Reliability Engineering and System Safety 160 (2017) 1–10

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Application of the NSGA-II algorithm to a multi-period inventory-redundancy allocation problem in a series-parallel system

Najmeh Alikar[a], Seyed Mohsen Mousavi[b], Raja Ariffin Raja Ghazilla[a,*], Madjid Tavana[c,d], Ezutah Udoncy Olugu[a]

[a] Department of Mechanical Engineering, Faculty of Engineering, University of Malaya, Kuala Lumpur, Malaysia
[b] Young Researchers and Elite Club, Qazvin Branch, Islamic Azad University, Qazvin
[c] Business Systems and Analytics Department, Distinguished Chair of Business Analytics, La Salle University, Philadelphia, United States
[d] Business Information Systems Department, Faculty of Business Administration and Economics, University of Paderborn, Germany

ARTICLE INFO

Keywords:
Bi-objective multi-period inventory system
Redundancy allocation problem
All-unit discount
NSGA-II

ABSTRACT

In this paper, we formulate a mixed-integer binary non-linear programming model to study a series-parallel multi-component multi-periodic inventory-redundancy allocation problem (IRAP). This IRAP is a novel redundancy allocation problem (RAP) because components (products) are purchased under an all unit discount (AUD) policy and then installed on a series-parallel system. The total budget available for purchasing the components, the storage space, the vehicle capacities, and the total weight of the system are limited. Moreover, a penalty function is used to penalize infeasible solutions, generated randomly. The overall goal is to find the optimal number of the components purchased for each subsystem so that the total costs including ordering cost, holding costs, and purchasing cost are minimized while the system reliability is maximized, simultaneously. A non-dominated sorting genetic algorithm-II (NSGA-II), a multi-objective particle swarm optimization (MOPSO), and a multi-objective harmony search (MOHS) algorithm are applied to obtain the optimal Pareto solutions. While no benchmark is available in the literature, some numerical examples are generated randomly to evaluate the results of NSGA-II on the proposed IRAP. The results are in favor of NSGA-II.

1. Introduction

Redundancy allocation problems (RAPs) are commonly observed in complex telecommunication, safety, transportation, satellite, and electrical power systems where strict system reliability requirements are needed [22]. Several RAP structures and configurations are considered in the literature for series [19], parallel [4], series-parallel [11,12], and k-out-of-n [7] systems among others. Series-parallel RAP systems are generally more common than other types [11,12,21,24,9].

Traditionally, the RAP literature has focused heavily on system reliability with little attention to the preparation and transformation of the system components. However, there are some studies in the literature which have considered the production process of the components prior to system installation. Sadeghi et al. [19] considered a two-objective vendor managed inventory and redundancy allocation problem in a vendor-retailer supply chain problem. The goal of the problem is to optimize the number of machines working in series to manufacture a single product. Xie et al. [23] optimized the system reliability of a repairable k-out-of-n RAP where the spare parts inventory in addition to the redundancy allocation of components were taken into account.

In this study, a multi-objective series-parallel RAP is modeled in a multi-component multi-period inventory control problem. The multi-product multi-period inventory problem is a common problem often studied in the literature. Mousavi et al. [14] formulated a multi-item multi-period inventory control problem in which both all unit and incremental quantity discounts, in addition to interest and inflation factors were considered. In addition, Mousavi et al. [15] modeled a fuzzy multi-product multi-period inventory control problem with back-order and lost sales, where the budget was restricted and the items were delivered in pre-specified boxes. Gholamian et al. [5] considered a multi-objective multi-period multi-site multi-product inventory problem in a fuzzy multiple-echelon supply chain problem where two different linearization methods were applied to solve the proposed problem. Ghoniem and Maddah [6] studied a multi-product multi-period selling horizon pricing-inventory control problem which was modeled as a mixed-integer nonlinear mathematical programming. Mousavi et al. [13] formulated a multi-periodic multi-item inventory

* Corresponding author.
E-mail addresses: najme.alikar@gmail.com (N. Alikar), mousavi.mohsen8@gmail.com, mousavi@qiau.ac.ir (S.M. Mousavi), r_ariffin@um.edu.my (R.A. Raja Ghazilla), tavana@lasalle.edu (M. Tavana), olugu@um.edu.my (E.U. Olugu).

http://dx.doi.org/10.1016/j.ress.2016.10.023
Received 19 February 2016; Received in revised form 5 October 2016; Accepted 30 October 2016
Available online 01 November 2016
0951-8320/

control problem modeled as a mixed integer problem. Shortages were allowed and in case of shortage, a fraction of demand was considered as backorders and a fraction as lost sales. Pasandideh et al. [17] proposed a mixed-integer nonlinear mathematical programming model for a seasonal multi-product multi-period inventory control problem in which the total budget and the total storage space were restricted. They used the genetic algorithm and memetic algorithm to optimize their proposed problem. In this study, a non-dominated sorting genetic algorithm-II (NSGA-II) along with MOPSO and MOHS are utilized to solve the proposed IRAP.

Many researchers have studied series-parallel RAPs. Mousavi et al. [12] studied a fuzzy multi-state RAP in a fuzzy environment where the homogenous components were installed in each subsystem. Yeh [24] used an orthogonal simplified swarm optimization to solve a series-parallel RAP with a mix of components in which the aim was to maximize the system reliability. Soltani et al. [21] solved a series-parallel redundancy allocation problem with uncertainty in the reliability of the components. Hsieh and Yeh [9] employed an artificial bee colony algorithm to optimize a series-parallel RAP where a penalty strategy was used to remove the equalities in the constraints and also to find the infeasible solutions, generated randomly. Mousavi et al. [11] considered an improved fruit fly optimization algorithm to solve a series-parallel RAP in a fuzzy environment in which identical components were allowed to be installed within each subsystem. The aim was to obtain the optimal number of identical components in each subsystem so that the system reliability was maximized. Cao et al. [1] provided a multi-objective series-parallel RAP where an exact method i.e. a decomposition-based approach was applied to solve the problem. Khalili-Damghani and Amiri [10] used both the epsilon-constraint approach and data envelopment analysis to solve a binary-state multi-objective series-parallel RAP in which an epsilon-constraint was employed to generate Pareto fronts.

Multi-objective RAP has been of interest to many researchers in the recent literature. Zhang and Chen [25] provided a multi-objective RAP with imprecise components in an interval environment where the total cost and the system reliability were optimized simultaneously. They solved their problem using an improved multi-objective particle swarm optimization (MOPSO) algorithm. Mousavi et al. [12] solved a multi-objective multi-state series-parallel RAP in a fuzzy environment where the components in each subsystem were selected from the identical type. They applied two controlled elitism non-dominated ranked genetic algorithms and a non-dominated sorting genetic algorithm (NSGA-II) to optimize their problem. Ghorabaee et al. [7] utilized NSGA-II to solve a bi-objective k-out-of-n RAP in which both objectives of the total cost and the system reliability were optimized simultaneously. Dolatshahi-Zand and Khalili-Damghani [2] used a multi-objective RAP to design a SCADA water source management control center where both MOPSO and TOPSIS algorithms were applied. Garg et al. [3] modeled a fuzzy multi-objective series-parallel RAP where a MOPSO and a bi-objective particle swarm optimization were employed to solve the proposed RAP. Cao et al. [1] used NSGA-II to solve a decomposition-based approach of multi-objective RAP for series-parallel systems. Ghorabaee et al. [8] solved a *k*-out-of-*n* multi-objective series-parallel RAP using NSGA-II where the objectives were to optimize both the total cost and the system reliability simultaneously. Rahmati et al. [18] optimized a multi-objective location model within a multi-server queuing framework in which facilities were based on M/M/m queues where both NSGA-II and NRGA algorithms were applied to solve the problem.

