Article citation info:
ZARINCHANG A, FAGHIH N, ZARINCHANG J. An application of Genetic Algorithm toward solving the reliability problem of multiobjective series-parallel systems. Eksploatacja i Niezawodnosc – Maintenance and Reliability 2012; 14 (3): 243-248.

Amir ZARINCHANG
Nezameddin FAGHIH
Jafar ZARINCHANG

AN APPLICATION OF GENETIC ALGORITHM TOWARD SOLVING THE RELIABILITY PROBLEM OF MULTIOBJECTIVE SERIES-PARALLEL SYSTEMS

ZASTOSOWANIE ALGORYTMU GENETYCZNEGO DO ROZWIĄZYWANIA ZADAŃ NIEZAWODNOŚCIOWYCH DOTYCZĄCYCH WIELOKRYTERIALNYCH SYSTEMÓW SZEREGOWO-RÓWNOLEGŁYCH

Since developing an appropriate solution for reliability optimization problem with mathematical programming methods has been considered as difficult techniques, the heuristic approaches increasingly has been applied. Multiobjectve Genetic Algorithm (MGA) has been among heuristic methods that was developed to find solutions for series-parallel systems to obtain maximum reliability, and minimum cost and weight at the system level. These are very common problems in engineering design such as mechanical and electrical systems. It has been shown that the Multiobjectve Genetic Algorithm offers proper results to these problems while it respects to the several objective functions such as reliability, cost and weight. This paper presents the combination of probabilistic search, and one of the decision making methods called Technique for Order Preference by Similarity to Ideal Solution (TOPSIS). The Multiobjectve Genetic Algorithm, allows us to achieve a proper design solution while it saves a considerable time compared with some other approaches. At the same time as the reliability, cost and weight were chosen as objective functions, the results obtained by this method showed an overall improvement in comparison to the existing GA method considering cost and weight as constraints.

Keywords: Multiobjective Genetic Algorithm, Reliability Optimization, Redundancy Apportionment, Series-Parallel Systems, TOPSIS Method.

Ponieważ znalezienie odpowiedniego rozwiązania zadania optymalizacji niezawodnościowej przy wykorzystaniu metod programowania matematycznego uznaje się za trudne, coraz częściej stosuje się do tego celu metody heurystyczne. Algorytm genetyczny do optymalizacji wielokryterialnej (Multiobjective Genetic Algorithm, MGA) jest jedną z metod heurystycznych, stworzoną w celu znajdowania rozwiązań dla systemów szeregowo-równoległych, pozwalającą na uzyskanie maksymalnej niezawodności oraz minimalnych kosztów i ciężaru na poziomie systemu. Zadania takie występują powszechnie w dziedzinie projektowania i konstrukcji systemów mechanicznych i elektrycznych. Wykazano, że MGA pozwala uzyskać odpowiednie rozwiązania tego typu zadań uwzględniając przy tym funkcje celu, takie jak niezawodność, koszty i ciężar. W niniejszej pracy przedstawiono połączenie metody wyszukiwania probabilistycznego oraz jednej z metod rozwiązywania problemów decyzyjnych o nazwie TOPSIS (Technique for Order Preference by Similarity to Ideal Solution). MGA pozwala uzyskać odpowiednie rozwiązania konstrukcyjne dając przy tym znaczną oszczędność czasu w porównaniu z niektórymi innymi metodami. Jednocześnie potraktowanie kosztów i ciężaru jako funkcji celu daje lepsze wyniki w porównaniu do metody wykorzystującej algorytm genetyczny, w której koszty i ciężar rozpatrywane są jako ograniczenia.

