Scientia Iranica E (2020) 27(2), 970–982

Sharif University of Technology
Scientia Iranica
Transactions E: Industrial Engineering
http://scientiairanica.sharif.edu

Presenting a series-parallel redundancy allocation problem with multi-state components using recursive algorithm and meta-heuristic

M. Sharifi[a,*], M. Saadvandi[a], and M.R. Shahriari[b]

a. *Faculty of Industrial & Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran.*
b. *Faculty of Management & Accounting, South Tehran Branch, Islamic Azad University, Tehran, Iran.*

Received 9 June 2018; received in revised form 15 September 2018; accepted 19 November 2018

KEYWORDS
Reliability optimization;
Multi-state components;
Redundancy allocation problem;
Recursive algorithm;
Genetic algorithm.

Abstract. Redundancy Allocation Problem (RAP) is one of the most important problems in the field of reliability. This problem is aimed at increasing system reliability under constraints such as cost, weight, etc. This study works on a system with series-parallel configuration and multi-state components. To draw the problem nearer to the real condition, this study merges this problem with discount levels in purchasing components. For calculating the reliability of sub-systems, a recursive algorithm is used. Because the redundancy allocation problem belongs to NP-hard problems, for optimizing the presented model, a new Genetic Algorithm (GA) was used. The algorithm parameters were tuned using Response Surface Methodology (RSM), and an enumeration method was used for the validation of GA.

1. Introduction

The simplest model of the Redundancy Allocation Problem (RAP) is to assign identical components to each subsystem. In mathematical models originally provided for redundant allocation problems, it is assumed that the components of the systems are in a binary state. This means that the components have only two states: working or failed. This study intends to model the problem with multi-state system components in order to get the problem closer to real-world conditions. Multi-state components have several functional states ranging from working to failed states.

Fyffe et al. [1] presented a mathematical model of the general RAP problem. Their proposed objective function was to maximize the reliability under weight and cost constraints. They solved the model by using dynamic programming. Ida and Yokota [2,3] provided a simple Genetic Algorithm (GA) for solving RAP without the possibility of allocating non-identical components to each subsystem in a series-parallel system of several failed states. Applying changes in the objective function, Coit and Smith [4] solved the problem using GA. One of the major difficulties in solving RAP with GA is the production and selection of infeasible solutions. For this reason, the penalty functions were defined to reduce the chance of selecting these infeasible solutions. Coit and Smith [5] presented an effective penalty function for RAP. Coit and Smith [6,7] introduced a new model with a solution to RAP. They used GA to solve the proposed new model in parallel-series systems with k-out-of-n: G subsystems. The main characteristic of the proposed algorithm is the presentation of the algorithm chromosome. Coit [8] presented a new model with a solution method for RAPs with

*. *Corresponding author.*
E-mail address: mani.sharifi@yahoo.com (M. Sharifi)

doi: 10.24200/sci.2018.51176.2040

a parallel-series structure. Tavakkoli-Moghaddam et al. [9] applied the GA to solve RAP. The main characteristics of the proposed algorithm include the design of chromosomes and mutation operators. Tavakkoli-Moghaddam and Safari [10] provided a new model for redundant allocation problems with the possibility of allocating non-homogeneous components to each subsystem and, also, choosing a redundancy policy for each subsystem. Chambari et al. [11] presented a two-objective model for the RAP in parallel-series systems under assumptions such as non-reparability. Zaretalab et al. [12] solved the model presented by Chmbari by means of knowledge-based-archive Simulated Annealing (SA) algorithm (knowledge-based archive simulated annealing). They showed that their proposed meta-heuristic algorithm was better than other algorithms. For the first time, Ushakov introduced the concept of Universal Generating Function (UGF) and applied it to calculate the reliability of systems with multi-state components. Li and Zuo [13] reported that their proposed method (when the number of system components is high) could reduce the computational time significantly, compared to the UGF method. The proposed method is well known as a recursive algorithm. Pourkarim Guilani et al. [14] used a modified Markov process and provided a new method for calculating the reliability of a system with three-state components and yielding a much shorter computational time than UGF method and recursive algorithm. Pourkarim Guilani et al. [15] solved a mathematical model of RAP with subsystems consisting of three-state components using complete numerical methods and GAs that cannot be generalized to other multi-state systems.

The main objective of this paper is to consider discount levels when the redundant components are purchased from the suppliers. Therefore, when the suppliers offer a general discount on each component, then the unit price of each component depends on the total number of the components purchased from that supplier. In this case, the price is the level of discount considered for the total purchased components [16]. Table 1 shows some new research results in the field.

In this paper, in order to optimize the system reliability, the recursive method is used and preferred over the UGF method due to its faster computing speed. The performance of the recursive algorithm to evaluate the reliability of multi-state systems is satisfactory. This method also enjoys a shorter computational time to perform than other evaluation methods such as universal generation function. In the research studies conducted by Guilani et al. [14] and Li and Zuo [13], the aforementioned result is confirmed. Moreover, due to the affiliation of RAP to NP-hard problems, the GA was used to obtain an optimal combination of components.

This paper is divided into five sections. The second section defines the problem definitions. The third section deals with the solving methods. Section 4 is a numerical example, and the last section deals with conclusion and further studies.

2. Problem definition

2.1. The proposed model

Consider a system consisting of s subsystems that are connected serially, and each subsystem has n_i parallel components. The components of each subsystem are multi-state and non-repairable. Moreover, the price of each component is calculated according to the total amount of the purchase and has a discount level. This model aims to determine the optimal number and type of components in each subsystem, considering that only one type of component is assigned to each subsystem from a list of component types, and that the objective function is to minimize the system cost.

2.2. Model assumptions

- Each component is of multi-state type;
- System parameters such as cost and weight are constant;
- The components are non-repairable;
- Components' failures are independent and the failure of each component does not damage the system.

