CX	748	7.01	35	**-6.22**
		MO-OX	7114	3.03	109	-
		MO-GA	7329	6.14	152	**-6.19**
ft53	6905	OX	7387	6.98	197	**-6.52**
		PMX	7412	7.34	211	**-6.76**
		CX	7518	8.88	273	**-7.40**
		MO-OX	2932	6.42	175	-
		MO-GA	3061	11.11	208	**-2.56**
ftv170	2755	OX	3112	12.96	221	**-3.44**
		PMX	3088	12.09	198	**-3.18**
		CX	3191	15.83	277	**-4.26**
		MO-OX	1997	2.41	28	-
		MO-GA	2071	6.21	70	**-5.29**
brg180	1950	OX	2059	5.59	62	**-4.91**
		PMX	2053	5.28	72	**-3.90**
		CX	2087	7.03	95	**-4.89**
		MO-OX	3518	29.34	317	-
		MO-GA	3714	36.54	414	**-2.02**
rbg443	2720	OX	3609	32.68	343	-1.05
		PMX	3683	35.40	309	**-2.01**
		CX	3802	39.78	449	**-2.78**
		MO-OX	28962	4.61	466	-
		MO-GA	29271	5.72	573	**-2.25**
att532	27686	OX	29152	5.30	522	-1.46
		PMX	29436	6.32	487	**-5.97**
		CX	29629	7.02	627	**-4.60**

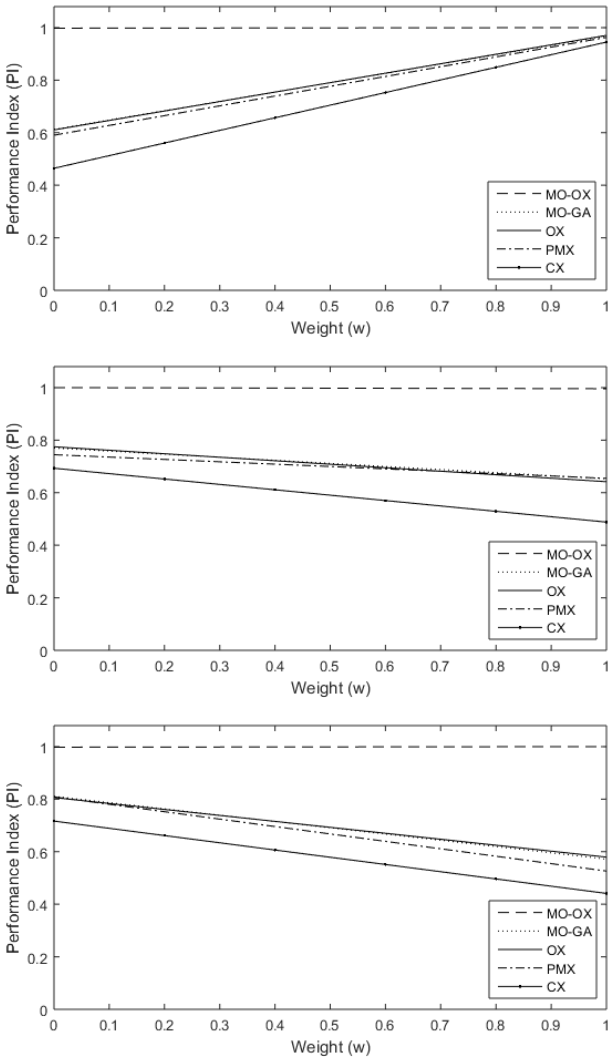

Figure 2. The display of performance index (PI) when: (a) $k_1 = w$ and $k_2 = k_3 = \frac{1-w}{2}$; (b) $k_2 = w$ and $k_1 = k_3 = \frac{1-w}{2}$; and (c) $k_3 = w$ and $k_1 = k_2 = \frac{1-w}{2}$