Słowa kluczowe: *Algorytm genetyczny do optymalizacji wielokryterialnej, optymalizacja niezawodności, podział nadmiarowości, systemy szeregowo-równoległe, metoda TOPSIS.*

1. Introduction

Reliability is an important factor in different kinds of electrical and mechanical systems. In many practical system designs, the overall system is divided to certain series parts called subsystems, according to the function requirements of the system (Fig. 1). For each subsystem, there are several parallel positions that can be filled with different component types available with varying reliability, costs, weight, volume and other characteristics [13]. Reliability of a system is calculated after computing the reliability of each subsystem. For optimizing system reliability, the following approach can be applied: (a) using more reliable components or applying better technical and organizational actions such as condition monitoring systems [19], (b) using redundant configuration with active or stand-by components in parallel, or (c) a combination of (a) and (b) [9], [13]. A well-known and complex reliability optimization problem is the Redundancy Apportionment Problem (RAP) for series-parallel systems which can be identified as the selection of the optimal combination of component type and redundancy level for each subsystem in order to meet various objectives on the overall system [13]. Conflicting objectives, such as minimizing the system cost and system weight or volume, while simultaneously maximizing the system reliability, make this problem complex. The RAP has proven to be NP-hard [2], [13]. Some approaches have been tried out to solve this problem [13], [14], [20]. The increase of the size and number of constrains affects computational difficulty exponentially, therefore, the majority of previous methods restricted the problem in a way. For instance, the number of components that could be chosen for a subsystem should not be more than one component. It means for providing redundancy only the same type can be used. Integer programming, dynamic programming, mixed integer and nonlinear programming, and multiobjective approaches could not find a solution if they did not apply restrictions. A method that could overcome these obstacles was Genetic Algorithm. Coit and Smith [3], [4], [13] solved the RAP problems by using Genetic Algorithm (GARAP). The results of GARAP have been much better. Through many generations GA collects best solutions with an improving strategy and this method even results for large-scale problems [13]. In the existing GARAP cost and weight are considered as constraints, however, in this paper cost and weight are considered as objective functions.

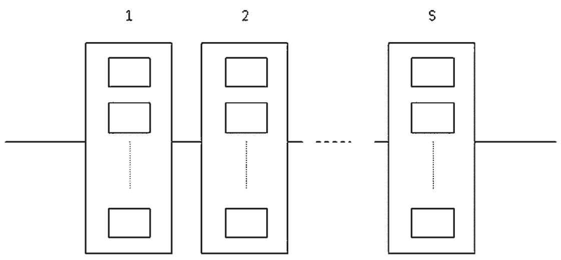

Fig.1. A Series-Parallel System

2. Notations, acronyms and assumptions

A. *Notations*

R_s	reliability of system
C_s	cost of system
W_s	weight of system
s	number of subsystems
m_i	number of available component choices for subsystem i (i = 1,...,s)
r_{ij}	reliability of j th component for subsystem i
c_{ij}	cost of j th component for subsystem i
w_{ij}	weight of j th component for subsystem i
n_i	total number of components used in subsystem i
$n_{\max}$	maximum number of components in parallel (user specified)
$R_i(x_i)$	reliability of subsystem i
$C_i(x_i)$	total cost of subsystem i
$W_i(x_i)$	total weight of subsystem i
v_j	vector encoding of solution j
λ_r	importance of system reliability (weight of reliability in TOPSIS method)
λ_w	importance of system weight (weight of weight in TOPSIS method)
λ_c	importance of system cost (weight of cost in TOPSIS method)
p	population size
f(v_j)	fitness for j th member of the population

B. *Acronyms*

GA(s)	Genetic Algorithm(s)
MGA	Multi-Objective Genetic Algorithm
RAP	Redundancy Apportionment Problem
OIMP	Overall Improvement

C. *Assumptions*

1. The reliability of each component is known.
2. Components are not repairable.
3. Failure of each component do not depends on other components.

