J Ind Eng Int (2017) 13:81–92
DOI 10.1007/s40092-016-0169-3

ORIGINAL RESEARCH

Redundancy allocation problem for *k*-out-of-*n* systems with a choice of redundancy strategies

Mahsa Aghaei[1] · Ali Zeinal Hamadani[1] · Mostafa Abouei Ardakan[2]

Received: 8 December 2014 / Accepted: 11 September 2016 / Published online: 2 November 2016
© The Author(s) 2016. This article is published with open access at Springerlink.com

Abstract To increase the reliability of a specific system, using redundant components is a common method which is called redundancy allocation problem (RAP). Some of the RAP studies have focused on *k*-out-of-*n* systems. However, all of these studies assumed predetermined active or standby strategies for each subsystem. In this paper, for the first time, we propose a *k*-out-of-*n* system with a choice of redundancy strategies. Therefore, a *k*-out-of-*n* series–parallel system is considered when the redundancy strategy can be chosen for each subsystem. In other words, in the proposed model, the redundancy strategy is considered as an additional decision variable and an exact method based on integer programming is used to obtain the optimal solution of the problem. As the optimization of RAP belongs to the NP-hard class of problems, a modified version of genetic algorithm (GA) is also developed. The exact method and the proposed GA are implemented on a well-known test problem and the results demonstrate the efficiency of the new approach compared with the previous studies.

Keywords Redundancy allocation problem · Reliability optimization · Choice of redundancy strategies · *k*-out-of-*n* system

✉ Mahsa Aghaei
mahsa.aghaei@in.iut.ac.ir

[1] Department of Industrial & Systems Engineering, Isfahan University of Technology, Isfahan 84156-83111, Iran

[2] Department of Industrial Engineering, Faculty of Engineering, Kharazmi University, Tehran, Iran

Introduction

Reliability optimization is one of the most important goals in high-tech industries. Redundancy allocation problems (RAP) are an efficient approach for improving system reliability that generally involves the selection of components type and number of redundant components to maximize system reliability under certain constraints. Chern (1992) proved that RAP belongs to the NP-hard class of optimization problems. In general, the redundancy allocation problem formulation has been presented for various systems with different structures and solved by numerous optimization approaches, such as dynamic programming, integer programming, and different meta-heuristic algorithms (Kuo and Prasad 2000; Tillman et al. 1977). Traditionally, there are two general strategies for using redundant components in system. These strategies are called active and standby strategies. Fyffe et al. (1968) formulated the RAP with the active strategy for all subsystems and used a dynamic programming algorithm to solve the problem. Coit (2001) proposed a new formulation for the cold-standby strategy with imperfect switching. An assumption that commonly made in the literature is that the type of redundancy strategy for each subsystem is predetermined and that either the active or the standby strategy may be used for each subsystem. However, Coit (2003) proposed a new formulation of the redundancy allocation problem with a choice of redundancy strategies for each subsystem. He added the type of redundancy strategy as an additional decision variable to the model and used integer programming to solve the problem. He obtained better results when the redundancy strategy could be chosen for each subsystem compared with approaches that consider predetermined strategies. After Coit's study, several studies considered a system with a choice of strategies. Tavakkoli-

Moghaddam et al. (2008), used a genetic algorithm (GA) to solve the same problem. Then, Safari and Tavakkoli-Moghaddam (2010) used a Memetic algorithm for an RAP problem with a choice of strategies. Recently, Chambari et al. (2012) and Safari (2012) extended this model by adding the overall system cost as the second objective and converted the single objective optimization (SOO) problem to a multi-objective optimization (MOO) one. The application of RAP problems with choice of redundancy strategies in these researches shows the advantageous of selecting the best redundancy strategy for each subsystem.

In addition, many studies have been done in the field of k-out-n systems. Wu and Chen (1994), proposed an algorithm for computing the reliability of a weighted k-out-of-n system. Coit and Smith (1996), and used a genetic algorithm to solve a RAP problem in a series–parallel k-out-of-n system with a multiple component choice and an active redundancy strategy. Coit and Liu (2000), considered the redundancy allocation problem in k-out-of-n systems when either the active or the cold-standby redundancy could be used, although it was assumed that the redundancy strategy (i.e., active or cold-standby) for each subsystem was predetermined. A consecutive k-out-of-n system is considered in Cui and Xie (2005). In this research, formulas are derived to calculate the k-out-of-n system reliabilities for both linear and circular cases. Tian and Zuo (2006) applied a GA for the reliability evaluation of a multi-state k-out-of-n system and Li and Zuo (2008) discussed multi-state, weighted k-out-of-n systems. In most of these studies, k is considered to be a fixed and predetermined parameter. However, Sooktip et al. (2011) considered k as a decision variable in the design process. Zhao and Cui (2010) proposed a finite Markov chain imbedding (FMCI) approach for generalized multi-state k-out-of-n systems. Chang et al. (2013) proposed a new method for calculating the reliability of a complex k-out-of-n system and the reliability model is designed based on expandable reliability block diagram. None of these studies in terms of k-out-of-n systems include the choice of strategy in system. Therefore, in this paper, a k-out-of-n system is considered when the redundancy strategy is a decision variable. To solve the proposed model, an exact method based on integer programming is developed to obtain the optimal solution. Besides, a modified GA is extended to solve the model and the results are compared with the previous studies in the literature.

Recently, a lot of new studies have also been devoted to RAPs which lead to significant results. Nematian (2007) considered a special redundancy allocation problem with fuzzy variables. Sardar Donighi and Khanmohammadi (2011) presented a new model which the reliability of components is as fuzzy set. Taghizadeh and Hafezi (2012) investigated the reliability evaluation of available relationships in supply chain which is suitable for computing reliability of supply chain for organizations. Abouei Ardakan and Zeinal Hamadani (2014b) and Ardakan et al. (2015) proposed a new redundancy strategy called mixed strategy which uses both active and cold-standby strategies in one subsystem simultaneously. Abouei Ardakan et al. (2016) extended the mixed strategy for reliability–redundancy allocation problem (RAP). These new studies show that mixed strategy in more powerful than active and standby strategies and leads to higher reliability values. In most of studies in terms of allocation of redundant components, it is assumed that the components are either non-repairable or repairable. Newly some researches considered both repairable and non-repairable components in one subsystem simultaneously (Zoulfaghari et al. 2014, Zoulfaghari et al. 2015a, b). Dolatshahi-Zand and Khalili-Damghani (2015) and (Khalili-Damghani et al. 2013a) considered a multi-objective RAP problem and proposed a multi-objective particle swarm optimization (MOPSO) method to solve it. Khalili-Damghani et al. (2013b) and Khalili-Damghani and Amiri (2012) also proposed an efficient ε-constraint method for solving multi-objective redundancy allocation problems. Then, a data envelopment analysis model is used to prune the non-dominated solutions. Srinivasa Rao and Naikan (2014) combined the Markov approach with system dynamics simulation approach to study the reliability of a repairable system with standby strategy.

