Reliability Optimization of a k-out-of-n Series-Parallel System with Warm Standby Components

Mani Sharifi 1,2, *, Mohammadreza Shahriari 3, Ahmad Khajehpoor 2, Seyed Ali Mirtaheri 2,

1 Department of Mechanical and Industrial Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada,
2 Faculty of Industrial & Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran,
3 Faculty of Management & Accounting, South Tehran Branch, Islamic Azad University, Tehran, Iran,
* Corresponding Author: manisharifi@ryerson.ca

Abstract: In this research, a new hybrid model for the redundancy allocation problem (RAP) in a series-parallel configuration with the k-out-of-n subsystem is presented. The redundancy policy is set to an active, warm standby, or no redundancy in the given model. In warm standby policy, an imperfect switch detected the component's failure and replaced the fail component with a new standby. So, the subsystems' redundancy policy is one of the model's decision variables. We presented a new objective function for the RAP to calculate the reliability of a system that consists of active and warm standby subsystems. The presented model aims to determine the subsystems' redundancy policy, the type and number of redundant components to maximize the system's reliability under the system's cost, volume, and weight constraints. To solve the proposed model, we used two Genetic Algorithm (GA) and a hybrid GA (HGA) meta-heuristic algorithm with local search. Since the %RPD of HGA is 2.1% (on average) better than GA in solving ten large-scale instances, the result shows the superiority of HGA compared to GA for solving the presented RAP.

Keywords: Redundancy allocation problem, warm standby, reliability, meta-heuristic methods, imperfect switch,

1- Introduction

Due to the market being competitive, it needs to have a more reliable design in recent decades. Nowadays, the term reliability includes reliability requirements, reliability design, reliability prediction, reliability modeling, and retrievals. One of the goals of reliability is designing the systems with higher quality during its life cycle. Usually, the system's reliability improves through improving the reliability of each component or allocating redundant components. This improvement in practice happens by using better materials, better manufacturing processes, or better design principles. Many research methods have been conducted in reliability improvement according to the system structure, problem type, resolve method, objective function, and components' failure distribution of the components [1]. The system's structure can be series, parallel,
k-out-of-n [2], and/or the combination of series and parallel [3]. The system’s reliability may improve by redundancy allocation [3] or reliability allocation [4]. Solving methods include the exact techniques [2, 5], approximate methods [6, 7], heuristic methods [8], and meta-heuristic methods [9-12]. The Objective function of the redundancy allocation problem (RAP) is usually considered as maximizing the system’s reliability [3] and minimizing the system’s cost [13]. The components’ failure rates can be considered and constant (Exponential distribution) [14], or time-dependent (i.e., Weibull distribution) [15]. In this paper, we worked on a RAP series-parallel system’s structure and k-out-of-subsystems’ configuration. In this paper, the presented RAP aims to optimize the number and type of the redundant components in each subsystem as well as the redundancy strategy of each subsystem to maximize the system’s reliability under some constraints.

The RAP is divided into two categories based on the allocated redundant components to the subsystems: RAP without component mixing (RAPCM) and RAP with a Mix of Components (RAPMC). The subsystems’ redundancy strategy includes active and standby, and the standby policy has three different types, based on the components' characteristics: cold standby, warm standby, and hot standby. Misra and shamara [16] considered the RAP for a series-parallel system with the k-out-of-n subsystem. In their model, they considered the active redundancy policy without component mixing. They solved the presented model with binary integer programming. Coit and smith [17] offered a new model for RAPMC and an active redundancy policy. They considered the series-parallel system’s structure with a k-out-of-n subsystem.

Coit and Liu [18] presented a new RAPCM model for a series-parallel system with k-out-of-n subsystems. In their model, they considered active and cold standby redundancy policies simultaneously, for the first time. They assumed the components with constant failure rate (CFR) and a non-linear model and converted the model to a binary integer programming using variable change. Coit [19] presented a new model in which the redundancy policy was considered a model’s decision variable. The variable redundancy policy was active, cold standby, and no redundancy. This paper considered the hot standby systems components to draw the problem near real-world conditions. Since the RAP in computational time is Np-hard problems, we solved the presented model using the meta-heuristic method. A comparative search upon recent research (after 2010) related to RAP is shown in Table 1.

| Insert Table 1 here |

In this paper, we aim to fill the literature gap by considering the warm standby redundancy strategy for a RAPCM. The contribution of the current research is as follows:

- Calculating the system reliability with warm standby components,
- Considering warm standby redundancy strategy for a RAPCM,

The current research methodology is presented in Figure 1.
The rest of the paper is as follows: Section 2 is the problem definition. Section 3 deals with calculating the system's reliability and subsystems' reliability with active, warm standby, and no redundancy strategy. In Section 3, the solving methodologies are presented. In Section 5, firstly, some instances are solved to determine the algorithms' performance. Then the effect of changing on model’s parameters is investigated using a sensitivity analysis. Finally, the model and algorithm are validated. Section 6 is the conclusion and further studies.

2- Model description

This section discusses the mathematical model of a RAP with a series-parallel Structure and k-out-of-n Subsystem. The identical components can be allocated to each subsystem, and the redundancy strategy is the system's variable. In the presented model, the problem's objective function is to maximize the system's reliability under the system’s cost, volume, and weight constraints.

2-1- Assumptions

The mathematical model of the above-mentioned RAP is established based on the following assumptions.

- Two active and warm standby redundancy policies are considered for each subsystem,
- Different components type is available to allocate to each subsystem,
- All the allocated components to each subsystem must be the same,
- Cost, weight, volume, and reliability of the components are constant and pre-defined,
- Components are binary state and have two working or failed states,
- And imperfect switch detects and replace the failed components,
- Components are non-repairable,
- Components are CFR and failed independently,

The first assumption is the main novelty of the current research, which fills the literature review gap.

2-2- Nomenclatures

\( i \): Subsystems' index \( (i = 1, \ldots, s) \)
\( n_i \): Number of allocated components to subsystem \( i \),
\( j \): Index of the allocated components to each subsystem, \( (j = 1, \ldots, n_i ) \)
\( m_i \): Number of available components type for subsystem \( i \),
\( z_i \): Index of component's type which is allocated to subsystem \( i \), \( z_i = (1, \ldots, m_i) \)
\( R(t) \): System's reliability at time \( t \) depending on design vectors \( z \) and \( n \)
\( k_i \): Minimum required number of components for subsystem \( i \),
Mathematical model

Based on the presented assumption for the paper, the mathematical RAP model is as follows:

\[
Max R(t) = \prod_{i=1}^{s} R_i(t, z_i, n_i, k_i) 
\]

\[s.t: \]
\[
\sum_{i=1}^{s} c_{i,x_i} n_i \leq C \]  
\[
\sum_{i=1}^{s} v_{i,x_i} n_i \leq V \]  
\[
\sum_{i=1}^{s} w_{i,x_i} n_i \leq W \]  
\[
n_i \in \{ k_i, ..., n_{max,i} \} \]  
\[
z_i \in \{1, ..., m_i \} \]

Equation (1) is the model’s objective function, which aims to maximize system reliability. A description of calculating the system’s reliability will present in the next section. Equations (2) to (4) are the system cost, volume, and weight constraints subsequently. Equation (5) determines the minimum and maximum allocated components to each subsystem, and finally, Equation (6) defines the different available components’ type for each subsystem.

3- Calculation of the system’s Reliability
If only \( k \) component allocates to a subsystem, the subsystem has no redundancy strategy. If more than \( k \) components allocate to a subsystem, the subsystem may have an active or hot standby redundancy strategy. In this case, the Reliability of the Subsystem depends on its redundancy strategy. The subsystems’ reliability with active and warm standby strategies is presented in subsection 3-1 and 3-3, respectively.

### 3-1- Subsystems’ reliability with active redundancy

Reliability of a k-out-of-n Subsystem with active redundancy when the components are identical and independent is computed using standard techniques. Therefore, the Reliability of \( i \) th Subsystem with active redundancy is calculated as follows:

\[
R_i (t) = \sum_{l=k_i}^{n_i} \binom{n_i}{l} (e^{-\lambda_i z_i t})^l (1 - e^{-\lambda_i z_i t})^{n_i-l} \tag{7}
\]

Assume that \( n_i \) components of type \( z_i \) are allocated to the subsystem \( i \). In Equation (7), \( e^{-\lambda_i z_i t} \) is the reliability of the component, and \( (e^{-\lambda_i z_i t})^l \) is the probability than \( l \) components are working during the mission horizon \( t \). Besides, \( (1 - e^{-\lambda_i z_i t}) \) is the failure probability of the component fails, and \( (1 - e^{-\lambda_i z_i t})^{n_i-l} \) is the probability than \( (n_i - l) \) components are failed during the mission horizon \( t \).