There is almost no research in the literature which considers both the inventory and transportation components of a series-parallel RAP. The main contribution of this work is to fill this gap. A bi-objective binary-state RAP is formulated in a multi-component (product) multi-period inventory control problem with restrictions on the total storage space, transportation vehicle capacity and the total budget available to purchase the components. To solve the proposed IRAP, a NSGA-II algorithm is derived where both MOPSO and MOHS are also used to validate the results obtained by the NSGA-II.

A real world example of the proposed problem would be an airplane where several engines are connected in series, each containing parts which are installed in parallel (see Feller 1950). The components of each engine work in parallel, meaning that the engine subsystem operates if at least one of the components operates. The components are manufactured by a wide range of companies storing their products in a special warehouse and are sold to satisfy the required demand. The manufacturer determines the price and reliability for each component. It is assumed that the airplane manufacturing companies order the identical type of the components for each engine in order to profit from the discount provided by the manufacturer.

Another real world example is to consider the case of the engines installed on each wing of an airplane as a subsystem. The components (engines) of each wing work in parallel, meaning that the engine subsystem operates if at least one of the engines of a wing operates. In order to make sense of the contribution of the current study, a comparison of the current work with the recent literature is shown in Table 1.

The rest of the paper is organized as the follows. The next section describes the notations, parameters and the decision variables applied in this article to formulate the IRAP under investigation. In Section 3, the proposed IRAP is explained and modeled. The explanation of the NSGA-II, MOPSO and MOHS algorithms comes in Section 4. In Section 5, some numerical illustrations are generated to evaluate the performance of the provided algorithms on the IRAP. The conclusions and recommendations for future research are given in Section 6.

2. Notations, parameters and decision variables

The indices, notations, parameters and the decision variables employed to formulate the IRAP are described as follows:

Indices:

i=1, 2,..., I is the index of the subsystem.

j=1, 2,..., J is the index of the components (products).

t=0, 1,..., T is the index of the time periods.

m=1, 2,..., M is the index of the price break-point.

Notations:

h_{ijt}: Inventory holding cost per unit of the j^{th} component for subsystem i in period t

A_{ijt}: Ordering cost (transportation cost) per unit of the j^{th} component for subsystem i in period t

c_{ijtm}: Purchasing cost per unit of the j^{th} component for subsystem i at the m^{th} price break-point in period t

s_{ijt}: The required warehouse space to store per unit of the j^{th} component for subsystem i in period t

S: The available capacity of the warehouse.

T_t: Total time elapsed up to and including the t^{th} replenishment cycle of components.

e_{ijtm}: m^{th} price break-point for purchasing the j^{th} component for subsystem i in period t ($e_{ijkt1} = 0$).

r_{ij}: Reliability of component j purchased for subsystem i

w_{ij}: Weight of component j purchased for subsystem i

R: The system reliability.

x_{ijt}: The initial (remained) positive inventory of the j^{th} component purchased for subsystem i in period t ($x_{ij1} = 0$) (decision variable).

d_{ijt}: The demand quantity of component j purchased for subsystem i in period t (in the second model).

O_t: The capacity of the transportation vehicle in period t.

W: The total system weight.

V: The upper bound for each order quantity.

Decision variables:

Q_{ijt}: Ordering quantity of j^{th} component purchased for subsystem i in period t

y_{ijtm}: A binary variable that is set to 1 if component j is purchased for

Table 1
Recent literature most related to this study.

Reference	Inventory costs	System reliability	Objective function	Constraints	Discount policy	Solving methodology
[12]	–	Multi-state Series-parallel	Max Reliability, Min Cost	Weight, Budget	AUD, IQD	CE-NRGA, NSGA-II
[11]	–	Series-parallel	Max Reliability	Weight, Cost	AUD, IQD	Fruit fly optimization
[14]	Multi-item multi period inventory	–	Min Cost	Storage, Cost, Packets	AUD, IQD	GA and SA
[7]	–	k-out-of-n series	Max Reliability, Min Cost	Weight	–	NSGA-II
[21]	–	Series-parallel	Max Reliability	Weight, Cost	–	Robust optimization, Monte Carlo
[19]	Production-inventory machines	Series machines	Max Reliability, Min Cost	Warehouse, Inventory, Replenishment	AUD, IQD	NSGA-II, NRGA
[23]	Spares logistics	Repairable Series system	Max Availability	–	–	Markov chain, Branch & bound
[4]	Production-inventory machines	Series-parallel machines	Max Reliability, Max min production capacity	Cost	–	Robust optimization
[6]	Pricing inventory	–	Max Profit	Ordering, Pricing	–	Lingo
[13]	Multi-item multi-period inventory	–	Min cost	Budget, Warehouse, Ordering, Production	AUD, IQD	PSO, GA
[17]	Multi-item multi-period inventory	–	Min cost	Budget, Warehouse, Ordering, Production	AUD, IQD	GA, Memetic algorithm
[24]	–	Series-parallel	Max Reliability	Weight, Cost	–	Simplified swarm optimization
[9]	–	Series-parallel	Min Cost	Reliability, Consumer demands	–	Bee colony
[1]	–	Series-parallel	Max Reliability, Min Cost, Min Weight	Number of components	–	Decomposition
[10]	–	Series-parallel	Max Reliability, Min Cost, Min Weight	Weight, Cost, Number of components	–	ε-Constraint, DEA
[25]	–	Series-parallel	Max Reliability, Min Cost	Weight, Cost	–	PSO
[2]	–	Series-parallel	Max Reliability	Weight, Cost, Number of components	–	ε-Constraint
[3]	–	Series-parallel	Max Reliability, Min Cost	Weight, Cost	–	PSO
[18]	–	–	Min travel, Min max idle time, Min budget	Open facility, Service, nearest facility, Waiting time, Integer restriction	–	Harmony search, NSGA-II, NRGA
This paper	**Bi-Objective Multi-Period Inventory**	**Series-Parallel RAP**	**Max Reliability, Min Cost**	**Budget, Weight, Transportation, Production, Warehouse**	**AUD, IQD**	**NSGA-II, MOPSO, MOHS**

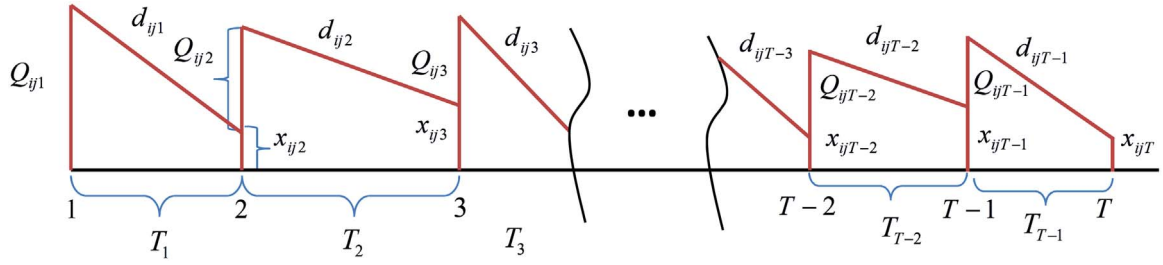

Fig. 1. A presentation of some scenarios of the proposed inventory problem.

subsystem i at m^{th}price break-point in period t, and set to 0 otherwise. λ_{ijt}: A binary variable that is set to 1 if component j is ordered for subsystem$i$ in period t, and set to 0 otherwise.

