On the Optimal Redundancy Allocation for Multi-State Series-Parallel Systems under Epistemic Uncertainty

Mu-Xia Sun, Yan-Fu Li and Enrico Zio

***Abstract* — In this paper, we study the redundancy allocation problem (RAP) for multi-state series-parallel systems (MSSPSs). For each multi-state component, the exact values of its state probabilities are assumed to be unknown, due to epistemic uncertainty (EU), and only conservative lower and upper bounds of them are given. The objective of the RAP is to simultaneously maximize the supremum and infimum of the system's uncertain availability, under a cost constraint. The problem is two-stage and multi-objective. In this work, we: 1. provide a linear-time algorithm to obtain the component state distribution, under which the uncertain system availability will be at its supremum or infimum; 2. show that the problem is reducible to one-stage; 3. analyze the landscape of MSSPS RAP under EU and propose a modified NSGA-II, with targeted designs of repair and local search operation. The proposed algorithm is compared with standard NSGA-II on multiple benchmarks. The results show that the proposed algorithm significantly outperforms the standard NSGA-II in both optimality and time efficiency.**

Index Terms*— *redundancy allocation; multi-state series-parallel systems; epistemic uncertainty; stochastic dominance; approximate system availability analysis; repair algorithm; local search.

Acronyms

CDF	cumulative probability distribution function
GA	genetic algorithm
MOEA	multi-objective evolutionary algorithm
RAP	redundancy allocation problem
MSS	multi-state system
MSSPS	multi-state series parallel system
MSSPS RAP EU	MSSPS RAP under epistemic uncertainty
RV	random variable
LP	linear programming
iff.	if and only if

I. INTRODUCTION

The redundancy allocation problem (RAP) is a well-known combinatorial optimization problem in the field of reliability engineering, [1]. Series-parallel structures are typically considered because they are common in practice. Traditionally, binary-state reliability models are often used, in which a system and its components only have two performance states: perfect functioning and complete failure. In real-world engineering systems, however, this binary-state description is often insufficient, and intermediate states between the above two extremes have to be considered. For these reasons, the multi-state models have become increasingly popular [1, 3~6,34].

In recent years, significant research efforts have been devoted to the solution of multi-state series-parallel system (MSSPS) RAP [1, 3~6, 31-32], which was first introduced in [30]. Due to the hardness of the problem, meta-heuristics [1,3~6] have been often used to solve the problem, even though they can become time-consuming, especially on large systems [1].

On the other hand, theoretical analysis for MSSPS RAP has been lacking. To the authors' knowledge, neither effective exact algorithms nor approximated algorithms with bounded error have been reported in MSSPS RAP literature. Meanwhile, theoretical guidance for heuristic design of MSS RAP is also rarely reported, whereas it is important because the application of RAP to multi-state models often requires exhaustive computational recourses. Indeed, the difficulty of solving MSSPS RAP is not only due to the well-known hardness in MSS reliability evaluation, but also to the RAP's discrete, probabilistic and nonlinear nature.

In reliability theory, the state of a component is often assumed to be aleatory uncertain, i.e., the uncertainty is due to the inherent randomness of the component behavior. Aleatory uncertainty is irreducible since it is the natural variability of the component. On the other hand, epistemic uncertainty is caused by the lack of data or knowledge, which often leads to inaccuracies in the structure/parameters of a stochastic/probabilistic model that is built to address aleatory uncertainty. In reliability engineering, such inaccuracy is often critical and has to be accounted, especially for those critical systems/components which are required to be highly reliable (e.g., the valves and pumps in nuclear plant or aircraft) [2, 12~14,33,36,37]. A good example can be found in the illustrated distributed generation system in section VI.B, [37]: firstly, it is difficult to estimate the mechanical failure rates in a wind turbine or transformer, due to the lack of data or the imprecision of the physical models. The related failure parameters are often elicited from by human

experts, given by interval values or only linguistically. On the other hand, stochastic models for wind power output are often not satisfyingly accurate, so that the state probability of wind output is often presented as interval, in order to provide some distributional robustness. In MSS literature, epistemic uncertainty above the MSS and component state distribution has been carried out by interval or fuzzy multi-state models [13, 14, 33, 37]. Unfortunately, the existing models only produce inexact probabilistic bounds, since the interval/fuzzy representations are often in conflict with the law of total probability.

Traditional MSS RAP aims at obtaining optimal trade-offs between the costs and availability of a multi-state system. However, when facing epistemic uncertainty on the components' state distributions, the system's availability becomes uncertain as well, which has to be accounted for in RAP problems. In this work, we consider for an MSS RAP in the formulation of availability maximization, i.e., to maximize both the supremum and infimum of the system's uncertain availability under a maximum system cost constraint. The exact state probability values are assumed to be unknown, while conservative upper and lower bounds of them are given. Now, we see the RAP is a two-stage multi-objective optimization problem: while the outer stage problem searches for Pareto-optimal system compositions that maximize both the supremum and infimum of the uncertain system availability, the inner stage problem determines the supremum and infimum of the uncertain availability of a given system. In this paper, we name this optimization problem as MSSPS RAP EU (Epistemic Uncertainty).

The original contributions of this work are:

1. We show that, for the studied MSSPS, for each of its component there exists a unique probability vector, such that the uncertain system availability will be at its infimum/supremum under the uncertain set, if the state probability distributions of all the components are given by these probability vectors.

2. We show that, each of the above probability vector is the unique optimal solution of a multi-objective linear programming problem, and each of the problem is linear time solvable.

3. We show that, the inner stage problem of the studied MSSPS RAP EU is reducible to a one-stage multi-objective optimization problem.

4. We analyze the landscape of the reduced problem, and discuss some basic principles for its heuristic solutions. The principles can also be implemented to related single-objective MSSPS RAPs, if they are reducible from the studied MSSPS RAP EU.

5. We extend NSGA-II [15] to improve its performance on the MSSPS RAP EU.

The rest of this paper is organized as follows: in Section II, the formulation of MSSPS RAP EU is presented; in Section III, we reduce the studied MSSPS RAP EU into a one stage problem; in Section IV, we analyze the landscape of the reduced problem and propose repair and local search operators for its heuristic solutions; in Section V, we propose our modified version of NSGA-II, namely co-SP-NSGA-II to solve the reduced MSSPS RAP EU; in Section VI, we test the proposed algorithm, competing with the standard NSGA-II. For comparison, previous results and benchmarks for single-objective MSSPS RAPs are also used/extended in this section, to show that the proposed algorithm can efficiently obtain the currently best solutions reported in the single-objective MSSPS RAP literature. In section VII, we conclude this work and discuss the advantages, drawbacks of the proposed evolutionary approach, as well as the potential extensions for future evolutionary solutions of MSSPS RAPs.

II. The Statement of the Problem

Notation

N	number of subsystems in a MSSPS
i	index of the subsystems, $i \in \{1,2,\dots,N\}$, unless otherwise stated
n_i	number of component versions proposed on the market, for the subsystem i
j_i	component version index of subsystem i
x_{j_i}	number of components in version j_i that installed on the MSSPS
X_{j_i}	the maximum x_{j_i} that available
$\vec{x}_i$	$(x_{1_i}, x_{2_i}, \dots, x_{j_i}, \dots, x_{n_{i_N}})$
$\boldsymbol{x}$	$(\vec{x}_1, \vec{x}_2, \dots, \vec{x}_i, \dots, \vec{x}_N)$, the system composition vector
c_{j_i}	unit component cost for the components in version j_i
$\vec{c}_i$	$(c_{1_i}, c_{2_i}, \dots, c_{j_i}, \dots, c_{n_{i_i}})$
$\boldsymbol{c}$	$(\vec{c}_1, \vec{c}_2, \dots, \vec{c}_i, \dots, \vec{c}_N)$
$C_i(\vec{x}_i)$	cost of subsystem i
$C_s(\boldsymbol{x})$	cost of the MSSPS
h_{j_i}	index of the components in version j_i

$G_{h_{j_i}}$	performance of the h_{j_i}th component in version j_i
F_{j_i}	cumulative probabilistic distribution function of $G_{h_{j_i}}$
k_{j_i}	index of performance levels (i.e., state) for components in version j_i
K_{j_i}	number of performance levels for components of version j_i
$g_{k_{j_i}}$	k_{j_i}th performance level of components in version j_i
supp_{j_i}	$\{g_{k_{j_i}}\}$, $k_{j_i} \in \{1,2,\dots,K_{j_i}\}$, the support of F_{j_i}
$p_{k_{j_i}}$	$\Pr\left(G_{h_{j_i}} = g_{k_{j_i}}\right)$, the state probability for the k_{j_i}th state, of components in version j_i
$\tilde{p}_{k_{j_i}-}$	lower bound of $p_{k_{j_i}}$
$\tilde{p}_{k_{j_i}+}$	upper bound of $p_{k_{j_i}}$
$\vec{P}_{j_i}$	$(p_{0_{ij}}, p_{1_{ij}}, \dots, p_{k_{ij}}, \dots, p_{K_{ij}})$
$\vec{P}_i$	$(\vec{P}_{1_i}, \vec{P}_{2_i}, \dots, \vec{P}_{j_i}, \dots, \vec{P}_{n_{i_i}})$
$\boldsymbol{P}$	$(\vec{P}_1, \vec{P}_2, \dots, \vec{P}_i, \dots, \vec{P}_N)$, the system probability vector
$\wp$	$\left\{\boldsymbol{P} \middle\| p_{j_i} \in \left[\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}\right], \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}} = 1\right\}$, the uncertainty set
$G_i(\vec{x}_i, \vec{P}_i)$	state of subsystem i
$G_s(\boldsymbol{x}, \boldsymbol{P})$	state of MSSPS
D	demand performance of MSSPS
k_D	index of demand performance levels, $k_D = 1,2,\dots,K_D$
K_D	number of demand performance levels
g_{k_D}	k_Dth performance level of system demand D
supp_D	$\{g_{k_D}\}$, $k_D \in \{1,2,\dots,K_D\}$, the support of D
p_{k_D}	$\Pr(G_s = g_{k_D})$, the probability that the system demand D is at its k_Dth state
$A(\boldsymbol{x}, \boldsymbol{P})$	MSSPS availability
$\sup_{\boldsymbol{P} \in \wp} A(\boldsymbol{x}, \boldsymbol{P})$	supremum of the uncertain MSSPS availability

$\inf_{\boldsymbol{P}\in\wp} A(\boldsymbol{x},\boldsymbol{P})$	infimum of the uncertain MSSPS availability
$\boldsymbol{P}_{-}$	number of subsystems in a MSSPS
$\wp_{j_i}$	feasible region of $\vec{P}_{j_i}$, $\left\{\vec{P}_{j_i} \middle\| p_{k_{j_i}} \in \left[\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}\right], \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}} = 1\right\}$
$\boldsymbol{\mathcal{F}}_{j_i}$	feasible region of j_i, $\{j_i \| j_i \in \{1,2,\dots,n_i\}, i \in \{1,2,\dots N\}\}$
$\boldsymbol{\mathcal{F}}_x$	feasible region of $\boldsymbol{x}$, $\{\boldsymbol{x} \| x_{j_i} \in \{0,1,2,\dots,X_{j_i}\}, j_i \in \boldsymbol{\mathcal{F}}_{j_i}\}$
$\mathcal{I}$	$\{i = 1,2,\dots,N\}$, the set of subsystem indexes
$\mathcal{P}$	a partition of the subsystem index set $\mathcal{I}$
$\mathcal{P}_s$	the sth element of $\mathcal{P}$
$\mathcal{A}_s$	$\Pr(\min_{i\in\mathcal{P}_s} G_i \geq D)$, the MSSPS availability when the system is only composed of subsystems from $\mathcal{P}_s$.
A_i	$\Pr(G_i \geq D)$, the availability of the ith subsystem
$\bar{A}_{(\cdot)}$	$1 - A_{(\cdot)}$, the unavailability of a system or subsystem indexed by $(\cdot)$.
$\lfloor\cdot\rfloor, \lceil\cdot\rceil$	round-down and round-up operator for non-integer real numbers

A. Deterministic MSSPS RAP

The MSSPS consists of N subsystems in series, where each subsystem consists of several functionally equivalent components in parallel (Figure 1). There are n_i component versions available on the market for each subsystem i, where the state of each component is characterized only by the component's performance. The performance of each multi-state component is a discrete random variable and the performances of the components in same version are i.i.d.

In this paper, we assume the multi-state components' performances are mutually stochastic-independent. For each component version j_i, the maximum number of components available on market X_{j_i}, the unit component cost c_{j_i} and the components' performance distribution F_{j_i} are specified. Then, the performance of the system is defined as

$$G_s = \min_{i=1,2\dots,N} G_i \tag{1}$$

where

$$G_i = \sum_{j_i=1}^{n_i} \sum_{h_{j_i}=1}^{x_{j_i}} G_{h_{j_i}}$$

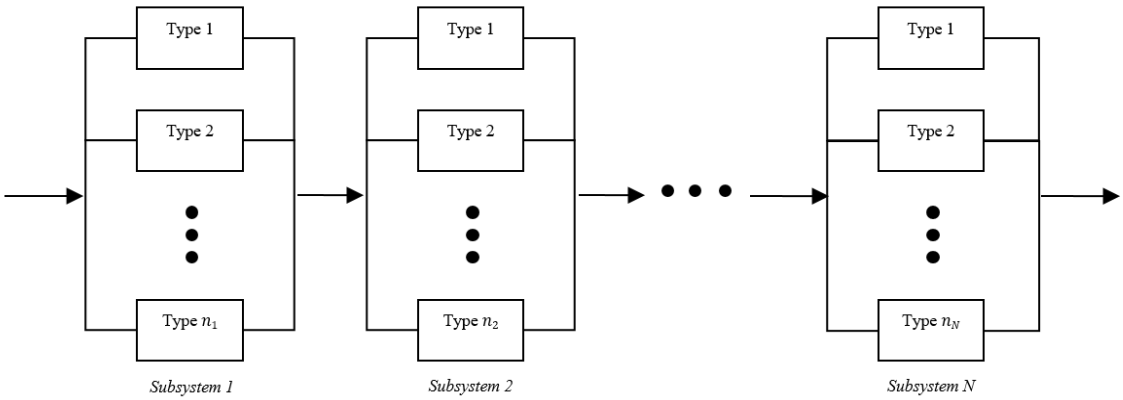


Figure 1. The MSSPS

The MSSPS RAP for availability maximization, then, generally takes the form

$$\max_{\boldsymbol{x}} A(\boldsymbol{x}, \boldsymbol{P}) = \Pr(G_s \geq D)$$

s.t.

$$\boldsymbol{c}^T \boldsymbol{x} \leq C_0 \tag{2}$$

$$\boldsymbol{x} \text{ integer, } 0 \leq x_{j_i} \leq X_{j_i}$$

B. MSSPS RAP under epistemic uncertainty

Following the previous work [33], we assume that the exact value of each state probability $p_{k_{ij}}$ is unknown and only a lower and upper bound of it are provided. For each k_{j_i},

$$p_{k_{j_i}} \in [\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}]$$

$$\sum_{k_{j_i}=1}^{K_{j_i}} \tilde{p}_{k_{j_i}-} \leq \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}} = 1 \leq \sum_{k_{j_i}=1}^{K_{j_i}} \tilde{p}_{k_{j_i}+} \tag{3}$$

which describe the uncertain set $\wp$. For a fixed system $\boldsymbol{x_0}$, to obtain the infimum $\inf_{\boldsymbol{P}\in\wp} A(\boldsymbol{x_0}, \boldsymbol{P})$ and supremum $\sup_{\boldsymbol{P}\in\wp} A(\boldsymbol{x_0}, \boldsymbol{P})$ is equivalent to solve the following optimization problems (4) and (5):

$$\min_{\boldsymbol{P}\in\wp} A(\boldsymbol{x_0}, \boldsymbol{P})$$

$$\text{s.t.} \quad \wp = \left\{ \boldsymbol{P} | p_{k_{j_i}} \in \left[\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+} \right], \sum_{k_{j_i}=1}^{k_{j_i}} p_{k_{j_i}} = 1 \right\} \tag{4}$$

$$\max_{\boldsymbol{P}\in\wp} A(\boldsymbol{x_0}, \boldsymbol{P})$$

$$\text{s.t.} \quad \wp = \left\{ \boldsymbol{P} | p_{k_{j_i}} \in \left[\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+} \right], \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}} = 1 \right\} \tag{5}$$

It is noted that the "upper and lower bounds" for the uncertain system availability defined in [33] and related interval-sized fuzzy approaches [13], [14] are only approximations of $\sup_{\boldsymbol{P}\in\wp} A(\boldsymbol{x}, \boldsymbol{P})$ and $\inf_{\boldsymbol{P}\in\wp} A(\boldsymbol{x}, \boldsymbol{P})$, and the worst-case approximation error bounds are not guaranteed. Without loss of generality, in this paper, we formulate the MSSPS-RAP EU as following:

$$\max_{\boldsymbol{x}} \max_{\boldsymbol{P}} A(\boldsymbol{x}, \boldsymbol{P})$$

$$\max_{\boldsymbol{x}} \min_{\boldsymbol{P}} A(\boldsymbol{x}, \boldsymbol{P})$$

$$s.t. \quad \boldsymbol{c}^T \boldsymbol{x} \leq C_0$$

$$x_{j_i} \in \{0,1,2,\dots,X_{j_i}\} \tag{6}$$

$$p_{k_{j_i}} \in [\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}]$$

$$\sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}} = 1$$

III. Solution Method To The Inner Stage Problem: A Decomposition Approach

In this section, we show Problem (6) is reducible into a one stage problem. We first reduce the second-stage

problem into small multi-objective linear programming problems, and, then, prove that each of the small problems has a unique optimal solution, and the solution is linear-time computable. Finally, we show that the optimal solutions are independent of the system composition $\boldsymbol{x}$, and reduce the problem into one-stage.

