RESOLUTION OF NONLINEAR OPTIMIZATION PROBLEMS SUBJECT TO BIPOLAR MAX-MIN FUZZY RELATION EQUATION CONSTRAINTS USING GENETIC ALGORITHM

H. DANA MAZRAEH AND A. ABBASI MOLAI

Abstract. This paper studies the nonlinear optimization problems subject to bipolar max-min fuzzy relation equation constraints. The feasible solution set of the problems is non-convex, in a general case. Therefore, conventional nonlinear optimization methods cannot be ideal for resolution of such problems. Hence, a Genetic Algorithm (GA) is proposed to find their optimal solution. This algorithm uses the structure of the feasible domain of the problems and lower and upper bound of the feasible solution set to choose the initial population. The GA employs two different crossover operations: 1- N-points crossover and 2- Arithmetic crossover. We run the GA with two crossover operations for some test problems and compare their results and performance to each other. Also, their results are compared with the results of other authors' works.

1. **Introduction**

Fuzzy Relation Equations (FREs) and their associated problems have been investigated by many researchers from two viewpoints of theory and application. The system of FREs has firstly been studied by Sanchez in 1976 [27]. They have many applications in different areas such as fuzzy decision-making, fuzzy optimization, medical diagnosis, chemical engineering, image compression and reconstruction, and et cetera [25, 26, 30].
The consistency of the system can be verified in polynomial time and it is well-known that its solution set can be determined by a maximum solution and a finite number of minimal solutions. Various methods have been proposed to solve the system. An algebraic approach was designed to find all its minimal solutions in [15]. The matrix pattern was used to compute the minimal solutions [20]. The algebraic approach was improved by a universal algorithm in [24]. Also, an iterative algorithm and an algorithm based on the concept of a fuzzy determinant were presented in [1, 2], respectively. Recently, Lichun and Boxing's approach was improved by Yeh [35]. A comprehensive review of the resolution methods of FREs has been presented in [3] and references therein.
The linear optimization problem provided to the max-min FRE system was firstly proposed by Fang and Li [4]. They decomposed the problem into two subproblems and employed the branch-and -bound method for its resolution. Some researchers

Received: December 2016; Revised: March 2017; Accepted: May 2017
Key words and phrases: Bipolar fuzzy relation equations, Max-min composition, Nonlinear optimization, Genetic algorithm.

then improved their methods by providing an upper bound for its optimal objective function value and some necessary conditions [34, 33]. They then proposed some simplification procedures to reduce the computation based on the facts. The problem with different compositions has been studied in [12, 16, 17, 29, 31, 32, 36]. The nonlinear optimization problem provided to FRE constraints has been studied by some researchers[19, 18, 13, 7]. Since the feasible domain of the problem is non-convex, in a general case, we cannot use the conventional approaches to solve the problems. Lu and Fang [19] firstly studied the nonlinear optimization problem subject to the max- min FRE constraints. They proposed a Genetic Algorithm (GA) for its resolution using the structure of the solution set of FREs. The individuals of the initial population are chosen from its feasible domain and are kept within it during the mutation and crossover operations. Khorram and Hassanzade [13] studied the nonlinear optimization problem with the max-average composition operator. They presented a modified GA to solve it and some of its components were changed for its resolution. Also, the problem subject to the max-product FRE constraints was investigated by Hassanzade et al [7]. They designed a GA to find an approximate optimal solution for convex or non-convex solution set and evaluated its performance by some test problems.
The applied FREs in the optimization problems are increasing in each of the variables. In some applications, the variables should have bipolar characters. For example, we can see the roles of these variables to formulate the application problem in the product public awareness in revenue management [6]. The used operator is the max-min composition operator. Ferson et al. [6] firstly considered the system of bipolar max-min FREs. They investigated the solution set of each of its equations. With regard to this point, the solution set of the system was determined with the max-min composition. This set can completely be determined by a finite set of maximal and minimal solution pairs. The linear optimization problem subject to the system was then investigated. An algorithm was designed to find its optimal solutions based on the structure of its feasible domain. Checking the consistency of the system of bipolar max-min FREs is NP-complete [14]. Hence, finding the optimal solutions of the linear programming problem with Bipolar Fuzzy Relation Equation (BFRE) constraints will be NP-hard. This problem with max-Lukasiewiz t-norm was studied by Li and Liu [14]. They solved the problem using the integer programming techniques. Up to now, this optimization problem provided to BFREs has only been studied in [6, 14] with the linear objective function. On the other hand, we cannot formulate all of the real-world problems by linear objective functions. Hence, it is necessary to study the nonlinear optimization problems with the max-min BFRE constraints. Up to now, these kinds of problems have not been studied and the proposed GAs in [7, 13, 19] can't be applied to solve the nonlinear optimization problems subject to the max-min bipolar FRE constraints. Therefore, we are motivated to study the nonlinear optimization problems provided to the max-min BFRE constraints. The feasible domain of the proposed problem in this paper is completely different to the problems given in [7, 13, 19]. The second motivation is to design a new genetic algorithm to solve the problems. First of

all, we investigate to the structure of its feasible domain with the max-min composition. Its consistency is briefly considered. The lower and upper bound of its feasible solution set are then determined. We use this point to generate the initial population. Some properties of its feasible domain are investigated. The study of difference between the given GAs in [7, 13, 19] and the proposed algorithm is the third motivation. The proposed GA applies two different crossovers: N-points crossover and Arithmetic crossover. The GA with two crossovers are run for some test problems and compared their results and performance to each other.
The proposed algorithm can also be applied to solve the presented problem in [7, 13, 19] since the proposed problem in this paper is a generalization of the given problems in [7, 13, 19]. The comparison between the proposed GAs with two crossovers in this paper is the forth motivation. The comparison between the GAs in [7, 13, 19] and the proposed GA in this paper is the fifth motivation. This proposed algorithm can be applied for resolution of any constrained optimization problem with fuzzy relation equations or inequalities or bipolar fuzzy relation equations or inequalities. Since the inequality constraints can be converted to equality constraints [6], the proposed GA in this paper can also be applied to solve the nonlinear optimization problems subject to bipolar fuzzy relation inequality constraints. The given GAs in [7, 13, 19] can only be applied to solve the nonlinear optimization problems subject to fuzzy relation equation constraints not fuzzy relation inequality. Moreover, the fuzzy relation equation or inequality is special cases of bipolar fuzzy relation equation or inequality. Therefore, the proposed GA in this paper is more general from the presented GAs in [7, 13, 19]. It can be applied to solve the given problems in [7, 13, 19]. But the proposed algorithm in [7, 13, 19] cannot be applied to solve the proposed problem in this paper. Therefore, the proposed GA in this paper is more general from the presented GA in [7, 13, 19]. The studied problem in [18] is completely deferent from the proposed problem in this paper. The problem in [18] is a multi-objective optimization problem subject to fuzzy relation equation constraints that intend to find the Pareto optimal solutions but in this paper, we will find the optimal solutions. The aims of two problems are completely deferent.

The structure of this paper is as follows. Section 2 is divided to two subsections. The formulation of the problem is introduced in the first subsection and the structure of feasible domain is studied in the second subsection. The algorithm for initializing with a population and the fitness function are designed in Section 3. The steps of the proposed GA are presented in Section 4. Some test problems are given to show its performance. Also, we compare the GA with two crossover operations to each other. Their results are compared to the real optimal solution of the test problems in Section 5. The analysis of the results is discussed in Section 6. The results are compared to the results of the other authors' work in Section 6. Conclusions are presented in Section 7.

2. The Nonlinear Optimization Problem with BFRE Constraints

This section is divided to two subsections. The first subsection introduces the optimization problem and the second subsection investigates the structure of its feasible domain.