3. The proposed IRAP

First, the definition of the problem is described and then the mathematical modeling of the problem is formulated.

3.1. The problem definition

In this study, a mixed-integer nonlinear mathematical programming is considered to formulate a bi-objective binary-state series-parallel RAP in a multi-component multi-period inventory. The manufacturer produces the different components first and then stores them given a pre-specified space capacity. The owner of the series-parallel system purchases the components from the manufacturer under an AUD strategy where the total available budget is limited. The components are transferred from the storage to the owner of the system using the vehicles, with a limitation on their capacity where the vehicles have different capacities in each time-period. The aim of the problem is to determine the optimal number of components for each subsystem so that the total inventory cost is minimized and the system reliability is maximized at the same time. A multi-objective version of the genetic algorithm based on the Pareto front concept i.e. NSGA-II is applied to solve the proposed model. To validate the results obtained by NSGA-II, the MOPSO and MOHS are also derived. Fig. 1 shows some scenarios of the proposed inventory control problem. In the first period of Fig. 1, the order quantity (Q_{ij1}) arrives and is consumed by customers with the demand rate of d_{ij1}. As a result, the remaining inventory in the first period (x_{ij2}) plus Q_{ij2} are stored for the next period until the demand rate d_{ij2} is met. The description for the remaining periods is the same as for the first period.

The series-parallel system configuration proposed in this study is depicted in Fig. 2. As shown in these figures, a quantity Qijt of component ni is ordered according to the process described in Fig. 1.

3.2. The bi-objective mathematical model

The mathematical model of the developed IRAP is formulated by Eq. (1) as follows:

$$MinTC = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T} Q_{ijt}A_{ijt}\lambda_{ijt} + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T-1}((Q_{ijt}+x_{ijt})+x_{ijt+1})T_{ijt}h_{ijt}/2 + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{m=1}^{M} Q_{ijt}c_{ijtm}y_{ijtm} \quad MaxR = \prod_{i=1}^{I}\left(1-\prod_{j=1}^{J}(1-r_{ij})^{\sum_{t=1}^{T-1}Q_{ijt}}\right) \tag{1}$$

Subject to:

$$x_{ijt+1} = x_{ijt} + Q_{ijt} - d_{ijt}T_{ijt} \tag{1-1}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T}(Q_{ijt}+x_{ijt})s_{ijt} \le S \text{ Total storage capacity} \tag{1-2}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T}\sum_{m=1}^{M} Q_{ijt}c_{ijtm}y_{ijtm} \le B \text{ The total available budget} \tag{1-3}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{t=1}^{T} Q_{ijt}\lambda_{ijt}w_{ij} \le W \text{ Total weight} \tag{1-4}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J} Q_{ijt}\lambda_{ijt} \le O_t \text{ Transportation capacity} \tag{1-5}$$

$$Q_{ijt} \le V \tag{1-6}$$

$$x_{ijt} \ge 0;\ Q_{ijt} \ge 0;\ Q_{ijt},\ x_{ijt} \in Z$$

We consider two objectives in Eq. (1): minimizing the total cost and maximizing the total system reliability. The first objective is intended to

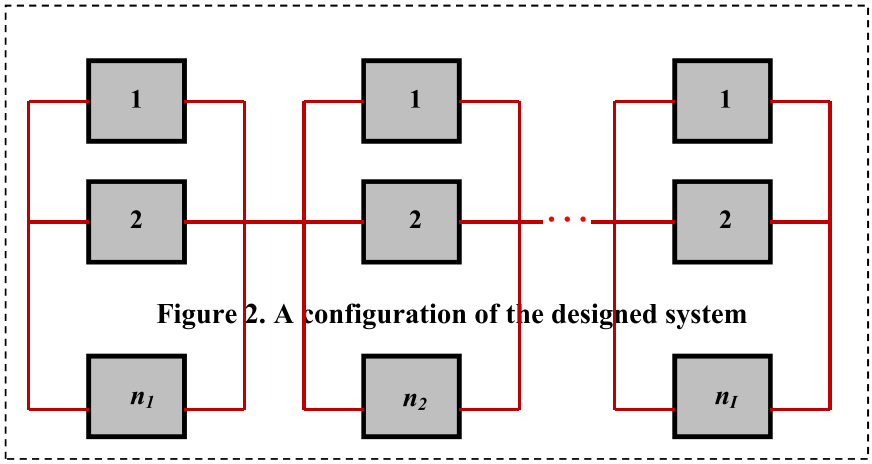

Figure 2. A configuration of the designed system

Fig. 2. A configuration of the designed system.

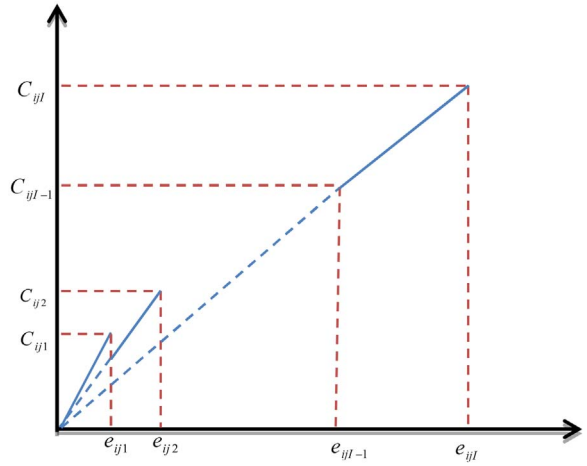

Fig. 3. The all-unit discount policy to purchase component.

minimize the total inventory costs including the ordering, holding, and the purchasing costs. The total holding cost is calculated by aggregating the trapezoids in Fig. 1 times their related holding cost. The second objective is to maximize the total system reliability (R) where $r_{ij}$is the reliability of component j purchased for subsystem i. Eqs. (1–1) defines the number of components j purchased for subsystem i remaining from the previous time-period in storage. Eqs. (1–2) models the total storage space available for storing the components. Eqs. (1–3) depicts the total available budget to purchase the components resulting from an AUD as shown in Fig. 3. The total weight that the system can sustain is defined by Eq. (1–4). The total transportation vehicle capacity in each period is formulated by Eqs. (1–5). Note that different vehicles with different capacities are used to transfer the components. The order quantity of each component purchased for each subsystem in each period is limited due to some production restrictions which are provided by Eq. (1–6).