Definition 3.1

The usual stochastic order "$\preccurlyeq$" between two RVs X and Y (or their CDFs F_X, F_Y) is defined as

$$X \preccurlyeq Y \ \ iff. \ \forall t \in \mathbb{R}: \ F_X(t) \geq F_Y(t) \tag{7}$$

where $F_X(t)$ and $F_Y(t)$ are the cumulative distribution functions (CDFs) of X and Y, respectively. Furthermore, we have

$$\begin{gathered} X \prec Y \ \ iff. \ X \preccurlyeq Y \wedge X \neq Y \\ Y \succcurlyeq X \ \ iff. \ X \preccurlyeq Y \end{gathered} \tag{8}$$

Lemma 3.2 [16]

Assume $X_1, X_2, \dots, X_m, \ Y_1, Y_2, \dots, Y_m$ are independent RVs satisfying $X_i \preccurlyeq Y_i, \ i = 1,2, \dots, m$; then

$$\sum_{j=1}^{m} X_j \preccurlyeq \sum_{j=1}^{m} Y_j \tag{9}$$

Corollary 3.3

Assume $\boldsymbol{P}_- = (\vec{P}_{1-}, \vec{P}_{2-}, \dots, \vec{P}_{i-}, \dots \vec{P}_{N-})$, is a feasible solution of Problem (4), where $\vec{P}_{i-} = (\vec{P}_{1_i-}, \vec{P}_{2_i-}, \dots, \vec{P}_{j_i-}, \dots, \vec{P}_{n_{i_i}-})$, $\vec{P}_{j_i-} = (p_{1_{j_i-}}, p_{2_{j_i-}}, \dots, p_{k_{j_i-}}, \dots, p_{K_{j_i-}})$, and it satisfies

$$\forall j_i \in \boldsymbol{\mathcal{F}}_{j_i} \ \ \forall \vec{P}_{j_i} \in \wp_{j_i} : F_{j_i}(\vec{P}_{j_i-}) \preccurlyeq F_{j_i}(\vec{P}_{j_i}) \tag{10}$$

then

$$\forall \boldsymbol{x_o} \in \boldsymbol{\mathcal{F}}_x : \inf_{\boldsymbol{P} \in \wp} A(\boldsymbol{x_o}, \boldsymbol{P}) = A(\boldsymbol{x_o}, \boldsymbol{P}_-) \tag{11}$$

Proof

For the readers conveniences, in the rest of the paper we will abridge the logic formula "$\forall(\cdot) \in \boldsymbol{\mathcal{F}}_{(\cdot)}$" into "$\forall(\cdot)$", where $\boldsymbol{\mathcal{F}}_{(\cdot)}$ is the feasible region of the variable $(\cdot)$ in the studied problem. Let us consider a MSSPS with arbitrary system composition $\boldsymbol{x_o}$ and system probability vector $\boldsymbol{P}$. With (1), we have

$$\Pr(G \geq d) = \prod_{i=1}^{N} \Pr(G_i \geq d) \tag{12}$$

for arbitrary constant d. Then, with (2) we have

$$A(\boldsymbol{x_o}, \boldsymbol{P}) = \sum_{k_D=1}^{K_D} \Pr\big(G \geq d_{k_D}\big)\, p_{k_D} = \sum_{k_D=1}^{K_D} \prod_{i=1}^{N} \Pr(G_i \geq d_{k_D}) p_{k_D} \tag{13}$$

Let us define $G_{i\_} = G_i(\vec{P}_{1_i-}, \vec{P}_{2_i-}, \dots, \vec{P}_{n_{i_i}-})$. According to Lemma 3.2, we have

$$\forall i\colon G_{i\_} \preccurlyeq G_i(\vec{P}_{1_i}, \vec{P}_{2_i}, \dots, \vec{P}_{n_{i_i}}) \;\; if \;\; \forall j_i\colon G_{j_i}(\vec{P}_{j_i-}) \preccurlyeq G_{j_i}(\vec{P}_{j_i}) \tag{14}$$

thus

$$\forall i \forall k_D\colon \Pr(G_{i\_} \leq d_{k_D}) \geq \Pr(G_i \leq d_{k_D}) \tag{15}$$

thus

$$\forall i \forall k_D\colon \sum_{k_D=1}^{K_D} \prod_{i=1}^{N} \Pr(G_{i\_} \geq d_{k_D}) p_{k_D}) \leq \sum_{k_D=1}^{K_D} \prod_{i=1}^{N} \Pr(G_i \geq d_{k_D}) p_{k_D}) = A(\boldsymbol{x_o}, \boldsymbol{P}) \tag{16}$$

i.e. for the given MSSPS, we have

$$\inf_{\boldsymbol{P}} A(\boldsymbol{x_o}, \boldsymbol{P}) = \sum_{k_D=1}^{K_D} \prod_{i=1}^{N} \Pr(G_{i\_} \geq d_{k_D}) p_{k_D} \tag{17}$$ ■

According to definition 3.1, to verify the existence of $\boldsymbol{P}_{\_}$ is equivalently to verify that the following multi-objective LP problem (18) has a *unique optimal solution*:

$$\begin{aligned} \text{for } k_{j_i} = 1,2,\dots,K_{j_i}\colon \quad & \max_{\vec{P}_{j_i\sim}} F_{j_i}(k_{j_i}, \vec{P}_{j_i\sim}) \\ \text{s.t.} \quad & p_{k_{j_i\sim}} \in [\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}] \\ & \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i\sim}} = 1 \end{aligned} \tag{18}$$

where $\vec{P}_{j_i\sim} = (p_{1_{j_i\sim}}, p_{2_{j_i\sim}}, \dots, p_{k_{j_i\sim}}, \dots, p_{K_{j_i\sim}})$, and $F_{j_i}(t, \vec{P}_{j_i\sim})$ is the cumulative state probability distribution function for each component in version j_i, when its state probability vector is assumed to be $\vec{P}_{j_i} = \vec{P}_{j_i\sim}$. Without loss of generality, we assume that the state performances for each component version j_i, i.e., the elements in $\text{supp}_{j_i} = \{g_{k_{j_i}}\}$, are sorted in ascending order with respect to k_{j_i}. Then, we have

Theorem 3.4

A probability vector $\vec{P}_{j_i}^{*}$ is the *unique optimal solution* of Problem (18) if it is feasible and satisfies:

$$\exists k_o \in \{1,2,\dots,K_{j_i}\},\ \text{s.t.} \qquad p^*_{k_{j_i}} = \begin{cases} \tilde{p}_{k_{j_i}+}, \text{for } k_{j_i} < k_o \\ \tilde{p}_{k_{j_i}-}, \text{for } k_{j_i} > k_o \\ 1 - \sum_{k_{j_i}=1}^{k_o-1} \tilde{p}_{k_{j_i}+} - \sum_{k_{j_i}=k_o+1}^{K_{j_i}} \tilde{p}_{k_{j_i}-}, \text{for } k_{j_i} = k_o \end{cases} \tag{19}$$

Proof

Assume $\vec{P}^*_{j_i}$ is feasible and satisfies (19). Let us consider arbitrary feasible solution $\vec{P}_{j_i\sim} \neq \vec{P}^*_{j_i}$ and define

$$k'_o = \begin{cases} k_o, \text{if } p^*_{k_o} \geq p_{k_o} \\ k_o - 1, \text{if } p^*_{k_o} < p_{k_o} \end{cases} \tag{20}$$

then

$$\forall k \in \{1,2,\dots,K_{j_i}\}: \begin{cases} p^*_{k_{j_i}} - p_{k_{j_i}\sim} \geq 0, \text{if } k \leq k'_o \\ p^*_{k_{j_i}} - p_{k_{j_i}\sim} \leq 0, \text{if } k > k'_o \end{cases} \tag{21}$$

since $\vec{P}_{j_i} \neq \vec{P}^*_{j_i}$, we have

$$\forall k \in \{1,2,\dots,K_{j_i}\}: \begin{cases} \sum_{k_{j_i}=1}^{k} \left(p^*_{k_{j_i}} - p_{k_{j_i}\sim}\right) > 0, \text{if } k \leq k'_o \\ \sum_{k_{j_i}=k+1}^{K_{j_i}} \left(p^*_{k_{j_i}} - p_{k_{j_i}\sim}\right) < 0, \text{if } k > k'_o \end{cases} \tag{22}$$

i.e.

$$\forall k \in \left\{1,2,\dots,K_{k_{j_i}}\right\}:\ F_{j_i}\left(k, \vec{P}^*_{j_i}\right) > F_{j_i}\left(k, \vec{P}_{j_i\sim}\right) \tag{23}$$

thus, $\vec{P}^*_{j_i}$ is an optimal solution of (18). Meanwhile, according to (23), we see arbitrary feasible solution $\vec{P}_{j_i\sim} \neq \vec{P}^*_{j_i}$ is not an optimal solution, thus $\vec{P}^*_{j_i}$ is unique.■

Now let us build an algorithm to find k_o and compute $\vec{P}^*_{j_i}$. Let us first define a decreasing sequence

$$[a_k]:\ a_k = 1 - \sum_{k_{j_i}<k+1} \tilde{p}_{k_{j_i}+} - \sum_{k+1\leq k_{j_i}\leq K_{j_i}} \tilde{p}_{k_{j_i}-}, k = 0,1,2,3,\dots,K_{j_i} \tag{24}$$

where $a_0 \geq 0, a_{K_{j_i}} \leq 0$ and $\Delta a_k = a_{k+1} - a_k = -(\tilde{p}_{k+} - \tilde{p}_{k-}) \leq 0$. Thus, we can always find a $k_{so} \in \{1$

$, 2,3, \dots, K_{j_i}\}$, such that

$$\tilde{p}_{k_{so}+} - \tilde{p}_{k_{so}-} \geq a_{k_{so}-1} \geq 0, a_{k_{so}} \leq 0 \tag{25}$$

Now let us construct a probability vector $\vec{P}_{j_i}^{**}$ by setting

$$p_{k_{j_i}}^{**} = \begin{cases} \tilde{p}_{k_{j_i}+}, \text{for } k_{j_i} < k_{so} \\ \tilde{p}_{k_{j_i}-}, \text{for } k_{j_i} > k_{so} \\ 1 - \sum_{k_{j_i}=1}^{k_{so}-1} \tilde{p}_{k_{j_i}+} - \sum_{k_{j_i}=k_{so}+1}^{K_{j_i}} \tilde{p}_{k_{j_i}-}, \text{for } k_{j_i} = k_{so} \end{cases} \tag{26}$$

Then we have

Theorem 3.5

$\vec{P}_{j_i}^{**}$ is the *unique optimal solution* of Problem (18), i.e. $\vec{P}_{j_i}^{*} = \vec{P}_{j_i}^{**}$.

Proof

From (25) and (26), we see $p_{k_{so}}^{**} = a_{k_{so}} + \tilde{p}_{k_{so}+}$, thus $p_{k_{so}}^{**} \in \left[\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}\right]$, i.e. $\vec{P}_{j_i}^{**}$ is feasible. As $\vec{P}_{j_i}^{**}$ also satisfies (19), it is the *unique optimal solution* of Problem (18), according to Theorem 3.4. ■

As the existence of an integer $k_{so} \in \{1,2,3, \dots, K_{j_i}\}$, which satisfies (25), is guaranteed, Theorem 3.5 guarantees that Problem (18) always has a unique optimal solution $\vec{P}_{j_i}^{**}$. Now we have

Corollary 3.6

There exists a unique $\boldsymbol{P}_{-}$ in Corollary 3.3, which satisfies

$$\forall j_i: \vec{P}_{j_i-} = \vec{P}_{j_i}^{**} \tag{27}$$

Proof

The definition of $\boldsymbol{P}_{-}$, as given in Corollary 3.3, requires $\boldsymbol{P}_{-}$ to be unique if it exists; meanwhile, with Theorem 3.5 and (23), we have

$$\forall j_i \ \forall \vec{P}_{j_i} \in \wp_{j_i}: F_{j_i}\left(\vec{P}_{j_i}^{**}\right) \preccurlyeq F_{j_i}\left(\vec{P}_{j_i}\right) \tag{28}$$

thus the existence of $\boldsymbol{P}_{-}$ is guaranteed. ■

Let us now analysis the time complexity to construct $\boldsymbol{P}_-$. To construct $\vec{P}_{j_i}^{**}$, we can iteratively compute a_k for $k = 0,1,2,\ldots$ until $a_k \leq 0$, i.e., when we find a k_{so} that satisfies (25). It takes $O(K_{ij})$ to compute a_0, and $O(1)$ to compute a_k from a_{k-1}, since

$$a_k = a_{k-1} + \left(\tilde{p}_{k_{j_i}+} - \tilde{p}_{k_{j_i}-}\right) \tag{29}$$

To construct $\boldsymbol{P}_-$, we need to at most compute $\vec{P}_{j_i}^{**}$ for every feasible j_i. Thus the total time complexity for computing $\boldsymbol{P}_-$ is $O\left(\sum_{i=1}^{N}\sum_{j_i=1}^{n_i} K_{j_i}\right)$. The pseudo-code for computing each $\vec{P}_{j_i-} = \vec{P}_{j_i}^{**}$ is given in Algorithm 1 in Appendix A.

Corollary 3.7

Assume $K \in \mathbb{N}$ is an upper bound of K_{j_i}. Then, there exists a $O(nK)$ time algorithm to compute $\boldsymbol{P}_-$.