Table 3. Comparison Results of Crossover Operators with BTS and INM

Instance	Optimal	Crossover	Average	R.E	S.D	t-test
		MO-OX	718	2.72	14	-
		MO-GA	743	6.29	24	**-4.85**
dantzig42	699	OX	751	7.44	31	**-5.22**
		PMX	744	6.44	28	**-4.47**
		CX	759	8.58	33	**-6.16**
		MO-OX	7209	4.40	187	-
		MO-GA	7394	7.08	208	**-3.56**
ft53	6905	OX	7406	7.26	273	**-3.21**
		PMX	7493	8.52	298	**-4.35**
		CX	7527	9.01	330	**-4.51**
		MO-OX	2951	7.11	192	-
		MO-GA	3097	12.41	179	**-3.00**
ftv170	2755	OX	3176	15.28	253	**-3.81**
		PMX	3140	13.97	244	**-3.28**
		CX	3231	17.28	291	**-4.33**
		MO-OX	2018	3.49	54	-
		MO-GA	2131	9.28	101	**-5.31**
brg180	1950	OX	2099	7.64	88	**-4.22**
		PMX	2152	10.36	112	**-5.80**
		CX	2190	12.31	131	**-6.54**
		MO-OX	3573	31.36	350	-
		MO-GA	3648	34.12	395	-0.77
rbg443	2720	OX	3741	37.54	402	-1.70
		PMX	3788	39.26	374	**-2.26**
		CX	3856	41.76	471	**-2.60**
		MO-OX	29623	7.00	499	-
		MO-GA	29931	8.11	548	**-2.24**
att532	27686	OX	30111	8.76	573	**-3.46**
		PMX	30749	11.06	732	**-6.84**
		CX	31008	12.00	987	**-6.74**

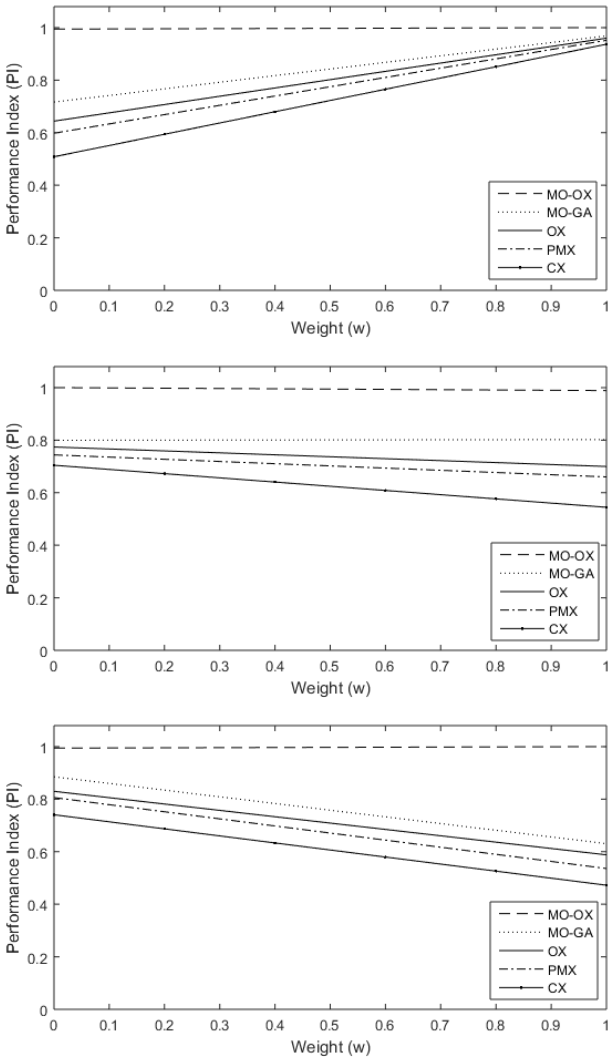

Figure 3. The display of performance index (PI) when: (a) $k_1 = w$ and $k_2 = k_3 = \frac{1-w}{2}$; (b) $k_2 = w$ and $k_1 = k_3 = \frac{1-w}{2}$; and (c) $k_3 = w$ and $k_1 = k_2 = \frac{1-w}{2}$