To clarify Eq. (1), the aim of the problem is to find out the optimal order quantity of each component for each subsystem in each period so that the total system costs are minimized and the system reliability is maximized. The ordered components are transferred by the different trucks with the different capacity in each period (Eqs. (1–5)) to the storage area which is limited by a specific capacity (Eqs. (1–2)). The owner of system allocates a fixed amount of budget to buy the components to be installed on the system (Eqs. (1–3)). The components are installed on each subsystem based on the demand requested in each period while the remaining components are stored along with the components ordered for the next period (Eqs. (1–1)). The system has a pre-specified weight and the number of components for each subsystem are chosen subject to the total system weight (including the component weights) while the weight of the subsystems do not exceed this pre-specified weight (Eq. (1–4)). Furthermore, the supplier of the components is able to produce as much as its production capacity (Eqs. (1–6)) allows. The model proposed in Eq. (1) has been formulated under the following assumptions:

- The planning horizon is finite.
- The components are purchased under AUD and then installed on a series-parallel system.
- The components are delivered with the vehicles with different capacities.
- The lead time is considered to be zero.
- The order quantity is limited for each component in each period.
- There is no supply constraint for the components.
- Failed components are not repairable.
- The reliability, weight, and cost of all components are deterministic and known.
- The number of subsystems is fixed.
- Shortages are not allowed.

4. The solution algorithms

In order to solve the proposed multi-objective problem, a NSGA-II algorithm is applied to obtain the optimal order quantities of the components purchased for each subsystem in each period. Both the MOPSO and MOHS algorithms are also utilized to evaluate the results of the NSGA-II.

4.1. The NSGA-II

The steps of the NSGA-II employed in this research are explained as follows:

4.1.1. Initializing the parameters and chromosomes

First, the algorithm parameters i.e. the number of populations (*NoP*), the number of generations (*NoG*), the probability of crossover (*PoC*) and the probability of mutation (*PoM*) are initialized. Next, in order to initialize the solutions, we generate the value of Q_{ijt}(for $i=1, 2, ..., I$; $j=1, 2,..., J$; $t=0, 1,..., T$) randomly in the range [0, V]. The rest of the decision variables as well as $y_{ijtm}$and $\lambda_{ijt}$are automatically initialized according to the values chosen for Q_{ijt}. Fig. 4 shows a chromosome structure of the NSGA-II.

4.1.2. Evaluating the solutions

Both objectives formulated in Eq. (1) are used to evaluate each chromosome initialized in the previous step. Note that the infeasible solutions, i.e. those violating the restrictions of Eq. (1), are penalized by the following penalty functions:

$$\begin{cases} F_1(x) + (Z - f(x))^{\alpha} \\ F_2(x)/(Z - f(x))^{\beta} \end{cases} \quad (2)$$

where $F_1(x)$and $F_2(x)$are the first and second objectives in Eq. (1) respectively, $f(x)$and Z are the left and right hand sides of a constraint. $\alpha$and $\beta$are the penalty coefficients respectively.

4.1.3. Fast non-dominated sorting process

In this step, the *NoP* populations that were generated in the previous steps are compared and sorted. To do this, all the solutions in the first non-dominated front are first found where the solutions are chosen using the concept of domination. The concept of domination states that solution$X_i$is said to dominate solutionX_j, if $\forall$ e $\in$ {1, 2}we have $f_e(X_i) \le f_e(X_j)$ for $e \in \{1, 2\}$ such that $f_e(X_i) < f_e(X_j)$. In this case, we say that $X_i$is the non-dominated solution within the solution set $\{X_i, X_j\}$. Otherwise, it is not. In addition, 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.

4.1.4. Crowding distance

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

Q_{111}	Q_{112}	...	Q_{ijt}	...	Q_{IJT}	y_{1111}	...	y_{IJTM}	λ_{111}	...	λ_{111}

Fig. 4. The chromosome representation.

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 the end solutions in each front ($m_t = 1, T_t$) are $d_1 = d_{T_t} = \infty$

4: The crowding distance for the objective functions $f_{1,2}$ for $m_t = 2,...,T_t - 1$ is calculated by $d_{m_t} = d_{m_t} + (f_{1,2} - f_{1,2})$

Fig. 5. A pseudo code to determine the crowding distance.

(m_t = 1, 2, … , T_t) be the value of the crowding distance for the solutionm_t. Then, the crowding distance is obtained using the steps proposed in Fig. 5. Finally, populations are sorted based on their ranks and crowding distances.

4.1.5. Selection process

In this study, a two-chromosome tournament selection operator is used to find the best solutions for the next generation in which two chromosomes are selected randomly and then compared in terms of the front rank and the crowding distance. The one with the least front rank is chosen. In the case of the front ranks being equal, the one with the maximum crowding distance will be chosen.

4.1.6. Crossover and mutation operators

First generate a random value between 0 and 1. If the value is less than *PoC,* the crossover operator will be done, otherwise the mutation operator will be performed. If Q_1 and Q_2 are two chromosomes (parents) selected randomly, the crossover operator works as follows:

$$Q_1^{'} = Round(Q_1\lambda + Q_2(1-\lambda))$$
$$Q_2^{'} = Round(Q_1(1-\lambda) + Q_2\lambda) \quad (3)$$

In Eq. (3), $Q_1^{'}$ and $Q_2^{'}$are offspring,λ is a random variable in the range of [0,1] and *Round* Signifies that the value must be an integer. Note that if the values of the offspring are not in the range [0, V], the crossover operators will be done for different values of λ until the offspring values fall in the range.

A one-point mutation operator is also accomplished in this study as two genes with different values are selected randomly and replaced.

4.1.7. Combining the populations

In this step, both the *NoP* solutions and the new solutions (after updating the population according to their ranks and crowding distance) are combined to generate a population of size (2×*NoP*) to ensure elitism. Then, (2×*NoP*) populations are evaluated and sorted based on their ranks and crowding distance. Finally, according to the sorted solutions, the *NoP* solutions are selected as initial population in the next generation. Fig. 6 demonstrates the approach taken in NSGA-II to combine the populations and to select the *NoP* solutions for the next generation.

4.2. The MOPSO

The MOPSO is briefly explained as follows:

- Initializing the parameters i.e. the number of populations (*nop*), the number of generations (*nog*), and two parameters C_1 and C_2, as well as the articles generated randomly in the range.
- Evaluating the objectives modeled in Eq. (1) for each article of the population.
- Fast non-dominated sorting process wherein *nop* articles are compared and ranked as well as the process explained in step 3 of the NSGA-II.
- Crowding distance is also done the same as in step 4 of the NSGA-II.
- The articles of the population are updated as the position and velocity, two variables in the MOPSO algorithm, are initialized using Eqs. (4) and (5), respectively.

$$v_{n+1}^{l} = \omega.\ v_n^l + C_1.\ u_1.\ (pBest_n^l - z_n^l) + C_2.\ u_2.\ (gBest_n - z_n^l) \quad (4)$$

$$z_{n+1}^{l} = z_n^l + \eta.\ v_{n+1}^l \quad (5)$$

In Eq. (4), $u_1$and u_2 are two numbers generated randomly in the interval (0, 1), the coefficients $C_1$and $C_2$are the given acceleration constants towards *pBest* and *gBest*, respectively, and w is the inertia weight which is expressed as Eq. (6) [16]. Furthermore, $pBest_n^l$ and $gBest_n$ are the best fitness value for particle l until time *n*, (n =1, 2,..., *nog*) and the best particle over all until time *n*, respectively.

$$v = v_{\max} - \frac{(v_{\max} - v_{\min})}{nog}.\ n \quad (6)$$

In Eq. (6), *nog* is the maximum number of iterations and *n* is the current number of iterations. [20] and [16] have claimed the best result will be obtained since [v_{min}, v_{max}] = [0. 4, 0. 9].