For $\sup_{\boldsymbol{P}\in\wp} A(\boldsymbol{x_o}, \boldsymbol{P})$, we have the similar properties (as exhibited from Corollary 3.3 to Corollary 3.7), and thus can similarly compute $\boldsymbol{P}_+$ as shown in the following:

Theorem 3.8

Assume $\boldsymbol{P}_+ = (\vec{P}_{1+}, \vec{P}_{2+}, \ldots, \vec{P}_{i+}, \ldots \vec{P}_{N+})$ is a feasible solution of Problem (5), where $\vec{P}_{i+} = (\vec{P}_{1_i+}, \vec{P}_{2_i+}, \ldots, \vec{P}_{j_i+}, \ldots, \vec{P}_{n_{i_i}+})$, $\vec{P}_{j_i+} = (p_{1_{j_i+}}, p_{2_{j_i+}}, \ldots, p_{k_{j_i+}}, \ldots, p_{K_{j_i+}})$, and it satisfies

$$\forall j_i \ \ \forall \vec{P}_{j_i} \in \wp_{j_i}: F_{j_i}(\vec{P}_{j_i+}) \succcurlyeq F_{j_i}(\vec{P}_{j_i}) \tag{30}$$

Then, we have

a.

$$\sup_{\boldsymbol{P}\in\wp} A(\boldsymbol{x_o}, \boldsymbol{P}) = A(\boldsymbol{x_o}, \boldsymbol{P}_+) \tag{31}$$

b. each $\vec{P}_{j_i+}$ is the unique optimal of the following multi-objective LP

$$\begin{aligned} \text{for } k_{j_i} = 0,1,2,\ldots,K_{j_i}: \quad & \min_{\vec{P}_{j_i\sim}} F_{j_i}(k_{j_i}, \vec{P}_{j_i\sim}) \\ \text{s.t.} \quad & p_{k_{j_i}\sim} \in [\tilde{p}_{k_{j_i}-}, \tilde{p}_{k_{j_i}+}] \\ & \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}\sim} = 1 \end{aligned} \tag{32}$$

c. A feasible solution $\vec{P}_{j_i}^{**}$ is the optimal solution of Problem (32) if

$$\exists k_o \in \{1,2,\dots,K_{j_i}\},\ \text{s.t.} \qquad p_{k_{ij}}^{**} = \begin{cases} \tilde{p}_{k_{j_i}-}, \text{for } k_{j_i} < k_o \\ \tilde{p}_{k_{j_i}+}, \text{for } k_{j_i} > k_o \\ 1 - \sum_{k_{j_i}=1}^{k_o-1} \tilde{p}_{k_{j_i}-} - \sum_{k_o+1}^{K_{j_i}} \tilde{p}_{k_{j_i}+}, \text{for } k_{j_i} = k_o \end{cases} \tag{33}$$

d. The time complexity to compute $\boldsymbol{P}_+$ is $O\left(\sum_{i=1}^{N}\sum_{j_i=1}^{n_i} K_{j_i}\right)$.

The pseudo-code for computing each $\vec{P}_{j_i+}$ is given in Algorithm 2 in Appendix A.

It is essential to note that, the analysis shown in Corollary 3.3 to Corollary 3.7 and Theorem 3.8 implies that the optimal solutions of the inner stage problems in (6) are independent of the system's composition $\boldsymbol{x}$. Thus, we have

Theorem 3.9

For Problem (6), let us define

$$\begin{aligned} A_-(\boldsymbol{x}) &= \sup_{\boldsymbol{P}\in\wp} A(\boldsymbol{x},\boldsymbol{P}) = A(\boldsymbol{x},\boldsymbol{P}_-) \\ A_+(\boldsymbol{x}) &= \inf_{\boldsymbol{P}\in\wp} A(\boldsymbol{x},\boldsymbol{P}) = A(\boldsymbol{x},\boldsymbol{P}_+) \end{aligned} \tag{34}$$

where $\boldsymbol{P}_-$, $\boldsymbol{P}_+$ are constant vectors whose value are already computed from Algorithm 1. and 2. Then, Problem (6) is reducible to the following one-stage optimization problem

$$\begin{aligned} &\max_{\boldsymbol{x}} \boldsymbol{A}(\boldsymbol{x}) = (A_-(\boldsymbol{x}), A_+(\boldsymbol{x})) \\ \text{s.t.} \quad &C(\boldsymbol{x}) = \sum_{i=1}^{N}\sum_{j_i=1}^{n_i} c_{j_i} x_{j_i} \le C_0 \\ &x_{j_i} = 0,1,2,\dots,X_{j_i} \end{aligned} \tag{35}$$

IV. LANDSCAPE OF THE PROBLEM AND OPERATORS DESIGN

In this section, our goal is to analyze the global landscape of MSSPS RAP EU, as presented in (35), such that it could be more efficiently solved in an approximate manner. Since both $A_-(\boldsymbol{x}), A_+(\boldsymbol{x})$ are MSSPS availability functions, we will simply analyze the MSSPS availability function $A(\boldsymbol{x})$ to show the common

properties they share.

A. Landscape of problem (35)

Property 4.1

The evaluation of $A(\boldsymbol{x})$ is #P-hard.

Property 4.2 ***monotonicity***

The optimal Pareto-front of MSSPS-EU RAP can be obtained within the set

$$B_c = \{\boldsymbol{x} | \forall ij_i : C_0 - C(\boldsymbol{x}) < \min\{c_{j_i} | x_{j_i} < X_{j_i}\}\} \tag{36}$$

Property 4.1 is the direct result of Theorem 2.1 from [7]. It implies that "the less samplings the better" for the RAP optimization in the worst case; on the other hand, the true Pareto front could be very hard to obtain when the benchmark goes large. Property 4.2 shows that the optimal Pareto front of the problem is obtainable at the boundary region, since

1. both objective functions are monotonically increasing;

2. the inequality constraints in (35) give rise to a convex set.

Property 4.2 allows us to use repair algorithms to repair infeasible or feasible solutions into the set B_c. As both objective functions are monotonically increasing, the repair algorithm does not have to reduce the value of $A_-(\boldsymbol{x})$ or $A_+(\boldsymbol{x})$ for any feasible solution. In evolutionary computation, repair algorithms [17~19,26~28] are often used to achieve such purpose. Although the repair algorithms [17~19,26~28] often seem time-consuming, like in knapsack problems, the time complexity of them is in general ignorable comparing with #P-hardness.

Property 4.3 ***upper availability bound*** **[35]**

Let $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_s, \dots, \mathcal{P}_{N_\mathcal{P}}\}$ be a partition of the subsystem index set $\mathcal{I} = \{i = 1,2, \dots, N\}$, where each $\mathcal{P}_s$ is a subset $\mathcal{I}$. For given $\boldsymbol{x}$, assume $\mathcal{A}_s = \Pr(\min_{i \in \mathcal{P}_s} G_i \geq D)$; then, $\mathcal{A}_s$ is an upper bound of $A(\boldsymbol{x})$ and we have

$$0 \leq \frac{\min_{s \in \{1,2,\dots,N_\mathcal{P}\}} \mathcal{A}_s - A(\boldsymbol{x})}{1 - A(\boldsymbol{x})} \leq N_\mathcal{P} \tag{37}$$

where $N_\mathcal{P}$ is the number of partitions. Property 4.3 implies that $A(\boldsymbol{x})$ can be approximately optimized through a search among the solutions that are near-optimal to the problem $\max_{\boldsymbol{x} \in B_c} \min_{s \in \{1,2,\dots,N_\mathcal{P}\}} \mathcal{A}_s$, which could be obtained by constructing local search operators/algorithms in which locality is measured by subsystem

availabilities A_i or $\mathcal{A}_s$. For MSSPS RAP EU, a local search operator that exploits the subsystem availability would also be helpful to expand the Pareto front around/from each currently Pareto optimal solution $\boldsymbol{x}$, because what is needed is to find solutions which are dominated/non-dominated by the current Pareto optimal solutions.

Property 4.4 ***approximated concavity in subsystem availability***

a. Assume $\vec{x}_i, \vec{x}_i'$ are two feasible vectors for composing the ith subsystem, where $\Pr G_i(\vec{x}_i \geq d_{K_D}) = \alpha$; then, we have

$$\frac{\bar{A}_i(\vec{x}_i') - \bar{A}_i(\vec{x}_i' + \vec{x}_i)}{\bar{A}_i(\vec{x}_i')} > \alpha \tag{38}$$

b. Assume $\Delta c = \vec{c}_i(\vec{x}_i)$, $A_{im}(c)$ gives the maximum subsystem availability when the total cost of system i is restricted to be less than c. Then, for any $\Delta c'' \geq 2\Delta c$ we have

$$0 < \frac{A_{im}(c + \Delta c'') - A_{im}(c + \Delta c)}{A_{im}(c + \Delta c) - A_{im}(c)} \leq \frac{1 - \alpha}{\alpha} \text{ if } A_{im}(c + \Delta c) > A_{im}(c) \tag{39}$$

c. When $\alpha > 0.5$, $A_{im}(k\Delta c)$ is concave to $k \in \mathbb{N}_+$.

Example 4.5

Assume subsystem i is only composed of x_{j_i} two-state (i.e. $K_{j_i} = 1$) components from the fixed version $j_i = 1$. Then

a. $A_i(x_{j_i})$ is strictly concave to x_{j_i} when $(x_{j_i} + 1)p_{K_{j_i}} \geq d_{K_D} g_{K_{j_i}}$;

b. $A_i(x_{j_i}\Delta c)$ is strictly concave to x_{j_i} when $(x_{j_i} + 1)p_{K_{j_i}} \geq d_{K_D} g_{K_{j_i}}$;

Property 4.4 a. is the direct result of convolution and Property 4.4 b., c. can be directly derived from Property 4.4 a., b., respectively. Property 4.4 implies the approximate concavity on each $A_{im}(c)$ at the global landscape. For example, when $A_{im}(c)$ is discretized as $A_{im}(k\Delta c)$ with $\alpha = 0.9$, we observe that the terms in the first-order differential equation of $A_{im}(k\Delta c)$ decrease in an exponential ratio which is faster than $\frac{1-\alpha}{\alpha}$, as shown in Property 4.4 b.

As MSSPS RAP EU is integer and non-convex, to introduce small perturbations on subsystem composition is often insufficient (as small perturbations on subsystem availability often require large perturbations on subsystem composition, due to the discrete nature of the problem). Property 4.4 implies that we can introduce larger perturbations on subsystem availability, while the chance that it causes

significant reduction on subsystem availability can be reduced through implementing a compensation scheme: if several components are removed, then several components have to be added to the subsystem such that the perturbation of the total subsystem cost remains small. When we search within the set B_c, both too large decrements/increments on subsystem cost should be carefully avoided during the local search, since the subsystem costs are linearly dependent. Details for the local operator design considering the linear dependence are introduced in the next sub-section.

B. Repair and local search operator

In this section, basic designs of repair and local search operators are proposed according to the previous analysis. The pseudo codes of the operators are given in Appendix.

1. The repair operator

In general, heuristics are designed to generate off-springs which are close or similar to the currently-found best solutions under certain distance metrics. According to Property 4.2, it is in general sufficient to search within certain relaxed boundary sets such as

$$\tilde{B}_c = \{\boldsymbol{x} | \|C_0 - C(\boldsymbol{x})\| \leq \varepsilon\} \tag{40}$$

For simplicity, in this paper we consider an evolutionary scheme that only searches within the set B_c, which has appeared in eq. (36). As MSSPS RAP is strongly non-linear and locally non-convex, we expect that the repair algorithm should be stochastic rather than greedy. In principle, we expect to do the repair in two steps: first we remove the installed components according to a uniform probability, until the current solution satisfies all the inequality constraints. Afterwards, the uninstalled components, whose costs are smaller than the absolute value of the difference between the system cost constraint C_0 and the current system cost $C(\boldsymbol{x})$, should then be added to the system. The components should be added uniformly, with respect to their component versions. Such repair procedure terminates when the system is already repaired back into B_c. In this paper, the repair algorithm is implemented through the following steps:

Algorithm 3. The Repair algorithm

Step 1. repair infeasible solutions to the feasible region *F*:

Step 1.1 compute the value $d = C_0 - \boldsymbol{c}^T\boldsymbol{x}$ for the current system;

Step 1.2 if $d < 0$, uniformly select one component from the current system design and remove it, and then go back to Step 1.1; otherwise, go to Step 2.

Step 2. repair feasible solutions to B_C:

Step 2.1 compute the value $d = C_0 - \boldsymbol{c}^T\boldsymbol{x}$ for the current system;

Step 2.2 initialize the set $V = \{j_i: x_{j_i} < X_{j_i}, c_{j_i} \leq d\}$, go to Step 2.4;

Step 2.3 compute the value $d = C_0 - \boldsymbol{c}^T\boldsymbol{x}$ for the current system;

Step 2.4 if V is non-empty, uniformly select a component version j_i from the version index set V and add a component of this version to the current system design, and, then, go back to Step 2.3;

otherwise, the algorithm is terminated.

The time complexity of Algorithm 3 is $\mathrm{O}(n_c \ln n_c)$. In Step 1, we need to compute d in Step 1.1 and to perform the uniform selection in Step 1.2 for at most $\mathrm{O}(n_c)$ times, where n_c is the number of components installed on the system before we run the repair algorithm. In step 2, we need to build the component set $V = \{j_i: x_{j_i} < X_{j_i}, c_{j_i} \leq d\}$ and iteratively perform the uniform selection for at most $\mathrm{O}(n_c)$ times. To build V, it requires $\mathrm{O}(n_c \ln n_c)$ time if we built it as a linked list and sort the index in ascending order, with respect to the component cost c_{j_i}; then, to update it, we only need to remove the version of component if it becomes: 1.$x_{j_i} = X_{j_i}$; 2. $c_{j_i} > d$; at this iteration. It requires $\mathrm{O}(\ln n_c)$ time to update such linked list of V, which is the time complexity for searching the cut point after which we have $c_{j_i} > d$ for all the versions stored in the list.

In this paper, the repair operator will be performed after initialization, crossover and mutation for the modified NSGA-II. The details of the modified NSGA-II will be introduced in the next subsection.

2. Local search operator

Various literature works exist on the development of specific 'local search' techniques for solving MSSPS RAPs. For example, in [10] the mutation operator in a genetic algorithm (GA) is designed as a $(-1, +1)$ local search operator. In [4], the $(-1, +1)$ operator is used for Tabu search (TS). In [3], PSO is mixed with the more sophisticated $(-x, +y)$ local search operators to improve performance. From our analysis of the above works, it results that a simple $(-1, +1)$ could be too "naive" to jump out the local traps; meanwhile, the local search should also guarantee that the system does not move too far away from the set B_c. To address these issues, in this subsection, we propose a local search algorithm which performs specifically designed $(-\alpha, +\beta)$ operators on selected subsystems. Considering Properties 4.3 and 4.4, we also expect that the cost of the subsystem should be perturbed "slightly" with respect to the unit costs of the installed subsystem components, during each $(-\alpha, +\beta)$ operation.

Also, in our evolutionary algorithm design we will make sure that the system is always in B_c before performing the local search, which can be guaranteed by implementing the repair algorithm. The local search is, then, designed to satisfy the following rules:

a. after the $(-\alpha, +\beta)$ operator is applied to one subsystem, the current system cost should always be close to the system cost constraint, either slightly above or below.

b. after the $(-\alpha, +\beta)$ operator is applied to the last perturbed subsystem, the current system cost should be just below the system cost constraint, so that Step.1 of the repair algorithm could be avoided if the current solution needs to be repaired.

3. the selected subsystems should be perturbed by the $(-\alpha, +\beta)$ operator in a random order, since the selection procedure of the last perturbed subsystem should be independent of the subsystem index.

