INTERNATIONAL JOURNAL OF SYSTEMS SCIENCE: OPERATIONS & LOGISTICS, 2017
VOL. 4, NO. 3, 275–296
http://dx.doi.org/10.1080/23302674.2016.1193254

Taylor & Francis
Taylor & Francis Group

A multi-objective multi-state series-parallel redundancy allocation model using tuned meta-heuristic algorithms

Najmeh Alikar[a], Seyed Mohsen Mousavi[a,†], Madjid Tavana[b,c] and S. T. A. Niaki[d]

[a]Department of Mechanical Engineering, Faculty of Engineering, University of Malaya, Kuala Lumpur, Malaysia; [b]Business Systems and Analytics Department, Distinguished Chair of Business Analytics, La Salle University, Philadelphia, USA; [c]Business Information Systems Department, Faculty of Business Administration and Economics, University of Paderborn, Paderborn, Germany; [d]Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran

ABSTRACT

In this paper, we study a series-parallel multi-objective multi-state redundancy allocation problem (MSRAP) with known performance levels and corresponding state probabilities. The problem is comprised of multiple subsystems in series and each subsystem is comprised of multiple components in parallel. The system components have a range of performance level from complete working to complete failure. The subsystems contain homogenous redundant components and the component prices come under an all-unit discount policy if a unique brand (type) is chosen for purchasing all subsystem components. Each component is characterised by its cost, weight and availability. The goals are to find the optimal combination of the components in each subsystem that maximises system availability and minimises the total cost under a weight constraint. We propose a multi-objective harmony search (MOHS) algorithm, a non-dominated sorting genetic algorithm (NSGA-II), and a multi-objective genetic algorithm (MOGA) to solve this problem. In addition, the Taguchi method is utilised to tune the parameters in each algorithm. We use a number of numerical examples to demonstrate the applicability and exhibit the efficacy of the three algorithms. The results show that the MOHS outperforms the NSGA-II and MOGA with respect to all of the considered metrics.

ARTICLE HISTORY
Received 16 January 2016
Accepted 18 May 2016

KEYWORDS
Redundancy allocation problem; series-parallel multi-state system; universal generating function; multi-objective meta-heuristic algorithms; Taguchi method

1. Introduction

Maintaining a high-level of reliability is crucial for efficacious operations in most modern industrial organisations. Kuo and Zuo (2003) suggests two strategies for constructing such systems. One approach pursues improving the reliability of the system components and another approach considers placing the redundant components with the same functionality in parallel to perform a single task. The first approach is typically expensive and beyond the scope of the system designers who select standard off-the shelf products from catalogs. Therefore, the second approach is usually recognised as the preferred option for designing systems with a high-level of reliability (Kuo & Zuo, 2003).

In order to enhance system reliability using redundancy, system designers should consider the tradeoff between system performance and the cost associated with enhancing system reliability. Three different reliability optimisation strategies including: reliability allocation, redundancy allocation, and reliability–redundancy allocation (Chern, 1992; Kuo & Prasad, 2000; Kuo & Wan, 2007; Ravi, Reddy, & Zimmermann, 2000) have been studied in the literature. The fundamental goal of the redundancy allocation problem (RAP) is to find the optimal system structure for achieving high system availability. The RAP is a NP-hard problem (Chern, 1992). Although traditional exact approaches such as branch and bound (Sup & Kwon, 1999) and lexicographic search (Prasad, Kuo, & Kyungmee, 2001) are able to provide exact optimal solutions, they generally involve high computational efforts that make them prohibitively expensive for large or even medium size problems (Coit & Smith, 1996).

The reliability models in the literature are divided into two groups: binary state systems (BSS) and multi-state systems (MSS) (Aven, 1985, 1993; Boedigheimer & Kapur, 1994; Brunelle & Kapur, 1999; Levitin & Lisnianski, 2003). The BSS group consists of the traditional binary reliability models that consider only two states for a system: operating and complete failure. However, in practice, many systems exhibit noticeable gradations of performance between these two extremes. The MSS group introduces multi-state models that allow for any finite number of states in the system and/or its components. The BSS is a simple case of the MSS. Availability,

CONTACT Madjid Tavana tavana@lasalle.edu
[†]Present address: Young Researchers and Elite Club, Qazvin Branch, Islamic Azad University, Qazvin, Iran.

which refers to operational continuity, is commonly used as a measure of reliability for MSS.

There are generally different brands (types) of components available to system designers when constructing a system. Moreover, the supplier of each brand usually offers incentives and/or discounts for purchasing large quantities of a specific component. This is referred to as the unit prices under the all-unit discount policy. The incentives usually take the form of various services such as installation, repair services, etc.

The universal generating function (UGF) approach is an exact method commonly used to evaluate the availability of a formulated solution in a search procedure. The UGF method obtains the availability of an MSS using simple algebraic operations with relatively small computational resources. In the literature, effective procedures which use functions such as the universal z-transform are also used for large-size combinatorial problems. The UGF method demonstrates an expansion of this well-known moment generating function (Ross, 2009).

The remainder of this paper is organised as follows. In the next section, we provide a detailed literature review. In Sections 3, we define the RAP under investigation. In Section 4, we present a mathematical formulation of the multi-objective MSRAP problem. We also introduce our assumptions and notations for a series-parallel RAP in a multi-state situation. This section also contains an introduction to the UGF method to calculate a multi-state version of system availability. Three meta-heuristics used to solve the multi-objective MSRAP problem are introduced in Section 5. In Section 6, several measures are utilised to compare the performance of the resulting Pareto fronts using the test problems provided by (Taboada, Espiritu, & Coit, 2008). In addition, a new criterion of the Taguchi method is proposed to tune the parameters of the algorithms. Finally, in Section 7 we present our conclusions.

2. Literature review

In this paper, we study a series-parallel multi-objective multi-state RAP (MSRAP) where subsystems are designed in series and the components in each subsystem are organised in parallel. All components in each subsystem are assumed homogeneous to assure purchasing under the all-unit discount policy. The objectives are maximising the system availability and minimising the total cost. Moreover, the approach applied in this research (to evaluate availability of system) is based on the universal z-transform that originally was introduced by (Ushakov, 1986). We use three metaheuristics including the multi-objective harmony search (MOHS) algorithm, the non-dominated sorting genetic algorithm (NSGA-II), and the multi-objective genetic algorithm (MOGA) to solve the proposed multi-objective MSRAP problem.

There are a number of research papers in the literature that consider the all-unit discount policy for calculating the costs such as those which consider inventory control and supply chain problems (Mousavi, Hajipour, Niaki, & Alikar, 2013; Mousavi & Pasandideh, 2011; Mousavi et al., 2014; Soltani, Sadjadi, & Tofigh, 2013) recently considered the RAP by maximising the reliability of a system with a limited budget.

In recent years, numerous studies have utilised the UGF to obtain system availability in the MSS (Chang, Lin, & Liu, 2010; Chang & Mori, 2013; Hamadani & Khorshidi, 2012; Konak, Kulturel-Konak, & Levitin, 2011; Levitin, Xing, Ben-Haim, & Dai, 2011; Wang, Li, Huang, & Chang, 2012; Zhou, Zhang, Ran Lin, & Ma, 2012). In addition, a large number of studies have utilised different meta-heuristic algorithms such as variable neighborhood search, ant colony optimisation, particle swarm optimisation, genetic algorithms, and Tabu search to solve NP-hard problems similar to RAP (Coelho, 2009; Konak, Coit, & Smith, 2006; Kumar, Izui, Yoshimura, & Nishiwaki, 2009; Liang & Chen, 2007; Ouzineb, Nourelfath, & Gendreau, 2008; Sadjadi & Soltani, 2009; Salazar, Rocco, & Galván, 2006; Zhao, Liu, & Dao, 2007). The multi-objective version of the RAP is often utilised to solve real-world system engineering optimisation problems. Several researchers have studied the multi-objective version of RAP and used various metaheuristic algorithms such as NSGA-II (Cao, Murat, & Chinnam, 2013; Ghorabaee, Amiri, & Azimi, 2015; Khalili-Damghani, Abtahi, & Tavana, 2013; Safari, 2012; Sudeng & Wattanapongsakorn, 2014), MOGA (Ghorabaee et al., 2015; Taboada et al., 2008; Zoulfaghari, Hamadani, & Ardakan, 2015), and MOPSO (Chambari, Rahmati, & Najafi, 2012) to solve these problems. In addition, (Salcedo-Sanz et al., 2012), (Landa-Torres, Gil-Lopez, Salcedo-Sanz, Ser, & Portilla-Figueras, 2012; Marković, Madić, & Petrović, 2012; Rahmati, Hajipour, & Niaki, 2013; Ricart, Hüttemann, Lima, & Barán, 2011) have applied MOHS to solve a wide range of multi-objective RAP problems. It should be noted that the most distinctive characteristic of multi-objective problems is independent objectives (Deb, 2001).

The harmony search (HS) algorithm, proposed by (Geem, Kim, & Loganathan, 2001), is a population-based algorithm that imitates the improvisation process of musicians. HS has been proven to be an effective algorithm for solving multi-objective problems (Forsati, Mahdavi, Shamsfard, & Reza Meybodi, 2012; Liu & Zhou, 2012; Omran, Geem, & Salman, 2011; Yi, Duan, & Liao, 2012). For instance, (Nahas & Thien-My, 2010) proposed an HS algorithm for optimising an RAP problem with multiple performance levels of components.

(Santos Coelho & de Andrade Bernert, 2009) introduced a modified HS approach combined with an operator of differential evolution, a paradigm of evolutionary computation, to solve RAP optimisation problems. More recently, several researchers have studied the multi-objective version of the HS algorithm to solve different problems such as traffic and mobility (Salcedo-Sanz et al., 2012), facility location (Rahmati et al., 2013), health-care facility location (Landa-Torres et al., 2012), and refuse collection vehicles (Marković et al., 2012).

(Ricart et al., 2011) have proposed two HS methods for solving multi-objective optimisation problems using the *Zitzler–Deb–Thiele* functions as a test-bed which performed well against the NSGA-II method. That is the main reason why this method has been chosen in this study as a solution algorithm for the NP-hard RAP for the first time. Moreover, (Taboada et al., 2008) used MOGA to solve a series-parallel multi-state RAP. The differences between the methods proposed by (Chambari et al., 2012; Safari, 2012; Taboada et al., 2008) with the model proposed in this study are summarised as follows. (Cao et al., 2013) and (Khalili-Damghani et al., 2013) proposed a series-parallel multi-objective binary-state RAP in which the components in each subsystem are different from each other. While (Cao et al., 2013) applied decomposition-based approach to solve the problem, (Khalili-Damghani et al., 2013) used MOPSO. (Safari, 2012) proposed NSGA-II to solve the multi-objective binary-state RAP with different types of components in each subsystem where the type of redundancy strategy was not specified clearly. (Taboada et al., 2008) designed a series-parallel multi-state RAP in which different types of components in the subsystems were allowed. They considered two-objective as well as three-objective versions of RAP and applied MOGA to solve the RAP without any comparisons with other algorithms. Finally, (Chambari et al., 2012) modeled a multi-objective binary-state RAP with the choice of redundancy strategy where the subsystems consisted of different types of the components. They used NSGA-II and MOPSO to optimise their problem.

(Salazar et al., 2006) used a multi-objective binary-state RAP in which the components in each subsystem were different and a NSGA-II was employed to optimise the RAP. The major weakness of their method is not using any algorithms to validate the NSGA-II's performance. (Abdullah Konak et al., 2006) proposed a multi-objective binary-state RAP where a Tabu search algorithm was first developed to obtain the Pareto solutions and a Monte Carlo simulation then provided the decision maker with a pruned and prioritised set of Pareto-optimal solutions based on user-defined objective function preferences. (Wang, Chen, Tang, & Yao, 2009) solved a single-objective binary-state RAP for a gas turbine using the HS algorithm where a mix of different types of components were allowed to be placed in each subsystem. (Wang, Tang, & Yao, 2010) proposed a single-objective multi-state RAP for a serial system where a memetic algorithm was used to solve the RAP.

We model a series-parallel multi-objective multi-state RAP in which all the components in each subsystem are homogeneous. The homogeneity assumption with price discount, which has already been introduced in the literature (see Lisnianski, Levitin, Ben-Haim, & Elmakis, 1996), reflects the case where a supplier selling a specific brand usually offers an all-unit discount for large homogeneous components. In addition, an MOHS algorithm is employed to solve the problem and a Taguchi method is used to obtain the optimal levels of the parameters for the algorithms.