### 3-2- Subsystems’ reliability with no redundancy

If the model allocates \( k \) components to a k-out-of-n subsystem, all \( k \) components should start working at the beginning of the mission horizon, and the subsystem has no standby component(s). Therefore, the subsystem has no redundancy strategy. In this case, the subsystem stops working when the first components’ failure happens. So, the reliability of the subsystem \( i \), with \( n_i \) components of type \( z_i \) calculates as follow:

\[
R_i (t) = (e^{-\lambda_i z_i t})^{n_i} = (e^{-\lambda_i z_i t})^{k_i} = e^{-k_i \lambda_i z_i t} \tag{8}
\]

### 3-3- Subsystems’ reliability with warm standby redundancy

She and Pecht [53] were calculated the Reliability of a k-out-of-n Warm-Standby System. In their model, the switching system was perfect. In this paper, a discrete imperfect switch detects the component’s failure and replaces the failed one with a new one on standby (if it is available). The success probability for each detection and replacement is equal \( \rho_i \). She and Pecht [53] divided the warm standby reliability formula into two parts: fixed coefficients (c-part) and below the integral (I-part).

- **C-part**

  The switch starts its function when one of the working components fails, and at least one component is available on standby. When one of the \( k_i \) working components fail, the switch failure probability is added to the system’s probability function. But when one of them \( (n_i - k_i) \) on the standby component fails, there is no switch failure probability. Besides, when the system has \( k_i \) working components, and no component
on standby, the switch failure probability is not added to the system's probability function. So, the C-part calculates as follows:
\[
C - \text{Part} = \left[ \binom{k_i}{1} \rho_i \lambda a_{t,\text{st}} + \binom{n_i}{k_i} \lambda d_{t,\text{st}} \right] \times \left[ \binom{k_i}{1} \rho_i \lambda a_{t,\text{st}} + \binom{n_i}{k_i} \lambda d_{t,\text{st}} \right] \times \ldots
\]
\[
\times \left[ \binom{k_i}{1} \rho_i \lambda a_{t,\text{st}} + \binom{1}{1} \lambda d_{t,\text{st}} \right] \times \left[ \binom{k_i}{1} \lambda a_{t,\text{st}} \right]
\]
\[
= \left[ \binom{k_i}{1} \lambda a_{t,\text{st}} \right] \prod_{i=1}^{n_i-k_i} (\rho_i k_i \lambda a_{t,\text{st}} + i \lambda d_{t,\text{st}})
\]

- **I-part**

She and Pecht [53] calculate the I Part as follows:
\[
I - \text{Part} = \int_{\tau}^{\infty} \left[ \prod_{i=1}^{t} \int_{t_i}^{t} e^{-k_i \lambda a_{t,\text{st}} t - \sum_{j=1}^{k_i} \lambda d_{t,\text{st}} t_j} \prod_{j=1}^{n_i-k_i} dt_j \right] dt
\]

With simplification and Integration, I-part is simplified as follows:
\[
I - \text{Part} = \int_{\tau}^{\infty} e^{-k_i \lambda a_{t,\text{st}} t} \left[ \sum_{i=0}^{n_i-k_i} (-1)^i \frac{e^{-i \lambda d_{t,\text{st}} t}}{i! (n_i - k_i - i)! \lambda d_{t,\text{st}}^{n_i-k_i}} \right] dt
\]

Finally, with the integration of Equation (11), I-Part is obtained as follows:
\[
I - \text{Part} = \frac{1}{\lambda d_{t,\text{st}}^{n_i-k_i}} \sum_{i=0}^{n_i-k_i} \frac{(-1)^i}{i! (n_i - k_i - i)! (k_i \lambda a_{t,\text{st}} + i \lambda d_{t,\text{st}})} e^{-(k_i \lambda a_{t,\text{st}} + i \lambda d_{t,\text{st}}) t}
\]

Now with multiply the C-part and I-part (Equations (9) and (12)), the Subsystem’s Reliability calculates as follows:
\[
R_i(t) = (C - \text{part}) \times (I - \text{part}) \rightarrow R_i(t)
\]
\[
= \left\{ \binom{k_i}{1} \lambda a_{t,\text{st}} \right\} \prod_{i=1}^{n_i-k_i} (\rho_i k_i \lambda a_{t,\text{st}} + i \lambda d_{t,\text{st}}) \frac{1}{\lambda d_{t,\text{st}}^{n_i-k_i}}
\]
\[
\times \sum_{i=0}^{n_i-k_i} \frac{(-1)^i}{i! (n_i - k_i - i)! (k_i \lambda a_{t,\text{st}} + i \lambda d_{t,\text{st}})} e^{-(k_i \lambda a_{t,\text{st}} + i \lambda d_{t,\text{st}}) t}
\]

### 3-4- System’s Reliability

In a series-parallel system’s structure, the subsystems are connected serially, when in each subsystem, the components are parallel. So, the system’s reliability calculates by multiplying the subsystems’ reliabilities as follows:
\[ R(t) = \prod_{i \in A} \left\{ \sum_{l=k_i}^{n_i} \left( \binom{n_i}{l} (e^{-\lambda_{i,l} t})^l \left( 1 - e^{-\lambda_{i,l} t} \right)^{n_i-l} \right) \right\} \]
\[ \times \prod_{i \in S} \left\{ \sum_{l=k_i}^{n_i-k_i} \left( \binom{n_i-k_i}{l} \lambda_{i,l,x} \prod_{l=1}^{n_i-k_i} (\rho_l k_i \lambda_{i,l} + i \lambda_{i,l}) \right) \cdot \frac{1}{\lambda_{i,l,x} (n_i-l)} \right\} \times \]
\[ \times \prod_{i \in N} e^{-k_i \lambda_{i,l,x} t} \]

\[ (14) \]

4- Solving Methods

Chern [54] proved that RAP belongs to Np. Hard category of problems, so we used two metaheuristic algorithms to solve the presented model. The first algorithm is the Genetic Algorithm (GA), a wild application for addressing the RAP (Table 1). The second one is a hybrid GA (HGA), which combines the GA with a local search to improve the GA’s performance.

4-1- Genetic Algorithm

The genetic algorithm has a wide range of applicability in different engineering optimization problems. This algorithm is a population-based algorithm that starts from an initial population and with the inspiration of the natural genetic moves to the global optimal solution. GA begins with a set of solutions called the initial population (initial generation), shown through the chromosome structure. Then generate the next generation, using some operators like crossover, mutation, and elitism. The new generations at least have the characteristics of the previous generation. The pseudo-code of the proposed GA is presented in Figure 2.

4-1-1. Solution encoding

Each solution (chromosome) of the presented model is codded as a \(3 \times s\) matrix [55]. In this chromosome, \(s\) in the number of system’s subsystems. The first, second, and third rows of the chromosome represent the type of redundancy strategy, type of selected components, and the number of allocated components to each subsystem. In this chromosome and the first row, three choices are available as \(A\): active strategy, \(S\): standby strategy, and \(N\): No redundancy strategy. The values of the second row of the chromosome vary from 1 to \(m_i\) \((i = 1, ..., s)\), and the values of the third row varies from \(k_i\) to \(n_{\text{max},i}\). A sample of chromosomes for a system with 14 subsystems shows in Figure 3.

7
As is presented in Figure 3, the first subsystem’s redundancy strategy is active, and two components of type 3 allocate to the subsystem.

4-1-2. Initial population
The initial population is generated randomly.