3. Problem formulation

For modeling an RAP for the series-parallel systems, three objective functions must be optimized, that is, to maximize reliability (1) and minimize cost (3) and weight (4). At least on component in parallel is specified for each subsystem to operate. This problem can be easily expanded by adding more objective functions and be solved by the GA - the feature that previous formulations did not have [3] [10].

$$Max\ R_s = \Pi_{i=1}^{S} R_i(x) \tag{1}$$

$$R_i = (1 - \Pi_{j=1}^{n_{\max}} (1 - r_{ij})) \tag{2}$$

$$Min\ C_s = \sum_{i=1}^{S} \sum_{j=1}^{n_{\max}} c_{ij} \tag{3}$$

$$Min\ W_s = \sum_{i=1}^{S} \sum_{j=1}^{n_{\max}} w_{ij} \tag{4}$$

The total number of unique system configurations is given by the following equation [7], [13]:

$$N = \Pi_{i=1}^{S} \left[\binom{m_i + n_{\max}}{m_i} - \binom{m_i + 1 - 1}{m_i} \right] \tag{5}$$

4. Genetic algorithm implementation

GA is a stochastic global search method that mimics the process of natural biological evolution [6], GA operates on a population of potential solutions applying the principle of survival of the fittest to generate (hopefully) better and better approximations to a solution [21]. The steps of GA are [3], [15]:

1. Encoding of solutions
2. Initialing a population of chromosomes
3. Selecting parents for breeding
4. Creating new chromosomes by transferring best strings and crossover operator; applying mutation as the parents mate
5. If the termination criterion is met, stop and return the best chromosome; If not go to 3

Crossover and mutation operators play an important role in the GA. The rate of convergence is specified by the effectiveness of the crossover operator at the same time as the mutation operator restrains the algorithm not to stop in a local optimal and transferring best strings help algorithm to keep the best solutions in the next generations. The number of subsystems, maximum number of components in parallel, transferring and mutation rate is tunable parameters but constant in specific experiment [3], [4].

A. *Solution Encoding*

Although traditional GA encoded solution using a binary string [3], [11] for combinatorial optimization it is preferred and more efficient to encode solution with integer values. In this approach the second encoding method has been selected. Each subsystem includes n_i parts in parallel ($1 \le n_i \le n_{\max}$) in order to form a possible solution. The n_i parts can be selected from m_i components that are available for i th subsystem. Components are sorted according to their reliabilities from 0 to

(m_i-1), that is, the most reliable component is shown by 0 and the least reliable is shown by, m_i-1. Chromosome is a vector with ($n_{\max}$ * s) positions. It means for each of s subsystems there are $n_{\max}$ positions that can be filled with an integer number. An index −1 is assigned to the empty position where there is no component (i.e., $n_i < n_{\max}$). At last all the subsystem representations are placed next to each other and form a vector that shows a chromosome [3]. As an example, consider a system with $s=3$, $m_1=4$, $m_2=3$, $m_3=4$ and $n_{\max}$ predetermined to be 5. The following example:

v_j = (3 3 3 −1−1 | 0 0 1 −1 −1 | 2 1 −1 −1 −1)

represents a prospective solution with three of the fourth most reliable components for the first subsystem; two of the most reliable component and one of the second most reliable component used in parallel for the second subsystem; and the one of the third most reliable and one of the second most reliable component used in parallel for the third subsystem.

B. *Initial Population*

When user determined population size (*p*) and the number of subsystems (s), the initial population could be generated. For each gene (position) an integer between −1 and (m_i−1) was generated randomly and uniformly (with replacement) according to which subsystem this gene belonged to (to determine m_i). Previous article [5] indicated that a population size of 40 converged quickly and produced good solutions. In general, the minimum effective population size would grow with problem size [3], [11].

C. *Objective Function*

By generating initial population, p vectors were produced that each one demonstrates a system design. Reliability, weight and cost of generated solutions are calculated according to (1), (2), (3) and (4), therefore, a (p*3) primary matrix can be formed as illustrated in (fig.2).