3. Problem definition

In this paper, a series-parallel multi-objective MSRAP is considered for which the subsystems are configured in series and in each subsystem, the components are located in parallel. Figure 1 shows a configuration of the proposed series-parallel system that consists of S subsystems, where subsystem i ($i = 1,2, \ldots,S$) includes n_i components. Furthermore, the components in each subsystem are homogenous (all the components in a subsystem are identical and only chosen from one special type).

To clarify the problem under investigation, consider a real-world example. Let a system have four subsystems, each consisting of a variety of component types supplied from three different brands made in the US, Japan, and England. Furthermore, each component has its own cost, weight and availability. If one of the types (brands) is chosen, the corresponding supplier would offer some kind of discount on the unit price. Therefore, in order to make the problem more realistic, the purchasing cost is considered under the all-unit discount policy as a part of the total cost function. This encourages the owners to buy all the

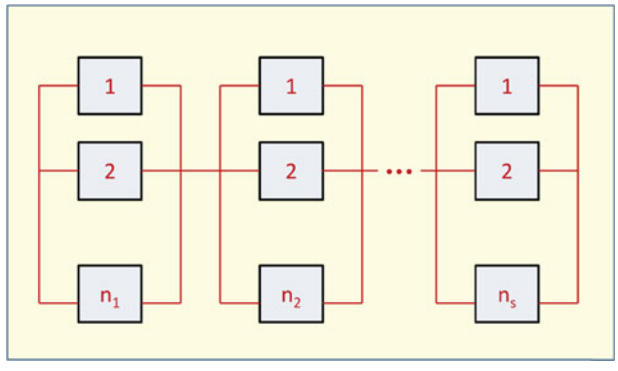


Figure 1. A configuration of the designed system.

```
For i = 1 : S
  For j = 1 : n_i
    Select d ∈ (0,1)
    If  d ≤ HMCR
        X_ij ← X_ij ∈ (X_11, X_12, ..., X_1n1, ..., X_SnS)
    Else  X'_ij ← Generate a new solution into [L_i, U_i]
  End
End
```

Figure 2. The pseudo code of the selection process of the HM algorithm.

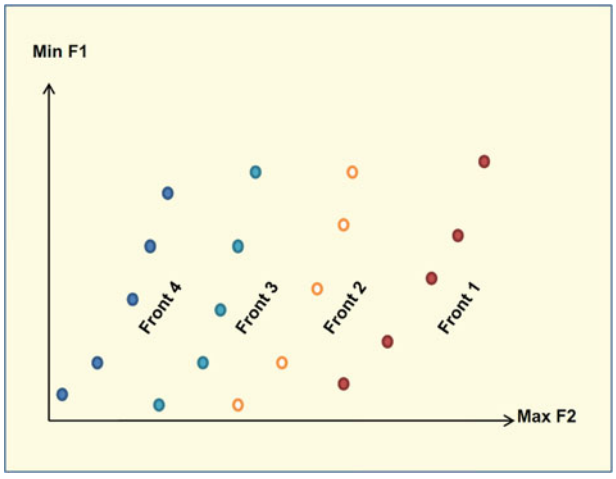


Figure 3. A representation of non-dominated fronts.

```
1: Set d_{m_t} = 0 for t = 1,2,...,M ; m_t = 1,2,...,T_t
2: Sort both objective functions in ascending order
3: The crowding distance for end solutions in each front (m_t = 1, T_t) are d_1 = d_{T_t} = ∞
4: The crowding distance for the objective functions f_{1,2} for m_t = 2,...,T_t − 1 is
calculated by d_{m_t} = d_{m_t} + (f_{1,2} − f_{1,2})
```

Figure 4. A pseudo code to determine the crowding distance.

components of a subsystem from a specific supplier. Moreover, it is assumed that the supplier guarantees services such as installation and repair if all the components are purchased from him/her. Furthermore, it is more cost-effective for an owner to buy all the items from a single supplier (brand) in order to minimise some costs such as transportation (ordering cost) and also the amount of time used to prepare the items. Therefore, this study tries to optimise the number of components in each subsystem such that all of the elements of a subsystem are purchased from a type (a supplier) under an all-unit discount policy. Moreover, the total cost and the system availability must become minimised and maximised, respectively.

The main contributions of this paper are explained as follows. First, we consider a series-parallel multi-objective MSRAP by assuming that only identical redundant components are allowed to be placed in each subsystem as opposed to previous works where the components are assumed to be a combination of different types. Second, each component type is purchased from a certain supplier who sells his/her components under an all-unit discount policy if all of the components are

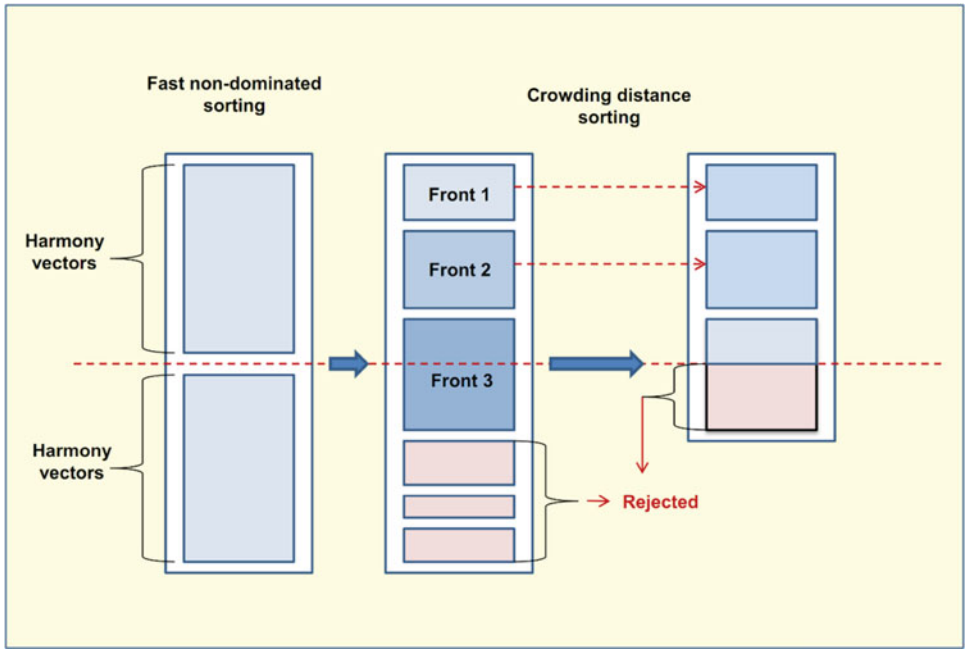


Figure 5. The MOHS procedure.

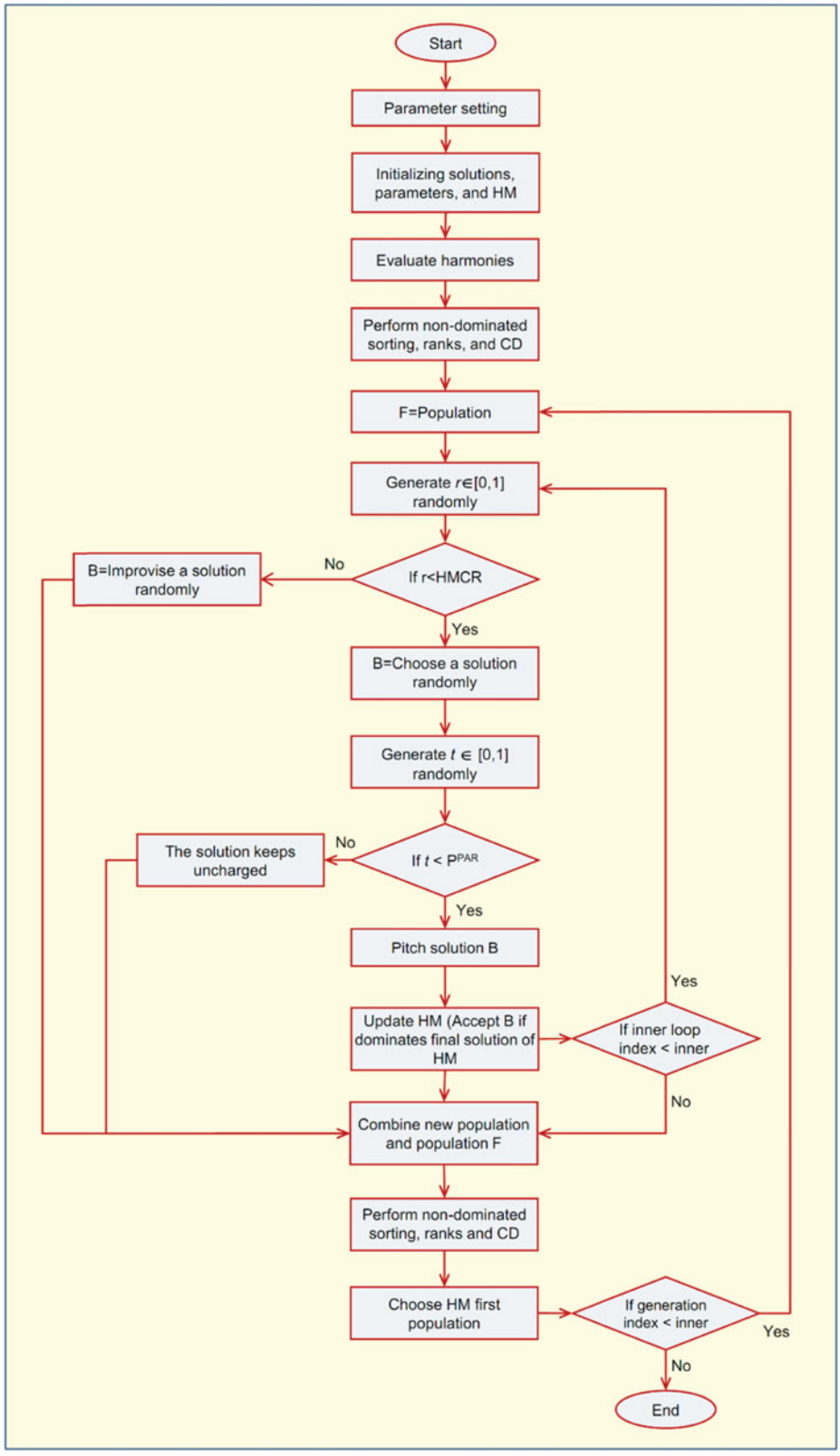


Figure 6. The flow chart of the MOHS.

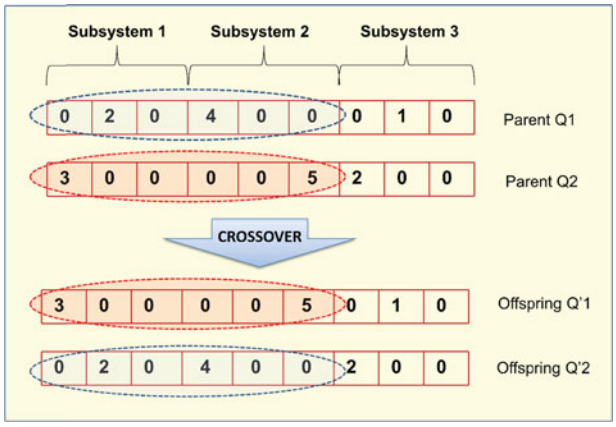


Figure 7. A uniform crossover operator.

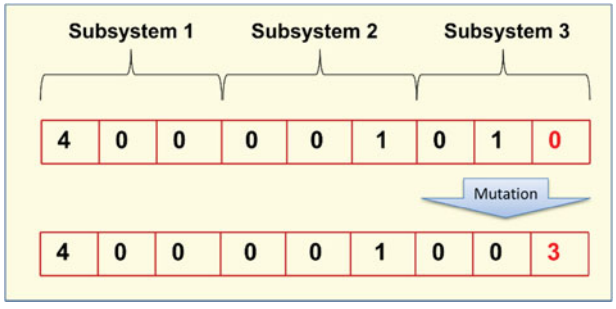


Figure 8. A one-point mutation operator.

purchased from the type proposed by him/her. Third, an MOHS algorithm is utilised for optimising the MSRAP. Furthermore, a new metric using the Taguchi method has been applied to tune the parameters of the algorithm. Finally, a combination of a penalty function and modification strategies is used to handle the constraints in the proposed MSRAP model. The overall objective is to find a system configuration that simultaneously maximises the system availability while minimising the total cost within a limited system weight.

