Reliability Engineering and System Safety 111 (2013) 154–163

Contents lists available at SciVerse ScienceDirect

Reliability Engineering and System Safety

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

# Efficient exact optimization of multi-objective redundancy allocation problems in series-parallel systems

Dingzhou Cao, Alper Murat, Ratna Babu Chinnam *

*Industrial & Systems Engineering Department, Wayne State University, Detroit, MI 48202, USA*

ARTICLE INFO

*Article history:*
Received 9 January 2012
Received in revised form
19 September 2012
Accepted 25 September 2012
Available online 7 November 2012

*Keywords:*
Redundancy allocation problem
Multi-objective optimization
Decomposition
Series-parallel systems

ABSTRACT

This paper proposes a decomposition-based approach to exactly solve the multi-objective Redundancy Allocation Problem for series-parallel systems. Redundancy allocation problem is a form of reliability optimization and has been the subject of many prior studies. The majority of these earlier studies treat redundancy allocation problem as a single objective problem maximizing the system reliability or minimizing the cost given certain constraints. The few studies that treated redundancy allocation problem as a multi-objective optimization problem relied on meta-heuristic solution approaches. However, meta-heuristic approaches have significant limitations: they do not guarantee that Pareto points are optimal and, more importantly, they may not identify all the Pareto-optimal points. In this paper, we treat redundancy allocation problem as a multi-objective problem, as is typical in practice. We decompose the original problem into several multi-objective sub-problems, efficiently and exactly solve sub-problems, and then systematically combine the solutions. The decomposition-based approach can efficiently generate all the Pareto-optimal solutions for redundancy allocation problems. Experimental results demonstrate the effectiveness and efficiency of the proposed method over meta-heuristic methods on a numerical example taken from the literature.



## 1. Introduction

The redundancy allocation problem (RAP) is a well-known problem in the "design-for-reliability" literature. It has a broad application in the real-world, such as electrical power systems design [1], transportation systems design [2], and telecommunications design [3]. The reliability of a system can be increased by allocating redundancies to its subsystems, but this can also increase the design cost and may negatively affect other considerations (such as system weight and volume). The objective of RAP is to determine optimal system designs that maximize system reliability and other considerations given certain constraints on the system.

In the past several decades, there have been a number of studies and approaches to the RAP. Roughly, they can be grouped into three methods: (1) single objective optimization with constraints, (2) aggregated objective function for multi-objective optimization, and (3) Pareto-based ranking for multi-objective optimization. The first set of methods treat the RAP as a single objective optimization problem (maximizing system reliability or minimizing cost) with constraints. Various single-objective optimization approaches have been used to solve such formulations, including dynamic programming [4–6], integer programming [7–9], mixed integer and nonlinear programming [16], column generation method [17], and meta-heuristics [10–15]. These single-objective optimization techniques have their own advantages. However, in practical applications, multiple considerations must be taken into account when determining the redundancy allocation of the system (e.g., when it is important to have high system reliability and low design cost). The aggregated objective function method can be implemented to solve this problem by summing the multiple objective functions into a single objective function. Then the new objective function can be solved using a single-objective optimization approach. Studies in [18–22] use this method. For example, Dhingra [18] presented a multi-objective reliability apportionment problem. However, [18] solved a multi-objective, nonlinear, mixed-integer mathematical programming problem by sequential unconstrained minimization techniques in conjunction with heuristic algorithms. The parallel-series system considered in this study included time-dependent reliability. The study in [19] provided an efficient computational method to obtain the optimal system structure of electronic devices by using a single or a multi-objective simulated annealing algorithm based optimization approach. Studies [20–22] used multi-criteria formulations with genetic algorithm (GA). The approach in [21] was based on GA and Monte Carlo simulation; while in [22] GA and physical programming were combined to solve the RAP.

* Corresponding author. Tel.: +1 313 577 4846; fax: +1 313 578 5902.
*E-mail addresses:* dcao@wayne.edu (D. Cao), amurat@wayne.edu (A. Murat), ratna.chinnam@wayne.edu (R.B. Chinnam).

0951-8320/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ress.2012.09.013

The above studies involving multiple objectives made important contributions toward finding more effective and efficient approaches for RAP. However, aggregating multiple objectives into a single objective to obtain promising results is a challenge as the scaling process necessary to normalize the objective space might affect the set of Pareto solutions found. In addition, the aggregation of multiple objectives may eliminate the possibility of identifying some non-dominated solutions [32]. To cope with these drawbacks, other multi-objective optimization approaches have been proposed. Multi-objective optimization refers to the process of solving problems with two or more objectives to be simultaneously optimized. Unlike the single-objective optimization problem, multi-objective optimization problems usually have a set of solutions called Pareto-optimal (i.e., non-dominated) solutions (e.g., [23–26]). In [23], the authors formulated the RAP as a tri-objective problem (i.e., maximize reliability, minimize cost and weight) and solved this problem using the Non-dominated Sorting Genetic Algorithm (NSGA2) originally proposed by Deb et al. [27]. In [24], the same authors from [23] presented an improved version of NSGA2 called MOMS-GA to solve the tri-objective redundancy allocation problem in multi-state systems. Authors of [25] employed a Tabu search and Monte-Carlo simulation method to solve the bi-objective (reliability and cost) redundancy allocation problem. In [26], authors employed a problem-specific evolutionary algorithm to solve the continuous reliability optimization problems where the decision variables are the reliabilities of the components.

The meta-heuristic based optimization approaches mentioned above are very popular for solving multi-objective RAP. However, meta-heuristic based approaches have several limitations: they do not guarantee that Pareto points are optimal; they may not identify all the Pareto-optimal points, and they may become computationally cumbersome, e.g., large population sizes in GA. In this paper, we regard RAP as a multi-objective problem and propose a decomposition driven approach to address these limitations.

A key step of the proposed approach is the exact solution of multi-objective sub-problems to generate the whole set of non-dominated solutions. The $\epsilon$-constraint method [28] is a classical method to generate whole set of non-dominated solutions, but it is generally computationally impractical for large problems. To improve the computational efficiency of the $\epsilon$-constraint method, an adaptive $\epsilon$-constraint method [29] was proposed for multi-objective combinatorial optimization problems and requires integral objective function values. While this method is more efficient than the traditional $\epsilon$-constraint method, it identifies many duplicate solutions, affecting the efficiency of the algorithm. To solve the multi-objective sub-problems exactly, we modify the adaptive $\epsilon$-constraint method to account for continuous objective values (due to reliability objective) and greatly reduce the need to solve problems with duplicate solutions.

The rest of this paper is organized as follows: Section 2.1 presents the multi-objective redundancy allocation problem for series-parallel systems. The proposed decomposition driven approach for multi-objective RAP is described in Section 2.2. The modified adaptive $\epsilon$-constraint method is presented in Section 2.3. The proposed method is applied to a numerical example in Section 3. Finally, the paper concludes with summary and directions for future research.

## 2. Decomposition-based solution framework for multi-objective RAP in series-parallel systems

We first briefly introduce the multi-objective RAP for series-parallel systems. Next we describe the decomposition-based approach and its properties. Lastly, we present an efficient method to generate the whole set of non-dominated solutions.

### 2.1. Redundancy allocation problem for series-parallel systems

A series-parallel system has a total of $s$ independent subsystems arranged in series; for the $i$th subsystem, it can have up to $n_{max,i}$ functionally equivalent components arranged in parallel. Each component potentially varies in reliability, cost, weight and other characteristics. A subsystem can work properly if at least one of its components is operational. The $n_i$ components are selected from $m_i$ available component types where multiple copies of each type can be selected. The typical structure of a series-parallel system is illustrated in Fig. 1. Increasing the number of redundant components will increase the system reliability, but that also increases its cost and weight. The goal is to optimally allocate the redundant components while balancing multiple competing objectives.

Without loss of generality, we restrict our attention in the rest of this manuscript to reliability, cost, and weight considerations in RAP. It is straightforward to incorporate additional characteristics as long as they are linear in decision variables (i.e., the number of components of a certain type used in each subsystem). We note here that reliability is not linear in decision variables; however, we overcome this difficulty through a transformation discussed in Section 2.2.