(R_{sj}, W_{sj}, C_{sj}) Show reliability, weight and cost of j^{th} design.

$$\begin{bmatrix} R_{s1} & W_{s1} & C_{s1} \\ R_{s2} & W_{s2} & C_{s2} \\ . & . & . \\ . & . & . \\ R_{sp} & W_{sp} & C_{sp} \end{bmatrix}$$

Fig. 2.Primary matrix

As primary matrix shows, there are three attributes for each alternative, so, they are not easily ranked. Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) is applied to overcome ranking problem. Hwang and Yoon [12] developed the TOPSIS technique based on the concept that “the best alternative should have the shortest distance from the positive–ideal solution and the longest distance from the negative-ideal solution” and the ideal solution is the collection of ideal scores (or ratings) in all attributes considered [1]. Therefore, to use TOPSIS the following steps should be passed:

Step 1) Normalizing the Primary Matrix (Fig.3)

Through this step, three attribute dimensions transform to non-dimensional attributes which allow comparison across criteria.

$$R'_{sj} = \left[R_{sj} \Big/ \left(\sum_{j=1}^{p} R_{sj}^2 \right)^{\frac{1}{2}} \right] \quad (6)$$

$$W'_{sj} = \left[W_{sj} \Big/ \left(\sum_{j=1}^{p} W_{sj}^2 \right)^{\frac{1}{2}} \right] \quad (7)$$

$$C'_{sj} = \left[C_{sj} \Big/ \left(\sum_{j=1}^{p} C_{sj}^2 \right)^{\frac{1}{2}} \right] \quad (8)$$

$$\begin{bmatrix} R'_{s1} & W'_{s1} & C'_{s1} \\ R'_{s2} & W'_{s2} & C'_{s2} \\ . & . & . \\ . & . & . \\ R'_{sp} & W'_{sp} & C'_{sp} \end{bmatrix}$$

Fig.3. Normalized matrix

Step 2) Calculating Weighted Normalized Matrix (Fig.4)

Since attributes have different importance, a set of weights (λ_r, λ_w, λ_c) are assigned to them depends on how important the attributes are for users. It should be considered that sum of weights is equal to one.

($\lambda_r + \lambda_w + \lambda_c = 1$)

$$R''_{sj} = \left(\lambda_r \times R'_{sj} \right) \quad (9)$$

$$W''_{sj} = \left(\lambda_w \times W'_{sj} \right) \quad (10)$$

$$C''_{sj} = \left(\lambda_c \times C'_{sj} \right) \quad (11)$$

$$\begin{bmatrix} R''_{s1} & W''_{s1} & C''_{s1} \\ R''_{s2} & W''_{s2} & C''_{s2} \\ . & . & . \\ . & . & . \\ R''_{sp} & W''_{sp} & C''_{sp} \end{bmatrix}$$

Fig. 4. Weighted Normalized matrix

Step 3) Identifying Positive-Ideal and Negative-Ideal Solutions

$$I^+ = \left\{ \max R''_{sj}, \min W''_{sj}, \min C''_{sj} \right\} = \left\{ R_s^+, W_s^+, C_s^+ \right\}$$

I^+ is a set of the best values for attributes among all alternatives. Best values for 2nd and 3rd columns are their minimum values since the least is the best.

$$I^- = \left\{ \min R''_{sj}, \max W''_{sj}, \max C''_{sj} \right\} = \left\{ R_s^-, W_s^-, C_s^- \right\}$$

I^- is a set of the worst values for attributes among all alternatives.