4. Problem formulation

In order to formulate the problem at hand, the assumptions are first explicitly stated in Section 4.1. The required notations are defined in Section 4.2.

4.1. Assumptions and notations

The assumptions made to simplify the model are as follows:

(1) Components are characterised by their availability, capacity and cost according to their type.
(2) Components are always available in the market.
(3) Components are chosen from an existing list in the market.

1: Initialize population (chromosomes)
2: Evaluate objective values (Availability and Cost)
3: Assign rank based on Pareto dominance
4: Assign crowding distance based on Pareto dominance
5: Gen=number of generation
6: *For* t=1: Gen
7: Select the chromosomes from the mating pool based on the tournament strategy
8: *If* $rand \in (0,1) < P_C$
9: Uniform crossover operator
10: *Else* One point mutation
11: *End If*
12: Combine parents and offspring chromosomes and construct a (2×Pop-size) population
13: Evaluate all chromosomes
14: Assign rank based on Pareto dominance
15: Assign crowding distance based on Pareto dominance
16: *End*

Figure 9. The pseudo code of the developed NSGA-II.

```
1: Initialize population (chromosomes)
2: Evaluate objective values (Availability and Cost)
3: Assign rank based on Pareto dominance
4: Assign crowding distance based on Pareto dominance
5: Gen=number of generation
6: For t=1: Gen
7:     Select the chromosomes from the mating pool based on the roulette wheel
       strategy
8:     Uniform crossover operator for P_C ×Pop-size chromosomes
9:     One-point mutation operator for P_m ×Pop-size chromosomes
10:    Elitism operator for (1-P_C ×Pop-size-P_m ×Pop-size) chromosomes
11:    Evaluate the population
12:    Assign rank based on Pareto dominance
13:    Assign crowding distance based on Pareto dominance
14: End
```

Figure 10. The pseudo code of the developed MOGA.

(4) The capacity of a component determines its level of performance.
(5) Failed components are repaired and the components availabilities are known.
(6) The components of a subsystem are identical when selected from the market list.
(7) If all components of a subsystem are purchased from a supplier offering a unique type, the supplier will give the prices under an all-unit discount policy in addition to some other free services such as installation, repair, etc.
(8) The number of subsystems is fixed.
(9) The component states are mutually s-independent, and the system has a finite number of states that are s-independent

4.2. Notations and acronyms

The following notations are used in the proposed model:

i: An index for a subsystem ($i = 1, 2, \ldots, S$)
j: An index of a component type ($j = 1, 2, \ldots, n_i$)
q: An index for the price break-point ($q = 1, 2, \ldots, Q$)
S: Number of subsystems
n_i: Number of components in the ith subsystem
X_{ij}: Number of component type j used in the ith subsystem (a decision variable)
C_{ij}: Cost of component type j in the ith subsystem
C: Total system cost
v_i; V_i; P_{v_i}: Parameters in $U_i(z)$
b, B, A_b, P_b: Parameters in $U(z)$
T_m: mth operation interval
$U_{ij}(z)$: UGF of component type j in the ith subsystem
$U_i(z)$: UGF of the ith subsystem
$U(z)$: UGF of the whole MSRAP
P_{ijl}: Performance level of component of type j at state l in the ith subsystem
P_{Om}: Demanded system performance level during the mth operation interval
w_{ij}: Weight of component type j in the ith subsystem
W: Total system weight
A_{ijl}: Availability of component of type j at state l in the ith subsystem
P_{ij}: Performance level of component type j in the ith subsystem
$A(\omega)$: System availability
L_i: Minimum number of components that can be used in the ith subsystem
U_i: Maximum number of components that can be used in the ith subsystem
e_{ijq}: qth price break-point proposed to purchase the jth component type for the ith subsystem

β_{ijq}: A binary variable that is set to 1 if the jth component type is purchased for the ith subsystem at the q-th price break-point, and set to 0 otherwise

ω: Minimum demand availability of the designed system

The following acronyms are used throughout this manuscript:

MSRAP:	Multi-state redundancy allocation problem
MOHS:	Multi-objective harmony search
MOGA:	Multi-objective genetic algorithm
NSGA-II:	Non-dominated sorting genetic algorithm
RAP:	Redundancy allocation problem
MSS:	Binary-state systems
MSS:	Multi-state systems
UGF:	Universal generating function
HS:	Harmony search
HMS:	Harmony memory size
HMCR:	Harmony memory considering rate
PAR:	Pitch adjusting rate
HMS:	Harmony memory size
NG:	Number of generating
CD:	Crowding distance
ER:	Error ratio
NNS:	Number of non-dominated solutions
MID:	Mean ideal distance
DM:	Diversification metric

4.2. The mathematical model

Based on the assumptions and notations mentioned above, the bi-objective optimisation formulation of the MSRAP problem is as follows:

$$
\begin{aligned}
&\mathrm{MinC}\,(\mathrm{X}) = \sum_{i=1}^{S}\sum_{\mathrm{j}=1}^{\mathrm{n_i}}\sum_{\mathrm{q}=1}^{Q}\mathrm{C_{ij}X_{ij}}\beta_{ijq}\\
&Max\ A(\omega)\\
&s.t.:\\
&\sum_{\mathrm{i}=1}^{S}\sum_{\mathrm{j}=1}^{\mathrm{n_i}}\mathrm{w_{ij}X_{ij}} \le \mathrm{W}\\
&\begin{cases}\mathrm{L_i} \le X_{ij} \le \mathrm{U_i} \ if \mathrm{j} = \mathrm{k}\\ X_{ij} = 0 \quad if\, j \ne k\end{cases},\\
&(i = 1, 2, \ldots, S), \quad k \in (1, 2, \ldots, n_i)\\
&X_{ij} \in \Re\ ,\ \ X_{ij} \ge 0\ ,\ \ (i = 1, 2, \ldots, S;\ j = 1, 2, \ldots, n_i)
\end{aligned}
\tag{1}
$$

The first two terms in Equation (1) are the two objectives of the problem, i.e. minimising total cost and maximising the system availability. The other two terms are the two constraints; the total weight of the system is limited and the components in each subsystem are chosen in the range of $[\mathrm{L}_i, \mathrm{U}_i]$, where $\mathrm{U_i}$ is an upper bound and $\mathrm{L_i}$a lower bound. If component type k $(k = 1, 2, \ldots, n_i)$ is chosen, then $X_{ik} > 0$, otherwise $X_{ij} = 0$for $j \ne k$.

In Equation (1), the total cost is obtained under an all-unit discount policy as all of the components of a subsystem are purchased from a supplier with a certain type. The price break-points of the problem are defined as follows:

$$
\begin{cases}
C_{ij1} & e_{ij1} \le x_{ij} < e_{ij2}\\
C_{ij2} & e_{ij2} \le x_{ij} < e_{ij3}\\
 & \vdots\\
C_{ijQ} & e_{ijQ} \le x_{ij}
\end{cases}
\tag{2}
$$

4.3. The universal generating function (UGF)

In order to evaluate system availability of MSRAP that is required in the second objective function, the UGF method is applied and it is briefly explained as follows. Interested readers are referred to (Ushakov, 2000) and (Levitin, 2005) for more details. The UGF of component j in subsystem i is defined as

$$
U_{ij}(z) = \sum_{l=1}^{L} A_{ijl} z^{P_{ijl}} \tag{3}
$$

Assuming M is a random variable, we have

$$
P(M \ge d) = \sigma\,(U(z) z^{-d}) \tag{4}
$$

where σ is the disruptive operator defined by the following expressions:

$$
\sigma\,(A_{ijl} z^{P_{ijl}-d}) = \begin{cases} A_{ijl} \ if P_{ijl} \ge d \\ 0 \quad \text{otherwise} \end{cases} \tag{5}
$$

$$
\sigma\left(\sum_{l=1}^{L} A_{ijl} z^{P_{ijl}-d}\right) = \sum_{l=1}^{L} \sigma\,(A_{ijl} z^{P_{ijl}-d}) \tag{6}
$$

By using the operator δ, the coefficients of polynomial $U(z)$are summed for every term with $P_{ijl} \ge d$, and the probability that M is not less than some specified value d is systematically obtained (Taboada et al., 2008).

Consider single components with total failures and each component i has nominal performance M_{ijl} and availabilityA_{ijl}. The UMGF of such a component has only two terms and can be defined as (Taboada et al., 2008)

$$
\begin{aligned}
U_{ijl}(z) &= (1 - A_{ijl}) z^0 + A_{ijl} z^{M_{ijl}}\\
&= (1 - A_{ijl}) + A_{ijl} z^{M_{ijl}}
\end{aligned}
\tag{7}
$$

As a result, the UGF of all parallel components in subsystem i is obtained by

$$U_i(z) = \prod_{j=1}^{n_i} [U_{ij}(z)]^{X_{ij}} = \prod_{j=1}^{n_i} \sum_{l=1}^{L} A_{ijl} z^{P_{ijl}} \tag{8}$$

Equation (9) can be expanded into a polynomial-like form as (Wang & Li, 2012)

$$U_i(z) = \sum_{v_i=1}^{V_i} A_{v_i} z^{P_{v_i}}. \tag{9}$$

The UGF of the whole system with subsystems in series and components in parallel is given by

$$\begin{aligned} U(z) &= \Omega_{i=1}^{S}[U_i(z)] = \Omega_{i=1}^{S}[\sum_{v_i=1}^{V_i} A_{v_i} z^{P_{v_i}}] \\ &= \sum_{v_1=1}^{V_1} \sum_{v_2=1}^{V_2} \cdots \sum_{v_S=1}^{V_S} \left[(\prod_{i=1}^{S} A_{v_i}) z^{\min\{P_{v_1}, P_{v_2}, \ldots, P_{v_S}\}} \right], \end{aligned} \tag{10}$$

which after simplifications becomes

$$U(z) = \sum_{b=1}^{B} A_b z^{P_b} \tag{11}$$

As suggested by (Ouzineb et al., 2008) and (Levitin, Lisnianski, & Elmakis, 1997), the entire operation period is divided into Mparts. Let each part T_m ; $m = 1, 2, \ldots, M$have a required performance demand level P_{Om}. Then, the MSS availability is obtained by

$$A_m(\omega) = \sum_{\{v | P_v \geq P_{Om}\}} A_v \tag{12}$$

Finally, the MSS availability for the entire operational period is formulated as follows (Levitin et al., 1997; Ouzineb et al., 2008):

$$A(\omega) = \frac{\left(\sum_{m=1}^{M} A_m(\omega) T_m \right)}{\sum_{m=1}^{M} T_m} \tag{13}$$

Using Equation (12) as the second objective to be maximised, the mathematical formulation of the problem at hand is strongly NP-hard (Chern, 1992). Hence, in the next section three multi-objective meta-heuristic algorithms are developed to solve it.

5. Solving methodologies

To solve the proposed bi-objective MSRAP, three multi-objective meta-heuristic algorithms of MOHS, NSGA-II, and MOGA are utilised. They are explained in the following three subsections.

5.1. MOHS algorithm

The harmony search algorithm has been recently developed as an analogy with the music improvisation process where music players improvise the pitches of their instruments to obtain better harmony (Lee & Geem, 2005). The improvisation seeks to find musically pleasing harmony (a perfect state) as determined by an aesthetic standard, just as the optimisation process seeks to find a global solution (a perfect state) as determined by an objective function (Mahdavi, Fesanghary, & Damangir, 2007; Zou, Gao, Wu, Li, & Li, 2010). The pitch of each musical instrument determines the aesthetic quality, just as the objective function value is determined by the set of values assigned to each decision variable (Zou et al., 2010). The following steps are taken for the developed MOHS algorithm of this paper.

Step 1: Initialising solutions and algorithm parameters

In order to initialise solutions, X_{ij} is randomly generated in the range $[L_i, U_i]$ for $i = 1, 2, \ldots, S$ and $j = 1, 2, \ldots, n_i$. The MOHS parameters are the harmony memory size (HMS), the harmony memory considering rate (HMCR), the pitch adjusting rate (PAR), and the number of generations (NG) or the number of improvisations. A solution vector of a designed system is depicted as follows:

X_{11}	X_{12}	...	X_{1n1}	X_{21}	X_{22}	...	X_{2n2}	...	X_{SnS}

The harmony memory (HM) is a memory location where all the solution vectors (sets of decision variables) are stored. This HM is similar to the genetic pool in the GA (Geem, Kim, & Loganathan, 2002). Also, HMCR and PAR are parameters that are used to improve the solution vector.