2.3. Nomenclatures

i	Subsystem index, $i = 1, 2, \cdots, S$
S	Number of subsystems
$n_{i,\max}$	Maximum allowable components in subsystem i
j	Components type index, $j = 1, 2, \cdots, m_{i,\max}$
$m_{i,\max}$	Maximum available component types for the subsystem i
c_{ijk}	Price of components type j in subsystem i at discount level k, $k = 1, 2, \cdots, \lambda_{ij,\max}$
λ_{ijk}	Discount level k for components type j in subsystem i
n_{ijk}	Maximum purchase amount for discount level k for components type j in subsystem i
N_{ij}	Order value of components type j in subsystem i
$\lambda_{ij,\max}$	Maximum discount level of components type j in subsystem i
w_{ijk}	Weight of components type j in subsystem i at discount level k, $k = 1, 2, \cdots, m_{ij}$
W	Maximum acceptable weight of system

Table 1. Some recent studies on reliability area.

Authors	Year	State	Algorithm	Repairable	Objective	Parameter setting	Failure rate
Garg et al. [17]	2013	Binary	Bee colony	—	Single	No	Constant
Levitin et al. [18]	2013	Multi-state	GA	—	Single	No	Constant
Maatouk et al. [19]	2013	Multi-state	GA	✓	Single	No	Constant
Chambari et al. [11]	2013	Binary	SA	—	Single	No	Constant
Gago et al. [20]	2013	Binary	Greedy, walk back	—	Single	No	Constant
Ebrahimipour et al. [21]	2013	Binary	Fuzzy Inference System (FIS)	—	Single	No	Constant
Liu et al. [22]	2013	Multi-state	Imperfect repair model	✓	Single	No	Constant
Khalili-Damghani et al. [23]	2014	Binary	e-constraint	—	Multiple	No	Constant
Guilani et al. [15]	2014	Multi-state	Markov model	—	Single	No	Constant
Sharifi et al. [24]	2015	Binary	GA, MA	—	Single	RSM	Time dependent-load sharing
Mousavi et al. [25]	2015	Multi-state	CE-NRGA	—	Multiple	Taguchi	Constant
Zaretalab et al. [12]	2015	Multi-state	MOSA	—	Multiple	No	Constant
Miriha et al. [26]	2017	Binary	NSGA-II MOEA/D	—	Multiple	Taguchi	Time dependent
Guilani et al. [27]	2017	Multi-state	SPEA-II NSGA-II	—	Multiple	No	Constant
Hadipour et al. [28]	2018	Binary	NSGA-II NRGA	✓	Single	Taguchi	Constant
Guilani et al. [29]	2018	Binary	Simulation	—	Single	—	Time dependent

ω	Minimum acceptable system performance rate
$A(\omega)$	System availability
A_0	Minimum acceptable system availability

2.4. Mathematical model

The mathematical model of RAP considering the model assumptions is as follows:

$$\min \quad Z=\sum_{j=1}^{S}\sum_{j=1}^{m_{i,\max}}\left(\sum_{k=1}^{\lambda_{ij,\max}}c_{ijk}.\lambda_{ijk}\right).N_{ij}, \tag{1}$$

$$\text{S.t.:}\quad \sum_{j=1}^{S}\sum_{j=1}^{m_{i,\max}}w_{ij}.N_{ij}\leq W, \tag{2}$$

$$N_{ij}\geq\lambda_{ijk}.n_{ij(k-1)}$$

$$\forall:\begin{cases}i=1,2,\cdots,S\\ j=1,2,\cdots,m_{i,\max}\\ k=1,2,\cdots,\lambda_{ij,\max}\end{cases} \tag{3}$$

$$\sum_{k=1}^{\lambda_{ij,\max}}\lambda_{ijk}=1$$

$$\forall:\begin{cases}i=1,2,\cdots,S\\ j=1,2,\cdots,n_{i,\max}\end{cases} \tag{4}$$

$$\sum_{j=1}^{n_{i,\max}}N_{ij}\geq 1 \qquad \forall\ i=1,2,\cdots,S, \tag{5}$$

$$\sum_{j=1}^{n_{i,\max}}N_{ij}\leq n_{i,\max} \qquad \forall\ i=1,2,\cdots,S, \tag{6}$$

$$A(\omega)\geq A_0. \tag{7}$$

Eq. (1) is an objective function that minimizes system costs. Eq. (2) is the system weight constraint. Eq. (3) defines the discount level, i.e., it establishes the value of purchase at each discount level after determining the purchase value of each component in each subsystem. Eq. (4) ensures that the purchase value of component type j in subsystem i should be at the discount levels, and Eq. (5) implies that there is at least one component in each subsystem. Eq. (6) ensures that the number of components in each subsystem does not exceed the maximum number of acceptable components. Finally, Eq. (7) specifies the minimum expected availability of the system.

For calculating the system availability (Eq. (7)), the recursive algorithm is used. This algorithm is presented in the next section.

3. Solving methods

3.1. Recursive algorithm

3.1.1. The weighted multi-state k-out-of-n: G system

In the multi-state system, each component of the system may be in different states and, in each state, the component shows a specific performance. When a component completely fails, its performance is 0 [13].

Definition 1. In a system with n components, each component of the system may be in one of the possible $(m+1)$ states.

The component i $(1\leq i\leq n)$, when placed in state j, has performance g_{ij}. In this case, the system is in the state j if the sum of the performances of all components of the system is greater than or equal to k_j. Assume that ϕ is the system structure function that indicates the state of the system, and G is the sum of the performances of all the system components. Based on the above definition, we have:

$$\Pr\{\phi\geq j\}=\Pr\{G\geq k_j\}. \tag{8}$$

Since state 0 is the worst state in the system, we have:

$$\Pr\{\phi\geq 0\}=1. \tag{9}$$

3.1.2. Recursive algorithm

To evaluate the reliability of the multi-state weighted k-out-of-n: G system using the recursive algorithm, the following parameters are first introduced. These parameters are only used in this section.

n	Number of components in the system
M	State with the highest possible performance
g_{ij}	The performance of component i in state j
p_{ij}	The probability that component i is in state j
q_{ij}	The performance of the component i when it is in a state lower than j, $q_{ij}=\sum_{i=0}^{j-1}p_{ij}$
k_j	Minimum total performance required to ensure that the system is in state j or higher
$R_j^I(k_j,n)$	The probability that the system is in the state j or higher.

