Computers, Materials & Continua
DOI: 10.32604/cmc.2023.031325
Article

Tech Science Press

Genetic Crossover Operators for the Capacitated Vehicle Routing Problem

Zakir Hussain Ahmed[1,*], Naif Al-Otaibi[1], Abdullah Al-Tameem[2] and Abdul Khader Jilani Saudagar[2]

[1]Department of Mathematics and Statistics, College of Science, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia
[2]Department of Information Systems, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh, Saudi Arabia
*Corresponding Author: Zakir Hussain Ahmed. Email: zaahmed@imamu.edu.sa
Received: 14 April 2022; Accepted: 15 June 2022

Abstract: We study the capacitated vehicle routing problem (CVRP) which is a well-known NP-hard combinatorial optimization problem (COP). The aim of the problem is to serve different customers by a convoy of vehicles starting from a depot so that sum of the routing costs under their capacity constraints is minimized. Since the problem is very complicated, solving the problem using exact methods is almost impossible. So, one has to go for the heuristic/metaheuristic methods and genetic algorithm (GA) is broadly applied metaheuristic method to obtain near optimal solution to such COPs. So, this paper studies GAs to find solution to the problem. Generally, to solve a COP, GAs start with a chromosome set named initial population, and then mainly three operators-selection, crossover and mutation, are applied. Among these three operators, crossover is very crucial in designing and implementing GAs, and hence, numerous crossover operators were developed and applied to different COPs. There are two major kinds of crossover operators-blind crossovers and distance-based crossovers. We intend to compare the performance of four blind crossover and four distance-based crossover operators to test the suitability of the operators to solve the CVRP. These operators were originally proposed for the standard travelling salesman problem (TSP). First, these eight crossovers are illustrated using same parent chromosomes for building offspring(s). Then eight GAs using these eight crossover operators without any mutation operator and another eight GAs using these eight crossover operators with a mutation operator are developed. These GAs are experimented on some benchmark asymmetric and symmetric instances of numerous sizes and various number of vehicles. Our study revealed that the distance-based crossovers are much superior to the blind crossovers. Further, we observed that the sequential constructive crossover with and without mutation operator is the best one for the CVRP. This estimation is validated by Student's t-test at 95% confidence level. We further determined a comparative rank of the eight crossovers for the CVRP.

Keywords: Vehicle routing problem; NP-hard; genetic algorithm; sequential constructive crossover; mutation

1 Introduction

There are numerous real-life distribution management problems, where constructing delivery route from a headquarters to some customers is very important. If same customers must be provided service very frequently, it is preferred to set up an optimal tour. This problem is called vehicle routing problem (VRP) which is a very challenging optimization problem. It was introduced first in [1] as follows: There is a convoy of vehicles with the same or different capacities in a depot. These vehicles are expected to serve some customers with different demands such that cost of the tours of the vehicles is minimum. The VPR has numerous variants, namely, the VRP with pickup and delivery (VRPPD), the VRP with time window (VRPTW), the VRP with Backhauls (VRPB), the multi-depot VRP (MDVRP), the split delivery VRP (SDVRP), the capacitated VRP (CVRP) [2].

We consider the CVRP for our study. Several researchers extensively studied on the CVRP, where a convoy of vehicles provides services to different customers from a depot (headquarters) so that sum of the routing costs under their capacity constraints is minimized. The CVRP is a mixture of the standard travelling salesman problem (TSP) and the bin packing problem (BPP). The CVRP has applications in many real-life problems, such as renting-sharing problems for urban bicycles [3], scheduling and routing of retail stores [4], dispensing medical supplies [5], etc. The classical CVRP is NP-hard [6] and is a very complicated problem. For example, for a *n*-customer and *m*-vehicle problem, we are listing *n* customers in a sequence (that can be implemented in n! ways), and then placing *m–1* constraints to determine when a tour has concluded after $(m-1)$ of $(n-1)$ customers, that can be implemented in $\binom{n-1}{m-1}$ ways, producing $n!\binom{n-1}{m-1}\Big/m!$ probable solutions (as the vehicle sequence is irrelevant, we divide by m!). One can visualize that with more than 10 customers and 3 vehicles, there are many solutions. For any size n, the number of feasible solutions is too large; so, an extensive search is too complicated. That is, the problem is too complicated to solve. Various procedures to solve the CVRP were reported in numerous literatures, which are categorized into two major categories–exact methods and heuristic methods. Exact methods, for instance, branch and bound [7], branch and price [8], branch and cut [9], lexisearch algorithm [10], give exact solutions. However, as the problem size increases, finding exact solutions using these methods is very difficult. On the other hand, heuristic methods don't ensure the exact solutions but give near exact solutions rapidly. The most recent methods which can be applied to several optimization problems are called metaheuristics. Several metaheuristics were proposed to solve this problem. These algorithms are simulated annealing [11], tabu search [12], ant colony optimization [13], particle swarm optimization [14], genetic algorithms [15], variable neighbourhood method [16], etc. However, genetic algorithms (GAs) are observed to be widely used methods amongst latest metaheuristic approaches, and hence, we are using GAs to obtain solution to the CVRP.

Introduced first by John Holland, GAs are proposed according to the survival-of-the-fittest theory amongst various species produced by random differences in the structure of chromosomes in the natural science. Because GAs are easy, flexible and simple to code, they are extremely successful. Generally, GAs start with a chromosome set named initial population. Then mainly three operators, specifically, selection, crossover and mutation, are applied to find solution to any problem. Selection operator selects better chromosomes among the existing chromosomes, crossover operator probabilistically produces new chromosome(s) called offspring(s) by mating two chromosomes, and mutation probabilistically changes some genes in the chromosomes. Among these three operators crossover is very crucial in designing and implementing GAs [17]. Hence, different crossover operators can be applied in GAs for the CVRP which were originally proposed for the TSP. This paper intends to compare the performance of some crossover operators to test the suitability of the operators to

solve the CVRP. These operators were originally proposed for the standard TSP. We further aim to determine a comparative rank of the crossovers for the CVRP.

This manuscript is prepared as follows: Section 2 reports definition and model of the CVRP. Section 3 briefly discusses existing literatures for the CVRP using genetic crossover operators. Section 4 discusses GAs using existing crossover operators for the CVRP. Section 5 reports the experimental results of the GAs on some asymmetric and symmetric instances. Section 6 presents a conclusion.

2 Problem Definition: The CVRP

The CVRP can be stated as: Suppose $V = \{0, 1, 2, \ldots, n\}$ be a set of customers (or nodes or cities), 'customer 0' is the headquarters and there are m vehicles, each with capacity Q. Each customer $i \in V$, related to a non-negative demand q_i, is required to be visited exactly once. The number of vehicles may be predetermined or determined during the process. Also, suppose, $C = [c_{ij}]$ be a travel cost (or time or distance) matrix associated with each pair of customers. The matrix C may be symmetric or asymmetric; the latter case is due to the presence of one-way roads in urban areas. The problem is to find an optimal tour that has minimum total arrival cost at n customers utilizing all m vehicles, so that starting from and stopping at the headquarters all customers are to be visited (serviced) just once and the total demand serviced does not exceed vehicle capacity Q [18]. That is, a solution of the CVRP with m routes $\{R_1, R_2, \ldots, R_m\}$ must satisfy the following equation:

$$\sum_{i \varepsilon R_j} q_i \leq Q \tag{1}$$

The CVRP aims to minimize the total cost of the tour, S, described as:

$$f(S) = \sum_{i,j \varepsilon V} c_{ij} \tag{2}$$

As shown in Fig. 1, the CVRP is concerned with finding the optimal tours for 3 vehicles to fulfill the demands of 10 customers. Note that 'customer 1' is the headquarters (depot). The Fig. 1. denotes the tour $\{1\rightarrow5\rightarrow3\rightarrow\ 1\rightarrow10\rightarrow8\rightarrow\ 2\rightarrow7\rightarrow1\rightarrow\ 6\ \rightarrow4\rightarrow\ 9\ \rightarrow1\}$ of the vehicles. Here, the customers 5 and 3 are served in the order by the 1st vehicle, the customers 10, 8, 2 and 7 are served in the order by the 2nd vehicle, and the customers 6, 4 and 9 are served in the order by the 3rd vehicle.

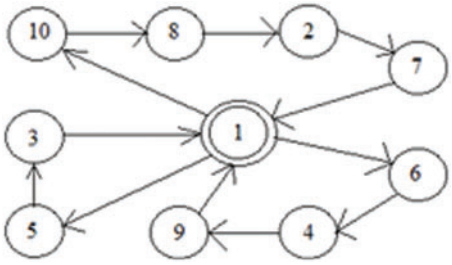

Figure 1: Graphical representation of the CVRP with n = 10 and m = 3

3 Literature Review

We aim to compare the performances of numerous crossover operators to find solution to the CVRP. Therefore, we report a short review on the literatures that applied different crossover operators to find solution to the problem. It is noted that the crossover operators which are designed for the TSP could also be utilized for the CVRP.

In [19], three crossovers operators, namely order crossover (OX), partially mapped crossover (PMX) and uniform crossover (UX), and three mutation operators, namely, swapping, inversion, swapping + inversion, with two types of solution representation, namely, direct coding and indirect coding, for the CVRP are compared. Experimental study on seven instances from a standard set of [20] showed that PMX with inversion mutation had the best performance for direct encoding, and OX with inversion mutation gave best solution values for indirect representation.

In [21], an optimized crossover genetic algorithm (OCGA) is introduced for the VRP with capacity restrictions. The main feature of the algorithm was that vehicles with similar capacities located in a depot were used in a way to optimize the routes and satisfy the demands of the customers. The proposed algorithm used an optimized crossover operator that employed a directed complete bipartite graph for finding an optimal set of delivery routes, to satisfy demands and minimize the total costs. The experimental results indicated that the algorithm is competitive.

In [22], a comparative study among eight crossover operators is reported for the VRP. The crossovers are OX, PMX, alternating edges crossover (AEX), edge recombination crossover (ERX), cycle crossover (CX), heuristic greedy crossovers (HGreX), heuristic probabilistic crossover (HProX) and heuristic random crossover (HRndX). It is reported that HGreX and AEX are competing and are found to be very good operators.

In [23], some crossover operators are applied for solving the CVRP. Further, they proposed a new crossover operator named sinusoidal motion crossover (SMC) and demonstrated it with two examples.

In [24], a giant tour best cost crossover (GTBCX) for the CVRP is proposed. Further, a non-dominated sorting GA-II (NSGA-II) utilizing GTBCX is proposed to optimize two objective values in the CVRP, i.e., the overall distance travelled by the given fleet of vehicles and the length of the longest route.

In [25], the sequential constructive crossover (SCX) is developed for the usual TSP and then utilized for other related problems.

In general, there are two kinds of crossover operators available in the literature. One is the distance-based crossover, and another is the blind crossover. The distance-based crossovers consider distances among customers (or cities or nodes) and the blind crossovers do not consider any data related to the problem instance. The comparison will not be fair if the effectiveness of the distance-based crossover is evaluated against the blind crossovers [26]. So, we consider four blind crossovers, namely, OX, PMX, CX and AEX, and four distance-based crossovers, namely, heuristic crossover (HX), greedy crossover (GX), modified HX (MHX) and SCX for our comparative study. Illustration of these crossover operators are reported for a pair of parent chromosomes. Further, these crossover operators without mutation and with mutation operators are encoded in Visual C++ and then experimented on some benchmark instances with different sizes and various number of vehicles. Our investigation indicates the superiority of the SCX for this problem.

4 GAs for the CVRP

The aim of this current study is to evaluate the performance of various crossover operators in GAs for the CVRP. Therefore, the general structure of the developed GA is selected to differentiate the performance of the compared crossover operators.