Table 4. Comparison Results of Crossover Operators with RWS and EM

Instance	Optimal	Crossover	Average	R.E	S.D	t-test
		MO-OX	710	1.57	08	-
		MO-GA	727	4.01	18	**-4.65**
dantzig42	699	OX	718	2.72	14	**-2.67**
		PMX	731	4.58	29	**-3.76**
		CX	756	8.15	40	**-6.07**
		MO-OX	7142	3.43	122	-
		MO-GA	7388	6.99	149	**-6.88**
ft53	6905	OX	7408	7.28	231	**-5.48**
		PMX	7453	7.94	260	**-5.83**
		CX	7485	8.40	279	**-6.07**
		MO-OX	2937	6.61	223	-
		MO-GA	3096	12.38	217	**-2.75**
ftv170	2755	OX	3153	14.45	263	**-3.37**
		PMX	3164	14.85	251	**-3.64**
		CX	3172	15.14	318	**-3.26**
		MO-OX	2009	3.03	057	-
		MO-GA	2121	8.77	103	**-5.12**
brg180	1950	OX	2095	7.44	084	**-4.56**
		PMX	2140	9.74	099	**-6.18**
		CX	2179	11.74	111	**-7.34**
		MO-OX	3550	30.51	374	-
		MO-GA	3693	35.77	452	-1.31
rbg443	2720	OX	3761	38.27	412	**-2.04**
		PMX	3742	37.57	384	-1.93
		CX	3783	39.08	477	**-2.07**
		MO-OX	29311	5.87	776	-
		MO-GA	29894	7.98	707	**-2.99**
att532	27686	OX	30040	12.48	689	**-3.78**
		PMX	29993	8.33	887	**-3.12**
		CX	30301	13.06	990	**-4.24**

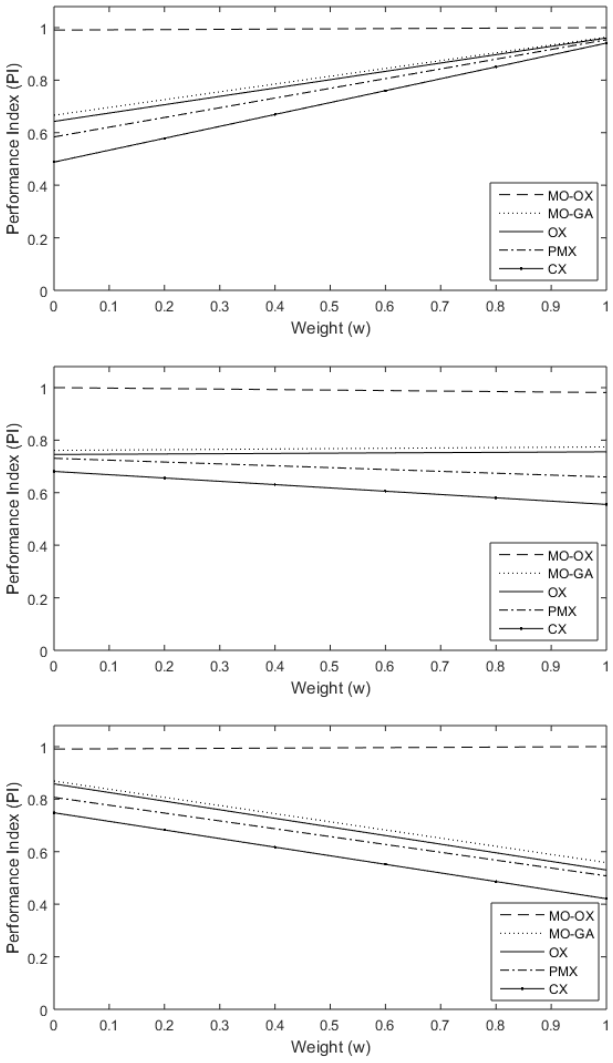

Figure 4. The display of performance index (PI) when: (a) $k_1 = w$ and $k_2 = k_3 = \frac{1-w}{2}$; (b) $k_2 = w$ and $k_1 = k_3 = \frac{1-w}{2}$; and (c) $k_3 = w$ and $k_1 = k_2 = \frac{1-w}{2}$

Table 5. Comparison Results of Crossover Operators with RWS and INM

Instance	Optimal	Crossover	Average	R.E	S.D	t-test
		MO-OX	719	2.86	12	-
		MO-GA	735	5.15	20	**-3.69**
dantzig42	699	OX	728	4.15	19	**-2.16**
		PMX	749	7.15	38	**-4.05**
		CX	744	6.44	31	**-4.05**
		MO-OX	7304	5.78	167	-
		MO-GA	7474	8.24	202	**-3.49**
ft53	6905	OX	7525	8.98	253	**-3.93**
		PMX	7582	9.80	288	**-4.50**
		CX	7483	8.37	312	**-2.72**
		MO-OX	3071	11.47	211	-
		MO-GA	3233	17.35	264	**-2.58**
ftv170	2755	OX	3210	16.52	283	**-2.12**
		PMX	3303	19.89	246	**-3.85**
		CX	3346	21.45	298	**-4.06**
		MO-OX	2024	3.79	069	-
		MO-GA	2132	9.33	113	**-4.39**
brg180	1950	OX	2168	11.18	142	**-4.91**
		PMX	2185	12.05	130	**-5.89**
		CX	2262	16.00	128	**-8.81**
		MO-OX	3591	32.02	383	-
		MO-GA	3689	35.63	456	-0.89
rbg443	2720	OX	3794	39.49	375	**-2.04**
		PMX	3799	39.67	394	**-2.04**
		CX	3851	41.58	459	**-2.34**
		MO-OX	29654	7.11	865	-
		MO-GA	30753	11.08	954	**-4.60**
att532	27686	OX	31631	14.25	876	**-8.64**
		PMX	31892	15.19	989	**-9.17**
		CX	31504	13.79	981	**-7.62**