Therefore, the recursive equation for evaluating the distribution of the system state is as follows [13]:

$$R_j^I(k_j,i)=\sum_{r=0}^{r=M}p_{i,r}.R_j^I(k_j-g_{i,r},i-1). \tag{10}$$

The partial conditions for this recursive equation are as follows:

Table 2. Components' state probabilities (p_{ij}) [1].

	$j=0$	$j=1$	$j=2$
$i=0$	0.1	0.2	0.7
$i=1$	0.4	0.2	0.4
$i=2$	0.3	0.5	0.2

Table 3. Components' state performance (g_{ij}) [1].

	$j=0$	$j=1$	$j=2$
$i=0$	1	2	3
$i=1$	1	3	4
$i=2$	1	3	5

$$R_j^I(k_j,0)=0 \qquad \text{when} \quad 0<k\leq k_j,$$

$$R_j^I(k_j,i)=1 \qquad \text{when} \quad i\geq 0 \quad \text{and} \quad k\leq 0. \tag{11}$$

For example, consider a weighted multi-state k-out-of-n: G system with three components. Each component has three possible states 0, 1, and 2. Tables 2 and 3 show the reliability and performance of all components.

In this example, $n=3$, $M=2$, $k_1=5$, and $k_2=10$. The reliability of the system is obtained through Eqs. (10) and (11) as follows [13]:

$$\begin{aligned}R_1^I(5,3)=&\sum_{r=0}^{2}p_{3,1}.R_1^I(5-g_{3,r},3-1)\\ =&p_{3,0}.R_1^I(5-1,2)+p_{3,1}.R_1^I(5-3,2)\\ &+p_{3,2}.R_1^I(5-5,2)=p_{3,0}.R_1^I(4,2)\\ &+p_{3,1}.R_1^I(2,2)+p_{3,2}.R_1^I(0,2)\\ =&p_{3,0}.R_1^I(4,2)+p_{3,1}+p_{3,2},\end{aligned} \tag{12}$$

$$\begin{aligned}R_1^I(4,2)=&\sum_{r=0}^{2}p_{2,1}.R_1^I(4-g_{2,r},2-1)\\ =&p_{2,0}.R_1^I(4-1,1)+p_{2,1}.R_1^I(4-3,1)\\ &+p_{2,2}.R_1^I(4-4,1)=p_{2,0}.R_1^I(3,1)\\ &+p_{2,1}.R_1^I(1,1)+p_{2,2}.R_1^I(0,1)\\ =&p_{2,0}.p_{1,2}+p_{2,1}+p_{2,2}\\ =&0.4^*0.7+0.2+0.4=0.88,\end{aligned} \tag{13}$$

$$\begin{aligned}R_1^I(5,3)=&p_{3,0}.R_1^I(4,2)+p_{3,1}+p_{3,2}\\ =&p_{3,0}.0.88+p_{3,1}+p_{3,2}=0.3^*0.88\\ &+0.5+0.2=0.964,\end{aligned} \tag{14}$$

$$\begin{aligned}R_2^I(10,3)=&\sum_{r=0}^{2}p_{3,1}.R_1^I(10-g_{3,r},3-1)\\ =&p_{3,0}.R_1^I(10-1,2)+p_{3,1}.R_2^I(10-3,2)\\ &+p_{3,2}.R_2^I(10-5,2)=p_{3,0}.R_2^I(9,2)\\ &+p_{3,1}.R_2^I(7,2)+p_{3,2}.R_2^I(5,2),\end{aligned} \tag{15}$$

$$\begin{aligned}R_1^I(5,2)=&\sum_{r=0}^{2}p_{2,r}.R_2^I(5-w_{2,r},2-1)\\ =&p_{2,0}.R_2^I(5-1,1)+p_{2,1}.R_2^I(5-3,1)\\ &+p_{2,2}.R_2^I(5-4,1)=p_{2,0}.R_2^I(4,1)\\ &+p_{2,1}.R_2^I(2,1)+p_{2,2}.R_2^I(1,1)\\ =&p_{2,0}.0+p_{2,1}.(p_{1,1}+p_{1,2})+p_{2,2}\\ =&0.2.(0.2+0.7)+0.4=0.58,\end{aligned} \tag{16}$$

$$\begin{aligned}&R_2^I(7,2)=p_{1,2}.p_{2,2}=0.7^*0.4,\\ &R_2^I(9,2)=0,\\ &R_2^I(10,3)=p_{3,1}.0.28+p_{3,2}.0.58\\ &\qquad=0.5^*0.28+0.2^*0.58=0.256.\end{aligned} \tag{17}$$

Therefore, the distribution of the system state is as follows [13]:

$$\begin{aligned}&\Pr(\phi\geq 0)=1,\\ &\Pr(\phi\geq 1)=0.964,\\ &\Pr(\phi\geq 2)=0.256,\\ &\Pr(\phi=2)=0.256,\\ &\Pr(\phi=1)=0.964-0.256=0.708,\\ &\Pr(\phi\geq 0)=1-0.964=0.036.\end{aligned} \tag{18}$$

3.2. Genetic Algorithm (GA)

In 1975, this algorithm was first introduced by Holland [30] at Michigan University and developed by him and his students. The original idea of this algorithm was derived from the Darwinian evolutionary theory in 1895. According to this theory, those creatures that are more adaptable to the environment survive. Information transmitted from each generation to the next generation is enclosed in chromosomes, and inherited properties are transmitted in this way. In this algorithm, according to the principle of survival of the fittest, the better population are combined together and, based on the suitability of each solution, this solution is repeated more often in the next generation. This process continues to reach an optimal solution.

3.2.1. Algorithm steps

Step 1: Generate a random population including n chromosome or initial solution;

Step 2: Evaluate the fitness function of each chromosome population;

Step 3: Create a new population based on the following steps:

- Selection of parent chromosomes by selective methods such as roulette wheel, tournament, randomly, competitive, and so on by crossover and mutation operators;
- Considering a certain value for the probability of crossover operator and, then, performing a combination operation on parents to create offspring;
- Considering a certain value for the probability of mutation operator and, then, using this operation to change one or more genes from a parent chromosome to achieve a new chromosome.