4.1 Chromosome Representation and Initial Population

To apply GA to solve any optimization problem, a representation of a solution as chromosome should be defined initially. In our GAs, an integer chromosome using path representation is considered

whose length is $n + m - 1$, where n and m be the numbers of customers and vehicles respectively. In this representation, additional $m - 1$ customers representing dummy depots are added to represent the starting of tours by new vehicles [27]. Fig. 2a displays an example chromosome (1, 6, 9, 8, 5, 2, 10, 4, 3, 7) and its corresponding tour {1→6→9→ 8→5→2→ 10→4→3→7 →1} with 9 customers and 2 vehicles, where the integers 1 and 10 are the depots, the others are customers. Fig. 2b shows the routes {1→6→9→ 8→5→2→ 1} and {1→4→3→7 →1} of the 1st and 2nd vehicles respectively, and their graphical versions are displayed in Fig. 2c. It means that the customers 6, 9, 8, 5 and 2 are served in the order by the 1st vehicle, and the customers 4, 3 and 7 are served in the order by the 2nd vehicle. Therefore, the given cost matrix should be augmented for showing the dummy depots. For this reason, $(m - 1)$ copy of the depot (customer 1) column and row (i.e., 1st column and 1st row) are attached to the original given cost matrix.

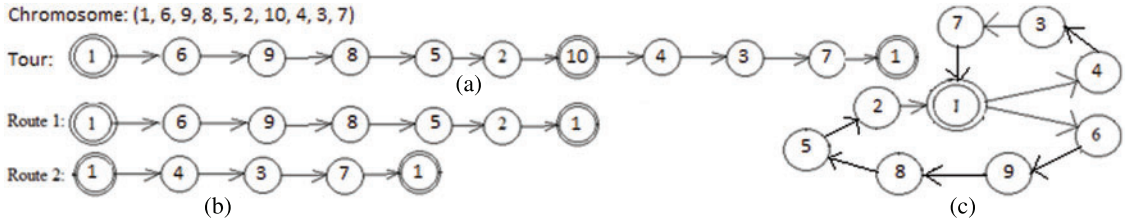

Figure 2: (a) A chromosome example, (b) the CVRP routes and (c) the graphical version

The initial population of chromosomes of predefined size (P_s) is created arbitrarily as in Algorithm 1.

Algorithm 1: *Initial population algorithm*

Input: *Number of customers n; Number of vehicles m; Capacity of vehicles Q; Demands of all vehicles q_i for all i; Population size P_s.*
Output: *Population of chromosomes.*
for *i = 1* ***to*** *P_s* ***do***
  Set *p = 1.*
  Set n=n+m-1;
  Create a list of customers randomly, A=[a_j], including dummy depots at the end of the list.
  Set *total demand of the customers present in the route of the current vehicle (D) = 0.*
  Current chromosome contains only 'customer 1' and delete it from the list
  for *j = 2* ***to*** *n* ***do***
    for *k = 1* ***to*** *n-j* ***do***
      Select next customer p=a_k from the list with its demand q_p.
      If *(D + q_p <=Q), then*
        Include the 'customer p' in the route of the present vehicle and delete it from the customers' list A.
        Compute D = D + q_p.
        If the 'customer p' is a depot, then it is a beginning of a new route by a new vehicle. So, set D = 0.
        Exit from loop k.
      endif
    endfor
  end for
end for
Return *the population*

We illustrate a chromosome generation through the 9 customers and 2 vehicles with given matrix (Tab. 1). We modify the given cost matrix by combining a copy of headquarters (customer 1) column and row (i.e., 1st column and 1st row) to the matrix [28] (Tab. 2). Given the capacity of every vehicle, Q = 100. Further, demands of the customers are given in the Tab. 2. Tab. 3 shows the generation of the chromosome (1, 6, 9, 8, 5, 3, 10, 2, 4, 7). Similarly, a population of chromosomes is generated.

Table 1: The given cost matrix

Customer	1	2	3	4	5	6	7	8	9
1	999	16	22	15	5	17	14	9	17
2	16	999	6	9	21	16	10	15	3
3	13	7	999	16	13	8	9	8	15
4	17	11	22	999	6	14	21	16	14
5	19	15	21	24	999	17	4	14	8
6	17	21	14	10	26	999	21	15	22
7	13	9	12	22	14	11	999	8	23
8	15	11	8	20	13	9	13	999	28
9	19	5	21	14	9	18	5	27	999

Table 2: The modified cost matrix

Customer	1	2	3	4	5	6	7	8	9	10
1	999	16	22	15	5	17	14	9	17	999
2	16	999	6	9	21	16	10	15	3	16
3	13	7	999	16	13	8	9	8	15	13
4	17	11	22	999	6	14	21	16	14	17
5	19	15	21	24	999	17	4	14	8	19
6	17	21	14	10	26	999	21	15	22	17
7	13	9	12	22	14	11	999	8	23	13
8	15	11	8	20	13	9	13	999	28	15
9	19	5	21	14	9	18	5	27	999	19
10	999	16	22	15	5	17	14	9	17	999
Demand	0	24	13	20	27	25	29	18	12	0

Table 3: Generation of a chromosome

Random list of customers	Select 'customer p' with q_p and D	Is $(D + q_p <= Q)$?	Chromosome with D
6, 9, 8, 5, 2, 4, 3, 7, 10	'customer 6' with $q_6 = 25$ and $D = 0$	Yes	(1, 6) with $D = 25$
9, 8, 5, 2, 4, 3, 7, 10	'customer 9' with $q_9 = 12$ and $D = 25$	Yes	(1, 6, 9) with $D = 37$
8, 5, 2, 4, 3, 7, 10	'customer 8' with $q_8 = 18$ and $D = 37$	Yes	(1, 6, 9, 8) with $D = 55$
5, 2, 4, 3, 7, 10	'customer 5' with $q_5 = 27$ and $D = 55$	Yes	(1, 6, 9, 8, 5) with $D = 82$
2, 4, 3, 7, 10	'customer 2' with $q_2 = 24$ and $D = 82$	No	(1, 6, 9, 8, 5) with $D = 82$
2, 4, 3, 7, 10	'customer 4' with $q_4 = 20$ and $D = 82$	No	(1, 6, 9, 8, 5) with $D = 82$
2, 4, 3, 7, 10	'customer 3' with $q_3 = 13$ and $D = 82$	Yes	(1, 6, 9, 8, 5, 3) with $D = 95$
2, 4, 7, 10	'customer 7' with $q_7 = 29$ and $D = 95$	No	(1, 6, 9, 8, 5, 3) with $D = 95$
2, 4, 7, 10	'customer 10' with $q_{10} = 0$ and $D = 95$	Yes	(1, 6, 9, 8, 5, 3, 10) with $D = 0$
2, 4, 7	'customer 2' with $q_2 = 24$ and $D = 0$	Yes	(1, 6, 9, 8, 5, 3, 10, 2) with $D = 24$
4, 7	'customer 4' with $q_4 = 20$ and $D = 24$	Yes	(1, 6, 9, 8, 5, 3, 10, 2, 4) with $D = 44$
7	'customer 7' with $q_7 = 29$ and $D = 44$	Yes	(1, 6, 9, 8, 5, 3, 10, 2, 4, 7) with $D = 73$

4.2 Chromosome Evaluation and Selection Operator

The objective function value of a chromosome (solution) is the total service cost of the routes by all vehicles. The cost of every route is the addition of the service costs among the customers. Since the CVRP is a minimization problem, so, the fitness function, f_i, is the inverted objective function, that is,

$$f_i = \frac{1}{1 + o_i}; o_i \textit{ is the objective function value of ith chromosome} \tag{3}$$

In selection operator, a subpopulation (some chromosomes) is selected from the existing population for producing the next population. The performance of GA depends on selection operator without which GA is same as random sampling that produces different results in the generations. Several selection procedures exist in the literature. For our GAs, the fitness proportional selection (FPS) using roulette wheel selection (RWS) [17] is applied. The RWS is widely used method where the fitness of every chromosome is related to the area of roulette wheel portion. Then, the roulette wheel is rotated and the chromosome, which is pointed by the wheel pointer, is selected for next generation

mating pool. Based on the fitness of each chromosome, a probability p_i of selection is calculated as follows:

$$p_i = \frac{f_i}{\sum_{j=1}^{Ps} f_j}; i \in \{1,\ 2,\ \ldots\ldots,\ Ps\} \tag{4}$$

where P_s is the population size. Thus, fitter chromosomes have greater chance of being selected as parents. During the selection procedure, there is no change of the segment size and selection probability. This procedure is very easy to implement that provides unbiased distributed probabilities to the chromosomes and allocates higher probability to the best chromosome.

4.3 Crossover Operators

The selection process provides a consistency between exploration and exploitation of search area. The crossover is the most important process in GAs that is implemented on a chromosome pair to produce offspring(s). Selection and crossover operators together can speed up the convergence of GAs. The fundamental crossover operators cannot produce feasible offspring(s) for the CVRP. The crossover operators which were designed for the usual TSP can be implemented to the CVRP. As the blind crossovers and the distance-based crossover operators are two major kinds of crossover operators, we propose to utilize some of them and then compare them.

4.3.1 Partially Mapped Crossover (PMX) Operator

In [29], the PMX is developed, that describes an exchange mapping in the segment between two selected points. It was the first designed crossover in GA for the TSP. As an example, we choose two parent chromosomes P_1: (1, 6, 9, 8, 5, 3, 10, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 10, 7, 5, 2) with costs 172 and 148 respectively (for costs, see Tab. 2). These same parent chromosomes will be used for demonstrating all crossovers here. It is to be noted that crossover operators are applied on the parents after omitting the dummy depots. So, after omitting the dummy depots, we consider the parents: P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2) Once offsprings are created, dummy depots are inserted into them such that capacity constraint is preserved.

Further, we set headquarters (1st gene) as 'customer 1'. Assume that the arbitrarily selected cut points are between 2nd and 3rd genes, and between 6th and 7th genes which are shown by "|".

P_1: (1, 6 | 9, 8, 5, 3 | 2, 4, 7) and P_2: (1, 8 | 6, 9, 4, 3 | 7, 5, 2).

The first gene is always 'customer 1'. The mapping segments are between these cut points, and the systems are 9↔6, 8↔9, 5↔4, and 3↔3. These segments are copied in offsprings as below:

O_1: (1, ∗ | 9, 8, 5, 3 | ∗, ∗, ∗) and O_2: (1, ∗ | 6, 9, 4, 3 | ∗, ∗, ∗).

We add other genes from the alternative parents that do not lead to infeasible chromosome:

O_1: (1, ∗ | 9, 8, 5, 3 | 7, ∗, 2) and O_2: (1, ∗ | 6, 9, 4, 3 | 2, ∗, 7)

The 1st ∗ in the 1st offspring would be 8 which is from 2nd parent, however, it is already available in the offspring, so we consider 9 using the map 8↔9, but 9 is also available in the offspring, so, we add 6 using the map 9↔6. Similarly, the 2nd ∗ in that offspring would be 5 which is from 2nd parent, however, it is available in the offspring, so we add 4 using the map 5↔4. So, the 1st offspring leads to O_1: (1, 6, 9, 8, 5, 3, 7, 4, 2). Likewise, the 2nd offspring is created as: O_2: (1, 8, 6, 9, 4, 3, 2, 5, 7).

Next, we add dummy depots at the end of the offspring chromosomes. Then offsprings become: O_1: (1, 6, 9, 8, 5, 3, 7, 4, 2, 10) and O_2: (1, 8, 6, 9, 4, 3, 2, 5, 7, 10). Then calculate total demand of the

customers. We have for O_1, $(q_1 + q_6 + q_9 + q_8 + q_5 + q_3) = 0 + 25 + 12 + 18 + 27 + 13 = 95$. Now if we add $q_7 = 29$, then $95 + 29 = 124 > 100$ (=Q), hence, we exchange 'customer 7' with 'customer 10'. So, the first offspring leads to O_1: (1, 6, 9, 8, 5, 3, 10, 4, 2, 7) with cost 162. Likewise, the 2nd offspring is created as: O_2: (1, 8, 6, 9, 4, 3, 10, 5, 7, 2) with cost 123.