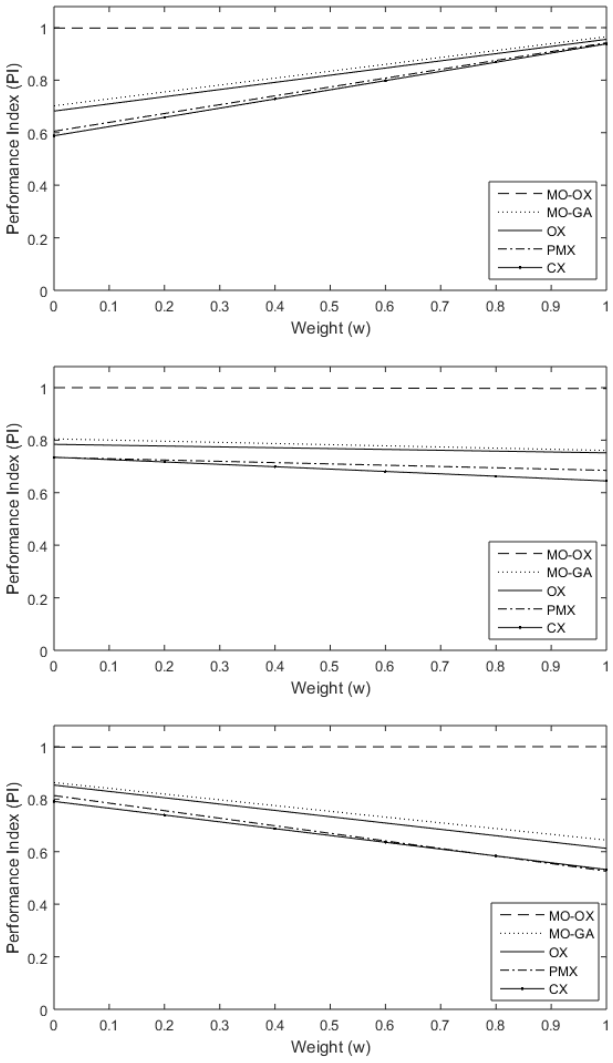

Figure 5. The display of performance index (PI) when: (a) $k_1 = w$ and $k_2 = k_3 = \frac{1-w}{2}$; (b) $k_2 = w$ and $k_1 = k_3 = \frac{1-w}{2}$; and (c) $k_3 = w$ and $k_1 = k_2 = \frac{1-w}{2}$

Table 6. Comparison Results of Crossover Operators with LRS and EM

Instance	Optimal	Crossover	Average	R.E	S.D	t-test
		MO-OX	725	3.72	18	-
		MO-GA	741	6.01	29	**-2.52**
dantzig42	699	OX	753	7.73	35	**-3.83**
		PMX	754	7.86	39	**-3.64**
		CX	760	8.73	52	**-3.43**
		MO-OX	7217	4.52	131	-
		MO-GA	7376	6.82	183	**-3.80**
ft53	6905	OX	7371	6.75	214	**-3.31**
		PMX	7439	7.73	241	**-4.36**
		CX	7502	8.65	298	**-4.71**
		MO-OX	2953	7.19	169	-
		MO-GA	3084	11.94	219	**-2.55**
ftv170	2755	OX	3059	11.03	204	**-2.15**
		PMX	3104	12.67	217	**-2.96**
		CX	3214	16.66	306	**-4.02**
		MO-OX	2001	2.62	32	-
		MO-GA	2093	7.33	65	**-6.84**
brg180	1950	OX	2114	8.41	79	**-7.14**
		PMX	2177	11.64	97	**-9.28**
		CX	2197	12.67	113	**-8.99**
		MO-OX	3573	31.36	392	-
		MO-GA	3794	39.49	416	**-2.08**
rbg443	2720	OX	3716	36.62	388	-1.40
		PMX	3791	39.38	405	**-2.08**
		CX	3865	42.10	483	**-2.53**
		MO-OX	29782	7.57	864	-
		MO-GA	30764	11.12	968	**-4.08**
att532	27686	OX	31081	12.26	892	**-5.63**
		PMX	30749	11.06	979	**-3.99**
		CX	31138	12.47	785	**-6.26**