As it is clear from the literature, numerous researches have been done in terms of redundancy allocation problem. Some of these studies considered the k-out-of-n systems. However, none of these studies include the choice of strategy in a k-out-of-n system and considered only active or standby strategy in their models. Therefore, in this paper, a k-out-of-n system is considered when the redundancy strategy is a decision variable and we have both active and standby strategies in the model. To solve the proposed model, an exact method based on integer programming is developed to obtain the optimal solution. Besides, a modified GA is extended to solve the model and the results are compared with the previous studies in the literature. The remainder of the paper is organized as follows. In the second section, we describe the system structure and the assumptions of the problem. Third section explains how to compute system reliability in a k-out-of-n system. In the fourth section, the choice of redundancy strategies is described in more detail. Fifth section presents the modeling of the problem and the exact solution for solving the model. In the sixth section, the design of our genetic algorithm is described. Seventh section considers a well-known benchmark problem and the experimental results to demonstrate the efficiency of the proposed methodology. Finally, conclusions are presented in eighth section.

Problem description

In this paper, a non-repairable k-out-of-n system with a choice of redundancy strategies is considered. In a k-out-of-n system, a subsystem operates correctly if at least k out of its n parallel components are operating. A typical structure of such a system is shown in Fig. 1. Since we want to solve the problem by an exact method, we have to consider special design restrictions in practice. Indeed, the model proposed in this study is an extension on Coit (2003) when the strategy adopted for each subsystem is not predetermined, but it is considered as a decision variable to be determined by the model. The assumptions and parameters are almost the same as Coit (2003), namely the type of failure time distribution and no component mixing once a component selection has been made. There are different component choices with different levels of cost, reliability, and weight, as well as other characteristics. Moreover, the redundant components for each subsystem are selected from the same type. The cold-standby strategy is considered with an imperfect switching and an exponential distribution for time-to-failures of components. Thus, the problem involves the simultaneous selection of component type, redundancy level, and the best redundancy strategy for each subsystem to maximize system reliability subject to budget and weight constraints. The assumptions of the model and the notations for a system with S subsystems are mentioned in the following section.

Assumptions

- The components can be in either of two possible states (namely, a good state or a bad state).

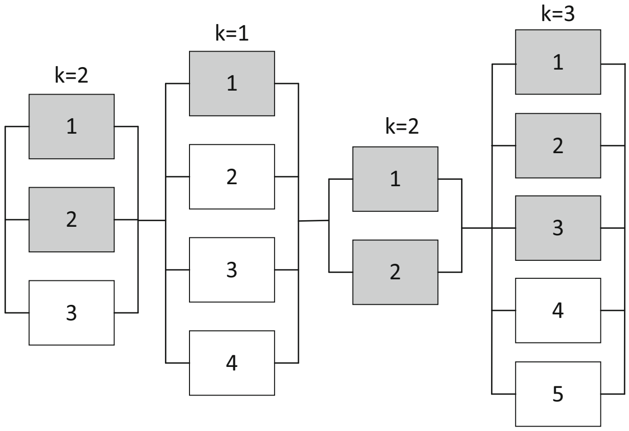

Fig. 1 Series–parallel system with k-out-of-n subsystems

- There are multiple component choices for each subsystem, while one of them should be selected for placing in subsystem.
- The component attributes, such as reliability, weight, and cost, are known and deterministic.
- Failures of components are statistically independent from each other and do not damage the system.
- The types of components used in each subsystem are the same.
- Subsystems with the cold-standby redundancy have an imperfect switching.
- Components and thereby the system are non-repairable.
- Components' time-to-failures follow an exponential distribution.

Notations

i	Index for subsystem
j	Index for component type
m_i	Number of available component choices for subsystem i
n_i	Number of components in subsystem i
n	$= (n_1, n_2, \ldots, n_s),\ n_i \le n_{\max,\mathrm{i}}\ \forall i$
z_i	Type of component selected for subsystem i, $z_i \in \{1, 2,\ldots,m_i\}$
z	$= (z_1, z_2,\ldots,z_s)$
$n_{\max,i}$	Maximum number of components allowed in each subsystem
s	Number of subsystems
t	Mission time
k_i	Minimum number of operating components required per subsystem
w_{ij}	Weight of component j used for subsystem i
c_{ij}	Cost of component j used for subsystem i
C	Cost constraint limit
W	Weight constraint limit
λ_{ij}	Exponential distribution parameter for failure rate of component j used in subsystem i
$\rho_i(t)$	Failure-detection/switching reliability at time t for scenario 1
ρ_i	Failure-detection/switching reliability at time t for scenario 2
$r_{ij}(t)$	Reliability of component j used in subsystem i at time t
A	Set of subsystems with active redundancy
S	Set of subsystems with cold-standby redundancy
N	Set of subsystems which can use active or cold-standby redundancy
$R(t, z, n)$	Overall reliability of system reliability at time t based on vectors z and n
$\tilde{R}(t,z,n)$	Approximation of $R(t, z, n)$

System reliability for *k*-out-of-*n* subsystems

As already mentioned, the objective function of the RAP problem is maximizing system reliability under cost and weight constraints, and is generally formulated as follows:

Problem P1

$$\begin{aligned}&\max R(t,z,n)\\&\text{s.t.}\\&\sum_{i=1}^{s} c_{iz_i} n_i \le C, \qquad n_i \in \{k_i, k_{i+1}, k_{i+2}, \ldots, n_{\max,i}\},\\&\sum_{i=1}^{s} w_{iz_i} n_i \le W, \qquad z_i \in \{1,2,\ldots,m_i\}\end{aligned}$$

As we consider a system with a choice of redundancy strategies, the objective function consists of the reliability of *k*-out-of-*n* system with active and standby strategy. The reliability of a *k*-out-of-*n* system with the active redundancy can be determined using the binominal techniques. The formulation for computing this reliability is as follows:

$$R_i(t) = \sum_{l=k_i}^{n_i} \binom{n_i}{l} (r_{ij}(t))^l (1 - r_{ij}(t))^{n_i - l} \tag{1}$$

where $r_{i,j}$ represents the reliability of component *j* used for subsystem *i*. Since the component time-to-failure distribution is exponential, $r_{i,j}$ is calculated as follows:

$$r_{ij}(t) = \exp(-\lambda_{ij} t). \tag{2}$$

Considering the reliability of cold-standby systems with switching failures is more realistic and important. Coit (2001) described two scenarios for imperfect switching in a cold-standby system. In the first scenario, switching hardware or software controls the system performance continually and activates the redundancy component when a failure occurs in the active component. In the second scenario, failure is only possible when the switch is required, and the probability of success at any time the switch is required is a constant value ρ_i. More details can be found in Coit (2001). Failure-detection/switching reliabilities for the two scenarios are determined by:

