systems

MDPI

Article

A New Lagrangian Problem Crossover—A Systematic Review and Meta-Analysis of Crossover Standards

Aso M. Aladdin [1,2,*] and Tarik A. Rashid [3]

[1] Department of Information Systems Engineering, Erbil Technical Engineering College, Erbil Polytechnic University, Erbil 44001, Iraq
[2] Department of Computer Science, College of Science, Charmo University, Sulaymaniyah, Chamchamal 46023, Iraq
[3] Computer Science and Engineering Department, University of Kurdistan Hewler, Erbil 44001, Iraq
* Correspondence: aso.dei20@epu.edu.iq or aso.aladdin@charmouniversity.org

Abstract: The performance of most evolutionary metaheuristic algorithms relies on various operators. The crossover operator is a standard based on population-based algorithms, which is divided into two types: application-dependent and application-independent crossover operators. In the process of optimization, these standards always help to select the best-fit point. The high efficiency of crossover operators allows engineers to minimize errors in engineering application optimization while saving time and avoiding overpricing. There are two crucial objectives behind this paper; first, we provide an overview of the crossover standards classification that has been used by researchers for solving engineering operations and problem representation. This paper proposes a novel standard crossover based on the Lagrangian Dual Function (LDF) to enhance the formulation of the Lagrangian Problem Crossover (LPX). The LPX for 100 generations of different pairs parent chromosomes is compared to Simulated Binary Crossover (SBX) standards and Blended Crossover (BX) for real-coded crossovers. Three unimodal test functions with various random values show that LPX has better performance in most cases and comparative results in other cases. Moreover, the LPB algorithm is used to compare LPX with SBX, BX, and Qubit Crossover (Qubit-X) operators to demonstrate accuracy and performance during exploitation evaluations. Finally, the proposed crossover stand operator results are demonstrated, proved, and analyzed statistically by the Wilcoxon signed-rank sum test.

Keywords: evolutionary metaheuristic algorithm; crossover standards; crossover operators; Lagrangian Dual Function; Lagrangian Problem Crossover

Citation: Aladdin, A.M.; Rashid, T.A. A New Lagrangian Problem Crossover—A Systematic Review and Meta-Analysis of Crossover Standards. *Systems* **2023**, *11*, 144. https://doi.org/10.3390/systems11030144

Academic Editors: Mahmoud Efatmaneshnik, Shraga Shoval and Larissa Statsenko

Received: 30 January 2023
Revised: 1 March 2023
Accepted: 6 March 2023
Published: 9 March 2023

1. Introduction

Combinatorial optimization is one of the most widely investigated areas of artificial intelligence. Every year, several research projects focus on issues that arise in this domain. The solution structure, rather than its coding, is utilized by a knowledge-based crossover mechanism for strategic metaheuristic algorithms [1]. Thus, there are several different approaches that researchers have used in the past to solve single or dual problems. However, in the majority of these kinds of applications, research is limited to solve linear programs, quadratic programs, or, more broadly, convex programming issues. Because the primal optimal solution is closely related to the optimal single or dual solution for convex problems, such a study has assisted investigators in better understanding the relationship [2]. Therewith, a set of optimization algorithms influenced by natural events and animal intelligence is characterized as evolutionary nature-inspired metaheuristic algorithms. They are, thus, frequently nature-inspired algorithms, and samples of these evolutionary metaheuristic algorithms are Genetic Algorithm (GA) [3], Artificial Bee Colony [4], Differential Evolution [5], and Learner Performance based Behavior algorithm (LPB) [6]. Therefore, nature-inspired

computing is a field of computer science that could be used and shared with an optimization algorithm, computational intelligence, data mining, and machine learning [7].

A problem with a fluctuating objective function is one of the more difficult metaheuristic optimization methods, but it is much more common in real-world search or self-adaptive optimization. The search technique used to solve such problems for optimality must be flexible enough to adjust to the present function [8]. A problem with population-based optimizers is that once the investigation has found a locally optimal solution, there may not be enough diversity to move the search forward to a new, better solution. In many of these circumstances, diversity-preserving strategies such as a high amount of crossover or the use of a clustering operator are required. Therefore, most of the algorithms have been improved and enhanced in the field of metaheuristic optimization by these standard operators [9]. In addition, several effective methods have been developed in this research to improve existing optimizers based on crossing genes between parents. Alternatively, updated standards are always accepted as long as the suggested methods offer novel improvements or comparable results.

When the genetic recombination operator is chosen correctly, crossing genes can match the best-known techniques for a wide range of problems involving restrictions. It would be better to suggest or use the superlative crossover standard to accomplish the goal. The efficiency of this new method could be shown to work for selected features [10]. The motivation behind this paper is the use of conventional techniques and direct search procedures due to the complexity of variables in the problematic domain. Depending on the situation, fixing the problem might involve modifying the basic algorithm, presenting a systematic and comprehensive overview of the meta-analysis, or utilizing innovative metaheuristics. In addition, this paper proposes a novel standard from several strategies linked to mathematical assessments with originally generated methods.

The fundamental goal of this research is to investigate the effectiveness of experience and understanding methods in GAs. This paper is focused on the generation and standards of crossover and how it affects metaheuristic algorithms. Crossover, also known as recombination, is a genetic operator in the special process which is exploited to connect the genetic codes of two parents to make new offspring (children). Further, crossover techniques can be considered to be extremely useful for generating new solutions from a current population stochastically [11]. In many studies, crossover and mutation operators have been associated with GA's success. Some of them conclude that success rests in both, whether the crossover is performed alone or through mutation or both. Crossover operators play a substantial role in balancing exploitation and exploration, allowing for feature extraction from both chromosomes (parents), with the intention that the developing offspring have beneficial qualities from both chromosomes [12].

Lagrangian Dual Function (LDF) [13] technique can support to exchange genes between chromosomes by locating the replacement chromosome at the highest or lowest point to produce offspring with improved traits. A proposed standard known as Lagrangian Problem Crossover (LPX) is imperative for generating new operators; the crucial characteristic of LPX is that it allows offspring to inherit some characteristics from their parents for finding a new optimal solution in population solution-based metaheuristics. Likewise, depending on the meaning of the worst chromosome for each problem, this point (its worst chromosome) can be chosen on both chromosomes (parents), or by applying crossover between chromosomes to produce the best novel genes. Furthermore, the new operator is identified automatically based on performance ratings and statistical evidence, so the time spent selecting the best operator is taken into account.

The crossover strategy starts with a low value and adjusts it every generation to avoid premature convergence. Moreover, several crossover functions are employed as a strategy to avoid convergence rates. Population (swarm) algorithms are one of the most successful metaheuristics for managing these kinds of numerous case problems. As a result, population-based methods have become one of the most efficient methods for combinatorial function optimization [14]. These techniques operate with several populations of solutions

that evolve in tandem with algorithm operation. Accordingly, the following summarizes this paper's main contributions:

- There are several standard operators used to illustrate how the implementation was conducted and illustrate the mathematical crossover form using small examples and technique operations;
- As a systematic development of previous standards, it has enabled the use of binary form, real-coded form, and ordered-coded form methods;
- Based on LDF, a crossover operator has been proposed that can provide a novel optimum solution for population metaheuristic algorithms that use original metaheuristic optimization;
- The new anticipated LPX is evaluated by comparing it with selected previous tuning methods, a variation on the traditional GA, as discussed in the next sections;
- LPX is compared with other well performing crossover operators using the LPB algorithm as a single objective population-based algorithm and the affected random values and elapsed time are measured;
- The proposed standard operator is statistically analyzed and compared, using non-parametric statistical tests.

The next section describes how this paper is organized; this section is dedicated to discussing related work on crossover standards. Several mathematical crossover standards are presented by thinkable pseudocode in the third section of the paper. The fourth segment examines the novel formula which can be developed as a standard crossover in the future metaheuristic algorithm known as LPX. The fifth section is devoted to proving the heuristic and exploitation crossover results by comparing LPX results with specific real-coded crossover standards. At the end of the paper, the conclusion and novel features are discussed.

2. Crossover Standards Overview

As stated, the paper includes two major subjects. The first part includes a systematic review of crossover standards; the second, a proposed a method to generate novel offspring by generating a newly created evolutionary algorithm.

A systematic review has been performed to assess several efficient methods presented in the research. The reviewed articles, as well as standard sources, were examined using search terms including "crossover standard generation", "generations of crossover standards of parent chromosomes in evolutionary metaheuristic algorithms", or "Genetic Algorithm Based on crossover standard generation". These queries are searched in ScienceDirect, PubMed, and Google Search Engine. These articles and sources were culled to address only crossover standards, evolutionary algorithms, Gas, and crossover operators in GAs. Search dates include up to March 2022. A huge number of articles was retrieved, but in the end, the standard crossover operators were approved by 41 eligible papers.

The second part is the proposed LPX standard which is used as the superlative crossover standard to accomplish the target; the efficiency of this new method might be proved for the feature selection optimization problems. The standard is generated from the mathematical model of the LDF theorem.

Many forms of the crossover have been produced over the years, and comparisons between various types have been proposed. These started with one-point crossover and evolved into a range of ways to cover a variety of conditions, including uniform crossover [15]. To generate improved self-adaptive combinational optimization, a set of assumptions (rules) have been developed to simplify natural/biological events, and these include a list of control parameters to define intensification and diversification rules [16]. The best solution is identified, and other solutions advance toward the most optimal solution according to the given rules. In the stochastic mode, the location of a few solutions can be altered and controlled, such as crossover or mutation operations in several metaheuristic algorithms, which is illustrated in a simple flowchart in Figure 1.

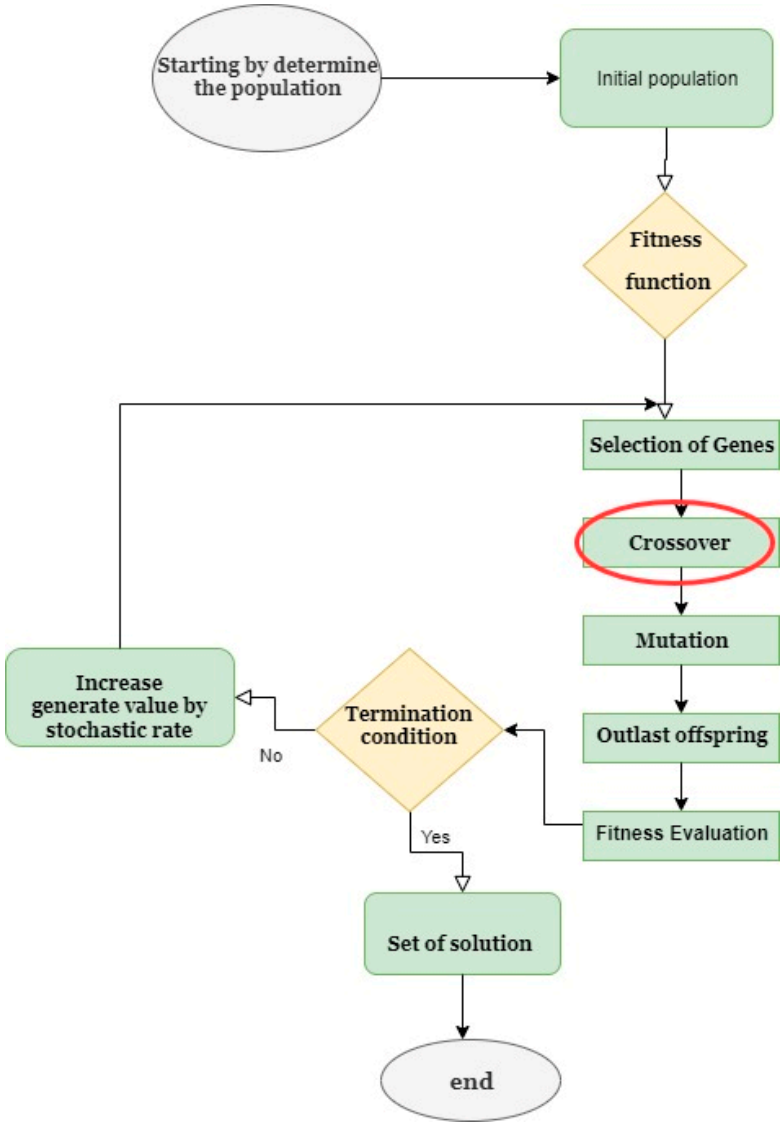

Figure 1. Simple deterministic processes in metaheuristic algorithms.