4.3.2 Ordered Crossover (OX) Operator

In [30], the OX is designed that creates offspring by selecting a segment of a tour from one parent and maintaining the relative sequence of customers from the other one. Consider the following same parents with two cut points shown by "|":

P_1: (1, 6 | 9, 8, 5, 3 | 2, 4, 7) and P_2: (1, 8 | 6, 9, 4, 3 | 7, 5, 2)

We always fix first gene as 'customer 1'. First, we copy the genes between two cut points to the offspring as:

O_1: (1, ∗ | 9, 8, 5, 3 | ∗, ∗, ∗) and O_2: (1, ∗ | 6, 9, 4, 3 | ∗, ∗, ∗)

Next, starting from 2nd cut point of one parent, genes from other one, excluding existing ones, are copied by maintaining sequence. The order of genes in 2nd parent from 2nd cut point is "7→5→2→8→6→9→4→3." After excluding existing genes 9, 8, 5 and 3, the order becomes "7→2→6→4", that is put in 1st offspring starting from 2nd cut point as: O_1: (1, 4 | 9, 8, 5, 3 | 7, 2, 6). Similarly, we build the second offspring as: O_2: (1, 5 | 6, 9, 4, 3 | 2, 7, 8).

Next, we add dummy depots at the end of the offspring chromosomes. Then offsprings become: O_1: (1, 4, 9, 8, 5, 3, 7, 2, 6, 10) and O_2: (1, 5, 6, 9, 4, 3, 2, 7, 8, 10). Then calculate total demand of the customers. We have for O_1, $(q_1 + q_4 + q_9 + q_8 + q_5 + q_3) = 0 + 20 + 12 + 18 + 27 + 13 = 90$. Now if we add $q_7 = 29$, then $90 + 29 = 119 > 100$ (=Q), hence, we exchange 'customer 7' with 'customer 10'. So, the first offspring becomes O_1: (1, 4, 9, 8, 5, 3, 10, 2, 6, 7) with cost 169. Similarly, we build the second offspring as: O_2: (1, 5, 6, 9, 4, 3, 10, 7, 8, 2) with cost 142.

4.3.3 Alternating Edges Crossover (AEX) Operator

In [31], the AEX is proposed, that presumes chromosome as a directed cycle of arcs. It produces only one offspring by choosing alternative arcs from the parents. However, in case of infeasibility, random gene is added to the offspring. Consider the following same parents: P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2).

Initially, the arc (1, 6) is chosen from 1st parent and copied to the offspring. Then arcs (6, 9) from 2nd one, and (9, 8) from 1st one, are chosen and copied to the offspring. Next, since the chosen arc (8, 6) from 2nd one makes a cycle, so, a random arc (8, 2) is chosen. Later, the arcs (2, 4) from 1st parent and (4, 3) from 2nd one, are chosen and copied to the offspring. Finally, the arcs (3, 5) and (5, 7) are chosen randomly. This way the following offspring is created: O: (1, 6, 9, 8, 2, 4, 3, 5, 7).

Next, we add dummy depots at the end of the offspring chromosome which becomes: O: (1, 6, 9, 8, 2, 4, 3, 5, 7, 10). Then calculate total demand of the customers. So, we have, $(q_1 + q_6 + q_9 + q_8 + q_2 + q_4) = 0 + 25 + 12 + 18 + 24 + 20 = 99$. Now if we add $q_3 = 13$, then $99 + 13 = 112 > 100$ (=Q), hence, we exchange 'customer 3' with 'customer 10'. Finally, the offspring leads to O: (1, 6, 9, 8, 2, 4, 10, 5, 7, 3) with cost 137.

4.3.4 Cycle Crossover (CX) Operator

In [32], CX is developed, which creates an offspring where each gene and its related position derived from any parent. Consider the same example parent chromosomes P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2).

The first position in the offspring 1, for 2nd position, we choose 'customer 6' from 1st parent, then the 1st offspring leads to: O_1: (1, 6, ∗, ∗, ∗, ∗, ∗, ∗, ∗).

In the offspring, each gene is selected from any parent with the same location, so gene 8 must be considered, since gene 8, in 2nd parent, is below gene 6. In 1st parent, gene 8 is at 4th location; so, the offspring leads to: O_1: (1, 6, ∗, 8, ∗, ∗, ∗, ∗, ∗).

Next, we select gene 9 as, in 2nd parent, it is below gene 8. In 1st parent, gene 9 is at 3rd location; so, the offspring leads to: O_1: (1, 6, 9, 8, ∗, ∗, ∗, ∗, ∗).

Next, we select gene 6 as, in 2nd parent, it is below gene 9. However, it leads to a cycle. So, we fill up the empty positions by the genes available in corresponding positions in 2nd parent. So, the offspring leads to: O_1: (1, 6, 9, 8, 4, 3, 7, 5, 2). Similarly, we build the second offspring as: O_2: (1, 8, 6, 9, 5, 3, 2, 4, 7).

Next, we add dummy depots at the end of the offspring chromosomes. Then offsprings become: O_1: (1, 6, 9, 8, 4, 3, 7, 5, 2, 10) and O_2: (1, 8, 6, 9, 5, 3, 2, 4, 7, 10). Then calculate total demand of the customers. We have for O_1, $(q_1 + q_6 + q_9 + q_8 + q_4 + q_3) = 0 + 25 + 12 + 18 + 20 + 13 = 88$. Now if we add $q_7 = 29$, then $88 + 29 = 117 > 100$ (=Q), hence, we exchange 'customer 7' with 'customer 10'. So, the first offspring becomes O_1: (1, 6, 9, 8, 4, 3, 10, 5, 2, 7) with cost 164.

Similarly, we build the second offspring as: O_2: (1, 8, 6, 9, 5, 3, 10, 4, 7, 2) with cost 144.

4.3.5 Greedy Crossover (GX) Operator

In [33], GX is developed for the TSP, which creates only one offspring. It chooses 1st gene randomly, then, in every step, it considers four neighbor genes of present gene in both parents and chooses the cheapest gene that is not available in the offspring. If the cheapest one or all these neighbour genes are available in the offspring, any other valid gene is chosen randomly. Consider the same example parent chromosomes P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2).

As the offspring is started with (1), its neighbors in both parents are 6 and 8 with their costs 17 and 9 respectively. As the customer 8 is the cheapest one, so, it is added to the offspring that leads to: (1, 8).

Next, the customer 8 has neighbours 5, 9, 6 and 1 with their costs 13, 28, 9 and 15 respectively. As the customer 6 is the cheapest one, so, it is added to the offspring that leads to: (1, 8, 6).

Next, the customer 6 has neighbours 9, 1, 9 and 8 with their costs 22, 17, 22 and 15 respectively. The customer 8 is the cheapest one, but it is available in the offspring. So, customer 2 is chosen randomly and added to the offspring that leads to: (1, 8, 6, 2).

Next, the customer 2 has neighbours 4, 3 and 5 with their costs 9, 6 and 21 respectively. The customer 3 is the cheapest one, and so, it is added to the offspring that leads to: (1, 8, 6, 2, 3).

Next, the customer 3 has neighbours 2, 5, 7 and 4 with their costs 7, 13, 9 and 16 respectively. The customer 2 is the cheapest one, but it is available in the offspring. So, customer 4 is chosen randomly and added to the offspring that leads to: (1, 8, 6, 2, 3, 4). This way, we create the complete offspring as: O: (1, 8, 6, 2, 3, 4, 5, 7, 9).

Next, we add dummy depots at the end of the offspring chromosome which becomes: O: (1, 8, 6, 2, 3, 4, 5, 7, 9, 10). Then calculate total demand of the customers. So, we have, ($q_1 + q_8 + q_6 + q_2 + q_3 + q_4$) = 0 + 18 + 25 + 24 + 13 + 20 = 100. Now if we add $q_5 = 27$, then 100 + 27 = 127 > 100 (=Q), hence, we exchange 'customer 5' with 'customer 10'. So, the offspring becomes O: (1, 8, 6, 2, 3, 4, 10, 7, 9, 5) with cost 143.

4.3.6 Heuristic Crossover (HX) Operator

In [33], the HX operator is developed that creates only one offspring as follows. It picks a starting customer randomly, then creates a tour by looking at the costs of the arcs leaving that customer in the parents. It adds the lowest cost arc which does not create a cycle. If the lowest cost arc makes a cycle, select a customer randomly which does not create a cycle. We illustrate the HX using same example parent chromosomes P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2).

As the offspring is started with (1), we look at the arcs 1→6 and 1→8 with costs 17 and 9 respectively in the parents and add customer 8 to the offspring that leads to: (1, 8).

Next, we look at the arcs 8→5 and 8→6 with their respective costs 13 and 9 in the parents and add customer 6 to the offspring that leads to: (1, 8, 6).

Next, we look at the arcs 6→9 and 6→9 in the parents and add customer 9 to the offspring that leads to: (1, 8, 6, 9). This way, we create the complete offspring as: (1, 8, 6, 9, 4, 2, 3, 7, 5).

Next, we add dummy depots at the end of the offspring chromosome which becomes: O: (1, 8, 6, 9, 4, 2, 3, 7, 5, 10). Then calculate total demand of the customers. So, we have, ($q_1 + q_8 + q_6 + q_9 + q_4 + q_2$) = 0 + 18 + 25 + 12 + 20 + 24 = 99. Now if we add $q_3 = 13$, then 99 + 13 = 112 > 100 (=Q), hence, we exchange 'customer 3' with 'customer 10'. So, the offspring becomes O: (1, 8, 6, 9, 4, 2, 10, 7, 5, 3) with cost 143.

4.3.7 Modified Heuristic Crossover (MHX) Operator

In [34], a modified HX (MHX) is described that creates only one offspring as follows. Select a gene arbitrarily as the 1st gene in the offspring chromosome. Evaluate two arcs going out from the 1st gene in the parents and add the cheaper one to the offspring. Keep on adding cheaper arc to the offspring. If the cheaper one creates a cycle in the offspring, and 2nd one does not create a cycle, add 2nd one to the offspring. Else, add the cheapest one from a set of at most 20 arbitrarily chosen arcs that do not create a cycle. Keep on until a complete offspring is obtained and replace the 1st parent chromosome. Consider the same example parent chromosomes P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2).

As the offspring is started with (1), we look at the arcs 1→6 and 1→8 with costs 17 and 9 respectively in the parents and add customer 8 to the offspring that leads to: (1, 8).

Next, we look at the arcs 8→5 and 8→6 with their respective costs 13 and 9 in the parents and add customer 6 to the offspring that leads to: (1, 8, 6).

Next, the arcs leaving the customer 6 are considered, i.e., both 6→9 with cost 22, so, add customer 9 to the offspring that leads to: (1, 8, 6, 9).

Next, between the arcs 9→8 and 9→4, the 1st creates a cycle, so, add customer 4 to the offspring that leads to: (1, 8, 6, 9, 4). Continuing in this way, one can obtain a complete offspring as: (1, 8, 6, 9, 4, 7, 5, 2, 3).

Next, we add dummy depots at the end of the offspring chromosome which becomes: O: (1, 8, 6, 9, 4, 7, 5, 2, 3, 10). Then calculate total demand of the customers. So, we have, ($q_1 + q_8 + q_6 + q_9 + q_4$) = 0 + 18 + 25 + 12 + 20 = 75. Now if we add $q_7 = 29$, then 75 + 29 = 104 > 100 (=Q), hence, we exchange 'customer 7' with 'customer 10'. So, the offspring becomes O: (1, 8, 6, 9, 4, 10, 5, 2, 3, 7) with cost 119.

4.3.8 Sequential Constructive Crossover (SCX) Operator

In [25], the SCX operator is proposed that creates only one offspring. It sequentially investigates parents and considers 1st legitimate gene (i.e., unvisited gene) that is found after the current gene in each parent. If no legitimate gene is discovered in a parent, it sequentially investigates from the starting of the parent and choose the legitimate gene. It then evaluates the cost of each gene and add the better one to the offspring. Algorithm 2 reports the algorithm for the SCX.

