Solving binary-state multi-objective reliability redundancy allocation series-parallel problem using efficient epsilon-constraint, multi-start partial bound enumeration algorithm, and DEA

Kaveh Khalili-Damghani a,*, Maghsoud Amiri b

a Department of Industrial Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran
b Industrial Management Department, Management and Accounting Faculty, Allameh Tabataba’i University, Tehran, Iran

ARTICLE INFO

Article history:
Received 1 April 2011
Received in revised form 17 February 2012
Accepted 8 March 2012
Available online 18 March 2012

Keywords:
e-constraint method
Pareto front
Multi-objective redundancy allocation problem
DEA

ABSTRACT

In this paper, a procedure based on efficient epsilon-constraint method and data envelopment analysis (DEA) is proposed for solving binary-state multi-objective reliability redundancy allocation series-parallel problem (MORAP). In first module, a set of qualified non-dominated solutions on Pareto front of binary-state MORAP is generated using an efficient epsilon-constraint method. In order to test the quality of generated non-dominated solutions in this module, a multi-start partial bound enumeration algorithm is also proposed for MORAP. The performance of both procedures is compared using different metrics on well-known benchmark instance. The statistical analysis represents that not only the proposed efficient epsilon-constraint method outperform the multi-start partial bound enumeration algorithm but also it improves the founded upper bound of benchmark instance. Then, in second module, a DEA model is supplied to prune the generated non-dominated solutions of efficient epsilon-constraint method. This helps reduction of non-dominated solutions in a systematic manner and eases the decision making process for practical implementations.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

The utilization of redundancy is assumed to be one of the main attributes to meet high level reliability. The problem is then to select the feasible design configuration that maximizes some measurement functions (i.e., reliability, cost, weights, and risk). This is called reliability redundancy allocation problem (RAP) which was first introduced by [18]. A series-parallel system is basically characterized through a predefined number of subsystems which are connected serially. Multiple component choices and redundancy levels are available to connect in parallel for each sub-system. A given component may have a binary-state or a multi-state in RAP [14,27]. In binary-state RAP, the problem of a proper structure can be handled through increasing the reliability of components or supplying parallel redundant components at some stages. In some other cases the states of a given component may follow more than two different levels, ranging from perfectly working to completely failed. A well-known performance measure for multi-state systems is system utility [1].

In the case of a multi-state system, the status of the system is represented through state distribution. The system utility of a multi-state series-parallel system can also be improved through providing redundancy at each stage, or improving the component state distribution.

Single objective RAP is assumed to be an NP-hard problem [5]. The multi-objective RAP can properly be matched with real world system engineering optimization problems. Different heuristic and meta-heuristic methods such as evolutionary computation methods, variable neighborhood search, ant colony optimization, particle swarm optimization, genetic algorithms, and Tabu search algorithm have been proposed in this area (Salazar et al. [21]; [15,6,26,13,20,22,12] represented an overview and tutorial describing genetic algorithms (GA) developed specifically for problems with multiple objectives. Different application of meta-heuristic methods in RAP can be found in literature ([24,16,28,29] (a) and (b)). [8] surveyed GA-based approach for various reliability optimization problems. [16] proposed a two-stage approach for solving multi-objective system reliability optimization problems. In first stage, a multi-objective evolutionary algorithm (MOEA) generated non-dominated solutions. Then, a self-organizing map (SOM) was supplied to cluster similar solutions and finally a DEA represented to select the most efficient solutions in each cluster.

1.1. Motivation and novelty of our proposed procedure

As mentioned, the complexity of reliability design problems, especially multi-objective versions, made researchers to propose...
several heuristic and meta-heuristic procedures. If decision maker (DM) would like to achieve non-dominated solutions while there are no prior articulations of preference on multiple objectives, meta-heuristic methods are also suggested. But these are not enough to ignore efficient mathematical and tailor-made heuristic models. Although, the usefulness of meta-heuristic methods has been proved in different areas but, the existence of commercial OR softwares (i.e., CPLEX, GAMS or LINGO) which are able to model and solve complicated real life problems, affect the rationalization usage of time consuming procedure of designing customized meta-heuristic methods. More clearly, the application and development of meta-heuristic procedures, which mainly require large amount of customization and parameter tuning as well as soft-ware code development is rational if no mathematical procedure be able to properly solve an NP-Hard problem using bounded resources (i.e., time, hard-ware, and soft-ware). Although different source codes are available for meta-heuristic methods but they usually need a time consuming customization phase which results in re-coding the main parts of the original code. As mentioned, the several parameters tuning of meta-heuristic methods is another issue. Some improved mathematical procedure can be assumed as a proper method in this area.