Likewise, several standards for permutation applications, such as the Traveling Salesman Problem (TSP), were defined. There are several ways to approach the TSP using evolutionary algorithms, including binary, route, closeness, ordinal, and vector representations. To reduce the overall distance, the researchers presented a novel crossover operator for the TSP [17]. Another research study confirmed that sequential constructive crossover (SCX) fixed the TSP in 2010. The primary idea behind this strategy is to choose a random point, termed the crossover point; then, before the crossing point, use an SCX technique with improved edges. After the crossover site, the remaining chromosomes are swapped between parents to generate two children; if a chromosome currently exists, it is replaced with an unoccupied chromosome [18].

Ring Crossover was offered as a solution to the recombination problem. Parents were grouped in the design of a ring of this type, and then a cut point was chosen. Parents were grouped in the design of this circle procedure, and then a slice point was selected randomly. The other location was the length of the chromosome; the first offspring develops clockwise from the line (the original cut), and the second offspring evolves counter clockwise. They employed this type of crossover for the aspects mentioned and it outperformed the other types of assessed crossover [19]. Despite that, to prevent creating erroneous solutions, evolutionary algorithms that optimize the ordering of a very large series require specific

crossover operators. It is difficult to list all of them. Thus, several standard crossovers have already been documented in Table 1 and each of them has been generated for a specific global solution. However, some of them produced offspring from parents based on real code, whereas others relied on the binary-coded crossover started in the next section.

Table 1. Standard crossovers generation.

No.	Standard Crossover Operator Name	Initial Abbreviation	Related Work
1	Order Crossover Operator	OX1	[17,20,21]
2	Sequential Constructive crossover	SCX	[18]
3	Order-Based Crossover Operator	OX2-OBX	[20,22]
4	Maximal Preservation Crossover	MPX	[22,23]
5	Alternating Edges Crossover	AEX	[21,23,24]
6	Edge Recombination Crossover	ERX	[20,21]
7	Position-Based Crossover Operator	POS	[20,22,25]
8	Voting Recombination Crossover Operator	VR	[20,22]
9	Alternating Position Crossover Operator	AP	[20,26]
10	Automated Operator Selection	AOS	[27]
11	Complete Sub-tour Exchange Crossover	CSEX	[22,28]
12	Double Masked Crossover	BMX	[22,29]
13	Fuzzy Connectives Based Crossover	FCB	[30,31]
14	Unimodal Normal Distribution Crossover	UNDX	[32,33]
15	Discrete Crossover	DC	[34]
16	Arithmetical Crossover	AC	[19,31,35]
17	Average Bound Crossover	ABX	[36]
19	Heuristic Crossover	HC	[17,37]
20	Parent Centric Crossover	PCX	[22,38]
21	Spin Crossover	SCO	[39]

In application, crossover standards have typically been classified based on the representation of the gene; genetic sequence has been stored in a chromosome represented by a bit matrix or real code in the different algorithms. Crossover strategies for both techniques are popular, and illustrative instances or classes are genetic recombinations, which are thoroughly explained in the following sections. Several current methods ensure that these techniques can be applied to global numerical optimization and current practical problems as a recently proposed meta-heuristic; for example: Slime Mold Algorithm [40], Moth Search Algorithm [41], Hunger Games Search [42], Harris Hawks Optimization [43], and Colony Predation Algorithm [44]. As a result, novel standards should be proposed for evolving evolutionary algorithms.

The capacity of solutions to learn from one another, however, is what gives rise to the intriguing behavior of GAs. Solutions can combine to form offspring for the next generation. Occasionally they will share their poor information, but if we use crossover in conjunction with a harsh selection method, better solutions will emerge; there are many details to crossover with permutations, as in previous population-based algorithms, especially these algorithms based on GA, such as the Quantum-based Avian Navigation Optimizer Algorithm [45]. Thus, we will cover the basic crossover techniques, known as "modification genes", by Lagrange method techniques. Accordingly, Table 2 highlights the weak and strong points of crossover operators in metaheuristic population-based algorithms.

Table 2. Perfection and shortcomings of crossovers generation.

Perfection of Crossovers Generation	Shortcomings of Crossovers Generation
▪ Through crossover, it is possible to mix incomplete solutions from several competitors. This frequently entails taking a pretty significant step away from either of the parents and can swiftly push one outside of a local optimum; ▪ Crossover is crucial since the tested parents share genetic material, which can result in a more effective solution; ▪ It signifies that even without crossover, the change would be slow, and it would be impossible to push your population over a local optimum; ▪ Without crossover, a population-based algorithm would resemble a search algorithm that uses random numbers, and finding a workable solution would require some luck. However, with crossover, it improves with each iteration while keeping the positive aspects of the previous one [46]; ▪ A high crossover rate causes the genomes in the next generation to be more random. This is because there will be more genomes that are a mix of previous-generation genomes rather than a low crossover rate. The second approach decreases the chance that a very accurate genome will be produced by the crossover operation [47].	▪ To ensure that their representations enable crossover to be reasonable, limited generations should be required for crossover to occur; ▪ An ideal generation is also determined by standard crossover operators. Look at the type of two-point crossovers compared to one-point crossovers. There might be a loss of genetic material from the parent solution in a point crossover or by another standard. As a result, most evaluations depend on complex mathematical models of the new generation [48]; ▪ The actual permutations would be the only additional representations in the second generation. In that situation, the crossover will presumably result in a completely distinct local optimum that is both meaningless and unachievable; ▪ According to several references, single-point crossover results in offspring genomes being less diversified since they are more likely to resemble their parents rather than multi-point which is a hybrid of single-point and uniform [49].

3. Mathematical Crossover Standards

In metaheuristic algorithms, the exploration of the optimal solution is based on the generation of new members from existing members. The crossover process facilitates the interchange of genetic code between parents, which results in a higher probability of genes being exceptional to the parents. However, there are numerous crossover techniques recorded in the cited study. Researchers should focus on finding and tackling the question of whether the most effective standard strategy has been improved and adopted. As mentioned, the crossover operator is comparable to multiplication and biological recombination. The data suggest that more than one genome must be selected and that children are produced using the genetic codes (genomes as blue balls on the parents' chromosomes [50]) and then two more children using two new offspring genes (pink balls). This probabilistic scale is illustrated in Figure 2. Then, the graphic depicts the range of potential offspring in two-dimensional constrained real space between x and y dimensions by generating a box crossover between genes and new offspring.

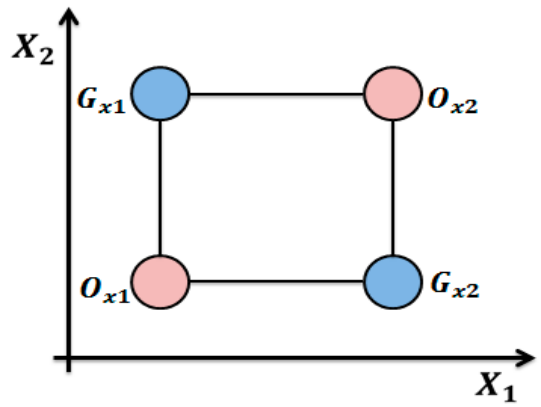

Figure 2. Significant probability in the real-coded crossover.

Typically, the crossover is used in metaheuristic algorithms with a significant probability, particularly in a GA, as challenging in real-coded crossovers [51]. Consequently, the goal of establishing crossover likelihood is to prevent gene loss from the parents, even

if the offspring are not better than the parents. According to the distribution of crossover standards, Section 3.1 has determined the forms of binary crossover. The Section 3.2 has classified the categories of real-coded or floating-point crossover. In Section 3.3, the form of order-coded crossover is distributed.

3.1. Binary Form Crossover

This section provides a broader collection of crossover operators used in binary representation for metaheuristic algorithms. Improvements to previous results demonstrate that, according to the current challenges, most of these results are effective crossovers. How to implement some types of crossover standards and point out some interesting comparisons between others could also be shown [52]. Traditionally, genetic material has been stored in a gene, which is represented as a bit collection in various techniques. Crossover procedures for bit-order are prominent, and specific examples or categories include genetic manipulation, as explained in the points below.

Binary Single-point crossover [53]: A crossing point in the parent entity string is picked. Apart from that point in the biological sequence, all data transfer between two units, including biological parents and situational bias, is conducted through strings. As indicated in Figure 3, Genes with sub-blocks that include three bits to the right of that point are transferred correspondingly between the two parents. As illustrated by pink and blue color bites, it has generated two new offspring.

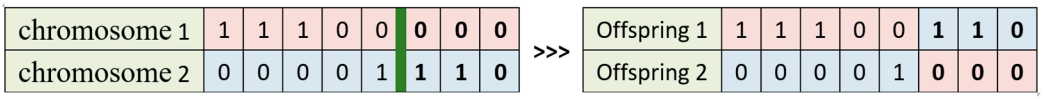

Figure 3. Generate new offspring with single-point crossover.

Double-point and n-point crossover [53]: two randomly generated locations (strings) or n-point locations on the individual chromosomes are chosen, and the gene code is switched at these locations. As seen in Figure 4, two equally spaced points on the right and left sides are selected on parent chromosomes, then pink color and blue color bits are swapped to perform two single-point crossovers.

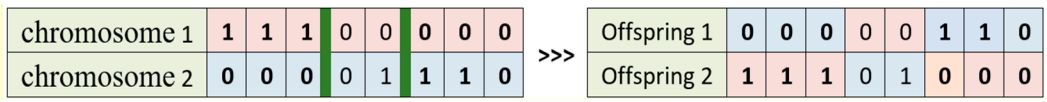

Figure 4. Double-points crossover for generating two new children.

Uniform crossover [49] **and half-Uniform crossover** [54]: such as in the coin-throwing approach, each gene (bit) is drawn randomly from one of the comparable genes in the selected parents. Each genome is addressed separately rather than being separated into segments. In this situation, we just flip a coin to see if each genome is present in the child. The idea may be tossed about to support one parent having more genetic information in their newborn. Figure 5 shows two chromosomes arrayed as a two-dimensional array with bits exchanged in a light blue color and a pink color.

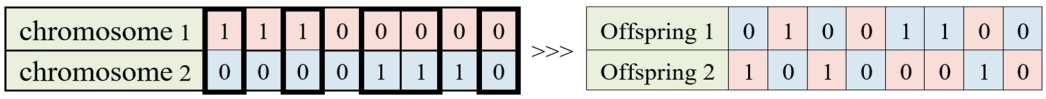

Figure 5. Producing two fresh offspring by uniform crossover.

Uniform Crossover with Crossover Mask (UCM) [27]: The matrices are separated into several non-overlapping zones, and the matrix created by the logical operator is known as the crossover mask (CM) generated by pseudocode control. In Figure 6, we present an example and pseudocode to display how the new offspring has spawned between chromosomes and CMs based on these conditions.

```
1   //To generate CM:
2   if G1= 0 and G2=0
3      then CM =0
4   if G1!=0 or G2=0
5      then CM = 1
6   //To generate the gene of the first offspring:
7   if CM=0
8      then select G1
9   if CM=1
10     then select G2
11  //To generate the gene of the second offspring:
12  if CM=0
13     then select G2
14  if CM=1
15     then select G1
```

chromosomes 1	**1**	1	**1**	0	**0**	**0**	0	**0**
chromosomes 2	**0**	0	**0**	0	**1**	**1**	1	**0**

CM	1	1	1	0	1	1	1	0

Offspring 1	**0**	0	**0**	0	**1**	**1**	1	**0**
Offspring 2	**1**	1	**1**	0	**0**	**0**	0	**0**

Figure 6. Pseudocode and example of uniform crossover deliberation from crossover mask.