4-1-3. Fitness function
The objective function of the presented model is maximizing the system’s reliability. Since the initial population is generated randomly, some of the generated chromosomes are not feasible. We used a penalty function to give a better chance to the feasible solutions for the algorithm’s operators. The fitness function of the model is presented in Equation (15) as follow:

\[ F = \frac{R}{(b \times pf)} \]  

(15)

In Equation (15), \( F \) is the chromosome’s fitness function, \( R \) is the chromosome’s reliability, and \( pf \) is the penalty function. The value of \( pf \) depend on the cost, volume, and weight of the chromosome and calculates as follows:

\[ Pf = \prod_{i=1}^{3} pf_i = pf_1 \times pf_2 \times pf_3 \]  

(16)

\[ pf_1 = \max \left( \frac{\sum_{i=1}^{k} c_i z_i n_i}{c}, 1 \right) \]  

(17)

\[ pf_2 = \max \left( \frac{\sum_{i=1}^{k} v_i z_i n_i}{v}, 1 \right) \]  

(18)

\[ pf_2 = \max \left( \frac{\sum_{i=1}^{k} w_i z_i n_i}{w}, 1 \right) \]  

(19)

For a chromosome, if all constraints are satisfied, the chromosome is feasible and \( pf_1 = pf_2 = pf_3 = 1 \) and the value of fitness function are equal to the chromosome’s reliability. But if at least one of the constraints is not satisfied, the chromosome is not feasible. So \( pf_1 \times pf_2 \times pf_3 > 1 \) and the value of the chromosome’s fitness function is less than its reliability.

4-1-4. Parents selection strategy
We used a roulette wheel selection strategy for selecting the parents for the operators. This method gives more chances to the chromosomes with better fitness function.

4-1-5. Crossover operator
We used the uniform crossover in this research. In this type of crossover operator, first, we select two chromosomes using a roulette wheel. We then generate a random chromosome whose genomes have a binary random value (0 or 1). The size of the random chromosome is equal to the size of the problem's chromosomes. For each genome of the chromosome, if the genome’s value is equal to one, the correspondence genome of the parents replaces with each other. The crossover procedure shows in Figure 4.

Insert Figure 4 here

4-1-6. Mutation operator

For mutation, one parent is selected using the roulette wheel. Then we generate a random chromosome which its genomes have a real random value between 0 and 1. For each genome, if the genome's value is less than a pre-defined value (mutation rate), the corresponding genome in the parent chromosome will mutate. For the first row of the chromosome, the parent's genome is equal to \( N, A, \) or \( S \). For mutation, each genome will change randomly to two other redundancy strategies. For example, if the redundancy strategy is \( A \), it will be changed randomly to \( N \), or \( S \). For the second and third rows of the chromosome, the genome's value will be increased or decreased one unit randomly. Figure 5 shows the procedure of the mutation operator.

Insert Figure 5 here

4-1-7. The algorithm’s stopping criteria

A pre-defined maximum generation Is the Algorithm’s stopping criteria.

4-2- HGA with adaptive local search

The GA performs a random search within all feasible and insensible solutions. In many problems, most of the time, a considerable part of the random initial populations are not feasible. Since one of the most critical factors in GA to find an optimal (or near-optimal) solution is the quality of the initial population, using a random initial population decreases the chance of finding the right answers. To elimination these weaknesses, many different methods combine with GA. One of these methods is a local search algorithm that leads the reliability optimization problems to a better result (Tavakoli-Moghadam and Safari [55]). The local search is a technique to search near the generated random solution to find the potential better solutions, so it improves the GA's performance. Yun [56] presented the adaptive local search, which searched for the solutions’ neighborhood in each iteration of the GA. Using the adaptive local search, decreasing the local solution trap in GA and leads GA to the optimal global solution. In this paper, we present the HGA with an adaptive local search for solving the presented RAP.

4-2-1. Adaptive local search scheme
The adaptive local search which we applied in this paper uses the average fitness function values of two consecutive generations as follows

\[ F_{vr}(g) = \frac{Af_v(g)}{Af_v(g-1)} \] (20)

\[
\begin{cases}
    \text{if } F_{vr}(g) > 1 : & \text{Applying GA with local search in the iteration,} \\
    \text{if } F_{vr}(g) \leq 1 : & \text{Applying only GA in the iteration}
\end{cases}
\] (21)

In Equation (20), \( Af_v(g) \) is the Average fitness function values of the best population, based on elitist selection strategy at generation \( g \), \( Af_v(g-1) \) is the Average fitness function values of the best population, based on elitist selection strategy at generation \( (g-1) \), and \( F_{vr}(g) \) is the fitness function value ratio at generation \( g \).

4-2-2. HGA with local search

In this proposes HGA, we used the hill-climbing (HL) local search method. Firstly, we apply the HL local search for each of the chromosomes selected by the elitist selection strategy for the next generation. The new chromosomes are then obtained from the HL local search algorithm, replaced by the old chromosomes, and moves to the next generation. The HL local search algorithm includes the following steps:

Step 1: Select one of the chromosomes that are selected by elitist selection strategy for the next generation,

Step 2: Randomly generate some neighborhoods of the selected chromosomes, and calculate their fitness function. The number of generated chromosomes' neighborhoods is equal to the problem’s population size.

Step 3: Select the neighborhood with the best fitness function.

Step 4: If the fitness function of the neighborhood which is selected in Step 3 is better than the fitness function of the selected chromosome, replace the chromosome by neighborhood, and go to Step 2,

Step 5: Repeat Steps 1 to 4 for all chromosomes selected by the elitist selection strategy.

How to generate solution encoding, generate the initial population, parents' selection mechanism, calculate fitness function, perform the crossover and mutation operators, selection strategy of the next generation, and stop condition are precise as the presented GA. The pseudo-code of the proposed HL local search is presented in Figure 6.

| Insert Figure 6 here |

4-3- Parameters’ tuning

The results of the metaheuristic algorithms depend on the input parameters. So, we used the response surface methodology [57] for algorithms’ parameters tuning. The range of the algorithms’ parameters is presented in Table 2.
In Table 2, `popsize` defines the algorithms' population size, `p_c` is the crossover probability, `p_m` is the mutation probability, `b` is the penalty constant, and `maxgen` is the maximum number of algorithms' generations. The optimal values for both algorithms are presented in Table 3.

5- Numerical analysis

In this section, firstly, we solve ten different instances to have a comparison between metaheuristics. Then the effect of changing the parameters of the objective functions is investigated in the sensitivity analysis section. Next, the model and algorithms are validated comparing with other researches. Finally, some managerial insights are presented.

5-1- Numerical example

For comparison of the proposed algorithms, we used a numerical instance presented by Fyffe et al. [3]. The instance contains a system with a k-out-of-n series-parallel structure and 14 subsystems. In each subsystem, three or four different components' type is available. Other instance parameters are presented in Table 4. The switch success probability is equal to 0.999, and the mission horizon is 100 hours. The maximum number of components for each subsystem is six, and the constraints' right-hand sides are equal to \( C = 130, V = 110, \) and \( W = 170. \) The number of unique solutions to the problem is \( 7.996 \times 10^{23}. \)

The proposed GA and HGA are both coded with MatLab R2019b. The result of GA and HGA are presented in Tables 5 and 6.

The results of Tables 5 and 6 show the superiority of the HGA in comparison to GA. To better compare these two algorithms, we selected ten problems within the 33 problems presented by Nakagawa and Miyazaki [5] and solve them using both algorithms. These problems are quite similar to the solved instance except that the weight constraint (Right-hand side of the weight constraint) varies from 166 to 175. Each algorithm is run five times, and then we report the best, the average, and standard
deviation of the system’s reliability within these runs. The results for these ten instances are presented in Table 7, and Table 8 shows the %PDA of the algorithms.

The result of %PDA in Table 8 shows that HGA has better performance for best-case and average-case results for all instances. The best-case and average-case results of GA are %2.41 and %2.1 (on average) less than HGA, respectively.

To illustrate the significant differences among the results obtained by the proposed HGA and the GA, a two-sample T-test is performed using Minitab 17, and the result is presented in Table 9 and Figure 7.

These results prove that the HGA algorithm is preferred at a confidence level of 95%. The difference between the obtained results of both algorithms under the statistical test presented in Equation (22) is investigated. Table 9 shows the results of the T-test for the above comparison. The $P-value = 0.000$ indicates that the difference between these two algorithms is significant. The following typical test of the hypothesis is performed after normalizing the data.

\[
\begin{align*}
\mu_{HGA} &= \mu_G \\
\mu_{HGA} &\neq \mu_G
\end{align*}
\] (22)

The box-plot has shown in Figure 7 also supports a significant difference between the mean of the results obtained from the HGA algorithm and the GA algorithm.

5-2- Sensitivity analysis

For sensitivity analysis, different values for $C$, $W$, and $V$ are considered to investigate the effect of changing these parameters on the optimal system’s reliability. Since the HGA has the superiority in solving the instances, we only solve sensitivity analysis instances using HGA. Moreover, we consider that the maximum allocatable components to each subsystem are equal to 4.

