Journal of Optimization in Industrial Engineering 8 (2011) 65-72

Genetic Algorithm and Simulated Annealing for Redundancy Allocation Problem with Cold-standby Strategy

Aida Karimi[a], Mani Sharifi[b,*], Amirhossain Chambari[c]

[a] *MSc, Faculty of Mechanical and Industrial Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran*

[b] *Assistant Professor, Faculty of Mechanical and Industrial Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran*

[c] *Young Research Club, Qazvin Branch, Islamic Azad university, Qazvin, Iran*

Received 10 December, 2010; Revised 6 January, 2011; Accepted 11 February, 2011

Abstract

This paper presents a new mathematical model for a redundancyallocation problem (RAP) withcold-standby redundancy strategy and multiple component choices.The applications of the proposed model arecommon in electrical power, transformation,telecommunication systems,etc.Manystudies have concentrated onone type of time-to-failure, butin thispaper, two components of time-to-failures which follow hypo-exponential and exponential distributionare investigated. The goal of the RAP is to select available components and redundancy level for each subsystem for maximizing system reliability under cost and weight constraints.Sincethe proposed model belongs to NP-hard class, we proposed two metaheuristic algorithms; namely, simulated annealing and genetic algorithm to solve it. In addition, a numerical example is presented to demonstrate the application of the proposed solution methodology.

Keywords: Redundancy allocation problem; Cold-standby; Series-parallel systems; Genetic algorithm; Simulated annealing.

1. Introduction

One of the most well-known reliability optimization problems is redundancy allocation problem (RAP) which involves the selection of components from among discrete choices with appropriate levels of redundancy to maximize system reliability under some predefined constraints. The RAP has been studied in great detail as an efficient means to select sound design configurations. Furthermore, the RAP is considered for various system structures such as series, parallel, network, parallel-series Yalaoui et al. [27], k-out-of-n by Coit et al. [7]. Accordingly, the series-parallel RAP is investigated in this paper. The configuration of the series-parallel system is presented in Fig. 1.The RAP can be classified into two groups: 1) Redundancy allocation problems without component mixing (RAPCM): those problems where a mix of components within a subsystem is not allowed; and 2) Redundancy allocation problems with a mix of components (RAPMC): those problems in which a mix of components is allowed within a subsystem (Kuo et al. [16]). This problem pertains to the first classification.

Whereas in active redundancy all components are operated from the time zero simultaneously, in the standby redundancy arrangement the redundant components are sequentially used in the system during component failure times. When the component in Operation fails, one of the redundant units is switched on to continue the system operation.

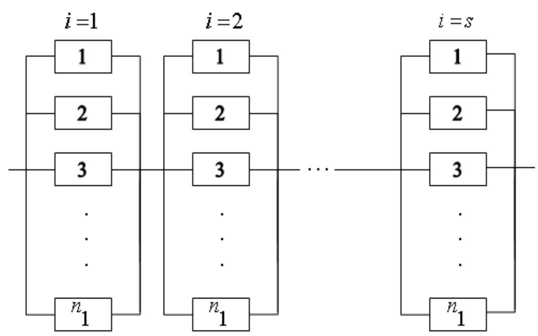

Fig. 1. series-parallel system

There are three variants of the standby redundancy; namely, cold, warm, and hot. This paper pertains to cold-standby redundancy strategy. In the cold-standby redundancy, the component does not fail before it operates by Tavakkoli-Moghaddam et al. [25]. We classified literature review in this area based on active and cold-standby redundancy.

In active redundancy, in order to maximize the reliability of the system, different methods and algorithms were