Shuffle Crossover (SHX) [55,56]: Initially, we choose a crossover point at random, such as the highlighted line in Figure 7, then mix the gene code of both parents. It should be emphasized that Shuffle chromosomes for the right and left sites are handled independently. A single point of crossing is chosen, which splits the chromosome into two sections, known as schema. Chromosomes are scrambled in each schema by both parents. To produce offspring, schemas are transferred (as in a single crossover).

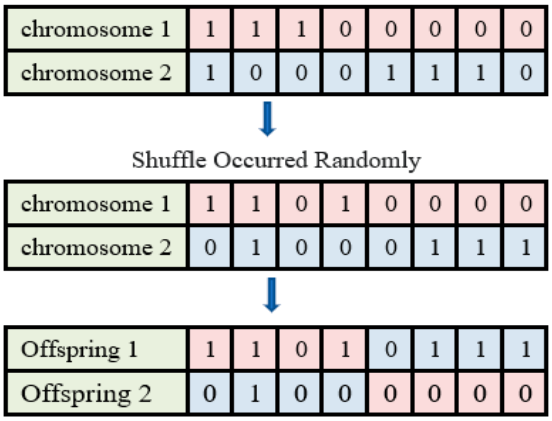

Figure 7. SHX random occurrence.

Three-Parent Crossover (TPX) [57]: According to the prior solution approach, in this kind of operator, there are numerous probability rate algorithms with which to create innovative offspring from three parent genes. In the elucidation example, Figure 8 highlights the problems involved in calculating future generations based on deliberate offspring generated by swapping genes according to the general pseudocode.

```
1   //To generate Offspring1 from (C1, C2, C3) combination
2   if C1bit is equal to C2 bit
3         then select C1 bit
4   if C1 bit is not equal to C2 bit
5         then select C2 bit
6   // To generate Offspring2 from (C1, C2, C3) combination
7   if C1 bit is equal to C3 bit
8         then select C1 bit
9   if C1 bit is not equal to C3 bit
10        then select C2 bit
11  //To generate Offspring3 from (C1, C2, C3) combination
12  if C2 bit is equal to C3 bit
13        then select C2 bit
14  if C2 bit is not equal to C3 bit
15        then select c3 bit
16  //The generatation will be continue by the conditions...
```

chromosome 1	1	1	1	0	0	0	0	0
chromosome 2	0	0	0	0	1	1	1	0
chromosome 3	0	1	0	0	1	0	1	0

⩔

Offspring 1	0	0	0	0	1	1	1	0
Offspring 2	0	1	0	0	1	0	1	0
Offspring 3	0	1	0	0	1	0	1	0

Figure 8. Pseudocode and example to explain TPX deliberation.

3.2. Real-Coded (Floating Point) Form Crossover

The genes are real-valued without encoding or decoding into binary, and the digits are left alone to speed up the process. Nevertheless, it is less logical than binary representation since crossover has demonstrated that floating-point format can function as well as, if not better than, ordinary binary strings. Therefore, there is no reason to be concerned about algorithm efficiency if the floating-point encoding is utilized [51]. Several crossover techniques for real-coded crossovers were developed. The method is based on effectively adjusted real-coded crossover operations that use the likelihood function to create very distinct sequences that may be candidates for alternative solutions [58]. The crossover techniques are described mathematically in the next points.

Real Single-point Crossover (RSPX) [59]: It is marginally comparable with a binary single-point crossover; it could be combined with two chromosomes and use a real number for each gene at the crossover point. It can also be generated for two-point, three-point, and n-point crossovers. As shown in Figure 9, two genes were crossed and real numbers were swapped between them, resulting in two new offspring.

chromosome 1	0.88	0.13	0.25	0.08	0.34	>>>	Offspring 1	0.88	0.13	0.35	0.63	0.34
chromosome 2	0.64	0.94	0.35	0.63	0.48		Offspring 2	0.64	0.94	0.25	0.08	0.48

Figure 9. Explain swapping RSPX.

Single arithmetic crossover (SAX) [57]: This standard should be derived from a single genome (k), randomly chosen from both chromosomes (n). For instance, in Figure 10, $k = 2$ and then we define a random parameter ($\alpha = 0.5$). We modify the kth gene of chromosome 1 and chromosome 2 to generate offspring by the selected asthmatic formula that is calculated as Equation (1) [60,61].

$$\text{Gene}_{\text{offspring(n,k)}} = (1 - \alpha) * G_{nk} + \alpha * G_{nk} \quad (1)$$

$$\text{Gene}_{\text{offspring(n,2)}} = (1 - 0.5) * 0.13 + 0.5 * 0.94 = 0.535$$

chromosome 1	0.88	0.13	0.25	0.08	0.34	>>>	Offspring 1	0.88	0.535	0.25	0.08	0.34
chromosome 2	0.64	0.94	0.35	0.63	0.48		Offspring 2	0.64	0.535	0.35	0.63	0.48

Figure 10. Target example for SAX.

Whole Arithmetic Crossover (WAX) and Linear Crossover (LX) [62]: The calculations for the whole arithmetic (linear) crossover have been handled for all genes on the chromosome (n) with a single similar arithmetic crossover. The calculation is shown in Equation (2) [58], and an example of how to create new offspring can be seen in Figure 11. Despite this, the probability rates are higher since the number of offspring produced will equal the number of chromosomes (genes = k). It should be defined as a random parameter (α and β) and as ($\alpha_1 = 0.5$, $\alpha_2 = 1.5$, $\alpha_3 = -0.5 \ldots\ldots \alpha_m$) and ($\beta_1 = -0.5$, $\beta_2 = 0.5$, $\beta_3 = 1.5 \ldots\ldots \beta_m$) and, for this example, when the numbers of genes (k) equals three. This example calculated and produced three offspring when $k = 1$; in consequence, it can be calculated for k numbers. However, Figure 11 just generates three offspring, the generation could be developed by other children according to the k numbers.

$$\text{Gene}_{\text{offspring(m,n,K)}} = \alpha_m * G_{nk} + \beta_m * G_{nk} \quad (2)$$

$$\text{Gene}_{1,n,1} = \alpha_1 * G_{11} + \beta_1 * G_{21} = \alpha_1 * 0.88 + \beta_1 * 0.64 = 0.12$$
$$\text{Gene}_{2,n,1} = \alpha_2 * G_{11} + \beta_2 * G_{21} = \alpha_1 * 0.88 + \beta_1 * 0.64 = 1.64$$
$$\text{Gene}_{3,n,1} = \alpha_3 * G_{11} + \beta_3 * G_{21} = \alpha_1 * 0.88 + \beta_1 * 0.64 = 0.52$$

chromosome 1	0.88	0.13	0.25	>>>	Offspring 1	0.12	0.13	0.25
chromosome 2	0.64	0.94	0.35		Offspring 2	1.64	0.13	0.25
					Offspring 3	0.52	0.13	0.25

Figure 11. Generation of offspring by LX when k = 1.

Blended Crossover (BX) [50,63]: One of the effective crossovers that improved several algorithms. If the two-parameter values in a pair of chromosomes are $G1$ and $G2$, $G1$ is thus smaller than $G2$, and the blend crossover method produces an offspring option in the range $[G1 - \alpha\,(G2 - G1),\ G2 - \alpha\,(G2 - G1)]$. Where α is a constant to be established, the solutions of the offspring do not exceed the scope of the single variable [50,64].

The example is stated in Figure 12 which indicated k number is equal to 2, $G1 = 0.13 < G2 = 0.94$, so we calculate the range by $[G1 - \alpha\,(G2 - G1),\ G2 - \alpha\,(G2 - G1)]$; when $\alpha = 0.5$ the range is [1.345, 0.535] so, we randomly select $G1$ and $G2$ between the range.

chromosome 1	0.88	0.13	0.25	>>>	Offspring 1	0.88	1. 2	0.25
chromosome 2	0.64	0.94	0.35		Offspring 2	0.64	0.7	0.35

Figure 12. BX for second genes by the range calculation.

If the process was evaluated in the previous range, it could not find a global solution, as documented in several improvement algorithms. According to earlier researchers [50], the condition might be calculated by the updated blend formula. Thus, a novel technique has been developed for the BX standard. The parameter γ must be determined using α and a random integer r in the range limitation between (0.0, 1.0), both of which are excluded from the Equation (3) [65].

$$\gamma = (1 + 2\alpha) \quad (3)$$

$Genome_1$ and $Genome_2$ are the offspring solutions which are regulated by the parents as Equations (4) and (5) [65], consecutively:

$$\text{Gene}_1 = (1-\gamma)* G_1 + \gamma * G_2\ ; \tag{4}$$

$$\text{Gene}_2 = (1-\gamma)* G_2 + \gamma * G_1. \tag{5}$$

Figure 13 pointed out the example when k (gene) = 2, then identified randomly $\alpha = 0.5$ and $r = 0.5$, so parameter γ is calculated by Equation (3).

$$\gamma = (1+2\alpha)*\text{r} - \alpha = (1+2*0.5)*0.5-0.5 = \mathbf{0.5}$$
$$\text{Gene}_1 = (1-\gamma)* G_1 + \gamma * G_2 = (1-0.5)*0.13+0.5*0.94 = \mathbf{0.535}$$
$$\text{Gene}_2 = (1-\gamma)* G_2 + \gamma * G_1 = (1-0.5)*0.94+0.5*0.13 = \mathbf{0.535}$$

chromosome 1	0.88	0.13	0.25	>>>	Offspring 1	0.88	0.535	0.25
chromosome 2	0.64	0.94	0.35		Offspring 2	0.64	0.535	0.35

Figure 13. BX for second genes depending on γ parameter.

Simulated Binary Crossover (SBX) [66,67]: It is preferable and common to implement a standard crossover operation across all standards. Due to these reasons, SBX was applied to the real-coded parameter without any mutation operator, and SBX was also developed based on the single-point crossover [66]. This approach focuses on the probability distribution of obtainable offspring (Gene) by the specified parents (Genes), as shown in Equation (11) [68]. SBX initially computes the number of children using Formulas (6) and (7) [68], or by using Formulas (8) and (9), that enhance the last two formulas by Azevedo [66], which are the most commonly used. The steps to calculate the float number resulting from the crossover are started by fixing a random number $\mu \sim (0, 1)$ at first; then, calculate the α and generate offspring by using α.

$$\text{Gene}_1 = 0.5[(1+\alpha_i)G_1 + (1-\alpha_i)G_2] \tag{6}$$

$$\text{Gene}_2 = 0.5[(1-\alpha_i)G_1 + (1+\alpha_i)G_2] \tag{7}$$

$$\text{Gene}_1 = 0.5[(G_1+G_2) - \alpha_i|G_2-G_1|] \tag{8}$$

$$\text{Gene}_2 = 0.5[(G_1+G_2) + \alpha_i|G_2-G_1|] \tag{9}$$

As a function of (9), Alpha (α_i) [63] must be calculated as a function of the two previous offspring. H is the index of a user-defined distribution (not negative), which means η is the number of parameters chosen by the user.

$$\alpha = \begin{cases} (2\mu)^{\frac{1}{\eta+1}},\ if\ \mu < 0.5 \\ \left(\frac{1}{2(1-\mu)}\right)^{\frac{1}{\eta+1}},\ otherwise \end{cases} \tag{10}$$

Utilize the probability distributions to compute the function of Alpha (α_i).

$$\alpha = \begin{cases} 0.5(\eta+1)\alpha^{\eta},\ if\ \alpha \le 1\ \text{(Contracting Crossover)} \\ 0.5(\eta+1)\frac{1}{\alpha^{\eta+2}},\ otherwise\ \text{(Expanding Crossover)} \end{cases} \tag{11}$$

When selecting the second gene as a parent 1 and 2 from Figure 14 will produce two new offspring genes, we need to find (α_i) if $\mu = 0.4$, and the user chooses two parameters. The calculation is executed as follows:

$\alpha = (2 * 0.4)^{\frac{1}{2+1}} = 0.928$;
$\text{Gene}_1 = 0.5[(0.13 + 0.94) - 0.928|0.94 - 0.13|] = 0.1592$;
$\text{Gene}_2 = 0.5[(0.13 + 0.94) + 0.928 * |0.94 - 0.13|] = 0.9108$ (out-of-range probability distribution).