Regarding the system’s cost ($C$), the system’s weight and volume constraints are relaxed, and increase the value of $C$ increases from 130 to 220 by steps of 10. The results are presented in Table 10.
In Table 10, the system with $C = 130$ is considered the main system, and for other values of $C$, the changes are highlighted as **bold** and **underlined** letters and numbers. When the value of $C$ increases, firstly, the model allocates more components to the subsystems with minimum allocated components (i.e., the subsystem with $n = k$). The model increases the number of allocated components to each subsystem. When $C = 180$, all subsystems as four components, which is the maximum allocatable component to each subsystem. In this case, the redundancy strategy of all subsystems is changed to warm standby. After that, by increasing the value of $C$ from 190 to 220, only the types of the components were changed. By increasing the value of $C$ from 130 to 220, the system’s reliability increases from 0.5039 to 0.7643, which shows a $51.77\%$ increase.

Regarding the system’s weight ($W$), the system’s cost and volume constraints are relaxed, and increase the value of $W$ increases from 170 to 350 by steps of 20. The results are presented in Table 11.

In Table 11, the system with $W = 170$ is considered the main system, and for other values of $W$, the changes are highlighted as **bold** and **underlined** letters and numbers. When the value of $W$ increases, the model allocates more components to the subsystems with minimum allocated components (i.e., the subsystem with $n = k$). The model increases the number of allocated components to each subsystem. When $C = 290$, all subsystems are four components, which is the maximum allocatable component to each subsystem. In this case, the redundancy strategy of all subsystems is changed to warm standby. After that, by increasing the value of $W$, the model allocates the components with better performance. Thus by an increase in the value of $W$ from 290 to 350, only the types of the components were changed. By increasing the value of $W$ from 170 to 290, the system’s reliability increases from 0.4403 to 0.7626, which shows a $73.20\%$ increase.

Regarding the system’s volume ($V$), the system’s cost and weight constraints are relaxed, and increase the value of $V$ increases from 110 to 200 by steps of 20. The results are presented in Table 12.

In Table 12, the system with $V = 110$ is considered the main system, and for other values of $V$, the changes are highlighted as **bold** and **underlined** letters and numbers. When the value of $V$ is equal to 110, the system allocates the components with the highest performance to each subsystem. So, by increasing the value of $V$, the components’ type doesn’t change, and only the number of allocated components to each subsystem increase. By increasing the value of $V$ from 110 to 180, the system’s reliability increases from 0.6286 to 0.7741, which shows a $23.14\%$ increase.
The results of the sensitivity analysis demonstrated that the system is more sensitive to the value of \( W \), then of the value of \( C \), and finally on the value of \( V \).

5-3- Model’s and algorithms’ validation

For model validation, we relaxed the volume constraint and reduced the switch success probability to 0.99. Then we multiply the values of the components’ warm standby failure rate by \( \gamma \) and reduces the value of \( \gamma \) from one to zero by steps of 0.2. changing the value of \( \gamma \) does not affect the number and type of the allocated components to each subsystem as well as the redundancy strategy of each subsystem. Only the value of the system’s reliability increases smoothly as we expected. The system’s reliability for different values for \( \gamma \) is presented in table 13.

| \( \gamma \) | System’s Reliability |
|-------------|----------------------|
| 0.5         | 0.4505               |
| 0.7         | 0.4700               |
| 0.8         | 0.4850               |
| 0.9         | 0.4950               |
| 1.0         | 0.5000               |

The system’s reliability for \( \gamma = 0 \) is equal to 0.4505. when the value of \( \gamma \) is equal to zero. The standby components' failure rates are equal to zero, so the model is turned to a system with cold standby components. The result for \( \gamma = 0 \) in terms of the subsystems’ allocated components, type of the allocated components to each subsystem, redundancy strategy of the subsystems, and the system’s reliability is the same as the results of Aghaei et al. [58]. It shows the presented RAP’s ability to deal with warm and cold standby components and demonstrates the solving methodologies are precisely designed.

Moreover, Table 13 shows that the presented model is applicable to cold and warm standby components simultaneously. For this reason, and for the subsystems with cold standby components, the warm standby failure rates should be set to zero.

5-4- Managerial insights

The presented model will help the managers and system designers optimize the redundant systems in terms of reliability when the components are warm. Using the results of the presented models leads the managers to operate the systems with lower cost and system designers to a beneficial trade-off between the system’s reliability and cost. The systems that use warm standby components like batteries (i.e., UPSs) and radioactive components (i.e., Nuclear power plant and nuclear submarines) and the electricity transmission systems may use the result of the presented model to design and operate more reliable systems.

The results of Tables 10-12 show that increasing the right-hand-side of the model’s constraints firstly leads the model to allocate more components to the subsystems. Then the models use the components with higher performance to increase the system’s reliability. It means that the model is more sensitive to the number of subsystems’ components than their type.
6- Conclusions and recommendations for future research

In most of the research conducted on RAP, the subsystems’ components are considered a cold standby. But in real-world systems such as UPSs, nuclear power plants, and nuclear submarines, the components are warm standby. So, it is essential to present the new practical models to figure out these systems’ reliability. This paper presents a new HGA for solving the RAPCM with k-out-of-n subsystems’ configuration and warm standby components with CFR. In this model, the redundancy of the subsystems was considered as the model’s decision variable. Since the proposed model is an Np. Hard non-linear programming model, we solve the presented model with an HGA and compare the results with a genetic algorithm. The results show the superiority of the HGA compared to GA, and HGA reaches results in average 2.1% better than GA in terms of the system’s reliability for ten different large-scale problems. Moreover, the results show that the model is more sensitive to the number of the allocated components to the subsystem compared to the type of the allocated components. By changing the values of the warm standby components’ failure rates, we showed that the presented model is applicable for the systems with cold and warm standby components simultaneously.

Future studies may have two directions. The first direction deal with the model's assumptions. Considering the systems with repairable components makes the problem more realistic. Besides, the structure of the current research may apply to a RAPMC. Finally, considering the multi-state warm standby components is a proper way to draw the problem near real-world conditions. The second direction is using different solving methodologies. Considering the multi-objective RAP with the current assumptions brings more options for decision-makers.

7- References

[1] Kuo, W., Prasad, V. R., Tillman, F. A., et al. “Optimal reliability design: fundamentals and applications”, Cambridge university press (2001).

[2] Pham, H. “Optimal design of k-out-of-n redundant systems”, Microelectronics Reliability, 32(1-2), pp. 119-126 (1992).

[3] Fyffe, D. E., Hines, W. W., & Lee, N. K. “System reliability allocation and a computational algorithm”, IEEE Transactions on Reliability, 17(2), pp. 64-69 (1968).

[4] Yalaoui, A., Chu, C., & Chatellet, E. “Reliability allocation problem in a series-parallel system”, Reliability engineering & system safety, 90(1), pp. 55-61 (2005).

[5] Nakagawa, Y., & Miyazaki, S. “Surrogate constraints algorithm for reliability optimization problems with two constraints”, IEEE Transactions on Reliability, 30(2), pp. 175-180 (1981).

[6] Tillman, F. A., Hwang, C. L., & Kuo, W. “System effectiveness models: an annotated bibliography”, IEEE Transactions on Reliability, 29(4), pp. 295-304 (1980).
[7] Prasad, V. R., & Kuo, W. “Reliability optimization of coherent systems”, IEEE Transactions on Reliability, 49(3), pp. 323-330 (2000).

[8] You, P. S., & Chen, T. C. “An efficient heuristic for series-parallel redundant reliability problems”, Computers & operations research, 32(8), pp. 2117-2127 (2005).

[9] Coit, D. W., & Smith, A. E. “Reliability optimization of series-parallel systems using a genetic algorithm”. IEEE Transactions on Reliability, 45(2), pp. 254-260 (1996).

[10] Kulturel-Konak, S., Smith, A. E., & Coit, D. W. “Efficiently solving the redundancy allocation problem using tabu search”, IIE transactions, 35(6), pp. 515-526 (2003).

[11] Kim, H. G., Bae, C. O., & Park, S. Y. “Simulated annealing algorithm for redundancy optimization with multiple component choices”, Advanced Reliability Modeling, pp. 237-244 (2004).

[12] Liang, Y. C. & Smith, A. E. “An ant colony optimization algorithm for the redundancy allocation problem (RAP)”, IEEE Transactions on Reliability, 53(3), pp. 417-423 (2004).