In this paper an efficient epsilon-constraint method has been proposed to generate set of non-dominated solutions on Pareto front of an MORAP problem. In order to evaluate the performance of efficient epsilon-constraint method it has been compared with a new developed multi-start partial bound enumeration algorithm. Statistical analysis has been supplied by using different metrics of multi-objective optimization. Then, a DEA model is supplied to prune non-dominated solutions in favor of more practical implementation issues for DM. On the other hand, the DEA model selects the designs on the DEA efficient frontier which have less cost and weight, and higher reliability. The aforementioned methods form a hybrid procedure. Then, a binary-state multi-objective reliability redundancy allocation series-parallel problem has been introduced and attacked through proposed hybrid procedure. The proposed procedures have been applied on benchmark instance of MORAP. Methods have been coded using LINGO 12.0 and MS-Excel, and Visual Basic 6.0. The rest of the paper is arranged as follows. The MODM concepts, including some preliminary definitions, epsilon-constraint method, and partial bound enumeration algorithm have been revisited in Section 2. Section 3 has been allocated to develop the efficient epsilon-constraint method and multi-start partial bound enumeration algorithm for binary-state MORAP. The experimental results including comparison metrics and statistical analysis are supplied in Section 4. In Section 5, the paper will be ended with conclusion remarks.

2. Multi-objective decision making (MODM) concepts

MODM problems are laid among common paradigm of the world of engineering optimization. In real world situations, a set of objectives belonging to a wide variety of problem’s attributes should be considered, simultaneously. Since an optimal solution is hard to find for an MODM problem so, a proper alternative is to look for a set of non-dominated solutions. So, DM can pick a certain solution or a set of preferred solutions among them. There are several MODM procedures in literature [9].

Formally, an MODM model considers a vector of decision variables, objective functions, and constraints. DMs attempt to optimize the objective functions. Since this problem has rarely a unique solution, DMs are expected to choose a solution from among the set of efficient solutions. Generally, the MODM problem with minimum objective functions can be formulated as (1).

\[
\text{(MODM)} \{ \begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad x \in S = \{ x \in \mathbb{R}^n | g(x) \leq b, x \geq 0 \}
\end{align*} \]

Where, \(f(x)\) represents \(k\) conflicting objective functions, \(g(x) \leq b\) represents \(m\) constraints, \(S\) is feasible solution space, and \(x\) is an \(n\)-vector of decision variables, \(x \in \mathbb{R}^n\).

During the process of decision making, some preference information articulation from DM may be required, and what type of information and when it is given play a critical role in the actual decision-making method. Under this consideration, the methods for solving MODM problems have been systematically classified into four classes by [9]. In one of the aforementioned classes, when there is a posterior articulation of preference information on priority of objective functions generating non-dominated solutions on Pareto front of MODM problem is desirable. The methods in this class strictly deal with constraints and do not consider the preference of DMs. The desired outcome, however, is to narrow the possible courses of actions and select the preferred course of action easier. They are also called non-dominated solutions generation methods. The \(\varepsilon\)-constraints method is a one of such techniques to MODM proposed by [2].

2.1. Epsilon-constraint method

For special kind of MODM problems, including linear programing and those problems which have a non-decreasing objective function of decision variables, there are also methods that produce the entire efficient set. These methods can provide a representative subset of the Pareto set which in most cases is adequate. In this method, DM chooses one objective out of \(n\) to be optimized; the remaining objectives are constrained to be less than or equal to given target values. In mathematical terms, DM let \(f_j(x), j \in \{1, \ldots, k\}\) be the objective function chosen to be optimized, we have the following problem \(\text{P}(\varepsilon_j), j \in \{1, \ldots, k\}\):

\[
\min \{f_j(x), j \in \{1, \ldots, k\}; f_i(x) \leq \varepsilon_i, \forall i \in \{1, \ldots, k\}, i \neq j; x \in S\}. \tag{2}
\]

Where, \(S\) is feasible solution space.

One advantage of the \(\varepsilon\)-constraints method is that it is able to achieve efficient points in a non-convex Pareto curve. Therefore, as proposed in [23] the decision maker can vary the upper bounds \(\varepsilon_i\) to obtain weak Pareto optima. Clearly, this is also a drawback of this method, i.e., the decision maker has to choose appropriate upper bounds for the \(\varepsilon_i\) values. Moreover, the method is not particularly efficient if the number of the objective functions increase. Several research works are dedicated to improvements of \(\varepsilon\)-constraint method. [17] recently presented a research work as an effort to effectively implement the \(\varepsilon\)-constraint method for producing the Pareto optimal solutions in a MODM problem. [17] has been proposed a novel version of the \(\varepsilon\)-constraint method (i.e., augmented \(\varepsilon\)-constraint method – AUGMECON) that avoids the production of weakly Pareto optimal solutions and accelerates the whole process by avoiding redundant iterations. Following, the AUGMECON method has briefly been revisited.