We formulate the RAP in a multi-objective setting with reliability, cost and weight considerations as:

($RAP$)

$$max f_{\mathrm{R}}(\mathbf{x}),\ min f_{\mathrm{C}}(\mathbf{x}),\ min f_{\mathrm{W}}(\mathbf{x}) \tag{1}$$

$s.t$

$$1 \le \sum_{j=1}^{m_i} x_{ij} \le n_{max,i} \quad \forall i \in S, \tag{2}$$

$$\mathbf{x} = \{x_{ij} \mid \forall i \in S, j = 1,\ldots,m_i\} \text{ and } x_{ij} \in \{0,1,2,\ldots,n_{max,i}\}, \tag{3}$$

where,

$$f_{\mathrm{R}}(\mathbf{x}) = \left[\prod_{i=1}^{s}\left(1-\prod_{j=1}^{m_i}(1-r_{ij})^{x_{ij}}\right)\right], \tag{4}$$

$$f_{\mathrm{C}}(\mathbf{x}) = \left[\sum_{i=1}^{s}\sum_{j=1}^{m_i} c_{ij}x_{ij}\right], \text{ and } f_{\mathrm{W}}(\mathbf{x}) = \left[\sum_{i=1}^{s}\sum_{j=1}^{m_i} w_{ij}x_{ij}\right]. \tag{5}$$

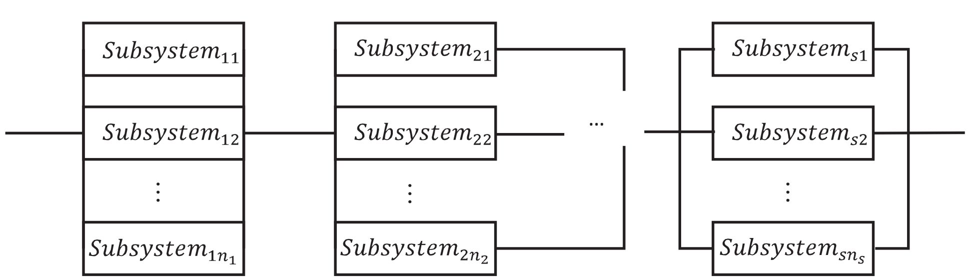


**Fig. 1.** General series-parallel redundancy system.

The set $S$ denotes the set of subsystems and $s=|S|$ is the number of subsystems; $x_{ij}$ is the decision variable denoting the number of $j$th type components used in subsystem $i$; $m_i$ denotes the number of available component types for subsystem $i$; $n_{max,i}$ denotes the maximum number of components in parallel used in subsystem $i$. The parameters $r_{ij}$, $c_{ij}$, and $w_{ij}$ denote the reliability, cost, and weight of the $j$th available component type for subsystem $i$, respectively. Eq. (5) implicitly assumes that the overall system weight and cost criteria are linear and additive which holds in most practical applications.

The solutions to the multi-objective RAP is a set of Pareto-optimal (non-dominated) solutions. Pareto-optimal solutions are those for which improvement in one objective can only occur with the worsening of at least one other objective. Thus, instead of a unique solution to the problem, the solution of a multi-objective optimization problem is a set of Pareto-optimal solutions. These solutions are characterized in terms of non-dominance relationship in the objective space and efficiency in the decision space [32]. In a minimization problem, the non-dominance relationship can be defined as follows solution $\mathbf{x}_1$ is more efficient than solution $\mathbf{x}_2$ if and only if: (a) $\mathbf{x}_1$ is no worse than $\mathbf{x}_2$ in all objectives, i.e., $f_k(\mathbf{x}_1) \le f_k(\mathbf{x}_2) \forall\, k \in \{1,2,\ldots,n\}$; and (b) $\mathbf{x}_1$ is strictly better than $\mathbf{x}_2$ in at least one objective, i.e., $f_k(\mathbf{x}_1) < f_k(\mathbf{x}_2)$ for atleast one $k$. Accordingly, if $\mathbf{x}_1$ is more efficient than solution $\mathbf{x}_2$, the objective vector corresponding to $\mathbf{x}_1$ dominates that of $\mathbf{x}_2$.

### 2.2. Decomposition approach

In solving multi-objective RAP problems, decomposing the original problem into sub-problems and systematically combining the solutions can greatly improve the efficiency of constructing the Pareto-optimal solution set for the original problem. The proposed decomposition-based solution framework is illustrated in Fig. 2. We first summarize the three phases of this approach below and then provide details in the remainder.

**Phase 1.** —*Linearization and decomposition*: We decompose the multi-objective RAP formulation in Section 2.1 into smaller multi-criteria sub-problems corresponding to each subsystem. This is achieved through by first linearizing the reliability objective and then decomposing the resulting formulation into individual sub-system multi-objective RAPs. The equivalence of the linearized reliability objective is established in Proposition 1.

**Phase 2.** —*Pareto set generation for each subsystem*: Next, we solve each subsystem's multi-objective RAP using an exact efficient Pareto set generation method presented in Section 2.3 so as to identify all the non-dominated solutions for each sub-problem. Proposition 2 establishes that any non-dominated solution for the original multi-objective RAP (e.g., integrated system) can be obtained as a combination of the non-dominated solutions for sub-system multi-objective RAPs.

**Phase 3.** —*Sequential combination and filtering*: As shown in Proposition 3, combining non-dominated subsystem solutions could result in dominated solutions for the multi-objective RAP of the integrated system. Hence, we sequentially combine the non-dominated solution sets of consecutive sub-problem pairs while filtering out the dominated solutions. This sequential combining and filtering continues until a single non-dominated solution set remains. Proposition 4 establishes that sequential combination and filtering does not eliminate any non-dominated solution for the original multi-objective RAP.

The decomposition benefits computational efficiency in two ways. First, the solution of decomposed problems is much easier than

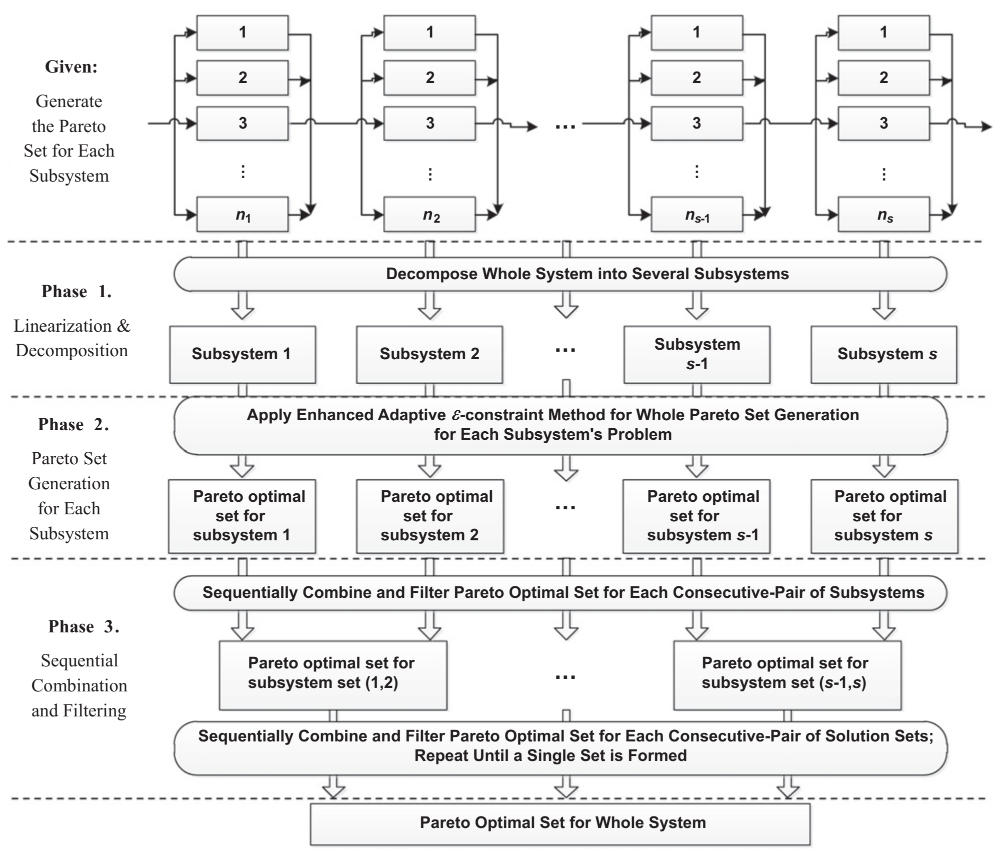


**Fig. 2.** Decomposition based solution framework of multi-objective RAP problems.

solving the integrated system problem. Secondly, the decomposition greatly facilitates the generation of the whole Pareto-optimal solution set by removing those dominated subsystem solutions early on during the sequential combining and filtering process. In what follows, we describe the details of the decomposition based approach. First, we describe a linearization approach for reliability objective, which allows us to decompose the multi-objective RAP into series-parallel systems (Phase 1, Fig. 2).