Now, let us give a more precise description of the $(-\alpha, +\beta)$ operator performed on each subsystem: The operator first selects a component version $j_{i,\alpha}$ from the installed component versions with uniform probability, and, then, removes α components from this version, where the value of α is sampled from uniform distribution on the discrete set $\{1,2,\dots,x_{j_{i,\alpha}}\}$ and $x_{j_{i,\alpha}}$ is the number of currently installed components in version $j_{i,\alpha}$. The operator, then, selects a component version $j_{i,\beta}$ from the component versions satisfying $x_{j_i} \leq X_{j_i}$, with uniform probability. To decide the value of β, the algorithm first sets $x_{j_{i,\beta}}$ to be the maximum value that allows the system to be feasible; then, if there are still subsystems to be perturbed and $x_{j_{i,\beta}} < X_{j_{i,\beta}}$, then it will set $x_{j_{i,\beta}} = x_{j_{i,\beta}} + 1$ with 50% chance, such that the current system cost could just violate the system cost constraint; otherwise, it will not set $x_{j_{i,\beta}} = x_{j_{i,\beta}} + 1$, so that the system stays feasible and close to the set B_c (with respect to the component cost $c_{j_{i,\beta}}$) after the local search. Algorithm 4 below shows the procedure of the local search operator.

Algorithm 4. Local Search

Step 1. take the vector S_v, in which the index of selected subsystems are stored in random order.

Step 2. for $i = 1{:}\,\text{length}(S_v)$

Step 2.1 uniformly select a component version j_α from the version index set $\{j{:}\, x_{j_i} > 0\}$;

Step 2.2 remove α components of version j_α according to the distribution $U\left(1, x_{j_{i,\alpha}}\right)$;

Step 2.3 uniformly select a component version j_β from the index set $\{j: x_{j_i} < X_0\}$;

Step 2.4 add β components of version j_β according to the equation

$$\beta = \max\left\{X, X_0 - x_{j_{i,\beta}}\right\})$$

where

$$X \sim U\left\{\left\lfloor\frac{C_0 - C(\vec{x})}{c_{j_{i,\beta}}}\right\rfloor, \left\lceil\frac{C_0 - C(\vec{x})}{c_{j_{i,\beta}}}\right\rceil\right\} \text{ if } i \neq \text{length}(S_v)$$

$$X = \left\lfloor\frac{C_0 - C(\vec{x})}{c_{j_{i,\beta}}}\right\rfloor \text{ if } i = \text{length}(S_v)$$

and $C(\vec{x})$ is the current cost of the system before the β components are added;

Step 2.5 update $C(\vec{x})$; if $C_0 - C(\vec{x}) > \min\{c_{j_i}: x_{j_i} < X_{j_i}\}$, go to Step 2.3; otherwise determinate.

In this work, the local search operator is performed during the mutation step of our modified NSGA-II.

V. THE MODIFIED NSGA-II

In this section, we present our proposed modified MOEA (namely co-sp-NSGA-II), which integrates NSGA-II with the previously proposed repair and local search operators. The general scheme of the whole Algorithm is presented as below. The details of each step are, then, presented in the Subsections A to E.

Algorithm 5. co-sp-NSGA-II

Step 1. Initialization: randomly generate initial population and then repair if necessary;

Step 2. Parents selection: binary tournament selection of the parent population;

Step 3. Crossover: select chromosomes to perform uniform crossover on subsystems and, then, repair if necessary;

Step 4. Mutation: perform the local search operator to distribute slight disturbances on subsystem costs, by modifying its composition;

Step 5. Fitness evaluation: compute the fitness values for the children population;

Step 6. Elitism Selection: combine the parents and children populations; perform fast non-dominated sorting and local search operator to select the next population;

Step 7. Termination: the algorithm terminates if a maximum number of generations is reached; otherwise go to Step 2.

A. Encoding

The MSSPS design is represented in an integer string, i.e., $\boldsymbol{x} = (x_{1_i}, x_{2_i}, \dots, x_{j_i}, \dots, x_{n_{i_N}})$, where x_{j_i} is the number of components of version j_i installed. The substring $\vec{x}_i = (x_{1_i}, x_{2_i}, \dots, x_{j_i}, \dots, x_{n_{i_i}})$ represents one subsystem i.

B. Initialization

In initialization, the initial population is designed to be generated within the set B_C. A randomized integer number generator with a repair algorithm is used to generate the initial population.

Algorithm 6. Initialization

Step 1. Assign a random integer value to each x_{j_i} with

$$x_{j_i} \sim U(0, x_o)$$

where $U(0, x_o)$ is a uniform distribution on the set $\{0, 1, \ldots, x_o\}$.

Step 2. if $\boldsymbol{x} \notin B_c$, then it is repaired by the algorithm presented in the following section.

C. Crossover

To avoid introducing unnecessary significant perturbation to the subsystem costs, we set the crossover point only between subsystems. Since the MSSPS availability is independent of the order of subsystem index, we perform uniform crossover to exchange subsystems between the parent individuals. After the uniform crossover is done, we perform the repair algorithm to retain the MSSPS within the set B_c.

Algorithm 7 Crossover

Step 1. select the chromosomes in the current population to perform crossover according to the crossover rate p_c;

Step 2. decide the number of subsystems, i.e. c_{num}, that will perform uniform crossover for each paired chromosomes;

Step 3. for each of the pair, uniformly select the subsystems to perform the crossover according to c_{num}.

Step 4. perform repair operator on each newly generated individual.

The pseudo codes of the crossover operator is given in Appendix.

D. The mutation operator

In mutation, we select subsystems to perform local search with uniform probability. After local search, the repair operator is performed to repair the system back to the set B_c. The pseudo codes of the mutation operator are given in Appendix.

Algorithm 8. Mutation

for each individual:

Step 1. select subsystems to perform mutation, with a uniform probability p_m;

Step 2. check if more than 1 subsystem is selected; if not, randomly select one subsystem to perform the local search.

Step 3. perform the local search operator on the selected subsystems;

Step 4. perform the repair operator for the individual.

E. Illustration of the modified crossover and mutation operators

The procedure of the crossover and mutation operator is illustrated in Fig.2. Note that subsystems are randomly selected to perform local search during mutation. Meanwhile, $(-\alpha, +\beta)$ operators are performed on the selected subsystems in random order during the local search.

On the top left it illustrates the procedure of crossover: here the 2nd and 4th subsystem are selected to perform the uniform crossover with probability p_c. In mutation, subsystems are randomly selected to perform local search.

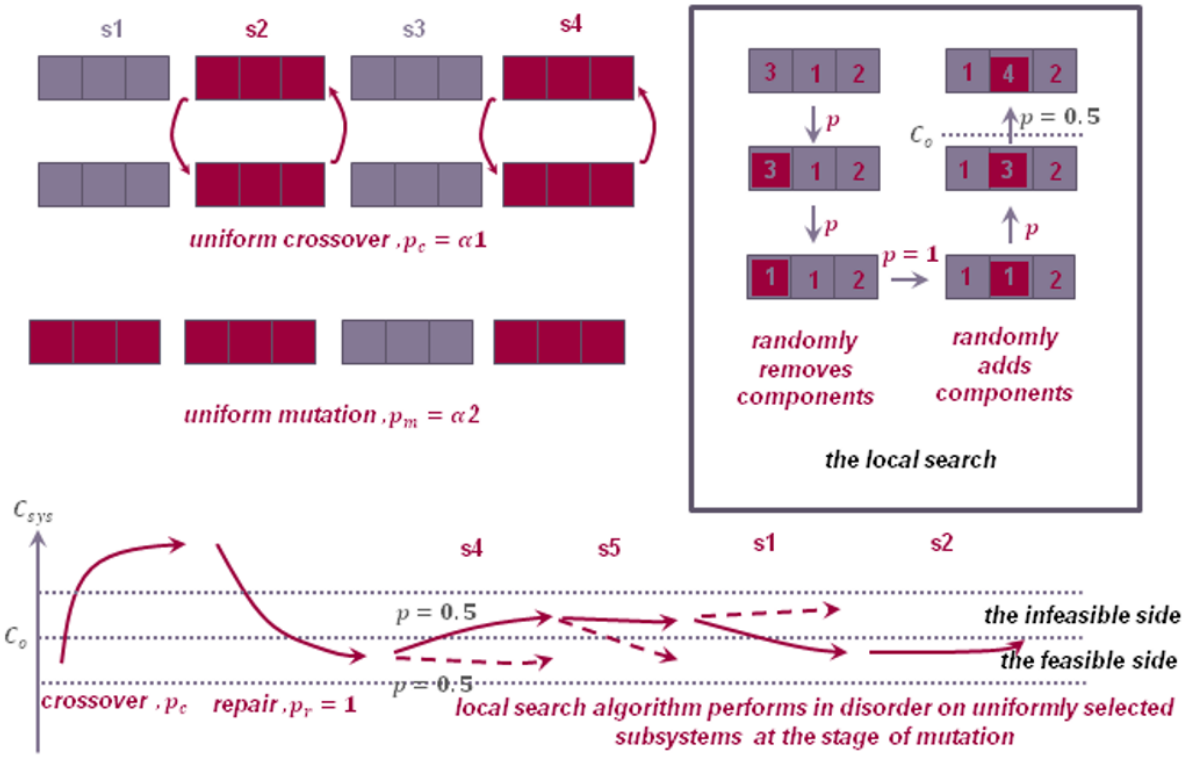


Fig 2. The crossover and mutation operators

On the top right, it illustrates the $(-\alpha, +\beta)$ operation. Here the $(-\alpha, +\beta)$ operator is performed on a subsystem with 3 candidate component versions, where the numbers 3, 1, 2 in the grey chromosome given the number of components at each version installed on the current system, respectively. According to the top right of Figure. 2, the $(-\alpha, +\beta)$ operator first randomly sets $j_{i,\alpha} = 1$ and $\alpha = 2$, such that two components in the 1st version is removed from the system. Afterwards, it randomly selects sets $j_{i,\beta} = 2$ and found that it should also set $\beta = 2$, in order to make sure that the total system cost is just lower than C_0, after the $(-\alpha, +\beta)$ operation. If the subsystem is not the last one that performs the $(-\alpha, +\beta)$ operator

during the local search, the $(-\alpha, +\beta)$ operator will set $\beta = \beta + 1$ with a probability of 0.5. Such process allows the local search to sufficiently exploit around the set B_c, including the infeasible side of the cost constraint $C(\boldsymbol{x}) \leq C_0$.

At the bottom of Fig. 2, it illustrates how the system cost of a child chromosome varies during crossover and mutation. The procedure begins with a uniform crossover, during which the system cost is unconstrained. Then, the system is repaired into the set B_C by using the repair operator. After crossover, it moves to the procedure of mutation. During the local search, the system's cost oscillate around the maximum system cost C_0, while the $(-\alpha, +\beta)$ operator is performed on the selected subsystems one after another. After the local search is done, the repair operator is used again in order to guarantee the system will fall back into the set B_C.

VI. Computational Experiments

In this section, a number of benchmarks are generated to test the proposed optimization method. The benchmarks are either newly created or extended from the existing ones that are designed for single-objective MSSPS RAP. To solve the inner stage problem of (6), we use algorithms 1& 2 to obtain the values $p_{k_{j_i-}}$ and $p_{k_{j_i+}}$ for each component version. As the problem is then reduced to (35), we are allowed to compare the proposed co-SP-NSGA-II with the standard NSGA-II on each benchmark, to test their performances. In the tests, we also combine the near-Pareto optimal solutions to analyze their clustering characteristics on subsystem costs and performances.

A. Benchmark Generation

1) Creation of a novel benchmark

In this subsection, we stochastically create a novel MSSPS RAP EU benchmark. The benchmark consists of 15 subsystems, where each subsystem has 10 candidate component version, and each component version have three states.

To simulate what might commonly appear in practice, the following properties are considered:

1. The system and candidate components should be highly reliable;
2. The uncertainty level should be non-significant.

To guarantee that the benchmark is "difficult" to solve, we also let the candidates within the same subsystem have mutually-similar expected performance/cost ratios while the average component cost of a subsystem

differs from one to another. In this benchmark, we set the candidate components satisfying the following rules:

$$\begin{gathered} 0.8 \le \tilde{g}_{K_{j_i}-}, 0.9 \le \tilde{g}_{K_{j_i}+} \\ 0.85 \le \sum \tilde{g}_{k_{j_i}-} \le 1 \le \sum \tilde{g}_{k_{j_i}+} \le 1.15 \\ \frac{\mathrm{E}\tilde{g}_{j_i-} + \mathrm{E}\tilde{g}_{j_i+}}{c_{j_i}} = \left(1 + \varepsilon_{j_i}\right)\delta_i \end{gathered} \tag{41}$$

where

$$\begin{gathered} \delta_i \sim U(0.2,1) \\ \varepsilon_{j_i} \sim U(0,0.1) \end{gathered} \tag{42}$$

δ_i is the bias for the expected performance/cost ratio at each subsystem i, and ε_{j_i} is the noise introduced for each version j_i. The created benchmark, namely slz-15 is given in Appendix B, where Table 3 gives the probability distribution of system demand and Table 4 gives the uncertain candidate components.

2) Benchmark extension

In literature, benchmark systems lev-4 [10], [3] and lev-5 [30] have been frequently tested for MSSPS RAP problems. For examples, lev-5 has been tested for the availability maximization in [9] and lev-4 has only been tested for cost minimization. In this work, we extend them to include epistemic uncertainties by adding the interval constraints in (3) to the state probabilities. Since all components are binary-stated, the linear dependence between the two states allows us to extend the benchmarks in the following way:

$$\begin{gathered} \tilde{p}_{2_{j_i}+} = p_{2_{j_i}o} \\ \tilde{p}_{1_{j_i}+} = p_{1_{j_i}o} + \Omega_{j_i}, \Omega_{j_i} \sim \mathrm{U}(0,0.2) \\ \tilde{p}_{2_{j_i}-} = p_{2_{j_i}o} - \Omega_{j_i} \\ \tilde{p}_{1_{j_i}-} = p_{1_{j_i}o} \end{gathered} \tag{43}$$

where $p_{2_{j_i}o}$ is the probability that the component functions, in the previous benchmark lev-4 or lev-5, and $p_{1_{j_i}o} = 1 - p_{2_{j_i}o}$. is the failure probability of the component in version j_i. Then, we immediately have

$$p_{2_{j_i}+} = \tilde{p}_{2_{j_i}+}; p_{1_{j_i}+} = 1 - p_{2_{j_i}+} \tag{44}$$

$$p_{1_{ij}-} = \tilde{p}_{1_{j_i}+}; p_{2_{j_i}-} = 1 - \tilde{p}_{1_{j_i}+}$$

in respect to Algorithm 1 and 2. Then, (44) allows us to directly solve the MSSPS EU RAP in the formulation of Problem (35).

Now we explain how the optimal solutions of the single-objective MSSPS RAP benchmarks reported in literature can be used in comparison with our multi-objective optimization results. Let $\boldsymbol{x_{o,min4}}$ and $\boldsymbol{x_{o,max5}}$ denote the optimal solutions to the benchmarks lev-4 and lev-5 obtained in [10], [3] and [9], respectively. For lev-5, we keep the constraints in (35) the same as the ones used in [9] and name our new benchmark as lev-5a; for lev-4, since the problem has only been tested in cost minimization form, we set the maximum system cost C_o in (35) to be $C_o = C(\boldsymbol{x_{o,min4}})$ and name our new benchmark as lev-4a. Now, it is obvious that $A(\boldsymbol{x_{o,min4}})$ is the current known best solution for the objective function $\max A_+(\boldsymbol{x})$ on lev-4a and so is $A(\boldsymbol{x_{o,max5}})$ for lev-5a. In this paper, we use $A(\boldsymbol{x_{o,min4}})$ and $A(\boldsymbol{x_{o,max5}})$ to validate the Pareto optimality of the results by the proposed algorithm on the new benchmarks. The uncertain components in the extended benchmarks are given in Table 5, Appendix B.