Step 4) Calculating Separation Measures

The distances d_j^+ and d_j^- to I^+ and I^- for all solutions are correspondingly computed according to (13) and (14):

$$d_j^+ = \left[\left(R''_{sj} - R_s^+\right)^2 + \left(W''_{sj} - W_s^+\right)^2 + \left(C''_{sj} - C_s^+\right)^2\right]^{1/2} \qquad (13)$$

$$d_j^- = \left[\left(R''_{sj} - R_s^-\right)^2 + \left(W''_{sj} - W_s^-\right)^2 + \left(C''_{sj} - C_s^-\right)^2\right]^{1/2} \qquad (14)$$

Step 5) Calculating the Relative Closeness to the Ideal Solution

The TOPSIS technique defines a "Similarity index" (or relative closeness) by combining the closeness to the positive-ideal solution and the remoteness of the negative-ideal solution. A relative distance D_j^+ comprised between [0,1] is assigned to each alternative which is the GA fitness function f(v_j) in this method. (The more relative closeness to the Ideal Solution means the solution is better)

$$D_j^+ = d_j^- \,/\, (d_j^+ + d_j^-) \qquad (15)$$

Figure 5 [16] illustrates how TOPSIS works. Given an alternative like a_j, the distances d_j^+ and d_j^- to I^+ and I^- correspondingly are computed, subsequently a relative distance D_j^+ comprised between [0,1] is assigned to each alternative.

D. *Transferring Best Strings*

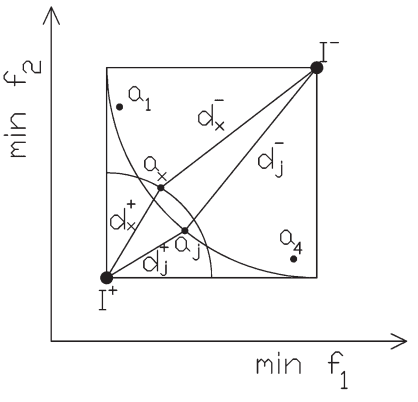

Fig. 5. TOPSIS distances [16]

Transferring best strings helped GA not to lose the best solutions in iterations. Transferring rate (0<TR<1) predetermined by user, therefore, in each generation (p * TR) chromosomes were kept without any changes according to their objective function values.

***E.** Selection of Parents and Crossover Breeding Operator*

Crossover is the most important search operator in Genetic Algorithm, as it is a recombination operator which constructs two offspring out of genetic information encoded in two selected parents [21]. Roulette wheel [15] mechanism is employed by many selection techniques to select individuals on the basis of some measure of their performance probabilistically. The basic roulette wheel selection method is stochastic sampling with replacement (SSR) [15]. In each generation a string has an objective or fitness function value and sum of these values formed total fitness (16). v_j could be selected as a parent with the probability of $\frac{\mathrm{f(v_j)}}{\mathrm{totalfit}}$ based on roulette wheel method.

$$\text{Totalfit} = \sum_{j=1}^{p} f(v_j) \qquad (16)$$

This crossover operator is a variation of the Single-Point operator which has been shown [22] to be superior to the Uniform crossover strategies for analyzing these problems. It also prevents creation of infeasible offspring during evolutions. In Single-Point operator same cutting point is selected randomly in both parents and then all data before the point is swapped between the parents. Two generated chromosomes are offspring.

***F.** Mutation Operator*

In simple Genetic Algorithm [21], mutation is the less used but not the less important operator in comparison to crossover. Each gene in an offspring which has been created by crossover can be mutated with respect to the mutation probability (mutation rate) which is predetermined by user according to the size of the problem (length of the chromosome) [21]. A mutated component was changed to (its index + 1) and the last component (m_i) was changed to a position where an additional component was not used (−1).

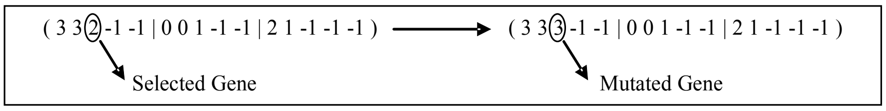

Fig. 6. Mutation Operator

***G.** Evolution*

A survival of the fittest strategy is applied [3]. (p * TR) of the best solutions were copied to the next generation without any changes (transferring best strings) and the rest of new population that created by crossover operator were [p * (1 − TR)]. Mutation had its effect while producing new offspring with crossover operator. The best strings were never mutated because the best solutions should not be altered via GA. Since the GA is a stochastic search method, it is difficult to certainly find termination criteria. A common practice is to terminate the GA after a preselected number of generations in spite of having reached optimal solution much earlier [3], [15].