Step 2: Initialising HM

The HM matrix is filled by HMS solution vectors randomly. After filling the HM vector, the objective functions, i.e. the system cost (f_{1k}) and the system availability (f_{2k}) are evaluated for $k = 1, 2, \ldots, HMS$. Moreover, to reduce the probability of selecting solutions that are not feasible, two adaptive penalty functions $P_1(x)$ and $P_2(x)$ are used in this research to penalise infeasible solutions. These functions employ the notion of a near-feasibility threshold for the constraints as proposed by (Coit &

Smith, 1996). As a result, the adaptive penalty functions are defined as:

$$F_1(X) = P_1(X) \times C(X),$$
$$F_2(X) = P_2(X) \times A(X),$$

where

$$P_1(X) = Max\left(1, \frac{W(X)}{W}\right)^{\beta}$$
$$P_2(X) = Min\left(1, \frac{W}{W(X)}\right)^{\alpha} \quad (14)$$

The HM matrix is filled by The HMS solution vectors randomly as shown as follows:

X_{11}	X_{12}	...	X_{1n1}	X_{21}	X_{22}	...	X_{2n2}	...	X_{SnS}	**1**
X_{11}	X_{12}	...	X_{1n1}	X_{21}	X_{22}	...	X_{2n2}	...	X_{SnS}	**2**
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
X_{11}	X_{12}	...	X_{1n1}	X_{21}	X_{22}	...	X_{2n2}	...	X_{SnS}	**HMS**

In (14), $F_1(X)$ and $F_2(X)$are the functions related to availability and cost of solution X, respectively, A(X) is the overall availability of the solution X, and α and β are the penalty factors.

Step 3: Improvising a new harmony

A new harmony solution$(X'_{11}, X'_{12}, \ldots, X'_{1n_1}, \ldots, X'_{Sn_S})$ is generated based on three rules: (1) memory consideration, (2) pitch adjustment, and (3) random selection, where the generation of these new solutions is called 'improvisation' (Lee & Geem, 2005).

In the HM consideration rule, as a musician plays any pitch out of the preferred pitches in his/her memory in HM with a probability of *HMCR*, it is randomly chosen with a probability of $(1 - HMCR)$in a random selection process (Geem, Lee, & Park, 2005). Figure 2 depicts a pseudo code of the choice and the selection processes (Geem et al., 2001).

In pitch adjustment, every component obtained by the memory consideration is examined to determine whether it should be pitch adjusted or not. Let *CPA*be the amount of changes for pitch adjustment. Then, the value of the decision variable is first changed in the range using Equation (11) with probability of P^{PAR}. This value is kept the same with probability $(1 - P^{PAR})$:

$$X' = X' \pm Rand.CPA \quad (15)$$

Where *Rand* is a uniform random number between 0 and 1. In Equation (15), the selection for increasing or decreasing the decision variable is carried out for each vector component with the same probability (Taleizadeh, Niaki, Shafii, Meibodi, & Jabbarzadeh, 2010). Then, the two objectives are evaluated based on the changed decision variable where the penalty functions mentioned in Equation (14) are also applied to penalise infeasible solutions.

Step 4: Fast non-dominated sorting

In this step, the HMSpopulations that were generated in the previous steps are compared and sorted. For this purpose, all the solutions in the first non-dominated front are first found using the concept of domination. A solution X_i is said to dominate solution X_j, if $\forall e \in \{1, 2\}$we have $f_e(X_i) \leq f_e(X_j)$for $e \in \{1, 2\}$such that $f_e(X_i) < f_e(X_j)$. In this case, we say X_i is the non-dominated solution within the solution set$\{X_i, X_j\}$. Otherwise, it is not. Figure 3 shows non-dominated fronts of a problem with two objectives F_1 and F_2 where the solutions of front 1 are Pareto solutions. Moreover, in order to find the solutions in the next non-dominated front, the solutions of the previous fronts are disregarded temporarily. This procedure is repeated until all the solutions are set into fronts.

Step 5: Crowding distance

After sorting the populations based on their ranks, a measure called the crowding distance (CD) is defined to evaluate solution fronts of populations in terms of the relative density of individual solutions. To do this, let Z_t; $(t = 1, 2, \ldots, M)$ be the *t-th* front, where T_t is the number of non-dominated solutions in the particular front t, and f_1 and $f_2$are the objective functions. In addition, let d_{m_t}; $(m_t = 1, 2, \ldots, T_t)$ be the value of the crowding distance for the solution m_t. Then, the crowding distance is obtained using the steps proposed in Figure 4. Finally, populations are sorted based on their ranks and crowding distances.

Step 6: Updating harmony memory and CD

In this step, the constraint handling part checks whether all of the constraints of the model are satisfied or not. If they are satisfied, then the population update action occurs according to their ranks and CD.

Step 7: Combining the populations

In this step, both the HMS solutions and the new solutions are combined to generate a population of size (2×HMS) to ensure elitism. Then, the (2×HMS) populations are evaluated and sorted based on their ranks and CD. Finally, according to the sorted solutions, the HMS solutions are selected as the initial population in the next generation. Figure 5 demonstrates the approach taken in MOHS to combine the populations and to select HMS solutions for the next generation.

5.2. NSGA-II

NSGA-II which first was proposed by (Deb, Pratap, Agarwal, & Meyarivan, 2002) was applied by many researchers to optimise their problems. The NSGA-II developed in this research is briefly explained as follows.

The representation of the chromosomes and initialising stages are similar to MOHS where HM in MOHS is replaced by Pop-size (population size). Also, the solution fronts and CD are obtained using the methods explained in MOHS. Now, in order to select individuals of the next generation, the crowded tournament selection operator is applied. If both chromosomes are in the same front, then the chromosome with the most CD is selected. This process is repeated Pop-size times. Figure 6 shows the flow chart of the MOHS provided in this research.

Let r_1 be a uniform random number between zero and one in the crossover operation that takes place with probability P_C. In this operation, if r_1 is less than P_C, for each of the Pop-size chromosomes, then two parent chromosomes $\mathbf{Q}_1$ and $\mathbf{Q}_2$are selected randomly to generate offspring $\mathbf{Q}_1^{'}$ and $\mathbf{Q}_2^{'}$. Let a be an integer random number between 1 and Sthat is related to number of subsystems being selected. Then, the uniform crossover operation of GA is depicted in Figure 7 for a system with 3 subsystems, 3 types of components with$L_i = 1$,$U_i = 5$, and $a = 2$. In this figure, first two out of the available three subsystems are selected for the crossover operation. Then, the chromosomes in the two subsystems of the parents are replaced with each other to create offspring.

After coding and testing different mutation methods, a one-point mutation operator has been found appropriate for the NSGA-II using a trial and error procedure. In this operation which is depicted in Figure 8, first a subsystem is chosen randomly and then the number of its randomly selected components with a non-zero value is replaced randomly by an integer number within its range. In Figure 8, subsystem 3 is first selected and then$X_{32} = 1$ is replaced randomly with $X_{33} = 3$. Note that $(1 - P_C)$ percentage of the population enters into the mutation operator.

Similar to MOHS, the populations generated based on the crossover and mutation operators on the one hand and the initial one on the other hand are combined into a population with size (2×Pop-size). Finally, a population for the next generation is created based on non-dominated sorting, CD, and removing the extra ones (see Figure 5). The algorithm continues running until a predetermined number of iterations (Gen) is reached. Figure 9 displays a pseudo code of the developed NSGA-II of this research.

5.3. MOGA

MOGA is a well-known multi-objective meta-heuristic algorithm that has been used by many researchers to solve different optimisation problems. The developed MOGA of this paper is similar to NSGA-II with the difference that in MOGA a roulette wheel method is utilised to select the chromosomes entering the pool. The roulette wheel proposed in this article is explained as follows.

In order to select individuals of the next generation, the crowded roulette wheel selection operator '$\succ$' is applied. In this operation, two ranked-based tournament selection features including: (I) select the fronts and (II) choose solutions from the fronts, are used. The selection probability of fronts, P_f, and the selection probability of solutions, P_{fs}, are obtained as follows:

$$P_f = \frac{2 \times Rank_f}{NF \times (NF + 1)} \; ; \quad f = 1, \ldots, NF, \tag{16}$$

$$P_{fs} = \frac{2 \times Rank_{fs}}{NS_f \times (NS_f + 1)} ; \quad f = 1, \ldots, NF \; , \; s = 1, \ldots, NS \; , \tag{17}$$

where NF and $NS_f$are the number of fronts and the number of solutions in front f, respectively. Equation (16) ensures that a front with the highest rank has the most chance to be selected. Similarly, based on Equation (17), solutions with more crowding distances are assigned to higher selection probabilities. The roulette wheel selection is iterated until a desired number of solutions are selected. For more information on the crowded roulette wheel selection, interested readers are referred to (Al Jadaan, Rajamani, & Rao, 2008a, 2008b; Al Jadaan, Rao, & Rajamani, 2006). At the end, the algorithm stops when a predetermined number of iterations are reached. Interested readers are referred to (Al Jadaan et al., 2008b) and (Al Jadaan et al., 2008a) for more information on the crowded roulette wheel selection method. Figure 10 shows a pseudo code of the MOGA. Note that unlike NSGA-II, MOGA does not include the stage in which the populations are combined. Instead, it uses an elitism operator to transfer some chromosomes directly to the next generation.

In all the algorithms proposed in this work, we first generate the initial solutions in the range randomly. After evaluating the objective functions using these solutions, we test the solutions with the constraints. If a solution does not satisfy the constraints, it will be penalised by the penalty functions provided in Equation (14). It prompts infeasible solutions to have a lower chance to enter to the next generations of the algorithms. Moreover, a newly generated solution is also checked for feasibility and is modified if needed (we generate a solution in the range randomly instead). After evaluating the solution, the penalty functions are activated in the fitness function to prevent entering this solution to the next generations.

Table 1. Input of the test problems.

	Subsystem	Number of states	Component type	Availability	Feeding capacity	Cost ($)	Weight (kg)
Values	$U[1, 5]$	$U[2, 4]$	$U[1, 8]$	$U[0.05, 0.9]$	$U[0, 2.5]$	$U[50, 300]$	$U[50, 100]$

Table 2. Data for problem no. 7.

Subsystem	Component type	Availability	Feeding capacity	Cost ($)	Weight (kg)
1	1	0.44	1.80	248	73
		0.26	1.60		
		0.17	1.30		
		0.13	0.00		
	2	0.71	1.90	196	69
		0.29	0.00		
	3	0.67	1.40	125	74
		0.18	0.70		
		0.25	0.00		
2	1	0.50	2.00	120	70
		0.25	1.40		
		0.20	1.00		
		0.05	0.00		
	2	0.22	2.00	270	94
		0.20	1.60		
		0.14	1.20		
		0.16	0.90		

Table 3. Data for problem no. 11.

Subsystem	Component type	Availability	Feeding capacity	Cost ($)	Weight (kg)
1	1	0.73	2.20	92	68
		0.27	0.00		
	2	0.83	1.50	218	85
		0.17	0.00		
	3	0.55	1.90	107	66
		0.22	1.10		
		0.23	0.00		
2	1	0.46	2.10	235	78
		0.32	1.60		
		0.22	0.00		
	2	0.68	1.80	214	67
		0.32	0.00		
3	1	0.54	2.30	109	79
		0.46	0.00		
	2	0.44	1.80	186	93
		0.37	0.70		
		0.19	0.00		
	3	0.73	2.00	122	63
		0.27	0.00		
	4	0.18	2.30	264	87
		0.29	2.10		
		0.21	0.80		
		0.42	0.00		
	5	0.43	1.90	228	83
		0.57	0.00		

6. Implementations and comparisons

This section involves the implementation process of the proposed methodology and the performance comparisons of the three meta-heuristic algorithms, where the parameters of all the algorithms are calibrated employing a parameter tuning procedure.

6.1. Test problems

In order to demonstrate the application of the proposed methodology, 15 test problems are generated. The input data of these problems that is uniformly generated in their specified ranges is shown in Table 1, where $U[m, n]$denotes a uniform random variate between m and n. Moreover, in order to generate the price break-points proposed by the suppliers, the number of price break-points is first generated randomly in the range $U[1, 3]$ and then the amounts of e_{ijq}(for $i = 1, 2, \ldots, S$; $j = 1, 2, \ldots, n_i$; $q = 1, 2, \ldots, Q$) are generated in the range $U[0, 500]$ randomly in ascending order.