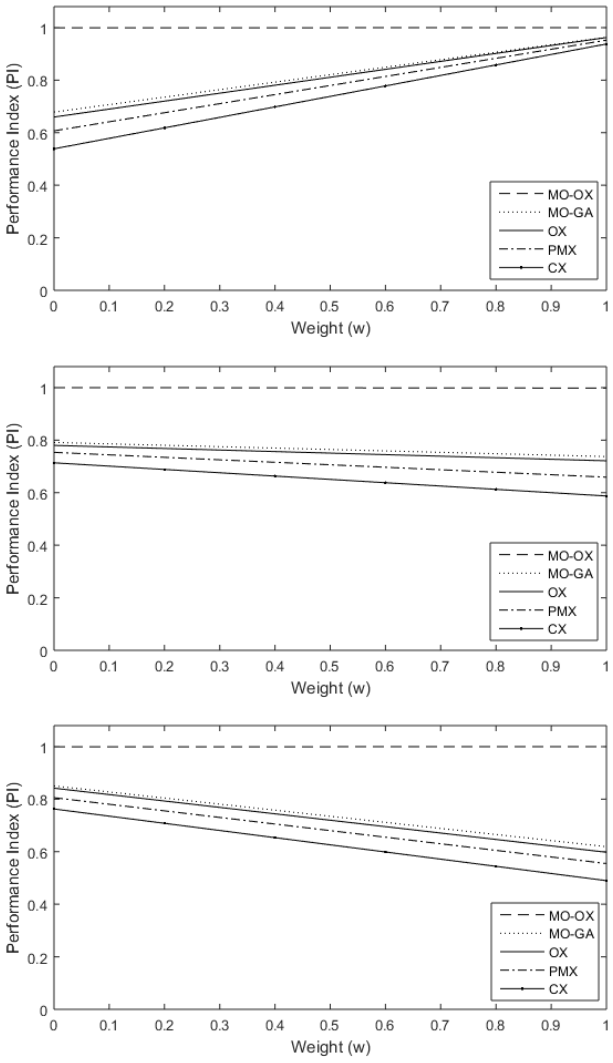

Figure 6. The display of performance index (PI) when: (a) $k_1 = w$ and $k_2 = k_3 = \frac{1-w}{2}$; (b) $k_2 = w$ and $k_1 = k_3 = \frac{1-w}{2}$; and (c) $k_3 = w$ and $k_1 = k_2 = \frac{1-w}{2}$

Table 7. Comparison Results of Crossover Operators with LRS and INM

Instance	Optimal	Crossover	Average	R.E	S.D	t-test
		MO-OX	735	5.15	20	-
		MO-GA	749	7.15	27	**-2.24**
dantzig42	699	OX	759	8.58	33	**-3.35**
		PMX	772	10.44	47	**-3.90**
		CX	770	10.16	59	**-3.03**
		MO-OX	7184	4.04	138	-
		MO-GA	7279	5.42	162	**-2.40**
ft53	6905	OX	7387	6.98	227	**-4.12**
		PMX	7421	7.47	249	**-4.48**
		CX	7490	8.47	213	**-6.49**
		MO-OX	2968	7.73	171	-
		MO-GA	3004	9.04	192	-0.75
ftv170	2755	OX	3143	14.08	246	**-3.15**
		PMX	3175	15.25	217	**-4.03**
		CX	3185	15.61	278	**-3.58**
		MO-OX	2012	3.18	36	-
		MO-GA	2077	6.51	77	**-4.12**
brg180	1950	OX	2114	8.41	86	**-5.89**
		PMX	2210	13.33	123	**-8.32**
		CX	2233	14.51	116	**-7.04**
		MO-OX	3547	30.40	443	-
		MO-GA	3796	39.56	487	**-2.04**
rbg443	2720	OX	3828	40.74	513	**-2.23**
		PMX	3841	41.21	519	**-2.32**
		CX	3854	41.69	468	**-2.57**
		MO-OX	30313	9.49	795	-
		MO-GA	30975	11.88	876	**-3.01**
att532	27686	OX	30844	11.41	922	**-2.35**
		PMX	31759	14.71	985	**-6.15**
		CX	31627	14.24	878	**-5.97**