5. Test problem and results

***A.** Test Problem*

The GA was used to analyze a different problem with very good results so it was implemented on the problem of the Fyffe, Hines and

Table 1. Best solution

Best Solution Vector	R_{sb}	W_{sb}	C_{sb}
002-1-1-10-10-1232-1-1-10-11011-1-1-13-1-1-13-1-11-1110-10-100-1-1-1-112-11-1-10-10101-100-1-10-1-1-1-13-1	0.9572	150	94

Table 2. Coit and Smith results

No.	Cost (C_{sa})	Weight (W_{sa})	Reliability (R_{sa})	OIMP%	No.	Cost (C_{sa})	Weight (W_{sa})	Reliability (R_{sa})	OIMP%
1	130	191	0.95675	0.95	15	123	174	0.97435	0.60
2	129	190	0.95603	0.90	19	122	173	0.97362	0.54
3	130	159	0.95556	0.94	20	120	172	0.97266	0.43
4	130	155	0.95503	0.93	21	121	171	0.97156	0.50
5	129	157	0.95429	0.59	22	120	170	0.97076	0.46
6	125	156	0.95362	0.53	23	120	169	0.96922	0.52
7	130	155	0.95311	0.93	24	119	165	0.96513	0.45
5	125	154	0.95239	0.52	25	115	167	0.96634	0.50
9	130	153	0.9519	0.92	26	116	166	0.96504	0.40
10	126	152	0.95102	0.71	27	117	165	0.96371	0.52
11	125	151	0.95006	0.55	25	115	164	0.96242	0.42
12	129	150	0.97942	0.90	29	114	163	0.96064	0.42
13	125	179	0.97906	0.64	30	114	162	0.95912	0.45
14	127	175	0.9751	0.75	31	113	161	0.95503	0.43
15	125	177	0.97715	0.65	32	114	160	0.95567	0.63
16	124	176	0.97642	0.62	33	110	159	0.95432	0.35
17	122	175	0.97552	0.51					

Lee problem [3], [8]. Five trials were performed for 1200 generation because the nature of GAs is stochastic, the final solution was found among the each and the best solution of five trials [3]. The results by the Multiobjective GA were used to compare its performance to existing GA. The size of the search space in this problem is greater than 7.6 x $10^{33}$from (5) [13].

B. Results

The best solution is showed (where (λ_r λ_w, λ_c) = (0.8, 0.1, 0.1)) in Table 1 and it is compared with results of Coit and Smith [4] in Table 2 which only considered reliability as an objective.

The overall improvement (OIMP %) is calculated according to (17) :

$$OIMP\ \ (\%) = \begin{bmatrix} (R_{sb}-R_{sa})/R_{sa}*\lambda_r + (W_{sa}-W_{sb})/ \\ W_{sa}*\lambda_w + (C_{sa}-C_{sb})/C_{sa}*\lambda_c \end{bmatrix}*100 \qquad (17)$$

6. Conclusion

GA has been demonstrated as a useful approach for solving Redundancy Apportionment Problem. In this paper we introduced MGA to provide a solution for systems that had to consider more than one objective, which since then it had not been reported that this problem was solved under this Multiobjective Genetic Algorithm formulation. Although the reliability of this result was not as proper as the previous formulation, the overall result showed a relative improvement. This algorithm did not use a complex decision making technique or local search to improve solutions but it seems that considering these features provides opportunities to have better results and more effective and efficient MGA.