2.1. **Formulation of the Problem.**

Let $A^+ = (a_{ij}^+)$ and $A^- = (a_{ij}^-)$ are two $m \times n$ fuzzy relation matrices with $a_{ij}^+, a_{ij}^- \in [0,1]$. Also, assume that $b = (b_1, \ldots, b_m)^T \in [0,1]^m$. Then the nonlinear optimization problem subject to the constraints of bipolar max-min fuzzy relation equations is formulated as follows:

$$\begin{aligned} &Min(or\ Max)\ f(x),\\ &s.t.\ \ A^+ \circ x \vee A^- \circ \neg x = b,\\ &x \in [0,1]^n, \end{aligned} \tag{1}$$

where the function of $f : R^n \to R$ is a nonlinear function. The notation of "$\circ$" is the max-min composition operator. Moreover, the vector of $x = (x_1, \ldots, x_n)^T \in [0,1]^n$ is the vector of decision variables to be determined and $\neg x = (1 - x_1, \ldots, 1 - x_n)^T$. The notation of "$\bigvee$" is the maximum operation. With regard to the above notations, the part of constraints of problem (1) is a system of the bipolar max-min FREs as follows:

$$A^+ \circ x \vee A^- \circ \neg x = b,\ where\ x \in [0,1]^n, \tag{2}$$

which its solutions are vectors as $x = (x_1, \ldots, x_n)^T$, $x \in [0,1]^n$, such that

$$\max_{j \in J}\ max\ \{min(a_{ij}^+, x_j), min(a_{ij}^-, (1 - x_j))\} = b_i,\ \forall i \in I, \tag{3}$$

where the index sets I and J are as $I = \{1, 2, ..., m\}$ and $J = \{1, 2, ..., n\}$, respectively.
A system of bipolar max-min FREs is called consistent if its solution set, i.e., $X(A^+, A^-, b)$, is not empty. Otherwise, it is inconsistent. We now investigate the structure of the solution set of system (2).

2.2. **The Structure of the Solution Set of System (2).**

In this subsection, we express some important results about system (2) of [6]. Some conditions are firstly presented to be nonempty of the solution set of system (2). Then, the lower and upper bound of the solution set of system (2) are reviewed in terms of some lemmas.

Lemma 2.1. [6] *Consider the i^{th} equality constraint of (3). Then a necessary and sufficient condition for this equation to have a solution is as follows:*

$$b_i \in \left[\max_{j=1,\ldots,n} min\left(a_{ij}^+, a_{ij}^-, \frac{1}{2}\right), \max_{j=1,\ldots,n} \max\left(a_{ij}^+, a_{ij}^-\right)\right]. \tag{4}$$

We will now consider the general case of system (3). In this case, the feasible domain is confronted with the existence of holes in its solution set. A necessary condition for existence of solution is presented in the following lemma.

Lemma 2.2. [6] *A necessary and sufficient condition for existence of solution of system (3) is as follows:*

$$\forall i \in I \ b_i \in \left[\max_{j=1,\dots,n} min \left(a_{ij}^{+}, a_{ij}^{-}, \frac{1}{2} \right), \max_{j=1,\dots,n} \max \left(a_{ij}^{+}, a_{ij}^{-} \right) \right]. \tag{5}$$

We remind some notations to determine the solution set of system (3) or (2) from [6] as follows.

Remark 2.3. Let vector $g^{i+} = \left(\left(g^{i+}\right)_1, \dots, \left(g^{i+}\right)_n \right)^T$ where its components are computed using the following relation:

$$\left(g^{i+}\right)_j = \begin{cases} 1 & if\, a_{ij}^{+} \leq b_i, \\ b_i & if\, a_{ij}^{+} > b_i, \end{cases} \quad j = 1, \dots, n \ .$$

Also, let $S^{i+} = \left\{ s_k^{i+} = ((s_k^{i+})_1, ..., (s_k^{i+})_n) \left| a_{ik}^{+} \geq b_i \right. \right\}$ where the components of vector s_k^{i+} are computed using the following relation:

$$(s_k^{i+})_j = b_i \delta_{kj}, j = 1, ..., n,$$

where δ_{kj} is Kronecker's delta.

Remark 2.4. Let vector $g^{i*} = \left(\left(g^{i*}\right)_1, \dots, \left(g^{i*}\right)_n \right)^T$ where its components are computed using the following relation:

$$\left(g^{i*}\right)_j = \begin{cases} 1 & if\, a_{ij}^{-} \leq b_i, \\ b_i & if\, a_{ij}^{-} > b_i, \end{cases} \quad j = 1, \dots, n \ ,$$

Now, we introduce the vector of s_k^{i-} based on the vector of g^{i*} as follows:

$$s^{i-} = \overline{g^{i*}} = (1 - (g^{i*})_1, ..., 1 - (g^{i*})_n).$$

Also, let $S^{i*} = \left\{ s_k^{i*} = ((s_k^{i*})_1, ..., (s_k^{i*})_n) \left| a_{ik}^{-} \geq b_i \right. \right\}$ where the components of vector s_k^{i*} are computed using the following relation:

$$(s_k^{i*})_j = b_i \delta_{kj}, j = 1, ..., n.$$

Moreover, let $G^{i-} = \{ g \mid \exists s \in S^{i*}, \ g = \overline{s} \}$.

With regard to the above remarks, the solution set of system (3) is determined as follows.

Lemma 2.5. [6] *The solution set of system (3) is determined by the following relation:*

$$D = \bigcap_{i=1}^{m} [s^{i-}, g^{i+}] \cap \left(\left(\bigcup_{s \in S^{i+}} [s \vee s^{i-}, g^{i+}] \right) \cup \left(\bigcup_{g \in G^{i-}} [s^{i-}, g \wedge g^{i+}] \right) \right),$$

where notation $\wedge$ denotes the minimum operation.

The following corollary is a direct result of Lemma 2.5.

Corollary 2.6. [6] *Let $s^{-} = \sup\limits_{i=1,\dots,m} s^{i-}$ and $g^{+} = \inf\limits_{i=1,\dots,m} g^{i+}$. Then vectors s^{-} and g^{+} are lower and upper bound of the solution set of system 3, respectively.*

With regard to Corollary 2.6, we can obtain a lower and upper bound for the optimal value of variable x_i. The lower and upper bound are as $LB_i = s_i^-$ and $UB_i = g_i^+$, respectively, where s_i^- and g_i^+ are the ith components of the given vectors of s^- and g^+ in Corollary 2.6, respectively. We are now ready to present the proposed genetic algorithm.

3. A proposed GA for Nonlinear Optimization Problems Subject to the Bipolar Max-Min FRE Constraints

The GA has been developed by Holland in 1975 [9]. The GA is a heuristic search that mimics the process of the natural evolution of genetics. GAs are routinely applied to find the solutions of the optimization problems. These algorithms usually start from a population of randomly generated individuals and improve solutions using two genetic operators, called crossover and mutation operators. In GAs, the solutions for a problem are denoted as individuals. The individuals can be represented in terms of the genetics structure of chromosomes. The offsprings are generated with regard to the process of selection and the operators of crossover and mutation. In each generation, individuals (solutions) are selected by a fitness-based process and some selection criteria. The fitter solutions are typically more likely to be selected. Certain selection approaches measure their fitness and preferentially select the best solutions. Some approaches were designed to solve the constrained optimization problems using GAs. To do this, these methods used the penalty or barrier functions [5, 10, 11, 28]. Several genetic operators were introduced to solve the optimization problems with convex domain [21, 22, 23]. In this paper, we design a Genetic Algorithm for Optimization problems with Bipolar Fuzzy Relation Constraints (GAOBFRC).
The proposed GAOBFRC is specially designed for resolution of nonlinear optimization problem with bipolar fuzzy relation constraints. Its feasible domain is non-convex in a general case and the structure of its feasible domain is completely different to the structure of the feasible domain in [19, 18, 13, 7]. Also, the proposed GAOBFRC uses real-value representation for individuals. We illustrate the details of each step of the algorithm below.

3.1. **Initialization.** In a genetic algorithm, the evaluation usually starts from a population of randomly generated individuals, and is an iterative process. The initial population is randomly generated in the search space. The population in each iteration is called a generation. This procedure works for the unconstrained optimization problems well. In the constrained optimization problems, the generations may not be in the feasible domain. Since the proposed GA want to keep the solutions in the feasible domain, an initialization plan is presented to initialize a population by randomly generating the individuals inside the feasible domain. To do this, we use Corollary 2.6. The number of solutions of the feasible domain is usually infinite. Therefore, the solution set can be found and some solution is randomly picked from it. The algorithm for initializing with a population is designed below.

```
Initialization:
Let $i := 1$
While $i \leq$ population-size do
Generate randomly $x = (x_1, ..., x_n)$, $x_i \in [LB_i, UB_i]$, as a chromosome
If $x$ is feasible then
population(i).chromosome=$x$
Let i:=i+1
Endif
Endwhile
```

We now illustrate the fitness function in the next subsection.