Chromosome 1	0.88	0.13	0.25	>>>	Offspring 1	0.88	0.1592	0.25
Chromosome 2	0.64	0.94	0.35		Offspring 2	0.64	0.9108	0.35

Figure 14. SBX for the second Genes.

However, this technique has many advantages, such as a wider range of offspring explored, and results are reliable and often reach global optima. In Figure 15, it is illustrated that sometimes the results of the offspring gene are out of range based on the probability distributions. This is because the new gene's impact should be bigger than the old gene, but in this case, the updated gene is smaller than the old gene.

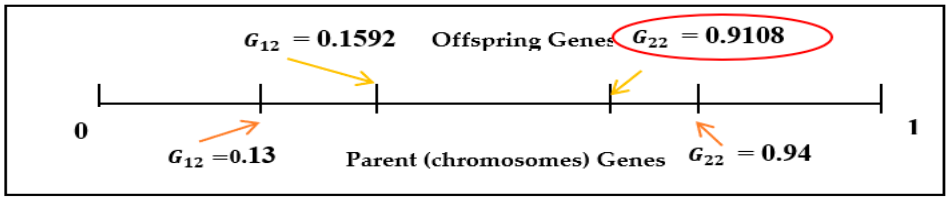

Figure 15. Probability distributions of genes out of range at SBX.

3.3. Order-Coded Problem Methods Crossover

Several different crossover operators have been proposed to reproduce either the relative order or the precise organization of chromosomes from the homologous chromosome. The majority of standard-matched crossovers are operators that maintain exact locations [69]. As a result, the following sections focus on the most basic forms of order-coded crossovers.

Partially Mapped Crossover (PMX) [70]: The procedure produces offspring solutions by transferring sub-orders from one genome to the other while maintaining the original sequence (order) with possible several points, as shown in Figure 16. To initiate, we choose a random range of crossovers and produce children by swapping genes. Unselected substrings are then used to identify the mapping relationship at the time of legalization of the resulting offspring [71].

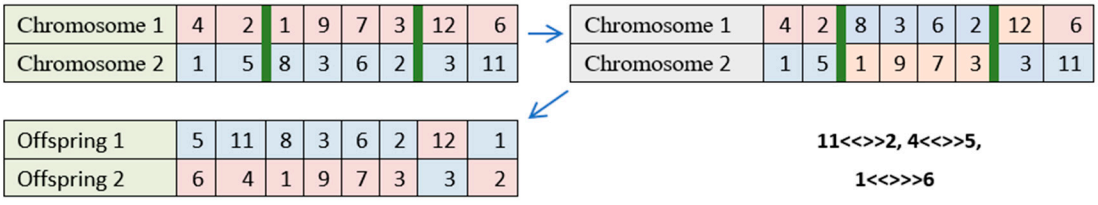

Figure 16. PMX development steps.

The Cycle Crossover Operator (CX) [17,72]: Determines the number of cycles that exist between two-parent chromosomes. It may be used with numerical strings in which each component appears only once. This assures that each index point in the resultant offspring is filled with a value from one of his parents [73]. Figure 17 shows that the first offspring world is generated by relying on the pseudocode when the random cycle includes {2, 5, 7, 6, 11}.

```
1    //To generate offspring 1
2    if Gene at C1 is in the cycle
3        then select Gene at C1
4    if Gene at C1 is not in the cycle
5        then select Gene at C2
6    //To generate offspring 2
7    if Gene at C2 is in the cycle
8        then select Gene at C2
9    if Gene at C2 is not in the cycle
10       then select Gene at C1
```

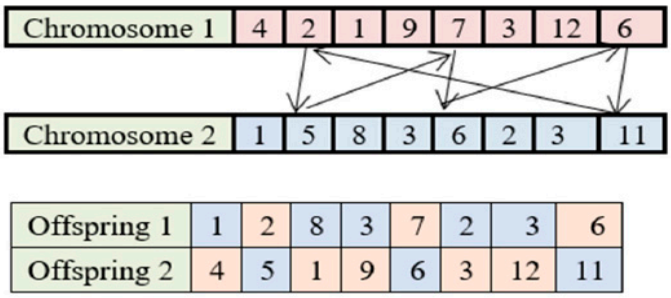

Figure 17. CX operator progressive.

4. Lagrangian Problem Crossover

Two types of crossover operators have been used in population-based algorithms. The first type is operated with binary numbers, while the other is operated with real numbers. By using real code, the crossover operator is vindicated by most of the algorithms rather faster than by binary code [74,75]. As discussed in the private table, crossover standards have several weak and strong points. As a result, the major motivation for proving the crossover standards is to propose and demonstrate a distinctive crossover technique. Using novel generated algorithms, crossover operators will take on those responsibilities with varying degrees of precision to find global convergence quickly. The suggested technique is based on LDF for gene crossings. Thus, some following points highlight the novel properties of LPX that should be obvious.

- When there are limitations on the input values that can be used, the Lagrange multiplier technique can be utilized. This technique is used to determine the maximum or minimum of a multivariable function [76,77];
- The main goal is to find locations where the contour lines of the multivariable function and the input space are adjacent to one another [76];
- The constrained population-based optimization problem is transformed into an unconstrained problem using the Lagrange multiplier technique [77,78], which provides the optimum solution when used as a point in the crossover standard;
- Optimization with the Lagrangian method explores the application of Lagrange multiplier methods to find local and global convergence for Lagrangian methods under constraint minimization and maximization [79];
- Based on LDF, LPX attempts to calculate the most appropriate offspring, which often involves taking a fairly significant step away from each parent.

The event points show that the Lagrange multiplier method is used for determining a function of local maxima and local minima with equality constraints or requirements. This issue contains the requirement that one or more equations be exactly solved by the selected variable values [80]. The correlation between the gradient (slope) of the function and the gradients (slopes) of the constraints leads to a natural form of formulation of the global problem, characterized as the Lagrangian Function [81,82]. These points around or near the slopes may have a function to generate new genes within the specific chromosome.

According to Figure 18, there may be an objective function, signified as $f(x, y)$, which must be optimized while it is subject to the restriction $g(x, y) = c$. Let us put this problem statement into perspective before delving further into its discussion. The Gradient $\nabla f(x, y)$ is a vector which, at every position (x, y), indicates the direction in which f should be increased as effectively as possible. f will rise as long as the stationary point continues to move in that direction, which is the steepest curve in this direction. Any function's gradient that is calculated at a specific position (x, y) will always result in a vector that

is perpendicular to the contour line that passes through that point. It is worth noting the exploring point must always remain on the constraint curve $g(x, y) = c$ when ascending this top of the climax point (at Gradient Vector), such as the global point. In other words, the only directions the solution can travel are tangents to this constraint curve. These tangents' values remain constant throughout the constraint curve $g(x, y) = c$ because they are orthogonal to the gradient of the limitation function g. Then Gradient Vector must track the optimizer's motion on the surface of f as the optimizer moves along the constraint curve $g(x, y) = c$. The solution point should continue to raise f, even if it is shifted in a direction along the Gradient $\nabla f(x, y)$ that has a non-trivial component. A gradient can only be shifted orthogonally to the gradient $\nabla f(x, y)$ once if it moves only in the direction orthogonal to the gradient. In this case, the solution has reached a local maximum. The gradients of f and g are now pointing in the same general direction.

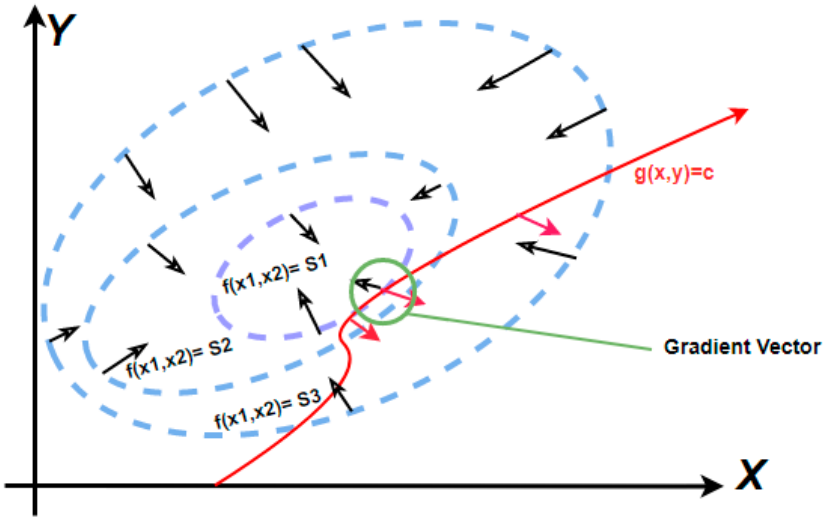

Figure 18. The Lagrange multiplier shows the contour lines of the tangent function when gradient vectors are parallel.

Thus, as illustrated in Figure 18, the restriction $g(x, y) = c$ appears as a red curve. The blue curves are characteristics $f(x_i, y_i)$. Because $S1 > S2$, the point where the red constraint tangentially contacts a blue curve is the maximum $f(x_1, y_2)$, which means tangential to $S1$ in the sideways constraint. Moreover, it shows that the assumption that line graphs are tangential has no bearing on the size of any of these gradient vectors, but it is well. When two vectors have the same orientation, we may multiply one by a constant to obtain the other [83]. The Lagrange multiplier works by assuming that the local minimums and maximums along the constraint occur when the constraint is parallel to the contours. This is stated in $S1,\ S2,\ S3$.

In each of the above scenarios, assuming the point is on the contour line, Figure 18 shows how to calculate the stationary point of $\mathcal{L}$ given a function $\nabla f(x, y) = \lambda \nabla g(x, y)$ and the Lagrange multiplier λ. Thus, it uses a general method, called the Lagrange multiplier method, as formulated in Equation (12), for solving constrained optimization problems [84,85]. Equation (12) is explained in the following evaluation.

$$\mathcal{L}(x, y) = f(x, y) - cg(x, y) = c \text{ If } c \text{ is a constant} \mathcal{L}(x, y) = f(x, y) - g(x, y)$$

From these points, it is proved that the Lagrange multiplier λ in Equation (12) is improved by maximizing (or minimizing).

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda\, g(x, y) \tag{12}$$

Gradient vectors indicate that this model should be fixed to compute many examples and to determine the optimum point. It will, for example, assist with physical routing. It

will be used to select the smallest point to find the shortest physical path. However, the Lagrange dual function may be convenient for identifying several global solutions [86]. As such, the LDF theorem depends on real equation samples [87]. To identify several local points, the theorem would be formulated using a novel crossover operator. Thus, each station should generate a gene for offspring from parent chromosome(s). The developed alternative to the Conic Duality theory is called LDF theory. The Lagrangian Duality Problem theory is more applicable to generic nonlinear limitations [88]. From the stationary point in Equation (12), an offspring is derived in Equation (13) using the LDF theorem.

$$\text{Offspring} = \mathcal{L}\,(x_1, x_2, \alpha) = f(x_1, x_2) - \alpha\; g(x_1, x_2) = f(x_1, x_2) - \sum_{i=1}^{n=2} \alpha\, g_i\,(x_1, x_2) \tag{13}$$

$$\begin{aligned} &\text{For Offspring 1}: \; f(x_1, x_2) = (x_1 - x_2)^{\,2} + (x_2 - 1)^{\,2} \\ &\text{Subject to } g_1(x_1, x_2) = x_1 + 2x_2 - 1 \\ &\qquad\qquad g_2(x_1, x_2) = 2x_1 + x_2 - 1 \\ &\text{For Offspring 2}: \; f(x_2, x_1) = (x_2 - x_1)^{\,2} + (x_1 - 1)^{\,2} \\ &\text{Subject to } g_1(x_2, x_1) = x_2 + 2x_1 - 1 \\ &\qquad\qquad g_2(x_2, x_1) = 2x_2 + x_1 - 1 \end{aligned}$$

Consequently, it is possible to generate Offspring1 and Offspring2 at the stationary point by including Equation (13) if the Lagrangian Multiplayer has produced a random value for (α) in the specified range according to the population-based generations crossover rate. Therefore, we developed Equations (14) and (15).

$$\text{Offspring1} = (x_1 - x_2)^{\,2} + (x_2 - 1)^{\,2} - (\alpha\,(x_1 + 2x_2 - 1) + \alpha\,(2x_1 + x_2 - 1)) \tag{14}$$

$$\text{Offspring2} = (x_2 - x_1)^{\,2} + (x_1 - 1)^{\,2} - (\alpha\,(x_2 + 2x_1 - 1) + \alpha\,(2x_2 + x_1 - 1)) \tag{15}$$

In this case, the proposed standard for this crossover standard (LPX) is the same as the real-coded crossover. It creates offspring solutions by inserting a sub-sequence from one of the genomes into the parent, while the initial order would be as many point states as reasonable, such as in the example calculated by Equations (14) and (15) to generate new offspring, which is illustrated in Figure 19 as a modification sample crossover. While x_1 is indicated as Gene two G_1 in chromosome one, x_2 is indicated as Gene two G_2 in chromosome two when the stationary multiplier is defined randomly as $(\alpha = 0.2)$. In the next section, the comparison results are improved heuristically and statistically.