References

1. Azar F. S. Multiattribute Decision-Making, use of three scoring Methods to compare the performance of imaging techniques for breast cancer detection, Technical Report, MS-BE-00-01MS-CIS-00-10, 2000.
2. Chen M. S. On the computational complexity of reliability redundancy allocation in a series system. Operation Research Letters 1992; 11:309–315.
3. Coit D. W, Smith A. E. Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Transaction on Reliability 1996; 45(2): 254–260.

4. Coit D. W, Smith A. E. Penalty guided genetic search for reliability design optimization, Computer and Industrial Engineering 1996; 30(4):895–904.
5. Coit D. W, Smith A. E. Use of a genetic algorithm to optimize a combinatorial reliability design problem, Proceedings of the 3rd International Engineering Research Conference 1994; 467-472.
6. Ebson H. A genetic algorithm for macro cell placement, Proceeding of the European Design Automation Conference 1992; 52-57.
7. Feller W. An introduction to probability theory, New York, Wiley 1965.
8. Fyffe D. E, Hines W.W, Lee N.K. System reliability allocation and a computational algorithm, IEEE Transactions on Reliability 1965; vol R-17, 64-69.
9. Gen M, Cheng M. Genetic algorithms and engineering design, John Wiley & Sons, New York, 1997.
10. Glaß M, Lukasiewycz M, Haubelt C, Teich J. Lifetime Reliability Optimization for Embedded Systems: A System-Level Approach, Proceedings of International Workshop on Reliability Aware System Design and Test 2010; 17-22.
11. Goldberg D. E. Genetic Algorithms in Search, Optimization & Machine Learning, Addison Wesley, 1959.
12. Hwang C. L, Yoon K. Multiple Attribute Decision Making: Methods and applications, A state of the art survey, Springer-Verlag, Berlin, 1951.
13. Jian-Hua Z, Zhaoheng L, My-Thien D. Reliability optimization using multiobjective ant colony system approaches, Reliability Engineering and System Safety 2007; 92(1): 109-120.
14. Kuo W, Rajendra Prasad V. An annotated overview of system reliability optimization, IEEE Transaction on Reliability 2000; 49(2): 176-187.
15. Lawrence D. Handbook of genetic algorithm, Van Nostrand Reinhold, 1991.
16. Méndez M, Galván B, Salazar D, Greiner D, Multiple-Objective Genetic Algorithm using the Multiple Criteria Decision Making Method TOPSIS ,7th International Conference on Multi-Objective Programming and Goal Programming, 2006; 145-150.
17. Syswerda G. Uniform crossover in Genetic Algorithms, Proceedings of the 3rd International Conference on Genetic Algorithms 1959; 2-9.
18. Tate D. M, Smith A. E. A genetic approach to the quadratic assignment problem , Computers and Operations Research 1994; vol. 22, 73-53.
19. Tian Z, Levitin G, Zuo M. J. A joint reliability redundancy optimization approach for multi-state series–parallel systems, Reliability Engineering and System Safety 2009; 1568–1576.
20. Tillman F. A, Hwang C. L, Kuo W, Optimization techniques for system reliability with redundancy a review, IEEE Transaction on Reliability 1977; 26(3):145–55.
21. Zalzala A. M. S, Fleming P. J. Genetic algorithms in engineering systems, The Institution of Electrical Engineers, London, 1997.
22. Zarinchang A. Applications of Genetic Algorithm for Solving the Reliability Problem of Multi-Objective Series-Parallel Systems, MS Thesis, 2008.

Mr. Amir ZARINCHANG, Lecturer
Department of Industrial Engineering
Shiraz Branch, Islamic Azad University
Shiraz, Iran
amir_zarinchang@yahoo.com

Prof. Nezameddin FAGHIH, Professor
Department of Management
Shiraz University
Shiraz, Iran
faghihnezam@ut.ac.it

Dr Jafar ZARINCHANG , Associate Professor
Department of Mechanical Engineering
Shiraz University
Shiraz, Iran
jmzarin@yahoo.com