3.2. **Fitness Function.** One of the most important components of heuristic algorithms is the definition of fitness function. In this study, we use the objective function value to define the fitness function. In the maximization problem, the fitness function value is proportional to the objective function value. Also, in the minimization problems, it is proportional to the inverse of objective function value. The fitness function is defined as follows.

```
Fitness-function(vector $x$ as a chromosome)
If problem is maximization then
Let $t1 := f(x)$
else
Let $t1 := \frac{1}{f(x)}$
Endif
Let $t2 := feasible(x)$  "To check the feasibility of vector $x$"
Return($t1 - (1 - t2) * penalty\_value + Constant\_value$)
```

In the above procedure, we added the penalty value to the statement of $(t1 - (1 - t2) * penalty\_value)$ to shift this statement by a constant value which is equal to the penalty value. We do this work because the returned value of procedure becomes positive. The variable of $t1$ denotes the objective function value or the inverse of objective function value. The variable of $t2$ is a boolean variable. If the chromosome is feasible, then its value is one. Otherwise, its value is zero. We are now ready to present a real-value genetic algorithm to solve the nonlinear optimization problem subject to the system of bipolar max-min fuzzy relation equations.

4. A Real-value Genetic Algorithm for Resolution of Problem (1)

The steps of the proposed algorithm are as follows.

Step 1: Initialization.
Step 2: Evaluate each chromosome.
Step 3: Parent selection.
Step 4: Recombination.
Step 5: Mutation.
Step 6: Evaluate offspring.
Step 7: Survivor.
Step 8: Repeat Step 3 to Step 7, until predefined number of generations is accomplished.

We now illustrate the above procedures in Steps 2-7, respectively.

4.1. **Evaluate Each Chromosome.** Let $i := 1$

```
While $i \leq$ population-size do
Let population(i).fitness:=fitness-function(population(i).chromosome)
let $i := i + 1$
Endwhile
```

4.2. **Parent Selection.** Roullete wheel:

```
Let i:=1
While $i \leq 2$ do
Pick a random value $r$ uniformly from [0,Sum of All Fitnesses]
Let $j := 1$
Let a:=population(j).fitness
While $a < r$ do
Let j:=j+1
Let a:=a+population(j).fitness
Endwhile
Let Parent(i).chromosome:=population(j).chromosome
Let i:=i+1
Endwhile
```

4.3. **Recombination.** In this part, we apply two kinds of crossover: a. N-points crossover and b. Arithmetic crossover.

a. N-points crossover

```
Pick two random integer values $n1$ and $n2$ from [1,n]
If $n1 > n2$ then
Exchange($n1$,$n2$)
Endif
For $i := 1$ to $n1$ do
Let offspring.chromosome(i):=Parent(1).chromosome(i)
Endfor
For $i := n1 + 1$ to $n2$ do
Let offspring.chromosome(i):=Parent(2).chromosome(i)
Endfor
For $i := n2 + 1$ to $n$ do
Let offspring.chromosome(i):=Parent(1).chromosome(i)
Endfor
```

b. Whole Arithmetic crossover

```
For $i = 1$ to $n$ do
Pick a random value $r$ uniformly from [0,1]
Let offspring.choromosome(i):=r*Parent(1).chromosome(i)+(1-r)*Parent(2). chromosome(i)
Endfor
```

4.4. **Mutation.** Pick a random small value r uniformly from $[-\alpha, \alpha]$, where α is a positive small value.

Pick an integer value i from $[1, n]$
Let offspring.chromosome(i):=offspring.chromosome(i)+r

4.5. **Evaluate Offspring.** Let offspring.fitness:=fitness-function(offspring. chromosome)

4.6. **Survivor.** Replace offspring with the first individual in population with lower fitness.

4.7. **Termination Condition.** It will continue until the predefined number of generations is accomplished. The predefined number of generations is given by a decision-maker. Those are often determined as $100, 200, 300, 400, 500, 600, 700, 800, 900, 1000$ in the GA. The number of generations has been determined in Tables 1-10. We are ready to present some test problems to show the performance of the proposed GA with two crossovers for resolution of Problem (1).

5. Test Problems and Numerical Results

We present two test problems from [6, 19] and three test problems that three their objective functions have been taken from Hock and Schittkowski's book [8]. In this part, the test problems have been solved by the proposed GA with two crossovers: a. N-points crossover and b. Arithmetic crossover. The results show their performance and time.

Example 5.1. [6] Consider the problem (1), where

$$Max\ f(x) = 2x_1 + 6x_2,\ A^+ = \begin{pmatrix} 0.30 & 0.60 \\ 0.90 & 0.60 \end{pmatrix},\ A^- = \begin{pmatrix} 0.70 & 0.70 \\ 0.50 & 0.30 \end{pmatrix}, and$$

$$b = (0.70, 0.60)^T.$$

In Tables 1 and 2, the solutions of Example 5.1 and their times in each generation have been presented using the proposed GA with two crossovers:N-points and Arithmetic.

Generations	x_1	x_2	$f(x)$	Execution time of each generation (Second)
100	0.2543	0.9400	6.1488	0.021822
200	0.2785	0.9950	6.5271	0.075823
300	0.2968	0.9976	6.5773	0.058363
400	0.2972	0.9848	6.5030	0.079144
500	0.2967	0.9998	6.5923	0.136101
600	0.2983	0.9962	6.5735	0.138805
700	0.2953	0.9975	6.5761	0.168818
800	0.2932	0.9962	6.5637	0.165860
900	0.2903	0.9994	6.5768	0.172156
1000	0.2999	0.9981	6.5884	0.176013

Table 1. The Results of the Proposed GA with N-points Crossover on Example 5.1

Generations	x_1	x_2	$f(x)$	Execution time of each generation(Second)
100	0.2152	0.9755	6.2833	0.031202
200	0.2930	0.9948	6.5547	0.061252
300	0.2906	0.9984	6.5716	0.054502
400	0.2645	0.9911	6.4757	0.102274
500	0.2890	0.9956	6.5516	0.092436
600	0.2656	0.9952	6.5026	0.104286
700	0.2916	0.9944	6.5497	0.114739
800	0.2950	0.9882	6.5194	0.130705
900	0.2911	0.9941	6.5470	0.151142
1000	0.2993	0.9974	6.5827	0.165587

Table 2. The Results of the Proposed GA with Arithmetic Crossover on Example 5.1

Figure 1 shows that the feasible domain of the problem in Example 5.1 is non-convex and we cannot apply the resolution methods of convex programming problems to solve these problems. Since non-convex programming problems are NP-hard problem, exact resolution methods for these problems have a high computational complexity. Therefore, we apply GA with N-points crossover and Arithmetic crossover to solve the problems. Also, Figure 2 shows the performance of the proposed GA with two crossovers in Example 5.1.

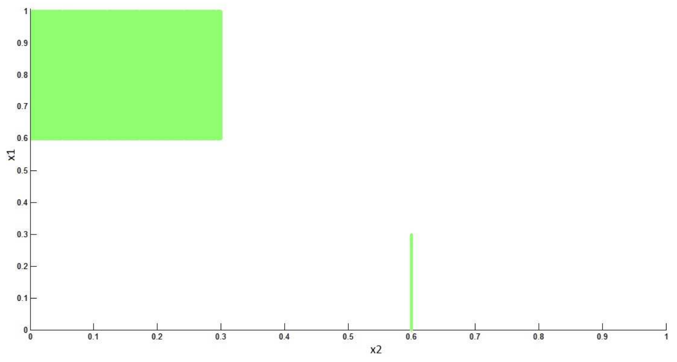

Figure 1. The Green Region Displays the Feasible Solution Set for Example 5.1

Example 5.2. [8] Consider the Problem of (1), where

$$Min\ f(x) = 3000x_1 + 1000x_1^3 + 2000x_2 + 666.667x_2^3,\ A^+ = \begin{pmatrix} 0.44 & 0.24 & 0.44 \\ 0.22 & 0.98 & 0.46 \\ 0.55 & 0.21 & 0.42 \end{pmatrix},$$

$$A^- = \begin{pmatrix} 0.22 & 0.05 & 0.44 \\ 0.22 & 0.44 & 0.66 \\ 0.11 & 0.64 & 0.20 \end{pmatrix}, and \quad b = (0.44, 0.66, 0.5)^T.$$