Chromosome 1	0.88	0.13	0.25	>>>	Offspring 1	0.88	0.4177	0.25
Chromosome 2	0.64	0.94	0.35		Offspring 2	0.64	1.171	0.35

Figure 19. Create two new offspring depending on LPX.

5. Results and Discussion

There are several standard benchmark test functions available to evaluate these standard operators and the performance of population-based algorithms. In Table 3, three test functions are selected to analyze this newly introduced operator. However, classical test functions are divided into unimodal modal test functions, multimodal test functions, and composite test functions [89]. Each set of these test functions is used to benchmark certain aspects of the algorithm. For determining the exploitation level and convergence of the standards in the population algorithms, we select these three functions from the unimodal test functions. LPX and BX are compared in the first part of this subsection as heuristic evaluation results. The second part compares our standard operator with BX, SBX, and Qubit-X [45,90] operators, while they are tested by LPB as a population-based algorithm as part of exploitation and convergence results. A Wilcoxon Rank Sum test is used to test all results statistically in the last part as statistical evaluation results. To determine the

most appropriate solution, all of the results are affected by the random number value. We illustrated the time consumed during processing to execute dynamic cost minimization.

Table 3. Three unimodal benchmark test functions [83].

TF	Functions	Range	Dimension
TF1	$F(x) = \sum_{i=1}^{n} {x_i}^2$	$[-100, 100]$	10
TF3	$F(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]$	10
TF7	$F(X) = \sum_{i=1}^{n} i{x_i}^4 + roundom\ [0, 1]$	$[-1.28, 1.28]$	10

5.1. Heuristic Evaluation Results

Using mathematical comparisons, this evaluation aimed to identify major usability and performance problems with the standard operators. Tests achieved a total of 100 stochastic generations (genes) for different pairs parents (chromosomes) with crossover rate values (α) are 0.3, 0.5, and 0.7. This difference in random values assists newly proposed algorithms in choosing the best range when selecting the values and achieving the best optima rapidly. Several standards have been reviewed in the previous section, but the test focuses on BX and SBX crossovers. As a result, these two standard heuristic results are compared with LPX results. In addition, performance can be measured based on the statistical value produced. Based on the random value, the more significant value is calculated according to the average, indicating a superior outcome [91–93]. Based on crossovers, the outcomes of mathematical calculations from chromosome parents are used to generate a gene in chromosomal offspring [94]. The operation of the arithmetic process is found by applying Equations (4) and (5) for BX, Equations (8) and (9) for SBX, and Equations (14) and (15) for LPX.

The test relies on three unimodal functions [95] based on a group of traditional benchmark functions to determine approval for each gene on a chromosome. The three-test equation in the benchmarks for a single variable includes test function one (TF1), test function three (TF3), and test function seven (TF7), as shown in Table 3 [96]. An algorithm must avoid all local optimal solutions to reach the global optimum, and these sample test functions can aid in mapping out a strategy for exploration. The single result has statistically calculated the summation of generation on chromosome parent and then the average (Mean) and standard deviation (STD) have been mentioned to compare all standards relying on alpha value (α) as a random value that will be generated during the steps of the algorithm.

The Lagrangian functions look for new points in complex applications by searching around the largest local optimum. As has been explained earlier, the Lagrange multiplier method in mathematics is a technique for identifying the local maximum or minimum value of an action that is subject to equality requirements. It is subject to the requirement that the exact values of the variables chosen must satisfy one or more equations as has been proved in LPX. Thus, Table 4 represents the results of the tests of summation, Mean, and STD values for three standard operators by three test functions. For all values of alpha (α), it is found that LPX is more powerful than BX and SBX. Although BX and SBX were reasonable in previous generations, this seems such as a more reasonable option now. In addition, results show that TF7 has high convergence and exploitation for all alpha values. Similarly, LPX could be computed to show the ranking of social classes and stochastically analyze metaheuristic algorithms [97].

Nevertheless, all standards were reasonable according to evolution's invincible algorithms. Thus, we proved two examples of the maximum and minimum genes on chromosome parents for TF1 for three alpha values. The performance of BX and SBX in TF1 is trivial, but LPX shows slightly better results. Moreover, LPX has demonstrated maximum and minimum gene expression when compared to both BX and SBX standards. Moreover, the process of convergence in LPX revealed that the relationships between gen-

erations are higher than the other two crossover standards for all alpha values in TF3, even though it showed the relationship between generations for maximum and minimum gene values based on TF3. Based on TF7 results for all alpha values, LPX has a strangely high result compared to BX and SBX. This is because they have defined and obtained maximum genes and minimum genes. TF7 shows how the chromosome comparative results of the divergent max and min genes are evaluated and highlighted. The frequency of subsequent generations derived from the parent genes is represented on the Y axis in Figures 20–28. This is because there are different max and min genes on each chromosome, which are used to evaluate the comparison results. Ultimately, the heuristic evaluation results indicate that the LPX frequencies are rapidly maximized for these three test functions.

Table 4. The performance result test for selected crossover standards with LPX.

	Standards	**BX**			**SBX**			**LPX**		
α	**Test Functions**	**Sum**	**Mean**	**STD**	**Sum**	**Mean**	**STD**	**Sum**	**Mean**	**STD**
0.2	TF1	42.36	0.42	0.30	31.37	0.31	0.32	1737.56	17.38	17.09
	TF3	60.00	0.60	0.66	60.00	0.60	0.66	3197.01	31.97	31.27
	TF7	779.24	7.79	10.78	487.58	4.88	9.52	**1,937,510.53**	**19,375.11**	**33,631.08**
0.5	TF1	30.00	0.30	0.30	38.58	0.39	0.30	2776.00	27.76	26.17
	TF3	60.00	0.60	0.66	60.00	0.60	0.66	5273.89	52.74	50.06
	TF7	461.88	4.62	9.41	661.72	6.62	10.21	**4,348,187.50**	**43,481.88**	**64,658.48**
0.7	TF1	35.49	0.35	0.31	46.82	0.47	0.30	3648.64	36.49	34.78
	TF3	60.00	0.60	0.66	60.00	0.60	0.66	7019.17	70.19	67.65
	TF7	579.02	5.79	9.88	941.45	9.41	11.81	**7,300,002.73**	**73,000.03**	**109,185.64**

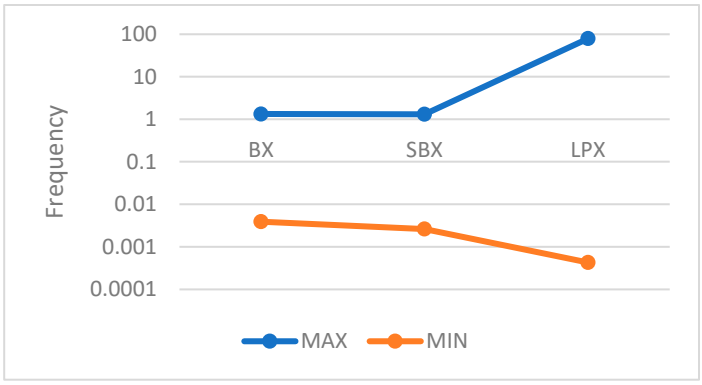

Figure 20. TF1(α = 0.2).

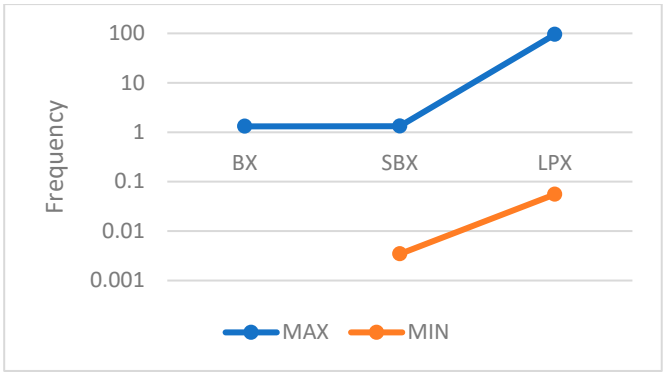

Figure 21. TF1(α = 0.5).

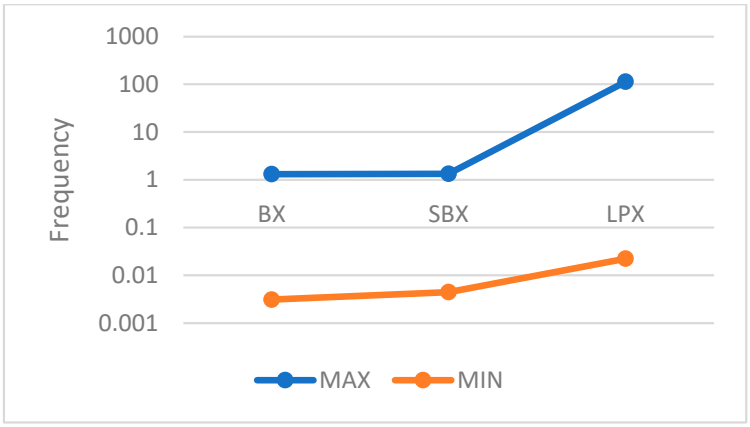

Figure 22. TF1(α = 0.7).

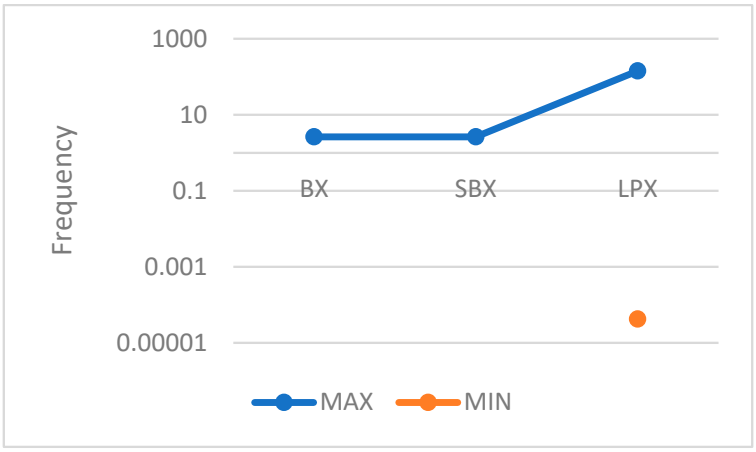

Figure 23. TF3(α = 0.2).

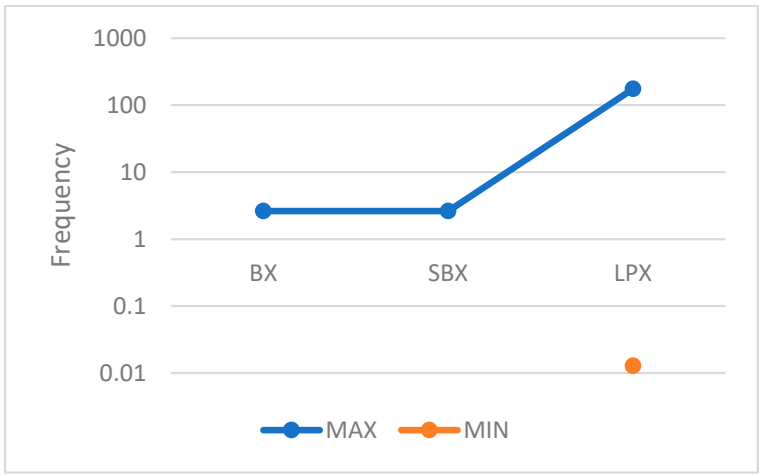

Figure 24. TF3 (α = 0.5).