2.2. Augmented \(\varepsilon\)-constraint method

It is obvious that the optimal solution of (2) is guaranteed to be an efficient solution only if the values of slack or surplus variables of the entire associated \(k-1\) objective functions’ constraints are equal to zero. Under any other conditions, the solution is not assumed to be efficient. Considering the aforementioned issue, the following slack based model (3) has been proposed.

\[
\min f_1(x) + \beta \times (s_1 + \ldots + s_{k-1} + s_{k+1} + \ldots + s_k)
\]
s.t. \( f_j(x) + s_i = e_i, \quad \forall i \in \{1, \ldots, k\}, i \neq j. \)
\( x \in S. \)
\( s_i \in \mathbb{R}^+ \quad \forall i \in \{1, \ldots, k\}, i \neq j. \) \( (3) \)

Where, \( \beta \) is chosen properly a small number usually between 0.001 and 0.000001. The above formulation of the \( \varepsilon \)-constraint method produces only efficient solutions. Some consideration about commensurability may be desirable in objective function. So the objective function will be \( f_j(x) + \beta \times (s_1/r_1 + \ldots + s_j/r_j + \ldots + s_k/r_k). \) Where, \( r_i, i = 1, \ldots, k, \) \( i \neq j \) represents the range of objective \( i \) which has been calculated from payoff table of associated single objective optimization problem of original MODM problem. It is clear that the augmented \( \varepsilon \)-constraint (AUGMECON) method generates efficient solutions.

2.3. Enumeration algorithm for integer optimization

Different direct enumeration algorithms, including PBEA, have been proposed to solve single objective integer optimization problems. [19] proposed an efficient enumeration algorithm to solve single objective integer programming optimization. [11] proposed partial bounded enumeration algorithm for single objective reliability design problems. Both algorithms follow fixed sequential general parts using different steps. They are actually looking for a solution point which lied in the feasibility region of the problem defined by the constraints and has an optimal objective function value. Using this fact that the optimum is generally expected to be close to the boundaries defined by the constraints, it is needed to generate a sequence of search points such that points of the feasible region are tested in a systematic manner. It is occurred due to this fact that for a single objective optimization problem, i.e., reliability maximization, with a non-decreasing objective function of decision variables, quality of bound solution is actually better than within the feasibility region solutions. Although their algorithms served no offer for multi-objective cases, but the concept persuade us to extend an efficient enumeration algorithm customized for multi-objective integer optimization problem.

3. Proposed methods for MOPRAP

In binary-state MOPRAP, a set of objective functions (i.e., reliability, cost, weight, and volume) are to be optimized considering a set of constraints (i.e., cost, weight, and volume).

Basic Assumptions for binary-state MORAP:

- All components are assumed to be non-repairable.
- All components are assumed to have two states (i.e., working/fail)
- The functioning and physical properties (i.e., reliability, volume, weight, and cost) of all components are assumed to be known, deterministic, and time-independent.
- The cost of each component is assumed to be time-independent

Following notations are assumed to be used in binary-state MORAP.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m )</td>
<td>Number of sub-systems</td>
</tr>
<tr>
<td>( i )</td>
<td>Index of sub-systems, ( i = 1, 2, \ldots, m )</td>
</tr>
<tr>
<td>( j )</td>
<td>Index of components in each sub-system, ( j = 1, 2, \ldots, n )</td>
</tr>
<tr>
<td>( r_{ij} )</td>
<td>Reliability of component ( j ) in sub-system ( i )</td>
</tr>
<tr>
<td>( c_{ij} )</td>
<td>Cost of component ( j ) in sub-system ( i )</td>
</tr>
<tr>
<td>( w_{ij} )</td>
<td>Weight of component ( j ) in sub-system ( i )</td>
</tr>
</tbody>
</table>

The model (4)–(11) represent a binary-state MOPRAP in which (4)–(6) are allocated to describe the reliability, cost, and weight objective functions, respectively. The constraint (7) holds the maximum allowed cost of system while constraint (8) is written for weight of system. The set of constraints (9) and (10) guarantee the minimum and maximum allowed number of components in each sub-system. The set of constraints (11) guarantee that the decision variable is a member of positive integers. Fig. 1, represents the schematic view of a series-parallel system.

