Reliability Engineering and System Safety 156 (2016) 148–158

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

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

Two-stage simplified swarm optimization for the redundancy allocation problem in a multi-state bridge system

Chyh-Ming Lai [a,*], Wei-Chang Yeh [b]

[a] The Institute of Resources Management and Decision Science, Management College, National Defense University, Taipei 11258, Taiwan
[b] Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 300, Taiwan

ARTICLE INFO

Article history:
Received 18 July 2015
Received in revised form
10 July 2016
Accepted 31 July 2016
Available online 1 August 2016

Keywords:
Redundancy allocation problem
Multi-state bridge system
Reliability
Simplified swarm optimization

ABSTRACT

The redundancy allocation problem involves configuring an optimal system structure with high reliability and low cost, either by alternating the elements with more reliable elements and/or by forming them redundantly. The multi-state bridge system is a special redundancy allocation problem and is commonly used in various engineering systems for load balancing and control. Traditional methods for redundancy allocation problem cannot solve multi-state bridge systems efficiently because it is impossible to transfer and reduce a multi-state bridge system to series and parallel combinations. Hence, a swarm-based approach called two-stage simplified swarm optimization is proposed in this work to effectively and efficiently solve the redundancy allocation problem in a multi-state bridge system. For validating the proposed method, two experiments are implemented. The computational results indicate the advantages of the proposed method in terms of solution quality and computational efficiency.

1. Introduction

The last three decades have seen a growing importance placed on research for the redundancy allocation problem (RAP) due to its wide and valuable application [1,2]. RAP involves configuring an optimal system structure with high reliability and low cost either by alternating the elements with more reliable elements and/or forming them redundantly. RAP can be divided into two categories based on the system framework: binary-state systems and multi-state systems [3–6]. Only the two states of operating or complete failure can be experienced in a binary-state system. Compared to a binary-state system, a multi-state system, which can experience more than two extreme states, is more practical in the real world [7,8]. It is becoming an important performance measurement tool for real world engineering systems from the design phase to the control phase [1,2].

RAP has been proved to be NP-hard because the burdens of computational complexity grow with the size of the system [9]. For reducing those burdens, existing approaches mainly focus on developing approximation methods, including the Monte-Carlo simulation [10], the linear programming with an approximate linear objective function [11], tabu search [12], genetic algorithm (GA) [13–17], ant colony optimization [18], particle swarm optimization (PSO) [19,20], artificial immune algorithm [21] and artificial bee colony algorithm [22].

When RAP is applied to a multi-state system, availability is a common measurement to evaluate the reliability of a system [23–26]. However, the availability of some systems cannot be evaluated by the abovementioned methods because those systems cannot be reduced to series and parallel combinations of elements [23,26–28]. One of them is a bridge system. The simplest bridge system is depicted in Fig. 1 in which subsystems 1 and 2; and subsystems 3 and 4 have the same functionality. Those subsystems construct two series substructures connected in parallel. Subsystem 5 performs as an intermediator to balance the load between two parallel flow lines. This system structure is widely used in various engineering systems for load balancing and control [23–26,29–33].

A multi-state bridge system (MSBS) allows the mixture of element type in the subsystem which will greatly increase the computational complexity of the problem. Consider the simplest bridge system in Fig. 1. Suppose that m represents the number of element types can be chosen for each subsystem and n represents the largest number of elements for each type that can be simultaneously mounted in the subsystem. Thus, there are a total of $(n+1)^m$ ways to construct each subsystem. Then the search space for the simplest bridge system is $(n+1)^{5m}$.

As the bridge system cannot be reduced to the series-parallel combinations and its computational complexity sharply increases

Abbreviations: RAP, Redundancy allocation problem; MSBS, Multi-state bridge system; GA, Genetic algorithm; PSO, Particle swarm optimization; SSO_{TS}, Two-stage simplified swarm optimization; UGF, Universal generation function; MTO, Merge-to-one; RIP, Re-initialize population; ESS, Exploitation search strategy

* Corresponding author.
E-mail address: chyh.ming.lai@gmail.com (C.-M. Lai).