Step 4: Replacing new offspring in the new population.

3.2.2. Solution encoding

The problem chromosome is an $n_{S\times M}$ matrix, presented in Figure 1. In this matrix, S is the number of subsystems and M is the maximum type of components. These chromosomes are presented by Tavakkoli-Moghaddam et al. [9].

Assume that the model has three subsystems and four different component types; in subsystem 1, there are 2 components of type 1, 3 components of type 3, and 1 component of type 4; in subsystem 2, there are 3 components of type 1, 3 components of type 2, and 1 component of type 3; in subsystem 3, there are 4 components of type 1, 2 components of type 2, and 4 components of type 3. The chromosome matrix of this solution is presented in Figure 2.

$$\begin{bmatrix} N_{11} & N_{21} & \cdots & N_{M1} \\ N_{11} & N_{22} & \cdots & N_{M2} \\ \vdots & \vdots & \ddots & \vdots \\ N_{1S} & N_{2T} & \cdots & N_{MT} \end{bmatrix}$$

Figure 1. Model chromosome.

$$\begin{bmatrix} 2 & 0 & 3 & 1 \\ 3 & 3 & 1 & 0 \\ 4 & 2 & 4 & 0 \end{bmatrix}$$

Figure 2. Sample chromosome.

3.2.3. Initial population

The initial population is generated randomly, referred to as *npop*.

3.2.4. Fitness function

Because of the model constraints, the produced chromosome is not feasible. Therefore, the most important problem concerning the use of GA for problems with constraints is how to deal with constraints. Penalty functions are one of the first methods to deal with problems with constraints in GA. The penalty functions reduce infeasible solutions in accordance with the violation ratio of the constraints. In fact, the penalty function turns constrained problems into problems without constraints. Because of the problem constraints, the penalty functions are as follows:

$$p_1 = \max\{A_0 - A(\omega), 0\}, \tag{19}$$

$$p_2 = \frac{\max\left\{\sum\limits_{j=1}^{S}\sum\limits_{j=1}^{m_{i,\max}} w_{ij}.N_{ij} - W, 0\right\}}{W}, \tag{20}$$

$$p_2 = \sum_{i=1}^{S} \frac{\left\{\max\left\{\sum\limits_{j=1}^{m_{i,\max}} N_{ij} - n_{i,\max}\right\}, 0\right\}}{n_{i,\max}}. \tag{21}$$

Therefore, the general penalty function of the problem is presented as follows:

$$p_{\text{Total}} = p_1 + p_2 + p_3. \tag{22}$$

Moreover, the fitness function of the model is as follows:

$$F(x) = f(x).(1 + p_T). \tag{23}$$

Now, if the equality is satisfied, the value of p_T is 0 and the fitness function is the same as the objective function.

3.2.5. Crossover operator

In this operator, first, the number of parents is calculated with a crossover rate and, then, parents are randomly selected using the roulette wheel. To perform the crossover operator, firstly, the parent is selected and, then, the offspring is created using a uniform crossover operator. The operation of this operator was described in [10]. Intersection operations are performed on the parent chromosomes so that offspring's chromosomes are formed. In this operator, for each genome in the parent chromosome, a binary number is randomly generated; if this number is 1, the genome is replaced in the parent chromosomes; in addition, if the number is 0, it is not replaced. The crossover operator in the proposed GA is shown in Figure 3.

$$\text{Parent 1} \quad \begin{bmatrix} 2 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 3 & 0 & 1 \\ 0 & 4 & 0 & 2 & 0 \\ 1 & 0 & 3 & 0 & 0 \end{bmatrix} \qquad \text{Offspring 1} \quad \begin{bmatrix} 2 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 3 & 0 & 1 \\ 0 & 4 & 0 & 2 & 0 \\ 0 & 0 & 3 & 0 & 0 \end{bmatrix}$$

$$\xrightarrow{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}}$$

$$\text{Parent 2} \quad \begin{bmatrix} 1 & 0 & 4 & 1 & 0 \\ 3 & 1 & 0 & 2 & 0 \\ 0 & 3 & 0 & 3 & 0 \\ 0 & 2 & 2 & 2 & 0 \\ 0 & 1 & 1 & 1 & 2 \\ 0 & 3 & 0 & 0 & 1 \end{bmatrix} \qquad \text{Offspring 2} \quad \begin{bmatrix} 1 & 0 & 4 & 0 & 0 \\ 3 & 1 & 0 & 2 & 0 \\ 0 & 3 & 0 & 3 & 0 \\ 0 & 2 & 2 & 2 & 0 \\ 0 & 1 & 1 & 1 & 2 \\ 1 & 3 & 0 & 0 & 1 \end{bmatrix}$$

Figure 3. Crossover operator.

$$\text{Parent} \quad \begin{bmatrix} 1 & 0 & 4 & 1 & 0 \\ 3 & 1 & 0 & 2 & 0 \\ 0 & 3 & 0 & 3 & 0 \\ 0 & 2 & 2 & 2 & 0 \\ 0 & 1 & 1 & 1 & 2 \\ 0 & 3 & 0 & 0 & 1 \end{bmatrix} \xrightarrow{\begin{bmatrix} 0.79 & 0.18 & 0.37 & 0.62 & 0.65 \\ 0.74 & 0.06 & 0.35 & 0.93 & 0.42 \\ 0.81 & 0.58 & 0.31 & 0.64 & 0.44 \\ 0.03 & 0.83 & 0.27 & 0.39 & 0.31 \\ 0.71 & 0.76 & 0.49 & 0.52 & 0.63 \\ 0.02 & 0.84 & 0.03 & 0.73 & 0.59 \end{bmatrix}} \text{Offspring} \quad \begin{bmatrix} 1 & 0 & 4 & 1 & 0 \\ 3 & 1 & 0 & 2 & 0 \\ 0 & 3 & 0 & 3 & 0 \\ 0 & 2 & 2 & 2 & 0 \\ 0 & 1 & 1 & 1 & 2 \\ 1 & 3 & 1 & 0 & 1 \end{bmatrix}$$

Figure 4. Mutation operator.