B. Benchmark Test

We test the proposed algorithm in comparison with standard binary-coded NSGA-II on the three benchmarks. The algorithm parameters settings are presented in in Table 1.

To measure the performance of the multi-objective optimization algorithms, the following metrics are used:

1. The two extreme values of the Pareto front, i.e., A_{-opt}, A_{+opt};

2. The number of solutions located on the Pareto front across all tests $\|pf_{all}\|$.

The former is used to measure the convergence property of each algorithm and the latter is used to identify the best parameter setting and the best single run of the tested algorithms.

When use the above metrics, it is possible that we have multiple best parameters and best runs. In this paper, we will illustrate only one of them in Table 2. test results.

	algorithm	p_c	Δp_c	p_m	Δp_m	size	max gen.	repetition	C_o
slz-15	Co-SP-NSGA-II	[0.1, 1]	0.1	[0.1, 1]	0.1	20	5000	5	27
	NSGA-II	[0.1, 1]	0.1	[0.0025 0.025]	0.0025	20	5000	5	27
lev-4a	Co-SP-NSGA-II	[0.1, 1]	0.1	[0.1, 1]	0.1	20	1000	5	8.18
	NSGA-II	[0.1, 1]	0.1	[0.01, 0.2]	0.01	20	1000	5	8.18
lev-5a	Co-SP-NSGA-II	[0.1, 1]	0.1	[0.1, 1]	0.1	20	2000	5	16

	NSGA-II	[0.1, 1]	0.1	[0.01, 0.2]	0.01	20	5000	5	16

Table 1 parameter settings of all algorithms on each benchmark

	algorithm	A_{-opt}	A_{+opt}	$\|pf_{all}\|$	para	$\|pf_{par\_all}\|$	A_{-par}	A_{+par}	$\|pf_{bst\_single}\|$	gen
mux-15	Co-SP-NSGA-II	0.9219	0.9808	16	(0.7,0.1)	8	0.9211	0.9806	8	(66,71)
	NSGA-II	0.8318	0.9319	2	(0.7,0.005)	1	0.8318	0.9262	1	/
lev-5a	Co-SP-NSGA-II	0.9850	0.9970	10	(0.6,0.2)	10	0.9326	0.9915	10	(30,30)
	NSGA-II	0.9779	0.9859	16	(0.8,0.05)	7	0.9702	0.9859	6	/
lev-4a	Co-SP-NSGA-II	0.9326	0.9915	9	(0.6,0.4)	9	0.9326	0.9915	9	(14,12)
	NSGA-II	0.9326	0.9862	7	(0.8,0.2)	4	0.9326	0.9837	4	/

Table 2 test results

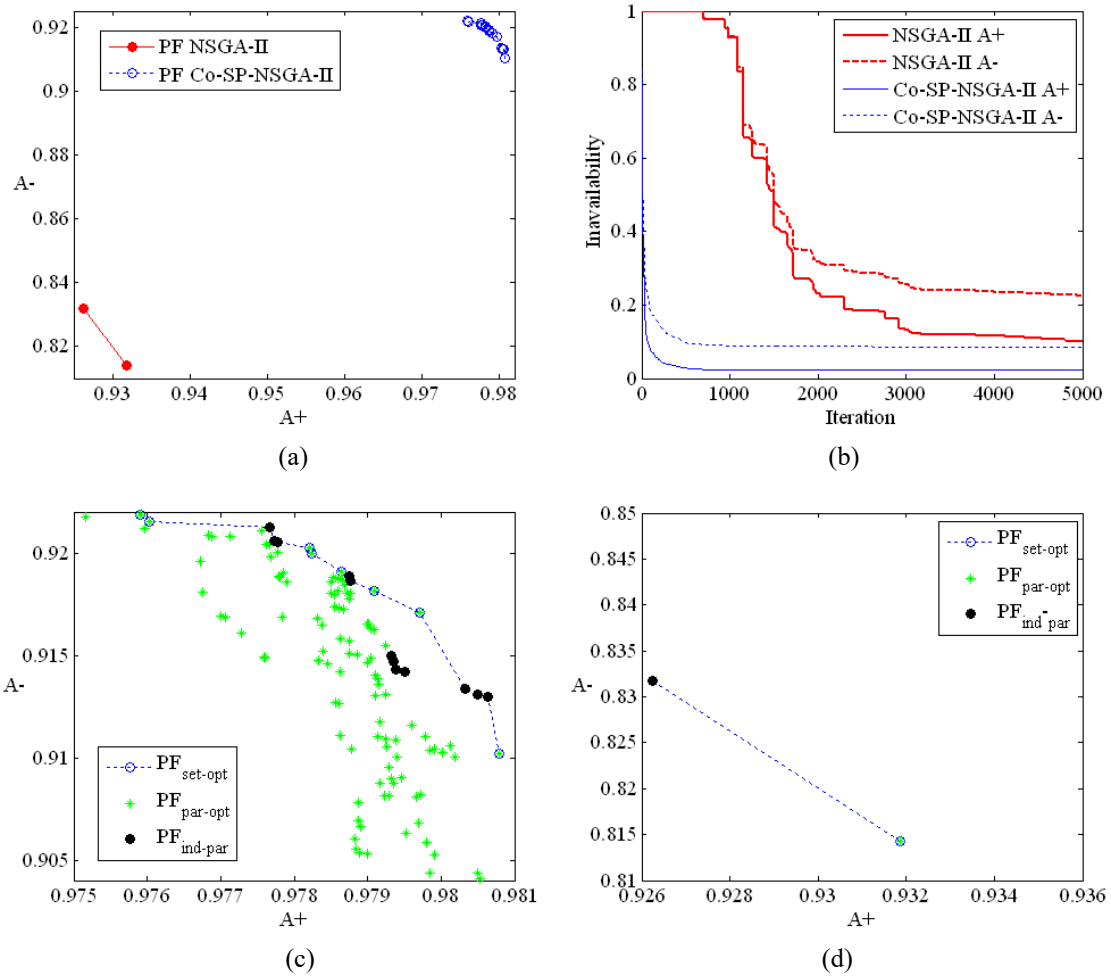


Fig. 3 test results in mux-15

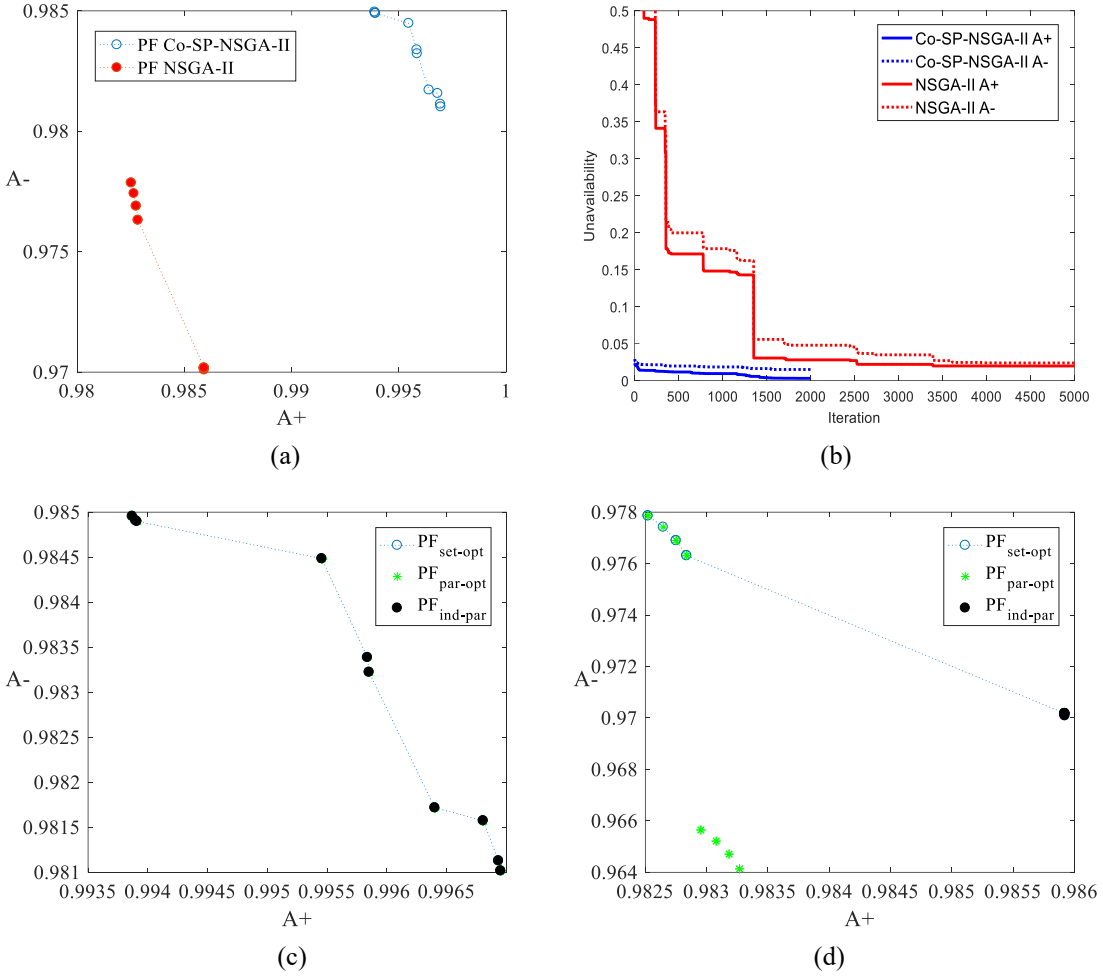
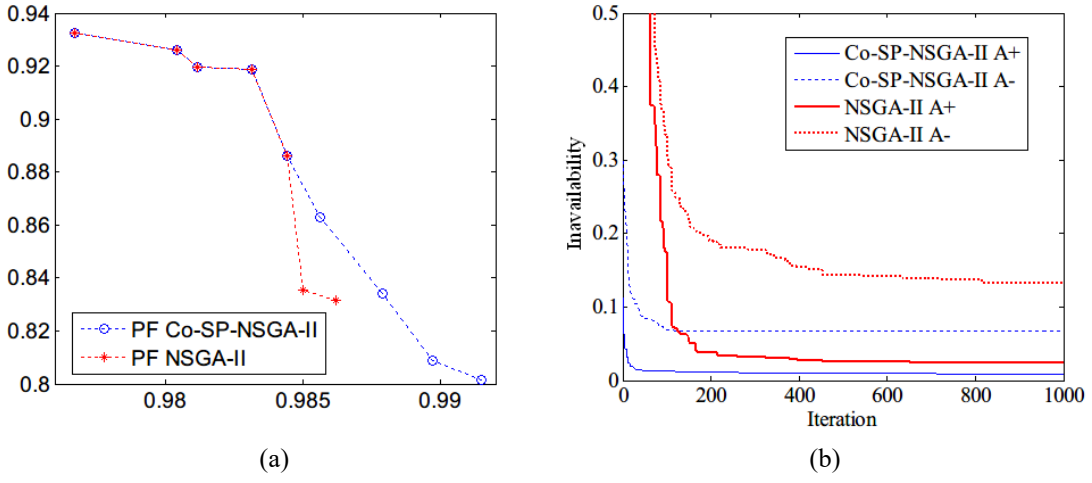
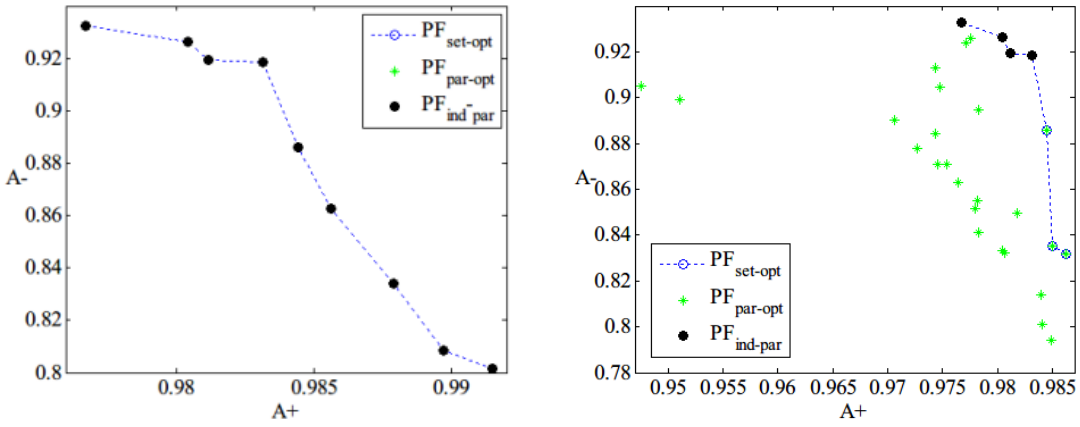


Fig. 4 test results in lev-5a





(c) (d)

Fig. 5 test results in lev-4a

Firstly, for each of the two competing algorithms, we have the Pareto front of all the feasible solutions obtained during the experiment. In Fig. 3a, Fig. 4a and Fig. 5a, we compare the above Pareto fronts obtained by the two algorithms. To look into the details, In Fig. 3c, Fig. 4c and Fig. 5c, we give the details of Pareto front obtained by Co-SP-NSGA-II at different parameters and the best run. In these figures, $PF_{set\text{-}opt}$ gives the Pareto front of all the feasible solutions, $PF_{par\text{-}opt}$ gives the combination of the solutions from the Pareto fronts obtained at all the best parameters, and $PF_{ind\text{-}par}$ gives the Pareto front obtained at one of the best single-runs. We also give the details of Pareto front obtained by NSGA-II, in Fig. 3d, Fig. 4d and Fig. 5d.

To compare the results, in Table 2 we also exhibit the following results for each algorithm: the extreme values for the Pareto front of all the solutions found during the whole experiment A_{-opt}, A_{+opt}; the extreme values for the Pareto front of all the solutions found at the best cross over/mutation parameter, A_{-par} and A_{+par}; the number of solutions for the Pareto front containing all the solutions found during the whole experiment $\|\mathrm{pf}_{all}\|$; the number of solutions on each Pareto front found with the best crossover/mutation parameters $\|\mathrm{pf}_{par\_all}\|$; the size of the Pareto front, at the best single run under the best parameters $\|\mathrm{pf}_{bst\_single}\|$; and one pair of the best parameters, i.e. the best crossover rate p_c and the best mutation rate p_m. These results clearly show that the proposed algorithm outperforms NSGA-II, in both Pareto-dominance and the diversity of Pareto front.

For convergence analysis, unavailability - iteration plots, of the extreme values averaged of 5 runs are drawn for the two algorithms at their best parameter settings in Fig. 3b, Fig. 4b, Fig. 5b, respectively. It is clear that:

1. The proposed algorithm has better final performance;

2. The proposed algorithm outperforms NSGA-II's final results at very early iterations.

To quantify such observation, in Table 2 we also record the following metric

$$gen = (gen1, gen2)$$

where

$$\forall gen' \geq gen1: l1(gen') \leq l3(maxgen)$$

$$\forall gen' \geq gen2: l2(gen') \leq l4(maxgen)$$

$l1(gen')$ and $l2(gen')$ are the average values of $\bar{A}_+$ and $\bar{A}_-$ obtained by our proposed algorithm at its best crossover and mutation rates at the iteration $gen1'$ and $gen2'$, respectively, and $l3(maxgen)$ and $l4(maxgen)$ are those average values of $\bar{A}_+$ and $\bar{A}_-$ obtained by NSGA-II at the maximum generation. With such metric, we observe that the proposed algorithm takes less than 2% of the total generations (or even much fewer) to achieve better performances than NSGA-II.