6.2. The well-known benchmark

A well-known benchmark case taken from (Taboada et al., 2008) is also selected from the literature to evaluate the performance of the algorithms. Table 2 shows the general data for the benchmark case of a system with 3 subsystems in which the component type, availability, feeding capacity, cost, and weight of each component are displayed in columns 2 to 6, respectively. In addition, the maximum and the minimum numbers of components in each sub-system are 5 and 1, respectively. Here, the generation strategy of the price break-points is similar to the one for the test problems where the amounts of price break-points are generated randomly in the range $U[0, 300]$.

In this work, all the algorithms are coded using MATLAB (2010b), where a PC with 2.2 GHz Intel Core 2 Duo CPU, and 4 GB of RAM memory is used for all the calculations.

6.3. Comparison metrics

In order to compare the performances of the three multi-objective meta-heuristic algorithms four metrics are used as follows.

(1) Number of non-dominated solutions (NNS)
(2) Error ratio (ER) that measures the on-convergence of the methods towards the real Pareto front (Khalili-Damghani & Amiri, 2012;

Table 4. Data for the benchmark case.

Subsystem	Component type	Availability	Feeding capacity	Cost ($)	Weight (kg)
1	1	0.70	1.30	65	80
		0.20	1.00		
		0.10	0.00		
	2	0.65	1.00	60	70
		0.25	0.80		
		0.10	0.00		
	3	0.60	0.95	50	75
		0.30	0.90		
		0.10	0.00		
	4	0.90	1.35	80	100
		0.05	0.80		
		0.05	0.00		
2	1	0.50	2.00	120	70
		0.25	1.40		
		0.20	1.00		
		0.05	0.00		
	2	0.60	2.20	130	100
		0.30	1.40		
		0.10	0.00		
	3	0.90	3.00	200	100
		0.10	0.00		
3	1	0.80	1.60	200	60
		0.15	0.90		
		0.05	0.00		
	2	0.85	1.40	160	100
		0.15	0.00		
	3	0.90	2.00	250	90
		0.10	0.00		
	4	0.65	1.00	100	70
		0.30	0.80		
		0.05	0.00		
	5	0.50	1.30	60	50
		0.30	1.00		
		0.15	0.50		
		0.05	0.00		

Table 5. The parameters of the algorithms with their levels.

Algorithms	Parameters	Low (1)	Medium (2)	High (3)
MOGA	Pop-size	50	80	100
	Pc	0.6	0.65	0.75
	Pm	0.1	0.15	0.2
	NG	200	700	1500
NSGA-II	Pop-size	30	50	70
	Pc	0.7	0.8	0.85
	Pm	0.05	0.1	0.15
	NG	100	500	1000
MOHS	HMS	30	40	50
	HMCR	0.92	0.95	0.98
	PAR	0.2	0.9	2
	NG	200	500	1000

Table 6. The optimal levels of the parameters of the algorithms.

Algorithms	Parameters	Optimal levels
MOGA	Pop-size	50
	Pc	0.6
	Pm	0.15
	NG	700
NSGA-II	Pop-size	50
	Pc	0.7
	Pm	0.1
	NG	1000
MOHS	HMS	40
	HMCR	0.98
	PAR	0.2
	NG	1000

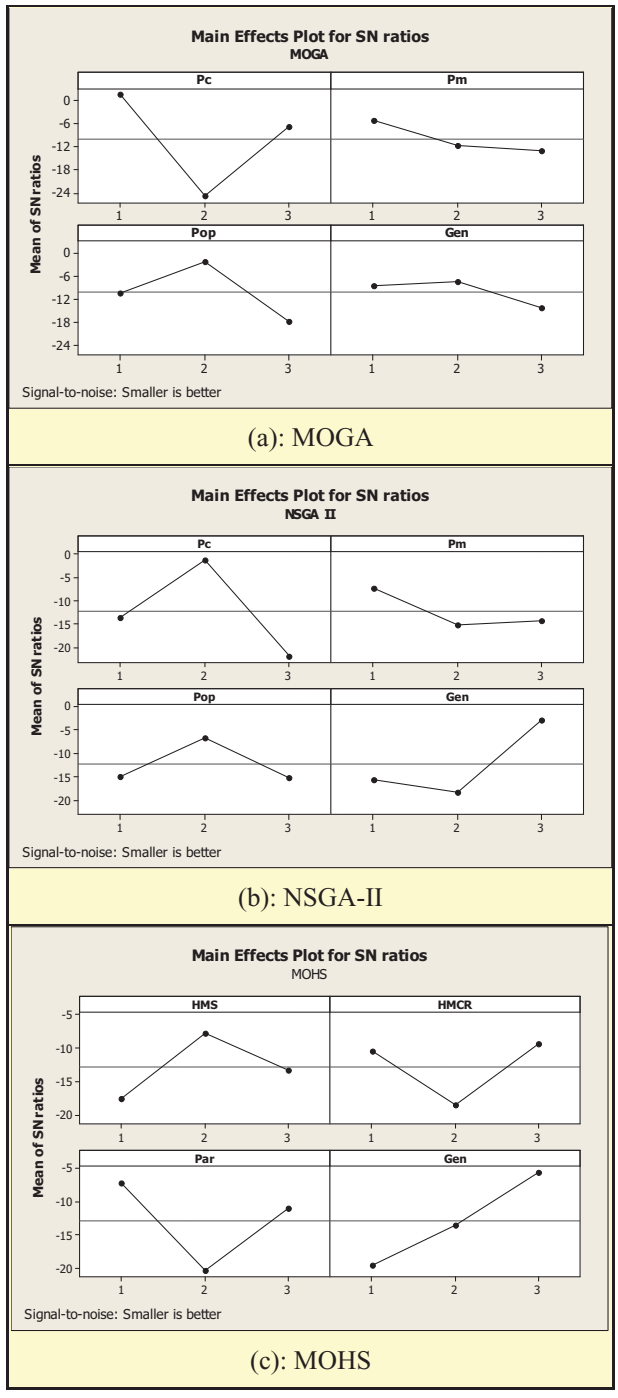


Figure 11. The mean S/N ratios obtained by MOGA, NSGA-II, and MOHS.

Van Veldhuizen, 1999). ER is defined as

$$ER = \frac{\sum_{t=1}^{N} e_t}{N}, \tag{18}$$

where, N is the number of non-dominated solutions and e_t is equal 1 if the solution t belongs to the Pareto front, 0 otherwise.

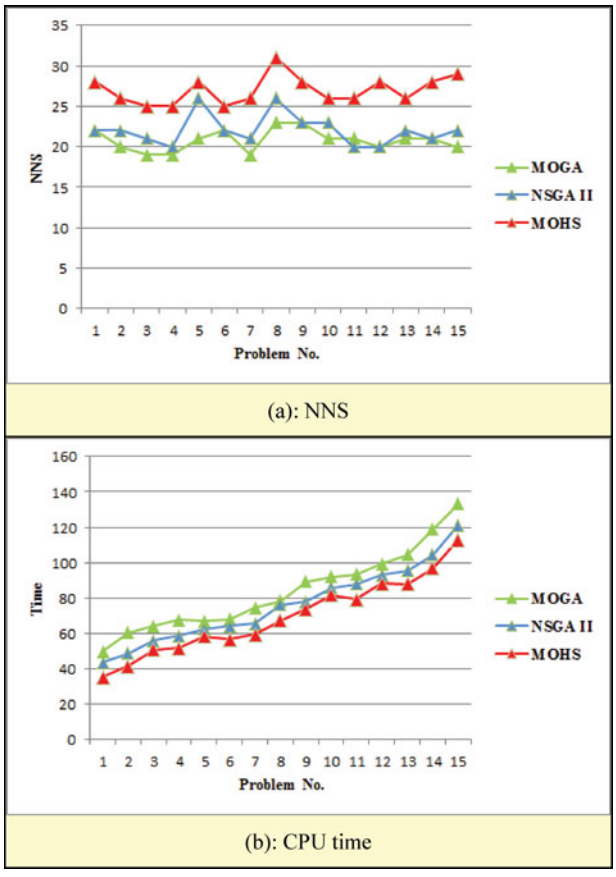


Figure 12. NNS and CPU time of the algorithms.

(1) Mean ideal distance (MID) that measures the convergence rate of the Pareto fronts to a certain point (0,0) (Zitzler & Thiele, 1998).
(2) Diversification metric (DM) that measures the spread of the solution set (Hyun, Kim, & Kim, 1998).

Table 8. The required CPU time of the algorithms.

Problem no.	MOGA	NSGA-II	MOHS
1	49.91	43.57	35.31
2	60.41	48.79	41.51
3	64.23	56.13	50.76
4	67.49	58.59	51.82
5	66.81	62.35	58.41
6	67.84	64.22	56.68
7	74.46	65.50	59.48
8	78.31	76.09	67.20
9	89.21	77.89	73.71
10	91.98	85.69	81.77
11	93.52	87.89	79.22
12	99.38	93.04	88.22
13	104.43	95.64	87.89
14	118.67	104.30	96.91
15	133.42	121.08	113.11
Ave	84.005	76051	69.466
Std. Dev.	23.209	21.832	21.734

Table 9. The ANOVA analysis of the metrics.

Metric's name	*F*-value	*P*-value	Test results
NNS	58.61	0.000	Reject null hypothesis
ER	9.19	0.000	Reject null hypothesis
MID	16.20	0.000	Reject null hypothesis
DM	12.14	0.000	Reject null hypothesis
CPU time	1.60	0.213	Accept null hypothesis

(3) The CPU time of running the algorithms to reach near optimum solutions.

6.4. Tuning parameters

Parameters of meta-heuristics play important roles in the quality of the obtained solutions. While there are three approaches, namely factorial designs, response

Table 7. Performance of the algorithms.

	NNS			ER			MID			DM		
Problem no.	MOGA	NSGA-II	MOHS	MOGA	NSGA-II	MOHS	MOGA	NSGA-II	MOHS	MOGA	NSGA-II	MOHS
1	22	22	28	0.271	0.211	0.158	991.412	327.342	71.192	223.786	89.045	81.902
2	20	22	26	0.348	0.378	0.259	496.172	171.432	343.128	21.213	219.132	51.902
3	19	21	25	0.202	0.183	0.096	617.672	183.270	78.328	375.834	142.145	25.790
4	19	20	25	0.251	0.242	0.172	429.209	534.180	109.701	318.478	273.253	80.109
5	21	26	28	0.125	0.259	0.052	265.409	211.123	481.725	69.681	219.932	41.251
6	22	22	25	0.117	0.298	0.125	391.032	667.124	362.022	479.390	281.199	48.118
7	19	21	26	0.324	0.422	0.109	881.111	206.456	201.099	430.278	61.410	55.208
8	23	26	31	0.247	0.251	0.201	783.331	844.174	190.403	471.459	205.657	51.262
9	23	23	28	0.102	0.244	0.093	628.155	553.202	29.934	370.110	154.672	49.610
10	21	23	26	0.290	0.364	0.016	1019.349	465.832	71.836	36.103	129.569	31.640
11	21	20	26	0.285	0.322	0.147	324.528	456.737	196.980	512.391	142.432	69.943
12	20	20	28	0.089	0.158	0.114	762.883	165.569	13.940	103.115	74.890	16.778
13	21	22	26	0.110	0.091	0.117	805.801	387.104	511.346	158.132	167.687	41.231
14	21	21	28	0.303	0.261	0.197	1089.119	19.960	32.351	201.253	146.752	72.274
15	20	22	29	0.286	0.251	0.215	459.038	50.809	341.024	28.213	287.128	70.430
Ave.	20.800	22.066	26.866	0.223	0.262	0.138	662.948	349.620	202.333	253.295	172.993	52.496
Std. Dev.	1.320	1.869	1.732	0.090	0.086	0.064	264.953	235.845	167.027	179.419	72.767	19.563