The RAP formulation presented in Section 2.1 is decomposable by subsystems except for the reliability objective. To make the reliability objective decomposable, we apply a logarithmic transformation to the reliability objective function in (4). The maximization of the product of reliability of all subsystems then becomes the maximization of the summation of logarithm of all subsystems' reliabilities.

$$max\, f_{R1}(\mathbf{x}) = \left[\sum_{i=1}^{s} \log\left(1-\prod_{j=1}^{m_i}(1-r_{ij})^{x_{ij}}\right)\right]. \tag{6}$$

The following proposition establishes that solutions to $max\, f_R(\mathbf{x})$ and $max\, f_{R1}(\mathbf{x})$, subject to the affine constraints in (2), are equivalent.

**Proposition 1.** The solutions of $\max f_R(\mathbf{x})$ and $max f_{R1}(\mathbf{x})$ subject to constraint set (2) are identical.

*Proof:* The $f_R$ is monotone increasing in $x_{ij}$ when $r_{ij}>0$ for all $i$ and $j$. The $f_{R1}$ is obtained through the logarithmic transformation, which is a monotonic transformation and preserves the rank order of the solutions, e.g., if $f_R(\mathbf{x})>f_R(\mathbf{x}')$ then we have $f_{R1}(\mathbf{x})>f_{R1}(\mathbf{x}')$. Further, the constraint set (2) is affine, hence does not change the rank order of feasible solutions. Therefore, a solution $\mathbf{x}$ maximizing $f_R(\mathbf{x})$ subject to (2) also maximizes $f_{R1}(\boldsymbol{x})$ subject to (2).

Through this monotonic transformation, we can then decompose the reliability objective by subsystems and solve each subsystem's reliability maximization independently (Phase 2, Fig. 2).

$$\max f_{R2}(\mathbf{x_i}) = \left[\log\left(1-\prod_{j=1}^{m_i}(1-r_{ij})^{x_{ij}}\right)\right] \forall i \in S, \tag{7}$$

where $\mathbf{x_i} = \{x_{ij} \mid j=1,\ldots,m_i\}$ is a solution for subsystem $i$.

Further, due to the monotonicity of logarithmic function, the maximization of the objective in (7) can be transformed to the following minimization objective while preserving the rank order of solutions:

$$\min f_{R3}(\mathbf{x_i}) = \left[\log\left(\prod_{j=1}^{m_i}(1-r_{ij})^{x_{ij}}\right)\right] \forall i \in S, \tag{8}$$

$$= \left[\sum_{j=1}^{m_i} \log(1-r_{ij})x_{ij}\right] \forall i \in S. \tag{9}$$

We also note that the constraint set (2) and the weight and cost criteria in (5) are decomposable by subsystems, e.g., $f_{C1}(\mathbf{x_i}) = [\sum_{j=1}^{m_i} c_{ij}x_{ij}]$ and $f_{W1}(\mathbf{x_i}) = [\sum_{j=1}^{m_i} w_{ij}x_{ij}]$.

There are several properties of the non-dominated solutions for the subsystem RAPs obtained through the proposed decomposition approach (Phase 2, Fig. 2). First property is concerned with whether the entire set of non-dominated solutions of the RAP can be obtained by solving decomposed subsystem problems. The following proposition establishes that any solution in the RAP's Pareto set can be obtained by Cartesian combining the individual subsystem's Pareto sets. In other words, the decomposition approach does not leave out any non-dominated solution of the RAP.

**Proposition 2.** Any non-dominated solution to the RAP problem can be obtained through Cartesian combining the non-dominated solutions of subsystems.

*Proof:* Proof is by contradiction. Without loss of generality, let us consider the case with two objectives $k=\{o_1,o_2\}$ and a decomposable problem by subsystems. The integrated problem is $\{minf_{o_1}(\mathbf{x}),\ minf_{o_2}(\mathbf{x}) | \mathbf{x} \in X\}$ and $f_{o_k} = \sum f_{o_k i}$ and $X = \cup_{i \in S} X_i$. The $i$th sub-problem as a result of the decomposition is then $\{minf_{o_1 i}(\mathbf{x_i}),\ minf_{o_2 i}(\mathbf{x_i}) | \mathbf{x_i} \in X_i\}$. Denote $P$ as the set of non-dominated solutions of the integrated problem, e.g., $f^*_{o_k} = (f^*_{o_1}, f^*_{o_2})$. Further denote the set of non-dominated solutions of the $i$th subsystem as $P_{di}$, e.g., $f^*_{o_k i} = (f^*_{o_1 i}, f^*_{o_2 i})$. Let us assume that there exists a non-dominated solution $\mathbf{x}' \in P$ which is not in the combination non-dominated solution set obtained by,

$P_d = \text{Fouriertrf;}(P_{d1} \times \ldots \times P_{d|S|})$,

where the operator $\times$ is the Cartesian combination of the non-dominated solution sets and Fouriertrf; represents the Pareto filter for dominated solutions in the combination set $\wedge_i P_{di}$. Since $\mathbf{x}'$ is a solution in $P$, we have $f_{o_k}(\mathbf{x}') < f_{o_k}(\mathbf{x}) \quad \forall \mathbf{x} \notin P$ for atleast an objective $k$ and the rest is $f_{o_{k'}}(\mathbf{x}') = f_{o_{k'}}(\mathbf{x})$ for all $k' \neq k$. Without loss of generality, lets also assume that we have two subsystems, e.g., $S=\{1,2\}$. Then $\mathbf{x}' = (\mathbf{x}'_1, \mathbf{x}'_2) \in P$ implies $f_{o_k}(\mathbf{x}') < f_{o_k}(\mathbf{x})$ or alternatively $f_{o_k}(\mathbf{x}'_1) + f_{o_k}(\mathbf{x}'_2) < f_{o_k}(\mathbf{x}_1) + f_{o_k}(\mathbf{x}_2)$ for atleast one objective. The last expression can be satisfied under the following four scenarios:

i. $f_{o_k}(\mathbf{x}'_1) < f_{o_k}(\mathbf{x}_1)$ and $f_{o_k}(\mathbf{x}'_2) \le f_{o_k}(\mathbf{x}_2)$
ii. $f_{o_k}(\mathbf{x}'_1) \le f_{o_k}(\mathbf{x}_1)$ and $f_{o_k}(\mathbf{x}'_2) < f_{o_k}(\mathbf{x}_2)$
iii. $f_{o_k}(\mathbf{x}'_1) \ge f_{o_k}(\mathbf{x}_1)$ and $f_{o_k}(\mathbf{x}'_2) < f_{o_k}(\mathbf{x}_2)$
iv. $f_{o_k}(\mathbf{x}'_1) < f_{o_k}(\mathbf{x}_1)$ and $f_{o_k}(\mathbf{x}'_2) \ge f_{o_k}(\mathbf{x}_2)$

We now show that, for each scenario, there cannot be an $\mathbf{x}' \in P$ but $x' \notin P_d$. Scenarios (i) and (ii) are trivial, since the $P_{d1}$ and $P_{d2}$ would include $\mathbf{x}'_1$ and $\mathbf{x}'_2\mathbf{x}'_2$ or better subsystem solutions with respect to objective $k$. The former case is a contradiction to the assumption of $x' \notin P_d$. The latter case is a contradiction to the assumption of $\mathbf{x}' \in P$ since a better solution implies the dominance of $\boldsymbol{x}'$. In scenarios (iii) and (iv), there is a solution which is better than $\mathbf{x}'$, i.e., $(\mathbf{x}_1, \mathbf{x}'_2)$ and $(\mathbf{x}'_1, \mathbf{x}_2)$, respectively, both of which dominate $\mathbf{x}'$. This is a contradiction to the original assumption of $\mathbf{x}' \in P$.

The above property ensures that any non-dominated solution of the RAP can be obtained by Cartesian combining the Pareto sets of subsystems. In other words, the decomposition based approach does not leave out any non-dominated solution to the RAP. However, some of the solutions obtained through Cartesian combinations could be dominated as established by the following proposition (Phase 3, Fig. 2).