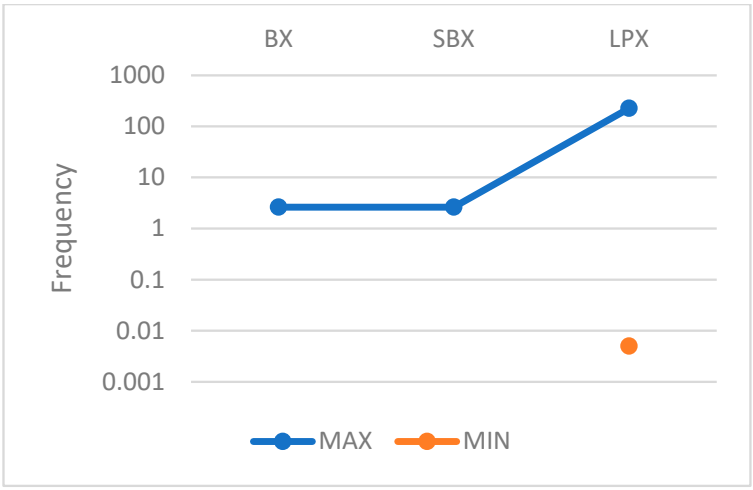

Figure 25. TF3($\alpha = 0.7$).

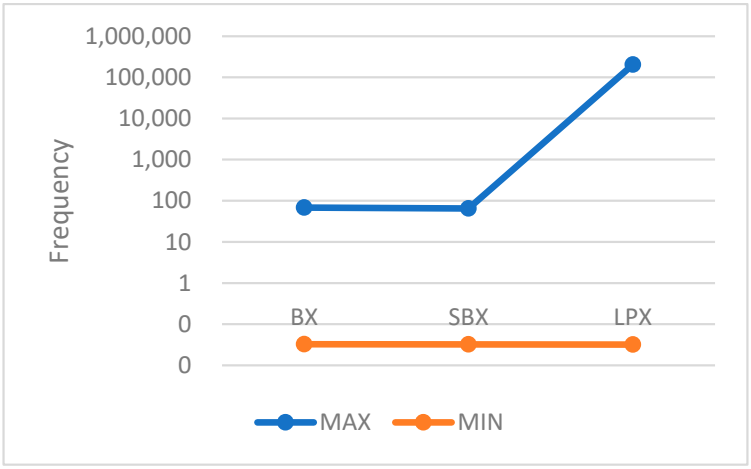

Figure 26. TF7 ($\alpha = 0.2$).

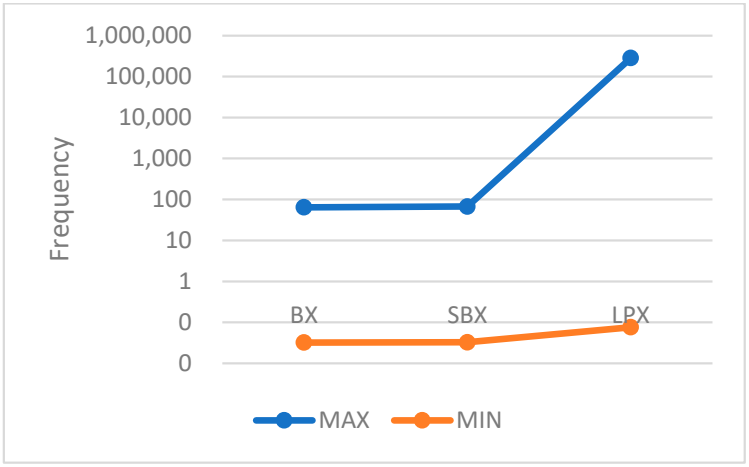

Figure 27. TF7 ($\alpha = 0.5$).

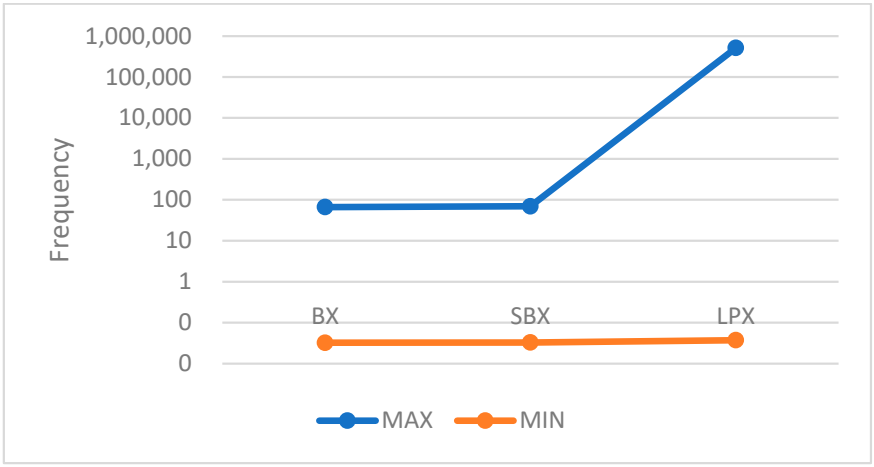

Figure 28. TF7 (α = 0.7).

5.2. Exploitation and Convergence Evaluation Results

The effectiveness of the LPX has been assessed using a range of experiments. These studies are designed differently to examine the behaviors of the LPX and reflect both its qualitative and quantitative characteristics. LPX's exploitation capabilities and convergence behavior for solving problems are demonstrated through a qualitative analysis that employs exploration and average fitness values. For this purpose, we choose the recently developed population algorithm LPB, which is mentioned in the previous section. In the quantitative analysis, the proposed LPX has been compared with other standard crossovers such as BX, SBX, and Qubit-X. We have also selected three classical test functions in Table 3. The effectiveness was evaluated by testing escape ability from local optima, and convergence speed by summing the elapsed time to obtain the fitness point. The novelty of this study lies in its proving the effectiveness of the random value; therefore, three different random values for each test function have been selected. Over 500 iterations, each standard with a different random value is tested with the LPB algorithm, which is executed 30 times using 80 search agents. The sum, STD, and processing time are calculated. The LPX metrics, sum, and STD are rated first or second on almost all three test functions. Due to the results and convergence in the fastest time calculations; thus, bold results are shown in Table 5. Specifically, the comparison of the LPX standard's unimodal test functions compared to others at TF7 and all random values showed that LPX had a faster rate of exploitation and convergence. Moreover, the execution time for LPX is marginally faster than TF7 calculations. In addition, if the random value is near (0.5) during TF3, LPX is the most optimal rate of convergence and exploitation.

Table 5. The crossover operator's comparison results of classical test functions.

Test Fun.	α	LPX			SBX			BX			Qubit-X		
		Mean	STD	Time (S.)	Mean	STD	Time (S.)	Mean	STD	Time (S.)	Mean	STD	Time (S.)
TF1	0.2	0.0635	0.0184	**141.740**	**0.01751**	**0.0236**	161.423	0.04428	0.0446	150.384	0.1758	0.0926	144.474
	0.5	0.0680	0.0281	**149.798**	**0.04161**	**0.0270**	162.160	0.04178	0.0323	157.700	0.1411	0.0510	151.992
	0.7	0.0596	0.0279	**151.112**	**0.02959**	**0.0172**	188.501	0.04150	0.0294	163.809	0.1425	0.1045	157.042
TF3	0.2	43.5652	**24.8093**	**159.289**	83.37500	59.0221	178.260	**41.60497**	28.4041	169.970	120.7210	73.2963	164.986
	0.5	**40.4260**	**26.2073**	165.144	78.18210	52.1304	**161.057**	52.58699	37.7840	169.130	175.6268	119.6147	165.608
	0.7	66.7197	58.8220	164.711	85.92191	71.4473	180.216	**50.67240**	**49.2447**	167.669	81.3198	52.6167	**164.450**
TF7	0.2	**0.0048**	**0.0031**	**143.005**	0.01351	0.0188	153.884	0.00624	0.0045	156.457	0.0076	0.0043	160.718
	0.5	**0.0049**	**0.0027**	152.029	0.00770	0.0066	164.322	0.00616	0.0029	162.973	0.0094	0.0051	**147.530**
	0.7	**0.0052**	**0.0033**	157.893	0.00773	0.0056	164.504	0.00709	0.0042	163.520	0.0123	0.0064	**155.061**

5.3. Statistical Evaluation Results

The optimization algorithms are stochastic, thus several non-parametric statistical tests, including Wilcoxon signed-rank sum, and analysis of variance (ANOVA) can be used to quantitatively examine the algorithms' overall performance. Before using a standard to solve optimization problems, its applicability should be determined statistically. The comparison is conducted only between LPX with SBX and LPX with BX standards by the Wilcoxon signed-rank sum test in Table 6. The LPX results are considered significant and thus rejected null hypnosis in all statistical tests; all p-value are smaller than 0.05.

Table 6. The Wilcoxon rank sum test (p-value) between crossovers operator for random generations.

Test Functions	α	Standards	
		LPX vs. SBX	**LPX vs. BX**
TF1	0.2	3.6746×10^{-16}	5.3124×10^{-16}
	0.5	4.7409×10^{-17}	4.0951×10^{-17}
	0.7	4.7409×10^{-17}	3.8618×10^{-17}
TF3	0.2	8.5768×10^{-16}	8.5768×10^{-16}
	0.5	9.5355×10^{-17}	9.5355×10^{-17}
	0.7	8.2482×10^{-17}	8.2482×10^{-17}
TF7	0.2	1.6983×10^{-16}	2.6103×10^{-16}
	0.5	4.0951×10^{-17}	3.2378×10^{-17}
	0.7	3.2378×10^{-17}	2.0802×10^{-17}

Based on Table 7, the Wilcoxon rank sum test was used to determine the statistical result (p-value) for all crossover operators with LPX. Qubit-X means significant results for all three test functions except TF2 ($\alpha = 0.7$), which is insignificant and rejects the null hypothesis. Moreover, significant results with SBX have been determined for all test functions for all values without ($\alpha = 0.7$) for TF3 and TF7. In addition, the significant value of LPX with the BX standard is significant for TF1 for all alpha values. This is due to a lower value than 0.05.

Table 7. The Wilcoxon rank sum test (p-value) between standards for the LPB algorithm.

Test Functions	α	Standards		
		LPX vs. SBX	**LPX vs. BX**	**LPX vs. Qubit-X**
TF1	0.2	0.000031	0.002415	0.000005
	0.5	0.000241	0.001965	0.000002
	0.7	0.00042	0.015658	0.000031
TF3	0.2	0.002765	**0.517048**	0.000002
	0.5	0.006836	**0.318491**	0.000002
	0.7	**0.328571**	**0.393334**	**0.271155**
TF7	0.2	0.000716	**0.298944**	0.009271
	0.5	0.044919	**0.085896**	0.000664
	0.7	**0.071903**	**0.057096**	0.000058

6. Conclusions

In conclusion, the most evolutionary metaheuristic algorithm is based on efficient computation techniques that are applied effectively to different problems. Encoding techniques and standard operators, particularly crossover operators for enhanced metaheuristic

optimization, have been responsible for determining their outcomes. In this research, several crossover standards have been collected to assist researchers in obtaining an effective crossover operator and selecting a global solution to the problem. The majority of them were easy to calculate; thus, the calculation was faster, as illustrated. In addition, they allowed for the production of a wide range of offspring from two parameters as determined by two parent values. In addition, the study provided an improved standard option crossover operator to assist in the enhancement of novel mathematical evolutionary algorithms.