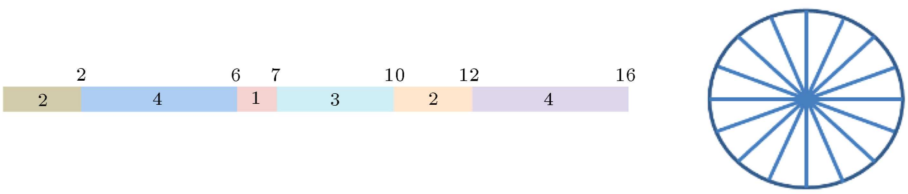

Figure 5. Roulette wheel method.

3.2.6. Mutation operator

In this operator, first, the number of parents is calculated with the mutation rate and, then, the parents are randomly selected using the roulette wheel. After selecting the parent, for each genome in the parent chromosome, a random number is generated between 0 and 1 and mutations are performed at a specific mutation rate of the parent chromosome genes. Now, if the generated random number is smaller than the desired mutation rate, the genome in the parent chromosome is randomly mutated. If the generated random number is larger than the mutation rate, the gene in the parent chromosome is not mutated [10]. This type of mutation is illustrated in Figure 4. In this figure, the mutation rate is considered as $p_M = 0.1$.

3.2.7. Selection

In this paper, the roulette wheel was used to select the population of the next generation. There are two main ideas in this way. First, better chromosomes have better chances of selecting and, second, the chances of selecting each chromosome are proportional to their fitness. For each chromosome, the fitness function is calculated and, then, the cumulative fitness function of the chromosomes is computed. Next, a random number is generated between 0 and the cumulative fitness function of the last chromosome. The corresponding number is compared with the cumulative fitness function, and the chromosome located at the corresponding distance is selected. The implementation of this method is shown in Figure 5.

3.2.8. Stop criteria

There are many criteria to stop the algorithm: the number of algorithms' iterations, the improvement of the objective function, and so on. In this algorithm, the number of algorithms' iteration has been used. It is implied here that this algorithm stops after a certain number of iterations and generation. The algorithm iteration is shown by $MaxIt$.

3.3. Parameter tuning

The time of the meta-heuristic algorithms depends on their input parameters. The goal of tuning the

Table 4. GA optimal values.

Parameter	Lower bound	Upper bound	Optimal value
$npop$	50	100	75
p_c	0.4	0.7	0.55
p_m	0.1	0.3	0.2

parameters of the algorithms is for them to reach appropriate solutions in a short amount of time. The parameter tuning method of the proposed algorithm is as follows. The proposed GA input parameters include population size ($npop$), intersection crossover probability (P_C), and probability of mutation (P_M). Response Surface Methodology (RSM) was used to identify the appropriate values of parameters. This study used a two-level factorial design method for tuning the algorithm parameters. For each experiment, two levels are considered that are high and low. In addition to the upper and lower limits, axial points and a number of central points (there are 5 central points) are also considered. In this model, considering the three existing parameters, the 2^3 factorial design is considered. Meanwhile, the stop criterion for parameter tuning is equal to 100 algorithm iterations, and the response variable in the model is the system reliability. The input values and the optimal value of each parameter are presented in Table 4.

4. Numerical example

In this section, the objective is to validate and solve the proposed model. To this end, first, a numerical example is designed. It should be noted that the numerical examples used in this paper are taken from [16]. It is assumed that the system consists of 14 subsystems and three price levels to buy components. The maximum acceptable weight for the system is 100, the maximum number of components in each subsystem is considered 6, the minimum acceptable availability (reliability) for the system is 0.9, and the minimum acceptable performance rate for the components is 50. It is also assumed that there are four types of components available to allocate to the subsystems.

Table 5 shows the values of reliability, weight, and cost for each component in all subsystems at different failure levels. Table 6 shows the performance rates of each component when placed in each of the subsystems. Table 7 shows the probability that the components would match the performance rate of each subsystem when placed in each of the subsystems.

Now, in order to confirm the correct operation of the GA, some small-sized problems are to be solved using the precise method of numerical rule, and the obtained solutions and their solving time are compared with the solutions and their solving time obtained from the GA. After ensuring that the proposed GA is validated in solving the proposed model, the proposed model is solved with large-sized problems by GA.

In this way, it is first assumed that the problem has five subsystems and, then, a problem with six subsystems is considered. More precisely, with the same number of parameters shown in Tables 2 to 4, the system with particular conditions is assumed: once only from the first to fifth subsystems and once only from the first to sixth subsystems. In addition, there are two types of components available to allocate to the subsystems.

The total number of problem solutions is obtained according to the numerical rule method of the following equation:

$$(n_{\max}+1)^{(S\times T)}. \tag{24}$$

It is also assumed that the maximum number of acceptable components in these problems is 2, the maximum acceptable weight is 60, the minimum acceptable performance rate for components is 30, and the minimum acceptable reliability for the system is 0.2.

Thus, according to Eq. (24), the total number of available solutions to the first problem is 59049 and to the second problem is 531441, and the highest number obtained in these repetitions is chosen as the optimal solution to each problem.

The enumeration method and the proposed GA are programmed for the two mentioned problems in MATLAB 17, and the best solution and the solving time for two problems are presented in Table 8.

By applying the enumeration method, an increase in the number of subsystems (or the number of component types increasing the feasible solutions of the problem) makes the enumeration method unable to generate optimal solutions to the problem in an appropriate amount of time. As observed earlier, the values of the solutions were the same for both methods of solving two problems, indicating the validity of the suggested GA for the proposed model. However, as the size of the problem enlarges, the solving time of the problem using the enumeration method becomes too long and the enumeration method is not a suitable method for solving the problem. Therefore, since the proposed GA has achieved the optimal solution to small-sized problems, it can be used to solve problems of larger sizes.