**Proposition 3.** Solutions obtained by Cartesian combining the Pareto sets of subsystems can be dominated.

*Proof:* Without loss of generality, we demonstrate this through an example with two subsystems and two objectives. Consider the following non-dominated solution sets for subsystem 1 and 2: $(f^*_{o_1}(\mathbf{x}_1), f^*_{o_2}(\mathbf{x}_1)) = \{(3,5),(2,6),(4,1)\}$ and $(f^*_{o_1}(\mathbf{x}_2), f^*_{o_2}(\mathbf{x}_2)) = \{(2,5),(4,3),(1,7)\}$. The Cartesian combination results in 9 solutions and 4 of these solutions are dominated by the remainder. For instance, the solution obtained by combining 1st solution of subsystem 1 (3,5) and 2nd solution of subsystem 2 (4,3) is (7,8), which is dominated by the solution (6,6) obtained by combining 3rd solution of subsystem 1 (4,1) and 1st solution of subsystem 2 (2,5). Hence, the Cartesian combination of the non-dominated solutions of multiple subsystems can result in dominated solutions.

In order to maintain the computational efficiency, the decomposition based approach in Fig. 2 alternates between the Cartesian combining and Pareto filtering steps, e.g., filtering out dominated solutions in smaller sets rather than all at once (Phase 3, Fig. 2). The following proposition establishes that this sequential combining and filtering process does eliminate any non-dominated solution of the RAP.

**Proposition 4.** Cartesian combining non-dominated solution sets and filtering out dominated solutions does not result in failure to identify any non-dominated solutions for the RAP.

*Proof:* We prove by contradiction. Lets consider the case with three subsystems and $\mathbf{x}'$ is a solution in RAP's Pareto set $P$, then we have $f_{o_k}(\mathbf{x}') < f_{o_k}(\mathbf{x}) \quad \forall \mathbf{x} \notin P$ for atleast an objective $k$ and the rest is $f_{o_{k'}}(\mathbf{x}') = f_{o_{k'}}(\mathbf{x})$ for all $k' \neq k$. For three subsystem case, this is equivalent to,

$$f_{o_k}(\mathbf{x}'_1) + f_{o_k}(\mathbf{x}'_2) + f_{o_k}(\mathbf{x}'_3) < f_{o_k}(\mathbf{x}_1) + f_{o_k}(\mathbf{x}_2) + f_{o_k}(\mathbf{x}_3).$$

Lets consider that $f_{o_k}(\mathbf{x}'_1) + f_{o_k}(\mathbf{x}'_2)$ is dominated by solution $\mathbf{x}'' = (\mathbf{x}''_1, \mathbf{x}''_2)$ in the Cartesian combination of subsystem 1 and 2, e.g., $f_{o_k}(\mathbf{x}'_1) + f_{o_k}(\mathbf{x}'_2) > f_{o_k}(\mathbf{x}''_1) + f_{o_k}(\mathbf{x}''_2)$. Hence, solution $\mathbf{x}' = (\mathbf{x}'_1, \mathbf{x}'_2)$ will be filtered out and will not appear in the final solution set. However, the solution $\mathbf{x}'' = (\mathbf{x}''_1, \mathbf{x}''_2)$ combined with $\mathbf{x}'_3$ will dominate $\mathbf{x}'$, e.g., $f_{o_k}(\mathbf{x}'_1) + f_{o_k}(\mathbf{x}'_2) + f_{o_k}(\mathbf{x}'_3) > f_{o_k}(\mathbf{x}''_1) + f_{o_k}(\mathbf{x}''_2) \quad + f_{o_k}(\mathbf{x}'_3)$. Hence $\boldsymbol{x}'$ cannot be in RAP's Pareto set $P$, which is a contradiction.

In summary, the decomposition approach determines all the non-dominated solutions of subsystems individually. Proposition 2 guarantees that every solution in the Pareto-optimal set of RAP can be constructed as a Cartesian combination of a non-dominated solution of each subsystems. Proposition 3 establishes that the Cartesian combinations can lead to dominated solutions which needs to be filtered out. Lastly, Proposition 4 states that sequential Cartesian combination and filtering does not eliminate any non-dominated solution of any subsystem that is part of a non-dominated solution of the RAP.

The pseudo-code of the algorithm for the decomposition approach illustrated in Fig. 2 for solving multi-objective RAP of the integrated system is as follows.

### 2.2.1. *Decomposition approach for multi-objective RAP*

Input: $S$, $m_i$, $n_{max,i}$, $r_{ij}$, $c_{ij}$, and $w_{ij}$.
Output: Set of Pareto-optimal solutions for RAP, $P$.
1: $P = \emptyset$
2: For each subsystem $i \in S$, solve the following sub-problem and obtain the Pareto-optimal set for each subsystem, i.e., $P_{di}$:
$$\min f_{R3}(\mathbf{x_i}),\ \min f_{C1}(\mathbf{x_i}),\ \min f_{W1}(\mathbf{x_i})$$
$$1 \leq \sum_{j=1}^{m_i} x_{ij} \leq n_{max,i}$$
3: $i := 1,\ j := 1,\ L := |S|$
4: **While** *flag*$=1$ **do**
5: Cartesian combine the decisions $P_{di}$ and $P_{d(i+1)}$, filter out dominated solutions and store efficient set
$$P_{dj} = \text{Fouriertrf}; (P_{di} \times P_{d(i+1)})$$
6: $i := i+2,\ j := j+1$
7: **If** $L=2$
8: *flag*$=0$
9: **End if**
10: **If** $i \geq L$ **and** *flag*$=1$
11: $i := 1,\ j := ,\ L := \lceil L/2 \rceil$
12: **End if**
13: **End while**
14: $P := P_{d1}$
15: Return $P$

The decomposition algorithm Cartesian combines the non-dominated solution set pairs in Step 5 (Phase 3, Fig. 2). The parameter $L$ represents the number of non-dominated solution sets to be combined in each loop (Steps 4–13). The operator $\lceil \cdot \rceil$ denotes rounding up operation. When the last pair of non-dominated solution set is combined and filtered (i.e., Step 7), the algorithm terminates with the Pareto-optimal set $P$. The above algorithm combines the non-dominated sets consecutively based on the subsystem order, e.g., first subsystem 1 and 2, next subsystem 3 and 4 and so on. However, the order of Cartesian combining is not important as per Proposition 4. For instance, one can first combine subsystems 1 and 2 and then combine subsystem 3 to this set. We also note that the algorithm converges to a non-empty set since $P_{dj} \neq \emptyset$ for any $j$.

The algorithm's computational performance depends on the efficiency of obtaining subsystems' non-dominated sets (Step 2), the efficiency of the filtering approach (Step 5), and the cardinality of the Pareto-optimal solution sets obtained through combination and filtering. By sequentially combining and filtering out the dominated solutions, we gain efficiency in the filtering task. For instance, consider the Simple Cull filtering algorithm which has a complexity of $O(p^2)$ where $p$ is the number of solutions in the set [33]. Let $p_i$ denote the number of non-dominated solutions for subsystem $i=1$, 2, 3, 4. The complexity of the Cartesian combination and filtering is $O(p_1^2 p_2^2)$ for subsystem pair 1 and 2 and is $O(p_3^2 p_4^2)$ for subsystem pair 3 and 4. Let $p_{12}$ and $p_{34}$ denote the number of non-dominated solutions after filtering the subsystem pairs. The filtering complexity of the Cartesian combination of $p_{12}$ and $p_{34}$ is $O(p_{12}^2 p_{34}^2)$. In comparison, Cartesian combination of all four subsystems and then filtering has complexity of $O(p_1^2 p_2^2 p_3^2 p_4^2)$. The worst case scenario is realized when all the solutions generated through Cartesian combinations of subsystem pairs are non-dominated, e.g., $p_{12} = p_1 p_2$ and $p_{34} = p_3 p_4$. In this case, the complexity of filtering for the decomposition approach is $O(p_1^2 p_2^2) + O(p_3{}^2 p_4^2) + O(p_{12}^2\ p_{34}^2) = O(p_1^2 p_2^2) + O(p_3^2 p_4^2) + O(p_1^2 p_2^2 p_3^2 p_4^2) \cong O(p_1^2 p_2^2 p_3^2 p_4^2)$, i.e., approximated equivalent to that of filtering once. Further, lets consider a stylized scenario where all subsystems have equal number of non-dominated solutions, $p = p_i\ \forall i$, and the non-dominated solution ratio is $\rho = p_{12}/p^2 = p_{34}/p^2 < 1$. In this case, the sequential combining and filtering has complexity $\rho^2$ of that of the one-time combining and filtering.