The binary-state MORAP:

\[ \text{Max } R_s = \prod_{i=1}^{m} (1 - \prod_{j=1}^{n} (1 - r_{ij})^{a_{ij}}) \] \( (4) \)

\[ \text{Min } C_s = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \] \( (5) \)

\[ \text{Min } W_s = \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} x_{ij} \] \( (6) \)

s.t. \[ \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \leq C_0 \] \( (7) \)

\[ \sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} x_{ij} \leq W_0 \] \( (8) \)

\[ \sum_{j=1}^{n} x_{ij} \leq n_{\max}, \quad i = 1, 2, \ldots, m \] \( (9) \)

\[ \sum_{j=1}^{n} x_{ij} \geq n_{\min}, \quad i = 1, 2, \ldots, m \] \( (10) \)

\[ x_{ij} \in \mathbb{Z}^+, \quad i = 1, 2, \ldots, m; \quad j = 1, 2, \ldots, n \] \( (11) \)

Fig. 1. Series-parallel system.
3.1. AUGMECON method for MORAP

In this sub-section, the resultant problem which is direct outcome of application of efficient epsilon-constraint (AUGMECON) method on binary-state MORAP is represented. Application of efficient epsilon-constraint method (i.e., model (3)) on model (4)–(11) results in model (12)–(15).

Max $R_k - \beta \times (S_2/r_2 + S_3/r_3)$

s.t. (12)

$C_i + S_2 = e_2, \quad e_2 \in [C_i^+, C_i^-]$ (13)

$W_1 + S_3 = e_3, \quad e_3 \in [W_1^+, W_1^-]$ (14)

$X \in S$ (15)

Where, $r_k, k = 1, 2$ represent the range of objective $i$ which has been calculated from payoff table of original binary-state MORAP (i.e., using the Ideal and Nadir value of $C_i^+, C_i^-, W_1^+, W_1^-$). $X \in S$ is feasible region of original binary-state MORAP (i.e., relations (7)–(11)), and $\beta$ is a little positive value (usually between 0.001 and 0.000001). In real world problem with $k$ objectives, like ours which has 3 conflicting objectives, the model (4)–(11) results in $k-1$ $\varepsilon$-constraints.

3.2. Proposed multi-start partial bound enumeration algorithm (PBEA) for MORAP

Consider a typical single objective integer optimization problem as follows.

Maximize $f(x) = f(x_1, x_2, \ldots, x_n)$

s.t. $g_j(x) \leq b_j, \quad j = 1, 2, \ldots, m$

$x_i \geq 1$ (integer), $i = 1, 2, \ldots, n$

The steps of proposed multi-start PBEA for MORAP can be summarized as follows:

**Step 1. Find lower and upper bound of variables.** Set $X_{lb}$ equal to 1 for all $i$ and all $k$. Then, simply find lower and upper bound of achievable values for each variable of each solution subject to all constraints. Call lower and upper bound of each variable of each solution $x_{lb}^i$ and $x_{ub}^i$, respectively.

**Step 2. Find feasible solution.** Considering $m$ different constraints of optimization problem, generate $k$ solutions like $X_k = \{X_{lb}^{1k}, X_{lb}^{2k}, \ldots, X_{lb}^{nk}\}$ in feasible region. Where, $k$ is a parameter of the algorithm.

**Step 3. Find bound solution.** Set $X_{lb}^i$ (i-th component of $k$-th feasible solution) to $X_{lb}^i + 1$ for all $i$ until at least one constraint is violated; then subtract 1 from $X_{lb}^i$ for all $i$. Finally, Add 1 to $X_{lb}^i$ one by one until all the constraints are violated.

**Step 4. Systematic search of solution space.** Using the lower and upper bound of each variable change the structure of a bound solution in a way that the combination of objective functions are improved in the sense of generating a non-dominated solution. Stop as there is no probability to generate a new non-dominated solution considering feasibility conditions. Put all generated non-dominated solutions in the temporary archive of non-dominated solutions.

**Step 5. Choose the final non-dominated solutions.** Rank all available solutions in temporary archive of non-dominated solutions according to non-dominated sorting. If the final archive of non-dominated solutions is empty then, put all solutions which have taken rank 1 into final archive of non-dominated solutions. Otherwise, only add the extracted non-dominated solutions from union of existing solutions in final archive of non-dominated solutions and temporary archive of non-dominated solutions. Empty the temporary archive of non-dominated solutions. If the termination conditions are met, print the set of existing solutions in final archive of non-dominated solutions; otherwise go to step 2.

Using aforementioned logic, there is no need to check all solution space of an integer optimization problem. Only those feasible solutions which are on the bound of solution space are candidate for checking. Moreover, using a systematic manner, all bound solutions are not required to be checked. So the search procedure is amazingly reduced. The concept can also be used for multi-objective integer programming with several constraints.