To perform sensitive analysis, 15 new problems were solved. It is assumed that the reliability of the component type 1 in the first subsystem varies from 0.82 to 0.96 and the other parameters of the problem are in accordance with those shown in Tables 4 to 6. The time and cost of these 15 problems are presented in Table 9.

Table 5. Components' reliability, cost, and weight.

Components type	Parameters	Number of subsystems													
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
	R	0.90	0.95	0.85	0.83	0.94	0.99	0.91	0.81	0.97	0.83	0.94	0.79	0.98	0.90
	W	3	8	7	5	4	5	7	4	8	6	5	4	5	6
	C_1	6	7	7	8	7	8	9	8	7	9	8	7	7	9
1	C_2	4	5	5	6	5	6	7	6	5	7	6	5	5	7
	C_3	3	4	4	4	4	4	5	4	4	5	4	4	4	5
	n_1	2	2	3	2	2	2	2	3	2	2	2	3	3	2
	n_2	4	3	4	3	4	3	4	5	3	3	4	5	3	3
	R	0.93	0.94	0.90	0.87	0.93	0.98	0.92	0.90	0.99	0.85	0.95	0.82	0.99	0.92
	W	4	10	5	6	3	4	8	7	9	5	6	5	5	7
	C_1	6	6	8	9	7	8	9	10	8	9	9	8	8	9
2	C_2	4	4	6	7	5	6	7	8	6	7	7	6	6	7
	C_3	3	3	4	5	4	4	5	6	4	5	5	4	4	5
	n_1	2	2	3	2	2	2	2	3	2	2	2	3	2	2
	n_2	4	3	4	3	4	3	4	5	3	3	4	5	3	3
	R	0.91	0.93	0.87	0.85	0.95	0.97	0.94	0.91	0.96	0.90	0.96	0.85	0.97	0.95
	W	2	9	6	4	5	5	9	6	7	6	6	6	6	6
	C_1	7	6	6	10	8	7	10	11	9	10	10	9	7	10
3	C_2	5	4	4	8	6	5	8	9	7	8	8	7	5	8
	C_3	4	3	3	6	4	4	6	7	5	6	6	5	4	6
	n_1	2	2	3	2	2	2	2	3	2	2	2	3	2	2
	n_2	4	3	4	3	4	3	4	5	3	3	4	5	3	3
	R	0.95	—	0.92	—	—	0.96	—	—	0.91	—	—	0.90	—	0.99
	W	5	—	4	—	—	4	—	—	8	—	—	7	—	9
	C_1	7	—	9	—	—	7	—	—	8	—	—	10	—	11
4	C_2	5	—	7	—	—	5	—	—	6	—	—	8	—	9
	C_3	4	—	5	—	—	4	—	—	4	—	—	6	—	7
	n_1	2	—	3	—	—	2	—	—	2	—	—	3	—	2
	n_2	4	—	4	—	—	3	—	—	3	—	—	5	—	3

Further, the optimal solution of the problem number 15 is presented in Figure 6, and the convergence diagram of GA is presented in Figure 7.

5. Conclusion and further studies

5.1. Conclusion

In this paper, the reliability of a multi-state RAP, in which discounted levels were considered for purchasing components, was investigated using a recursive method. A single-objective cost optimization model was investigated under various constraints including reliability, and since the RAP belonged to NP-hard problems, a GA was used to solve the model. Further, to validate the proposed model, some small-sized problems were solved using an enumeration method and

Table 6. Components' performance rate.

		Component type 1				Component type 2				Component type 3				Component type 4			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Subsystems	1	0	50	100	—	0	25	75	100	0	50	100	—	0	100	—	—
	2	0	25	75	100	0	50	100	—	0	50	100	—	0	—	—	—
	3	0	100	—	—	0	50	100	—	0	25	75	100	0	50	100	—
	4	0	100	—	—	0	50	100	—	0	50	100	—	0	—	—	—
	5	0	50	100	—	0	50	100	—	0	100	—	—	0	—	—	—
	6	0	50	100	—	0	50	100	—	0	100	—	—	0	50	100	—
	7	0	25	75	100	0	100	—	—	0	50	100	—	0	—	—	—
	8	0	50	100	—	0	50	100	—	0	50	100	—	0	—	—	—
	9	0	50	100	—	0	100	—	—	0	50	100	—	0	50	100	—
	10	0	50	100	—	0	50	100	—	0	50	100	—	0	—	—	—
	11	0	50	100	—	0	100	—	—	0	50	100	—	—	—	—	—
	12	0	50	100	—	0	100	—	—	0	50	100	—	0	50	100	—
	13	0	50	100	—	0	50	100	—	0	50	100	—	0	—	—	—
	14	0	25	75	100	0	100	—	—	0	50	100	—	0	50	100	—

Table 7. Correspondence probability of components' performance rate.

		Component type 1				Component type 2				Component type 3				Component type 4			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Subsystems	1	0.1	0.4	0.5	—	0.1	0.2	0.3	0.4	0.1	0.4	0.5	—	0.2	0.8	—	—
	2	0.1	0.2	0.3	0.4	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0	—	—	—
	3	0.2	0.8	—	—	0.1	0.4	0.5	—	0.1	0.2	0.3	0.4	0.1	0.4	0.5	—
	4	0.2	0.8	—	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0	—	—	—
	5	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0.2	0.8	—	—	0	—	—	—
	6	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0.2	0.8	—	—	0.1	0.4	0.5	—
	7	0.1	0.2	0.3	0.4	0.2	0.8	—	—	0.1	0.4	0.5	—	0	—	—	—
	8	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0	—	—	—
	9	0.1	0.4	0.5	—	0.2	0.8	—	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—
	10	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0	—	—	—
	11	0.1	0.4	0.5	—	0.2	0.8	—	—	0.1	0.4	0.5	—	—	—	—	—
	12	0.1	0.4	0.5	—	0.2	0.8	—	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—
	13	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—	0	—	—	—
	14	0.1	0.2	0.3	0.4	0.2	0.8	—	—	0.1	0.4	0.5	—	0.1	0.4	0.5	—

Table 8. The cost and time calculated by the numerical rule and the GA.