Scenarios 1: continuous detection and switching

$$R_i(t) = r_{i,z_i}(t) + \sum_{x=1}^{n_i - 1} \int_0^t \rho_i(u)\, r_{i,z_i}(t-u) f_{i,z_i}^{(x)}(u)\, \mathrm{d}u \tag{3}$$

Scenarios 2: switch activation only in response to a failure

$$R_i(t) = r_{i,z_i}(t) + \sum_{x=1}^{n_i - 1} \rho_i^x \int_0^t r_{i,z_i}(t-u) f_{i,z_i}^{(x)}(u)\, \mathrm{d}u \tag{4}$$

In these equations, $\rho_i(t)$ and ρ_i are the failure-detection/switching reliabilities at time *t* for scenarios 1 and 2, respectively; $r_{i,z_i}(t)$ is the reliability of component *j* used for subsystem *i* at time *t* and $f_{i,z_i}^{(x)}(u)$ is the *pdf* for the *x*th failure of component *j* used for subsystem *i*, i.e., sum of *x* *i.i.d* component failure times. This paper investigates continuous detection and switching (scenario 1).

The reliability of a *k*-out-of-*n* system with the cold-standby redundancy is the probability that there are strictly less than or equal to $n_i - k_i$ failures observed until time *t*. Thus, if the component time-to-failure is exponential, then Eq. (5) can be represented by considering the occurrences of subsystem failures as a homogeneous Poisson process prior to the $n_i - k_i$ failures. In this case, the subsystem failure process is considered as a Poisson process at a rate of $\lambda_{ij} k_i$. Therefore,

$$\int_0^t r_{i,z_i}(t-u) f_{i,z_i}^{(x)}(u)\, \mathrm{d}u = \frac{e^{-\lambda_{i,z_i} k_i t} (\lambda_{i,z_i} k_i t)^x}{x!} \tag{5}$$

In addition, an approximation of the overall system reliability will be as follows:

$$\begin{aligned}\tilde{R}(t,z,n) = &\prod_{i \in A} \sum_{l=k_i}^{n_i} \binom{n_i}{l} (\exp(-\lambda_{i,z_i} t))^l (1 - \exp(-\lambda_{i,z_i} t))^{n_i - l}\\&\times \prod_{i \in S} \exp(-\lambda_{i,z_i} k_i t) + \rho_i(t) \exp(-\lambda_{i,z_i} k_i t)\\&\times \sum_{l=0}^{n_i - k_i} \frac{(\lambda_{i,z_i} k_i t)^l}{l!}\end{aligned} \tag{6}$$

As mentioned in Coit (2001), it is difficult to determine a closed form for equations similar to Eq. (3). Therefore, an estimation for lower bound of Eq. (3) is determined in the equation above, because $\rho_i(t) \le \rho_i(u)$ for all $u \le t$.

Choice of redundancy strategies

As previously mentioned, when the redundancy strategy for subsystems is not predetermined, the best strategy should be selected among the standby and the active redundancies by comparing their reliabilities. Coit (2003) explained the relationship between the reliabilities of cold-standby and active redundancies for an individual subsystem. When there is a perfect switching and the cold-standby switching is not exposed to operating stresses, the cold-standby strategy is always preferable to the active one. With imperfect switching, for component *j* with special characteristics (switching reliability, time-to-failure distribution), there is a maximum redundancy level, n'_{ij}, where the reliability of the standby strategy is greater or equal to that of the active strategy. In other words, if the number of redundant components used in subsystem i is less than or equal to the maximum level ($n_i \leq n'_{ij}$), the cold-standby outperforms the active strategy; otherwise, the active strategy will have a better performance for all larger values of $n_i > n'_{ij}$ (as shown in Fig. 2).

To validate this conclusion, assume that the reliability of a component is 0.7408 and that of the switching for standby components is 0.99. Now suppose that the system includes a certain number of components in parallel. The following figure illustrates the reliability of the system for different values of n_i with the standby or active strategy.

Clearly, n'_{ij}, in this example, is equal to four. As previously mentioned, with the cold-standby strategy, we expect reliability to reduce with increasing number of redundant components and this is due to the increased requirements of imperfect switching.

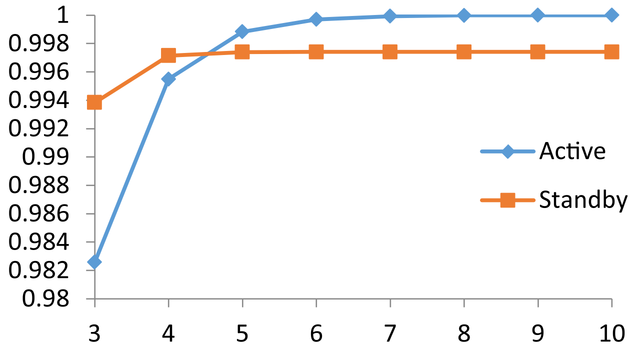

Fig. 2 Comparison of active and standby redundancies

Exact solution methodology

For obtaining optimal solution of the proposed model, an integer programming-based method is presented. The bases of this approach were proposed by Misra and Sharma (1973). It is based on transforming the problem by taking the logarithm of Eq. (6) to develop an equivalent problem. The solution methodology is described as the following five steps:

1. First, we define y_{ijp} as the new zero–one decision variables as follows:

$$y_{ijp} = \begin{cases} 1, & p \text{ numbers of component type } j \\ & \quad \text{will place in subsystem } i \\ 0, & \text{otherwise} \end{cases}$$

2. Determine sets of A, S, and C based on the requirements of system design and identify the set each subsystem belongs to. If a subsystem belongs to set C, but its switching detection is perfect, transfer this subsystem from set C to set S.
3. Compute the value of n'_{ij} for all $i \in c$ and $j = 1, \ldots, m_i$. n'_{ij} demonstrating the maximum level where the standby redundancy still has a higher reliability than the active one.