Algorithm 2: *Sequential constructive crossover algorithm*

Input: *Number of customers n; Number of vehicles m; Capacity of vehicles Q; Demands of customers q_i; Crossover probability P_c; Pair of chromosomes without dummy depots; Cost matrix $C = [c_{ij}]$.*
Output: *Offspring chromosome.*
Generate random number $r \in [0,1]$.
if *($r <= P_c$) then* ***do***
  Set *p = 1.*
  The offspring chromosome contains only 'customer 1'.
  for *i = 2* ***to*** *n* ***do***
    In each chromosome consider the first 'legitimate customer' appeared after 'customer p'.
    if *'legitimate customer' is not available in a chromosome,* ***then***
      Search from its 1st gene and select the first 'legitimate customer' appeared after 'customer p'.
    endif
    Suppose 'customer α' and 'customer β' are selected in 1st and 2nd chromosomes respectively.
    if *($c_{p\alpha} < c_{p\beta}$) then* ***do***
      Add 'customer α' to the offspring chromosome.
    else
      Add 'customer β' to the offspring chromosome.
    endif
    Rename the present customer as 'customer p' and continue.
  endfor
endif
n=n+m-1;
Add required dummy depots at the end of the offspring chromosome
Start from the beginning from the chromosome, consider the customer in the present vehicle until total demand of the customers does not exceed capacity of the vehicle.
Return *the offspring chromosome with dummy depots*

Consider the same example parent chromosomes P_1: (1, 6, 9, 8, 5, 3, 2, 4, 7) and P_2: (1, 8, 6, 9, 4, 3, 7, 5, 2).

Initially, the offspring is (1). The legitimate customers after customer 1 in the parents are 6 and 8 with their costs 17 and 9 respectively. Since customer 8 is cheaper, we add it to the offspring that leads to: (1, 8).

The legitimate customers after customer 8 in the parents are 5 and 6 with their costs 13 and 9 respectively. Since customer 6 is cheaper, we add it to the offspring that leads to: (1, 8, 6).

Since the legitimate customer after customer 6 in the parents are same 9, we add it to the offspring that leads to: (1, 8, 6, 9).

The legitimate customers after customer 9 in the parents are 5 and 4 with their costs 9 and 14 respectively. Since customer 5 is cheaper, we add it to the offspring that leads to: (1, 8, 6, 9, 5).

The legitimate customers after customer 5 in the parents are 3 and 2 with their costs 21 and 15 respectively. Since customer 2 is cheaper, we add it to the offspring that leads to: (1, 8, 6, 9, 5, 2). This way, we create the complete offspring as: (1, 8, 6, 9, 5, 2, 4, 7, 3).

Next, we add dummy depots at the end of the offspring chromosome which becomes: O: (1, 8, 6, 9, 5, 2, 4, 7, 3, 10). Then calculate total demand of the customers. So, we have, $(q_1 + q_8 + q_6 + q_9 + q_5) = 0 + 18 + 25 + 12 + 27 = 82$. Now if we add $q_2 = 24$, then $82 + 24 = 106 > 100$ (=Q), hence, we exchange 'customer 2' with 'customer 10'. So, the offspring becomes O: (1, 8, 6, 9, 5, 10, 4, 7, 3, 2) with cost 139.

Amongst the above eight crossovers GX, HX, MHX and SCX are the distance-based crossovers, and OX, PMX, CX and AEX are the blind crossovers. We propose to compare both categories of crossovers for the CVRP.

4.4 Mutation Operator

To diversify the population, mutation operator is applied with a prespecified probability. Generally, mutation probability is set very low compared to crossover probability. The exchange mutation that chooses randomly two places in a chromosome within a route of any vehicle and exchanges their values. If there are *m* vehicles, there will be *m* exchanges. This mutation will always produce legal chromosome. For example, let the Chromosome: (1, 8, 6, 2, **3**, **4**, 10, **7**, **9**, 5) with cost 143 is allowed for the mutation. Suppose 5^{th} and 6^{th} position values (route of first vehicle), and 8^{th} and 9^{th} position values (route of second vehicle) in the chromosome are swapped. Then the muted chromosome will be Muted: (1, 8, 6, 2, **4**, **3**, 10, **9**, **7**, 5) with cost 138. However, we do not see whether the value of muted chromosome is better than the original chromosome.

4.5 Design of GAs

Our simple GA for the CVRP is presented in Algorithm 3.

We consider four blind crossovers and four distance-based crossovers. In each crossover selection, a single crossover is used. For mutation, two possibilities of selection–absence or presence of mutation, are applied. So, there are eight choices for crossover and two choices of mutation, thus provides sixteen variations of GAs. The aim of such individual implementation is to evaluate usefulness of the crossovers and to find their relative ranking. Notice that each GA is solely simple, not hybrid, that is developed of GA processes and basic operators, is not merging any other algorithm.

Basically, GA process is led by some GA parameters, specifically, population size which fixes number of chromosomes in every population, crossover probability for executing crossover between

the parents, mutation probability for performing mutation operation, and a stopping criterion to end the GA process.

Algorithm 3: *Simple genetic algorithm*

Input: *Crossover probability* P_c; *Mutation probability* P_m; *Maximum generation MaxGen.*
Output: *Best tour and its total cost*
G_0 = *Initial population using Algorithm 1*
Evaluate (G_0)
BS = Best solution
BT = Best tour
i = 1
while (*i <= MaxGen*) ***do***
    G_i = *Population after using selection method*
    G_i = *Population after using a crossover operator with probability* P_c
    G_i = *Population after using a mutation operator (optional) with probability* P_m
    Evaluate (G_i);
    B_i = *Best solution in the current generation*
    if (B_i < *BS*) ***then***
        BS = B_i
        BT = Best tour
    end if
    i = i + 1
end while
The best tour and its total cost (best solution) are given by BT and BS respectively.

5 Computational Results

For comparing the effectiveness of various crossovers, simple GAs using various crossovers were encoded in Visual C++ on a Laptop with i7-1065G7 CPU@1.30 GHz and 8 GB RAM under MS Windows 10. We executed our GAs using various parameter sets, and finally, the parameters showed in Tab. 4 are chosen.

Table 4: GA parameter values

Parameters	Values
Population size	50
Crossover probability	100%
Mutation probability	10%
Termination criterion	5000 generations
No. of runs for each instance	50 times

To verify the performance of crossovers, computational experiment is made on sixteen benchmark instances of many sizes. In these sixteen instances, A034-02f, A036-03f, A039-03f, A045-03f, A048-03f,

A056-03f, A065-03f and A071-03f are asymmetric [35] , and E-n22-k4, E-n51-k5, E-n76-k7, E-n76-k8, E-n76-k10, E-n76-k14, E-n101-k8 and E-n101-k14 are symmetric [36]. The details of these sixteen problem instances are given in Tab. 5, where n is the number of customers (including depot), m is the number of vehicles, Q is the capacity of the vehicles, and BKS is the best-known/optimal solution.

Table 5: Details of some CVRP instances

Instance	n	m	Q	BKS	Instance	n	m	Q	BKS
A034-02f	34	2	1000	1406	E-n22-k4	22	4	6000	375
A036-03f	36	3	1000	1644	E-n51-k5	51	5	160	521
A039-03f	39	3	1000	1654	E-n76-k7	76	7	220	682
A045-03f	45	3	1000	1740	E-n76-k8	76	8	180	735
A048-03f	48	3	1000	1891	E-n76-k10	76	10	140	830
A056-03f	56	3	1000	1739	E-n76-k14	76	14	100	1021
A065-03f	65	3	1000	1974	E-n101-k8	101	8	200	815
A071-03f	71	3	1000	2054	E-n101-k14	101	14	112	1067

The experimental results by the sixteen GAs are summarized in Tabs. 6, 8, 10 and 12. All these tables are structured in the same way: each row is for an instance (its optimal or best-known solution is written within brackets) and each column is for a simple GA using one crossover. So, a table element reports brief results of subsequent instance by subsequent GA. Each result is defined using best solution, average solution, average percentage of excess to the best-known solution, standard deviation (SD) of solutions, and average computational time (in second). For each instance, the best result over all GAs is indicated by boldface. The percentage of excess to the best-known solution, stated in various literatures, is calculated by following formula.

$$Excess\ (\%) = \frac{Sol.\ Obtained - Best - KnownSol.}{Best\ KnownSol} x100 \tag{5}$$

Figs. 3 and 4 show results for the asymmetric instance A071-03f (considering only 100 generations). Fig. 3 is for the GAs without mutation operator, and Fig. 4 is for the GAs with mutation operator. In these figures, each graph relates to one crossover that indicates how the solution improves in consecutive generations. In these figures, label on left side denotes the percentage of excess to the best-known solution (Excess (%)). When seeking to combine an allele in chromosome, all crossovers have some arbitrary factors that can make them further efficient.

Fig. 3 shows that among blind-crossovers, CX has initial variation within first 5 generations, after which it shows no variation, and so, it is found to be the worst one. Among remaining three blind-crossovers, the crossovers PMX and OX have shown good variation throughout the generations, however, they show slow improvement. The crossover AEX has shown very good variation within first 20 generations, after which it shows little variation and slow improvement. Of course, among the blind-crossovers, it is found to be the best one. The Figure also shows that among distance-based crossovers, GX has shown good initial variation within first 5 generations, after which it shows no variation, and so, it is found to be the worst among the distance-based crossover. The crossovers HX and MHX have shown good variation within first 15 generations, after which they show no variation. The figure for the crossover SCX shows that it starts the search process with better initial solutions, has shown good

variation within first 25 generations, after which it shows no variation. Though SCX gets stuck in local minima after 25 generations, however, it is observed to be the best one among all crossover operators.

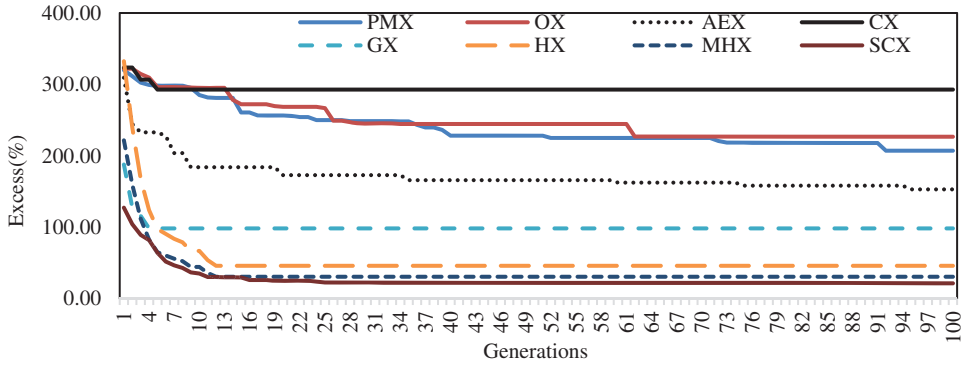

Figure 3: Performance of eight crossover operators without mutation for the instance A071-03f

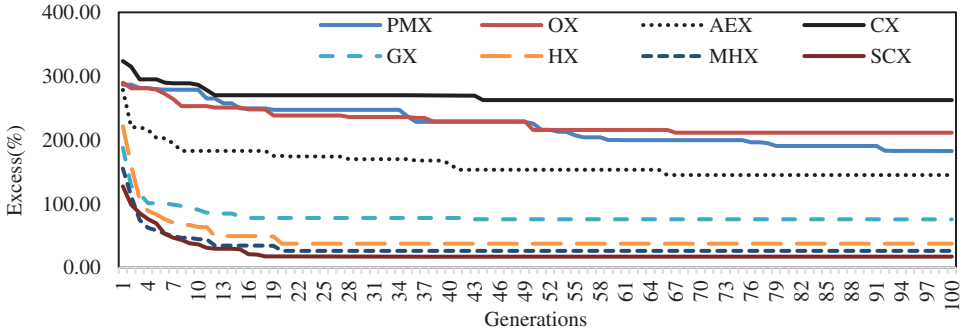

Figure 4: Performance of eight crossover operators with mutation for the instance A071-03f

Fig. 4 shows that among blind-crossovers, PMX, OX and CX, each with mutation, have initial variations, and are competing within first 50 generations, after which CX shows no variation, but PMX and OX have small variations. So, CX is found to be the worst one. The crossover AEX shows very good variation within first 5 generations, has variation up to 67 generations, and is found to be best one among blind crossovers with mutation. Among distance-based crossovers with mutation, GX has initial variation within first 20 generations, after which it shows no variation. The remaining three distance-based crossovers have good variations within first 25 generations, and after that they are competing each other, and finally, SCX with mutation is observed to be best one among all crossovers. So, in both cases–with and without mutation, the crossover SCX is found to be the best one and CX is found to be the worst one. Further, it is seen that mutation operator enhances performance of crossovers by avoiding local minimum.