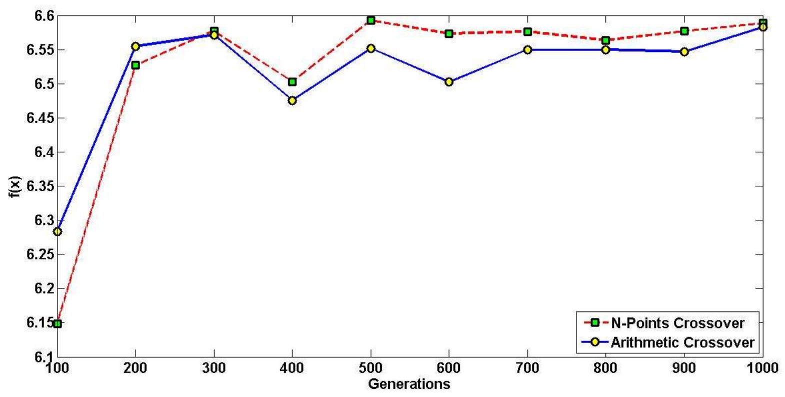

FIGURE 2. The Performance of the Proposed GA for Example 5.1

To accelerate for finding the initialization population, we use Corollary 2.6 and compute the lower and upper bound of each $x_i,\ for\ i = 1, 2, 3$, as follows: $g^{1+} = (g_1^{1+}, g_2^{1+}, g_3^{1+})^T = (1, 1, 1)^T,\ g^{2+} = (g_1^{2+}, g_2^{2+}, g_3^{2+})^T = (1, 0.66, 1)^T,\ and\ g^{3+} = (g_1^{3+}, g_2^{3+}, g_3^{3+})^T$
$= (0.5, 1, 1)^T$. Now, we compute the upper bound using $g^{i+},\ for\ i = 1, 2, 3$, as follows: $g^+ = \inf\limits_{i=1,\ldots,3} g^{i+} = (0.5, 0.66, 1)^T$. Similarly, using Corollary 2.6, we have: $s^{1-} = (0, 0, 0)^T,\ \ s^{2-} = (0, 0, 0)^T,\ and\ s^{3-} = (0, 0.5, 0)^T$. Now, we compute the lower bound using $s^{i-},\ for\ \ i = 1, 2, 3$, as follows: $s^- = \sup\limits_{i=1,\ldots,3} s^{i-} = (0, 0.5, 0)^T$. With regard to the values of vectors s^- and g^+, the lower and upper bound of each $x_i,\ for\ i = 1, 2, 3$, are as follows: $x_1 \in [0, 0.5]$, $x_2 \in [0.5, 0.66]$, and $x_3 \in [0, 1]$. With regard to these values, we can find the initialization population more quickly. In Tables 3 and 4, the solutions of Example 5.2 and their times in each generation have been presented using the proposed GA with two crossovers:N-points and Arithmetic.

Generations	x_1	x_2	x_3	$f(x)$	Execution time of each generation(Second)
100	0.002	0.500	0.160	1089.333383	12.66
200	0.002	0.500	0.187	1089.333383	16.177
300	0.001	0.500	0.178	1086.010091	19.858
400	0.001	0.500	0.057	1084.839131	17.076
500	0.000	0.500	0.288	1084.013869	19.814
600	0.000	0.500	0.324	1084.183784	20.496
700	0.000	0.500	0.125	1083.333375	27.212
800	0.000	0.500	0.113	1083.333375	25.782
900	0.000	0.500	0.027	1083.333375	26.859
1000	0.000	0.500	0.223	1083.333375	27.956

TABLE 3. The Results of the Proposed GA with N-points Crossover on Example 5.2

Generations	x_1	x_2	x_3	$f(x)$	Execution time of each generation(Second)
100	0.004	0.500	0.096	1094.585432	9.627
200	0.002	0.500	0.197	1087.862361	20.127
300	0.001	0.500	0.201	1086.333376	17.400
400	0.001	0.500	0.094	1086.333376	16.316
500	0.001	0.500	0.024	1086.333376	23.548
600	0.000	0.500	0.250	1083.838205	27.390
700	0.000	0.500	0.085	1083.976601	26.690
800	0.000	0.500	0.040	1083.963706	29.714
900	0.000	0.500	0.202	1083.493867	28.610
1000	0.000	0.500	0.277	1083.333375	27.43

Table 4. The Results of Proposed GA with Arithmetic Crossover Example 5.2

Figure 3 shows that the feasible domain of the problem in Example 5.2 is non-convex and we cannot apply the resolution methods of convex programming problems to solve these problems. Since non-convex programming problems are NP-hard problem, exact resolution methods for these problems have a high computational complexity. Therefore, we apply GA with N-points crossover and Arithmetic crossover to solve the problems. Also, Figure 4 shows the performance of the proposed GA with two crossovers in Example 5.2.

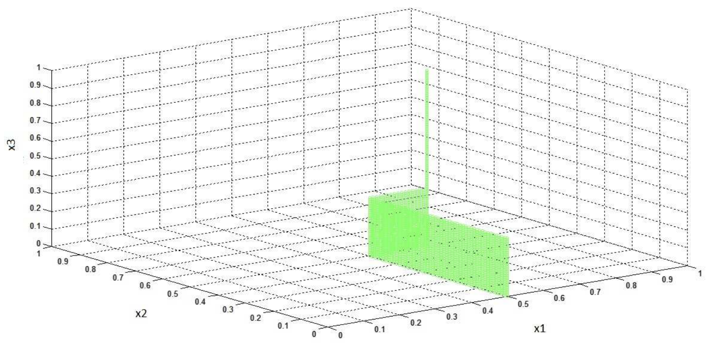

Figure 3. The Green Region Displays the Feasible Solution Set for Example 5.2

Example 5.3. [8] Consider the problem (1), where

$$Min\ f(x) = x_1x_2x_3x_4x_5,\ A^+ = \begin{pmatrix} 0.33 & 0.41 & 0.44 & 0.56 & 0.43 \\ 0.43 & 0.37 & 0.5 & 0.56 & 0.13 \\ 0.67 & 0.99 & 0.78 & 0.41 & 0.74 \\ 0.81 & 0.57 & 0.37 & 0.01 & 0.77 \end{pmatrix},$$

$$A^- = \begin{pmatrix} 0.32 & 0.5 & 0.43 & 0.64 & 0.12 \\ 0.24 & 0.87 & 0.24 & 0.02 & 0.01 \\ 0.56 & 0.13 & 0.89 & 0.91 & 0.08 \\ 0.11 & 0.23 & 0.45 & 0.33 & 0.02 \end{pmatrix}, and \quad b = (0.64, 0.69, 0.88, 0.45)^T.$$

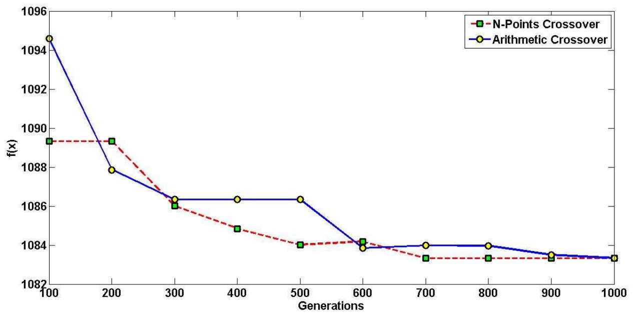

FIGURE 4. The Performance of the Proposed GA for Example 5.2

Similar to Example 5.2, we can use Corollary 2.6 to accelerate for finding the initialization population and compute the lower and upper bound of each $x_i,\ for\ i = 1,2,3,4,5$ as follows:
$g^{1+} = (1,1,1,1,1)^T,\ g^{2+} = (1,1,1,1,1)^T,\ g^{3+} = (1,0.88,1,1,1)^T,\ and\ g^{4+} = (0.45, 0.45,$
$1, 1, 0.45)^T$. Now, we compute the upper bound using $g^{i+},\ for\ i = 1,2,3,4$, as follows: $g^+ = \inf_{i=1,\dots,4} g^{i+} = (0.45, 0.45, 1, 1, 0.45)^T$.