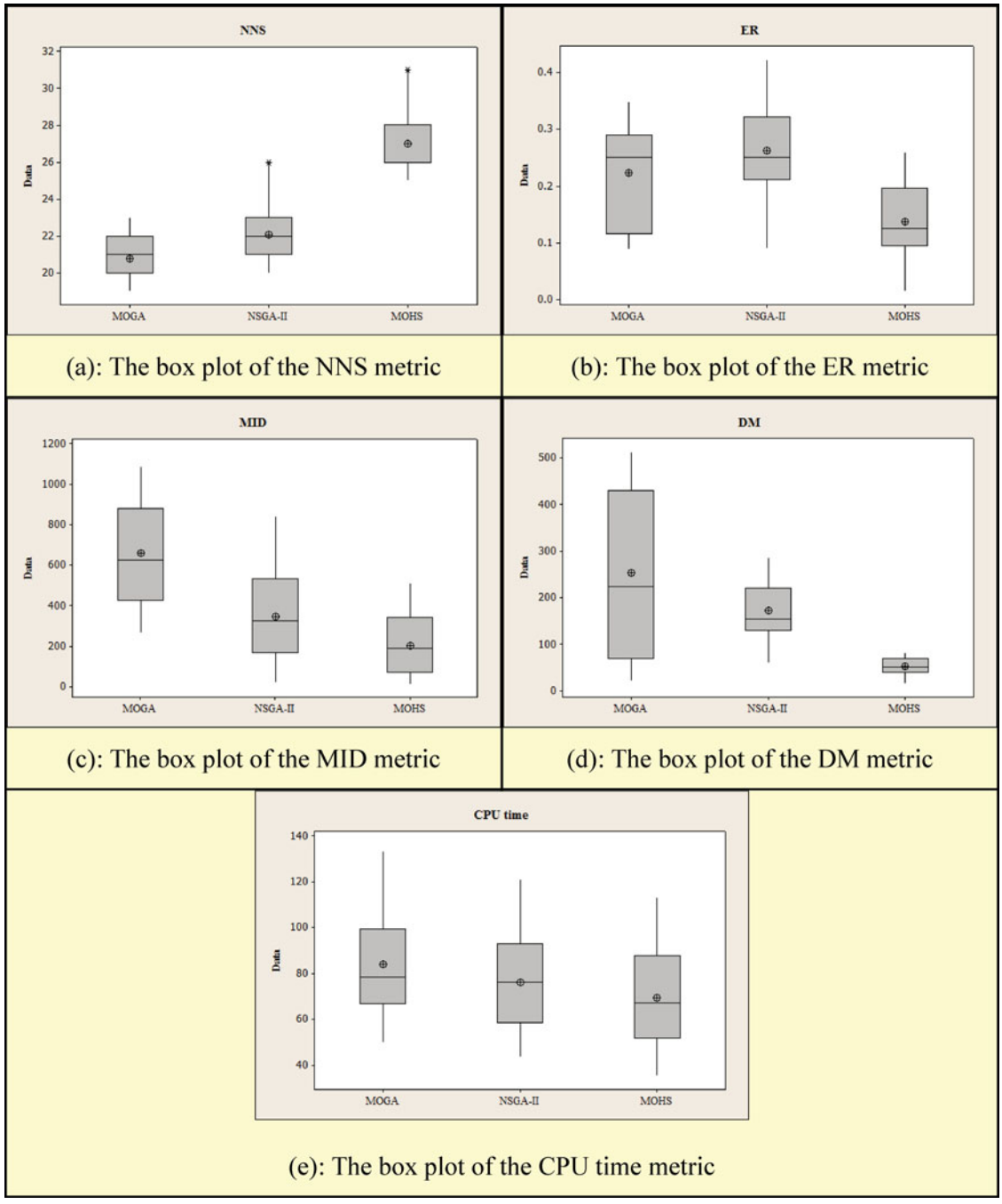


Figure 13. The box plot of the NNS, ER, MID, DM, and CPU time metrics.

Table 10. Computational results of the metric values for the benchmark case based on different weights and demands.

	NNS			ER			MID			DM		
(W, D%)	MOGA	NSGA-II	MOHS	MOGA	NSGA-II	MOHS	MOGA	NSGA-II	MOHS	MOGA	NSGA-II	MOHS
(800, 100)	24	27	32	0.178	0.129	0.021	375.799	219.939	214.944	309.321	231.316	72.245
(800, 80)	21	22	26	0.283	0.240	0.093	233.237	321.195	89.205	414.652	232.980	63.348
(500, 100)	19	21	23	0.289	0.141	0.107	440.589	223.155	33.128	313.327	125.876	73.219
(500, 80)	14	15	18	0.269	0.179	0.014	213.610	135.819	158.519	437.438	378.280	39.681
(300, 100)	12	13	16	0.142	0.159	0.123	226.274	163.512	161.571	473.218	376.432	88.327
(300, 80)	8	8	10	0.181	0.279	0.065	495.723	285.986	174.318	432.109	258.370	79.422
Ave.	16	17.166	19.833	0.223	0.187	0.070	330.872	224.934	138.614	396.677	267.209	69.373
Std. Dev.	6.261	7.054	7.626	0.064	0.059	0.045	122.846	70.336	65.715	68.813	96.725	16.740

Table 11. Pareto solutions obtained based on $W = 800$ and 100% demand.

	MOHS		NSGA-II		MOGA	
Solution no.	Availability%	Cost	Availability%	Cost	Availability%	Cost
1	89.82	330	89.95	360	76.49	250
2	90.52	350	84.11	280	75.04	250
3	88.85	310	96.01	500	85.94	300
4	72.83	190	97.32	540	81.15	260
5	91.43	370	90.88	380	90.72	370
6	76.52	210	74.23	230	89.35	340
7	96.76	520	99.25	600	84.04	280
8	96.02	490	77.81	250	93.71	430
9	93.81	430	83.12	270	96.32	490
10	97.76	550	98.81	590	98.31	550
11	78.03	220	97.86	560	97.17	510
12	85.74	280	93.61	440	92.79	410
13	83.71	260	92.81	420	91.73	390
14	94.51	450	96.62	520	95.02	460
15	92.61	400	88.82	340	89.82	350
16	97.16	530	94.34	460	94.41	440
17	98.45	570	80.93	260	87.73	320
18	93.25	420	72.27	220	84.89	290
19	87.73	300	78.93	250	80.22	260
20	79.42	230	98.16	570	97.76	530
21	84.71	270	76.91	240	73.28	240
22	95.42	470	75.34	240	98.91	580
23	65.78	160	87.04	310	78.25	260
24	96.57	510	81.95	260	82.42	270
25	98.95	590	87.73	320	–	–
26	87.04	290	85.9	300	–	–
27	70.38	180	79.55	250	–	–
28	75.03	200	–	–	–	–
29	81.31	240	–	–	–	–
30	82.46	250	–	–	–	–
31	99.29	600	–	–	–	–
32	68.44	170	–	–	–	–

Table 12. The Pareto solutions generated by the algorithms for $W = 800$ and $D = 80\%$.

	MOHS		NSGA_II		MOGA	
Solution no.	Availability%	Cost	Availability%	Cost	Availability%	Cost
1	84.12	290	69.4	270	99.25	600
2	99.25	600	98.63	600	89.35	340
3	89.35	320	72.2	270	74.79	290
4	72.2	260	79.31	280	92.4	370
5	92.9	370	88.83	340	81.22	290
6	81.22	270	87.12	320	97.76	520
7	97.56	520	81.12	290	95.63	450
8	96.03	450	93.81	420	93.81	410
9	94.81	420	96.12	480	98.52	580
10	98.52	580	97.76	550	80.12	290
11	80.23	270	97.06	520	85.74	310
12	85.74	300	91.79	370	94.79	430
13	67.4	250	93.1	400	93.1	390
14	94.51	410	95.42	460	97.18	500
15	93.71	390	89.97	350	87.24	320
16	97.96	550	86.21	310	76.44	290
17	97.24	500	76.94	280	82.76	300
18	87.24	310	94.61	440	73.89	290
19	76.44	260	74.34	280	91.15	350
20	82.76	280	70.32	270	78.29	290
21	69.89	250	75.48	280	72.32	290
22	91.15	340	83.91	300	–	–
23	77.68	260	–	–	–	–
24	79.44	270	–	–	–	–
25	74.34	260	–	–	–	–
26	96.61	470	–	–	–	–

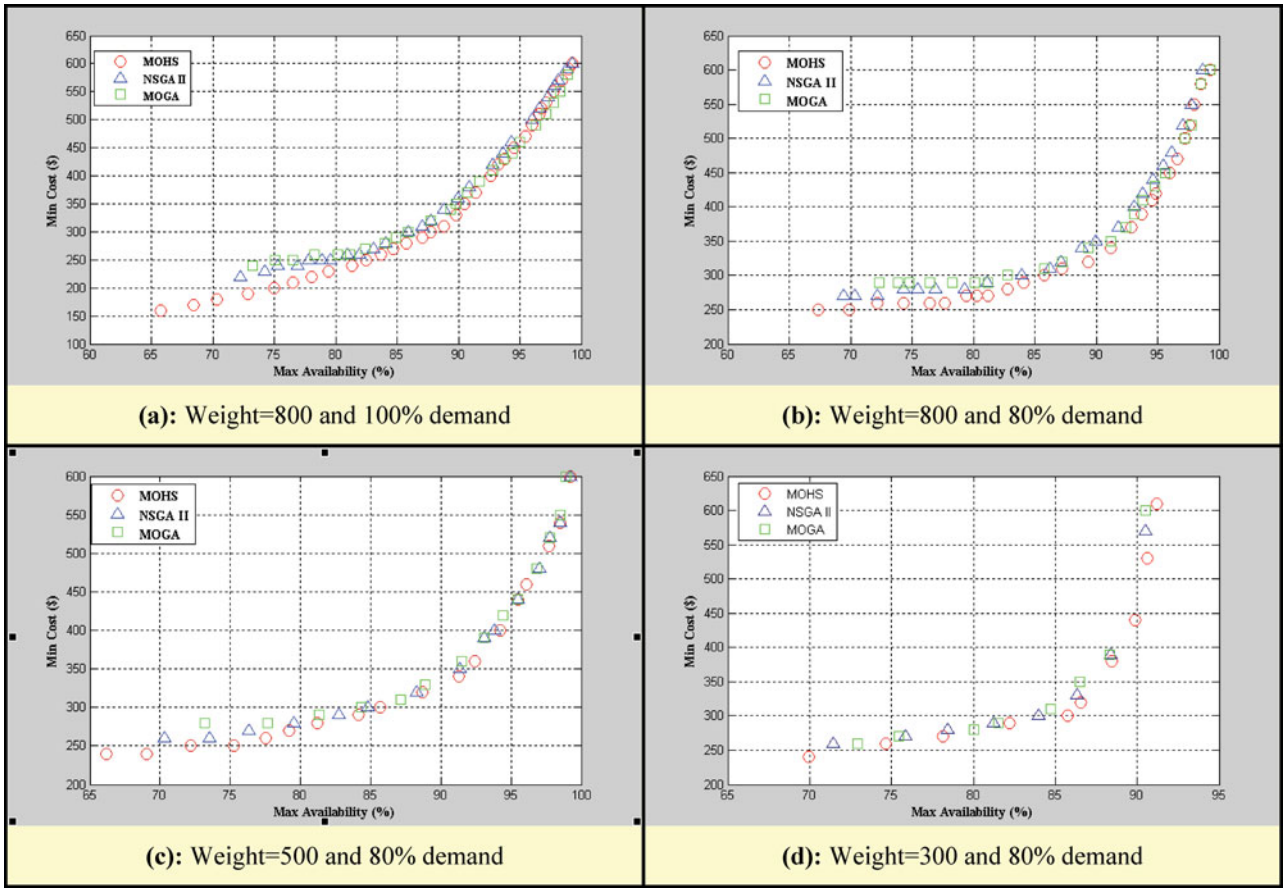


Figure 14. The graph of the Pareto solutions obtained for varying weights and demand levels.

surface methodology, and the Taguchi method, available in the literature to calibrate parameters of meta-heuristics, the latter is used in this research to tune the parameters of the three meta-heuristic algorithms. Taguchi proposed a transformation of the reiteration data to another value as a measure of variation. The transformation is the signal-to-noise (S/N) ratio that explains why this type of parameter design is called

Table 13. The Pareto solutions generated by the algorithms for $W = 500$ and $D = 80\%$.