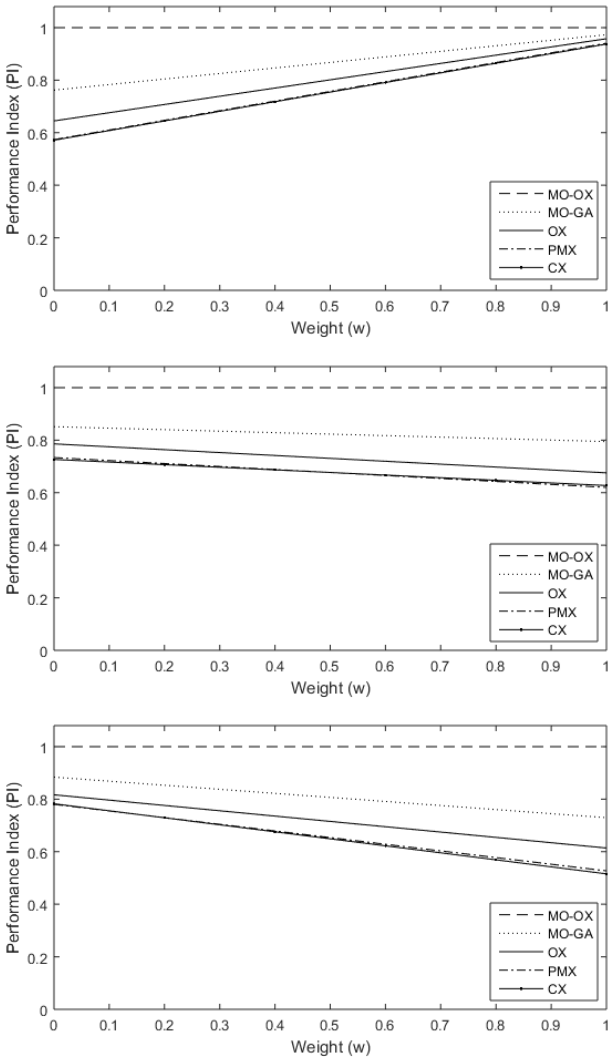

Figure 7. The display of performance index (PI) when: (a) $k_1 = w$ and $k_2 = k_3 = \frac{1-w}{2}$; (b) $k_2 = w$ and $k_1 = k_3 = \frac{1-w}{2}$; and (c) $k_3 = w$ and $k_1 = k_2 = \frac{1-w}{2}$

Data availability

The data used to support the findings of this manuscript are taken from the website of TSPLIB (https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/).

Disclosure statement

The author of this article declares that there is no conflicts of interest regarding the publication of this article.

References

[1] A. Philip, A. A. Taofiki, and O. Kehinde, "A genetic algorithm for solving traveling salesman problem," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 2, no. 1, pp. 26–29, 2011.

[2] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the traveling salesman problem: A review of representations and operators," *Artificial Intelligence Review*, vol. 13, no. 2, pp. 129–170, 1999.

[3] M. R, Garey and D. S. Johnson, *Computers and Intractability – A Guide to NP-completeness,* San Francisco: W.H. Freeman and Company, 1979.

[4] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization,* Prentice Hall Englewood Cliffs, 1982.

[5] S. Singh and E. A. Lodhi, "Study of variation in tsp using genetic algorithm and its operator comparison," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 3, pp. 264–267, 2013.

[6] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations Research*, vol. 21, no. 2, pp. 498–516, 1973.

[7] M. A. Mohammed, M. K. A. Ghani, R. I. Hamed, S. A. Mostafa, M. S. Ahmad, and D. A. Ibrahim, "Solving vehicle routing problem by using improved genetic algorithm for optimal solution," *Journal of Computational Science*, vol. 21, pp. 255–262, 2017.

[8] F. Alizadeh, R. M. Karp, L. A. Newberg, and D. K. Weisser, "Physical mapping of chromosomes: A combinatorial problem in molecular biology," *Algorithmica*, vol. 13, no. (1-2), pp. 52–76, 1995.

[9] C. Korostensky and G. H. Gonnet, "Using traveling salesman problem algorithms for evolutionary tree construction," *Bioinformatics*, vol. 16, no. 7, pp. 619–627, 2000.