Similarly, using Corollary 2.6, we have: $s^{1-} = (0,0,0,0,0)^T$, $s^{2-} = (0,0.31,0,0,0)^T$, $s^{3-} = (0,0,0.12,0.12,0)^T$, and $s^{4-} = (0,0,0,0,0)^T$. Now, we compute the lower bound using $s^{i-},\ for\ i = 1,2,3,4$, as follows: $s^- = \sup_{i=1,\dots,4} s^{i-} = (0, 0.31, 0.12, 0.12,$
$0)^T$. With regard to the values s^- and g^+, the lower and upper bound of each $x_i,\ for\ i = 1,2,3,4,5$, are as follows: $x_1 \in [0, 0.45]$, $x_2 \in [0.31, 0.45]$, $x_3 \in [0.12, 1]$, $x_4 \in [0.12, 1]$, and $x_5 \in [0, 0.45]$. With regard to these values, we can find the initialization population more quickly.

In Tables 5 and 6, the solutions of Example 5.3 and their times in each generation have been presented using the proposed GA with two crossovers:N-points and Arithmetic.

Generations	x_1	x_2	x_3	x_4	x_5	$f(x)$	Execution time of each generation(Second)
100	0.392	0.310	0.178	0.120	3.4694e-18	9.0055e-21	115.366
200	0.003	0.310	0.124	0.120	0.004	5.5354e-08	183.766
300	0.005	0.310	0.138	0.120	0.008	2.0534e-07	89.558
400	0.001	0.310	0.120	0.121	0.004	1.8005e-08	140.505
500	5.2042e-18	0.310	0.120	0.130	0.098	2.4664e-21	243.023
600	0.110	0.310	0.210	0.120	1.7347e-18	1.4907e-21	210.531
700	0.001	0.310	0.120	0.120	0.004	1.7856e-08	128.585
800	8.6736e-19	0.310	0.120	0.125	0.029	1.1696e-22	236.390
900	1.1276e-17	0.310	0.245	0.120	0.091	9.3518e-21	43.596
1000	0.036	0.310	0.154	0.120	1.2143e-17	2.5043e-21	162.71

TABLE 5. The Results of the Proposed GA with N-points Crossover on Example 5.3

Generations	x_1	x_2	x_3	x_4	x_5	$f(x)$	Execution time of each generation(Second)
100	0.249	0.310	0.120	0.160	0.020	0.000030	71.816
200	0.032	0.310	0.120	0.160	0	0	80.732
300	0.010	0.310	0.289	0.120	0	0	67.409
400	0	0.310	0.120	0.218	0.010	0	116.366
500	0	0.310	0.120	0.127	0.010	0	28.927
600	0.076	0.310	0.120	0.190	0	0	118.311
700	0	0.310	0.249	0.120	0	0	233.396
800	0	0.310	0.135	0.120	0.030	0	76.161
900	0.015	0.310	0.150	0.120	0	0	119.75
1000	0.010	0.310	0.120	0.300	0	0	101.577

TABLE 6. The Results of the Proposed GA with Arithmetic Crossover on Example 5.3

Also, Figure 5 shows the performance of the proposed GA with two crossovers in Example 5.3.

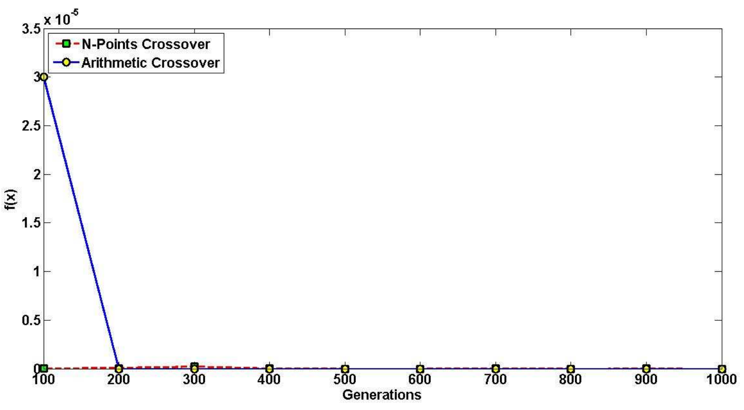

FIGURE 5. The performance of the proposed GA on Example 5.3.

Example 5.4. [8] Consider the problem (1), where

$$Max\ f(x) = e^{x_1+x_3-x_4} + sin(x_2+x_5+x_6),$$

$$A^+ = \begin{pmatrix} 0.15 & 0.75 & 0.25 & 0.12 & 0.28 & 0.37 \\ 0.35 & 0.85 & 0.41 & 0.8 & 0.35 & 0.15 \\ 0.7 & 0.22 & 0.25 & 0.28 & 0.12 & 0.2 \\ 0.9 & 0.51 & 0.85 & 0.4 & 0.2 & 1 \\ 0.1 & 0.11 & 0.21 & 0.25 & 0.8 & 0.65 \end{pmatrix},$$

$$A^- = \begin{pmatrix} 0.2 & 0.23 & 0.51 & 0.3 & 0.7 & 0.15 \\ 0.4 & 0.8 & 0.6 & 0.4 & 0.32 & 0.16 \\ 0.4 & 0.9 & 0.5 & 0.65 & 0.4 & 0.24 \\ 0.14 & 0.18 & 0.65 & 0.54 & 0.38 & 0.6 \\ 0.45 & 0.5 & 0.4 & 0.83 & 0.6 & 0.4 \end{pmatrix},\ and\, b = (0.51, 0.6, 0.65, 0.8, 0.83)^T.$$

Similar to Example 5.2, we can use Corollary 2.6 to accelerate for finding the initialization population and compute the lower and upper bound of each $x_i,\ for\ i = 1, 2, 3, 4, 5, 6$, as follows:
$g^{1+} = (1, 0.51, 1, 1, 1, 1)^T$, $g^{2+} = (1, 0.6, 1, 0.6, 1, 1)^T$, $g^{3+} = (0.65, 1, 1, 1, 1, 1)^T$, $g^{4+} = (0.8, 1, 0.8, 1, 1, 0.8)^T$, and $g^{5+} = (1, 1, 1, 1, 1, 1)^T$. Now, we compute the upper bound using $g^{i+},\ for\ i = 1, 2, 3, 4, 5$, as follows: $g^+ = \inf_{i=1,\ldots,5} g^{i+} = (0.65, 0.51, 0.8, 0.6, 1, 0.8)^T$. Similarly, using Corollary 2.6, we have: $s^{1-} = (0, 0, 0, 0, 0.49, 0)^T$, $s^{2-} = (0, 0.4, 0, 0, 0, 0)^T$, $s^{3-} = (0, 0.35, 0, 0, 0, 0)^T$, $s^{4-} = (0, 0, 0, 0, 0, 0)^T$, *and* $s^{5-} = (0, 0, 0, 0, 0, 0)^T$. Now, we compute the lower bound using $s^{i-},\ for\ i = 1, 2, 3, 4, 5$, as follows: $s^- = \sup_{i=1,\ldots,5} s^{i-} = (0, 0.4, 0, 0, 0.49, 0)^T$. With regard to the values of vectors s^- and g^+, the lower and upper bound of each $x_i,\ for\ i = 1, 2, 3, 4, 5, 6$, are as follows: $x_1 \in [0, 0.65]$, $x_2 \in [0.4, 0.51]$, $x_3 \in [0, 0.8]$, $x_4 \in [0, 0.6]$, $x_5 \in [0.49, 1]$, and $x_6 \in [0, 0.8]$. With regard to these values, we can find the initialization population more quickly.
In Tables 7 and 8, the solutions of Example 5.4 and their times in each generation have been presented using the proposed GA with two crossovers:N-points and Arithmetic.