From the above analysis, we can confirm the significant advantages of the proposed algorithm in terms of Pareto-dominance, diversity and speed of convergence.

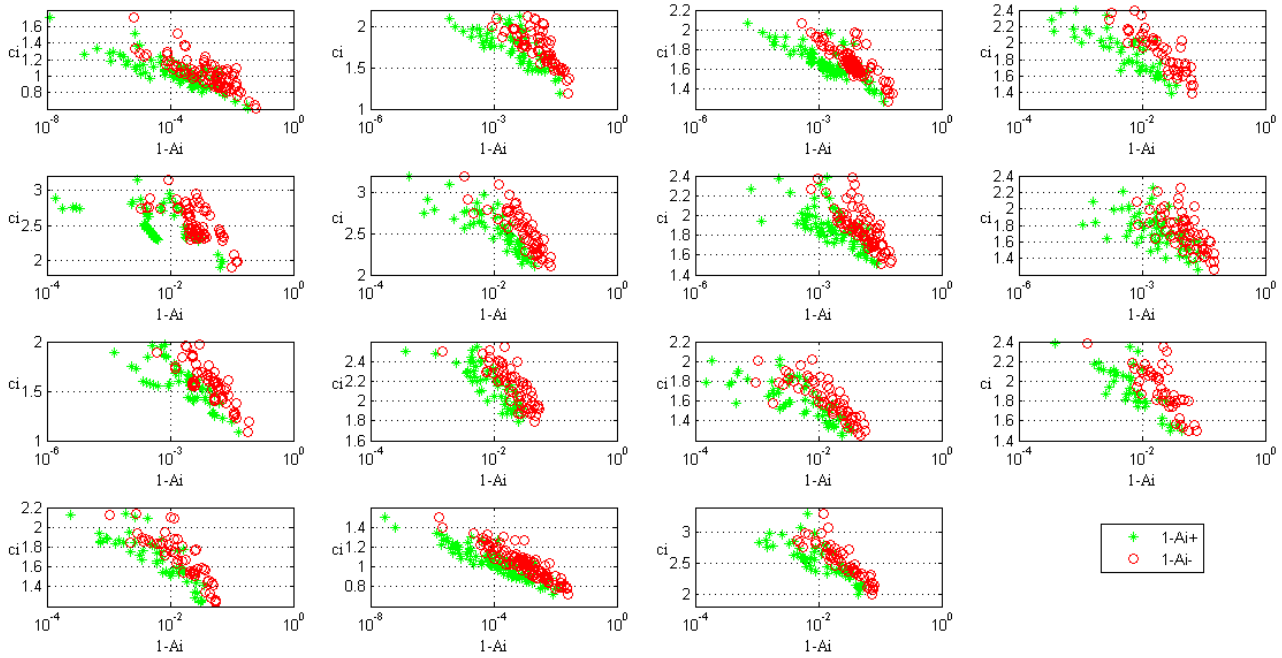
C. Comparing with previously published results in single-objective MSSPS RAP

In the above we have compared the testing results of the proposed algorithm with NSGA-II. Now let us compare the results of the proposed algorithm with the best results published in the previous literature. The problem type (cost minimization/availability maximization), the parameter settings, and the optimal solutions are presented in Table 3. (Note that the total number of evaluations required for different algorithms cannot be directly used for comparing the efficiency of the compared algorithms, since they were running in different environments, although the MO problem is in general the more difficult one.) The following remarks are obtained: for lev-5a, the proposed algorithm finds solutions that dominate the best one in [9], with fewer evaluations; for lev-4a, the proposed algorithm finds the same optimal solution in [3,4] with very high probability and costed only a few evaluations.

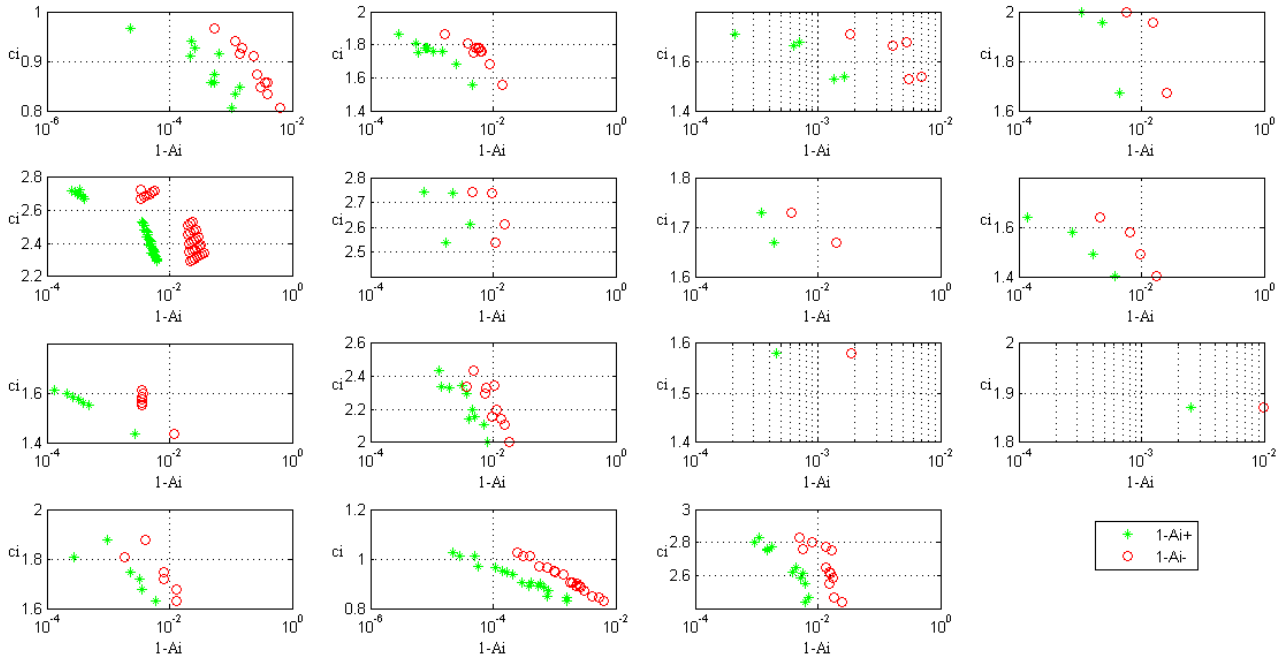
D. Subsystem performance distribution

The above experiments generate near-optimal solutions, which allow us to test the correctness of the analysis in Chapter IV Section A. Note that in the proposed algorithm, each individual is generated considering limited intensity of cost perturbation on each subsystem, whereas NSGA-II treats each component equally without taking into account subsystems' cost-performance relationships. Figs. 5 to 7

depict the scatter plots of availability and cost of each subsystem of the near-Pareto optimal solutions in slz-15, lev-5a, and lev-4a, respectively. It is clear that the proposed algorithm achieves relatively 'tighter' bounds for the cost values and certain concavity between the subsystem availability and costs, while certain level of tightness in bounds and concavities can also be obtained from the near-optimal solutions of standard NSGA-II.

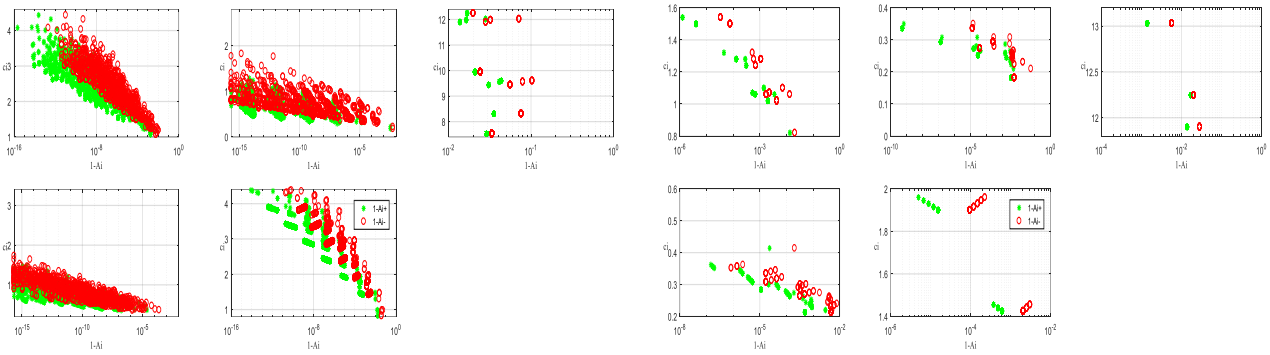


(a) NSGA-II



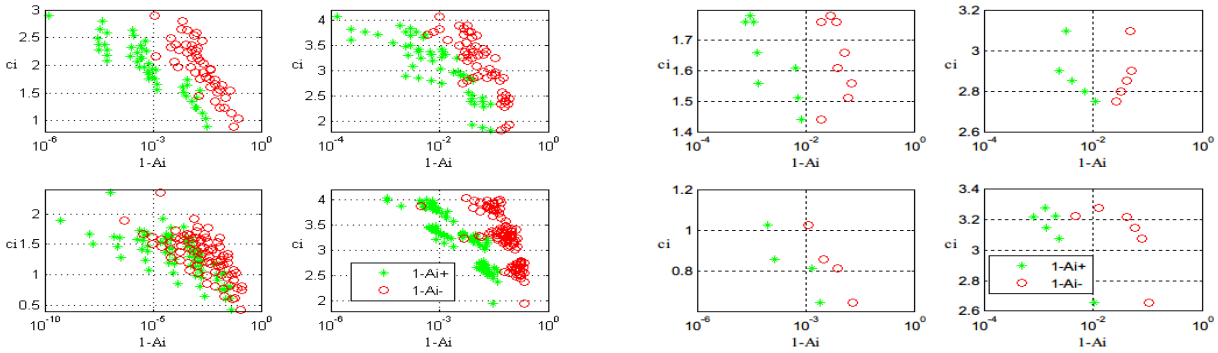
(b) Co-SP-NSGA-II

Fig. 6 subsystem performance distributions on slz-15



(a) NSGA-II (b) Co-SP-NSGA-II

Fig. 7 subsystem performance distributions on lev-5a



(a) NSGA-II (b) Co-SP-NSGA-II

Fig. 8 subsystem performance distributions on lev-4a

VII. CONCLUSIONS

In this work, we systemically analyze MSSPS RAP under the interval-bounded epistemic uncertainties. A linear-time algorithm is proposed to compute the exact state distributions for the multi-state components at which the uncertain system availability will be at its infimum/supremum under uncertain environment. To the authors' knowledge, it is the first algorithm that can achieve such goal. With such algorithm, the proposed MSSPS-EU RAP is reduced to a one-stage multi-objective optimization problem. Then, we analyze its landscape and propose effective repair and local search operators for its solution. A modified NSGA-II incorporating with the proposed operators in then proposed and compare with the standard NSGA-II on multiple benchmarks. The promising experiment results demonstrate that the proposed algorithm significantly outperforms NSGA-II, which finds solutions better than or non-dominated to the published ones at high time-efficiency.

As future work, it is interesting to further improve the design of evolutionary algorithms for MSSPS Rap

EU. For instance, in this paper, each component version is selected under uniform probability in local search. However, among the top Pareto-ranked individuals found by the evolutionary algorithm, some particular component versions could have a higher chance of appearance than the others. To increase the probability of selecting such component versions, estimation of distribution algorithms may be considered.

APPENDIX

A. The Pseudo Codes

Algorithm 1.

for each i, j_i
  for $k_{j_i} = K_{j_i}$ to 0
    $p_{k_{j_i}} = \tilde{p}_{k_{j_i}-}$;
  endfor
  for $k_{j_i} = 0$ to K_{j_i}
    $p_{k_{j_i}} = \tilde{p}_{k_{j_i}+}$;
    $s = \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}}$;
    if $s > 1$
      $p_{k_{j_i}} = \tilde{p}_{k_{j_i}+} - s + 1$;
      break;
    endif
  endfor
endfor

Algorithm 2.

for each i, j_i
  for $k_{j_i} = K_{j_i}$ to 0
    $p_{k_{j_i}} = \tilde{p}_{k_{j_i}-}$;
  endfor
  for $k_{j_i} = K_{j_i}$ to 0
    $p_{k_{j_i}} = \tilde{p}_{k_{j_i}+}$;
    $s = \sum_{k_{j_i}=1}^{K_{j_i}} p_{k_{j_i}}$;
    if $s > 1$
      $p_{k_{j_i}} = \tilde{p}_{k_{j_i}+} - s + 1$;
      break;
    endif
  endfor
endfor

The Crossover Operator

for $i_{indi} = 1$to $\lfloor n/2 \rfloor$ % for each subsystem that will crossover
  *if $\text{rand}(1) < p_c$
  **$r = \text{randperm}(N)$;
  ***$c_{num} = \text{randi}(1, N)$;
End for
for $i_r = 1$: c_{num}
    ****exchange$(i_{indi}, r(i_r))$;
      run Repair Operator; % repair the pair chromosomes to B_c
End for

* $\text{rand}(n)$ returns a n-dimension vector, the value of each element in the returned vector is uniformly sampled from the interval $[0,1]$;

** $\text{randperm}(N)$ returns a vector of one random permutation of the elements in the set $\{1,2,\dots,N\}$;

*** $\text{randi}(1, N)$ returns an integer value from the set $\{1,2,\dots,N\}$ with uniform probability;

**** $\text{exchange}(i_{indi}, r(i_r))$ exchanges the $r(i_r)$ th subsystems of the paired chromosomes whose index are

The Mutation Operator

for $i_{indi} = 1$ to n

$u = \text{rand}(n) < p_c$;

run Local Search Operator; % perform local search

run Repair Operator; % perform the repair algorithm

end for

The Repair Operator

%% repair to F

$d = C_0 - \boldsymbol{c}^T\boldsymbol{x}$;

$\vec{x}_i = (x_{1_i}, x_{2_i}, \dots, x_{j_i}, \dots, x_{n_{i_i}})$;

while $d < 0$

$*j_{o,i} = \text{rand1}(\vec{x})$;

$x_{j_{o,i}} = x_{j_{o,i}} - 1$;

$d = d + c_{j_{o,i}}$;

end while

%% repair to B_c

$V_1 = \{j_i : x_{j_i} < X_0, c_{j_i} \le d\}$;

while $d \ge \min(\{c_{j_i} : x_{j_i} < X_0\})$

$**(j_{o,i}) = \text{rand2}(V_1)$;

$x_{j_{o,i}} = x_{j_{o,i}} + 1$;

$d = d - c_{j_{o,i}}$;

$V_1 = \{j_i : x_{j_i} < X_0, c_{j_i} \le d\}$;

end while

* $\text{rand1}(\vec{x}, \vec{p})$ returns the index (i, j) of a component in $\vec{x}$, where the component is uniformly selected from all the installed components.