[13] Coit, D. W., & Smith, A. E. “Solving the redundancy allocation problem using a combined neural network genetic algorithm approach”, Computers & operations research, 23(6), pp. 515-526 (1996).

[14] Coit, D. W., & Smith, A. E. “Stochastic formulations of the redundancy allocation problem”, In Proceedings of the Fifth Industrial Engineering Research Conference, Minneapolis (1996).

[15] Amari, S. V., & Dill, G. “A redundancy optimization problem with warm-standby redundancy”, In 2010 Proceedings of IEEE Annual Reliability and Maintainability Symposium (RAMS), (2010).

[16] Misra, K. B., & Sharma, U. “An efficient algorithm to solve integer-programming problems arising in system-reliability design”, IEEE Transactions on Reliability, 40(1), pp. 81-91 (1991).

[17] Coit, D. W., & Smith, A. E. “Optimization approaches to the redundancy allocation problem for series-parallel systems”, In Fourth Industrial Engineering Research Conference Proceedings, (1995).

[18] Coit, D. W., & Liu, J. C. “System reliability optimization with k-out-of-n subsystems”, International Journal of Reliability, Quality and Safety Engineering, 7(02), pp. 129-142 (2000).

[19] Coit, D. W. “Maximization of system reliability with a choice of redundancy strategies”, IIE transactions, 35(6), pp. 535-543 (2003).

[20] Beji, N., Jarboui, B., Eddaly, M., et al. “A hybrid particle swarm optimization algorithm for the redundancy allocation problem”, Journal of Computational Science, 1(3), pp. 159-167 (2010).

[21] Yeh, W.-C., & Hsieh, T.-J. “Solving reliability redundancy allocation problems using an artificial bee colony algorithm”, Computers & Operations Research, 38(11), pp. 1465-1473 (2011).
[22] Hsieh, Y.-C., & You, P.-S. “An effective immune-based two-phase approach for the optimal reliability–redundancy allocation problem”, Applied Mathematics and Computation, 218(4), pp. 1297-1307 (2011).

[23] 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).

[24] Ardakan, M. A., & Hamadani, A. Z. “Reliability–redundancy allocation problem with cold-standby redundancy strategy”, Simulation Modelling Practice and Theory, 42, pp. 107-118 (2014).

[25] Guilani, P. P., Sharifi, M., Niaki, S., 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).

[26] Zareitalab, 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).

[27] Levitin, G., Xing, L., Peng, S., et al. “Optimal choice of standby modes in 1-out-of-N system with respect to mission reliability and cost”, Applied Mathematics and Computation, 258, pp. 587-596 (2015).

[28] Sharifi, M., Cheragh, G., Maljaii, K. D., et al. “Reliability Optimization of a Series-Parallel k-out-of-n System with Failure Rates Depends on Working Components of System”, International Journal of Industrial Engineering, 22(4), pp. 38-453 (2015).

[29] Lai, C.-M., & Yeh, W.-C. “Two-stage simplified swarm optimization for the redundancy allocation problem in a multi-state bridge system”, Reliability Engineering & System Safety, 156, pp. 148-158 (2016).

[30] Teimouri, M., Zareitalab, A., Niaki, S., et al. “An efficient memory-based electromagnetism-like mechanism for the redundancy allocation problem”, Applied Soft Computing, 38, pp. 423-436 (2016).

[31] Kim, H., & Kim, P. “Reliability–redundancy allocation problem considering optimal redundancy strategy using parallel genetic algorithm”, Reliability Engineering & System Safety, pp. 159, 153-160 (2017).

[32] Ghavidel, S., Azizivahed, A., & Li, L. “A hybrid Jaya algorithm for reliability–redundancy allocation problems”, Engineering Optimization, 50(4), pp. 698-715 (2018).

[33] Ardakan, M. A., & Rezvan, M. T. “Multi-objective optimization of reliability–redundancy allocation problem with cold-standby strategy using NSGA-II”, Reliability Engineering & System Safety, 172, pp. 225-238 (2018).
[34] Tavana, M., Khalili-Damghani, K., Di Caprio, D., et al. “An evolutionary computation approach to solving repairable multi-state multi-objective redundancy allocation problems”, Neural Computing and Applications, 30(1), pp. 127-139 (2018).

[35] Essadqi, M., Idrissi, A., & Amarir, A. An “Effective Oriented Genetic Algorithm for solving redundancy allocation problem in multi-state power systems”, Procedia Computer Science, 127, pp. 170-179 (2018).

[36] Peiravi, A., Karbasian, M., & Abouei Ardakan, M. “K-mixed strategy: A new redundancy strategy for reliability problems. Proceedings of the Institution of Mechanical Engineers”, Part O: Journal of Risk and Reliability, 232(1), pp. 38-51 (2018).

[37] Hadipour, H., Amiri, M., & Sharifi, M. “Redundancy allocation in series-parallel systems under warm standby and active components in repairable subsystems”, Reliability Engineering & System Safety, 192, 106048 (2019).

[38] Ouyang, Z., Liu, Y., Ruan, S.-J., et al. “An improved particle swarm optimization algorithm for reliability-redundancy allocation problem with mixed redundancy strategy and heterogeneous components”, Reliability Engineering & System Safety, 181, pp. 62-74 (2019).

[39] Peiravi, A., Karbasian, M., Ardakan, M. A., et al. “Reliability optimization of series-parallel systems with K-mixed redundancy strategy”, Reliability Engineering & System Safety, 183, pp. 17-28 (2019).

[40] Huang, X., Coolen, F. P., & Coolen-Maturi, T. “A heuristic survival signature based approach for reliability-redundancy allocation”, Reliability Engineering & System Safety, 185, pp. 511-517 (2019).

[41] Sharifi, M., Shahriri, M., & Zaretalab, A. “The Effects of Technical and Organizational Activities on Redundancy Allocation Problem with Choice of Selecting Redundancy Strategies Using the memetic algorithm”, International Journal of Industrial Mathematics, 11(3), pp. 165-176 (2019).

[42] Sun, M. X., Li, Y. F. & Zio, E. “On the optimal redundancy allocation for multi-state series-parallel systems under epistemic uncertainty”, Reliability Engineering & System Safety, 192, 106019 (2019).

[43] Sharifi, M., Moghaddam, T. A., & Shahriri, M. “Multi-objective Redundancy Allocation Problem with weighted-k-out-of-n subsystems”, Heliyon, 5(12), e02346 (2019).

[44] Yeh, W. C. “Solving cold-standby reliability redundancy allocation problems using a new swarm intelligence algorithm”, Applied Soft Computing, 83, 105582 (2019).

[45] Pourkarim Guilani, P., Azimi, P., Sharifi, M., et al. “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).
[46] Juybari, M. N., Abouei Ardakan, M., & Davari-Ardakani, H. “A penalty-guided fractal search algorithm for reliability–redundancy allocation problems with cold-standby strategy”, Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 233(5), pp. 775-790 (2019).

[47] Sharifi, M., Saadvandi, M., & Shahriari, M. R. “Presenting a series-parallel redundancy allocation problem with multi-state components using recursive algorithm and meta-heuristic”, Scientia Iranica, 27(2), pp. 970-982 (2020).

[48] Sharifi, M., & Taghipour, S. “Optimizing a redundancy allocation problem with open-circuit and short-circuit failure modes at the component and subsystem levels”, Engineering Optimization, pp. 1-17 (2020).

[49] Mellal, M. A., & Zio, E. “System reliability-redundancy optimization with cold-standby strategy by an enhanced nest cuckoo optimization algorithm”, Reliability Engineering & System Safety, 201, 106973 (2020).

[50] Sharifi, M., Cheragh, G., Maljaii, K. D., et al. “Reliability and Cost Optimization of a System with k-out-of-n Configuration and Choice of Decreasing the Components Failure Rates”, Scientia Iranica, (online).

[51] Borhani Alamdari, A. H., & Sharifi, M. “Solving a Joint Availability-Redundancy Optimization Model with Multi-State Components with Meta-Heuristic”, International Journal of Industrial Mathematics, 12(1), pp. 59-70 (2020).

[52] Zaretalab, A., Hajipour, V., & Tavana, M. “Redundancy allocation problem with multi-state component systems and reliable supplier selection”, Reliability Engineering & System Safety, 193, 106629 (2020).

[53] She, J., & Pecht, M. G. “Reliability of a k-out-of-n warm-standby system”, IEEE Transactions on Reliability, 41(1), pp. 72-75 (1992).