	First problem		Second problem	
Solving method	Cost	Time (sec)	Cost	Time (sec)
Enumeration method	34	66.949	42	892.133
GA	34	5.379	42	3.845

Table 9. The reliability and the cost of 15 solved problems.

Problem	Reliability of component type 1 in subsystem 1	System cost	Solving time (second)
1	0.82	282	17.21
2	0.83	299	17.44
3	0.84	287	16.46
4	0.85	306	17.00
5	0.86	301	16.51
6	0.87	285	17.48
7	0.88	288	17.30
8	0.89	291	16.97
9	0.90	287	17.03
10	0.91	284	17.05
11	0.92	290	17.37
12	0.93	289	16.97
13	0.94	289	17.18
14	0.95	292	16.12
15	0.96	290	17.79

$$\begin{matrix} C = 290 \\ w = 33.64 \\ R = 0.9048 \end{matrix} \quad \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 2 & 4 \\ 0 & 0 & 3 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 2 & 4 \\ 1 & 0 & 0 & 1 \\ 1 & 3 & 0 & 3 \\ 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 1 \\ 1 & 0 & 1 & 3 \\ 1 & 0 & 2 & 0 \\ 0 & 2 & 0 & 4 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Figure 6. Optimal solution of the problem number 15.

GA, and it was shown that the GA could reach the optimal solution. Finally, a GA was used to solve 15 large-scale problems.

5.2. Further studies

This study recommends a number of suggestions and adds some insights for future studies in the following:

- Providing multi-objective models for solving real-world problems by considering objectives such as discount levels, weight, volume, etc. in the proposed model;
- Considering incremental discounts instead of all units' discount in the presented model;

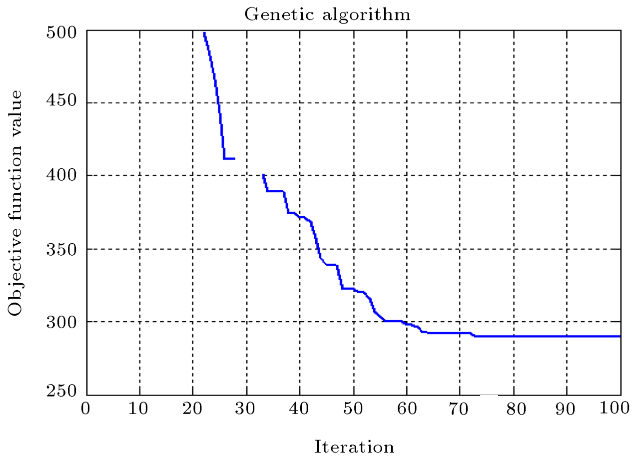

Figure 7. Convergence diagram of GA for the problem no. 15.

- Considering the problem parameters such as reliability, cost, etc. as probabilistic parameters;
- Considering the time-dependent failure rate rather than the constant failure rate;
- Considering repairable components and limits for the number of repairmen to bring the problem closer to real-world conditions.

Acknowledgements

This research was supported by a research project granted by the Qazvin branch, Islamic Azad University. We would like to appreciate the full support of the Deputy of Research of the Qazvin branch, Islamic Azad University for their invaluable assistance.

References

1. Fyffe, D.E., Hines, W.W., and Lee, N.K. "System reliability allocation and a computational algorithm", *IEEE Transactions on Reliability*, **17**(2), pp. 64–69 (1968).
2. Ida, K. "System reliability optimization with several failure modes by genetic algorithm", In *Proc. of 16th International Conf. on Comp. & Indust. Engg.* (1994).
3. Yokota, T., Gen, M., and Ida, K. "System reliability of optimization problems with several failure modes by genetic algorithm", *Japanese Journal of Fuzzy Theory and Systems*, **7**(1), pp. 117–135 (1995).
4. Coit, D.W. and Smith, A.E. "Redundancy allocation to maximize a lower percentile of the system time-to-failure distribution", *IEEE Transactions on Reliability*, **47**(1), pp. 79–87 (1998).
5. Coit, D.W. and Smith, A.E. "Penalty guided genetic search for reliability design optimization", *Computers & Industrial Engineering*, **30**(4), pp. 895–904 (1996).
6. Coit, D.W. and Smith, A.E. "Reliability optimization of series-parallel systems using a genetic algorithm",

IEEE Transactions on Reliability, **45**(2), pp. 254–260 (1996).