In this study, standard crossover operators such as binary-coded crossover, real-coded crossover (floating point), and order-coded problem crossover were mathematically and systematically reviewed. Then, we recommend the optimum crossover standard operator, and LPX has been discovered to be a novel mathematical approach to crossover standards. The LPX is derived from the LDF theorem, which is based on the stationary Lagrange multiplier. Additionally, the capability of the technique was evaluated heuristically for the generation of parent chromosomes and compared with BX and SBX. LPX has the most impressive performance in terms of rate of exploitation and convergence fitness for selected random values. In this manner, these random values assist in selecting the most appropriate range when generating newly developed population-based populations. Moreover, we evaluated LPX results by LPB as a metaheuristic algorithm when LPX, SBX, BX, and Qubit-X were determined by all algorithms. Most test functions have reasonable convergence in the exploitation evaluation for selected random values. Finally, most statistical results for LPX with other standards have proved the significant hypothesis.

In future works, the researcher can evaluate LPX by comparing it with other standardized crossover techniques based on binary form, real-code form, or order-coded problem methods for crossover, and it will be proved by comparison with more test functions such as two-dimensional test functions. LPX may be determined with functional tests that illustrate multimodal test functions and composite test functions. From another perspective, the researchers improved a novel evolutionary metaheuristic algorithm based on populations through single-objective optimization or multi-objective optimization [98,99]. Moreover, LPX equations will be improved based on algorithms such as frequency-modulated synthesis with multi-parent crossover [100] in the Ant Nesting Algorithm [101], Child Drawing Development Optimization [102], and Capuchin Search Algorithm [103]. Thus, LPX asserts that it can deliver the most effective fitness solution. Several research studies by heuristic suggest that large-scale structural optimization, modified to evaluate several strategies pooled with programming techniques, can be used for quantitative appraisal in compliance with enforcement learning [104–106]. As a result, LPX might be used for evaluating time-scales in heuristic game problems such as A* or (BA*) search algorithms [107].

Author Contributions: Conceptualization, A.M.A. and T.A.R.; methodology, A.M.A.; software, A.M.A.; validation, A.M.A.; formal analysis, A.M.A. and T.A.R.; investigation, T.A.R.; data curation, A.M.A.; writing—original draft, A.M.A.; writing—review and editing, A.M.A. and T.A.R.; supervision, T.A.R.; funding acquisition, A.M.A. and T.A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request. To find MATLAB code check this repository: https://github.com/asoaladdin/LPX.git (accessed on 3 January 2023).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Osaba, E.; Diaz, F.; Onieva, E.; Carballedo, R.; Perallos, A. AMCPA: A population metaheuristic with adaptive crossover probability and multi-crossover mechanism for solving combinatorial optimization problems. *Int. J. Artif. Intell.* **2014**, *12*, 1–23.
2. Potra, F.A.; Wright, S.J. Interior-point methods. *J. Comput. Appl. Math.* **2000**, *124*, 281–302. [CrossRef]
3. Goldberg, D.E.; Holland, J.H. Genetic algorithms and Machine Learning. *Mach Learn.* **1988**, *3*, 95–99. [CrossRef]
4. Karaboga, D.; Basturk, B. On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.* **2008**, *8*, 687–697. [CrossRef]
5. Price, K.V. Differential Evolution. In *Handbook of Optimization*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 187–214.
6. Rahman, C.M.; Rashid, T.A. A new evolutionary algorithm: Learner performance based behavior algorithm. *Egypt. Inform. J.* **2021**, *22*, 213–223. [CrossRef]
7. Yang, X.S. Social algorithms. *arXiv* **2018**, arXiv:1805.05855.
8. Beyer, H.G.; Deb, K. On self-adaptive features in real-parameter evolutionary algorithms. *IEEE Trans. Evol. Comput.* **2001**, *5*, 250–270. [CrossRef]
9. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
10. De Jong, K. Learning with genetic algorithms: An overview. *Mach. Learn.* **1988**, *3*, 121–138. [CrossRef]
11. Sivanandam, S.N.; Deepa, S.N. Genetic Algorithms. In *Introduction to Genetic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 15–37.
12. Hassanat, A.B.; Alkafaween, E.A. On enhancing genetic algorithms using new crossovers. *Int. J. Comput. Appl. Technol.* **2017**, *55*, 202–212. [CrossRef]
13. Ouattara, A.; Aswani, A. Duality approach to bilevel programs with a convex lower level. In Proceedings of the 2018 Annual American Control Conference (ACC), Milwaukee, WI, USA, 27–29 June 2018; IEEE: New York, NY, USA, 2016; pp. 1388–1395.
14. Karimi-Mamaghan, M.; Mohammadi, M.; Meyer, P.; Karimi-Mamaghan, A.M.; Talbi, E.G. Machine Learning at the service of Meta-heuristics for solving Combinatorial Optimization Problems: A State-of-the-Art. *Eur. J. Oper. Res.* **2022**, *296*, 393–422. [CrossRef]
15. Bäck, T.; Fogel, D.B.; Michalewicz, Z. (Eds.) *Evolutionary Computation 1: Basic Algorithms and Operators*; CRC Press: Boca Raton, FL, USA, 2018.
16. Blum, C.; Roli, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.* **2003**, *35*, 268–308. [CrossRef]
17. Hussain, A.; Muhammad, Y.S.; Nauman Sajid, M.; Hussain, I.; Mohamd Shoukry, A.; Gani, S. Genetic algorithm for traveling salesman problem with modified cycle crossover operator. *Comput. Intell. Neurosci.* **2017**, 7430125. [CrossRef]
18. Ahmed, Z.H. Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. *Int. J. Biom. Bioinform.* **2010**, *3*, 96.
19. Kaya, Y.; Uyar, M. A novel crossover operator for genetic algorithms: Ring crossover. *arXiv* **2011**, arXiv:1105.0355.
20. Dey, N. (Ed.) *Advancements in Applied Metaheuristic Computing*; IGI Global: Hershey, PA, USA, 2013.
21. Puljić, K.; Manger, R. Comparison of eight evolutionary crossover operators for the vehicle routing problem. *Math. Commun.* **2013**, *18*, 359–375.
22. Umbarkar, A.J.; Sheth, P.D. Crossover operators in genetic algorithms: A review. *ICTACT J. Soft Comput.* **2015**, *6*, 1083–1092.
23. Pongcharoen, P.; Stewardson, D.J.; Hicks, C.; Braiden, P.M. Applying designed experiments to optimize the performance of genetic algorithms used for scheduling complex products in the capital goods industry. *J. Appl. Stat.* **2001**, *28*, 441–455. [CrossRef]
24. Hameed, W.M.; Kanbar, A.B. A comparative study of crossover operators for genetic algorithms to solve travelling salesman problem. *Int. J.Res. –Granthaalayah* **2017**, *5*, 284–291. [CrossRef]
25. Gain, A.; Dey, P. Adaptive Position–Based Crossover in the Genetic Algorithm for Data Clustering. *Recent Adv. Hybrid Metaheuristics Data Clust.* **2020**, 39–59.
26. Larranaga, P.; Kuijpers, C.M.H.; Murga, R.H.; Inza, I.; Dizdarevic, S. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artif. Intell. Rev.* **1999**, *13*, 129–170. [CrossRef]
27. Hilding, F.G.; Ward, K. Automated Operator Selection on Genetic Algorithms. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 903–909.
28. Katayama, K.; Sakamoto, H.; Narihisa, H. The efficiency of hybrid mutation genetic algorithm for the travelling salesman problem. *Math. Comput. Model.* **2000**, *31*, 197–203. [CrossRef]
29. Patel, R.; Collins, D.; Bullock, S.; Swaminathan, R.; Blake, G.M.; Fogelman, I. The effect of season and vitamin D supplementation on bone mineral density in healthy women: A double-masked crossover study. *Osteoporos. Int.* **2001**, *12*, 319–325. [CrossRef]
30. Thapatsuwan, P.; Chainate, W.; Pongcharoen, P. Investigation of genetic algorithm parameters and comparison of heuristic arrangements for container packing problem. *Curr. Appl. Sci. Technol.* **2006**, *6*, 274–284.
31. Herrera, F.; Lozano, M.; Verdegay, J.L. Fuzzy connectives based crossover operators to model genetic algorithms population diversity. *Fuzzy Sets Syst.* **1997**, *92*, 21–30. [CrossRef]
32. Ono, I. Real-coded genetic algorithm for function optimization using unimodal normal distribution crossover. In Proceedings of the 7th ICGA, East Lansing, MI, USA, 19–23 July 1997; pp. 246–253.

33. Kita, H.; Ono, I.; Kobayashi, S. Theoretical analysis of the unimodal normal distribution crossover for real-coded genetic algorithms. *Trans. Soc. Instrum. Control Eng.* **1999**, *35*, 1333–1339. [CrossRef]
34. Bosch, W. Discrete Crossover Analysis. In *Dynamic Planet*; Springer: Berlin/Heidelberg, Germany, 2007; Volume 35, pp. 131–136.
35. Tawhid, M.A.; Ali, A.F. Simplex particle swarm optimization with arithmetical crossover for solving global optimization problems. *Opsearch* **2016**, *53*, 705–740. [CrossRef]
36. Ling, S.H.; Leung, F.H. An improved genetic algorithm with average-bound crossover and wavelet mutation operations. *Soft Comput.* **2007**, *11*, 7–31. [CrossRef]
37. Ackora-Prah, J.; Gyamerah, S.A.; Andam, P.S. A heuristic crossover for portfolio selection. *Appl. Math. Sci.* **2014**, *8*, 3215–3227. [CrossRef]
38. García-Martínez, C.; Lozano, M.; Herrera, F.; Molina, D.; Sánchez, A.M. Global and local real-coded genetic algorithms based on parent-centric crossover operators. *Eur. J. Oper. Res.* **2008**, *185*, 1088–1113. [CrossRef]
39. Hogue, R.W.; Singh, S.; Brooker, S. Spin crossover in discrete polynuclear iron (II) complexes. *Chem. Soc. Rev.* **2018**, *47*, 7303–7338. [CrossRef] [PubMed]
40. Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Future Gener. Comput. Syst.* **2020**, *111*, 300–323. [CrossRef]
41. Wang, G.G. Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Comput.* **2018**, *10*, 151–164. [CrossRef]
42. Yang, Y.; Chen, H.; Heidari, A.A.; Gandomi, A.H. Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Syst. Appl.* **2021**, *177*, 114864. [CrossRef]
43. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [CrossRef]
44. Tu, J.; Chen, H.; Wang, M.; Gandomi, A.H. The colony predation algorithm. *J. Bionic Eng.* **2021**, *18*, 674–710. [CrossRef]
45. Zamani, H.; Nadimi-Shahraki, M.H.; Gandomi, A.H. QANA: Quantum-based avian navigation optimizer algorithm. *Eng. Appl. Artif. Intell.* **2021**, *104*, 104314. [CrossRef]
46. Albrechtsen, S.; Rasmussen, S.; Thoresen, S.; Irgens, L.M.; Iversen, O.E. *Pregnancy Outcome in Women before and after Cervical Conisation: Population Based Cohort Study*; BMJ: London, UK, 2008; p. 337.
47. Hassanat, A.; Almohammadi, K.; Alkafaween, E.A.; Abunawas, E.; Hammouri, A.; Prasath, V.S. Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach. *Information* **2019**, *10*, 390. [CrossRef]
48. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126. [CrossRef]
49. Sumathi, S.; Hamsapriya, T.; Surekha, P. *Evolutionary Intelligence: An Introduction to Theory and Applications with Matlab*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2008.
50. Takahashi, M.; Kita, H. A crossover operator using independent component analysis for real-coded genetic algorithms. In Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea, 27–30 May 2001; IEEE Cat. No. 01TH8546. IEEE: New York, NY, USA, 2002; Volume 1, pp. 643–649.
51. Herrera, F.; Lozano, M.; Sánchez, A.M. Hybrid crossover operators for real-coded genetic algorithms: An experimental study. *Soft Comput.* **2005**, *9*, 280–298. [CrossRef]
52. Picek, S.; Golub, M. Comparison of a crossover operator in binary-coded genetic algorithms. *WSEAS Trans. Comput.* **2010**, *9*, 1064–1073.
53. Magalhaes-Mendes, J. A comparative study of crossover operators for genetic algorithms to solve the job shop scheduling problem. *WSEAS Trans. Comput.* **2013**, *12*, 164–173.
54. Hu, X.B.; Paolo, E.D. An Efficient Genetic Algorithm with Uniform Crossover for the Multi-Objective Airport Gate Assignment Problem. In *Multi-Objective Memetic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 71–89.
55. Singh, D.; Singh, V.; Ansari, U. Binary particle swarm optimization with crossover operation for discrete optimization. *Int. J. Comput. Appl.* **2011**, *28*, 15–20. [CrossRef]
56. Kötzing, T.; Sudholt, D.; Theile, M. How crossover helps in pseudo-Boolean optimization. In Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, Dublin, Ireland, 12–16 July 2011; pp. 989–996.
57. Zhang, X.; Zhang, Q.; Zhang, X. Nonuniform antenna array design by parallelizing three-parent crossover genetic algorithm. *EURASIP J. Wirel. Commun. Netw.* **2017**, *1*, 1–7. [CrossRef]
58. Devaraj, D.; Selvabala, B. Real-coded genetic algorithm and fuzzy logic approach for real-time tuning of proportional–integral–derivative controller in automatic voltage regulator system. *IETGener. Transm. Distrib.* **2009**, *3*, 641–649. [CrossRef]
59. Lee, K.Y.; Mohamed, P.S. A real-coded genetic algorithm involving a hybrid crossover method for power plant control system design. In Proceedings of the 2002 Congress on Evolutionary Computation, Honolulu, HI, USA, 12–17 May 2002; CEC'02 (Cat. No. 02TH8600). IEEE: New York, NY, USA, 2002; Volume 2, pp. 1069–1074.
60. Ankudinov, A.L.; Bouldin, C.E.; Rehr, J.J.; Sims, J.; Hung, H. Parallel calculation of electron multiple scattering using Lanczos algorithms. *Phys. Rev. B* **2002**, *65*, 104107. [CrossRef]
61. Picek, S.; Jakobovic, D.; Golub, M. On the recombination operator in the real-coded genetic algorithms. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013; IEEE: New York, NY, USA, 2013; pp. 3103–3110.