[10] B. D. Corwin and A. O. Esogbue, "Two machine flow shop scheduling problems with sequence dependent setup times: A dynamic programming approach," *Naval Research Logistics (NRL)*, vol. 21, no. 3, pp. 515–524, 1974.

[11] G. Finke. "A two-commodity network flow approach to the traveling salesman problem," *Congresses Numeration*, vol. 41, pp. 167–178, 1984.

[12] P. Miliotis, "Using cutting planes to solve the symmetric traveling salesman problem," *Mathematical Programming*, vol. 15, no. 1, pp. 177–188, 1978.

[13] M. Cunkas and M. Y. Ozsaglam, "A comparative study on particle swarm optimization and genetic algorithms for traveling salesman problems," *Cybernetics and Systems: An International Journal*, vol. 40, no. 6, pp. 490–507, 2009.

[14] C. H. Song, K. Lee, and W. D. Lee, "Extended simulated annealing for augmented tsp and multi-salesmen tsp," in *Proceedings of the International Joint Conference on Neural Networks, 2003, IEEE*, vol. 3, 2003, pp. 2340–2343.

[15] V. Maniezzo and M. Roffilli, "Very strongly constrained problems: an ant colony optimization approach," *Cybernetics and Systems: An International Journal*, vol. 39, no. 4, pp. 395–424, 2008.

[16] S. Bhide, N. John, and M. R. Kabuka, "A boolean neural network approach for the traveling salesman problem," *IEEE Transactions on Computers*, vol. 42, no. 10, pp. 1271–1278, 1993.

[17] Y. He, Y. Qiu, G. Liu, and K. Lei, "A parallel adaptive tabu search approach for traveling salesman problems," in *Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE'05. Proceedings of 2005 IEEE International Conference on*, IEEE, 2005, pp. 796–801.

[18] C. Moon, J. Kim, G. Choi, and Y. Seo, "An efficient genetic algorithm for the traveling salesman problem with precedence constraints," *European Journal of Operational Research*, vol. 140, no. 3, pp. 606–617, 2002.

[19] M. Bhattacharyya and A. K. Bandyopadhyay, "Comparative study of some solution methods for traveling salesman problem using genetic algorithms," *Cybernetics and Systems*, vol. 40, no. 1, pp. 1–24, 2008.

[20] Z. H. Ahmed, "Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator," *International Journal of Biometrics & Bioinformatics (IJBB)*, vol. 3, no. 6, pp. 96–105, 2010.

[21] A. Hussain, Y. S. Muhammad, M. N. Sajid, I. Hussain, A. M. Shoukry, and S. Gani, "Genetic algorithm for traveling salesman problem with modified cycle crossover operator," *Computational Intelligence and Neuroscience*, vol. 2017, pp. 1–7, 2017.

[22] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, 2nd ed., MIT press, 1992.

[23] C. Darwin, *On the Origin of Species by Means of Natural Selection Or the Preservation of Favored Races in the Struggle for Life*, H. Milford; Oxford University Press, 1859.

[24] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, 1994.

[25] D. E. Goldberg and R. Lingle, "Alleles, loci, and the traveling salesman problem," in *Proceedings of an international conference on genetic algorithms and their applications*, Hillsdale, NJ: Lawrence Erlbaum, vol. 154, 1985, pp. 154–159.

[26] I. C. Choi, S. I. Kim, and H. S. Kim, "A genetic algorithm with a mixed region search for the asymmetric traveling salesman prob-

lem," *Computers & Operations Research*, vol. 30, no. 5, pp. 773–786, 2003.

[27] Z. Ursani, D. Essam, D. Cornforth, and R. Stocker, "Localized genetic algorithm for vehicle routing problem with time windows," *Applied Soft Computing*, vol. 11, no. 8, pp. 5375–5390, 2011.

[28] H. Muhlenbein, "Parallel genetic algorithms, population genetics and combinatorial optimization," in *Workshop on Parallel Processing: Logic, Organization, and Technology*, Springer, 1989, pp. 398–406.

[29] J. E. Baker, "Adaptive selection methods for genetic algorithms," in *Proceedings of an International Conference on Genetic Algorithms and their applications*, Hillsdale, New Jersey, 1985, pp. 101–111.

[30] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," in *Foundations of Genetic Algorithms* (vol. 1), Elsevier, 1991, pp. 69–93.