$$n'_{ij} = \sup\Big\{n_{ij}; \exp(-\lambda_{i,z_i} k_i t) + \rho_i(t)\exp(-\lambda_{i,z_i} k_i t) \times \sum_{l=0}^{n_i-k_i} \frac{(\lambda_{i,sz_i} k_i t)^l}{l!} > \sum_{l=k_i}^{n_i} \binom{n_i}{l} (\exp(-\lambda_{i,z_i} t))^l (1-\exp(-\lambda_{i,z_i} t))^{n_i-l}$$

4. Compute α_{ijp}, β_{ijp}, and γ_{ijp} as problem parameters,

$$\alpha_{ijp} = c_{ij} p \quad for \quad 1 \leq i \leq s, \quad 1 \leq j \leq m_i, \quad 1 \leq p \leq n_{\text{max,i}}$$
$$\beta_{ijp} = w_{ij} p \quad for \quad 1 \leq i \leq s, \quad 1 \leq j \leq m_i, \quad 1 \leq p \leq n_{\text{max,i}}$$

$for\ \mathrm{i} \in A$

$$\gamma_{ijp} \begin{cases} -\lambda_{ij} k_i t & p = k_i \\ \ln\left(\sum_{l=k_i}^{n_i} \binom{n_i}{l} (\exp(-\lambda_{ij} t))^l (1-\exp(\lambda_{ij} t))^{n_i-l}\right) & k_i p \leq n_{\max,i} \end{cases}$$

$for\ \mathrm{i} \in S$

$$\gamma_{ijp} \begin{cases} -\lambda_{ij} k_i t & \mathrm{p} = \mathrm{k}_i \\ -\lambda_{ij} k_i t + \ln\left(\sum_{l=0}^{n_i-k_i} \frac{(\lambda_{ij} k_i t)^l}{l}\right) & \mathrm{k}_i < \mathrm{p} \leq n_{\text{max,i}} \end{cases}$$

$for\ \mathrm{i} \in C$

$$\gamma_{ijp} = \begin{cases} -\lambda_{ij} k_i t & \mathrm{p} = \mathrm{k}_i \\ -\lambda_{ij} k_i t + \ln\left(\sum_{l=0}^{n_i-k_i} \frac{(\lambda_{ij} k_i t)^l}{l!}\right) & \mathrm{k}_i < \mathrm{p} \leq n'_{ij} \\ \ln\left(\sum_{l=k_i}^{n_i} \binom{n_i}{l} (\exp(-\lambda_{ij} t))^l (1-\exp(\lambda_{ij} t))^{n_i-l}\right) & n'_{ij} < \mathrm{p} \leq n_{\text{max,i}} \end{cases}$$

5. Problem $P2$ is the linear form of problem $P1$ and belongs to the zero–one integer programming problems. The following problem can be solved by any convenient branch-and-bound or cutting plane algorithm.

Problem P2

$$\max \sum_{i=1}^{s} \sum_{j=1}^{m_i} \sum_{p=k}^{n_{max,i}} \gamma_{ijp} y_{ijp}$$

s.t.

$$\sum_{i=1}^{s} \sum_{j=1}^{m_i} \sum_{p=k}^{n_{max,i}} \alpha_{ijp} y_{ijp} \leq c$$
$$\sum_{i=1}^{s} \sum_{j=1}^{m_i} \sum_{p=k}^{n_{max,i}} \beta_{ijp} y_{ijp} \leq w$$
$$\sum_{j=1}^{m_i} \sum_{p=k}^{n_{max,i}} y_{ijp} = 1$$
$$y_{ijp} \in \{0,1\}.$$

Obtain the optimal solution of the above model. The results show that S numbers (subsystem index) of variable y_{ijp} are equal to one in the optimal solution and that the remainder is equal to zero. Therefore, the system structure will be as follows:

for $\mathrm{y}_{ijp} = 1$,

$$\begin{cases} \text{if} \quad \mathrm{i} \in A,\ p \text{ numbers of component } j \text{ will place in} \\ \quad \text{subsystem } i \text{ with active redundancy} \\ \text{if} \quad \mathrm{i} \in S,\ p \text{ numbers of component } j \text{ will place in} \\ \quad \text{subsystem } i \text{ with standby redundancy} \\ \text{if} \quad \mathrm{i} \in N \text{ and } p \le \mathrm{n}'_{ij},\ p \text{ numbers of component } j \text{ will} \\ \quad \text{place in subsystem } i \text{ with standby redundancy} \\ \text{if} \quad \mathrm{i} \in N \text{ and } p > \mathrm{n}'_{ij},\ p \text{ numbers of component } j \text{ will} \\ \quad \text{place in subsystem } i \text{ with active redundancy.} \end{cases}$$

It should be noted that in a k-out-of-n subsystem with p components in parallel and a cold-standby strategy, there are k_i active components in the active state and $p - k_i$ components in the cold-standby state.

Using this solution method for a large-scale problem will increase the number of variables to a too large set, so that the exact method would not be useful in this situation and it is hard to obtain optimal solution. It may be necessary to use other approaches for very large problems. Therefore, in addition to the existing solution, a genetic algorithm has been used to solve this model in the numerical example section. First, the structure of an appropriate genetic algorithm for this model is described in the following section.

Genetic algorithm implementation

Genetic algorithm (GA) is a well-known stochastic search method and belongs to the larger class of evolutionary algorithms (EA) which solve optimization problems using techniques inspired by natural selection in biological evolution. It was first popularized by Holland. It has successfully been applied to solve different system reliability optimization problems. The GA is described by the following features:

1. Solutions encoding
2. Generation of an initial population
3. Selection of parent solutions for breeding
4. Crossover operator
5. Mutation operator
6. Combine the offspring and former population and culling of the best solutions
7. Repeat steps 3 through 6 until termination criteria is satisfied.

Solution encoding

For this problem, each possible solution is a collection of redundancy strategies, type of selected components, and number of components in parallel for each subsystem. The solution encoding (chromosome) is presented as a $3 \times s$ matrix, where s denotes the number of subsystems. The first, second, and third rows of this matrix represent the number of components, type of selected component, and the selected redundancy strategy for the subsystems, respectively. Figure 3 presents an example of encoding solution for this problem with $s = 14$. This figure demonstrates a solution, in which the first subsystem ($s = 1$) uses active redundancy with two components of the second type in parallel, while the last subsystem uses standby redundancy strategy with three components of the fourth type in parallel.

Initial population

At first, the minimum appreciate population size should be selected in accordance to problem size. For a given population size (p), the initial population is generated by selecting p chromosomes randomly. To produce each solution of the current population, s integers between k_i and $n_{\max}$ were randomly selected to represent the number of components in parallel (n_i) for each subsystem. Types of components for each subsystem were randomly selected from among the m_i available components. Then, redundancy strategy (active or standby) is selected for each subsystem.