http://dx.doi.org/10.1016/j.ress.2016.07.025
0951-8320/

Notations

Nsub	The number of subsystems in a system
M_j	The number of element types available for the *j*th subsystem
X	$X=(X_1, X_2, \ldots, X_{Nsub})$ represents the integrated system structure
X_j	$X_j=(x_{j,1}, x_{j,2}, \ldots, x_{j,k})$ is the *j*th subsystem structure in *X*
$x_{j,k}$	The number of type *k* elements used in X_j
c_{jk}, a_{jk}, w_{jk}	The cost, availability and capacity of type *k* element available for the *j*th subsystem
$C(X)$, $A(X)$, $W(X)$	The total cost, availability and performance level of the system *X*
A_0, W_0	The given availability limitation and demand performance level
C_w, C_p, C_g	Three predefined parameters in simplified swarm optimization
pBest, p_{ij}	The personal best solution, $p_i=(p_{i1}, p_{i2},\ldots, p_{ij})$ represents the *pBest* of the *i*th solution, where p_{ij} denotes the *j*th variable in the *i*th *pBest*
gBest, g_j	The global best solution, $g=(g_1, g_2,\ldots g_j)$ represents the *gBest*, where g_j denotes the *j*th variable in *gBest*
ρ	An uniform random number in [0,1]
$UB_{j,k}$	The upper bound of $x_{j,k}$
λ_1, λ_2	The penalty coefficients
Nmto, *Nrip*	The parameters for MTO and RIP, respectively
Giter, *Iter*, *Niter*	The number of iterations that found the current *gBest*, current number of iterations and total number of iterations, respectively.
Nsol, *Nsub*, *Nele*	The population size, the total number of subsystems and element types in the integrated system structure *X*, respectively.
C_{min}, C_{avg}, C_{max}	The minimal, average and maximal fitness values, respectively
T_{min}, T_{avg}, T_{max}	The minimal, average and maximal CPU time, respectively
C_{std}, T_{std}	The standard deviations of the fitness value and CPU time, respectively
SR	The successful rate, a ratio of the number for obtaining final *gBest* that is equal to the best-known solution to the total number of runs
NFE	The total number of fitness evaluations

with the size of the system, Levitin and Lisnianki first addressed MSBS with three different constraints and evaluated the availability of MSBS by GA [23–25,31]. Recently, Wang and Li combined PSO and local refining strategies to address this problem [26]. The result shows that they have made important contributions to find nearer optimal solutions in a more efficient way than GA. However, there is still some room to improve the efficiency of the above two works. Hence, this paper proposes a novel algorithm called two-stage simplified swarm optimization (SSO_{TS}) that offers an alternative method for solving MSBS.

The rest of this paper is organized as follows: Section 1 describes the formulation of the problem and structure representation. The applications of the universal generating function (UGF) for RAP in MSBS and a short description of simplified swarm optimization (SSO) are given in Section 3. The proposed SSO_{TS} and its overall procedure are detailed in Section 4. The two experiments implemented for validating SSO_{TS} are illustrated in Section 5. Finally, the conclusions are presented in Section 6.

2. Problem formulation and structure representation

Consider a MSBS, let *Nsub* be the number of subsystems in the system and let M_j be the number of element types available for the *j*th subsystem. Let $x_{j,k}$ represents the number of *k* type elements used in the *j*th subsystem, and $X_j=(x_{j,1}, x_{j,2}, \ldots, x_{j,k})$ be the *j*th subsystem vector (structure), where $k=1, 2, \ldots, M_j$. Each element is characterized by its availability a_{jk}, capacity w_{jk} and cost c_{jk}. The capacity of an element is the quantitative measure of its performance and measured as percentage of nominal total system performance level. The integrated system structure is denoted by $X=(X_1, X_2, \ldots, X_{Nsub})$, and the total cost $C(X)$ can be calculated by using the following equation:

$$C(X) = \sum_{j=1}^{Nsub} \sum_{k=1}^{M_j} c_{j,k} x_{j,k} \tag{1}$$

The system availability $A(X)$ can be defined as the probability that total system performance level $W(X)$ is equal to or greater than a specific demand level W_0 is given by $P(W(X) \geq W_0)$ [23,26]. A common formulation of MSBS can be formulated as the following integer nonlinear programming problem:

$$\text{Minimize} \quad C(X) \tag{2}$$

$$\text{s. t.} \quad A(X) \geq A_0 \tag{3}$$

$$W(X) \geq W_0 \tag{4}$$

Eq. (2) indicates that a MSBS structure $X=(X_1, X_2, \ldots, X_{Nsub})$ is configured, while the total cost $C(X)$ is minimized. Simultaneously, the availability $A(X)$ and performance level $W(X)$ of the system should satisfy the given availability limitation A_0 and demand performance level W_0 as indicated by Eqs. (3) and (4), respectively.

For further explaining the structure representation and the calculation of the total cost, an example is illustrated by using a MSBS benchmark reported in [23,26,31] as shown in Fig. 2. The benchmark is a typical extension in which a bridge system consists of five subsystems connected with one subsystem in series, *i.e.*, $Nsub=6$. For each subsystem, $j=1, 2, \ldots, 6$ has $M_j=(6, 6, 4, 4, 5, 3)$ element types can be formed in parallel. The corresponding data of each element type for the benchmark are listed in Table 1.

Example 1. $X=(000102, 000102, 0010, 0100, 00110, 100)$ is a solution of the above benchmark, and its structure can be diagrammed as shown in Fig. 3. The total cost of this system is calculated as shown in Eq. (5):

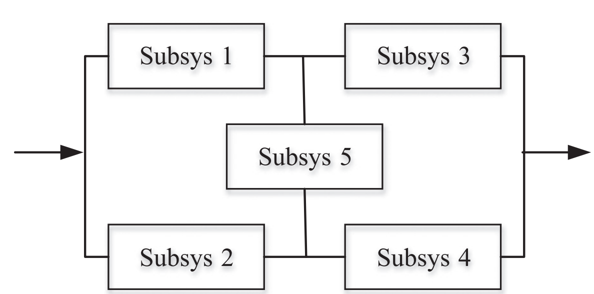

Fig. 1. The simplest bridge.