For large RAP instances where many components can be selected from a large set of components, the efficiency of obtaining subsystems' non-dominated sets becomes a critical determinant of the decomposition based approach for solving multi-criteria RAP. The available methods for solving multi-objective optimization problems with non-convex Pareto front can be categorized into exact and approximate methods. The exact methods aim to determine the entire set of Pareto-optimal solutions whereas approximate methods aim to determine an even representative set of the Pareto set. In addition, the approximate methods are further classified into deterministic and stochastic approaches based on whether the solutions found are on the Pareto frontier. Two commonly used deterministic approximate methods are Normal Boundary Intersection (NBI) [34] and Normal Constraint (NC) [35] methods. While these methods are shown to be effective for solving large multi-objective optimization problems, they are not suitable for solving the subsystems' sub-problems in the decomposition based approach. This is because, as stated with Proposition 1, the ability to obtain the Pareto-optimal set for RAP requires identifying the whole non-dominated sets for subsystems. Instead, we improve a recently proposed method called adaptive $\epsilon$-constraint [29], which is guaranteed to find the whole set of non-dominated solutions. We compare the decomposition based approach with NSGA2, a popular stochastic approximation method [27]. The reason for selecting stochastic approximate method (NSGA2) over deterministic approximate methods (NC or NBI) is to eliminate the need for solving nonlinear integer programming problems. While the NSGA2 uses only function evaluations, both NC and NBI would require solving nonlinear integer problems where the nonlinearity is due to the reliability objective. We note that it is possible to linearize the non-linear objective through logarithm transformation and then solve with either NC or NBI. However, these methods will generate an even distribution of Pareto optimal points on the logarithm transformed reliability function space which is not guaranteed to be evenly distributed in the original reliability function space.

### 2.3. Enhanced adaptive $\in$-constraint method

We now describe the Phase 2 of the decomposition-based solution framework (Fig. 2). The traditional $\in$-constraint method is a multi-objective optimization technique proposed by Chankong and Haimes [28] for generating Pareto-optimal solutions and guarantees to find all non-dominated solutions irrespective of the convexity of the Pareto front. This method solves a series of single objective problems of the form $\min f_i(\mathbf{x})\ s.t.\ f_j(\mathbf{x}) \le \epsilon_j \quad \forall j = 1,2,\ldots,m, \quad j \ne i$ where $i \in \{1,2,3,\ldots,m\}$ and $\in = (\in_1, \in_2, \ldots, \in_m)$ are the upper bounds for each corresponding objective function. These upper bounds are iteratively increased or decreased by a pre-defined constant $\Delta$ along the Pareto front for each objective. Fig. 3a illustrates the traditional $\in$-constraint method for bi-objective case.

There are two limitations to the traditional $\in$-constraint method. First, it is necessary to choose a pre-defined constant $\Delta$. Since only one solution can be found in each interval, the discretization has to be fine enough not to "miss" any Pareto-optimal solution. As shown in Fig. 3a, the non-dominated solution $\mathbf{x}_4$ is missed in iteration 3 due to the selection of large $\Delta$. Second, this method may identify dominated solutions since it takes only one objective function into consideration at any time. The selection of solution $\mathbf{x}_5$ (dominated by solution $\mathbf{x}_6$) in Fig. 3a illustrates this scenario.

To cope with the drawbacks of the $\in$-constraint method, Ozlen and Azizoglu [29] presented an adaptive $\in$-constraint method that exploits objective efficiency ranges for solving the multi-objective integer programming (MOIP) problem. Unlike the traditional $\in$-constraint method that determines $\in$ by decreasing a fixed $\Delta$ in each iteration, the adaptive $\in$-constraint method uses an adaptive $\in$ value based on the solution of the previous iteration. This dramatically increases the efficiency of the algorithm while not missing any Pareto-optimal solutions. Further, to avoid identifying dominated solutions, the adaptive $\in$-constraint method uses a lexicographically weighted objective function. Lets consider a problem with $K$ objectives, i.e., $\min\{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_K(\mathbf{x})\}$. The adaptive $\epsilon$-constraint method solves the following single objective optimization problem ($LOP$) with a lexicographically weighted objective function:

$$(LOP): \ \min f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + \ldots + w_K f_K(\mathbf{x})\ s.t. \quad f_k < \varepsilon_k \quad \forall k = 2,3\ldots K \tag{10}$$

where the weights $w_k$ for $k = 2,\ldots,K$ are calculated using the global upper ($f_k^{GUB}$) and lower ($f_k^{GLB}$) bounds on $f_k(\mathbf{x})$ as $w_k = w_{k-1}/(f_k^{GUB} - f_k^{GLB} + 1)$. The global lower (upper) bounds for minimization objectives can be found by minimizing (maximizing) each of the objective functions individually subject to the original constraints. With objectives only taking integer values, this choice of weights ensures that the minimal increment of $f_1(\mathbf{x})$ is 1, which is always greater than the maximal increment of $w_2 f_2(\mathbf{x})$. Similarly, the maximal increment of $w_3 f_3(\mathbf{x})$ is always less than the minimal increment of $w_2 f_2(\mathbf{x})$, and so on. Thus, this weighting makes sure that $f_1$ has the highest priority, it is prioritized over $f_2$, $f_2$ is prioritized over $f_3$, and so on. Note that the final Pareto solution set remains the same, irrespective of the selected lexicographical ordering of the objective criteria; the search needs any single fixed lexicographical ordering. Fig. 3b illustrates the efficacy of this weighting scheme in addressing the two shortcomings of the traditional $\in$-constraint method, e.g., correctly identify all Pareto-optimal solutions.

A requirement for the adaptive $\in$-constraint method is that the objective functions must only take integer values. When an objective $k$ is not integer, then selecting the weights $w_k$ becomes difficult as the minimal increment of $w_k f_k(\mathbf{x})$ cannot be guaranteed to be greater than the maximal increment of $w_{k+1} f_{k+1}(\mathbf{x})$. Hence, the adaptive $\in$-constraint method is not applicable in general instances where the objectives take continuous values. However, if only one of the objective functions is non-integral, then adaptive $\in$-constraint method can be employed by simply assigning this criterion to be the last objective in the lexicographical ordering. By assigning this criterion to be the last objective, we eliminate the need to know the minimum increment of this objective. In the case of our RAP, assuming that cost and weight objectives only take integer values, we assign the transformed reliability objective $f_{R3}(\mathbf{x_i})$, i.e., the log of unreliability, to be the last objective.

This adaptive method is a recursive approach and solves single objective problems by systematically adjusting the objective bounds in a hierarchical manner where the least prioritized objective is highest in the recursion hierarchy [29]. The choice of $\in_k$ for objective $k$ is based on the previously found non-dominated solutions, i.e., during the last recursion of objective $k$. Given the lexicographic ordering of objectives as $1,2,\ldots K$, the adaptive bound $\in_k$ is calculated as $\in_k = \{max(f_k(\mathbf{x})) - 1 : \mathbf{x} \in P_{k-1}\} \quad \forall k = 3,4,\ldots,K$ where $P_{k-1}$ is the set of non-dominated solutions found since the last update of $\in_k$.

The adaptive $\in$-constraint method improves the efficiency over the traditional method significantly. However, a major drawback of this method is that it identifies duplicate solutions which lead to computational inefficiency. In Table 1, we present

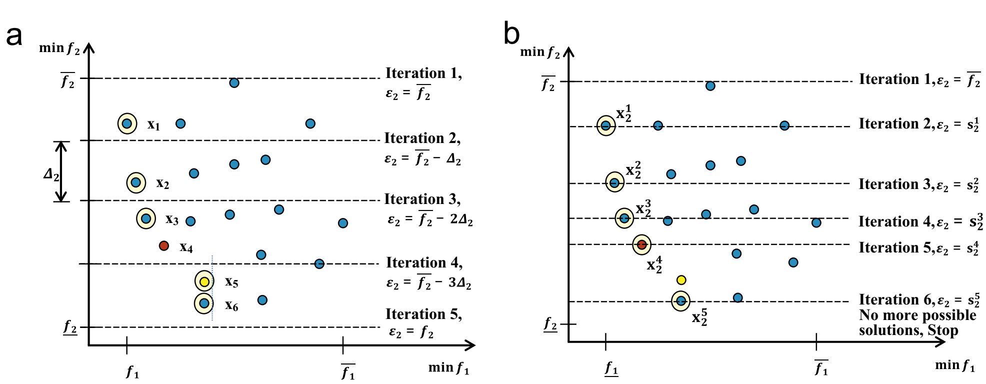


**Fig. 3.** An illustrative example of the (a) traditional $\in$-constraint method and (b) adaptive $\in$-constraint method in [29].