Crossover

Parents are selected from existing population at random in each iteration, then the proposed crossover operator is applied and two offspring will be generated from each two selected parents. Four crossover operators are applied to generate offspring from parents, namely, single-point crossover, double-point crossover, max–min crossover, and uniform crossover. In the double-point crossover, two points along the parent chromosomes are selected randomly, all the genes between the two points are swapped between the parent chromosomes, and then, two new

$$\begin{array}{r} \text{number of components} \\ \text{type of component} \\ \text{redundancy strategy} \end{array} \begin{bmatrix} 2 & 3 & 2 & 4 & 1 & 2 & 2 & 4 & 3 & 4 & 3 & 1 & 2 & 3 \\ 2 & 3 & 3 & 1 & 1 & 1 & 2 & 1 & 2 & 3 & 1 & 1 & 2 & 4 \\ A & S & S & S & S & S & A & S & S & S & S & S & S & S \end{bmatrix}$$

Fig. 3 Chromosome representation (solution encoding)

offspring are produced (Fig. 4). In the max–min crossover, the subsystems with the highest and lowest reliabilities of parents are specified and all the genes of each selected subsystem are exchanged with genes of the same subsystem in the other parent (Fig. 5). The uniform crossover operator is a powerful crossover, because it enables to random recombination of different genes in parents. So, by considering mixing ratio of 0.5, offspring genes are selected from the first or from the second parent randomly.

Mutation

After the crossover operation, mutation is performed to maintain the diversity of solution space and to avoid getting stuck at a local optimum. Two mutation operators are employed, namely, simple mutation and max–min mutation. In the simple mutation with rate of p_M, for each candidate solution, a random value is selected for each gene and the gene value is altered if the random value is smaller than the mutation rate. The max–min mutation operator was developed by Tavakkoli-Moghaddam et al. (2008). In a system with several subsystems, the subsystem with the minimum reliability has a bad effect in reliability of overall system. Therefore, max–min mutation operator selects subsystems with the highest reliability and the lowest reliability among the all subsystems and then mutates the gene values of each selected subsystem randomly. A typical max–min mutation is shown in Fig. 6.

Fitness function

To define the fitness function, a penalty approach is used for the proposed genetic algorithm to penalize infeasible solutions and to reduce their fitness values in proportion to their degrees of constraint violation. In other words, a value proportional to the constraints violation is added to the objective function as a dynamic penalty function, which results in adding a relatively large amount of penalty to the objective functions, if one solution violates a constraint. This penalty provides the feasibility of the final solution while keeping the efficient search through the infeasible region. It is important to search through the infeasible space of the problem, because it leads to reach appropriate diversity for the proposed GA and good feasible solutions can most efficiently be reached by breeding between a feasible and an infeasible solution.

Stopping criteria

The termination criterion for the proposed GA is a preselected number of generations.

Illustrative example

The example employed in this study is adopted from the example provided by Fyffe et al. (1968). The system consists of 14 subsystems with ($t = 100$) for mission time, ($C = 130$) for system cost constraint, ($W = 170$) for system weight constraint, ($n_{\max} = 6$) for maximum number of allowed components in each subsystem, and three or four component choices for each subsystem. Our specific example is, however, different from Fyffe's in a number of ways, namely, the redundancy strategy (active or standby) is considered to be selected for each subsystem and the exponential distribution is chosen for time-to-failure of components. Failure detection for the standby redundancy follows scenario 1 and the reliability of a switching (at 100 h) is 0.99 for each subsystem. $k_i = 1$ was considered in the original example by Fyffe et al. (1968), whereas its values in the present example are randomly selected for each subsystem. It should be emphasized that the results are highly dependent on k_i values; so the results would be fundamentally different as a result of changes in k_i. In addition, it has been assumed that all the subsystems pertain to set C. The data are presented in Table 1.

Exact solution

Initially, exact solution methodology is used to solve the example. To find the optimal solution, the model can be solved by any standard algorithm for zero–one integer

$$\begin{array}{r}\text{number of components}\\ \text{type of component}\\ \text{redundancy strategy}\end{array}\begin{bmatrix} 2 & 3 & 2 & 4 & 1 & 2 & 2 & 4 & 3 & 4 & 3 & 1 & 2 & 3 \\ 2 & 3 & 3 & 1 & 1 & 1 & 2 & 1 & 2 & 3 & 1 & 1 & 2 & 4 \\ A & S & S & S & S & S & A & S & S & S & S & S & S & S \end{bmatrix} \Rightarrow \begin{bmatrix} 2 & 3 & 2 & 3 & 2 & 2 & 2 & 2 & 3 & 4 & 3 & 3 & 2 & 3 \\ 2 & 3 & 3 & 3 & 1 & 2 & 1 & 3 & 2 & 3 & 1 & 2 & 2 & 4 \\ A & S & S & S & A & A & A & S & S & S & S & S & S & S \end{bmatrix}$$

$$\uparrow \qquad\qquad \uparrow \qquad\qquad\qquad\qquad \uparrow \qquad\qquad \uparrow$$

$$\begin{array}{r}\text{number of components}\\ \text{type of component}\\ \text{redundancy strategy}\end{array}\begin{bmatrix} 3 & 2 & 2 & 3 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 1 & 2 & 4 \\ 2 & 2 & 1 & 3 & 1 & 2 & 1 & 3 & 3 & 4 & 3 & 2 & 2 & 3 \\ S & A & A & S & A & A & A & S & A & S & A & S & A & A \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 2 & 2 & 4 & 1 & 2 & 2 & 4 & 3 & 3 & 3 & 1 & 2 & 4 \\ 1 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 3 & 4 & 3 & 2 & 2 & 3 \\ S & A & A & S & S & S & A & S & A & S & A & S & A & A \end{bmatrix}$$

$$\uparrow \qquad\qquad \uparrow \qquad\qquad\qquad\qquad \uparrow \qquad\qquad \uparrow$$