	MOHS		NSGA-II		MOGA	
Solution no.	Availability%	Cost	Availability%	Cost	Availability%	Cost
1	84.13	290	73.49	260	98.88	600
2	75.22	250	99.25	600	95.52	440
3	81.23	280	91.38	350	81.35	290
4	96.08	460	88.29	320	84.31	300
5	79.22	270	76.33	270	88.91	330
6	98.51	540	82.74	290	96.82	480
7	97.68	510	97.02	480	91.51	360
8	91.29	340	98.5	540	97.76	520
9	95.52	440	84.82	300	93.1	390
10	92.4	360	93.1	390	94.42	420
11	88.67	320	93.81	400	87.11	310
12	94.19	400	95.52	440	98.51	550
13	85.69	300	97.76	520	77.67	280
14	69.02	240	79.51	280	73.21	270
15	99.25	600	70.31	260	–	–
16	72.2	250	–	–	–	–
17	77.53	260	–	–	–	–
18	66.2	240	–	–	–	–

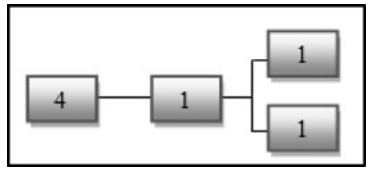
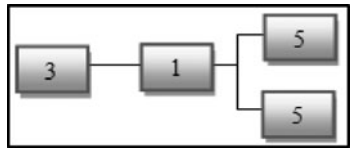
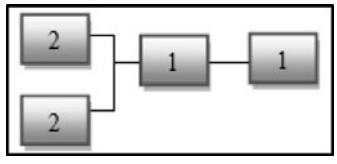
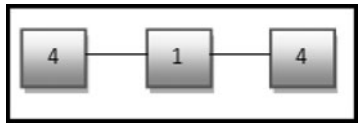
Table 14. The Pareto solutions obtained with $W = 300$ and 80% demand.

	MOHS		NSGA-II		MOGA	
Solution no.	Availability	Cost	Availability	Cost	Availability	Cost
1	86.53	320	71.46	260	84.73	310
2	88.45	380	90.53	570	86.51	350
3	89.9	440	88.41	390	72.94	260
4	85.74	300	75.83	270	75.43	270
5	91.21	610	78.42	280	81.53	290
6	69.94	240	86.33	330	80.02	280
7	90.64	530	84.07	300	90.51	600
8	74.62	260	81.23	290	88.35	390
9	78.15	270	–	–	–	–
10	82.18	290	–	–	–	–

robust (Phadke, 1989). Here, the term 'signal' denotes the desirable value of the mean response and 'noise' denotes the undesirable value (its standard deviation). The purpose is to maximise S/N. Furthermore, Taguchi classifies objective functions into three groups: smaller-the-better, larger-the-better, and nominal-the-better (Ross, 1996). Recently, many researchers utilised the Taguchi method to calibrate the parameters of their developed meta-heuristics in single-objective optimisation problems in the multi-objective version (see, for example, Mousavi, Hajipour, et al., 2013; Mousavi & Niaki, 2012; Mousavi, Niaki, Mehdizadeh, & Tavarroth, 2013; Pasandideh, Niaki, & Mousavi, 2013; Rahmati et al., 2013; Sadeghi, Mousavi, Niaki, & Sadeghi, 2013).

In the work proposed in (Deb, 2001), it is explicitly mentioned that 'by the duality principle, we can convert a maximisation problem into a minimisation one by multiplying the objective function by −1. The duality principle has made the task of handling mixed type of objectives much easier'. In this work, by converting

Table 15. A part of the design configuration obtained by MOHS with $W = 300$ and 80% demand.

Solution no.	System design configuration diagram	Availability	Cost
5		91.21	610
6		69.94	240
10		82.18	290
3		89.9	440
4		85.74	300

the second objective function, i.e. maximising system availability, to a minimising objective, both objective functions will be of the minimisation type. Therefore, smaller-the-better type of response is applied in the Taguchi method. Equation (19) shows its corresponding S/N ratio in which $Y_i$denotes the response in i-th experiment and $n$represents the number of orthogonal arrays, depending on which of the experiments are performed:

$$S/N = -10\log\left(\frac{\sum_{i=1}^{n} Y_i^2}{n}\right). \tag{19}$$

As mentioned, two main goals including (I) good convergence and (II) diversity are sought in Pareto-based algorithms. In addition, among the aforementioned four metrics, CPU time and MID are the ones that measure the convergence rate of the algorithms and the others are used to assess the diversity of the algorithms (Rahmati et al., 2013). In this research, the diversity and the MID metrics are combined into a new measure called *MOCV* and defined as

$$MOCV = \frac{MID}{DM}. \tag{20}$$

The parameters of the three algorithms along with their three levels in the Taguchi method are shown in Table 3. These values are used in the L_{27} Taguchi design. Figure 11 shows the main effect plots of the mean S/N ratios obtained employing MOGA, NSGA-II, and MOHS, respectively.

According to these figures, the optimal levels of the parameters are the highest ones that are listed in Table 4. For both the test problems and the benchmark, the Taguchi design is obtained in a similar way as in the aforementioned proposed example.

6.5. Experimental results

The results of the applications of the developed three meta-heuristics on 15 randomly generated problems given in Table 1 as well as the benchmark along with their performance analyses are discussed in this subsection, with the exception that the total system weight is assumed to be a uniform random variate between 500 and 1000, i.e. $U[500, 1000]$. Table 5 lists the performances of the three algorithms in terms of the above-mentioned metrics.

The averages of the NNS metric in Table 5 show that MOHS is the best. In other words, MOHS provides the highest number of non-dominated solutions on the average. Figure 12(a) shows this superiority. Moreover, in terms of the ER metric, MOHS demonstrates the lowest average error ratio and hence is the best.

In addition, since smaller values of MID and DM are preferred, the results in Table 5 reveal that MOHS is the best algorithm in terms of both metrics. Finally, Table 6 shows that the MOHS is the best algorithm in terms of CPU. The required CPU times of the algorithms versus problem sizes are also depicted in Figure 12(b), where MOHS in shown to be the best algorithm in all of the problems, followed by NSGA-II and then MOGA.

To compare the average performances of the algorithms statistically, an analysis of variance (ANOVA) is performed at a 95% confidence level on all the metrics. The ANOVA results summarised in Table 7 show that while there are no significant differences between the algorithms in term of average CPU, they are statistically different in terms of average NNS, average ER, average MID, and average DM.

Moreover, the box plots of the average metrics shown in Figure 13 illustrate that MOHS has the best performance in terms of all of the metrics.

The algorithms are also applied to the benchmark case presented in Table 2 in order to further clarify their effectiveness in solving the MSRAP problem at hand.

To do this, all the algorithms are utilised three times on the benchmark case for three values of weights chosen to be 800, 500, and 300 with 80% and 100% of the demand. In addition, the system operation intervals i.e. T_m for 80% and 100% of the demand of the performance levels are chosen to be 20 and 30 hours, respectively. The other required input are $\alpha = 10$, $\beta = 150$, $L_i = 1$, and $U_i = 5$ (for $i = 1, 2, 3$). Table 8 presents the metrics obtained based on different combinations of the weights and demands. Once again, the average metrics shown in Table 8 reveal that MOHS has the best performance in terms of almost all the metrics in all the cases.

In order to explain the strategy to find the solutions, consider Example 1, for which the number of subsystems is initially determined. The number of component types in this example is then randomly generated in the ranges $U[1, 5]$ and $U[1, 8]$. Next, for each type, the number of states is generated in the range $U[2, 4]$, where the performance level of each type for each state is generated randomly in the range $U[0, 2.5]$, starting from 0 for the first state and then increasing in ascending order for the rest of the states. For instance, if we have 2 types with 2 states for the first type and 3 states for type 2, the performance levels will be generated randomly as 0 and 1 for the first type and 0, 0.8, 1.4 for the second one. In other words, the performance levels for the first state is always zero and for the other ones is increased in ascending order with a random rate.

Based on a system weight of 800 using 100% demand, the Pareto solutions obtained using the three meta-heuristics are shown in Table 9 and Figure 14. Again, it can be easily seen that the NNS metric of MOHS is higher than the ones of the other two algorithms. Furthermore, Tables 10–12 and Figure 14 show the Pareto solutions obtained based on the weights 800, 500 and 300 and 80% demand, respectively. Finally, system configurations are shown in Table 13 based on the solutions 1, 2, 4, 6, and 10 with $W = 300$ and 100% demand obtained using MOHS. Table 14 shows the Pareto solutions obtained with $W = 300$ and 80% demand. In addition, Table 15 depicts a part of the design configuration obtained by the MOHS algorithm with $W = 300$ and 80% demand.

7. Conclusion and future research directions

In this study, we formulated a multi-objective multi-state redundancy allocation problem (MSRAP) for a series-parallel system where the subsystems were organised in series and the components were arranged in parallel in each subsystem. The primary goal in this study was to find the optimal number of the homogenous components required in each subsystem so that the system availability is maximised and the total system cost is minimised subject to a weight constraint. The MOGA, NSGA-II and MOHS, each with tuned parameters using the Taguchi method, were employed to solve the MSRAP. In addition to an available benchmark, a variety of test problems were randomly generated to evaluate the performances of the three algorithms utilising several metrics. Statistical, graphical, and tabular analysis of the metric results showed the superiority of the MOHS over the MOGA and NSGA-II algorithms for solving the MSRAP. Future research directions could include studying a repairable model for the problem as well exploring the use of other solution methods. Moreover, other meta-heuristic algorithms are recommended to solve the problem. Furthermore, NSGA-II and MOGA can be used with different approaches of crossover and mutation operators for solving the proposed problem.

Acknowledgments

The authors would like to thank the anonymous reviewers and the editor for their insightful comments and suggestions.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

S. T. A. Niaki http://orcid.org/0000-0001-6281-055X

References

Al Jadaan, O., Rajamani, L., & Rao, C. (2008a). Improved selection operator for GA. *Journal of Theoretical & Applied Information Technology*, *4*(4), 269–277.

Al Jadaan, O., Rajamani, L., & Rao, C. (2008b). *Non-dominated ranked genetic algorithm for Solving multiobjective optimization Problems. Journal of Theoretical and Applied Information Technology*, *4*(1), 640–651.

Al Jadaan, O., Rao, C.R., & Rajamani, L. (2006). Parametric study to enhance genetic algorithm performance, using ranked based roulette wheel selection method. *Paper presented at the International Conference on Multidisciplinary Information Sciences and Technology (InSciT2006)*. India.

Aven, T. (1985). Reliability/availability evaluations of coherent systems based on minimal cut sets. *Reliability Engineering*, *13*(2), 93–104.

Aven, T. (1993). On performance measures for multistate monotone systems. *Reliability Engineering & System Safety*, *41*(3), 259–266.

Boedigheimer, R.A., & Kapur, K.C. (1994). Customer-driven reliability models for multistate coherent systems. *IEEE Transactions on Reliability*, *43*(1), 46–50.

Brunelle, R.D., & Kapur, K.C. (1999). Review and classification of reliability measures for multistate and continuum models. *IIE Transactions*, *31*(12), 1171–1180.

Cao, D., Murat, A., & Chinnam, R.B. (2013). Efficient exact optimization of multi-objective redundancy allocation problems in series-parallel systems. *Reliability Engineering & System Safety*, *111*, 154–163.

Chambari, A., Rahmati, S.H.A., & Najafi, A.A. (2012). A bi-objective model to optimize reliability and cost of system with a choice of redundancy strategies. *Computers & Industrial Engineering*, *63*(1), 109–119.

Chang, P.C., Lin, J.J., & Liu, C.H. (2010). An attribute weight assignment and particle swarm optimization algorithm for medical database classifications. *Computer Methods and Programs in Biomedicine, 107*(1), 382–392.

Chang, Y., & Mori, Y. (2013). A study on the relaxed linear programming bounds method for system reliability. *Structural Safety*, *41*, 64–72.

Chern, M.S. (1992). On the computational complexity of reliability redundancy allocation in a series system. *Operations Research Letters*, *11*(5), 309–315.

Coelho, L.S. (2009). An efficient particle swarm approach for mixed-integer programming in reliability-redundancy optimization applications. *Reliability Engineering & System Safety*, *94*(4), 830–837.

Coit, D.W., & Smith, A.E. (1996). Reliability optimization of series-parallel systems using a genetic algorithm. *IEEE Transactions on Reliability*, *45*(2), 254–260, 266.

Deb, K. (2001). Multi-objective optimization. *Multi-Objective Optimization Using Evolutionary Algorithms*, 13–46.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Forsati, R., Mahdavi, M., Shamsfard, M., & Reza Meybodi, M. (2012). Efficient stochastic algorithms for document clustering. *Information Sciences*, *220*, 269–291.

Geem, Z., Kim, J., & Loganathan, G. (2002). Harmony search optimization: Application to pipe network design. *International Journal of Modelling & Simulation*, *22*(2), 125–133.