Generations	x_1	x_2	x_3	x_4	x_5	x_6	$f(x)$	Execution time of each generation(Second)
100	0.638	0.409	0.366	0.017	0.659	0.800	3.638859	45.74
200	0.634	0.430	0.376	0.010	0.517	0.800	3.705168	59.089
300	0.644	0.407	0.392	0.006	0.537	0.800	3.786104	79.782
400	0.647	0.412	0.393	0.004	0.516	0.800	3.805592	61.479
500	0.646	0.412	0.400	0.005	0.501	0.800	3.821954	98.273
600	0.650	0.406	0.396	0.003	0.490	0.800	3.829890	62.017
700	0.649	0.401	0.399	0.002	0.503	0.800	3.837385	104.883
800	0.648	0.400	0.456	0.009	0.521	0.800	3.977923	73.965
900	0.650	0.400	0.474	0.000	0.505	0.800	4.068146	109.587
1000	0.650	0.400	0.489	0.000	0.491	0.800	4.116427	99.77

TABLE 7. The Results of the Proposed GA with N-points Crossover on Example 5.4

Generations	x_1	x_2	x_3	x_4	x_5	x_6	$f(x)$	Execution time of each generation(Second)
100	0.630	0.444	0.381	0.042	0.508	0.800	3.618935	48.745
200	0.650	0.488	0.369	0.014	0.667	0.800	3.659004	83.359
300	0.637	0.408	0.394	0.010	0.584	0.800	3.751208	48.197
400	0.650	0.431	0.378	0.002	0.573	0.800	3.764173	69.119
500	0.642	0.410	0.395	0.003	0.581	0.800	3.788971	51.572
600	0.648	0.400	0.395	0.001	0.504	0.800	3.826580	53.744
700	0.648	0.408	0.399	0.001	0.502	0.800	3.836772	77.391
800	0.650	0.405	0.400	0.002	0.502	0.800	3.842680	59.658
900	0.649	0.400	0.457	0.002	0.504	0.800	4.007348	105.60
1000	0.650	0.400	0.468	0.000	0.492	0.800	4.051394	114.512

TABLE 8. The Results of the Proposed GA with Arithmetic Crossover on Example 5.4

Also, Figure 6 shows the performance of the proposed GA with two crossovers in Example 5.4. Before Example 5.5 is presented, it is necessary to remind a point. The constraints of studied problem in [19] is as $x \circ A = b$ which can be converted as $x^t \circ A^t = b^t$ where notation t is the transpose of matrix.

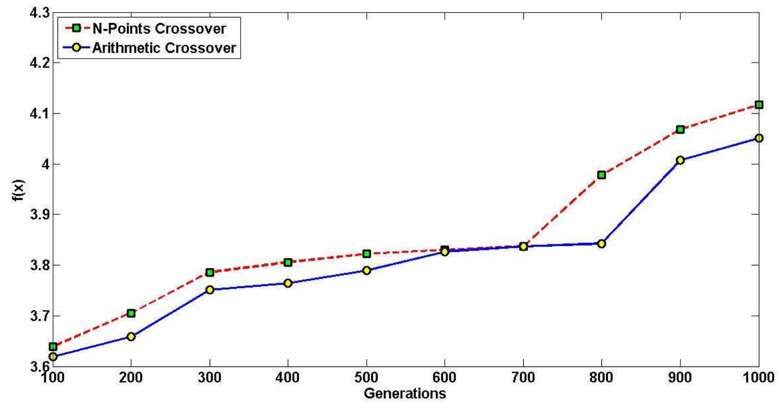

FIGURE 6. The Performance of the Proposed GA on Example 5.4

This FRE can equivalently be rewritten as $A^+ \circ y \vee A^- \circ \neg y = d$ where $A^+ = A^t$, $y = x^t$, $A^- = o$, and $d = b^t$ that notation o is a zero matrix. With regard to this point, we consider the following example.

Example 5.5. [19] Consider the problem (1), where

$$Max\ f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4,$$

$$A^+ = \begin{pmatrix} 0.5176 & 0.2278 & 0.8993 & 0.9858 \\ 0.1370 & 0.4585 & 0.6334 & 0.2790 \\ 0.4093 & 0.7399 & 0.0313 & 0.3039 \end{pmatrix},$$

$$A^- = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},\ and\ b = (0.7208, 0.6334, 0.4725)^T.$$

In Tables 9 and 10, the solutions of Example 5.5 and their times in each generation have been presented using the proposed GA with two crossovers:N-points and Arithmetic.

Generations	x_1	x_2	x_3	x_4	$f(x)$	Execution time of each generation(Second)
100	0.3261	0.4725	0.6732	0.7208	3.211402	8676.91
200	0.2604	0.4725	0.7108	0.7208	2.775089	7560.90
300	0.2604	0.4725	0.6858	0.7208	2.622769	10332.09
400	0.0811	0.4725	0.7046	0.7208	2.592573	8100.63
500	0.1570	0.4725	0.6951	0.7208	2.273279	9360.91
600	0.1883	0.4725	0.6444	0.7208	2.068840	11412.64
700	0.1526	0.4725	0.6443	0.7208	2.035064	10944.11
800	0.1593	0.4725	0.6355	0.7208	2.003492	9504.45
900	0.1583	0.4725	0.6350	0.7208	2.001897	10296.72
1000	0.1673	0.4725	0.6334	0.7208	1.999701	9720.80

TABLE 9. The Results of the Proposed GA with N-points Crossover on Example 5.5

Generations	x_1	x_2	x_3	x_4	$f(x)$	Execution time of each generation(Second)
100	0.3415	0.4725	0.7185	0.7208	3.676057	8316.71
200	0.3341	0.4725	0.6960	0.7208	3.433574	9720 .81
300	0.0372	0.4725	0.6489	0.7208	2.704444	9720.12
400	0.2373	0.4725	0.7022	0.7208	2.559593	8712.77
500	0.0551	0.4725	0.6564	0.7208	2.550977	9828.23
600	0.1204	0.4725	0.7046	0.7208	2.394250	8820.79
700	0.0704	0.4725	0.6461	0.7208	2.379471	11232.54
800	0.2275	0.4725	0.6454	0.7208	2.224468	10584.95
900	0.1550	0.4725	0.6346	0.7208	2.001175	9540.15
1000	0.1581	0.4725	0.6374	0.7208	2.009705	8748.96

Table 10. The Results of the Proposed GA with Arithmetic Crossover on Example 5.5

Also, Figure 7 shows the performance of the proposed GA with two crossovers in Example 5.5.

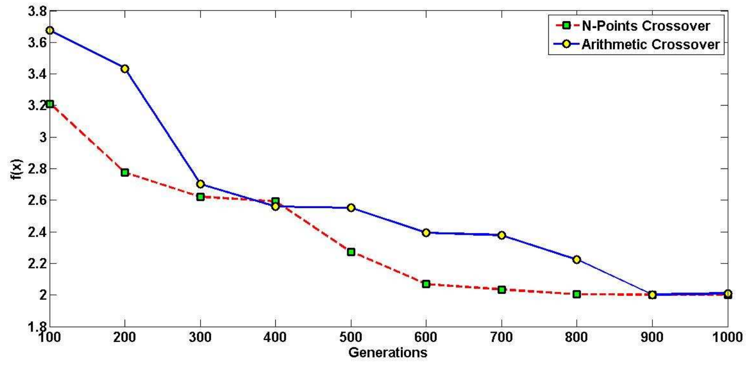

Figure 7. The Performance of the Proposed GA for Example 5.5

6. Discussion About the Results of the Test Problems

In this section, we discuss and analyze the obtained results from implementation of the five test problems.

Example 5.1 is a maximization problem. As Figure 2 and Tables 1 and 2 show the GA with N-points crossover in generations 300, 400, 500, 600, 700, 800, 900, and 1000, have produced values of objective function more than the GA with Arithmetic crossover. Only in generations 100 and 200, the GA with Arithmetic crossover has obtained better results. Table 11 confirms this fact. On the other hand, the results of GA with more generations have more reliability. Overall, it is concluded that the performance of the GA with N-points crossover is better and more exact than the GA with Arithmetic crossover. Moreover, the following table shows that the GA with N-points and Arithmetic crossover converge to the exact solution of the problem. The exact optimal solution of the problem is as:$\tilde{x}^{*(exact)} = (x_1^*, x_2^*) = (0.3, 1)$ with $f(x^*) = 6.6$ [6]. Table 12 shows the difference between the exact optimal objective function value, i.e.,$f^{*(exact)} = f(x^*) = 6.6$ and the approximate optimal objective function value in each generation for the GA with N-points crossover and Arithmetic crossover, respectively.