7. Coit, D.W. and Smith, A.E. "Optimization approaches to the redundancy allocation problem for series-parallel systems", In *Fourth Industrial Engineering Research Conference Proceedings* (1995).
8. Coit, D.W. "Cold-standby redundancy optimization for nonrepairable systems", *Iie Transactions*, **33**(6), pp. 471–478 (2001).
9. Tavakkoli-Moghaddam, R., Safari, J., and Sassani, F. "Reliability optimization of series-parallel systems with a choice of redundancy strategies using a genetic algorithm", *Reliability Engineering & System Safety*, **93**(4), pp. 550–556 (2008).
10. Tavakkoli-Moghaddam, R. and Safari, J. "A new mathematical model for a redundancy allocation problem with mixing components redundant and choice of redundancy strategies", *Applied Mathematical Sciences*, **45**(1), pp. 2221–2230 (2007).
11. Chambari, A., Najafi, A.A., Rahmati, S.H.A., et al. "An efficient simulated annealing algorithm for the redundancy allocation problem with a choice of redundancy strategies", *Reliability Engineering & System Safety*, **119**, pp. 158–164 (2013).
12. Zaretalab, A., Hajipour, V., Sharifi, M., et al. "A knowledge-based archive multi-objective simulated annealing algorithm to optimize series-parallel system with choice of redundancy strategies", *Computers & Industrial Engineering*, **80**, pp. 33–44 (2015).
13. Li, W. and Zuo, M.J. "Reliability evaluation of multi-state weighted k-out-of-n systems", *Reliability Engineering & System Safety*, **93**(1), pp. 160–167 (2008).
14. Guilani, P.P., et al. "Reliability evaluation of non-reparable three-state systems using Markov model and its comparison with the UGF and the recursive methods", *Reliability Engineering & System Safety*, **129**, pp. 29–35 (2014).
15. Guilani, P.P., Sharifi, M., Niaki, S.T.A., et al. "Redundancy allocation problem of a system with three-state components: A genetic algorithm", *International Journal of Engineering*, **27**(11), pp. 1663–1672 (2014).
16. Soltani, R., Sadjadi, S.J., and Tofigh, A.A. "A model to enhance the reliability of the serial parallel systems with component mixing", *Applied Mathematical Modelling*, **38**(3), pp. 1064–1076 (2014).
17. Garg, H., Rani, M., and Sharma, S. "An efficient two phase approach for solving reliability-redundancy allocation problem using artificial bee colony technique", *Computers & Operations Research*, **40**(12), pp. 2961–2969 (2013).
18. Levitin, G., Xing, L., Ben-Haim, H., et al. "Reliability of series-parallel systems with random failure propagation time", *IEEE Transactions on Reliability*, **62**(3), pp. 637–647 (2013).
19. Maatouk, I., Châtelet, E., and Chebbo, N. "Availability maximization and cost study in multi-state systems", In *Reliability and Maintainability Symposium (RAMS), Proceedings-Annual* (2013).
20. Gago, J., Hartillo, I., Puerto, J., et al. "Exact cost minimization of a series-parallel reliable system with multiple component choices using an algebraic method", *Computers & Operations Research*, **40**(11), pp. 2752–2759 (2013).
21. Ebrahimipour, V. Asadzadeh, S., and Azadeh, A. "An emotional learning-based fuzzy inference system for improvement of system reliability evaluation in redundancy allocation problem", *The International Journal of Advanced Manufacturing Technology*, **66**(9-12), pp. 1–16 (2013).
22. Liu, Y., Huang, H-Z., Wang, Z., et al. "A joint redundancy and imperfect maintenance strategy optimization for multi-state systems", *IEEE Transactions on Reliability*, **62**(2), pp. 368–378 (2013).
23. Khalili-Damghani, K., Abtahi, A.R., and Tavana, M. "A decision support system for solving multi-objective redundancy allocation problems", *Quality and Reliability Engineering International*, **30**(8), pp. 1249–1262 (2014).
24. Sharifi, M., Cheragh, G., Dashti Maljaii, K., et al. "Reliability optimization of a series-parallel k-out-of-n system with failure rate depending on working component of system", *International Journal of Industrial Engineering*, **22**(4), pp. 438–453 (2015).
25. Mousavi, S.M., Alikar, N., Niaki, S.T.A., et al. "Two tuned multi-objective meta-heuristic algorithms for solving a fuzzy multi-state redundancy allocation problem under discount strategies", *Applied Mathematical Modelling*, **39**(22), pp. 6968–6989 (2015).
26. Miriha, M., Niaki, S.T.A., Karimi, B., et al. "Bi-objective reliability optimization of switch-mode k-out-of-n series-parallel systems with active and cold standby components having failure rates dependent on the number of components", *Arabian Journal for Science and Engineering*, **42**(12), pp. 5305–320 (2017).
27. Guilani, P.P., Zaretalab, A., and Niaki, S.T. "A bi-objective model to optimize reliability and cost of k-out-of-n series-parallel systems with tri-state components", *Scientia Iranica, Transactions E, Industrial Engineering*, **24**(3), pp. 1585–602 (2017).
28. Hadipour, H., Amiri, M., and Sharifi, M. "Redundancy allocation in series-parallel systems under warm standby and active components in repairable subsystems", *Reliability Engineering & System Safety*, **192**, 106048 (2019).
29. Pourkarim Guilani, P., Azimi, P., and Sharifi, M. "Redundancy allocation problem with a mixed strategy for a system with k-out-of-n subsystems and time-dependent failure rates based on Weibull distribution: An optimization via simulation approach", *Scientia Iranica*, **26**(2), pp. 1023–1038 (2019).
30. Holland, J.H., *Adaptation in Natural and Artificial Systems. An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence*, Ann Arbor, MI: University of Michigan Press (1975).

Biographies

Mani Sharifi was born in 1973 in Tehran, Iran. He is an Associate Professor at Qazvin Islamic Azad University (QIAU), and holds a BSc degree from Qazvin Islamic Azad University in 1996, an MSc degree from south Tehran branch Islamic Azad University in 1998, and a PhD degree from Tehran Research and Science Islamic Azad University in Industrial Engineering field in 2006. Dr. Sharifi is a member of the Industrial Engineering Department at QIAU. He was the Dean of all research centers of QIAU and the Managerial Editor of "Journal of Optimization in Industrial Engineering". His areas of interest include reliability engineering, combinatorial optimization, statistical optimization, and fuzzy set theory. He has published a number of papers in journals such as Reliability Engineering and Safety System, Computers and Industrial Engineering, and among others.

Majid Saadvandi received an MS degree in Industrial Engineering from Islamic Azad University (IAU), Science and Research Branch, Qazvin, Iran, 2016. His research interests include reliability redundancy allocation problem and fuzzy mathematics.

Mohammadreza Shahriari received his PhD in Operational Research from the Islamic Azad University in 2009. He has conducted extensive research for more than 2 years to find a method for solving a Time-Cost Trade-off problem based on the time value of money, considering each crashing. He has been the Chancellor of Islamic Azad University in Dubai (UAE) and more than one year at the same position at Islamic Azad University in Oxford (UK) for 6 years. He has had another executive academic mandate such as establishing an Iranian University in Germany, Italy, Russia, and Oman. After almost finishing the above-mentioned mandate, he returned to his academic and teaching activities at the university.

Now, he is busy with teaching OR (1, 2, and 3) and advanced OR and reliability at bachelor and master levels and teaching the innovative field of decision theory. He is an Associate Professor at Azad University and is supervising some PhD and master's dissertations in the field of decision theory and reliability theory.