Fig. 4 Double-point crossover operator

$$\begin{array}{rcccc}
\begin{array}{r}\text{number of components}\\ \text{type of component}\\ \text{redundancy strategy}\end{array} &
\left[\begin{array}{cccccccccccccc}2&3&2&4&1&2&2&4&3&4&3&1&2&3\\2&3&3&1&1&1&2&1&2&3&1&1&2&4\\A&S&S&S&S&S&A&S&S&S&S&S&S&S\end{array}\right] & \Rightarrow &
\left[\begin{array}{cccccccccccccc}2&3&2&4&1&2&2&4&3&4&3&3&2&3\\2&3&3&1&1&1&2&1&2&3&2&2&2&4\\A&S&S&S&S&S&A&S&S&S&S&S&S&S\end{array}\right]\\
& \begin{array}{cccccccccccccc}\uparrow&&&&&&&&&&\uparrow&&&\\ \text{max}&&&&&&&&&&\text{min}&&&\end{array} & &
\begin{array}{cccccccccccccc}\uparrow&&&&&&&&&&\uparrow&&&\end{array}
\end{array}$$

$$\begin{array}{rcccc}
\begin{array}{r}\text{number of components}\\ \text{type of component}\\ \text{redundancy strategy}\end{array} &
\left[\begin{array}{cccccccccccccc}3&2&2&3&2&2&2&2&3&3&3&1&2&4\\2&2&1&3&1&2&1&3&3&4&3&2&2&3\\S&A&A&S&A&A&A&S&A&S&A&S&A&A\end{array}\right] & \Rightarrow &
\left[\begin{array}{cccccccccccccc}3&2&2&3&2&2&2&2&3&3&3&1&2&4\\1&2&1&3&1&2&1&3&3&4&3&2&2&3\\S&A&A&S&A&A&A&S&A&S&A&S&A&A\end{array}\right]\\
& \begin{array}{cccccccccccccc}\uparrow&\uparrow&&&&&&&&&&&&\\ \text{max}&\text{min}&&&&&&&&&&&&\end{array} & &
\begin{array}{cccccccccccccc}\uparrow&\uparrow&&&&&&&&&&&&\end{array}
\end{array}$$

Fig. 5 Max–min crossover operator

$$\begin{array}{rcccc}
\begin{array}{r}\text{number of components}\\ \text{type of component}\\ \text{redundancy strategy}\end{array} &
\left[\begin{array}{cccccccccccccc}2&3&2&4&1&2&2&4&3&4&3&1&2&3\\2&3&3&1&1&1&2&1&2&3&1&1&2&4\\A&S&S&S&S&S&A&S&S&S&S&S&S&S\end{array}\right] & \Rightarrow &
\left[\begin{array}{cccccccccccccc}3&3&2&4&1&2&2&4&3&4&3&1&2&3\\2&3&3&1&1&1&2&1&2&3&2&1&2&4\\S&S&S&S&S&S&A&S&S&S&S&S&S&S\end{array}\right]\\
& \begin{array}{cccccccccccccc}\uparrow&&&&&&&&&&\uparrow&&&\\ \text{max}&&&&&&&&&&\text{min}&&&\end{array} & &
\begin{array}{cccccccccccccc}\uparrow&&&&&&&&&&\uparrow&&&\end{array}
\end{array}$$

Fig. 6 Max–min mutation operator

Table 1 Component data for the illustrative example

i	k_i	Choice 1 ($j = 1$)			Choice 2 ($j = 2$)			Choice 3 ($j = 3$)			Choice 4 ($j = 4$)		
		λ_{ij}	c_{ij}	w_{ij}	λ_{ij}	c_{ij}	w_{ij}	λ_{ij}	c_{ij}	w_{ij}	λ_{ij}	c_{ij}	w_{ij}
1	1	0.001054	1	3	0.000726	1	4	0.000943	2	2	0.000513	2	5
2	2	0.000513	2	8	0.000619	1	10	0.000726	1	9	–	–	–
3	1	0.001625	2	7	0.001054	3	5	0.001393	1	6	0.000834	4	4
4	2	0.001863	3	5	0.001393	4	6	0.001625	5	4	–	–	–
5	1	0.000619	2	4	0.000726	2	3	0.000513	3	5	–	–	–
6	2	0.000101	3	5	0.000202	3	4	0.000305	2	5	0.000408	2	4
7	1	0.000943	4	7	0.000834	4	8	0.000619	5	9	–	–	–
8	2	0.002107	3	4	0.001054	5	7	0.000943	6	6	–	–	–
9	3	0.000305	2	8	0.000101	3	9	0.000408	4	7	0.000943	3	8
10	3	0.001863	4	9	0.001625	4	5	0.001054	5	6	–	–	–
11	3	0.000619	3	5	0.000513	4	6	0.000408	5	6	–	–	–
12	1	0.002357	2	4	0.001985	3	5	0.001625	4	6	0.001054	5	7
13	2	0.000202	2	5	0.000101	3	5	0.000305	2	6	–	–	–
14	3	0.001054	4	6	0.000834	4	7	0.000513	5	6	0.000101	6	9

programming. The results are presented in Table 2. The optimal solution yields reliability equal to 0.4505 for the system. This solution has also been compared with those for two other problems. In the first problem, only the active redundancy is considered. In the second problem, redundancy strategy (i.e., active or standby) is predetermined for each subsystem. This problem is in accordance with the example solved in Coit and Liu (2000), but the exponential distribution is used instead of Erlang distribution for time-to-failure of components. The exponential distribution parameters are chosen, so that the components have the same reliability as in the original example (for $t = 100$) and our answer is quite equal to the result obtained in Coit and Liu (2000). The numerical results indicate that the choice of strategies led to a higher reliability compared with the other approaches.

The performance of the proposed approach is evaluated by implementing on 33 test problems provided by Tavakkoli-Moghaddam et al. (2008). In this case, the value of cost constraint is fixed (130), while the weight constraint is varied from 159 to 191. Table 3 presents a comparison between the best results obtained by solving 33 problems in this paper and that reported in Coit and Liu (2000). In Table 3, as the weight constraint gradually increases, the optimal solution (best reliability) improves. Furthermore, maximum possible improvement (MPI) index is also used to measure the significance of the improvements made using the choice of strategies as compared with the

Table 2 Example results

i	Choice of strategies (this paper)			Mixed of strategies (Coit and Liu 2000)			Active strategy	
	z_i	n_i	Redundancy	z_i	n_i	Redundancy	z_i	n_i
1	3	2	Standby	3	2	Active	3	2
2	1	2	Active	1	2	Active	1	2
3	4	1	Active	4	1	Active	4	1
4	3	3	Standby	3	3	Active	3	3
5	2	1	Active	2	1	Active	2	1
6	2	2	Active	2	2	Active	2	2
7	2	1	Active	2	1	Active	2	1
8	1	3	Standby	1	3	Standby	1	3
9	3	3	Active	3	3	Standby	3	3
10	2	4	Standby	2	4	Standby	2	4
11	1	4	Standby	1	4	Standby	1	4
12	1	2	Standby	1	2	Standby	1	2
13	2	2	Active	2	2	Standby	2	2
14	3	4	Standby	3	4	Standby	3	4
System reliability	0.4505*			0.4417			0.4105	
System cost	118			118			118	
System weight	170			170			170	

* The best reliability value

previous best-known strategies. This index, which has been used in many previous studies, including Yeh and Hsieh (2011), Wu et al. (2011) and Abouei Ardakan and Zeinal Hamadani (2014a), is given by:

$$\mathrm{MPI}(\%) = [R_s(\text{New Approach}) - R_s(\text{Other})]/[1 - R_s(\text{Other})]. \qquad (7)$$

Here, R_s(NewApproach) represents the best system reliability obtained by the proposed approach and R_s(Other) represents the best system reliability obtained by any other method reported in the literature. Table 3 indicates that the choice of strategies used for the redundant components led to the improvements in the reliability levels of all the 33 benchmark problems compared to method which considered predetermined strategies. Therefore, this study provides better computational results than Coit and Liu (2000). Figure 7 presents the MPI index in all test problems.