**Table 1**
First two iterations of adaptive $\epsilon$-constraint method for the example in [29].

| Iteration 1 | | | | | Iteration 2 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Sol. # | $f_3 \le \epsilon_3 = 451$ | | | $f_2 \le \epsilon_2$ | Sol. # | $f_3 \le \epsilon_3 = 366$ | | | $f_2 \le \epsilon_2$ |
| | $f_1$ | $f_2$ | $f_3$ | | | $f_1$ | $f_2$ | $f_3$ | |
| 1 | 86 | 214 | 324 | $\epsilon_2 = 411$ | 6 | 86 | 214 | 324 | $\epsilon_2 = 411$ |
| 2 | 96 | 186 | 204 | $\epsilon_2 = 213$ | 7 | 96 | 186 | 204 | $\epsilon_2 = 213$ |
| 3 | 125 | 131 | 342 | $\epsilon_2 = 185$ | 8 | 125 | 131 | 342 | $\epsilon_2 = 185$ |
| 4 | 209 | 128 | 367 | $\epsilon_2 = 130$ | 9 | Infeasible | | | $\epsilon_2 = 130$ |
| 5 | Infeasible | | | $\epsilon_2 = 27$ | | | | | |
| | Max($f_3$) | | 367 | | | Max($f_3$) | | 342 | |

Note that solutions 1, 2, and 3 are identical to 6, 7, and 8, respectively. The adaptive $\epsilon$-constraint could have avoided solving for the latter by checking for the former solutions, which is our proposed enhancement. To formalize this we first note that the problems corresponding to 1, 2, and 3 are relaxations of 6, 7, and 8, respectively, e.g., same objective function and $\epsilon_{23} \le 366 < 451$. Hence, solutions 1, 2, and 3 satisfy the $f_3 \le \epsilon_3 = 366$ and the $f_3 \le \epsilon_2$ where $\epsilon_2$ is 411, 213, and 185, respectively. Therefore, before solving an instance of QUOTE , we scan through the previously solved problems formulations to identify the relaxations for the current instance. This is achieved by comparing the current instance's bound vector with the bound vectors of the non-dominated solutions found thus far. For each relaxation, we test the corresponding solution for feasibility with respect to the current problem. If the solution is feasible, then it must also be optimal for the current instance since it solves the relaxation problem. While this approach eliminates unnecessary problem solutions, it creates additional overhead of scanning through the incumbent set of non-dominated solutions. Hence there is an efficiency trade-off between solving fewer problems and repetitively checking for previous solutions.

the first two iterations of the adaptive $\epsilon$-constraint method for three objective MOIP in [29]. Here the value $\epsilon_3$ in iteration 2 is determined based on the results of iteration 1, i.e., $\epsilon_3 = max\{324,204,342,367\} - 1 = 366$.

For this trade-off, we developed a checking strategy which scans through only the non-dominated solutions collected in the most recent recursion of objective $k=3$, e.g., one level above the lowest level of recursion. Specifically, prior to solving the current *LOP* instance with bounds $\varepsilon_2$ and $\varepsilon_3$, we check for the existence of a non-dominated solution in the set of solutions ($P_3$) found during the previous recursion of $k=3$ that satisfies:

$$\mathbf{x}' = \{\mathbf{x} \in P_3 \,|\, l_2(\mathbf{x}) \ge \varepsilon_2 \; and \, f_3(\mathbf{x}) \le \varepsilon_3\}$$

The $l_2(\mathbf{x}')$ is the upper bound on objective $k=2$ in the LOP instance where the non-dominated solution x′ is found. The condition $l_2(\mathbf{x}') \ge \varepsilon_2$ implies that the problem, which x′ solves, is a relaxation of the current *LOP* instance. The condition $f_3(\mathbf{x}') \le \varepsilon_3$ checks if the non-dominated solution $\mathbf{x}'$ is a feasible solution to the current *LOP*. If there exists a solution $\mathbf{x}' \in P_3$ which satisfies both conditions, then $\mathbf{x}'$ also solves the current *LOP*. We provide the pseudo-code for the enhanced adaptive $\epsilon$-constraint method used in Phase 2 of Fig. 2 below.

#### 2.3.1. Enhanced adaptive $\epsilon$-constraint algorithm

Input: $f_k^{GLB}, f_k^{GUB}$ for $k \in \{2,\ldots,K\}$, $w_k$, $\epsilon_K = f_K^{GUB}$, $flag_k = 1$, $k \in \{2,\ldots,K\}$
Output: Set of Pareto-optimal solutions, $P$
1: $P = \emptyset$
2: **While** $flag_K = 1$ **do**
3: $\varepsilon_{K-1} := f_{K-1}^{GUB}$; $P_{K-1} := \emptyset$; $flag_{K-1} := 1$
4: **While** $flag_{K-1} = 1$ **do**
5: $\varepsilon_{K-2} := f_{K-2}^{GUB}$; $P_{K-2} := \emptyset$; $flag_{K-2} := 1$
6: ⋮
7: **While** $flag_3 = 1$ **do**
8: $\varepsilon_2 := f_2^{GUB}$; $P_2 := \emptyset$; $flag_2 := 1$

Input: $f_k^{GLB}, f_k^{GUB}$ for $k \in \{2,\ldots,K\}$, $w_k$, $\epsilon_K = f_K^{GUB}$, $flag_k = 1$, $k \in \{2,\ldots,K\}$
9: **While** $flag_2 = 1$ **do**
10: **If** $\mathbf{x}' = \{\mathbf{x} \in P_3 \,|\, l_2(\mathbf{x}) \ge \varepsilon_2 \; and \, f_3(\mathbf{x}) \le \varepsilon_3\} = \emptyset$
11: Solve (*LOC*) for $\mathbf{x}$,
12: **If** (*LOC*) is Infeasible
13: $flag_2 := 0$
14: **Else**
15: $P_2 := P_2 \cup \mathbf{x}$; $\varepsilon_2 := f_2(\mathbf{x}) - 1$
16: **End if**
17: **Else**
18: $P_2 := P_2 \cup \mathbf{x}'$; $\varepsilon_2 := f_2(\mathbf{x}') - 1$
19: **End if**
20: **End while**
21: **If** $P_2 = \emptyset$ **Then** $flag_3 := 0$ **End if**
22: $\varepsilon_3 := \{\max(f_3(\mathbf{x})) - 1 : \mathbf{x} \in P_2\}$, $P_3 := P_3 \cup P_2$
23: **End while**
24: ⋮
25: End while
26: If $P_{K-1} = \emptyset$ Then $flag_K := 0$ End if
27: $\varepsilon_K := \{\max(f_K(\mathbf{x})) - 1 : \mathbf{x} \in P_{K-1}\}$ $P_K := P_K \cup P_{K-1}$
28: End while
29: $P := P_K$
30: Return $P$

The main difference of this procedure from that of [29] is checking to see whether there is need to solve (step 10) and storing the previously found solution $\mathbf{x}'$ (step 18). The $flag_k$ indicates whether the recursive solution at objective level $k$ is complete ($flag_k = 0$) or not ($flag_k = 1$). Before the start of recursion at objective level $k$, we initialize the bound at the global upper bound ($\varepsilon_k := f_k^{GUB}$), reset its non-dominated solution set ($P_{K-1} = \emptyset$), and reset its flag ($flag_k = 1$). Prior to solving current *LOP*, we check for recalling a previously found solution $\mathbf{x}'$. If there is no $\mathbf{x}'$ satisfying the condition in step 10, then we solve the incumbent *LOP*. This process is repeated until there is no feasible solution for the current *LOP* and we set $flag_2 := 0$, which indicates the current recursion of objective $k=2$ is complete. For objectives $k \ge 3$, the recursion completion is triggered when the there are no non-dominated solutions found in all lower level objective recursions and we set $flag_k := 0$. The algorithm terminates when there exist no non-dominated solutions found in the current level of $\epsilon_K$.

For the example in [29], the enhancement reduced the number of integer problem (IP) solutions from 56 down to 35 to obtain 15 non-dominated solutions. This corresponds to $(56-35)/56 = 37.5\%$ improvement. To further evaluate the effect of enhancement method, we randomly generated 1000 problem instances of the three objective MOIP in [29]. The average improvement across 1000 is observed to be about 39%.

## 3. Case study example

In this section, we experimentally compare the proposed method with the popular meta-heuristic based approach NSGA2 on a series-parallel RAP example taken from the literature [30]. This series-parallel system consists of three subsystems ($s=3$), with an option of five, four and five types of components in each subsystem ($m_1 = 5$, $m_2 = 4$, $m_3 = 5$) respectively. The maximum number of components is seven ($n_{max,1} = n_{max,2} = n_{max,3} = 7$) in each subsystem. Table 2 presents the component parameters for each subsystem.