- Combining the populations is done as well as step 8 mentioned in NSGA-II.

4.3. The MOHS

The MOHS applied in this work is briefly described as follows:

a) Parameter and solution initialization: the parameters of the MOHS are the harmony memory size (*HMS*), the harmony memory considering rate (*HMCR*), the pitch adjusting rate (*PAR*), and the number of generations (*NoG*).
b) Evaluating the objectives modeled in Eq. (1) for each solution of the population *HMS*.
c) Fast non-dominated sorting process wherein *HMS* solutions are compared and ranked as well as the process explained in step 3 of the NSGA-II.
d) Crowding distance is also done the same as in step 4 of the NSGA-II.
e) New harmony improvisation which includes random selection [15], HM consideration, and pitch adjustment.
f) Harmony memory update [15].
g) Combining the populations.

5. Results and discussion

Because this study proposes a novel model, no benchmark is available in the literature to evaluate the NSGA-II performance on the proposed problem. Therefore, ten numerical examples are gener-

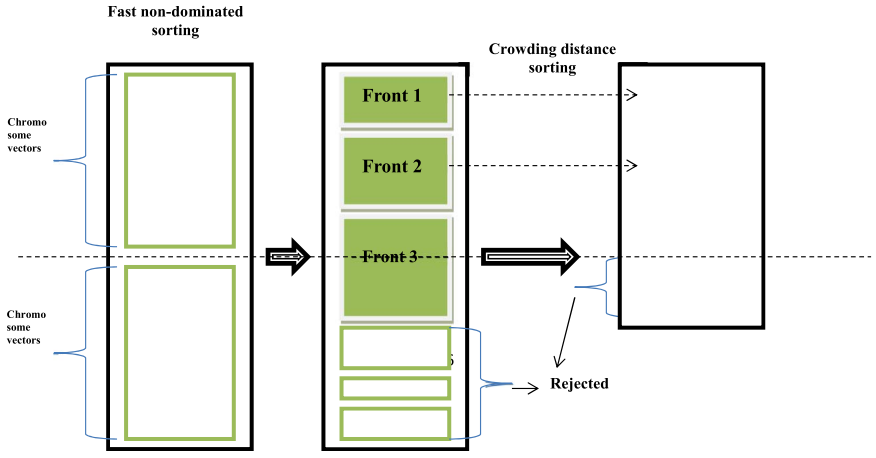

Fig. 6. The NSGA-II procedure.

Table 2
The input data of the generated numerical examples.

Problem No.	(I, J, T)	d	h	s	w	r	O	A	W	S	B	T	c	Q
1	(2, 2, 2)	[1,4]	[1,8]	[1,5]	[3,8]	[0.4, 0.8]	[200,300]	[1,5]	[200,300]	[200,300]	[200,300]	[1,2]	[1,5]	[1,10]
2	(3, 2, 2)	[1,4]	[1,8]	[1,5]	[3,8]	[0.4, 0.8]	[500,800]	[1,5]	[200,300]	[500,800]	[500,800]	[1,2]	[1,5]	[1,10]
3	(2, 3, 3)	[1,4]	[1,8]	[1,5]	[3,8]	[0.05, 0.25]	[800,1000]	[1,5]	[300,500]	[800,1000]	[800,1000]	[1,2]	[1,5]	[1,10]
4	(3, 1, 2)	[1,4]	[1,8]	[1,5]	[3,8]	[0.45, 0.6]	[300,500]	[1,5]	[100,200]	[200,300]	[200,300]	[1,2]	[1,5]	[1,10]
5	(3, 3, 4)	[1,4]	[1,8]	[1,5]	[3,8]	[0.05, 0.15]	[1000,1300]	[1,5]	[900,1000]	[900,1000]	[1000,1300]	[1,2]	[1,5]	[1,10]
6	(4, 1, 2)	[1,4]	[1,8]	[1,5]	[3,8]	[0.25, 0.55]	[300,500]	[1,5]	[300,500]	[300,500]	[300,500]	[1,2]	[1,5]	[1,10]
7	(4, 2, 2)	[1,4]	[1,8]	[1,5]	[3,8]	[0.25, 0.55]	[300,500]	[1,5]	[300,500]	[300,500]	[300,500]	[1,2]	[1,5]	[1,10]
8	(5, 2, 2)	[1,4]	[1,8]	[1,5]	[3,8]	[0.25, 0.55]	[300,500]	[1,5]	[300,500]	[300,500]	[300,500]	[1,2]	[1,5]	[1,10]
9	(6, 3, 3)	[1,4]	[1,8]	[1,5]	[3,8]	[0.03, 0.15]	[1500,1600]	[1,5]	[1200,1500]	[1200,1500]	[1200,1500]	[1,2]	[1,5]	[1,10]
10	(4, 4, 4)	[1,4]	[1,8]	[1,5]	[3,8]	[0.03, 0.15]	[1800,1900]	[1,5]	[1800,2000]	[1800,1900]	[1800,1900]	[1,2]	[1,5]	[1,10]

Table 3
Parameters used for the NSGA-II.

Parameters	Values
NoP	30
PoC	0.9
PoM	0.1
NoG	500

ated randomly for which the data input for the parameters are given in Table 2. A large number of runs tune the algorithm parameters in NSGA-II. The parameters used for NSGA-II and their values are given in Table 3.

In the multi-objective meta-heuristic algorithms, both the concepts of diversity (the widespread distribution of solutions along the Pareto front) and convergence (finding the Pareto-optimal set as close as possible to the true Pareto front) should be taken into account. Thus, in this research in order to compare the results of both algorithms, these two concepts are considered as well. In this study, the three algorithms i.e. NSGA-II, MOPSO and MOHS are compared in terms of the number of Pareto fronts (NP) and the CPU time taken to run the algorithms. Table 4 shows the results of the NSGA-II, MOPSO and MOHS algorithms in terms of NP and CPU for all ten generated numerical examples. The averages of the results depict that the NSGA-II has better performance than MOPSO and MOHS in terms of NP and CPU. Figs. 7 and 8 and represent the NP and CPU results generated by the two algorithms for the numerical examples respectively.

Figs. 9–18 show the Pareto fronts of the NSGA-II, MOPSO and MOHS for the ten generated numerical examples where the solutions with the red, blue, and green colors belong to the NSGA-II, MOPSO, and MOHS algorithms, respectively. In almost all of the cases, NSGA-II outperforms MOPSO and MOHS in terms of the convergence and diversity concepts.

Table 4
Comparison results of both algorithms for the generated numerical examples in terms of NP and CPU.

Problem No.	NSGA-II		MOPSO		MOHS	
	NP	CPU	NP	CPU	NP	CPU
1	21	23.646	16	34.765	19	42.831
2	26	25.781	18	35.632	17	43.114
3	13	25.981	12	36.385	9	43.312
4	13	26.434	11	36.872	13	43.924
5	17	25.895	17	35.703	20	43.201
6	12	27.457	14	38.293	16	44.764
7	16	27.365	14	38.373	15	44.813
8	19	26.809	12	35.847	16	42.473
9	20	28.742	19	39.073	17	45.056
10	20	29.342	19	39.684	14	45.203
Ave.	**17.7**	**26.745**	**15.2**	**37.062**	**15.6**	**43.869**

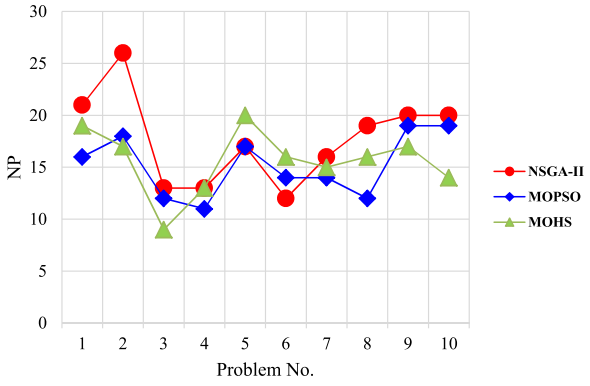

Fig. 7. Graphical representation of the NP results generated by the algorithms for the numerical examples.