Genetic algorithm

From the 33 test problems, seven problems are selected and the GA with penalty function which is discussed in "Genetic algorithm implementation" section is implemented on them. Because of the stochastic nature of GA, ten runs are performed for each problem and each run is terminated after 100 generations. The best and the worst solution amongst them are reported in Table 4. In all problems which are shown in Table 4, maximum reliability produced by the proposed GA is the same as optimal solution obtained in the previous section and it shows the good performance of our GA approach. In the primary instances with lower reliability, the improvement is small, but in the next instances, the improvement becomes larger. However, in high-reliability applications, even very small improvement in the reliability is often difficult to obtain. In all problems, we obtain system reliability higher than the previously studies.

Conclusion

This paper investigated the redundancy allocation problem for a k-out-of-n system with a choice of redundancy strategies. In contrast to the existing approaches that often consider a predetermined strategy for each subsystem, we considered both active and standby strategies and developed a model to select the best strategy for each subsystem. The choice of redundancy strategies is more realistic and will be more successful for implementing. To solve the proposed model, we developed an exact method to obtain the optimal solution of the problem. The main advantage of this methodology is that a complex non-linear problem is reduced to an equivalent simple problem in the form of an integer-linear programming. Furthermore, a modified version of GA is developed to solve the problem is large size or other tough situation. To evaluate the efficiency of the proposed strategy, a well-known benchmark problem was

Table 3 Computational results between the proposed approach and Coit and Liu (2000) for the 33 test problems

Problem	Weight constraint	This paper (reliability)	Coit and Liu (2000) (reliability)	MPI
1	159	0.3302	0.3238	0.9464
2	160	0.3374	0.3308	0.9862
3	161	0.3537	0.3479	0.8894
4	162	0.3576	0.3517	0.9100
5	163	0.3867	0.3792	1.2081
6	164	0.3910	0.3833	1.2485
7	165	0.3995	0.3917	1.2822
8	166	0.4145	0.4057	1.4807
9	167	0.4190	0.4102	1.4920
10	168	0.4282	0.4191	1.5665
11	169	0.4456	0.4369	1.5450
12	170	0.4505	0.4417	1.5762
13	171	0.4603	0.4513	1.6402
14	172	0.4776	0.4675	1.8967
15	173	0.4829	0.4726	1.9529
16	174	0.4934	0.4829	2.0305
17	175	0.4983	0.4874	2.1264
18	176	0.5170	0.5049	2.4439
19	177	0.5227	0.5105	2.4923
20	178	0.5341	0.5216	2.6128
21	179	0.5355	0.5242	2.3749
22	180	0.5491	0.5362	2.7813
23	181	0.5580	0.5450	2.8571
24	182	0.5672	0.5539	2.9813
25	183	0.5797	0.5661	3.1343
26	184	0.5861	0.5723	3.2265
27	185	0.5988	0.5848	3.3718
28	186	0.6021	0.5880	3.4223
29	187	0.6244	0.5976	6.6600
30	188	0.6346	0.6073	6.9518
31	189	0.6450	0.6173	7.2380
32	190	0.6476	0.6197	7.3363
33	191	0.6498	0.6346	7.7996

Fig. 7 Maximum possible improvement (MPI) index values

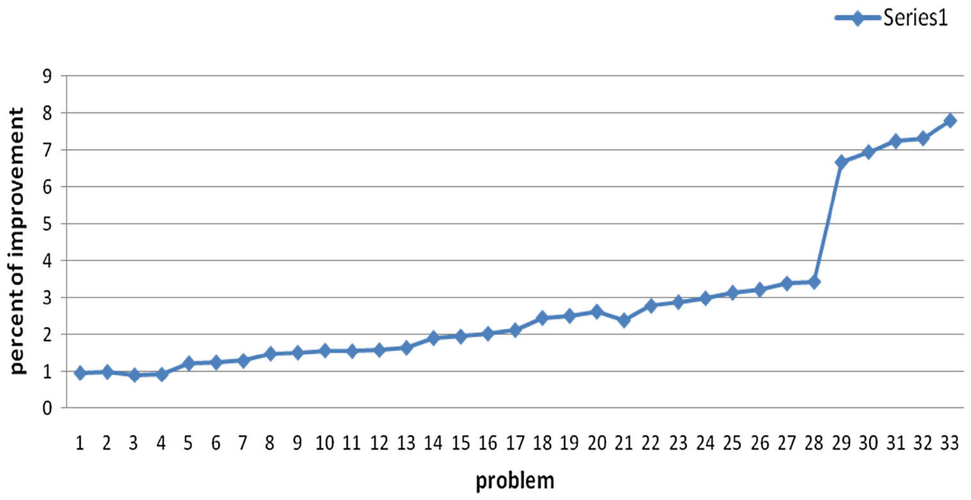

Table 4 Computational results of GA performance over ten trials for 33 test problems

Problem	Weight constraint	Coit and Liu (2000)	This paper (optimal solution)	Proposed GA		MPI
				Max Reli.	Min Reli.	
1	159	0.3238	0.3302	0.3302	0.30851	0.9464
2	165	0.3917	0.3995	0.3995	0.38322	1.2822
3	170	0.4417	0.4505	0.4505	0.43172	1.5762
4	175	0.4874	0.4983	0.4983	0.48354	2.1264
5	180	0.5362	0.5491	0.5491	0.52655	2.7813
6	185	0.5848	0.5988	0.5988	0.56895	3.3718
7	191	0.6346	0.6498	0.64985	0.62645	7.7996

considered. The results demonstrate considerable improvements in the reliability using the new approach. For future studies, one can focus on developing a bi-objective model to optimize this problem. In this model, one redundancy strategy is used for each subsystem. Another interesting extension may be the case where both active and standby strategies are simultaneously used in the subsystems. Furthermore, because RAPs belong to NP-hard problems for which it is not easy to obtain optimal solutions, researchers can test other meta-heuristic approaches for this problem.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