[31] J. Wang, O. K. Ersoy, M. He, and F. Wang, "Multi-offspring genetic algorithm and its application to the traveling salesman problem," *Applied Soft Computing*, vol. 43, pp. 415–423, 2016.

[32] A. M. Aibinu, H. B. Salau, N. A. Rahman, M. N. Nwohu, and C. M. Akachukwu, "A novel clustering based genetic algorithm for route optimization," *Engineering Science and Technology, an International Journal*, vol. 19, no. 4, pp. 2022–2034, 2016.

[33] L. M. A. Drummond, L. S. Ochi, and D. S. Vianna, "An asynchronous parallel metaheuristic for the period vehicle routing problem," *Future Generation Computer Systems*, vol. 17, no. 4, pp. 379–386, 2001.

[34] K. Ganesh and T. T. Narendran, "Cloves: A cluster-and-search heuristic to solve the vehicle routing problem with delivery and pick-up," *European Journal of Operational Research*, vol. 178, no. 3, pp. 699–717, 2007.

[35] M. M. S. Haghighi, M. H. Zahedi, and M. Ghazizadeh, "A multi level priority clustering ga based approach for solving heterogeneous vehicle routing problem (pcgvrp)," in *Innovations and Ad-*

vances in Computer Sciences and Engineering, Springer, 2010, pp. 331–335.

[36] L. Davis, "Applying adaptive algorithms to epistatic domains," in *IJCAI'85 Proceedings of the 9th international joint conference on Artificial intelligence*, vol. 1, pp. 162–164, 1985.

[37] I. M. Oliver, D. J. Smith, and J. R. C. Holland, "Study of permutation crossover operators on the traveling salesman problem," in *Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms* (July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA), Hillsdale, NJ: L. Erlhaum Associates, 1987, pp. 224–230.

[38] W. Banzhaf, "The 'molecular' traveling salesman," *Biological Cybernetics*, vol. 64, no. 1, pp. 7–14, 1990.

[39] D. B. Fogel, "An evolutionary approach to the traveling salesman problem," *Biological Cybernetics*, vol. 60, no. 2, pp. 139–144, 1988.

[40] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, New York: Springer, 1996.

[41] D. B. Fogel, "Applying evolutionary programming to selected traveling salesman problems," *Cybernetics and Systems*, vol. 24, no. 1, pp. 27–36, 1993.

[42] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[43] J. Berger and M. Barkaoui, "A parallel hybrid genetic algorithm for the vehicle routing problem with time windows," *Computers & Operations Research*, vol. 31, no. 12, pp. 2037–2053, 2004.

[44] M. Skok, D. Skrlec, and S. Krajcar, "The genetic algorithm method for multiple depot capacitated vehicle routing problem solving," in *Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000, Proc. Fourth International Conference on, vol. 2*, IEEE, 2000, pp. 520–526.

[45] K. Q. Zhu, "A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows," in *Tools with Artificial Intelligence, 2003. Proc. 15th IEEE International Conference on*, IEEE, 2003, pp. 176–183.

[46] X. Mou, D. l. Xie, and W. Yan, "Research based on genetic algorithm traveling sealer problem of trajectory optimization," *Journal of System Simulation*, vol. 25, pp. 86–89, 2013.

[47] G. Reinelt, "Tsplib http://www. iwr. uni-heidelberg. de/groups/comopt/software". [Online]. Available: *TSPLIB95*, 1995.

[48] Bharti, "Controlled random search technique and their applications," PhD thesis, Department of Mathematics, University of Roorkee, India, 1994.

[49] C. Mohan and H. T. Nguyen, "A controlled random search technique incorporating the simulated annealing concept for solving integer and mixed integer global optimization problems," *Computational Optimization and Applications*, vol. 14, no. 1, pp. 103–132, 1999.

[50] K. Deep and M. Thakur, "A new crossover operator for real coded genetic algorithms," *Applied Mathematics and Computation*, vol. 188, no. 1, pp. 895–911, 2007.

[51] M. Thakur, "A new genetic algorithm for global optimization of multimodal continuous functions," *Journal of Computational Science*, vol. 5, no. 2, pp. 298–311, 2014.

Received October 21, 2019

Abid Hussain
Department of Statistics, Quaid-i-Azam University, Islamabad, Pakistan.
E–mail: abid0100@gmail.com

Salman A. Cheema
School of Mathematical and Physical Sciences, University of Newcastle, Australia
E–mail: saqvn.cheema@gmail.com