Although the bound enumeration algorithm can achieve the real Pareto front of a MODM problem, but checking the feasibility of a solution against different set of constraints, selecting non-dominate solutions among the feasible solutions, putting them in a temporary archive of non-dominate solutions in each step of the algorithm and, re-generating the true Pareto front of an MODM problem may computational be expensive. The proposed PBEA has been customized for a typical MODM problem, called MORAP.

The multi-start property has been inserted in the proposed algorithm to utilize the capabilities of parallel computations in searching the feasible bound solutions of the MORAP. On the other hand, several local bound enumeration algorithms with different bound solutions are run in our proposed multi-start partial bound enumeration algorithm (MSPBEA) to efficiently search the solution space of the MORAP. After several iterations of the proposed MSPBEA, which is a parameter of our proposed algorithm, the archives of the local PBEAs are compared to delete the dominant solutions. This property will lead to faster convergence to Pareto front of MORAP.

Fig. 2 represents the schematic view of proposed multi-start PBEA for multi-objective redundancy allocation problem.

4. Experimental results

In order to test the proposed method, a well-known benchmark instance was intentionally selected from literature ([24,16,28,29] (a)).

4.1. Description of benchmark instance

The instance consists of a configuration of three subsystems, with an option of five, four, and five types of components in each subsystem, respectively. The problem involves determining optimum number of selected component types in each sub-system. The maximum number of components is eight in each sub-system. Table 1 defines the component choices for each subsystem.

4.2. Software–hardware implementation

The efficient epsilon-constraint method was coded using LINGO 12.0 and VBA for MS-Excel 12.0 while the multi-start PBEA was implemented in VB 6.0. Both methods were implemented on a PIV Pentium PC with MS-Windows XP Professional, 1 GB of RAM, and 2.0 GHz Core 2 CPU.

4.2.1. Multi-threads property

Multi-threads can exist within the same process and share memory. In particular, the threads of a process share the code and the values of variables at any given moment. On a single processor, multitreading generally occurs by time-division multiplexing as in multitasking. The processor switches between different threads. This context switching generally happens frequently enough that the
user perceives the threads or tasks as running at the same time. On a multiprocessor, including multi-core system, the threads or tasks will actually run at the same time, with each processor or core running a particular thread or task.

The multi-threads properties were supplied in VB codes in order to perform the parallel computations of multi-start PBEA using the dual-core processor.

4.3. Results of efficient epsilon-constraint and proposed multi-start PBE algorithm

Table 2 represents the pay-off table of different objectives. Considering the calculated values of Table 2 the range of each objective function can be easily found. Table 3 represents the configuration of both methods. Figs. 3 and 4 plots the

Fig. 2. Schematic view of proposed multi-start PBEA for multi-objective redundancy allocation problem.
non-dominated solutions of efficient epsilon-constraint method and multi-start PBE algorithm, respectively, during all runs. The Fig. 3 contains 117 non-dominated solutions of efficient epsilon-constraint method, while Fig. 4 represented 70 non-dominated solutions of multi-start PBE algorithm. It is notable that the best known upper bound of reliability objective for this benchmark instance (i.e., 0.999999) has been improved by the proposed procedure which generate solutions with the reliability objective equal to 0.999999999

### 4.4. Comparison metrics

Different metrics are considered to investigate the performance of efficient epsilon-constraint method and multi-start partial bound enumeration algorithm. The metrics have been intentionally selected [7, 25] to represent different aspects of both methods in regenerating the true Pareto front (PFtrue) or even best known Pareto front (PFbest) of MORAP.

#### 4.4.1. Quality measures

The following metrics are supplied to measure the quality of a procedure in regenerating PFtrue or PFbest.

- **Error ratio (ER).** The error ratio (ER) metric reports the number of generated non-dominated solutions that are not members of PFtrue or even PFbest. The definition of the error ratio is as follows.

\[
ER = \frac{\sum_{i=1}^{N} e_i}{N}
\]

Where \( N \) is the number of generated non-dominated solutions, and

\[
e_i = \begin{cases} 
0 & \text{if the solution } i \text{ belongs to } PF_{\text{true}} \text{ or even } PF_{\text{best}} \\
1 & \text{otherwise}
\end{cases}
\]

The closer this metric is to unit, the less the solution has converged toward PFtrue or PFbest.

- **Generational distance (GD).** This metric calculates the distance between the PFtrue or PFbest and the generated solution set. The definition of this metric as follows.

\[
GD = \frac{\sum_{i=1}^{N} d_i}{N}
\]

Where, \( d_i = \min_{p \in PF} \sqrt{\sum_{k=1}^{m} (z_{ik} - z_{pk})^2} \) is minimum Euclidean distance between solution \( i \) and PFtrue or PFbest in which \( m \) is number of objective functions.