Abouei Ardakan M, Zeinal Hamadani A (2014a) Reliability–redundancy allocation problem with cold-standby redundancy strategy. Simul Model Pract Theory 42:107–118

Abouei Ardakan M, Zeinal Hamadani A (2014b) Reliability optimization of series–parallel systems with mixed redundancy strategy in subsystems. Reliab Eng Syst Saf 130:132–139

Abouei Ardakan M, Sima M, Zeinal Hamadani A, Coit DW (2016) A novel strategy for redundant components in reliability–redundancy allocation problems. IIE Trans, (accepted), 00–00

Ardakan MA, Hamadani AZ, Alinaghian M (2015) Optimizing bi-objective redundancy allocation problem with a mixed redundancy strategy. ISA Trans 55:116–128

Chambari A, Rahmati SHA, Najafi AA (2012) A bi-objective model to optimize reliability and cost of system with a choice of redundancy strategies. Comput Ind Eng 63:109–119

Chang Z-Z, Zhao G-Y, Sun Y-F (2013) A calculation method for the reliability of a complex *k*-out-of-*n* system. In: 2013 international conference on quality, reliability, risk, maintenance, and safety engineering (QR2MSE). IEEE, New York, pp 204–207

Chern M-S (1992) On the computational complexity of reliability redundancy allocation in a series system. Oper Res Lett 11:309–315

Coit DW (2001) Cold-standby redundancy optimization for nonrepairable systems. IIE Trans 33:471–478

Coit DW (2003) Maximization of system reliability with a choice of redundancy strategies. IIE Trans 35:535–543

Coit DW, Liu JC (2000) System reliability optimization with *k*-out-of-*n* subsystems. Int J Reliab Qual Saf Eng 7:129–142

Coit DW, Smith AE (1996) Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Trans Reliab 45(254–260):266

Cui L, Xie M (2005) On a generalized *k*-out-of-*n* system and its reliability. Int J Syst Sci 36(5):267–274

Dolatshahi-Zand A, Khalili-Damghani K (2015) Design of SCADA water resource management control center by a bi-objective redundancy allocation problem and particle swarm optimization. Reliab Eng Syst Saf 133:11–21

Fyffe DE, Hines WW, Lee NK (1968) System reliability allocation and a computational algorithm. IEEE Trans Reliab 17:64–69

Khalili-Damghani K, Amiri M (2012) Solving binary-state multi-objective reliability redundancy allocation series-parallel problem using efficient epsilon-constraint, multi-start partial bound enumeration algorithm, and DEA. Reliab Eng Syst Saf 103:35–44

Khalili-Damghani K, Abtahi A-R, Tavana M (2013a) A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems. Reliab Eng Syst Saf 111:58–75

Khalili-Damghani K, Abtahi AR, Tavana M (2013b) A decision support system for solving multi-objective redundancy allocation problems. Qual Reliab Eng Int 30(8):1249–1262

Kuo W, Prasad VR (2000) An annotated overview of system-reliability optimization. IEEE Trans Reliab 49:176–187

Li W, Zuo MJ (2008) Optimal design of multi-state weighted *k*-out-of-*n* systems based on component design. Reliab Eng Syst Saf 93:1673–1681

Misra K, Sharma J (1973) Reliability optimization of a system by zero–one programming. Microelectron Reliab 12:229–233

Nematian J (2007) Fuzzy reliability optimization models for redundant systems. J Ind Eng Int 4:1–9

Safari J (2012) Multi-objective reliability optimization of series-parallel systems with a choice of redundancy strategies. Reliab Eng Syst Saf 108:10–20

Safari J, Tavakkoli-Moghaddam R (2010) A redundancy allocation problem with the choice of redundancy strategies by a memetic algorithm. J Ind Eng Int 6:6–16

Sardar Donighi S, Khanmohammadi S (2011) A fuzzy reliability model for series-parallel systems. J Ind Eng Int 7:10–18

Sooktip T, Wattanapongsakorn N, Coit D (2011) System reliability optimization with constraint *k*-out-of-*n* redundancies. In: 2011 IEEE international conference on quality and reliability (ICQR). IEEE, New York, pp 216–220

Srinivasa Rao M, Naikan VNA (2014) Reliability analysis of repairable systems using system dynamics modeling and simulation. J Ind Eng Int 10:1–10. doi:10.1007/s40092-014-0069-3

Taghizadeh H, Hafezi E (2012) The investigation of supply chain's reliability measure: a case study. J Ind Eng Int 8:1–10. doi:10.1186/2251-712x-8-22

Tavakkoli-Moghaddam R, Safari J, Sassani F (2008) Reliability optimization of series–parallel systems with a choice of redundancy strategies using a genetic algorithm. Reliab Eng Syst Saf 93:550–556

Tian Z, Zuo MJ (2006) Redundancy allocation for multi-state systems using physical programming and genetic algorithms. Reliab Eng Syst Saf 91:1049–1056

Tillman FA, Hwang C-L, Kuo W (1977) Optimization techniques for system reliability with redundancy: a review. IEEE Trans Reliab 26:148–155

Wu J-S, Chen R-J (1994) An algorithm for computing the reliability of weighted-*k*-out-of-*n* systems. IEEE Trans Reliab 43:327–328

Wu P, Gao L, Zou D, Li S (2011) An improved particle swarm optimization algorithm for reliability problems. ISA Trans 50:71–81

Yeh W-C, Hsieh T-J (2011) Solving reliability redundancy allocation problems using an artificial bee colony algorithm. Comput Oper Res 38:1465–1473

Zhao X, Cui L (2010) Reliability evaluation of generalised multi-state *k*-out-of-*n* systems based on FMCI approach. Int J Syst Sci 41:1437–1443

Zoulfaghari H, Zeinal Hamadani A, Abouei Ardakan M (2014) Bi-objective redundancy allocation problem for a system with mixed repairable and non-repairable components. ISA Trans 53:17–24

Zoulfaghari H, Hamadani A, Ardakan M (2015a) Multi-objective availability-redundancy allocation problem for a system with repairable and non-repairable components. Decis Sci Lett 4:289–302

Zoulfaghari H, Hamadani AZ, Ardakan M. A (2015b) Multi-objective availability optimization of a system with repairable and non-repairable components. In: 2015 international conference on industrial engineering and operations management (IEOM), IEEE, New York, pp 1–6