Tab. 6 describes the results on eight asymmetric instances by the eight GAs with no mutation operator. Based on the average solution and SD, the results are evaluated. With regard to the average solution, the distance-based crossovers are found much better than the blind crossovers. Among the blind crossovers, PMX and CX could not find lowest average solution for any of experimented eight instances. Between the crossovers PMX and CX, the crossover PMX is found to be better. So, the crossover CX is the worst one. The crossover OX obtained lowest solution for only one instance,

A034-02f, and AEX obtained lowest solution for the remaining seven instances. So, amongst the blind crossovers, AEX is observed to be the best one and CX is the worst one. However, CX takes less time.

Table 6: Results by the GAs without mutation for asymmetric instances

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
A034-02f	Best Sol	1965	1751	1806	3065	1679	1584	1485	1486
(1406)	Avg. Sol	2252.40	1994.15	2054.50	3204.85	1746.15	1603.20	**1503.20**	1503.80
	AvgExc(%)	60.20	41.83	46.12	127.94	24.19	14.03	**6.91**	6.96
	SD	157.78	233.88	172.82	143.03	83.45	95.62	76.16	87.98
	Avg. Time	0.02	0.06	0.05	0.01	0.02	0.05	0.06	0.00
A036-03f	Best Sol	2325	2338	1991	3342	1975	1835	1792	1731
(1644)	Avg. Sol	2568.18	2532.28	2295.68	3660.34	2136.48	1990.40	1898.40	**1797.55**
	AvgExc(%)	56.22	54.03	39.64	122.65	29.96	21.07	15.47	**9.34**
	SD	198.81	128.53	163.68	181.56	119.24	94.21	80.33	82.63
	Avg. Time	0.14	0.25	0.30	0.01	0.06	0.16	0.17	0.16
A039-03f	Best Sol	2665	2659	2126	3802	2057	1905	1838	1801
(1654)	Avg. Sol	2849.50	2823.72	2327.88	4003.66	2202.06	2015.02	1917.02	**1864.66**
	AvgExc(%)	72.28	70.72	40.74	142.06	33.14	21.83	15.90	**12.74**
	SD	189.17	123.60	121.71	157.09	156.66	115.23	106.45	79.99
	Avg. Time	0.16	0.34	0.54	0.01	0.16	0.12	0.13	0.28
A045-03f	Best Sol	2903	3207	2427	4444	2318	2143	2010	1943
(1740)	Avg. Sol	3297.55	3490.30	2757.76	4793.11	2517.04	2227.04	2147.04	**1999.05**
	AvgExc(%)	89.51	100.59	58.49	175.47	44.66	27.99	23.39	**14.89**
	SD	250.77	144.36	163.38	157.08	93.37	105.77	114.38	141.53
	Avg. Time	0.15	0.28	0.36	0.00	0.05	0.14	0.15	0.22
A048-03f	Best Sol	3131	3487	3258	4968	2982	2335	2220	2152
(1891)	Avg. Sol	3729.96	3814.04	3653.22	5383.92	3204.96	2587.05	2472.10	**2199.60**
	AvgExc(%)	97.25	101.69	93.19	184.71	69.48	36.81	30.73	**16.32**
	SD	315.73	185.55	284.48	244.58	115.78	123.06	147.41	127.58
	Avg. Time	0.19	0.32	0.33	0.01	0.11	0.27	0.29	0.45
A056-03f	Best Sol	3703	3828	3414	5415	2679	2105	1980	1924
(1739)	Avg. Sol	4146.94	4121.34	3760.16	5848.96	2989.24	2358.06	2145.40	**2048.15**
	AvgExc(%)	138.47	136.99	116.23	236.34	71.89	35.60	23.37	**17.78**
	SD	281.57	192.53	254.08	251.54	198.22	105.23	117.47	130.14
	Avg. Time	0.19	0.42	0.59	0.01	0.27	0.24	0.26	0.37
A065-03f	Best Sol	4779	4807	3870	6644	3354	2506	2356	2260
(1974)	Avg. Sol	5069.92	5065.78	4386.82	7038.26	3564.58	2614.32	2469.60	**2330.35**
	AvgExc(%)	156.83	156.63	122.23	256.55	80.58	32.44	25.11	**18.05**
	SD	265.73	229.29	283.32	226.75	186.82	152.04	111.18	155.59
	Avg. Time	0.33	0.71	0.97	0.06	0.30	0.06	0.10	0.15

(Continued)

Table 6: Continued

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
A071-03f (2054)	Best Sol	5037	5161	3755	7439	3329	2687	2452	2446
	Avg. Sol	5544.00	5472.96	4293.26	7901.64	3522.92	2695.03	2554.42	**2459.78**
	AvgExc(%)	169.91	166.45	109.02	284.70	71.52	31.21	24.36	**19.76**
	SD	304.82	293.10	298.67	275.55	185.07	152.03	121.67	171.86
	Avg. Time	0.34	0.96	1.07	0.02	0.54	0.56	0.64	0.97

Amongst the distance-based crossovers, HX and GX could not find lowest average cost for any of experimented eight instances. Between GX and HX, HX is found better than GX. So, GX is observed to be the worst one among distance-based crossovers, yet GX is found better than AEX. The crossover MHX obtained lowest cost for only one instance, A034-02f, SCX obtained lowest cost for the remaining seven instances. So, among distance-based crossovers, SCX is observed to be best one. In fact, among all eight crossovers, SCX is observed to be best one. In addition, since the solutions obtained by SCX has lowest SD, one can conclude that the results by SCX is stable. Next, if one looks very carefully, then one can tell that MHX is the 2nd best and CX is the worst one. Further, the results of GX, HX, MHX and SCX are depicted in Fig. 5, which also confirms our conclusion.

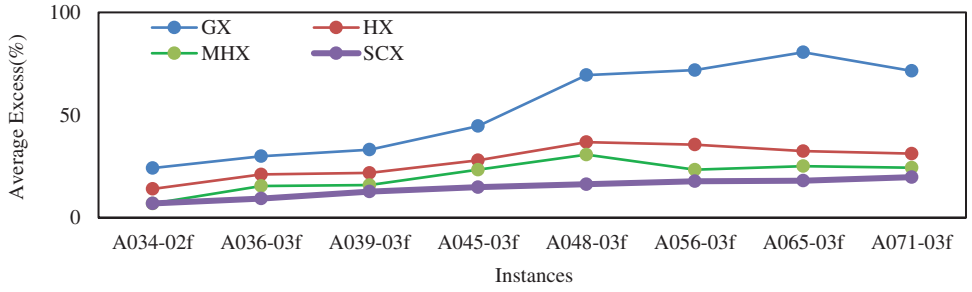

Figure 5: Average Excess (%) by GAs without mutation for asymmetric instances

To verify if GA using SCX (with no mutation) found much different average solution from the average solutions found by other GAs using distance-based crossovers, we conducted Student's t-test. Further, we verify whether GA using AEX found much different average solution from the average solutions found by other GAs using blind crossovers. We applied following t-test formula for two large independent samples [37]. The values of $\overline{X}_2$ and SD_2 are obtained by a GA using a particular crossover, while of $\overline{X}_1$ and SD_1 values are obtained by its competing GAs.

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{SD_1^2}{n_1 - 1} + \frac{SD_2^2}{n_2 - 1}}}$$

where,

$\overline{X}_1 - average\ of\ first\ sample$,

$SD_1 - standard\ deviation\ of\ first\ sample$,

$\overline{X}_2 - average\ of\ second\ sample$,

$SD_2 -$ *standard deviation of second sample*,

$n_1 -$ *first sample size*,

$n_2 -$ *second sample size*,

Tab. 7 reports the t-statistic values that may be negative or positive. The positive value suggests that a particular crossover-based GA found better solution than its competing GA variant. If the value is negative, the competing algorithm found better solution. The confidence interval is applied at 95% confidence level ($t_{0.05} = 1.96$). If t-value is more than 1.96 and positive, the two values have much difference, and so, the GA using the particular crossover found better solution. If t-value is more than 1.96 and negative, the competing GA found better solution. If t-value is less than 1.96, and positive or negative, the algorithms could not find different solutions. The table further concludes about which crossover is better.

Table 7: The t-statistic values (GAs without mutation) and the information about the crossovers that found significantly better solutions for asymmetric instances

Instance	t-values against AEX			t-values against SCX		
	PMX	OX	CX	GX	HX	MHX
A034-02f	5.92	−1.45	35.90	13.99	5.35	−0.04
Better	AEX	—	AEX	SCX	SCX	—
A036-03f	7.41	7.96	39.08	16.35	10.77	6.13
Better	AEX	AEX	AEX	SCX	SCX	SCX
A039-03f	16.23	20.01	59.03	13.43	7.50	2.75
Better	AEX	AEX	AEX	SCX	SCX	SCX
A045-03f	12.62	23.52	62.86	21.39	9.03	5.69
Better	AEX	AEX	AEX	SCX	SCX	SCX
A048-03f	1.26	3.31	32.29	40.85	15.30	9.78
Better	—	AEX	AEX	SCX	SCX	SCX
A056-03f	7.14	7.93	40.90	27.78	12.96	3.88
Better	AEX	AEX	AEX	SCX	SCX	SCX
A065-03f	12.31	13.04	51.15	35.54	9.14	5.10
Better	AEX	AEX	AEX	SCX	SCX	SCX
A071-03f	20.52	19.73	62.16	29.47	7.18	3.15
Better	AEX	AEX	AEX	SCX	SCX	SCX

According to the results shown in Tab. 7, when comparing AEX against blind crossovers, we found that AEX is better than CX on all eight instances. Further, AEX is better than PMX and OX on seven instances and for only one instance, AEX and PMX have no difference, and AEX and OX have no difference. So, AEX is the best crossover and CX is the worst one among the blind crossover operators. About the distance-based crossovers, on all instances SCX is better than GX and HX. For one instance only, SCX and MHX have no difference, and SCX is better than MHX on the remaining seven instances. So, SCX is the best one and GX is the worst one among the distance-based crossovers.

Further, SCX is compared with all other crossovers and found that SCX is the best one, however it is not reported.

Tab. 8 describes the results on asymmetric instances by all GAs using mutation. With respect to the average solution, it is very clear from Tab. 8 also that the distance-based crossovers are far better than the blind crossovers. Among the blind crossovers, PMX, OX and CX could not find lowest average solution for any of experimented eight instances. The crossover AEX obtained lowest solutions for all eight instances. So, among the blind crossovers, AEX is the best one and CX is the worst one for both GAs with and without mutation for these asymmetric instances. Among the distance-based crossovers, GX, HX and MHX could not find lowest average solution for any of experimented eight instances. SCX obtained lowest solutions for all instances, and hence it is the best. In addition, since the solutions obtained by SCX has lowest SD, one can conclude that the results by SCX is stable. Though GX is the worst among the distance-based crossovers, however, it is better than AEX for both GAs with and without mutation operator. The results of GX, HX, MHX and SCX are further displayed in Fig. 6, which shows that SCX is the best one, MHX is better than HX, and HX is better than GX.