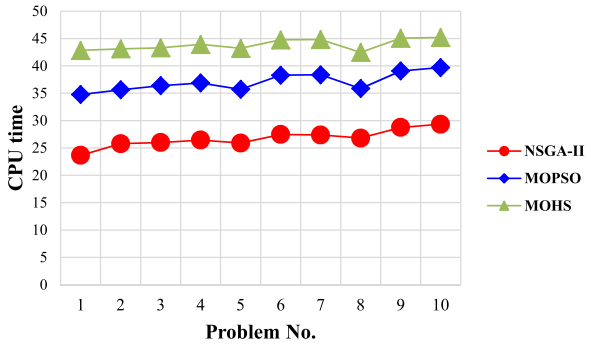

Fig. 8. Graphical representation of the CPU results generated by the algorithms for the numerical examples.

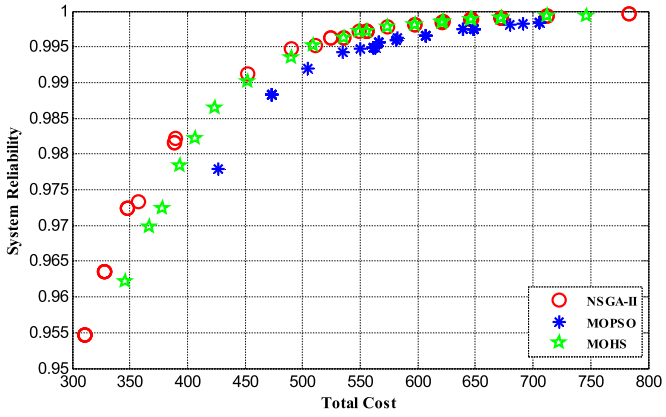

Fig. 9. Pareto fronts obtained by the algorithms for Problem No. 1.

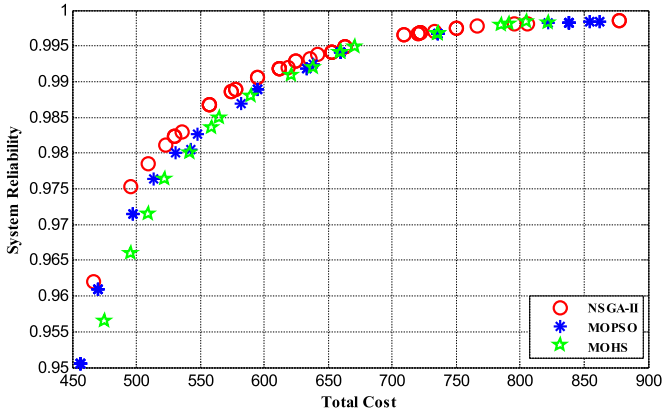

Fig. 10. Pareto fronts obtained by the algorithms for Problem No. 2.

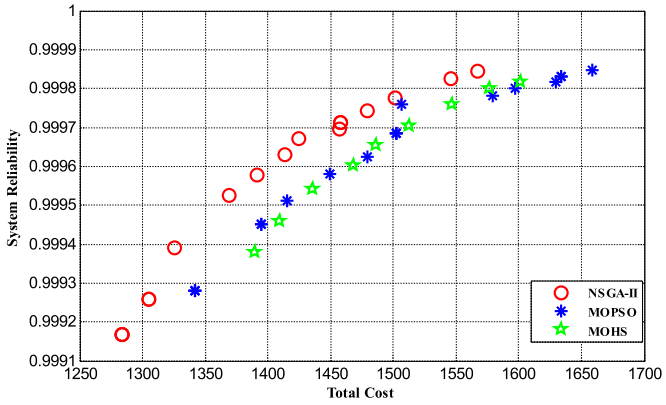

Fig. 11. Pareto fronts obtained by the algorithms for Problem No. 3.

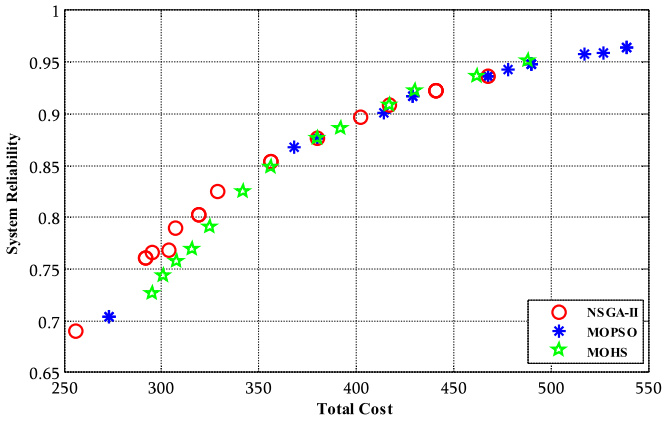

Fig. 12. Pareto fronts obtained by the algorithms for Problem No. 4.

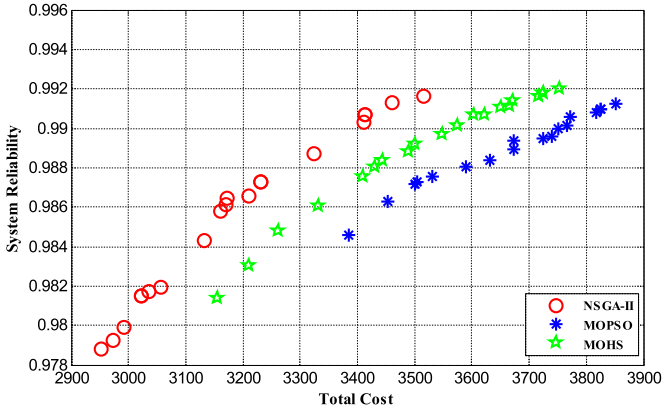

Fig. 13. Pareto fronts obtained by the algorithms for Problem No. 5.

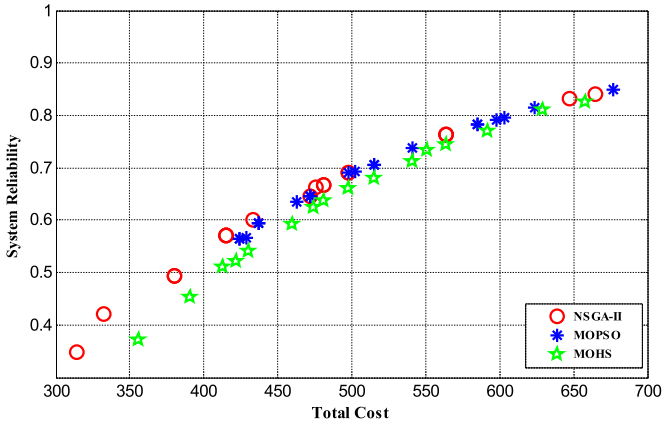

Fig. 14. Pareto fronts obtained by the algorithms for Problem No. 6.