[54] Chern, M. S. “On the computational complexity of reliability redundancy allocation in a series system”, Operations research letters, 11(5), pp. 309-315 (1992).

[55] Tavakkoli-Moghaddam, R., Safari, J., & 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).

[56] Yun, Y. “Hybrid genetic algorithm with adaptive local search scheme”, Computers & Industrial Engineering, 51(1), pp. 128-141 (2006).

[57] Montgomery, D. C. “Design and Analysis of Experiments” 6th edn., John Wiley and Sons. New York, NY (2005).
[58] Aghaei, M., Hamadani, A. Z., & Ardakan, M. A. “Redundancy allocation problem for k-out-of-n systems with a choice of redundancy strategies”, Journal of Industrial Engineering International, 13(1), pp. 81-92 (2017).
List of the figures’ captions:
- Fig. 1: The current research methodology.
- Fig. 2: Pseudo-code of the proposed GA.
- Fig. 3: A sample for the solution encoding.
- Fig. 4: Uniform crossover operator of the model.
- Fig. 5: Mutation operator.
- Fig. 6: Pseudo-code of the presented HL local search.
- Fig. 7: Box plots of the statistical test on HGA and GA performance.

List of the tables’ captions:
- Table 1. Related research studies after 2010
- Table 2. The range of the algorithms’ parameters.
- Table 3. The optimum value of the algorithms’ input parameter.
- Table 4. The instance's input parameters.
- Table 5. Results of the GA and HGA.
- Table 6. Comparison between the computational of GA and HA.
- Table 7. Results for the ten instances.
- Table 8. %PDA of the algorithms.
- Table 9. Two-Sample T-test for HGA and GA performance
- Table 10. Sensitivity analysis of the system’s available budget (C).
- Table 11. Sensitivity analysis of the system’s maximum acceptable weight (W).
- Table 12. Sensitivity analysis on the system’s maximum acceptable volume (V).
- Table 13. System’s reliability for different values of γ.
Fig. 1: The current research methodology.

Problem definition
- Model assumptions
- Building the mathematical model

Algorithms' Design
- Design the GA
- Design the Hybrid GA (HGA)
- Algorithms' parameters tuning

Algorithms' Comparison
- Design the instances
- Solve the Instances using GA and HGA
- Compare the results of GA and HGA

Discussion, Conclusion, and further studies
- Interpretation of the results
- Conclusion
- Some directions for further studies
Set $N_{pop}, p_c, p_m, p_e$ and number of algorithm’s iteration’s $Noi$.

**Generate the initial generation,**
Create random individuals,

**Evaluate the individuals**
Calculate individuals’ reliability
Calculate individuals’ penalty functions
Calculate individuals’ fitness function

**For** $i = 1$ to $Noi$ **do**

**Procedure: Create the next generation**
Create the pool containing the previous generation individuals

**Operators**
Parents selection: roulette wheel
Crossover $p_c$
Mutation $p_m$

**Procedure: Evaluate the individuals**

**Next-generation**
Elitism $p_e$
Roulette wheel selection $p_e(1 - p_e)$

**End if**

**Report:** The best individual

---

Fig. 2: Pseudo-code of the proposed GA.
Fig. 3. A sample for the solution encoding.
Fig. 4: Uniform crossover operator of the model.
Fig. 5: Mutation operator.
Set $n_e = p_e \times popsize$

For $i = 1$ to $n_e$ do

Select individual $i$ by elitism selection strategy
Set $BestInd = Individual$ $i$
Set $Best\ ff = f f_{individual\ i}$

For $j = 1$ to $popsize$ do

Randomly generate $neighborhood\ j$
Calculate $f f_{neighborhood\ j}$

If $f f_{neighborhood\ j} > BestInd$

$BestInd = neighborhood\ j$

End If

End For

End For

Move $BestInd$ to the next generation

Fig. 6: Pseudo-code of the presented HL local search.
Fig. 7: Box plots of the statistical test on HGA and GA performance.
| Name of the researcher(s)          | Year | Component's type | Objectives | The solving algorithm                                      |
|-----------------------------------|------|------------------|------------|------------------------------------------------------------|
| Beji et al. [20]                  | 2010 | Binary           | Single     | Hybrid particle swarm optimization                           |
| Yeh and Hsieh [21]                | 2011 | Binary           | Single     | Penalty guided artificial bee colony                          |
| Hsieh and You [22]                | 2011 | Binary           | Single     | Immune-based Algorithm                                        |
| Chambari et al. [23]              | 2013 | Binary           | Single     | Simulated annealing                                           |
| Ardakan and Hamadani [24]         | 2014 | Binary           | Single     | Modified genetic algorithm                                    |
| Guilani et al. [25]               | 2014 | Multi            | Single     | Knowledge-based archive simulated annealing                   |
| Zaretaalab et al. [26]            | 2015 | Binary           | Multi      | Genetic Algorithm                                             |
| Levitin et al. [27]               | 2015 | Binary           | Single     | Genetic Algorithm, Memetic Algorithm, Simulated Annealing, and Particle Swarm Optimization |
| Sharifi et al. [28]               | 2015 | Single           | Single     | Genetic Algorithm, Memetic Algorithm, Simulated Annealing, and Particle Swarm Optimization |
| Lai and Yeh [29]                  | 2016 | Multi            | Single     | Two-stage simplified swarm optimization                        |
| Teimouri [30]                     | 2016 | Binary           | Single     | Memory-based electromagnetism-like mechanism                  |
| Kim and Kim [31]                  | 2017 | Binary           | Single     | Parallel genetic algorithm                                    |
| Ghavidel et al. [32]              | 2018 | Binary           | Single     | LJaya-TVAC algorithm                                          |
| Ardakan and Rezvan [33]           | 2018 | Binary           | Multi      | NSGA-II                                                      |
| Tavana et al. [34]                | 2018 | Multi            | Multi      | NSGA-II                                                      |
| Essadqi et al. [35]               | 2018 | Multi            | Multi      | Effective Oriented GA                                         |
| Peiravi et al. [36]               | 2018 | Single           | Single     | Genetic Algorithm                                             |
| Hadipour et al. [37]              | 2019 | Binary           | Multi      | Multi-Objectives Water Flow algorithm, NSGA-II, and NRGA      |
| Ouyang et al. [38]                | 2019 | Binary           | Single     | Improved particle swarm optimization                          |
| Peiravi et al. [39]               | 2019 | Binary           | Single     | Genetic Algorithm                                             |
| Huang et al. [40]                 | 2019 | Binary           | Single     | Heuristic survival signature-based approach                   |
| Sharifi et al. [41]               | 2019 | Binary           | Single     | Memetic Algorithm                                             |
| Sun et al. [42]                   | 2019 | Multi            | Multi      | NSGA-II                                                      |
| Sharifi et al. [43]               | 2019 | Multi            | Multi      | NSGA-II and NRGA                                              |
| Yeh [44]                          | 2019 | Single           | Single     | Simplified Swarm Optimization (SSO),                           |
| Pourkarim et al. [45]             | 2019 | Single           | Single     | Optimization via Simulation Approach                          |
| Juybari et al. [46]               | 2019 | Single           | Single     | Stochastic Fractal Search                                     |
| Sharifi et al. [47]               | 2020 | Multi            | Multi      | Recursive and Genetic algorithms                               |
| Sharifi et al. [48]               | 2020 | Binary           | Multi      | NSGA-ii and NRGA                                              |
| Authors and Year | Year | Population Type | Solution Quality | Method Used |
|------------------|------|-----------------|------------------|-------------|
| Mellal and Zio [49] | 2020 | Binary | Binary | Enhanced Nest Cuckoo Optimization Algorithm (ENCOA) |
| Sharifi and Taghipour [50] | 2020 | Binary | Single | GA |
| Borhani-Alamdar and Sharifi [51] | 2020 | Multi | Single | GA and Simulated Annealing |
| Zaretalab et al. [52] | 2020 | Multi | Single | GA and MA |
| Current study | 2020 | Binary | Single | GA and HGA |
Table 2.
The range of the algorithms’ parameters.

| Parameter | Range   | Lower level | Middle level | High level |
|-----------|---------|-------------|--------------|------------|
| popsize   | 30 – 100| 30          | 65           | 100        |
| pc        | 0.60 – 1.00 | 0.60       | 0.80         | 1.00       |
| pm        | 0.01 – 0.30 | 0.01       | 0.155        | 0.3        |
| b         | 5 – 50   | 5           | 34.5         | 50         |
| maxgen    | 20 – 80  | 20          | 45           | 80         |
Table 3.
The optimum value of the algorithms’ input parameter.