Table 8: Results by the GAs with mutation for asymmetric instances

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
A034-02f	Best Sol	1575	1559	1619	1883	1562	1521	1414	1414
(1406)	Avg. Sol	1874.24	1799.34	1702.16	2090.88	1634.64	1531.21	1448.56	**1446.08**
	AvgExc(%)	33.30	27.98	21.06	48.71	16.26	8.91	3.03	**2.85**
	S.D.	149.12	146.13	92.49	136.03	76.17	72.24	72.24	84.76
	Avg. Time	0.13	0.36	0.36	0.29	0.49	0.37	0.38	0.56
A036-03f	Best Sol	2052	2299	1898	2440	1911	1766	1694	1694
(1644)	Avg. Sol	2268.78	2513.32	2085.98	2779.42	1976.26	1890.32	1792.42	**1726.58**
	AvgExc(%)	38.00	52.88	26.88	69.06	20.21	14.98	9.03	**5.02**
	S.D.	158.32	135.90	107.26	152.63	85.21	77.51	64.29	69.60
	Avg. Time	0.17	0.41	0.59	0.55	0.64	0.62	0.66	0.71
A039-03f	Best Sol	2241	2436	1977	2821	1886	1825	1768	1749
(1654)	Avg. Sol	2570.38	2792.60	2204.58	3132.20	2032.00	1874.21	1788.16	**1766.18**
	AvgExc(%)	55.40	68.84	33.29	89.37	22.85	13.31	8.11	**6.78**
	S.D.	149.75	122.55	81.83	225.74	121.64	73.15	73.15	48.76
	Avg. Time	0.19	0.38	0.56	0.55	0.33	0.51	0.54	0.60
A045-03f	Best Sol	2870	3200	2454	3208	2291	1954	1875	1858
(1740)	Avg. Sol	3004.96	3405.90	2562.88	3640.22	2263.76	2064.22	1953.56	**1875.04**
	AvgExc(%)	72.70	95.74	47.29	109.21	30.10	18.63	12.27	**7.76**
	S.D.	159.55	153.31	138.13	182.69	68.58	91.21	77.20	69.64
	Avg. Time	0.29	0.40	0.51	0.44	0.57	0.88	0.90	1.27
A048-03f	Best Sol	2898	3361	2891	3722	2631	2201	2017	2001
(1891)	Avg. Sol	3174.40	3750.01	3014.40	3960.64	2836.05	2287.21	2104.64	**2080.90**
	AvgExc(%)	67.87	98.31	59.41	109.45	49.98	20.95	11.30	**10.04**
	S.D.	269.54	187.91	186.75	175.65	142.39	102.54	95.20	81.66

(Continued)

Table 8: Continued

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
	Avg. Time	0.24	0.45	0.62	0.49	1.23	0.95	1.00	0.90
A056-03f	Best Sol	3362	3816	3050	4117	2450	2045	1857	1831
(1739)	Avg. Sol	3488.66	4095.44	3363.64	4385.42	2576.94	2094.62	2013.50	**1919.48**
	AvgExc(%)	100.61	135.51	93.42	152.18	48.19	20.45	15.78	**10.38**
	S.D.	172.62	177.55	281.92	230.87	190.61	112.02	93.94	91.93
	Avg. Time	0.34	0.91	1.20	1.32	1.72	1.45	1.52	1.96
A065-03f	Best Sol	4161	4796	3965	5302	3060	2250	2042	2042
(1974)	Avg. Sol	4491.94	5039.28	4152.68	5620.22	3152.58	2378.21	2261.64	**2196.20**
	AvgExc(%)	127.56	155.28	110.37	184.71	59.71	20.48	14.57	**11.26**
	S.D.	225.72	205.09	239.01	199.42	134.07	110.54	82.20	113.73
	Avg. Time	0.51	0.90	1.31	1.16	1.86	1.41	1.49	1.86
A071-03f	Best Sol	4500	5126	3662	5369	2883	2339	2206	2206
(2054)	Avg. Sol	4719.48	5436.40	4061.74	5868.64	3025.70	2544.12	2351.18	**2295.75**
	AvgExc(%)	129.77	164.67	97.75	185.72	47.31	23.86	14.47	**11.77**
	S.D.	200.98	234.93	332.35	279.62	175.79	120.32	93.88	109.87
	Avg. Time	0.60	1.43	2.18	2.13	3.65	1.95	1.99	2.53

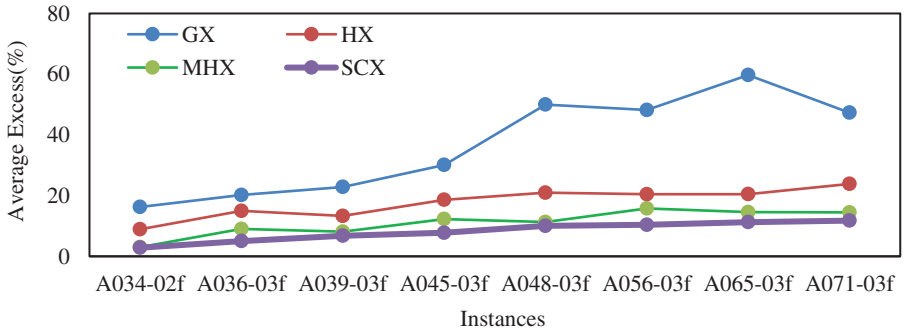

Figure 6: Average Excess (%) by GAs with mutation for asymmetric instances

Further, to verify if GA using SCX (including mutation) found much different average solution from the average solutions found by other GAs using distance-based crossovers, we conducted Student's t-test. Further, we verify whether GA using AEX found much different average solution from the average solutions found by other GAs using blind crossovers. The t-statistic values are reported in the Tab. 9.

Table 9: The t-statistic values (GAs with mutation) and the information about the crossovers that found significantly better solutions for asymmetric instances

Instance	t-values against AEX			t-values against SCX		
	PMX	OX	CX	GX	HX	MHX
A034-02f	6.86	3.93	16.54	11.58	5.35	0.16
Better	AEX	AEX	AEX	SCX	SCX	—
A036-03f	6.69	19.70	26.02	15.89	11.00	4.86
Better	AEX	AEX	AEX	SCX	SCX	SCX
A039-03f	15.01	32.68	27.04	14.20	8.60	1.75
Better	AEX	AEX	AEX	SCX	SCX	—
A045-03f	14.66	28.60	32.93	27.84	11.54	5.29
Better	AEX	AEX	AEX	SCX	SCX	SCX
A048-03f	3.42	19.44	25.84	32.20	11.02	1.32
Better	AEX	AEX	AEX	SCX	SCX	—
A056-03f	2.65	15.38	19.63	21.75	8.46	5.01
Better	AEX	AEX	AEX	SCX	SCX	SCX
A065-03f	7.22	21.93	33.00	38.08	8.03	3.26
Better	AEX	AEX	AEX	SCX	SCX	SCX
A071-03f	11.85	25.36	29.12	24.65	10.67	2.68
Better	AEX	AEX	AEX	SCX	SCX	SCX

According to the results shown in Tab. 9, when comparing AEX against other blind crossovers, we observed that AEX is better than CX, OX and PMX on all eight instances. Also, we found that among CX, OX and PMX, the crossover CX is the worst one. So, AEX is the best crossover and CX is the worst one among the blind crossover operators for asymmetric instances. About the distance-based crossovers, on all eight instances SCX is better than GX and HX. For three instances only, SCX and MHX have no difference, and SCX is better than MHX on the remaining five instances. So, SCX is the best one and GX is the worst one among the distance-based crossovers. Further, SCX is compared with all other crossovers and found that SCX is the best one, however it is not reported.

We now extend our study on symmetric instances. Tab. 10 describes results on eight symmetric instances by the eight GAs without mutation. With regard to the average solution, for symmetric instances also the distance-based crossovers are much better than the blind crossovers. Amongst the blind crossovers, CX could not find lowest average solution for a single instance. The crossovers OX, PMX and AEX could find lowest average solution for one, three and four instances respectively. So, among the blind crossovers, AEX is found to be the best one and CX is the worst one. Amongst the distance-based crossovers, GX and HX could not find lowest average solution for a single symmetric instance. However, between GX and HX, HX is better than GX, and GX is better than AEX. The crossovers MHX and SCX could find lowest solution for three and five instances respectively. Among the distance-based crossovers, though MHX and SCX are competing, still SCX is found to be the best one. Among all crossovers, SCX is the best one and CX is the worst one. The results of GX, HX, MHX and SCX are also displayed in Fig. 7 that verifies our conclusion.

Table 10: Results by the GAs without mutation for symmetric instances

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
E-n22-k4	Best Sol	397	387	451	494	385	401	382	380
(375)	Avg. Sol	415.85	414.95	489.05	548.55	439.75	408.23	399.55	**380.60**
	AvgExc(%)	10.89	10.65	30.41	46.28	17.27	8.86	6.55	**1.49**
	S.D.	23.26	20.68	48.19	35.37	24.53	24.08	18.79	19.72
	Avg. Time	0.06	0.08	0.14	0.01	0.07	0.12	0.13	0.11
E-n51-k5	Best Sol	928	920	970	1262	863	564	526	528
(521)	Avg. Sol	963.40	1015.15	998.10	1284.15	912.70	671.50	**568.70**	572.30
	AvgExc(%)	84.91	94.85	91.57	146.48	75.18	28.89	**9.16**	9.85
	S.D.	37.24	27.20	38.24	38.66	45.07	40.21	31.07	26.75
	Avg. Time	0.24	0.29	0.49	0.00	0.24	0.36	0.38	0.69
E-n76-k7	Best Sol	1583	1524	1559	1861	1131	807	701	701
(682)	Avg. Sol	1679.45	1681.05	1699.70	2055.75	1285.95	871.05	755.90	**751.65**
	AvgExc(%)	146.25	146.49	149.22	201.43	88.56	27.72	10.84	**10.21**
	S.D.	50.45	56.13	59.87	63.33	47.29	39.74	25.57	34.59
	Avg. Time	0.32	0.50	1.30	0.01	1.60	1.21	1.44	1.59
E-n76-k8	Best Sol	1595	1676	1665	1905	1205	893	753	744
(735)	Avg. Sol	1754.60	1773.05	1784.15	2111.65	1331.26	902.02	840.35	**828.25**
	AvgExc(%)	138.72	141.23	142.74	187.30	81.12	22.72	14.33	**12.69**
	S.D.	40.53	52.80	64.74	65.11	40.16	45.06	26.63	31.71
	Avg. Time	0.51	0.94	0.98	0.04	1.97	1.75	1.86	2.24
E-n76-k10	Best Sol	1518	1535	1544	1845	1145	1013	855	848
(830)	Avg. Sol	1802.95	1816.05	1750.60	2105.25	1367.75	1094.54	942.05	**923.20**
	AvgExc(%)	117.22	118.80	110.92	153.64	64.79	31.87	13.50	**11.23**
	S.D.	58.16	57.02	134.42	58.09	34.69	35.12	26.87	28.53
	Avg. Time	0.48	0.51	0.84	0.00	1.41	1.29	1.49	1.49
E-n76-k14	Best Sol	1674	1636	1391	1973	1382	1165	1048	1046
(1021)	Avg. Sol	1945.00	1914.90	1619.72	2196.48	1543.72	1306.21	**1159.32**	1161.96
	AvgExc(%)	90.50	87.55	58.64	115.13	51.20	27.93	**13.55**	13.81
	S.D.	64.32	47.98	64.70	56.80	28.16	35.14	23.99	27.54
	Avg. Time	0.68	0.87	1.50	0.01	3.36	2.65	2.72	3.15
E-n101-k8	Best Sol	2242	2268	2144	2702	1490	970	838	844
(815)	Avg. Sol	2428.32	2439.64	2321.42	2897.20	1614.16	1045.52	**925.68**	930.40
	AvgExc(%)	197.95	199.34	184.84	255.48	98.06	28.28	**13.58**	14.16
	S.D.	74.56	61.11	65.58	72.83	42.90	54.21	27.74	30.14
	Avg. Time	0.59	0.95	1.56	0.02	4.57	2.94	3.00	3.34

(Continued)