$Example 5.1$	x_1	x_2	$\tilde{f}^* = f(x)$	$\lvert f^{*(exact)} - \tilde{f}^* \rvert$
Approximate solution resulted from N-points crossover	0.2999	0.9981	6.5884	0.0116
Approximate solution resulted from Arithmetic crossover	0.2993	0.9974	6.5827	0.0173

Table 11. This Table Shows the Results for Generation 1000 and Compares the Results with Exact Optimal Solution on Example 5.1

Generations	$\tilde{f}^{*(NP)}$	$\tilde{f}^{*(Ar)}$	$\lvert \tilde{f}^{*(exact)} - \tilde{f}^{*(NP)} \rvert$	$\lvert \tilde{f}^{*(exact)} - \tilde{f}^{*(Ar)} \rvert$
100	6.1488	6.2833	0.4512	0.3167
200	6.5271	6.5547	0.0729	0.0453
300	6.5773	6.5716	0.0227	0.0284
400	6.5030	6.4757	0.097	0.1243
500	6.5923	6.5516	0.0077	0.0484
600	6.5735	6.5026	0.0265	0.0974
700	6.5761	6.5497	0.0239	0.0503
800	6.5637	6.5194	0.0363	0.0806
900	6.5768	6.5470	0.0232	0.053
1000	6.5884	6.5827	0.0116	0.0173

Table 12. $\tilde{f}^{*(NP)}$ and $\tilde{f}^{*(Ar)}$ Denote the Values of Approximate Optimal Objective Function Obtained by GA with N-points Crossover and Arithmetic Crossover, Respectively, on Example 5.1

Example 5.2 is a minimization problem. As Figure 4 and Tables 3 and 4 show the GA with N-points crossover in generations $100, 300, 400, 500, 700, 800$, and 900 has values of objective function less than the GA with Arithmetic crossover. Only in generation 200 and 600, the GA with Arithmetic crossover has obtained a better result. In generation 1000, the results are the same for the GA with N-points and Arithmetic crossover. In this test problem, the GA with N-points crossover has obtained better results with respect to the GA with Arithmetic crossover. The exact optimal solution of the problem is as: $\tilde{x}^{*(exact)} = (x_1^*, x_2^*, x_3^*) = (0, 0.5, 0.159)$ or $(0, 0.5, 0.007)$ with $\tilde{f}^{*(exact)} = f(x^*) = 1083.333$[6]. Moreover, Table 13 shows that the GA with N-Points and Arithmetic crossover converge to the exact solution of the problem. Tables 14 show the difference between the exact optimal objective

$Example 5.2$	x_1	x_2	x_3	$\tilde{f}^* = f(x)$	$\lvert f^{*(exact)} - \tilde{f}^* \rvert$
Approximate solution resulted from N-points crossover	0	0.500	0.223	1083.333375	0
Approximate solution resulted from Arithmetic crossover	0	0.500	0.277	1083.333375	0

Table 13. This Table Shows the Results for Generation 1000 and Compares the Results with Exact Optimal Solution on Example 5.2

function value, i.e.,$f^{*(exact)} = f(x^*) = 1083.3333$ and the approximate optimal objective function value in each generation for the GA with N-points crossover and Arithmetic crossover, respectively. Example 5.3 is a minimization problem. As Figure 5 and Tables 5 and 6 show the results of GA with N-points crossover and

Generations	$\tilde{f}^{*(NP)}$	$\tilde{f}^{*(Ar)}$	$\lvert\tilde{f}^{*(exact)} - \tilde{f}^{*(NP)}\rvert$	$\lvert\tilde{f}^{*(exact)} - \tilde{f}^{*(Ar)}\rvert$
100	1089.333383	1094.585432	6.000383	11.25243
200	1089.333383	1087.862361	6.000383	4.529361
300	1086.010091	1086.333376	2.677091	3.000376
400	1084.839131	1086.333376	1.506131	3.000376
500	1084.013869	1086.333376	0.680869	3.000376
600	1084.183784	1083.838205	0.850784	0.505205
700	1083.333375	1083.976601	0.000375	0.643601
800	1083.333375	1083.963706	0.000375	0.630706
900	1083.333375	1083.493867	0.000375	0.160867
1000	1083.333375	1083.333375	0.000375	0.000375

TABLE 14. $\tilde{f}^{*(NP)}$ and $\tilde{f}^{*(Ar)}$ Denote the Values of Approximate Optimal Objective Function Obtained by GA with N-points Crossover and Arithmetic Crossover, Respectively, on Example 5.2

Arithmetic crossover are approximately the same. In each generation, the GA with N-points crossover produces the approximate optimal objective value of zero and the GA with Arithmetic crossover produces the optimal objective value of zero except generation 100. In this test problem, the results are the same. The exact optimal objective function value of the problem is as: $x^* = (0.010, 0.310, 0.120, 0.300, 0)$ or $x^* = (0.036, 0.310, 0.154, 0.120, 0)$ with $\tilde{f}^{*(exact)} = f(x^*) = 0$. In this example, the difference between the exact optimal objective function value, i.e., $\tilde{f}^{*(exact)} = f(x^*) = 0$ and the approximate optimal objective function value in each generation for the GA with N-points crossover and Arithmetic crossover are the same approximately. Of course, the GA with Arithmetic crossover is more exact than the GA with N-points crossover, in this example. Table 16 shows the differ-

$Example 5.3$	x_1	x_2	x_3	x_4	x_5	$\tilde{f}^* = f(x)$	$\lvert f^{*(exact)} - \tilde{f}^*\rvert$
Approximate solution resulted from N-points crossover	0.036	0.310	0.154	0.120	1.2143e-17	2.5043e-21	2.5043e-21
Approximate solution resulted from Arithmetic crossover	0.010	0.310	0.120	0.300	0	0	0

TABLE 15. This Table Shows the Results for Generation 1000 and Compares the Results with Exact Optimal Solution on Example 5.3

ence between the exact optimal objective function value, i.e.,$f^{*(exact)} = f(x^*) = 0$ and the approximate optimal objective function value in each generation for the GA with N-points crossover and Arithmetic crossover, respectively.

Generations	$\tilde{f}^{*(NP)}$	$\tilde{f}^{*(Ar)}$	$\lvert\tilde{f}^{*(exact)} - \tilde{f}^{*(NP)}\rvert$	$\lvert\tilde{f}^{*(exact)} - \tilde{f}^{*(Ar)}\rvert$
100	9.0055e-21	0.000030	9.0055e-21	0.000030
200	5.5354e-08	0	5.5354e-08	0
300	2.0534e-07	0	2.0534e-07	0
400	1.8005e-08	0	1.8005e-08	0
500	2.4664e-21	0	2.4664e-21	0
600	1.4907e-21	0	1.4907e-21	0
700	1.7856e-08	0	1.7856e-08	0
800	1.1696e-22	0	1.1696e-22	0
900	9.3518e-21	0	9.3518e-21	0
1000	2.5043e-21	0	2.5043e-21	0

Table 16. $\tilde{f}^{*(NP)}$ and $\tilde{f}^{*(Ar)}$ Denote the Values of Approximate Optimal Objective Function Obtained by GA with N-points Crossover and Arithmetic Crossover, Respectively, on Example 5.3

Example 5.4 is a maximization problem. As Figure 6 and Tables 7 and 8 show the GA with N-points crossover in all the generations has values of objective function more than the GA with Arithmetic crossover. In this test problem, the GA with N-points crossover has obtained better results with respect to the GA with Arithmetic crossover, in all the generations. In this example, we have no the exact solution. Hence, we cannot present the table of comparison between the approximate optimal solution and the exact optimal solution.

Example 5.5 is a minimization problem. This example is exactly the given example in [19]. Its aim is to show that the proposed GA with two crossovers can solve the given problems in [19]. Also, the approximate optimal objective function values obtained by the proposed GA with two crossovers are very better than the obtained results in [19]. In the following, a comparison among the proposed GA with two crossovers with the GA presented in [19] has been provided.

Generations	$f^{Lu-Fang}$
1	25.66289115677
2	24.92549240573
10	24.48538689294
13	24.44449063700
26	24.39455537353
27	24.05118705966
120	24.04365588515
383	24.03526445700
679	23.99371873073
788	23.98606775704
1079	23.98359498123

Table 17. $f^{Lu-Fang}$ Denotes the Values of Approximate Optimal Objective Function in [19] on Example 5.5

Generations	f^{NP}	f^{Ar}
100	3.211402	3.676057
200	2.775089	3.433574
300	2.622769	2.704444
400	2.592573	2.559593
500	2.273279	2.550977
600	2.068840	2.394250
700	2.035064	2.379471
800	2.003492	2.224468
900	2.001897	2.001175
1000	1.999701	2.009705

Table 18. f^{NP} and f^{Ar} Denote the Values of Approximate Optimal Objective Function on Example 5.5 Obtained by GA with N-points Crossover and Arithmetic Crossover, Respectively

In generation 120, the approximate optimal objective function value in Lu and Fang's GA is $f^{Lu-Fang} = 24.04365588515$ and the approximate optimal objective function value in the proposed GA with N-points crossover is $f^{NP} = 3.211402$ and the proposed GA with Arithmetic crossover is $f^{Ar} = 3.676057$ in generation 100. In two current cases, the algorithm has obtained the less approximate optimal objective values with respect to Lu and Fang's GA. In the other generations, e.g., generations 383 and 300, generations 679 and 600, generations 788 and 700, generations 1079 and 1000, respectively, from Lu and Fang's GA and the proposed GA with two crossover, the approximate optimal objective function values obtained by the proposed GA with two crossovers are very less than the approximate optimal objective function values resulted by Lu and Fang's GA with less generations. Since the objective of the problem is minimization, the proposed GA with two crossovers is more efficient from Lu and Fang's GA. As Figure 7 and Tables 9 and 10 show the GA with N-points crossover in generations $100, 200, 300, 500, 600, 700, 800$, and 1000 has values of objective function less than the GA with Arithmetic crossover. Only in generations 400 and 900, the GA with Arithmetic crossover has obtained better results. Overall, the GA with N-points crossover is more efficient with respect to the GA with Arithmetic crossover.