#### 4.4.2. Diversity measures

The following measures are selected to compare both procedures in the sense of diversity of generated solutions on PFtrue or PFbest.

- **Spacing metric (SM).** The spacing metric measures the uniformity of the spread of the solution set. In order to calculate SM,
\( \bar{d} \) which is the mean value of all \( d_i \) should be calculated first as (18).

\[
\bar{d} = \frac{\sum_{i=1}^{N} d_i}{N}
\tag{18}
\]

Then, the \( SP \) which is the standard deviation of the closest distances is calculated as (19).

\[
SP = \left( \frac{\sum_{i=1}^{N} (\bar{d} - d_i)^2}{N-1} \right)^{0.5}
\tag{19}
\]

Diversification metric (DM). DM measures the spread of the solution set. DM is calculated as follows.

\[
DM = \left[ \sum_{i=1}^{N} \max\left( |x_i - y_j| \right) \right]^{0.5}
\tag{20}
\]

Where, \( |x_i - y_j| \) is the Euclidean distance between non-dominated solution \( x_i \) and the non-dominated solution \( y_j \).

The aforementioned metrics have been calculated for both procedures in 20 different runs on the top 50 non-dominated solutions of each run which had the largest reliability objective values. It is notable that as the PFtrue of the benchmark instance has not been known; the PFbest has been derived using all non-dominated solutions of both methods in all runs. So the metrics were calculated using the PFbest and the Pareto front which has been generated by each method during a single run. Results represented in Table 4. The CPU time of both algorithms are also represented in Table 3.

As the contents of Table 4 represents, the performance of efficient epsilon-constraint is better for ER, SM, DM, and CPU Time. The GD performance seems to be better in multi-start partial bound enumeration algorithm. These observations should be statistically tested.

4.5. Statistical testing of performance

First, the normality test has been accomplished in order to check whether the normal distribution is fitted on the distribution of aforementioned metrics over the solution space. The results of Kolmogorov–Smirnov test are represented in Fig. 5.

The number of observation is 20 which is a proper quantity for parametric statistical analysis. As the central limit theorem suggests, the Fig. 5 also represented, there is no enough evidence to reject the hypothesis of normality of performance metrics, so analysis of variance (ANOVA) can be proper to decide if there is a meaningful difference between performances of both methods. The one way ANOVA test is supplied to compare the performance of both procedures. The hypotheses are:

\( H_0: \) the population means are all equal
\( H_1: \) the population means are different.

The results are represented in Table 5. The contents of Table 5 represent the meaningful dominance of efficient epsilon-constraint method in comparison with the multi-start partial bound enumeration algorithm using proposed metrics.

As clear the time performance of efficient epsilon-constraint is statistically proved to be better than time performance of multi-start partial bound enumeration algorithm. So, the null hypothesis of equal CPU time for both methods is rejected under confidence level of %95. The ER values in efficient epsilon-constraint are statistically shown to be less than the same metric in multi-start partial bound enumeration algorithm. The null hypothesis is rejected for GD metric. The results represent better performance of multi-start partial bound enumeration algorithm in GD metric. The efficient epsilon-constraint reported less spacing metric which revealed the uniformity of generated non-dominated solutions in comparison with multi-start partial bound enumeration algorithm. The null hypothesis was also rejected on DM metric. This represented that the generated non-dominated solutions of efficient epsilon-constraint were more scattered on Pareto front of benchmark instance.

4.6. Pruning non-dominated solutions

As the generated designs of efficient epsilon-constraint method was illustrated to meaningfully better than the multi-start partial bound enumeration algorithm, so a pruning methodology was implemented on generated solutions of former method to ease the process of decision making about practical designs.