Table 10: Continued

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
E-n101-k14	Best Sol	2386	2396	2307	2782	1654	1374	1090	1094
(1067)	Avg. Sol	2484.55	2518.75	2403.10	2882.95	1678.45	1429.02	1250.25	**1232.95**
	AvgExc(%)	131.98	135.18	124.38	169.18	56.72	33.43	16.74	**15.12**
	S.D.	61.59	50.41	44.06	52.04	26.99	31.21	18.64	32.20
	Avg. Time	0.66	0.96	1.32	0.01	2.57	1.82	1.88	2.22

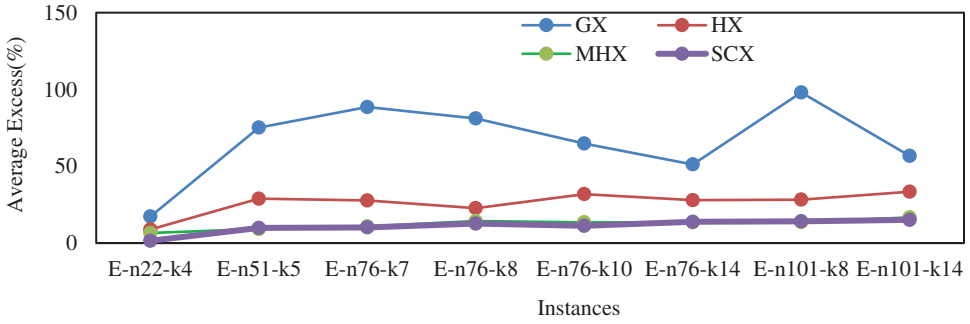

Figure 7: Average Excess (%) by GAs without mutation for symmetric instances

Further, to verify if GA using SCX (excluding mutation) found much different average solution from the average solutions found by other GAs using the distance-based crossovers, we conducted Student's t-test. Further, we verify whether GA using AEX found much different average solution from the average solutions found by other GAs using the blind crossovers. The t-statistic values are reported in the Tab. 11.

Table 11: The t-statistic values (GAs excluding mutation) and the information about the crossovers that found significantly better solutions for symmetric instances

Instance	t-values against AEX			t-values against SCX		
	PMX	OX	CX	GX	HX	MHX
E-n22-k4	−9.58	−9.89	6.97	13.16	6.21	4.87
Better	PMX	OX	AEX	SCX	SCX	SCX
E-n51-k5	−4.55	2.54	36.82	45.46	14.38	−0.61
Better	PMX	AEX	AEX	SCX	SCX	—
E-n76-k7	−1.81	−1.59	28.60	63.83	15.86	0.69
Better	—	—	AEX	SCX	SCX	—
E-n76-k8	−2.71	−0.93	24.97	68.81	9.37	2.05
Better	PMX	—	AEX	SCX	SCX	SCX
E-n76-k10	2.50	3.14	16.95	69.28	26.51	3.37
Better	AEX	AEX	AEX	SCX	SCX	SCX

(Continued)

Table 11: Continued

Instance	t-values against AEX			t-values against SCX		
	PMX	OX	CX	GX	HX	MHX
E-n76-k14	24.96	25.65	46.89	67.85	22.62	−0.51
Better	AEX	AEX	AEX	SCX	SCX	—
E-n101-k8	7.54	9.23	41.13	91.29	12.99	−0.81
Better	AEX	AEX	AEX	SCX	SCX	—
E-n101-k14	7.53	12.09	49.26	74.22	30.61	3.25
Better	AEX	AEX	AEX	SCX	SCX	SCX

According to Tab. 11, for a single instance, AEX and PMX have no difference. For three instances PMX is better than AEX, and for four instances AEX is better than PMX. So, AEX is better than PMX. For two instances, AEX and OX have no difference. For a single instance, OX is better than AEX, and for five instances AEX is better than OX. So, AEX is better than OX. For all instances, AEX is better than CX. So, AEX is the best crossover and CX is the worst one among the blind crossover operators for symmetric instances. About the distance-based crossovers, on all eight symmetric instances SCX is better than GX and HX. However, between GX and HX, HX is better than GX. For four instances, SCX and MHX have no difference, and SCX is better than MHX on the remaining four instances. So, SCX is the best one and GX is the worst one among the distance-based crossovers for symmetric instances.

We now continue our study of the GAs with mutation on symmetric instances, and Tab. 12 describes results on eight symmetric instances. With regard to the average solution, for symmetric instances also the distance-based crossovers with mutation are much better than the blind crossovers with mutation. Among the blind crossovers, OX and CX could not find lowest average solution for any of the eight instances. Between these two crossovers, OX is found to be better than CX. The crossovers PMX and AEX could find lowest average solution for four instances each. So, among the blind crossovers PMX and AEX, each with mutation, are competing. Among the distance-based crossovers, GX and HX, each with mutation, could not find lowest average solution for any of the eight symmetric instances. However, GX is better than PMX and AEX, and HX is found to be better than GX. The crossovers MHX and SCX, each with mutation, could find lowest cost for one and seven instances respectively. So, among the distance-based crossovers, SCX with mutation is the best crossover and GX with mutation is the worst one. So, among all crossovers, SCX with mutation is the best one and CX is the worst one. The results of GX, HX, MHX and SCX are also depicted in Fig. 8 that verifies our conclusion.

Table 12: Results by the GAs with mutation for symmetric instances

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
E-n22-k4	Best Sol	375	377	377	410	384	386	378	375
(375)	Avg. Sol	391.80	398.75	406.20	454.70	391.45	404.31	383.40	**377.05**
	AvgExc(%)	4.48	6.33	8.32	21.25	4.39	7.82	2.24	**0.55**

(Continued)

Table 12: Continued

Instance	Results	PMX	OX	AEX	CX	GX	HX	MHX	SCX
	S.D.	20.38	22.94	45.10	38.63	26.55	28.90	16.39	17.61
	Avg. Time	0.13	0.15	0.13	0.16	0.14	0.22	0.23	0.21
E-n51-k5	Best Sol	818	915	899	1085	830	542	523	526
(521)	Avg. Sol	929.70	1005.80	977.90	1229.60	896.45	665.47	**544.05**	545.85
	AvgExc(%)	78.45	93.05	87.70	136.01	72.06	27.73	**4.42**	4.77
	S.D.	54.06	39.13	33.68	62.84	42.76	33.54	13.92	12.40
	Avg. Time	0.26	0.34	0.50	0.29	0.28	0.58	1.01	0.79
E-n76-k7	Best Sol	1496	1541	1519	1678	1104	705	697	688
(682)	Avg. Sol	1585.15	1665.25	1671.45	1820.55	1255.50	841.21	749.05	**731.60**
	AvgExc(%)	132.43	144.17	145.08	166.94	84.09	23.34	9.83	**7.27**
	S.D.	50.24	48.32	62.16	53.23	21.18	34.85	16.72	21.98
	Avg. Time	0.36	0.53	0.97	1.21	1.12	1.35	1.46	1.41
E-n76-k8	Best Sol	1555	1661	1621	1749	1228	810	745	737
(735)	Avg. Sol	1699.25	1766.50	1760.70	1945.10	1330.15	892.05	817.40	**793.15**
	AvgExc(%)	131.19	140.34	139.55	164.64	80.97	21.37	11.21	**7.91**
	S.D.	49.07	36.58	46.51	60.63	30.56	31.24	25.56	22.60
	Avg. Time	0.38	0.54	0.76	0.82	1.22	1.44	1.52	1.09
E-n76-k10	Best Sol	1511	1527	1273	1839	1144	980	842	840
(830)	Avg. Sol	1734.50	1777.85	1677.95	2076.65	1333.90	1049.75	928.24	**896.85**
	AvgExc(%)	108.98	114.20	102.16	150.20	60.71	26.48	11.84	**8.05**
	S.D.	53.53	50.22	165.95	65.06	34.28	30.52	19.52	27.70
	Avg. Time	0.40	0.46	0.82	0.05	1.38	1.33	1.34	1.12
E-n76-k14	Best Sol	1661	1627	1336	1960	1346	1170	1041	1038
(1021)	Avg. Sol	1770.00	1753.70	1467.10	2034.25	1384.15	1243.05	1130.40	**1105.80**
	AvgExc(%)	73.36	71.76	43.69	99.24	35.57	21.75	10.71	**8.31**
	S.D.	54.13	46.05	73.92	40.79	26.74	18.69	15.68	28.07
	Avg. Time	0.47	0.53	1.17	0.04	1.62	1.18	1.49	1.18
E-n101-k8	Best Sol	2137	2188	2140	2445	1466	909	842	832
(815)	Avg. Sol	2235.50	2300.30	2227.80	2475.75	1470.30	947.90	891.70	**885.05**
	AvgExc(%)	174.29	182.25	173.35	203.77	80.40	16.31	9.41	**8.60**
	S.D.	89.03	76.73	55.42	55.45	39.88	33.20	28.56	33.47
	Avg. Time	0.73	1.12	1.49	2.16	2.13	2.29	2.38	2.14
E-n101-k14	Best Sol	2313	2385	2246	2541	1591	1321	1090	1091
(1067)	Avg. Sol	2429.15	2474.40	2303.60	2667.80	1628.80	1400.60	1196.30	**1180.50**
	AvgExc(%)	126.81	131.04	115.09	149.09	52.08	30.77	11.70	**10.22**
	S.D.	57.96	43.32	43.15	81.75	31.04	33.19	27.60	21.37
	Avg. Time	1.08	1.04	1.23	1.33	2.38	1.95	2.36	2.52

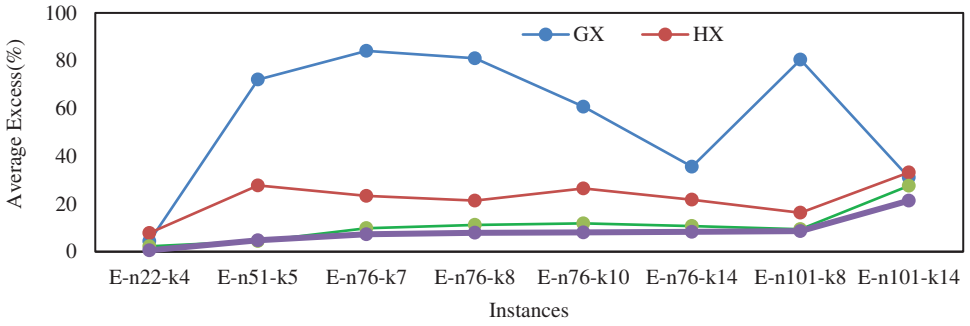

Figure 8: Average Excess (%) by GAs with mutation for asymmetric instances

Further, to verify if GA using SCX (including mutation) found much different average solution from the average solutions found by other GAs using the distance-based crossovers and mutation for symmetric instances, we conducted Student's t-test. Further, we verify whether GA using AEX found much different average solution from the average solutions found by other GAs using the blind crossovers and mutation. The t-statistic values are reported in the Tab. 13.

Table 13: The t-statistic values (GAs with mutation) and the information about the crossovers that found significantly better solutions for symmetric instances

Instance	t-values against AEX			t-values against SCX		
	PMX	OX	CX	GX	HX	MHX
E-n22-k4	−2.04	2.01	5.72	3.16	5.64	1.85
Better	PMX	AEX	AEX	SCX	SCX	—
E-n51-k5	−2.00	7.85	24.71	58.27	23.42	−0.68
Better	PMX	AEX	AEX	SCX	SCX	—
E-n76-k7	−7.56	−0.55	12.75	127.02	18.62	4.42
Better	PMX	—	AEX	SCX	SCX	SCX
E-n76-k8	−6.36	4.24	16.89	98.90	17.95	4.98
Better	PMX	AEX	AEX	SCX	SCX	SCX
E-n76-k10	2.27	4.03	15.66	72.59	29.36	6.48
Better	AEX	AEX	AEX	SCX	SCX	SCX
E-n76-k14	23.14	23.04	47.02	50.26	28.49	5.36
Better	AEX	AEX	AEX	SCX	SCX	SCX
E-n101-k8	0.51	5.36	22.14	80.03	9.33	1.06
Better	—	AEX	AEX	SCX	SCX	—
E-n101-k14	12.16	19.55	27.58	86.99	39.03	3.17
Better	AEX	AEX	AEX	SCX	SCX	SCX

According to Tab. 13, for a single instance, AEX and PMX have no difference. For three instances AEX is better than PMX, and for four instances PMX is better than AEX. So, PMX is better than AEX. For a single instance, AEX and OX have no difference. On the remaining seven instances, AEX

is better than OX. So, AEX is better than OX. On all eight instances, AEX is better than CX. So, PMX with mutation is the best one and CX with mutation is the worst one among the blind crossovers. About distance-based crossovers, on all eight symmetric instances SCX is better than GX and HX. For seven instances, HX is better than GX. For three symmetric instances SCX and MHX have no difference, and for other five instances SCX is better than MHX. For all instances, MHX is better than HX. So, SCX with mutation is the best one and GX with mutation is the worst one among the distance-based crossovers for symmetric instances.