The experiments of the proposed method and NSGA2 are run on an HP desktop, with an AMD Quad-Core CPU operating at 2.3 GHz and 8 GB of RAM. The proposed method is coded in

**Table 2**
Component parameters for each subsystem of the series-parallel system.

| Component type: *j* | Subsystem *i* | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | | | 2 | | | 3 | | |
| | *Rel.* | *Cost* | *Weight* | *Rel.* | *Cost* | *Weight* | *Rel.* | *Cost* | *Weight* |
| **1** | .94 | 9 | 9 | .97 | 12 | 5 | .96 | 10 | 6 |
| **2** | .91 | 6 | 6 | .86 | 3 | 7 | .89 | 6 | 8 |
| **3** | .89 | 6 | 4 | .70 | 2 | 3 | .72 | 4 | 2 |
| **4** | .75 | 3 | 7 | .66 | 2 | 4 | .71 | 3 | 4 |
| **5** | .72 | 2 | 8 | | | | .67 | 2 | 4 |

**Table 3**
Comparative results of the proposed method and NSGA2.

| **Proposed decomposition-based method** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **# of Pareto-optimal points** | **6112** | | | | | | |
| **CPU time (s)** | 1728 s | | | | | | |
| | **NSGA2** | | | | | | |
| **Population size** | 100 | 200 | 500 | 1000 | 2000 | 4000 | 5000 |
| **# of pareto points** | 85 | 141 | 289 | 589 | 1109 | 2109 | 2324 |
| **# of pareto-optimal points** | 15 | 27 | 66 | 214 | 558 | 1247 | **1263** |
| **CPU time (s)** | 12s | 28s | 69s | 177s | 442s | 1,231s | 1,699s |

MATLAB® R2008b and NSGA2 is coded in C, available from Deb's Lab [31]. For NSGA2, we vary its population size from 100 up to 5000, with parameters set as follows: *generations* = 100, *crossover probability* = .8 and *mutation probability* = .008. Results from the proposed method and NSGA2 are shown in Table 3.

From Table 3, we note that the proposed method identifies all the 6112 non-dominated points in 1728 s. For NSGA2, the number of Pareto points identified increases as the population size increases. When the population size is 5000, it identified 2324 Pareto points. However, only some of these solutions are Pareto-optimal points (1263 out of 2324). These results illustrate the two shortcomings of the NSGA2. First, it cannot guarantee the generation of all Pareto-optimal points. More importantly, it cannot guarantee that the points generated are Pareto-optimal. Fig. 4 shows the 6112 solutions identified by the proposed method (*) and the 1263 solutions found by NSGA2 (Δ) with population size 5000. Figs. 5 and 6 illustrate these results on bi-objective plots.

For this problem, the number of IPs solved using the adaptive $\epsilon$-constraint method is 5773, while only 1680 IPs needed to be solved using our proposed method to identify all the Pareto-optimal solutions, translating to an improvement of (5773−1680)/5773 = 70.9%. We can see that the proposed method significantly outperforms the adaptive $\epsilon$-constraint method. The additional reason we can identify 6112 Pareto-optimal solutions by 1680 IPs is because we also apply the decomposition scheme.

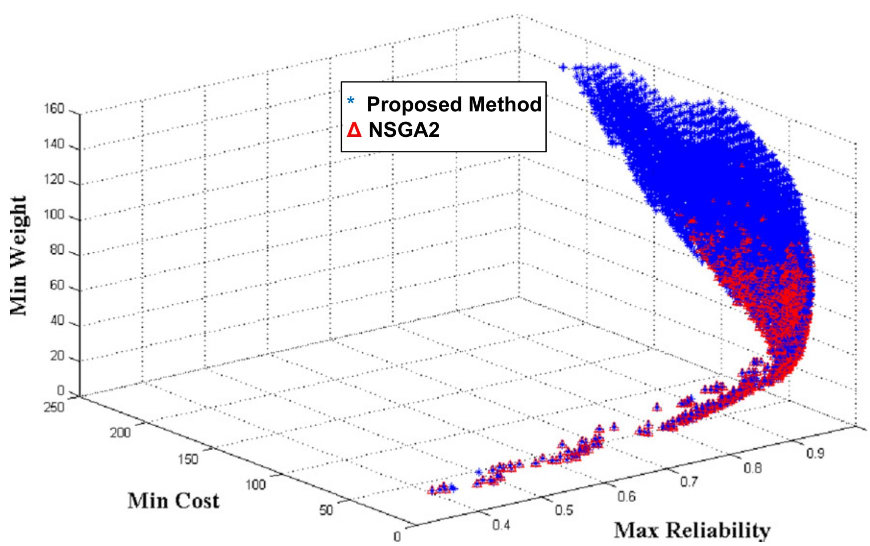


**Fig. 4.** Set of Pareto-optimal solutions obtained by the proposed method and NSGA2 for the RAP problem.

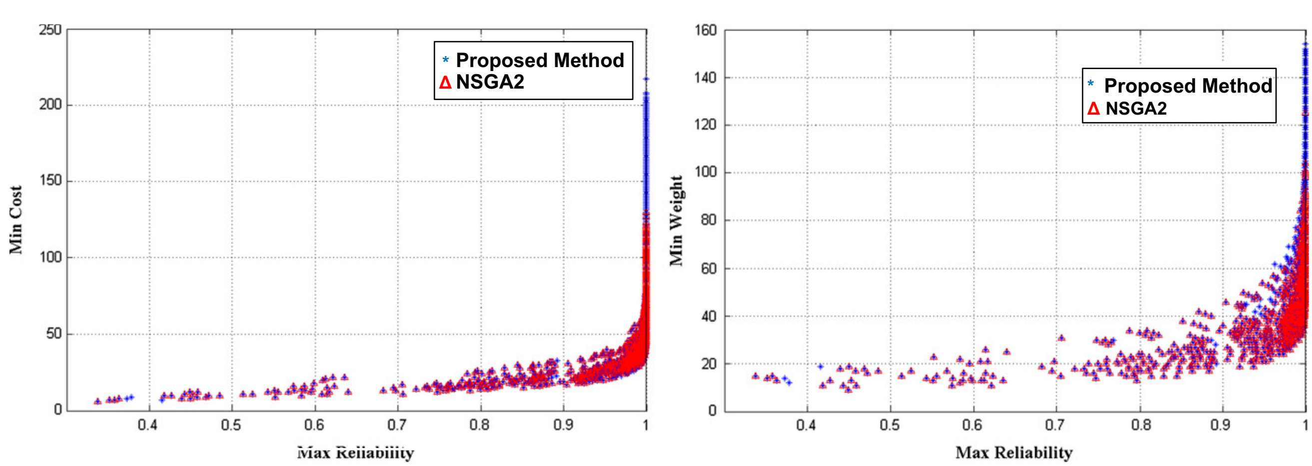


**Fig. 5.** Set of Pareto-optimal solutions in the space of reliability vs. cost (left); reliability vs. weight (right).

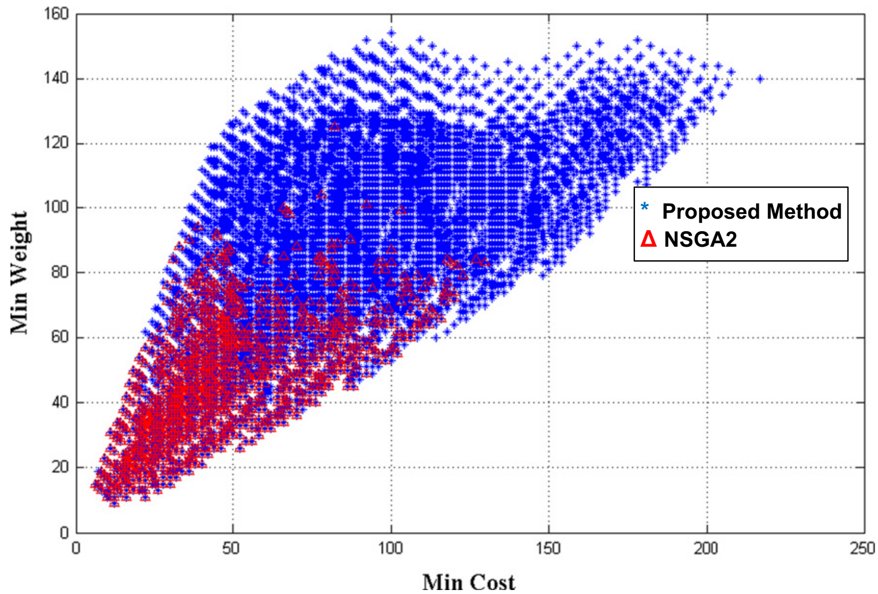


**Fig. 6.** Set of Pareto-optimal solutions in the space of cost vs. weight.

## 4. Conclusion and future work

In this paper, the redundancy allocation problem (RAP) problem is formulated as a multi-objective optimization problem. A decomposition-based approach is proposed to solve RAP problems in series-parallel systems efficiently and exactly. In this approach, we first linearize the reliability objective through logarithmic transformation. We then decompose the multi-objective RAP problem into smaller multi-objective sub-problems and efficiently solve each sub-problem through an enhanced whole Pareto set generation method. These non-dominated solution sets are then Cartesian combined and filtered in a sequential manner to obtain the whole Pareto-optimal set for the RAP. Using a series-parallel system example from the literature, we compare the proposed approach with the meta-heuristic based multi-objective evolutionary algorithm NSGA2 in terms of Pareto set representativeness and computational performance. The proposed method is not only more efficient but also identifies all the Pareto-optimal points.