7. Conclusions

The nonlinear programming problems subject to bipolar fuzzy relation equation constraints were studied in this paper. Their feasible domain were investigated and the lower and upper bound of each variable were determined. A genetic algorithm with two crossovers: N-points and Arithmetic was designed to solve the problems. Some test problems were solved by the algorithm with two crossovers. Their results were compared to each other and the exact optimal solution. This comparative analysis shows the efficiency of the algorithm genetic with two crossover operations. Moreover, the GA with two crossovers was compared with Lu and Fang's GA [19].

References

[1] S. Abbasbandy, E. Babolian and M. Allame, *Numerical solution of fuzzy max-min systems*, Applied Mathematics and Computation, **174** (2006), 1321-1328.

[2] M. Allame and B. Vatankhahan, *Iteration algorithm for solving Ax=b in max-min algebra*, Applied Mathematics and Computation, **175** (2006), 269-276.

[3] B. De Baets, *Analytical solution methods for fuzzy relational equations*, in: D. Dubois, H. Prade (Eds.), Fundamentals of Fuzzy Sets, The Handbooks of Fuzzy Sets Series, Kluwer Academic Publishers, Dordrecht, (2000), 291-340.

[4] S. C. Fang and G. Li, *Solving fuzzy relation equations with a linear objective function*, Fuzzy Sets and Systems, **103** (1999), 107-113.

[5] D.B. Fogel, *Evolving Artificial Intelligence*, Ph.D. Thesis, University of California, 1992.

[6] S. Freson, B. De Baets and H. De Meyer, *Linear optimization with bipolar max-min constraints*, Information Sciences, **234** (2013), 3-15.

[7] R. Hassanzadeh, E. Khorram, I. Mahdavi and N. Mahdavi-Amiri, *A genetic algorithm for optimization problems with fuzzy relation constraints using max-product composition*, Applied Soft Computing, **11** (2011), 551-560.

[8] W. Hock and K. Schittkowski, *Test Examples for Nonlinear Programming Codes*, Lecture Notes in Economics and Mathematical Systems, vol. 187, Springer, New York, 1981.

[9] J. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, MI, 1975.

[10] A. Homaifar, S. Lai and X. Qi, *Constrained optimization via genetic algorithms*, Simulation, **62** (1994), 242-254.

[11] J. A. Joines and C. Houck, *On the Use of Non-stationary Penalty Function to Solve Nonlinear Constrained Optimization Problems with GAs*, In: Z. Michalewicz (Ed.), Proc. 1st IEEE Internat. Conf. on Evolutionary Computation, IEEE Service Center, Piscataway, NJ, (1994), 579-584.

[12] E. Khorram, A. Ghodousian and A. A. Molai, *Solving linear optimization problems with max-star composition equation constraints*, Applied Mathematics and Computation, **178** (2006), 654-661.

[13] E. Khorram and R. Hassanzadeh, *Solving nonlinear optimization problems subjected to fuzzy relation equation constraints with max-average composition using a modified genetic algorithm*, Computers and Industrial Engineering, **55** (2008), 1-14.

[14] P. Li and Y. Liu, *Linear optimization with bipolar fuzzy relational equation constraints using the lukasiewicz triangular norm*, Soft Computing, **18** (2014), 1399-1404.

[15] C. Lichun and P. Boxing, *The fuzzy relation equation with union or intersection preserving operator*, Fuzzy Sets and Systems, **25** (1988), 191-204.

[16] J. Loetamonphong and S. C. Fang, *An efficient solution procedure for fuzzy relation equations with max-product composition*, IEEE Transactions on Fuzzy Systems, **7** (1999), 441-445.

[17] J. Loetamonphong and S. C. Fang, *Optimization of fuzzy relation equations with max-product composition*, Fuzzy Sets and Systems, **118** (2001), 509-517.

[18] J. Loetamonphong, S. C. Fang and R. E. Young, *Multi-objective optimization problems with fuzzy relation equation constraints*, Fuzzy Sets and Systems, **127** (2002), 141-164.

[19] J. Lu and Sh. Fang, *Solving nonlinear optimization problems with fuzzy relation equation constraints*, Fuzzy Sets and Systems, **119** (2001), 1-20.

[20] L. Luoh, W. J. Wang and Y. K. Liaw, *New algorithms for solving fuzzy relation equations*, Mathematics and Computers in Simulation, **59** (2002), 329-333.

[21] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, New York, 1994.

[22] Z. Michalewicz and C. Janikow, *Handling Constraints in Genetic Algorithms*, Proc. 4th Internat. Conf. on Genetic Algorithms, Morgan Kaufmann Publishers, Los Altos, CA, (1991), 151-157.

[23] Z. Michalewicz and C. Janikow, *Genetic algorithms for numerical optimization*, Statistics and Computing, **1** (1991), 75-91.

[24] K. Peeva, *Universal algorithm for solving fuzzy relational equations*, Italian Journal of Pure and Applied Mathematics, **19** (2006), 169-188.

[25] K. Peeva, *Composite Fuzzy Relational Equations in Decision Making: Chemistry*, In: Cheshankov B, Todorov M (eds) Proceedings of the 26th summer school applications of mathematics in engineering and economics, Sozopol (2000), Heron Press, (2001), 260-264.

[26] K. Peeva and Y. Kyosev, *Fuzzy Relational Calculus: Theory, Applications and Software*, World Scientific, New Jersey, 2004.

[27] E. Sanchez, *Resolution of composite fuzzy relation equations*, Information and Control, **30** (1976), 38-48.

[28] M. Schoenauer and S. Xanthakis, *Constrained GA Optimization, in: S. Forrest (Ed.)*, Proc. 5th Internat. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1993.

[29] B. S. Shieh, *Solutions of fuzzy relation equations based on continuoust-norms*, Information Sciences, **177** (2007), 4208-4215.

[30] W. B. Vasantha Kandasamy and F. Smarandache, *Fuzzy Relational Maps and Neutrosophic Relational Maps*, Hexis Church Rock, 2004.

[31] Y. K. Wu, *Optimization of fuzzy relational equations with max-av composition*, Information Sciences, **177** (2007), 4216-4229.

[32] Y. K. Wu and S. M. Guu, *A note on fuzzy relation programming problems with max-strict-t-norm composition*, Fuzzy Optimization and Decision Making, **3** (2004), 271-278.

[33] Y. K. Wu and S. M. Guu, *Minimizing a linear function under a fuzzy max-min relational equation constraint*, Fuzzy Sets and Systems, **150** (2005), 147-162.

[34] Y. K. Wu, S. M. Guu and J. Y. C. Liu, *An accelerated approach for solving fuzzy relation equations with a linear objective function*, IEEE Transactions on Fuzzy Systems, **10(4)** (2002), 552-558.

[35] C. T. Yeh, *On the minimal solutions of max-min fuzzy relational equations*, Fuzzy Sets and Systems, **159** (2008), 23-39.

[36] K. Zimmerman, *Disjunctive optimization, max-separable problems and extremal algebras*, Theoretical Computer Science, **293** (2003), 45-54.

Hassan Dana Mazraeh, School of Mathematics and Computer Sciences, Damghan University, Damghan, Iran
E-mail address: dana@du.ac.ir

Ali Abbasi Molai*, School of Mathematics and Computer Sciences, Damghan University, Damghan, Iran
E-mail address: a_abbasimolai@du.ac.ir

*Corresponding author