| Parameter | Optimal value |
|-----------|---------------|
|           | GA           | HGA          |
| popsize   | 100          | 81.45        |
| $p_c$     | 1.00         | 1.00         |
| $p_m$     | 0.22         | 0.30         |
| $b$       | 34.50        | 5.00         |
| maxgen    | 80           | 61           |
Table 4.
The instance's input parameters.

| Subsystem | Component type 1 | Component type 2 | Component type 3 | Component type 4 |
|-----------|------------------|------------------|------------------|------------------|
| i         | ki               | λa₁₁             | λs₁₁             | c₁₁              | w₁₁              | v₁₁              | λa₁₂             | λs₁₂             | c₁₂              | w₁₂              | v₁₂              | λa₁₃             | λs₁₃             | c₁₃              | w₁₃              | v₁₃              | λa₁₄             | λs₁₄             | c₁₄              | w₁₄              | v₁₄              |
| 1         | 1                | 0.001054         | 0.000100         | 1                 | 3                 | 5                 | 0.000726         | 0.000040         | 1                 | 4                 | 4                 | 0.000943         | 0.000080         | 2                 | 2                 | 3                 | 0.000513         | 0.000025         | 2                 | 5                 | 2                 |
| 2         | 2                | 0.000513         | 0.000025         | 2                 | 8                 | 2                 | 0.000619         | 0.000032         | 1                 | 10                | 1                 | 0.000726         | 0.000040         | 1                 | 9                 | 2                 | -                 | -                 | -                 | -                 | -                 |
| 3         | 1                | 0.001625         | 0.000425         | 2                 | 7                 | 4                 | 0.001054         | 0.000100         | 3                 | 5                 | 4                 | 0.001393         | 0.000708         | 1                 | 6                 | 2                 | 0.000834         | 0.000042         | 4                 | 4                 | 3                 |
| 4         | 2                | 0.001863         | 0.000538         | 3                 | 5                 | 3                 | 0.001393         | 0.000708         | 4                 | 6                 | 2                 | 0.001625         | 0.000425         | 5                 | 4                 | 3                 | -                 | -                 | -                 | -                 | -                 |
| 5         | 1                | 0.000619         | 0.000032         | 2                 | 4                 | 5                 | 0.000726         | 0.000040         | 2                 | 3                 | 4                 | 0.000513         | 0.000025         | 3                 | 5                 | 5                 | -                 | -                 | -                 | -                 | -                 |
| 6         | 2                | 0.000101         | 0.000010         | 3                 | 5                 | 4                 | 0.000202         | 0.000015         | 3                 | 4                 | 4                 | 0.000305         | 0.000020         | 2                 | 5                 | 3                 | 0.000408         | 0.000023         | 2                 | 4                 | 3                 |
| 7         | 1                | 0.000943         | 0.000080         | 4                 | 7                 | 3                 | 0.000834         | 0.000042         | 4                 | 8                 | 2                 | 0.000619         | 0.000032         | 5                 | 9                 | 4                 | -                 | -                 | -                 | -                 | -                 |
| 8         | 2                | 0.002107         | 0.000720         | 3                 | 4                 | 1                 | 0.001054         | 0.000100         | 5                 | 7                 | 1                 | 0.000943         | 0.000080         | 6                 | 6                 | 2                 | -                 | -                 | -                 | -                 | -                 |
| 9         | 3                | 0.000305         | 0.000020         | 2                 | 8                 | 5                 | 0.000101         | 0.000010         | 3                 | 9                 | 3                 | 0.000408         | 0.000023         | 4                 | 7                 | 4                 | 0.000943         | 0.000080         | 3                 | 8                 | 5                 |
| 10        | 3                | 0.001863         | 0.000550         | 4                 | 6                 | 3                 | 0.001625         | 0.000415         | 4                 | 5                 | 2                 | 0.001054         | 0.000100         | 5                 | 6                 | 1                 | -                 | -                 | -                 | -                 | -                 |
| 11        | 3                | 0.000619         | 0.000032         | 3                 | 5                 | 4                 | 0.000513         | 0.000025         | 4                 | 6                 | 3                 | 0.000408         | 0.000023         | 5                 | 6                 | 3                 | -                 | -                 | -                 | -                 | -                 |
| 12        | 1                | 0.002357         | 0.000835         | 2                 | 4                 | 4                 | 0.001985         | 0.000605         | 3                 | 5                 | 3                 | 0.001625         | 0.000708         | 4                 | 6                 | 4                 | 0.001054         | 0.000100         | 5                 | 7                 | 2                 |
| 13        | 2                | 0.000202         | 0.000015         | 2                 | 5                 | 5                 | 0.000101         | 0.000010         | 3                 | 5                 | 5                 | 0.000305         | 0.000020         | 2                 | 6                 | 3                 | -                 | -                 | -                 | -                 | -                 |
| 14        | 3                | 0.001054         | 0.000100         | 4                 | 6                 | 4                 | 0.000834         | 0.000042         | 4                 | 7                 | 2                 | 0.000513         | 0.000025         | 5                 | 6                 | 2                 | 0.000101         | 0.000010         | 6                 | 9                 | 4                 |
Table 5.
Results of the GA and HGA.

| Subsystem | GA | HGA |
|-----------|----|-----|
| i         | $z_i$ | $n_i$ | Redundancy strategy | $z_i$ | $n_i$ | Redundancy strategy |
| 1         | 3   | 2   | Warm standby        | 3   | 2   | Warm standby        |
| 2         | 1   | 2   | No Redundancy       | 1   | 2   | No Redundancy       |
| 3         | 4   | 2   | Warm standby        | 4   | 1   | No Redundancy       |
| 4         | 3   | 3   | Warm standby        | 3   | 3   | Warm standby        |
| 5         | 1   | 1   | No Redundancy       | 2   | 1   | No Redundancy       |
| 6         | 2   | 2   | No Redundancy       | 2   | 2   | No Redundancy       |
| 7         | 3   | 1   | No Redundancy       | 2   | 1   | No Redundancy       |
| 8         | 1   | 3   | Warm standby        | 1   | 3   | Warm standby        |
| 9         | 3   | 3   | No Redundancy       | 3   | 3   | No Redundancy       |
| 10        | 2   | 4   | Warm standby        | 2   | 4   | Warm standby        |
| 11        | 1   | 4   | Warm standby        | 1   | 4   | Warm standby        |
| 12        | 1   | 2   | Warm standby        | 1   | 2   | Warm standby        |
| 13        | 2   | 2   | No Redundancy       | 2   | 2   | No Redundancy       |
| 14        | 3   | 3   | No Redundancy       | 3   | 4   | Warm standby        |
Table 6. 
Comparison between the computational of GA and HA.

| Algorithm                  | GA    | HGA   |
|----------------------------|-------|-------|
| System reliability         | 0.4269| 0.4403|
| Resources consumed cost    | 118   | 118   |
| Resources consumed Weight  | 170   | 170   |
| Resources consumed volume  | 105   | 101   |
Table 7.
Results for the ten instances.

| Problem | $W$ | GA         | HGA         |
|---------|-----|------------|-------------|
|         |     | Best       | Average     | SD           | Best       | Average     | SD           |
| 1       | 166 | 0.3913     | 0.3828      | 0.0081       | 0.3975     | 0.3907      | 0.0085       |
| 2       | 167 | 0.3974     | 0.3942      | 0.0031       | 0.4108     | 0.4025      | 0.0091       |
| 3       | 168 | 0.4172     | 0.4125      | 0.0081       | 0.4211     | 0.4156      | 0.0064       |
| 4       | 169 | 0.4219     | 0.4199      | 0.0030       | 0.4355     | 0.4283      | 0.0068       |
| 5       | 170 | 0.4269     | 0.4221      | 0.0044       | 0.4403     | 0.4395      | 0.0014       |
| 6       | 171 | 0.4331     | 0.4262      | 0.0070       | 0.4499     | 0.4432      | 0.0060       |
| 7       | 172 | 0.4468     | 0.4423      | 0.0040       | 0.4547     | 0.4475      | 0.0063       |
| 8       | 173 | 0.4611     | 0.4591      | 0.0018       | 0.4713     | 0.4656      | 0.0057       |
| 9       | 174 | 0.4656     | 0.4642      | 0.0013       | 0.4765     | 0.4692      | 0.0084       |
| 10      | 175 | 0.4705     | 0.4664      | 0.0037       | 0.4816     | 0.4799      | 0.0024       |
Table 8.
%PDA of the algorithms.