Geem, Z.W., Kim, J.H., & Loganathan, G. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60–68.

Geem, Z.W., Lee, K.S., & Park, Y. (2005). Application of harmony search to vehicle routing. *American Journal of Applied Sciences*, *2*(12), 1552–1557.

Ghorabaee, M.K., Amiri, M., & Azimi, P. (2015). Genetic algorithm for solving bi-objective redundancy allocation problem with k-out-of-n subsystems. *Applied Mathematical Modelling*, *39*(20), 6396–6409.

Hamadani, A.Z., & Khorshidi, H.A. (2012). System reliability optimization using time value of money. *The International Journal of Advanced Manufacturing Technology*, 66(1), 97–106.

Hyun, C.J., Kim, Y., & Kim, Y.K. (1998). A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines. *Computers & Operations Research*, *25*(7), 675–690.

Khalili-Damghani, K., Abtahi, A.-R., & Tavana, M. (2013). A new multi-objective particle swarm optimization method for solving reliability redundancy allocation problems. *Reliability Engineering & System Safety*, *111*, 58–75.

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. *Reliability Engineering & System Safety*, *103*, 35–44.

Konak, A., Coit, D.W., & Smith, A.E. (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, *91*(9), 992–1007.

Konak, A., Kulturel-Konak, S., & Levitin, G. (2011). Multi-objective optimization of linear multi-state multiple sliding window system. *Reliability Engineering & System Safety, 98*(1), 24–34.

Kumar, R., Izui, K., Yoshimura, M., & Nishiwaki, S. (2009). Multi-objective hierarchical genetic algorithms for multi-level redundancy allocation optimization. *Reliability Engineering & System Safety*, *94*(4), 891–904.

Kuo, W., & Prasad, V.R. (2000). An annotated overview of system-reliability optimization. *IEEE Transactions on Reliability*, *49*(2), 176–187.

Kuo, W., & Wan, R. (2007). Recent advances in optimal reliability allocation. *Computational Intelligence in Reliability Engineering, 37*(2), 143–156.

Kuo, W., & Zuo, M.J. (2003). *Optimal reliability modeling: Principles and applications*. New York, NY: John Wiley & Sons.

Landa-Torres, I., Gil-Lopez, S., Salcedo-Sanz, S., Ser, J.D., & Portilla-Figueras, J.A. (2012). A novel grouping harmony search algorithm for the multiple-type access node location problem. *Expert Systems with Applications*, *39*(5), 5262–5270.

Lee, K.S., & Geem, Z.W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, *194*(36), 3902–3933.

Levitin, G. (2005). *The universal generating function in reliability analysis and optimization*. London: Springer.

Levitin, G., & Lisnianski, A. (2003). *Multi-state system reliability: Assessment, optimization and applications*. London: World Scientific Publishing.

Levitin, G., Lisnianski, A., & Elmakis, D. (1997). Structure optimization of power system with different redundant elements. *Electric Power Systems Research*, *43*(1), 19–27.

Levitin, G., Xing, L., Ben-Haim, H., & Dai, Y. (2011). Multi-state systems with selective propagated failures and imperfect individual and group protections. *Reliability Engineering & System Safety*, *96*(12), 1657–1666.

Liang, Y.C., & Chen, Y.C. (2007). Redundancy allocation of series-parallel systems using a variable neighborhood search algorithm. *Reliability Engineering & System Safety*, *92*(3), 323–331.

Lisnianski, A., Levitin, G., Ben-Haim, H., & Elmakis, D. (1996). Power system structure optimization subject to reliability constraints. *Electric Power Systems Research*, *39*(2), 145–152.

Liu, L., & Zhou, H. (2012). Hybridization of Harmony Search with Variable Neighborhood Search for Restrictive Single-machine Earliness/Tardiness Problem. *Information Sciences*, *226*, 68–92.

Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, *188*(2), 1567–1579.

Marković, M., Madić, M., & Petrović, G. (2012). Assessing the performance of improved harmony search algorithm (IHSA) for the optimization of unconstrained functions using Taguchi experimental design. *Scientific Research and Essays*, *7*(12), 1312–1318.

Mousavi, S.M., Hajipour, V., Niaki, S.T.A., & Alikar, N. (2013). Optimizing multi-item multi-period inventory control system with discounted cash flow and inflation: Two calibrated meta-heuristic algorithms. *Applied Mathematical Modelling*, *37*(4), 2241–2256.

Mousavi, S.M., & Niaki, S.T.A. (2012). Capacitated location allocation problem with stochastic location and fuzzy demand: A hybrid algorithm. *Applied Mathematical Modelling*, *37*(7), 5109–5119.

Mousavi, S.M., Niaki, S.T.A., Mehdizadeh, E., & Tavarroth, M.R. (2013). The capacitated multi-facility location–allocation problem with probabilistic customer location and demand: Two hybrid meta-heuristic algorithms. *International Journal of Systems Science*, *44*(10), 1897–1912.

Mousavi, S.M., & Pasandideh, S.H. (2011). A multi-periodic multi-product inventory control problem with discount: GA optimization algorithm. *Journal of Optimization in Industrial Engineering, 4*(7), 37–44.

Mousavi, S.M., Sadeghi, J., Niaki, S.T.A., Alikar, N., Bahreininejad, A., & Metselaar, H.S.C. (2014). Two parameter-tuned meta-heuristics for a discounted inventory control problem in a fuzzy environment. *Information Sciences, 276*, 42–62.

Nahas, N., & Thien-My, D. (2010). Harmony search algorithm: Application to the redundancy optimization problem. *Engineering Optimization*, *42*(9), 845–861.

Omran, M.G.H., Geem, Z.W., & Salman, A. (2011). Improving the performance of harmony search using opposition-based learning and quadratic interpolation. *International Journal of Mathematical Modelling and Numerical Optimisation*, *2*(1), 28–50.

Ouzineb, M., Nourelfath, M., & Gendreau, M. (2008). Tabu search for the redundancy allocation problem of homogenous series–parallel multi-state systems. *Reliability Engineering & System Safety*, *93*(8), 1257–1272.

Pasandideh, S.H.R., Niaki, S.T.A., & Mousavi, S.M. (2013). Two metaheuristics to solve a multi-item multiperiod inventory control problem under storage constraint and discounts. *The International Journal of Advanced Manufacturing Technology*, 5(69), 1671–1684.

Phadke, M.S. (1989). *Quality engineering using robust design*. Englewood Cliffs, NJ: Prentice Hall.

Prasad, V.R., Kuo, W., & Kyungmee, O. (2001). Maximization of a percentile life of a series system through component redundancy allocation. *IIE Transactions*, *33*(12), 1071–1079.

Rahmati, S.H.A., Hajipour, V., & Niaki, S.T.A. (2013). A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem. *Applied Soft Computing*, *13*(4), 1728–1740.

Ravi, V., Reddy, P., & Zimmermann, H.J. (2000). Fuzzy global optimization of complex system reliability. *IEEE Transactions on Fuzzy Systems*, *8*(3), 241–248.

Ricart, J., Hüttemann, G., Lima, J., & Barán, B. (2011). Multiobjective harmony search algorithm proposals. *Electronic Notes in Theoretical Computer Science*, *281*, 51–67.

Ross, S.M. (2009). *Introduction to probability models*. Cambridge: Academic Press.

Ross, P.J. (1996). *Taguchi techniques for quality engineering*. New York, NY: Mcgraw-Hill International.

Sadeghi, J., Mousavi, S.M., Niaki, S.T.A., & Sadeghi, S. (2013). Optimizing a multi-vendor multi-retailer vendor managed inventory problem: Two tuned meta-heuristic algorithms. *Knowledge-Based Systems*, *50*, 159–170.

Sadjadi, S.J., & Soltani, R. (2009). An efficient heuristic versus a robust hybrid meta-heuristic for general framework of serial–parallel redundancy problem. *Reliability Engineering & System Safety*, *94*(11), 1703–1710.

Safari, J. (2012). Multi-objective reliability optimization of series-parallel systems with a choice of redundancy strategies. *Reliability Engineering & System Safety*, *108*, 10–20.

Salazar, D., Rocco, C.M., & Galván, B.J. (2006). Optimization of constrained multiple-objective reliability problems using evolutionary algorithms. *Reliability Engineering & System Safety*, *91*(9), 1057–1070.

Salcedo-Sanz, S., Manjarres, D., Pastor-Sánchez, Á., Del Ser, J., Portilla-Figueras, J.A., & Gil-Lopez, S. (2012). One-way urban traffic reconfiguration using a multi-objective harmony search approach. *Expert Systems with Applications*, 40(9), 3341–3350.

Santos Coelho, L.d., & de Andrade Bernert, D.L. (2009). An improved harmony search algorithm for synchronization of discrete-time chaotic systems. *Chaos, Solitons & Fractals*, *41*(5), 2526–2532.

Soltani, R., Sadjadi, S.J., & Tofigh, A.A. (2013). A model to enhance the reliability of the serial parallel systems with component mixing. *Applied Mathematical Modelling, 38*(3), 1064–1076.

Sudeng, S., & Wattanapongsakorn, N. (2014). A preference-based multi-objective evolutionary algorithm for redundancy allocation problem. *Paper presented at the IT Convergence and Security (ICITCS), 2014 International Conference*. Beijing.

Sup, S.C., & Kwon, C.Y. (1999). Branch-and-bound redundancy optimization for a series system with multiple-choice constraints. *IEEE Transactions on Reliability*, *48*(2), 108–117.

Taboada, H.A., Espiritu, J.F., & Coit, D.W. (2008). MOMS-GA: A multi-objective multi-state genetic algorithm for system reliability optimization design problems. *IEEE Transactions on Reliability*, *57*(1), 182–191.

Taleizadeh, A.A., Niaki, S.T.A., Shafii, N., Meibodi, R.G., & Jabbarzadeh, A. (2010). A particle swarm optimization approach for constraint joint single buyer-single vendor inventory problem with changeable lead time and (r, Q) policy in supply chain. *The International Journal of Advanced Manufacturing Technology*, *51*(9–12), 1209–1223.

Ushakov, I. (1986). Universal generating function. *Soviet Journal of Computer and System Sciences*, *24*(5), 37–49.

Ushakov, I. (2000). The method of generalized generating sequences. *European Journal of Operational Research*, *125*(2), 316–323.

Van Veldhuizen, D.A. (1999). Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations: DTIC Document. Evolutionary Computation. Columbus, OH.

Wang, Y., & Li, L. (2012). Heterogeneous redundancy allocation for series-parallel multi-state systems using hybrid particle swarm optimization and local search. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 42*(2), 464–474.

Wang, Y., Li, L., Huang, S., & Chang, Q. (2012). Reliability and covariance estimation of weighted k-out-of-n multi-state systems. *European Journal of Operational Research*, *221*(1), 138–147.

Wang, Z., Chen, T., Tang, K., & Yao, X. (2009). A multi-objective approach to redundancy allocation problem in parallel-series systems. *Paper presented at the Evolutionary Computation, 2009. CEC'09. IEEE Congress*. Trondheim.

Wang, Z., Tang, K., & Yao, X. (2010). A memetic algorithm for multi-level redundancy allocation. *IEEE Transactions on Reliability*, *59*(4), 754–765.

Yi, H., Duan, Q., & Liao, T. (2012). Three improved hybrid metaheuristic algorithms for engineering design optimization. *Applied Soft Computing*, 13(5), 2433–2444.

Zhao, J.H., Liu, Z., & Dao, M.T. (2007). Reliability optimization using multiobjective ant colony system approaches. *Reliability Engineering & System Safety*, *92*(1), 109–120.

Zhou, Y., Zhang, Z., Ran Lin, T., & Ma, L. (2012). Maintenance optimisation of a multi-state series-parallel system considering economic dependence and state-dependent inspection intervals. *Reliability Engineering & System Safety, 111*, 248–259.

Zitzler, E., & Thiele, L. (1998). *Multiobjective optimization using evolutionary algorithms a comparative case study*. Paper presented at the Parallel problem solving from nature PPSN V.

Zou, D., Gao, L., Wu, J., Li, S., & Li, Y. (2010). A novel global harmony search algorithm for reliability problems. *Computers & Industrial Engineering*, *58*(2), 307–316.

Zoulfaghari, H., Hamadani, A.Z., & Ardakan, M.A. (2015). *Multi-objective availability optimization of a system with repairable and non-repairable components*. Paper presented at the Industrial Engineering and Operations Management (IEOM), 2015 International Conference. Dubai.