The proposed method deals with system reliability assuming that the system and its components have binary states. Future work will consider extending the proposed method to account for availability and multi-state systems in redundancy allocation problems. The proposed approach aims at identifying the whole Pareto-optimal solution set for multi-objective RAP. However, for large instances, the size of the whole Pareto-optimal solution set could be too large for the decision maker to effectively analyze and compare promising solutions [36–38]. Hence, there is usually a further step necessary to prune and reduce the Pareto-optimal set. The two practical approaches are pruning (1) by nonnumeric ranking preferences and (2) by clustering of the solutions in the objective space. A future research direction is to explore ways to reduce the pruning need by generating only those Pareto-optimal solutions that would eventually be part of the pruned set.

## Acknowledgments

We are thankful to the three reviewers for their constructive feedback and comments which helped greatly improve the presentation clarity and quality of the analytical derivations.

## References


[1] Oiddir R, Rahli M, Meziane R, Zeblah A. Ant colony optimization for new redesign problem of multi-state electrical power systems. Journal of Electrical Engineering 2004;55:57–63.

[2] Levitin G, Lisninaski A. A new approach to solving problem of multistate system reliability optimization. Quality and Reliability Engineering International 2001;17:93–4.

[3] Lyu MR, Rangarajan S, van Moorsel APA. Optimal allocation of test resources for software reliability growth modeling in software development. IEEE Transactions on Reliability 2002;51:183–92.

[4] Bellman R, Dreyfus S. Dynamic programming and the reliability of multicomponent devices. Operations Research 1958;6:200–6.

[5] Misra KB. Dynamic programming formulation of the redundancy allocation problem. International Journal of Mathematical Education in Science and Technology 1971;2:207–15.

[6] Fyffe DE, Hines WW, Lee NK. System reliability allocation and a computational algorithm. IEEE Transaction on Reliability 1968;17:64–9.

[7] Bulfin RL, Liu CY. Optimal allocation of redundant components for large systems. IEEE Transactions on Reliability 1985;34:241–7.

[8] Misra KB, Sharma U. An efficient algorithm to solve integer-programming problems arising in system-reliability design. IEEE Transactions on Reliability 1991;40:81–91.

[9] Billionnet A. Redundancy allocation for series-parallel systems using integer linear programming. IEEE Transactions on Reliability 2008;57:507–16.

[10] Coit DW, Smith AE. Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Transactions on Reliability 1996;45:254–66.

[11] Liang YC, Smith AE. An ant colony optimization algorithm for the redundancy allocation problem (RAP). IEEE Transactions on Reliability 2004;53:417–23.

[12] Painton L, Campbell J. Genetic algorithms in optimization of system reliability. IEEE Transactions on Reliability 1995;44:172–8.

[13] Kulturel-Konak S, Smith AE, Coit DW. Efficiently solving the redundancy allocation problem using tabu search. IIE Transactions 2003;35:515–26.

[14] Ravi V, Murty BSN, Reddy PJ. Nonequilibrium simulated annealing-algorithm applied to reliability optimization of complex systems. IEEE Transactions on Reliability 1997;46:233–9.

[15] Ramirez-Marquez JE, Coit. DW. A heuristic for solving the redundancy allocation problem for multi-state series-parallel systems. Reliability Engineering & System Safety 2004;83:341–9.

[16] Tillman FA, Hwang CL, Kuo W. Determining component reliability and redundancy for optimum system reliability. IEEE Transactions on Reliability 1977;26:162–5.

[17] Zia L, Coit DW. Redundancy allocation for series-parallel systems using a column generation approach. IEEE Transactions on Reliability 2010;59:706–17.

[18] Dhingra AK. Optimal apportionment of reliability and redundancy in series systems under multiple objectives. IEEE Transactions on Reliability 1992;41:576–82.

[19] Zafiropoulos EP, Dialynas EN. Reliability and cost optimization of electronic devices considering the component failure rate uncertainty. Reliability Engineering & System Safety 2004;84:271–84.

[20] Busacca PG, Marseguerra M, Zio E. Multiobjective optimization by genetic algorithms: application to safety systems. Reliability Engineering & System Safety 2001;72:59–74.
[21] Tian ZG, Zuo MJ. Redundancy allocation for multi-state systems using physical programming and genetic algorithms. Reliability Engineering & System Safety 2006;91:1049–56.
[22] Marseguerra M, Zio E, Podofillini L. A multiobjective genetic algorithm approach to the optimization of the technical specifications of a nuclear safety system. Reliability Engineering & System Safety 2004;84:87–99.
[23] Taboada HA, Baheranwala F, Coit DW, Wattanapongsakorn N. Practical solutions for multi-objective optimization: an application to system reliability design problems. Reliability Engineering & System Safety 2007;92:314–22.
[24] Taboada HA, Espiritu JF, Coit DW. MOMS-GA: A multi-objective multi-state genetic algorithm for system reliability optimization design problems. IEEE Transactions on Reliability 2008;57:182–91.
[25] Kulturel-Konak S, Coit DW, Baheranwala F. Pruned pareto-optimal sets for the system redundancy allocation problem based on multiple prioritized objectives. Journal of Heuristics 2008;14:335–57.
[26] Salazar D, Rocco CM, Galvan BJ. Optimization of constrained multiple-objective reliability problems using evolutionary algorithms. Reliability Engineering & System Safety 2006;91:1057–70.
[27] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-I. IEEE Transactions on Evolutionary Computation 2002;6:182–97.
[28] Chankong V, Haimes Y. Multiobjective decision marking theory and methodology. New York: Elsevier Science; 1983.
[29] Ozlen M, Azizoglu M. Multi-objective integer programming: a general approach for generating all non-dominated solutions. European Journal of Operational Research 2009;199:25–35.
[30] Taboada H, Coit DW. A new multiple objective evolutionary algorithm for reliability optimization of series-parallel systems. International Journal of Applied Evolutionary Computation 2012;3(2):1–18.
[31] Deb K. Multi-objective NSGA-II code in C; 2005. Available from: 〈http://www.iitk.ac.in/kangal/codes.shtml〉.
[32] Ehrgott M. Lecture notes in economics and mathematical systems, vol. 491. Multicriteria Optimization: Springer; 2005.
[33] Yukish M. Algorithms to identify pareto pointsin multi-dimensional data sets. PhD thesis, Pennsylvania State University; August, 2004, p. 1–126.
[34] Messac A, Ismail-Yahaya A, Mattson CA. The normalized normal constraint method for generating the Pareto frontier. Structural and Multidisciplinary Optimization 2003;25:86–98.
[35] Das I, Dennis JE. Normal-boundary intersection: a new method for generating the Pareto surface in nonlinear multicriteria optimization problems. SIAM Journal on Optimization 1998;8:631–57.
[36] Taboada H, Baheranwala F, Coit DW, Wattanapongsakorn N. Practical solutions for multi-objective optimization: an application to system reliability design problems. Reliability Engineering & System Safety 2007;92(3):314–22.
[37] Taboada H, Coit DW. Data clustering of solutions for multiple objective system reliability optimization problems. Quality Technology & Quantitative Management Journal 2007;4(2):35–54.
[38] Kulturel-Konak S, Coit DW, Baheranwala F. Pruned pareto-optimal sets for the system redundancy allocation problem based on multiple prioritized objectives. Journal of Heuristics 2008;14(4):335–57.