**$\text{rand2}(V)$ returns the index (i, j) by taking a sample from V with uniform probability.

The Local Search Operator

$S_v = \text{randperm}(u)$; % construct S_v

$d = C_0 - \boldsymbol{c}^T\boldsymbol{x}$;

for $i_v = 1{:}|S_v|$

$i = S_v(i_v)$;

$j_\alpha = \text{randi}(1, n_i)$;

$\alpha = \text{randi}(0, \alpha)$;

$d = d + c_{j_{i,\alpha}} \cdot \alpha$;

$[\beta, d] = \text{add}(d, i)$;

end for

%% $\text{add}(d, i_s)$

function $[x_{j_{i,\beta}}, d] = \text{add}(d, i_s)$

$j_\beta = \text{randi}(1, n_{i_s})$;

if $i_s \ne |S_v|$

$$\beta = \max\left\{\text{randi}\left(\left\lfloor\frac{d}{c_{j_{i,\beta}}}\right\rfloor, \left\lceil\frac{d}{c_{j_{i,\beta}}}\right\rceil\right), X_0\right\};$$

else

$$\beta = \max\left\{\left\lfloor\frac{d}{c_{j_{i,\beta}}}\right\rfloor, X_0\right\};$$

$d = d - c_{j_{i,\beta}} \cdot \beta$;

end if

end function

B. The benchmarks

k_D	1	2	3	4
g_{k_D}	20	50	80	100
p_{k_D}	0.1	0.4	0.3	0.2

Table 3 Probability distributions of system demand in slz-15

		$\tilde{p}_{2_{ij}+}$	$\tilde{p}_{1_{ij}+}$	$\tilde{p}_{0_{ij}+}$	$\tilde{p}_{2_{ij}-}$	$\tilde{p}_{1_{ij}-}$	$\tilde{p}_{0_{ij}-}$	$p_{2_{ij}+}$	$p_{1_{ij}+}$	$p_{0_{ij}+}$	$p_{2_{ij}-}$	$p_{1_{ij}-}$	$p_{0_{ij}-}$	$g_{2_{ij}}$	$g_{1_{ij}}$	c_{ij}
	1	0.946	0.056	0.054	0.896	0.019	0.014	0.946	0.04	0.014	0.896	0.05	0.054	69	41	0.426
$s1$	2	0.923	0.078	0.082	0.895	0.057	0.02	0.923	0.057	0.02	0.895	0.057	0.048	71	36	0.437
	3	0.981	0.083	0.025	0.939	0.035	0.005	0.96	0.035	0.005	0.939	0.036	0.025	34	16	0.214

	4	0.948	0.086	0.028	0.932	0.046	0.006	0.948	0.046	0.006	0.932	0.046	0.022	82	37	0.485
	5	0.928	0.063	0.05	0.887	0.033	0.009	0.928	0.063	0.009	0.887	0.063	0.05	57	30	0.327
	6	0.961	0.038	0.031	0.92	0.033	0.007	0.96	0.033	0.007	0.931	0.038	0.031	27	13	0.161
	7	0.927	0.113	0.061	0.868	0.037	0.023	0.927	0.05	0.023	0.868	0.071	0.061	32	14	0.194
	8	0.947	0.046	0.055	0.94	0.039	0.014	0.947	0.039	0.014	0.94	0.039	0.021	39	23	0.229
	9	0.951	0.051	0.049	0.881	0.045	0.012	0.943	0.045	0.012	0.9	0.051	0.049	91	37	0.566
	10	0.946	0.051	0.019	0.93	0.027	0.003	0.946	0.051	0.003	0.93	0.051	0.019	79	40	0.496
	1	0.923	0.075	0.037	0.89	0.01	0.001	0.923	0.075	0.002	0.89	0.073	0.037	85	40	1.003
	2	0.936	0.092	0.062	0.902	0.055	0.009	0.936	0.055	0.009	0.902	0.055	0.043	44	26	0.516
	3	0.959	0.05	0.039	0.924	0.011	0.006	0.959	0.035	0.006	0.924	0.037	0.039	80	42	0.957
	4	0.957	0.035	0.039	0.878	0.01	0.019	0.957	0.024	0.019	0.926	0.035	0.039	25	12	0.311
$s2$	5	0.936	0.092	0.063	0.93	0.045	0.019	0.936	0.045	0.019	0.93	0.045	0.025	84	41	0.981
	6	0.953	0.05	0.021	0.906	0.049	0.004	0.947	0.049	0.004	0.929	0.05	0.021	83	49	1.042
	7	0.953	0.037	0.019	0.944	0.003	0.001	0.953	0.037	0.01	0.944	0.037	0.019	79	36	0.979
	8	0.958	0.032	0.038	0.941	0.019	0.006	0.958	0.032	0.01	0.941	0.021	0.038	43	21	0.516
	9	0.94	0.102	0.05	0.92	0.057	0.003	0.94	0.057	0.003	0.92	0.057	0.023	36	19	0.438
	10	0.922	0.076	0.026	0.898	0.008	0.001	0.922	0.076	0.002	0.898	0.076	0.026	100	50	1.192
	1	0.94	0.081	0.039	0.9	0.05	0.011	0.939	0.05	0.011	0.9	0.061	0.039	21	13	0.262
	2	0.94	0.067	0.036	0.908	0.009	0.011	0.94	0.049	0.011	0.908	0.056	0.036	27	13	0.317
	3	0.944	0.053	0.051	0.849	0.05	0.018	0.932	0.05	0.018	0.896	0.053	0.051	24	13	0.29
	4	0.961	0.073	0.021	0.944	0.017	0.006	0.961	0.033	0.006	0.944	0.035	0.021	20	12	0.255
$s3$	5	0.961	0.039	0.013	0.945	0.016	0.002	0.961	0.037	0.002	0.948	0.039	0.013	39	23	0.473
	6	0.99	0.083	0.033	0.933	0.032	0.011	0.957	0.032	0.011	0.933	0.034	0.033	23	11	0.292
	7	0.97	0.024	0.012	0.964	0.012	0.001	0.97	0.024	0.006	0.964	0.024	0.012	40	17	0.515
	8	0.942	0.055	0.025	0.905	0.001	0.006	0.942	0.052	0.006	0.92	0.055	0.025	71	31	0.867
	9	0.97	0.071	0.03	0.933	0.021	0.016	0.963	0.021	0.016	0.933	0.037	0.03	65	39	0.785
	10	0.998	0.038	0.024	0.906	0.035	0.006	0.959	0.035	0.006	0.938	0.038	0.024	88	50	1.108
	1	0.941	0.052	0.057	0.841	0.038	0.016	0.941	0.043	0.016	0.891	0.052	0.057	59	26	0.684
	2	0.95	0.038	0.022	0.94	0.014	0.009	0.95	0.038	0.012	0.94	0.038	0.022	97	43	1.229
	3	0.963	0.066	0.055	0.864	0.043	0.028	0.929	0.043	0.028	0.879	0.066	0.055	69	37	0.774
	4	0.997	0.086	0.035	0.902	0.062	0.003	0.935	0.062	0.003	0.902	0.063	0.035	76	38	0.873
$s4$	5	0.982	0.057	0.054	0.841	0.038	0.023	0.939	0.038	0.023	0.889	0.057	0.054	33	19	0.4
	6	0.944	0.063	0.039	0.856	0.059	0.008	0.933	0.059	0.008	0.898	0.063	0.039	79	45	0.972
	7	0.927	0.097	0.064	0.881	0.024	0.029	0.927	0.044	0.029	0.881	0.055	0.064	95	42	1.157
	8	0.956	0.035	0.039	0.926	0.01	0.004	0.956	0.035	0.009	0.926	0.035	0.039	83	46	0.988
	9	0.933	0.072	0.068	0.888	0.054	0.013	0.933	0.054	0.013	0.888	0.054	0.058	73	39	0.857
	10	0.934	0.05	0.024	0.926	0.02	0.001	0.934	0.05	0.016	0.926	0.05	0.024	65	37	0.806
	1	0.911	0.106	0.079	0.872	0.064	0.025	0.911	0.064	0.025	0.872	0.064	0.064	69	34	1.287
	2	0.953	0.051	0.046	0.876	0.047	0.009	0.944	0.047	0.009	0.903	0.051	0.046	21	11	0.437
	3	0.921	0.079	0.079	0.819	0.027	0.043	0.921	0.036	0.043	0.842	0.079	0.079	63	27	1.167
	4	0.933	0.091	0.037	0.904	0.015	0.014	0.933	0.053	0.014	0.904	0.059	0.037	86	44	1.788
$s5$	5	0.936	0.083	0.06	0.849	0.083	0.007	0.91	0.083	0.007	0.857	0.083	0.06	21	11	0.391
	6	0.944	0.055	0.037	0.883	0.004	0.006	0.944	0.05	0.006	0.908	0.055	0.037	57	27	1.184
	7	0.934	0.076	0.055	0.901	0.034	0.019	0.934	0.047	0.019	0.901	0.044	0.055	20	12	0.382
	8	0.974	0.074	0.055	0.856	0.07	0.009	0.921	0.07	0.009	0.871	0.074	0.055	77	46	1.538
	9	0.935	0.055	0.031	0.914	0.035	0.009	0.935	0.055	0.01	0.914	0.055	0.031	60	28	1.16
	10	0.925	0.115	0.04	0.886	0.064	0.011	0.925	0.064	0.011	0.886	0.074	0.04	55	30	1.077
	1	0.952	0.04	0.04	0.92	0.028	0.007	0.952	0.04	0.008	0.92	0.04	0.04	43	21	0.88
	2	0.948	0.045	0.041	0.886	0.002	0.013	0.948	0.039	0.013	0.914	0.045	0.041	32	19	0.671
	3	0.965	0.043	0.014	0.896	0.042	0.001	0.957	0.042	0.001	0.943	0.043	0.014	42	19	0.845
	4	0.981	0.076	0.044	0.907	0.024	0.026	0.95	0.024	0.026	0.907	0.049	0.044	53	25	1.056
$s6$	5	0.95	0.047	0.025	0.933	0.014	0.009	0.95	0.041	0.009	0.933	0.042	0.025	62	28	1.194
	6	0.954	0.075	0.042	0.919	0.011	0.025	0.954	0.021	0.025	0.919	0.039	0.042	56	25	1.092
	7	0.97	0.025	0.023	0.91	0.023	0.006	0.97	0.024	0.006	0.952	0.025	0.023	26	15	0.548
	8	0.974	0.024	0.018	0.937	0.011	0.003	0.974	0.023	0.003	0.958	0.024	0.018	48	26	1.034
	9	0.969	0.07	0.016	0.964	0.023	0.008	0.969	0.023	0.008	0.964	0.023	0.013	86	44	1.713
	10	0.951	0.059	0.068	0.92	0.039	0.01	0.951	0.039	0.01	0.92	0.039	0.041	55	31	1.144
$s7$	1	0.943	0.046	0.053	0.861	0.044	0.014	0.942	0.044	0.014	0.901	0.046	0.053	46	21	0.654
	2	0.941	0.09	0.034	0.92	0.037	0.015	0.941	0.044	0.015	0.92	0.046	0.034	31	18	0.432

	3	0.995	0.051	0.013	0.952	0.032	0.003	0.965	0.032	0.003	0.952	0.035	0.013	70	36	0.971
	4	0.944	0.051	0.048	0.879	0.019	0.005	0.944	0.051	0.005	0.901	0.051	0.048	20	12	0.278
	5	0.967	0.061	0.048	0.962	0.027	0.006	0.967	0.027	0.006	0.962	0.027	0.011	41	18	0.585
	6	0.956	0.055	0.053	0.946	0.038	0.006	0.956	0.038	0.006	0.946	0.038	0.016	25	13	0.346
	7	0.937	0.056	0.025	0.919	0.04	0.001	0.937	0.056	0.007	0.919	0.056	0.025	82	39	1.108
	8	0.953	0.039	0.035	0.918	0.019	0.011	0.953	0.036	0.011	0.926	0.039	0.035	86	50	1.212
	9	0.939	0.06	0.047	0.846	0.038	0.014	0.939	0.047	0.014	0.893	0.06	0.047	23	12	0.328
	10	0.937	0.069	0.05	0.899	0.033	0.015	0.937	0.048	0.015	0.899	0.051	0.05	86	50	1.192
$s8$	1	0.914	0.079	0.036	0.865	0.066	0.008	0.914	0.078	0.008	0.885	0.079	0.036	63	29	0.753
	2	0.944	0.05	0.062	0.922	0.031	0.021	0.944	0.035	0.021	0.922	0.031	0.047	77	37	0.932
	3	0.944	0.05	0.051	0.887	0.019	0.029	0.944	0.027	0.029	0.899	0.05	0.051	33	15	0.4
	4	0.94	0.056	0.034	0.899	0.021	0.007	0.94	0.053	0.007	0.91	0.056	0.034	83	34	0.986
	5	0.944	0.081	0.045	0.914	0.026	0.01	0.944	0.046	0.01	0.914	0.041	0.045	28	13	0.321
	6	0.915	0.072	0.033	0.895	0.037	0.005	0.915	0.072	0.013	0.895	0.072	0.033	63	32	0.774
	7	0.907	0.079	0.051	0.891	0.012	0.003	0.907	0.079	0.014	0.891	0.058	0.051	70	34	0.831
	8	0.932	0.066	0.037	0.873	0.002	0.009	0.932	0.059	0.009	0.897	0.066	0.037	20	12	0.234
	9	0.912	0.101	0.083	0.845	0.01	0.032	0.912	0.056	0.032	0.845	0.072	0.083	72	32	0.812
	10	0.94	0.079	0.078	0.856	0.041	0.037	0.922	0.041	0.037	0.856	0.066	0.078	48	24	0.575
$s9$	1	0.941	0.073	0.036	0.912	0.006	0.012	0.941	0.047	0.012	0.912	0.052	0.036	31	17	0.31
	2	0.936	0.051	0.031	0.918	0.028	0.005	0.936	0.051	0.013	0.918	0.051	0.031	90	44	0.86
	3	0.922	0.082	0.064	0.832	0.075	0.015	0.91	0.075	0.015	0.854	0.082	0.064	90	52	0.868
	4	0.921	0.068	0.029	0.903	0.039	0.007	0.921	0.068	0.011	0.903	0.068	0.029	76	40	0.732
	5	0.929	0.083	0.064	0.904	0.047	0.023	0.929	0.048	0.023	0.904	0.047	0.049	60	25	0.576
	6	0.934	0.063	0.021	0.916	0.04	0.002	0.934	0.063	0.003	0.916	0.063	0.021	87	50	0.917
	7	0.938	0.099	0.049	0.922	0.052	0.01	0.938	0.052	0.01	0.922	0.052	0.026	47	27	0.469
	8	0.978	0.064	0.024	0.89	0.062	0.004	0.934	0.062	0.004	0.912	0.064	0.024	78	45	0.806
	9	0.914	0.076	0.036	0.888	0.029	0.01	0.914	0.076	0.01	0.888	0.076	0.036	66	33	0.644
	10	0.904	0.095	0.031	0.874	0.085	0.001	0.904	0.095	0.001	0.874	0.095	0.031	33	15	0.322
$s10$	1	0.967	0.042	0.037	0.927	0.016	0.021	0.963	0.016	0.021	0.927	0.036	0.037	42	25	0.735
	2	0.968	0.05	0.075	0.945	0.027	0.005	0.968	0.027	0.005	0.945	0.027	0.028	41	24	0.766
	3	0.98	0.022	0.01	0.952	0.019	0.003	0.978	0.019	0.003	0.968	0.022	0.01	91	43	1.662
	4	0.986	0.054	0.008	0.976	0.015	0.002	0.983	0.015	0.002	0.976	0.016	0.008	64	30	1.151
	5	0.961	0.061	0.021	0.947	0.031	0.004	0.961	0.035	0.004	0.947	0.032	0.021	61	35	1.155
	6	0.996	0.013	0.012	0.931	0.008	0.006	0.986	0.008	0.006	0.975	0.013	0.012	34	16	0.626
	7	0.97	0.024	0.016	0.96	0.024	0.006	0.97	0.024	0.006	0.96	0.024	0.016	54	29	0.998
	8	0.973	0.035	0.012	0.962	0.02	0.003	0.973	0.024	0.003	0.962	0.026	0.012	24	12	0.428
	9	0.961	0.07	0.058	0.939	0.031	0.008	0.961	0.031	0.008	0.939	0.031	0.03	33	19	0.62
	10	0.984	0.013	0.015	0.939	0.007	0.008	0.984	0.008	0.008	0.972	0.013	0.015	40	20	0.72
$s11$	1	0.967	0.026	0.031	0.931	0.004	0.014	0.967	0.019	0.014	0.943	0.026	0.031	62	34	0.752
	2	0.968	0.038	0.061	0.962	0.025	0.007	0.968	0.025	0.007	0.962	0.025	0.013	53	22	0.662
	3	0.975	0.02	0.018	0.962	0.017	0.002	0.975	0.02	0.005	0.962	0.02	0.018	42	23	0.528
	4	0.965	0.065	0.063	0.954	0.027	0.008	0.965	0.027	0.008	0.954	0.027	0.019	84	35	1.003
	5	0.978	0.069	0.013	0.966	0.017	0.005	0.978	0.017	0.005	0.966	0.021	0.013	26	15	0.316
	6	0.974	0.021	0.019	0.948	0.007	0.01	0.974	0.016	0.01	0.96	0.021	0.019	73	37	0.91
	7	0.953	0.054	0.051	0.937	0.04	0.007	0.953	0.04	0.007	0.937	0.04	0.023	34	16	0.432
	8	0.973	0.025	0.027	0.925	0.001	0.009	0.973	0.018	0.009	0.948	0.025	0.027	100	45	1.212
	9	0.971	0.027	0.024	0.927	0.008	0.003	0.971	0.026	0.003	0.949	0.027	0.024	40	21	0.507
	10	0.984	0.024	0.016	0.965	0.017	0.002	0.981	0.017	0.002	0.965	0.019	0.016	84	36	1.018
$s12$	1	0.962	0.035	0.034	0.92	0.016	0.014	0.962	0.024	0.014	0.931	0.035	0.034	22	11	0.312
	2	0.989	0.078	0.053	0.896	0.044	0.019	0.937	0.044	0.019	0.896	0.051	0.053	71	42	0.997
	3	0.952	0.043	0.056	0.942	0.042	0.006	0.952	0.042	0.006	0.942	0.042	0.016	51	22	0.69
	4	0.963	0.044	0.07	0.945	0.029	0.008	0.963	0.029	0.008	0.945	0.029	0.026	63	38	0.874
	5	0.932	0.089	0.049	0.924	0.041	0.012	0.932	0.056	0.012	0.924	0.041	0.035	63	37	0.869
	6	0.943	0.104	0.057	0.886	0.022	0.017	0.943	0.04	0.017	0.886	0.057	0.057	44	25	0.62
	7	0.931	0.069	0.035	0.903	0.058	0.012	0.93	0.058	0.012	0.903	0.062	0.035	60	34	0.848
	8	0.925	0.061	0.047	0.892	0.041	0.002	0.925	0.061	0.014	0.892	0.061	0.047	85	50	1.175
	9	0.975	0.065	0.048	0.906	0.03	0.019	0.951	0.03	0.019	0.906	0.046	0.048	78	43	1.118
	10	0.933	0.077	0.037	0.9	0.019	0.01	0.933	0.057	0.01	0.9	0.063	0.037	85	50	1.192
$s13$	1	0.967	0.072	0.014	0.964	0.024	0.009	0.967	0.024	0.009	0.964	0.024	0.012	72	43	0.931
	2	0.985	0.083	0.032	0.933	0.033	0.012	0.955	0.033	0.012	0.933	0.035	0.032	53	28	0.646