<table>
<thead>
<tr>
<th>Run</th>
<th>Efficient epsilon-constraint method</th>
<th>Multi-start partial bound enumeration algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ER</td>
<td>GD</td>
</tr>
<tr>
<td>1</td>
<td>0.16</td>
<td>3.08</td>
</tr>
<tr>
<td>2</td>
<td>0.16</td>
<td>3.69</td>
</tr>
<tr>
<td>3</td>
<td>0.16</td>
<td>2.8</td>
</tr>
<tr>
<td>4</td>
<td>0.16</td>
<td>1.87</td>
</tr>
<tr>
<td>5</td>
<td>0.16</td>
<td>2.59</td>
</tr>
<tr>
<td>6</td>
<td>0.16</td>
<td>4.87</td>
</tr>
<tr>
<td>7</td>
<td>0.16</td>
<td>2.59</td>
</tr>
<tr>
<td>8</td>
<td>0.16</td>
<td>2.22</td>
</tr>
<tr>
<td>9</td>
<td>0.16</td>
<td>2.22</td>
</tr>
<tr>
<td>10</td>
<td>0.16</td>
<td>4.11</td>
</tr>
<tr>
<td>11</td>
<td>0.16</td>
<td>4.38</td>
</tr>
<tr>
<td>12</td>
<td>0.16</td>
<td>3.12</td>
</tr>
<tr>
<td>13</td>
<td>0.16</td>
<td>6.38</td>
</tr>
<tr>
<td>14</td>
<td>0.16</td>
<td>2.61</td>
</tr>
<tr>
<td>15</td>
<td>0.16</td>
<td>2.84</td>
</tr>
<tr>
<td>16</td>
<td>0.16</td>
<td>3.8</td>
</tr>
<tr>
<td>17</td>
<td>0.16</td>
<td>3.45</td>
</tr>
<tr>
<td>18</td>
<td>0.16</td>
<td>3.23</td>
</tr>
<tr>
<td>19</td>
<td>0.16</td>
<td>1.64</td>
</tr>
<tr>
<td>20</td>
<td>0.16</td>
<td>2.58</td>
</tr>
<tr>
<td>Ave.</td>
<td>0.17</td>
<td>3.2</td>
</tr>
<tr>
<td>Std.</td>
<td>0.01</td>
<td>1.12</td>
</tr>
</tbody>
</table>
Selection from among the resultant non-dominated solutions is a hard job in real cases yet. This problem is independent of applied procedure and can be seen in literature [24,16] where the process of decision making about several design choices has been left off. More clearly, a set of non-dominated solutions, are usually represented to DM while she/he may have no clear sense to select from among...
them. Under these conditions, a systematic approach is required to make the choice easier. So, a data envelopment analysis (DEA) model was supplied to prune the generated non-dominated.

4.6.1. Data envelopment analysis to prune non-dominated solutions

In general, DEA is a technique based on linear programming in order to evaluate the efficiency of decision making units (DMUs) and determine efficient frontiers. [3] introduced the CCR model to make the choice easier. So, a data envelopment analysis (DEA) was applied to check whether a given DMU (i.e., a non-dominated solution) is efficient. Considering a set of \( n \) DMUs (DMU\(_i\), \( i = 1, \ldots, n \)) producing \( s \) outputs (\( y_{ij} \), \( r = 1, \ldots, s \)) by consuming \( m \) inputs (\( x_{ij}, \, i = 1, \ldots, m \)), the additive model is represented as (21).

\[
\max \sum_{i=1}^{m} S_{ip} + \sum_{r=1}^{s} S_{rp} \\
\text{s.t.} \quad \sum_{j=1}^{n} i_r x_{ij} + S_{ip} = x_{ip}, \, \quad i = 1, 2, \ldots, m \\
\quad \sum_{j=1}^{n} j_r y_{ij} - S_{rp} = y_{rp}, \quad r = 1, 2, \ldots, s \\
\quad \sum_{j=1}^{n} j_r = 1 \\
\quad i_r, S_{ip}, S_{rp} \geq 0, \quad j = 1, 2, \ldots, n, \quad i = 1, 2, \ldots, m, \quad r = 1, 2, \ldots, s 
\]

Where, \( S_{ip} \) and \( S_{rp} \) are input and output slacks and DMU\(_p\) (i.e., the \( p \)th non-dominated solution in a given set of non-dominated solution) is efficient under the additive model (21) if and only if the optimal value of its objective function is zero. On the other hand, the model (21) seeks the non-dominated solutions which consuming lower cost and weight, produce higher reliability from among the others.

The model (21) was run for 10 generated non-dominated solutions of model Table 3. Three designs as shown in Fig. 7 are determined as final design. The results are represented in Table 6. Generally, the number of proposed designs is reduced through model (21) so; process of decision making in which DM should prefer one of the non-dominated solutions for practical implementation is eased.

### Table 5

The results of ANOVA on comparison metrics.