| Problem | W    | GA      | HGA      |
|---------|------|---------|----------|
|         |      | Best    | Average  | Best   | Average |
| 1       | 166  | 1.56    | 2.02     | 0.00   | 0.00    |
| 2       | 167  | 3.26    | 2.06     | 0.00   | 0.00    |
| 3       | 168  | 0.93    | 0.75     | 0.00   | 0.00    |
| 4       | 169  | 3.12    | 1.96     | 0.00   | 0.00    |
| 5       | 170  | 3.04    | 3.96     | 0.00   | 0.00    |
| 6       | 171  | 3.73    | 3.84     | 0.00   | 0.00    |
| 7       | 172  | 1.74    | 1.16     | 0.00   | 0.00    |
| 8       | 173  | 2.16    | 1.40     | 0.00   | 0.00    |
| 9       | 174  | 2.29    | 1.07     | 0.00   | 0.00    |
| 10      | 175  | 2.30    | 2.81     | 0.00   | 0.00    |
| Average |      | 2.41    | 2.10     | 0.00   | 0.00    |
Table 9.
Two-Sample T-test for HGA and GA performance

| Algorithm | Number of test problem | Mean       | Standard Deviation | Degree of freedom | T-value | P-value |
|-----------|------------------------|------------|--------------------|-------------------|---------|---------|
| HGA       | 10                     | 0.49388    | 0.00223            | 18                | -12.25  | 0.000   |
| GA        | 10                     | 0.50612    | 0.00223            |                   |         |         |
Table 10.
Sensitivity analysis of the system's available budget (C).

| No. | C  | Subsystems | System's Reliability |
|-----|----|------------|----------------------|
| 1   | 130| z 3 1 4 3 1 2 2 1 2 1 1 2 3 |
|     |    | n 2 2 2 2 2 2 4 3 4 4 3 2 4 |
|     |    | S W N A W A N W W N W W N W |
| 2   | 140| z 3 1 4 3 1 2 2 1 2 1 1 2 3 |
|     |    | n 3 3 3 2 3 2 3 4 3 4 3 3 4 |
|     |    | S W W A W A A W W N W W A W |
| 3   | 150| z 2 1 4 3 1 2 2 1 1 2 1 1 2 3 |
|     |    | n 3 4 3 3 3 3 3 3 4 4 4 4 3 4 |
|     |    | S W W W A W A W W W W W W W |
| 4   | 160| z 2 1 4 3 1 2 2 1 1 2 1 1 2 3 |
|     |    | n 4 4 3 3 4 3 4 4 4 4 4 4 4 4 |
|     |    | S W W W A W A W W W W W W W |
| 5   | 170| z 2 1 4 3 1 2 2 1 1 2 1 1 2 3 |
|     |    | n 4 4 4 4 4 3 4 4 4 4 4 4 4 4 |
|     |    | S W W W W A W W W W W W W W |
| 6   | 180| z 4 1 4 3 1 2 2 1 2 1 1 2 3 |
|     |    | n 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |
|     |    | S W W W W W W W W W W W W W |
| 7   | 190| z 4 1 4 3 3 1 3 3 1 2 2 1 1 2 3 |
|     |    | n 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |
|     |    | S W W W W W W W W W W W W W |
| 8   | 200| z 4 1 4 3 3 1 3 3 1 3 3 2 2 1 1 2 4 |
|     |    | n 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |
|     |    | S W W W W W W W W W W W W W |
| 9   | 210| z 4 1 4 3 3 3 1 3 3 2 2 1 1 2 4 |
|     |    | n 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |
|     |    | S W W W W W W W W W W W W W |
| 10  | 220| z 4 1 4 3 3 3 1 3 3 2 2 1 1 2 4 |
|     |    | n 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |
|     |    | S W W W W W W W W W W W W W |
Table 11.
Sensitivity analysis of the system’s maximum acceptable weight ($W$).

| No. | $W$ | Subsystems | System's Reliability |
|-----|-----|------------|----------------------|
| 1   | 170 | n 3 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.4403 |
|     |     | z 2 2 1 3 1 2 1 3 3 4 4 2 2 4 |   |
|     |     | S W N W N N N W N W W W N W |   |
| 2   | 190 | n 3 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.4696 |
|     |     | z 4 2 2 3 2 2 2 3 3 4 4 2 2 4 |   |
|     |     | S W N W W N N A W N W W W A |   |
| 3   | 210 | n 3 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.5006 |
|     |     | z 4 2 3 3 2 3 2 3 3 4 4 2 2 4 |   |
|     |     | S W N W W W W W A W N W W W A |   |
| 4   | 230 | n 3 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.5319 |
|     |     | z 4 4 3 3 3 3 2 3 3 4 4 2 2 4 |   |
|     |     | S W W W W W W W N W W W A W |   |
| 5   | 250 | n 3 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.5759 |
|     |     | z 4 4 3 3 3 3 2 3 3 4 4 3 3 4 |   |
|     |     | S W W W W W W W W W W W W W |   |
| 6   | 270 | n 3 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.6210 |
|     |     | z 4 4 3 3 2 2 2 2 2 3 3 3 3 4 |   |
|     |     | S W W W W W W W W W W W W W |   |
| 7   | 290 | n 3 1 4 3 2 2 2 1 3 2 2 2 2 3 | 0.6310 |
|     |     | z 4 4 4 4 3 4 4 4 4 4 4 4 4 4 |   |
|     |     | S W W W W W W W W W W W W W |   |
| 8   | 310 | n 4 1 4 3 2 2 2 1 3 2 1 1 2 3 | 0.6412 |
|     |     | z 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |   |
|     |     | S W W W W W W W W W W W W W |   |
| 9   | 330 | n 4 1 4 3 2 2 2 1 3 2 2 2 2 2 3 | 0.6952 |
|     |     | z 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |   |
|     |     | S W W W W W W W W W W W W W |   |
| 10  | 350 | n 4 1 4 3 2 2 2 1 3 2 2 2 2 2 3 | 0.7626 |
|     |     | z 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |   |
|     |     | S W W W W W W W W W W W W W |   |
Table 12.
Sensitivity analysis on the system’s maximum acceptable volume ($V$).

| No. | $V$  | Subsystems | System’s Reliability |
|-----|------|------------|----------------------|
| 1   | 110  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.6286 |
|     |      | $z$ 2 2 2 3 2 2 2 3 3 4 4 2 2 4 |      |
|     |      | $S$ W N A W A N W W N W W N W |      |
| 2   | 120  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.6391 |
|     |      | $z$ 2 2 2 3 2 3 2 3 3 4 4 2 3 4 |      |
|     |      | $S$ W N A W A W W W N W W W W |      |
| 3   | 130  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.7047 |
|     |      | $z$ 2 3 3 3 2 3 3 3 4 4 3 3 4 |      |
|     |      | $S$ W W W W A W W W W N W W W W |      |
| 4   | 140  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.7269 |
|     |      | $z$ 3 3 3 3 3 3 3 3 4 4 4 3 4 |      |
|     |      | $S$ W W W W W W W W W W W W W |      |
| 5   | 150  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.7443 |
|     |      | $z$ 4 4 3 3 4 3 3 4 4 4 4 3 4 |      |
|     |      | $S$ W W W W W W W W W W W W W |      |
| 6   | 160  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.7674 |
|     |      | $z$ 4 4 3 4 4 4 4 4 4 4 4 4 3 4 |      |
|     |      | $S$ W W W W W W W W W W W W W |      |
| 7   | 170  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.7732 |
|     |      | $z$ 4 4 4 4 4 4 4 4 4 4 4 4 3 4 |      |
|     |      | $S$ W W W W W W W W W W W W W |      |
| 8   | 180  | $n$ 4 1 4 3 3 1 3 3 2 3 3 4 2 4 | 0.7741 |
|     |      | $z$ 4 4 4 4 4 4 4 4 4 4 4 4 4 4 |      |
|     |      | $S$ W W W W W W W W W W W W W |      |
Table 13.
System’s reliability for different values of $\gamma$.

| $\gamma$ | 1.00 | 0.80 | 0.60 | 0.40 | 0.20 | 0.00 |
|----------|------|------|------|------|------|------|
| System’s reliability | 0.4303 | 0.4440 | 0.4467 | 0.4489 | 0.4499 | 0.4505 |