	3	0.958	0.034	0.04	0.92	0.008	0.024	0.958	0.018	0.024	0.926	0.034	0.04	53	32	0.623
	4	0.967	0.066	0.023	0.945	0.025	0.005	0.967	0.028	0.005	0.945	0.032	0.023	53	22	0.63
	5	0.971	0.025	0.023	0.907	0.012	0.011	0.971	0.018	0.011	0.952	0.025	0.023	41	19	0.515
	6	0.983	0.07	0.026	0.945	0.018	0.012	0.97	0.018	0.012	0.945	0.029	0.026	100	48	1.243
	7	0.97	0.026	0.015	0.959	0.023	0.002	0.97	0.026	0.004	0.959	0.026	0.015	82	48	1.045
	8	0.955	0.033	0.019	0.948	0.031	0.002	0.955	0.033	0.012	0.948	0.033	0.019	76	32	0.959
	9	0.953	0.056	0.059	0.94	0.043	0.004	0.953	0.043	0.004	0.94	0.043	0.017	98	44	1.157
	10	0.954	0.039	0.054	0.945	0.032	0.004	0.954	0.039	0.007	0.945	0.032	0.023	24	10	0.301
$s14$	1	0.954	0.05	0.044	0.94	0.037	0.009	0.954	0.037	0.009	0.94	0.037	0.023	72	39	0.46
	2	0.931	0.096	0.051	0.884	0.006	0.027	0.931	0.042	0.027	0.884	0.065	0.051	21	9	0.121
	3	0.945	0.043	0.043	0.914	0.037	0.006	0.945	0.043	0.012	0.914	0.043	0.043	85	44	0.488
	4	0.957	0.051	0.042	0.919	0.001	0.023	0.957	0.02	0.023	0.919	0.039	0.042	28	14	0.166
	5	0.939	0.054	0.04	0.867	0.026	0.017	0.939	0.044	0.017	0.906	0.054	0.04	54	29	0.323
	6	0.95	0.039	0.036	0.925	0.005	0.008	0.95	0.039	0.011	0.925	0.039	0.036	41	23	0.25
	7	0.958	0.055	0.041	0.938	0.036	0.006	0.958	0.036	0.006	0.938	0.036	0.026	45	19	0.277
	8	0.929	0.059	0.031	0.91	0.004	0.01	0.929	0.059	0.012	0.91	0.059	0.031	71	41	0.429
	9	0.966	0.032	0.032	0.943	0.001	0.016	0.966	0.018	0.016	0.943	0.025	0.032	30	15	0.18
	10	0.955	0.052	0.03	0.932	0.028	0.015	0.955	0.03	0.015	0.932	0.038	0.03	56	26	0.324
$s15$	1	0.935	0.09	0.071	0.913	0.05	0.015	0.935	0.05	0.015	0.913	0.05	0.037	24	12	0.471
	2	0.983	0.055	0.034	0.923	0.039	0.014	0.947	0.039	0.014	0.923	0.043	0.034	69	32	1.376
	3	0.945	0.083	0.025	0.926	0.011	0.008	0.945	0.047	0.008	0.926	0.049	0.025	38	20	0.762
	4	0.948	0.037	0.017	0.946	0.001	0.006	0.948	0.037	0.015	0.946	0.037	0.017	73	37	1.449
	5	0.957	0.083	0.056	0.947	0.036	0.007	0.957	0.036	0.007	0.947	0.036	0.017	53	32	1.134
	6	0.917	0.074	0.044	0.852	0.047	0.017	0.917	0.066	0.017	0.882	0.074	0.044	72	35	1.366
	7	0.94	0.046	0.025	0.929	0.036	0.008	0.94	0.046	0.014	0.929	0.046	0.025	45	21	0.882
	8	0.938	0.094	0.047	0.917	0.049	0.013	0.938	0.049	0.013	0.917	0.049	0.034	36	19	0.729
	9	0.937	0.103	0.023	0.915	0.047	0.002	0.937	0.061	0.002	0.915	0.062	0.023	47	28	0.917
	10	0.924	0.069	0.068	0.882	0.034	0.012	0.924	0.064	0.012	0.882	0.05	0.068	89	47	1.758

Table 4. component versions in slz-15

lev-4a		$p_{1_{ij}+}$	$p_{1_{ij}-}$	$g_{1_{ij}}$	c_{ij}	*lev-5a*		$p_{1_{ij}+}$	$p_{1_{ij}-}$	$g_{1_{ij}}$	c_{ij}
	1	0.96	0.7739	150	1.02		1	0.98	0.9444	120	0.59
	2	0.969	0.831	100	0.89		2	0.977	0.9521	100	0.535
	3	0.98	0.9553	80	0.72		3	0.982	0.9501	85	0.47
$s1$	4	0.964	0.8068	80	0.62	$s1$	4	0.978	0.9691	85	0.42
	5	0.97	0.7729	50	0.52		5	0.983	0.9692	48	0.4
							6	0.92	0.8744	31	0.18
							7	0.984	0.9631	26	0.22
	1	0.953	0.8155	75	1.367		1	0.995	0.9743	100	0.205
	2	0.96	0.81	50	0.967		2	0.996	0.9713	92	0.189
$s2$	3	0.914	0.8647	50	0.916	$s2$	3	0.997	0.9952	53	0.091
	4	0.967	0.8641	20	0.516		4	0.997	0.9679	28	0.056
							5	0.998	0.9745	21	0.042
	1	0.96	0.877	240	0.813		1	0.971	0.9662	100	7.525
	2	0.97	0.9208	200	0.783		2	0.973	0.9506	60	4.72
	3	0.96	0.9364	200	0.614		3	0.971	0.9263	40	3.59
$s3$	4	0.959	0.7701	180	0.534	$s3$	4	0.976	0.9505	20	2.42
	5	0.97	0.9567	90	0.384						
	6	0.959	0.8806	60	0.214						
	1	0.98	0.8623	70	1.26		1	0.977	0.9755	115	0.18
	2	0.96	0.7808	70	1.19		2	0.978	0.9628	100	0.16
	3	0.98	0.8096	30	0.697		3	0.978	0.9732	91	0.15
$s4$	4	0.989	0.7992	25	0.683	$s4$	4	0.983	0.9724	72	0.121
	5	0.979	0.9681	25	0.645		5	0.981	0.9675	72	0.102
							6	0.971	0.947	72	0.096
							1	0.983	0.9467	55	0.071
							2	0.982	0.9352	25	0.049
						$s5$	3	0.977	0.969	25	0.044
							4	0.984	0.9575	128	0.986

			5	0.983	0.9616	100	0.825

Table 5. component versions in lev-4a and lev-5a

REFERENCES

[1] Kuo W, Wan R. Recent advances in optimal reliability allocation. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans. 2007;37(2):143-56.

[2] Der Kiureghian A, Ditlevsen O. Aleatory or epistemic? Does it matter? Structural Safety. 2009;31(2):105-12.

[3] Wang Y, Li L. Heterogeneous redundancy allocation for series-parallel multi-state systems using hybrid particle swarm optimization and local search. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans. 2012;42(2):464-74.

[4] Ouzineb M, Nourelfath M, Gendreau M. Tabu search for the redundancy allocation problem of homogenous series–parallel multi-state systems. Reliability Engineering & System Safety. 2008;93(8):1257-72.

[5] Ouzineb M, Nourelfath M, Gendreau M. A heuristic method for non-homogeneous redundancy optimization of series-parallel multi-state systems. Journal of Heuristics. 2011;17(1):1-22.

[6] Li CY, Chen X, Yi XS, Tao JY. Heterogeneous redundancy optimization for multi-state series–parallel systems subject to common cause failures. Reliability Engineering & System Safety. 2010 Mar 31;95(3):202-7.

[7] Kleinberg J, Rabani Y, Tardos É. Allocating bandwidth for bursty connections. SIAM Journal on Computing. 2000;30(1):191-217.

[8] Dengiz B, Altiparmak F, Smith AE. Local search genetic algorithm for optimal design of reliable networks. IEEE Transactions on Evolutionary Computation. 1997;1(3):179-88.

[9] Gupta R, Agarwal M. Penalty guided genetic search for redundancy optimization in multi-state series-parallel power system. Journal of combinatorial optimization. 2006;12(3):257.

[10] Levitin G, Lisnianski A, Ben-Haim H, Elmakis D. Redundancy optimization for series-parallel multi-state systems. IEEE Transactions on Reliability. 1998;47(2):165-72.

[11] Marseguerra M, Zio E, Podofillini L, Coit DW. Optimal design of reliable network systems in presence of uncertainty. IEEE Transactions on Reliability. 2005;54(2):243-53.

[12] Garg H, Sharma SP. Multi-objective reliability-redundancy allocation problem using particle swarm optimization. Computers & Industrial Engineering. 2013;64(1):247-55.

[13] Destercke S, Sallak M. An extension of universal generating function in multi-state systems considering epistemic uncertainties. IEEE Transactions on reliability. 2013;62(2):504-14.

[14] Ding Y, Lisnianski A. Fuzzy universal generating functions for multi-state system reliability assessment. Fuzzy Sets and Systems. 2008;159(3):307-24.

[15] Deb K, Pratap A, Agarwal S, Meyarivan TA. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation. 2002;6(2):182-97.

[16] Shaked M, Shanthikumar JG. Stochastic orders. Springer Science & Business Media; 2007.

[17] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE transactions on Evolutionary Computation. 1999;3(4):257-71.

[18] Han KH, Kim JH. Genetic quantum algorithm and its application to combinatorial optimization problem. In IEEE Proceedings of the 2000 Congress on Evolutionary Computation. 2000; (Vol. 2, p. 1354-1360).

[19] Han KH, Kim JH. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE transactions on evolutionary computation. 2002;6(6):580-93.

[20] Yalaoui A, Châtelet E, Chu C. A new dynamic programming method for reliability & redundancy allocation in a parallel-series system. IEEE transactions on reliability. 2005;54(2):254-61.

[21] Elegbede AC, Chu C, Adjallah KH, Yalaoui F. Reliability allocation through cost minimization. IEEE Transactions on reliability. 2003;52(1):106-11.

[22] 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(1):182-91.

[23] Ha C, Kuo W. Reliability redundancy allocation: An improved realization for nonconvex nonlinear programming problems. European Journal of Operational Research. 2006;171(1):24-38.

[24] Kleinberg J, Rabani Y, Tardos É. Allocating bandwidth for bursty connections. SIAM Journal on Computing. 2000;30(1):191-217.

[25] Li J, Shi T. A fully polynomial-time approximation scheme for approximating a sum of random variables. Operations Research Letters. 2014;42(3):197-202.

[26] Jaszkiewicz A. On the performance of multiple-objective genetic local search on the 0/1 knapsack problem-a comparative experiment. IEEE Transactions on Evolutionary Computation. 2002;6(4):402-12.

[27] Ishibuchi H, Yoshida T, Murata T. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE transactions on evolutionary computation. 2003;7(2):204-23.

[28] Jaszkiewicz A. Genetic local search for multi-objective combinatorial optimization. European journal of operational research. 2002;137(1):50-71.

[29] Zhang Q, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on evolutionary computation. 2007;11(6):712-31.

[30] Levitin G, Lisnianski A, Elmakis D. Structure optimization of power system with different redundant elements. Electric Power Systems Research. 1997;43(1):19-27.

[31] Hsieh TJ, Yeh WC. Penalty guided bees search for redundancy allocation problems with a mix of components in series–parallel systems. Computers & Operations Research. 2012 N;39(11):2688-704.

[32] Li YF, Zio E. A quantum-inspired evolutionary approach for non-homogeneous redundancy allocation in series-parallel multi-state systems. In IEEE International Conference on Reliability, Maintainability and Safety (ICRMS), 2014.

[33] Li CY, Chen X, Yi XS, Tao JY. Interval-valued reliability analysis of multi-state systems. IEEE Transactions on Reliability. 2011;60(1):323-30.

[34] Lisnianski A, Levitin G. Multi-state system reliability: assessment, optimization and applications. World Scientific Publishing Co Inc; 2003.

[35] Sun, MX. Li YF, Zio E. Approximate the optimal solution for reliability allocation problem of multi-state series systems. Working paper, preparing for pre-publication on arxiv.

[36] Lin YH, Li YF, Zio E. Fuzzy reliability assessment of systems with multiple-dependent competing degradation processes. IEEE Transactions on Fuzzy Systems. 2015;23(5):1428-38.

[37] Li YF, Ding Y, Zio E. Random fuzzy extension of the universal generating function approach for the reliability assessment of multi-state systems under aleatory and epistemic uncertainties. IEEE Transactions on Reliability. 2014;63(1):13-25.