Tables 5, 6 display the optimal Pareto fronts resulted by the NSGA-II, MOPSO and MOHS algorithms for Problem No. 7 and Problem No. 8 respectively. Figs. 19 and 20 depict box-plots of the NSGA-II, MOPSO and MOHS algorithms for both NP and CPU obtained from the generated numerical examples respectively where the results are in favor of NSGA-II.

6. Conclusions and recommendations for future research

In this article, a multi-component multi-period binary-state IRAP model was formulated in a series-parallel system in which shortages were not allowed and the components were purchased under an AUD policy. The total available budget to purchase the components, the storage space and the truck capacities were restricted. In order to solve

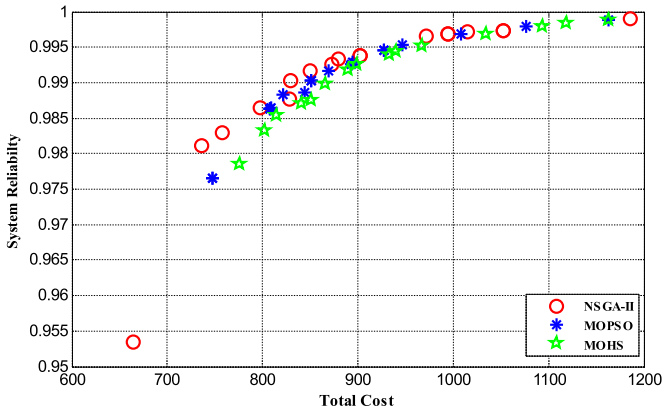

Fig. 15. Pareto fronts obtained by the algorithms for Problem No. 7.

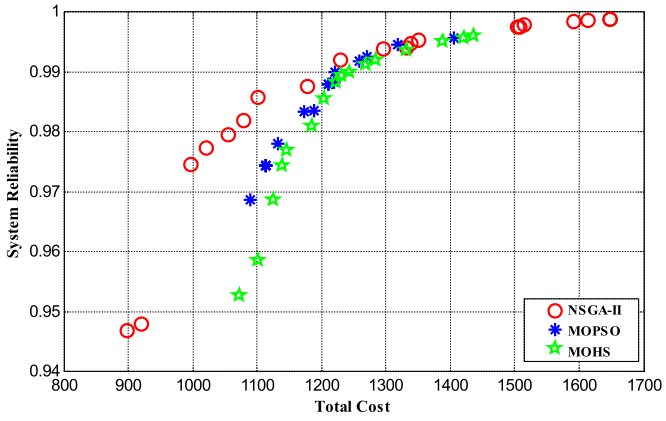

Fig. 16. Pareto fronts obtained by the algorithms for Problem No. 8.

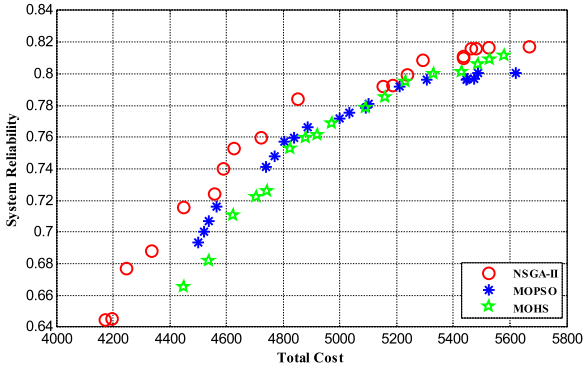

Fig. 17. Pareto fronts obtained by the algorithms for Problem No. 9.

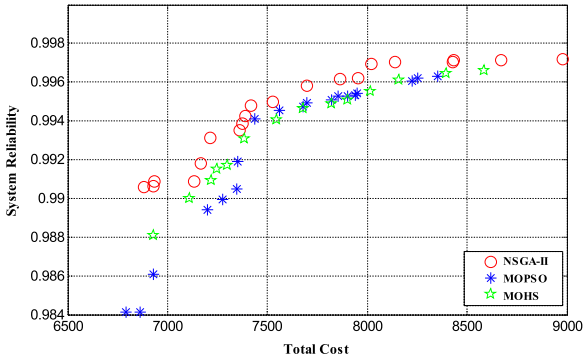

Fig. 18. Pareto fronts obtained by the algorithms for Problem No. 10.

Table 5
Pareto fronts generated by the three algorithms for Problem No. 7.

NSGA-II		MOPSO		MOHS	
Cost	**Reliability**	**Cost**	**Reliability**	**Cost**	**Reliability**
798	0.986	893.5	0.992	890	0.991
849.5	0.991	850.5	0.990	841	0.987
829.5	0.990	807.5	0.986	1119	0.998
971.5	0.996	869.5	0.991	865.5	0.989
757.5	0.982	927.5	0.994	932.5	0.994
879.5	0.993	894.5	0.992	899	0.992
1052.5	0.997	747.5	0.976	776	0.978
828.5	0.987	821.5	0.988	967.5	0.995
1185.5	0.999	1076.5	0.997	802.5	0.983
1119	0.998	1163.5	0.998	815	0.985
664	0.953	1008.5	0.996	939.5	0.994
872.5	0.992	808.5	0.986	1093	0.997
994.5	0.996	946.5	0.995	1163	0.998
902.5	0.993	844.5	0.988	1034	0.996
735.5	0.981	–	–	850.5	0.987
1014.5	0.997	–	–	–	–

Table 6
Pareto fronts generated by the three algorithms for Problem No. 8.

NSGA-II		MOPSO		MOPSO	
Cost	**Reliability**	**Cost**	**Reliability**	**Cost**	**Reliability**
1504	0.997	1090	0.968	1101	0.958
1613.5	0.998	1114	0.974	1139	0.9743
1648.5	0.998	1210	0.988	1232	0.989
1514.5	0.997	1219	0.988	1285	0.992
1339.5	0.994	1133	0.978	1185	0.981
1351	0.995	1259	0.991	1268	0.991
1591.5	0.998	1271	0.992	1435.5	0.995
1021	0.977	1173.5	0.983	1421	0.995
920.5	0.948	1188.5	0.983	1223	0.988
1101.5	0.985	1405.5	0.995	1146	0.977
997.5	0.974	1318	0.994	1125	0.968
1331.5	0.994	1221	0.990	1204	0.985
1296.5	0.993	–	–	1389	0.995
1055.5	0.979	–	–	1332	0.993
1507.5	0.997	–	–	1243	0.990
897.5	0.946	–	–	1072	0.952
1178.5	0.987	–	–	–	–
1078.5	0.981	–	–	–	–
1229	0.992	–	–	–	–

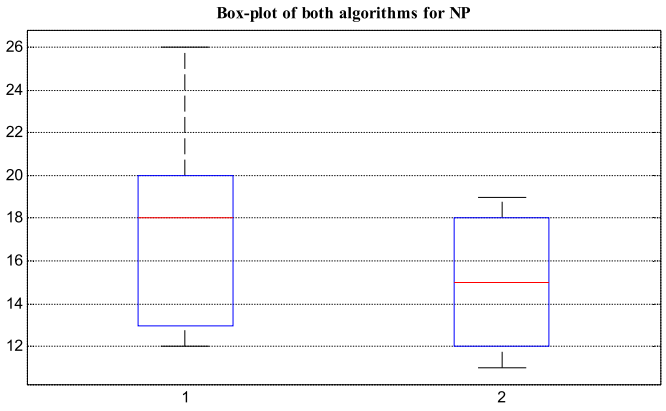

Fig. 19. Box-plot of the NSGA-II (1) and MOPSO (2) algorithms for NP obtained from the generated numerical examples.