62. Lim, S.M.; Sultan, A.B.M.; Sulaiman, M.N.; Mustapha, A.; Leong, K.Y. Crossover and mutation operators of genetic algorithms. *Int. J. Mach. Learn. Comput.* **2017**, *7*, 9–12. [CrossRef]
63. Hamid, Z.A.; Musirin, I.; Othman, M.M.; Rahim, N.A. Efficient power scheduling via stability index-based tracing technique and blended crossover continuous ant colony optimization. *Aust. J. Basic Appl. Sci.* **2011**, *5*, 1335–1347.
64. Zou, D.; Liu, H.; Gao, L.; Li, S. A novel modified differential evolution algorithm for constrained optimization problems. *Comput. Math. Appl.* **2011**, *61*, 1608–1623. [CrossRef]
65. Deep, K.; Thakur, M. A new crossover operator for real coded genetic algorithms. *Appl. Math. Comput.* **2007**, *188*, 895–911. [CrossRef]
66. Azevedo, C.R.B. Geração de Diversidade Na Otimização Dinâmica Multiobjetivo Evolucionária Por Paisagens de Não-Dominância. Master's Thesis, Universidade Federal de Pernambuco, Recife, Brazil, 2011.
67. Chacón, J.; Segura, C. Analysis and enhancement of simulated binary crossover. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; IEEE: New York, NY, USA, 2018; pp. 1–8.
68. Deb, K.; Beyer, H.G. Self-adaptive genetic algorithms with simulated binary crossover. *Evol. Comput.* **2001**, *9*, 197–221. [CrossRef] [PubMed]
69. Liao, C.C.; Ting, C.K. A novel integer-coded memetic algorithm for the set $ k $-cover problem in wireless sensor networks. *IEEE Trans. Cybern.* **2017**, *48*, 2245–2258. [CrossRef] [PubMed]
70. Desjardins, B.; Falcon, R.; Abielmona, R.; Petriu, E. Planning Robust Sensor Relocation Trajectories for a Mobile Robot with Evolutionary Multi-Objective Optimization. In *Computational Intelligence in Wireless Sensor Networks*; Springer: Cham, Switzerland, 2017; pp. 179–210.
71. Koohestani, B. A crossover operator for improving the efficiency of permutation-based genetic algorithms. *Expert Syst. Appl.* **2020**, *151*, 113381. [CrossRef]
72. Yoon, H.S.; Moon, B.R. An empirical study on the synergy of multiple crossover operators. *IEEE Trans. Evol. Comput.* **2002**, *6*, 212–223. [CrossRef]
73. Mudaliar, D.N.; Modi, N.K. Unraveling travelling salesman problem by genetic algorithm using m-crossover operator. In Proceedings of the 2013 International Conference on Signal Processing, Image Processing & Pattern Recognition, Coimbatore, India, 7–8 February 2013; IEEE: New York, NY, USA, 2013; pp. 127–130.
74. Viana, M.S.; Morandin Junior, O.; Contreras, R.C. A modified genetic algorithm with local search strategies and multi-crossover operator for job shop scheduling problem. *Sensors* **2020**, *20*, 5440. [CrossRef]
75. Ahmadi, A.; El Bouanani, F.; Ben-Azza, H.; Benghabrit, Y. A Novel Decoder Based on Parallel Genetic Algorithms for Linear Block Codes. *Int. J. Commun. Netw. Syst. Sci.* **2013**, *6*, 27468. [CrossRef]
76. Nikol'skii, S.M. *Approximation of Functions of Several Variables and Imbedding Theorems*; Springer Science & Business Media: Berlin, Germany, 2012; Volume 205.
77. Rao, D.K.; Khan, M.G.; Khan, S. Mathematical programming on multivariate calibration estimation in stratified sampling. *Int. J. Math. Comput. Phys. Electr. Comput. Eng.* **2012**, *6*, 58–62.
78. Avriel, M. *Nonlinear Programming: Analysis and Methods*; Courier Corporation: North Chelmsford, MA, USA, 2003.
79. Bertsekas, D.P. *Constrained Optimization and Lagrange Multiplier Methods*; Academic Press: Cambridge, MA, USA, 2014.
80. Lin, Z.; Chen, M.; Ma, Y. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv* **2010**, arXiv:1009.5055.
81. Liang, S.; Zeng, X.; Hong, Y. Distributed nonsmooth optimization with coupled inequality constraints via modified Lagrangian function. *IEEE Trans. Autom. Control* **2017**, *63*, 1753–1759. [CrossRef]
82. Gerstmayr, J.; Sugiyama, H.; Mikkola, A. Review on the absolute nodal coordinate formulation for large deformation analysis of multibody systems. *J. Comput. Nonlinear Dyn.* **2013**, *8*, 031016. [CrossRef]
83. Han, W.; Jung, D.W.; Lee, J.; Yu, C. Determination of eigenvectors with Lagrange multipliers. *J. Korean Phys. Soc.* **2021**, *78*, 1018–1022. [CrossRef]
84. Ito, K.; Kunisch, K. *Lagrange Multiplier Approach to Variational Problems and Applications*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2008.
85. Breusch, T.S.; Pagan, A.R. The Lagrange multiplier test and its applications to model specification in econometrics. *Rev. Econ. Stud.* **1980**, *47*, 239–253. [CrossRef]
86. Ehrgott, M. A discussion of scalarization techniques for multiple objective integer programming. *Ann. Oper. Res.* **2006**, *147*, 343–360. [CrossRef]
87. Bazaraa, M.S.; Sherali, H.D.; Shetty, C.M. Lagrangian duality and saddle point optimality conditions. *Nonlinear Program Theory Algorithms* **2013**, 199–242.
88. Mahmudov, E.N. *Approximation and Optimization of Discrete and Differential Inclusions*; Elsevier: Amsterdam, The Netherlands, 2011.
89. Arora, K.; Kumar, A.; Kamboj, V.K.; Prashar, D.; Jha, S.; Shrestha, B.; Joshi, G.P. Optimization methodologies and testing on standard benchmark functions of load frequency control for interconnected multi area power system in smart grids. *Mathematics* **2020**, *8*, 980. [CrossRef]
90. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H.; Mirjalili, S. Binary Approaches of Quantum-Based Avian Navigation Optimizer to Select Effective Features from High-Dimensional Medical Data. *Mathematics* **2022**, *10*, 2770. [CrossRef]

91. Vanneschi, L.; Castelli, M.; Silva, S. A survey of semantic methods in genetic programming. *Genet. Program. Evolvable Mach.* **2014**, *15*, 195–214. [CrossRef]
92. Picek, S.; Golub, M.; Jakobovic, D. Evaluation of Crossover Operator Performance in Genetic Algorithms with Binary Representation. In *International Conference on Intelligent Computing*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 223–230.
93. Hakimi, D.; Oyewola, D.O.; Yahaya, Y.; Bolarin, G. Comparative analysis of genetic crossover operators in knapsack problem. *J. Appl. Sci. Environ. Manag.* **2016**, *20*, 593–596. [CrossRef]
94. Malik, S.; Wadhwa, S. Preventing premature convergence in genetic algorithm using DGCA and elitist technique. *Int. J. Adv. Res. Comput. Sci. Softw. Eng.* **2014**, *4*, 410–418.
95. Abdullah, J.M.; Ahmed, T. Fitness dependent optimizer: Inspired by the bee swarming reproductive process. *IEEE Access* **2019**, *7*, 43473–43486. [CrossRef]
96. Jamil, M.; Yang, X.S. A literature survey of benchmark functions for global optimisation problems. *Int. J. Math. Model. Numer. Optim.* **2013**, *4*, 150–194. [CrossRef]
97. Hassan, B.A.; Rashid, T.A. A multidisciplinary ensemble algorithm for clustering heterogeneous datasets. *Neural Comput. Appl.* **2021**, *33*, 10987–11010. [CrossRef]
98. Lim, R.; Zhou, L.; Gupta, A.; Ong, Y.S.; Zhang, A.N. Solution representation learning in multi-objective transfer evolutionary optimization. *IEEE Access* **2021**, *9*, 41844–41860. [CrossRef]
99. Hong, H.; Ye, K.; Jiang, M.; Cao, D.; Tan, K.C. Solving large-scale multiobjective optimization via the probabilistic prediction model. *Memetic Comput.* **2022**, *14*, 165–177. [CrossRef]
100. Ting, C.K.; Su, C.H.; Lee, C.N. Multi-parent extension of partially mapped crossover for combinatorial optimization problems. *Expert Syst. Appl.* **2010**, *37*, 1879–1886. [CrossRef]
101. Hama Rashid, D.N.; Rashid, T.A.; Mirjalili, S. ANA: Ant Nesting Algorithm for Optimizing Real-World Problems. *Mathematics* **2021**, *9*, 3111. [CrossRef]
102. Abdulhameed, S.; Rashid, T.A. Child drawing development optimization algorithm based on child's cognitive development. *Arab. J. Sci.* **2022**, *47*, 1337–1351. [CrossRef]
103. Braik, M.; Sheta, A.; Al-Hiary, H. A novel meta-heuristic search algorithm for solving optimization problems: Capuchin search algorithm. *Neural Comput. Appl.* **2021**, *33*, 2515–2547. [CrossRef]
104. Papadrakakis, M.; Lagaros, N.D.; Tsompanakis, Y.; Plevris, V. Large scale structural optimization: Computational methods and optimization algorithms. *Arch. Comput. Methods Eng.* **2001**, *8*, 239–301. [CrossRef]
105. Feng, Y.; Deb, S.; Wang, G.G.; Alavi, A.H. Monarch butterfly optimization: A comprehensive review. *Expert Syst. Appl.* **2021**, *168*, 114418. [CrossRef]
106. Wang, Z.; Zheng, L.; Li, H. Distributed optimization over general directed networks with random sleep scheme. *Int. J. Control Autom. Syst.* **2020**, *18*, 2534–2542. [CrossRef]
107. Hasan, D.O.; Aladdin, A.M.; Talabani, H.S.; Rashid, T.A.; Mirjalili, S. The Fifteen Puzzle—A New Approach through Hybridizing Three Heuristics Methods. *Computers* **2023**, *12*, 11. [CrossRef]