Overall, SCX is the best and MHX is the second best, and CX is the worst for the CVRP. To make a ranking of the crossovers, we conducted a series of Student's t-tests on all kinds of instances together. In fact, for each pair of crossovers, we tested the hypothesis to know which one is better. We summarized the results in Tab. 14, where every row has two columns-1st column is for crossover(s) and 2nd one is for its inferior crossovers. It is observed that there is significant statistical difference between SCX and other crossovers, and so, as estimated, the best crossover is SCX, the 2nd best is MHX, and the worst is CX. The crossover SCX conserves the merits of parents and tries to reduce the local cost between the adjacent customers in the offspring chromosomes by sequentially scan the parent chromosomes. However, other crossovers could not reduce local costs between adjacent customers.

Table 14: The results of statistical hypothesis testing on all kinds of instances together

Crossover	Inferior crossovers
SCX	MHX, HX, GX, AEX, PMX, OX, CX
MHX	HX, GX, AEX, PMX, OX, CX
HX	GX, AEX, PMX, OX, CX
GX	AEX, PMX, OX, CX
AEX	PMX, OX, CX
PMX	OX, CX
OX	CX

6 Conclusion and Future Works

In this research we studied the capacitated vehicle routing problem (CVRP) that is a mixture of the travelling salesman problem (TSP) and the bin packing problem (BPP). A number of crossovers in genetic algorithms (GAs) were suggested for the TSP that can be utilized for the CVRP. Selecting efficient crossover may lead to efficient GA. We developed eight GAs using four blind crossovers-partially mapped crossover (PMX), order crossover (OX), alternating edges crossover (AEX), and cycle crossover (CX), and four distance-based crossovers-heuristic crossover (HX), greedy crossover (GX), modified heuristic crossover (MHX) and sequential constructive crossover (SCX) without mutation, and another eight GAs using above eight crossovers with mutation operator. First, the crossovers were illustrated using same parent chromosomes for building offspring(s), and then all GAs were coded in Visual C++. The usefulness of the crossovers is determined by solving some asymmetric and symmetric instances of numerous sizes. The investigational results reveal the usefulness of AEX among the blind crossovers, and SCX among the distance-based crossovers for the CVRP. So, our study revealed that the distance-based crossovers are much superior to the blind crossovers. Further, we observed that the best crossover is SCX, the 2nd best is MHX, and the worst is CX. This estimation is validated by Student's t-test at 95% confidence level. If only a single crossover operator is applied,

then, whether mutation is applied or not, the best performance is accomplished SCX. Exceptionally bad performance is shown by CX, if mutation is present or not.

There are many crossover operators present in the literature, but all of them may not provide legal solution to the CVRP. Therefore, this paper is confined to study few crossover operators. We aimed to compare amongst eight crossovers without and with a mutation operator. Our aim was not to develop efficient GA. We did not apply either a local search method or merge some heuristics to enhance the solution value to obtain optimal solution. We restricted ourselves to design only simple GAs. Further, the highest crossover probability is set to present exact character of crossovers. Though SCX obtained very good results, still it got stuck in local minima in the initial generations. Hence, successful local search and immigration procedures might be merged to hybridize the GA for obtaining improved results to various instances, which is our next research.

Data Availability: e data set used to support the findings of this study is available from the corresponding author upon request.

Funding Statement: The authors are very much thankful to the anonymous honorable reviewers for their careful reading of the paper and their many insightful comments and constructive suggestions which helped the authors to improve the paper. The authors extend their appreciation to the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University for funding this work through Research Group No. RG-21-09-17.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80–91, 1959.

[2] F. Daneshzand, "The vehicle-routing problem," *Logistics Operations and Management*, vol. 8, pp. 127–153, 2011.

[3] Y. Yao, Y. Zhang, L. Tian, N. Zhou, Z. Li *et al.,* "Analysis of network structure of urban bike-sharing system: A case study based on real-time data of a public bicycle system," *Sustainability*, vol. 11, no. 19, pp. 1–17, 2019.

[4] M. A. Rahman, A. -A. Hossain, B. Debnath, Z. M. Zefat, M. S. Morshed *et al.,* "Intelligent vehicle scheduling and routing for a chain of retail stores: A case study of Dhaka, Bangladesh," *Logistics*, vol. 5, no. 3, pp. 1–20, 2021.

[5] S. I. Khan, Z. Qadir, H. S. Munawar, S. R. Nayak, A. K. Budati *et al.,* "UAVs path planning architecture for effective medical emergency response in future networks," *Physical Communication*, vol. 47, 101337, 2021.

[6] P. Toth and D.Vigo, "Vehicle routing: Problems, methods, and applications," P. Toth and D. Vigo, (Eds.), *Society for Industrial and Applied Mathematics*, Philadelphia, PA, USA, pp. 1–33, 2014.

[7] J. D. E. Little, K. G. Murthy, D. W. Sweeny and C. Karel, "An algorithm for the travelling salesman problem," *Operations Research*, vol. 11, pp. 972–989, 1963.

[8] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. Savelsbergh and P. H. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, no. 3, pp. 316–329, 1998.

[9] R. E. Gomory, "Outline of an algorithm for integer solutions to linear programs and an algorithm for the mixed integer problem," in *50 Years of Integer Programming 1958–2008*, Berlin, Heidelberg: Springer, pp. 77–103, 2010.

[10] Z. H. Ahmed, "A Data-guided lexisearch algorithm for the asymmetric traveling salesman problem," *Mathematical Problems in Engineering*, vol. 2011, Article ID 750968, pp. 1–18, 2011.
[11] R. Tavakkoli-Moghaddam, N. Safaei and Y. Gholipour, "A hybrid simulated annealing for capacitated vehicle routing problems with the independent route length," *Applied Mathematics and Computation*, vol. 176, no. 2, pp. 445–454, 2006.
[12] M. -C. Bolduc, G. Laporte, J. Renaud and F. F. Boctor, "A tabu search heuristic for the split delivery vehicle routing problem with production and demand calendars," *European Journal of Operational Research*, vol. 202, no. 1, pp. 122–130, 2010.
[13] G. Fuellerer, K. F. Doerner, R. F. Hartl and M. Iori, "Ant colony optimization for the two-dimensional loading vehicle routing problem," *Computers and Operations Research*, vol. 36, no. 3, pp. 655–673, 2009.
[14] T. J. Ali and V. Kachitvichyanukul, "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery," *Computers and Operations Research*, vol. 36, no. 5, pp. 1693–1702, 2009.
[15] G. Jeon, H. O. Leep and J. Y. Shim, "A vehicle routing problem solved by using a hybrid genetic algorithm," *Computers and Industrial Engineering*, vol. 53, no. 4, pp. 680–692, 2007.
[16] N. Mladenovic, D. Urosevic, S. Hanafi and A. Ilic, "A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem," *European Journal of Operational Research*, vol. 220, no. 1, pp. 270–285, 2012.
[17] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, 1st ed., New York: Addison-Wesley Longman Publishing Co., 1989.
[18] C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem," *Computers and Operations Research*, vol. 31, no. 12, pp. 1985–2002, 2004.
[19] M. Haj-Rachid, W. Ramdane-Cherif, P. Chatonnay and C. Bloch, "Comparing the performance of genetic operators for the vehicle routing problem," *IFAC Proceedings Volumes*, vol. 43, no. 17, pp. 313–319, 2010.
[20] N. Christophides and S. Eilon, "An algorithm for the vehicle despatching problem," *Operational Research Quaterly*, vol. 20, no. 3, pp. 309–318, 1969.
[21] H. Nazif and L. S. Lee, "Optimised crossover genetic algorithm for capacitated vehicle routing problem," *Applied Mathematical Modelling*, vol. 36, no. 5, pp. 2110–2117, 2012.
[22] K. Puljic and R. Manger, "Comparison of eight evolutionary crossover operators for the vehicle routing problem," *Journal of Mathematical Communications*, vol. 18, no. 2, pp. 359–375, 2013.
[23] S. G. V. Kumar and R. Panneerselvam, "A study of crossover operators for genetic algorithms to solve vrp and its variants and new sinusoidal motion crossover operator," *International Journal of Computational Intelligence Research*, vol. 13, no. 7, pp. 1717–1733, 2017.
[24] M. Sajid, J. Singh, R. A. Haidri, M. Prasad, V. Varadarajan *et al.,* "A novel algorithm for capacitated vehicle routing problem for smart cities," *Symmetry*, vol. 13, no. 10, pp. 1–23, 2021.
[25] Z. H. Ahmed, "Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator," *International Journal of Biometrics & Bioinformatics*, vol. 3, no. 6, pp. 96–105, 2010.
[26] E. Osaba, R. Carballedo, F. Diaz, E. Onieva, A. D. Masegosa *et al.,* "Good practice proposal for the implementation, presentation, and comparison of metaheuristics for solving routing problems," *Neurocomputing*, vol. 271, no. 3, pp. 2–8, 2018.
[27] J. K. Lenstra and A. H. G. Rinnooy Kan, "Some simple applications of the travelling salesman problem," *Operational Research Quarterly*, vol. 26, no. 4, pp. 717–733, 1975.
[28] Z. H. Ahmed, "A lexisearch algorithm for the distance-constrained vehicle routing problem," *International Journal of Mathematical and Computational Methods*, vol. 1, pp. 165–174, 2016.
[29] D. E. Goldberg and R. Lingle, "Alleles, loci and the travelling salesman problem," in *Proc. ICGA*, Pittsburgh, PA, USA, pp. 154–159, 1985.
[30] L. Davis, "Job-shop scheduling with genetic algorithms," in *Proc. ICGA*, Pittsburgh, PA, USA, pp. 136–140, 1985.
[31] J. Grefenstette, R. Gopal, B. Rosmaita and D. Gucht, "Genetic algorithms for the traveling salesman problem," in *Proc. ICGA*, Pittsburgh, PA, USA, pp. 160–168, 1985.

[32] I. M. Oliver, D. J. Smith and J. R. C. Holland, "A study of permutation crossover operators on the travelling salesman problem," in *Proc. ICGA*, MIT, Cambridge, MA, pp. 224–230, 1987.
[33] J. J. Grefenstette, "Incorporating problem specific knowledge into genetic algorithms," in *Genetic Algorithms and Simulated Annealing*, ed., Los Altos, CA: Morgan Kaufmann, pp. 42–57, 1987.
[34] P. Jog, J. Y. Suh and D. Van Gucht, "The effects of population size, heuristic crossover and local improvement on a genetic algorithm for the traveling salesman problem," in *Proc. ICGA*, Morgan Kaufmann Publishers, San Francisco, CA, USA, pp. 110–115, 1989.
[35] M. Fischetti, P. Toth and D. Vigo, "A Branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs," *Operations Research*, vol. 42, no. 5, pp. 846–859, 1994.
[36] N. Christofides and S. Eilon, "An algorithm for the vehicle routing dispatching problem," *Operations Research Quaterly*, vol. 20, pp. 309–318, 1969.
[37] Z. H. Ahmed, "Adaptive sequential constructive crossover operator in a genetic algorithm for solving the traveling salesman problem," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 11, no. 2, pp. 593–605, 2020.