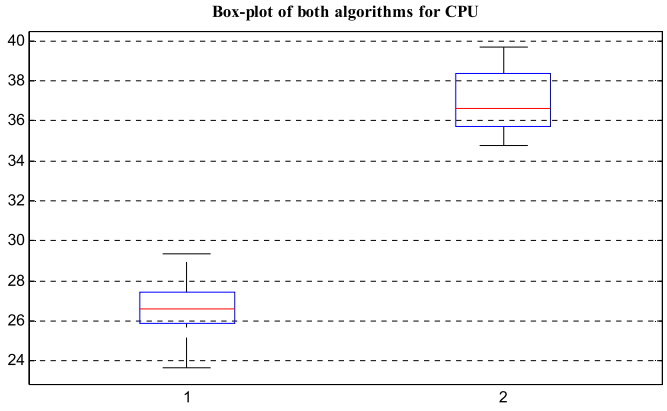

Fig. 20. Box-plot of the NSGA-II (1) and MOPSO (2) algorithms for CPU obtained from the generated numerical examples.

the proposed problem, a NSGA-II algorithm was derived to find out the order quantities of the components so that the total inventory cost was minimized. Furthermore, a MOPSO and a MOHS were used to evaluate the performance of the NSGA-II. Some numerical examples were generated to assess the performance of the two algorithms on the IRAP. The results showed that the NSGA-II outperformed MOPSO and MOHS in terms of both the number of Pareto fronts and the CPU.

As a recommendation for future research, the model can be formulated for the case that shortages are allowed. Also, the model can be considered for a supply chain network in both deterministic and uncertain environments. Future research can also consider the supply chain network of the components of airplane engines, where shortages can occur due to condtions of uncertainty.

Acknowledgement

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

References

[1] Cao D, Murat A, Chinnam RB. Efficient exact optimization of multi-objective redundancy allocation problems in series-parallel systems. Reliab Eng Syst Saf 2013;111:154–63.
[2] Dolatshahi-Zand A, Khalili-Damghani K. Design of SCADA water resource management control center by a bi-objective redundancy allocation problem and particle swarm optimization. Reliab Eng Syst Saf 2015;133:11–21.
[3] Garg H, Rani M, Sharma S, Vishwakarma Y. Bi-objective optimization of the reliability-redundancy allocation problem for series-parallel system. J Manuf Syst 2014;33(3):335–47.
[4] Gharakhani M, Taghipour T, Farahani K. A robust multi-objective production planning. Int J Ind Eng Comput 2010;1(1):73–8.
[5] Gholamian N, Mahdavi I, Tavakkoli-Moghaddam R, Mahdavi-Amiri N. Comprehensive fuzzy multi-objective multi-product multi-site aggregate production planning decisions in a supply chain under uncertainty. Appl Soft Comput 2015;37:585–607.
[6] Ghoniem A, Maddah B. Integrated retail decisions with multiple selling periods and customer segments: optimization and insights. Omega 2015;55:38–52.
[7] Ghorabaee MK, Amiri M, Azimi P. Genetic algorithm for solving bi-objective redundancy allocation problem with k-out-of-n subsystems. Appl Math Model 2015.
[8] Ghorabaee MK, Amiri M, Azimi P. Genetic algorithm for solving bi-objective redundancy allocation problem with k-out-of-n subsystems. Appl Math Model 2015;39(20):6396–409.
[9] Hsieh T-J, Yeh W-C. Penalty guided bees search for redundancy allocation problems with a mix of components in series–parallel systems. Comput Oper Res 2012;39(11):2688–704.
[10] Khalili-Damghani K, Amiri M. Solving binary-state multi-objective reliability redundancy allocation series-parallel problem using efficient epsilon-constraint, multi-start partial bound enumeration algorithm, and DEA. Reliab Eng Syst Saf 2012;103:35–44.
[11] Mousavi SM, Alikar N, Niaki STA. An improved fruit fly optimization algorithm to solve the homogeneous fuzzy series–parallel redundancy allocation problem under discount strategies. Soft Comput 2015:1–27.
[12] Mousavi SM, Alikar N, Niaki STA, Bahreininejad A. Two tuned multi-objective meta-heuristic algorithms for solving a fuzzy multi-state redundancy allocation problem under discount strategies. Appl Math Model 2015.
[13] Mousavi SM, Hajipour V, Niaki STA, Aalikar N. A multi-product multi-period inventory control problem under inflation and discount: a parameter-tuned particle swarm optimization algorithm. Int J Adv Manuf Technol 2013:1–18.
[14] Mousavi SM, Hajipour V, Niaki STA, Alikar N. Optimizing multi-item multi-period inventory control system with discounted cash flow and inflation: two calibrated meta-heuristic algorithms. Appl Math Model 2013;37(4):2241–56.
[15] Mousavi SM, Sadeghi J, Niaki STA, Alikar N, Bahreininejad A, Metselaar HSC. Two parameter-tuned meta-heuristics for a discounted inventory control problem in a fuzzy environment. Inf Sci 2014.
[16] Naka S, Genji T, Yura T, Fukuyama Y. Practical distribution state estimation using hybrid particle swarm optimization. Paper read at Power Engineering Society Winter Meeting, 2001. IEEE; 2001.
[17] Pasandideh SHR, Niaki STA, Mousavi SM. Two metaheuristics to solve a multi-item multiperiod inventory control problem under storage constraint and discounts. Int J Adv Manuf Technol 2013:1–14.
[18] Rahmati SHA, Hajipour V, Niaki STA. A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem. Appl Soft Comput 2013;13(4):1728–40.
[19] Sadeghi J, Sadeghi S, Niaki STA. A hybrid vendor managed inventory and redundancy allocation optimization problem in supply chain management: an NSGA-II with tuned parameters. Comput Oper Res 2014;41:53–64.
[20] Shi Y, Eberhart RC. Empirical study of particle swarm optimization. Paper read at Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on; 1999.
[21] Soltani R, Safari J, Sadjadi SJ. Robust counterpart optimization for the redundancy allocation problem in series-parallel systems with component mixing under uncertainty. Appl Math Comput 2015;271:80–8.
[22] Wang W-L, Xu G-Q. Stability analysis of a complex standby system with constant waiting and different repairman criteria incorporating environmental failure. Appl Math Model 2009;33(2):724–43.
[23] Xie W, Liao H, Jin T. Maximizing system availability through joint decision on component redundancy and spares inventory. Eur J Oper Res 2014;237(1):164–76.
[24] Yeh W-C. Orthogonal simplified swarm optimization for the series–parallel redundancy allocation problem with a mix of components. Knowl-Based Syst 2014;64:1–12.
[25] Zhang E, Chen Q. Multi-objective reliability redundancy allocation in an interval environment using particle swarm optimization. Reliab Eng Syst Saf 2016;145:83–92.