<table>
<thead>
<tr>
<th>Source</th>
<th>Degree of freedom</th>
<th>Sum of square</th>
<th>Mean square</th>
<th>F</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. First metric: CPU time (s.)</td>
<td>Factor 1</td>
<td>63,202</td>
<td>63,202</td>
<td>404.56</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Error 1</td>
<td>5,937</td>
<td>156</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 39</td>
<td>69,139</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S=12.50 R-Sq=91.41% R-Sq(adj)=91.19%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>II. Second metric: ER</td>
<td>Factor 1</td>
<td>0.48</td>
<td>0.48</td>
<td>113.00</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Error 1</td>
<td>0.16</td>
<td>0.004</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 39</td>
<td>0.54</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S=0.00500 R-Sq=74.83% R-Sq(adj)=74.16%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>II. Third metric: GD</td>
<td>Factor 1</td>
<td>16.18</td>
<td>16.18</td>
<td>19.87</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Error 1</td>
<td>30.94</td>
<td>0.81</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 39</td>
<td>47.12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S=0.9023 R-Sq=34.34% R-Sq(adj)=32.61%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IV. Fourth metric: SM</td>
<td>Factor 1</td>
<td>17.62</td>
<td>17.62</td>
<td>6.52</td>
<td>0.015</td>
</tr>
<tr>
<td></td>
<td>Error 1</td>
<td>102.66</td>
<td>2.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 39</td>
<td>120.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S=1.644 R-Sq=14.65% R-Sq(adj)=12.40%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V. Fifth metric: DM</td>
<td>Factor 1</td>
<td>6,105.35</td>
<td>6,105.35</td>
<td>4,595.13</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Error 1</td>
<td>50.49</td>
<td>1.33</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 39</td>
<td>6,155.84</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S=1.153 R-Sq=99.18% R-Sq(adj)=99.16%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 6. A given non-dominated solution as a DMU.

Fig. 7. Proposed structures.

---

In this paper, a procedure based on an extended version of \( \epsilon \)-constraint method and DEA was proposed to solve binary-state multi-objective reliability redundancy allocation series-parallel problem. A binary-state multi-objective reliability redundancy allocation series-parallel problem, which is assumed for efficiency measurement. Each non-dominated solution of previous module is considered as a DMU with two inputs (i.e., cost, and weight) and one output (i.e., reliability). Fig. 6 represents the schematic view of a DMU.
as an NP-Hard and proper testimonial of complicated and real life engineering design problems, was introduced. A comparative study was accomplished to investigate the performance of efficient ε-constraint method on a well-known benchmark instance of binary-state MORAP. To this aim, a tailor made heuristic procedure, namely multi-start partial bound enumeration algorithm, was proposed. The performance of proposed heuristic algorithm, which was an improved version of an existing algorithm in literature, was compared with efficient ε-constraint method by using different comparison metrics for multi-objective optimization problems. The statistical analyses were accomplished to survey the performance of both methods on benchmark instance of MORAP. The efficient ε-constraint method represented outperformed the multi-start partial bound enumeration algorithm. The efficient ε-constraint method generated more qualified non-dominated solutions on Pareto front of an MORDM problem. Then, a DEA based model was supplied to prune the generated non-dominated solutions of ε-constraint method. This resulted in selection of DEA efficient designs which lie over Pareto front of the problem. The number of final designs systematically reduced and the DM had fewer choices to implement. The results were straightforward and promising.

Acknowledgment

The authors would like to thank the referees which their comments improved the quality of the paper.

References


Table 6

Pruning the non-dominated solutions by additive model (15).

<table>
<thead>
<tr>
<th>Non-dominated Solution (DMU)</th>
<th>Input</th>
<th>Output</th>
<th>S_p</th>
<th>S_p</th>
<th>S_p</th>
<th>Objective value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMU1</td>
<td>81</td>
<td>91</td>
<td>0.9999458 0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>DMU2</td>
<td>89</td>
<td>89</td>
<td>0.9999688 1.00</td>
<td>55.00</td>
<td>0.06</td>
<td>56.06</td>
</tr>
<tr>
<td>DMU3</td>
<td>87</td>
<td>89</td>
<td>0.9999515 0.00</td>
<td>59.17</td>
<td>0.05</td>
<td>59.22</td>
</tr>
<tr>
<td>DMU4</td>
<td>85</td>
<td>90</td>
<td>0.9999409 0.00</td>
<td>58.17</td>
<td>0.05</td>
<td>58.22</td>
</tr>
<tr>
<td>DMU5</td>
<td>84</td>
<td>90</td>
<td>0.9999461 0.00</td>
<td>70.50</td>
<td>0.03</td>
<td>70.53</td>
</tr>
<tr>
<td>DMU6</td>
<td>99</td>
<td>74</td>
<td>0.9998862 0.00</td>
<td>31.17</td>
<td>0.05</td>
<td>31.22</td>
</tr>
<tr>
<td>DMU7</td>
<td>81</td>
<td>80</td>
<td>0.9999206 0.00</td>
<td>38.83</td>
<td>0.01</td>
<td>38.84</td>
</tr>
<tr>
<td>DMU8</td>
<td>98</td>
<td>53</td>
<td>0.9999218 0.00</td>
<td>19.00</td>
<td>0.06</td>
<td>19.06</td>
</tr>
<tr>
<td>DMU9</td>
<td>72</td>
<td>38</td>
<td>0.9984723 0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>DMU10</td>
<td>36</td>
<td>19</td>
<td>0.9372272 0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>