* Corresponding author Email: M.Sharifi@Qiau.ac.ir

developed including dynamic programming by Fyffe et al. [9] and Nakagawa et al. [19], integer programming by Bulfin et al. [3], and different types of meta-heuristic algorithms such as genetic by Ida et al. [13] and Coit et al. [8], tabu search by Ouzineb et al. [20], variable neighborhood search Liang et al. [17], and particle swarm optimization by Beji et al. [2]. Also, it is worth mentioning that an overview of research in this area can be found in this area can be found by Kuo et al. [15]. Besides, in cold-standby redundancy, a review of more than one hundred references descring reliability optimization researches with different types of redundancy is completed by Tillman et al. [26]. Among these studies, the minority pertained to standby redundancy. Robinson et al. [22] considered the cold-standby redundancy which concentrates on system design without repairable systems. Shankar et al. [23], Gurov et al. [10] investigated the problem of imperfect switching. Also, Coit et al. [7] presented a new problem formulation and solution method to determine the optimal system design configuration when a system design includes k-out-of-n subsystems which are designed with either active or cold-standby redundancy. Coit [6] proposed strictly cold-standby redundancy as an integer programming solution in the RAP area. Sharifi et al. [24] presented an efficient model in redundancy systems for cold-standby strategy with hypo-exponential Time-to-failure distribution.

In this article a new model for redundancy allocation problems without component mixing for series-parallel systems when redundancy strategy is cold-standbyis proposed. Most mathematical models of general redundancy allocation problem assume one type of time-to-failure. Nowadays, exponential distribution is receiving more attention. The conditions of time-to-failure within a particular system design are much closer to the real world. To do so, this paper proposed a new model with two time-to-failures including exponential and hypo-exponential. The objective function is maximizing system reliability under cost and weight constraints. Since the RAP has been shown to be NP-hard by Chern [5], the simulated annealing and genetic algorithms have been proposed.

The remainder of this paper is organized as follows: In section 2, the problem is defined and the mathematical model is illustrated.The proposedgenetic algorithm and simulated annealing for solving the problem are investigated in Section 3. In section 4, the computational experiment and the analysis of the results are provided. At the end, some conclusions and suggestions for future research are presented in section 5.

2. Problem Definition

In this section, the mathematical model of the series-parallel system with s subsystem under cost and weight constraints is illustrated.In the proposedmodel, redundancy strategy iscold-standby with perfect switching.

For this problem, component time-to-failure is distributed according to hypo-exponential and exponential. Inaddition, the following assumptions are provided:

Assumptions:

- Failed components do not damage the system, and are not repaired.
- Failures of individual components are s-independent.
- The states of the elements and the system are either good or have failed.
- The RAP without component mixing is considered.
- Components are cold-standby redundant.
- The supply of components is unlimited.
- The components of reliabilities, weights and costs, are known and deterministic.

3.1. Mathematical Model

Nomenclature

i index of subsystem $i = 1, 2, \ldots, s$;

s number of subsystems;

n_i number of components used in Subsystem i $n_i \in \{1, 2, \ldots, n_{Max,i}\}$;

$n\,(n_1, n_2, \ldots, n_s)$ n ;

m_i number of available component choices for a subsystem i ;

z_i index of component choice used for a subsystem i , $z_i \in \{1, 2, \ldots, m_i\}$;

$z\,(z_1, z_2, \ldots, z_s)$;

$n_{Max,i}$ upper bound for n_i $(n_i \le n_{Max,i})$;

t mission time (fixed);

$r_{i,j(t)}$ reliability at time t for the j^{th} available component fonr subsystem i ;

$\lambda_{i,j}$ scale parameter the exponential distribution for j^{th} available component for subsystem i ;

$\alpha_{i,j}, \beta_{i,j}$ parameters the hypo-exponential distribution for j^{th} available component for subsystem i ;

w system-level constraint limit for weight;

C system-level constraint limit for cost;

c_{ij}, w_{ij} cost and weight for the j^{th} available component for the subsystem $j^{th} i$

$R(t, z, n)$ system reliability at time t for designing vectors z and n ;

The mathematical formulation can be formulated as follows:

$$Max\ Z = R(t, z, n) \quad (1)$$

$$s.t:$$

$$\sum_i c_{iz_i} n_i \le C \quad (2)$$

$$\sum_i w_{iz_i} n_i \le W \quad (3)$$

$$n_i \in \{1, 2, \ldots, n_{Max,i}\}$$

$$z_i \in \{1, 2, \ldots, m_i\}$$

Objective function (1) is defined to maximize the reliability system.Constraint (2) considers the available cost. Constraint (3) considers the available weight.

3. Metaheuristics

In this section, two metaheuristic algorithms including genetic algorithms and simulated annealingare proposed to solve the problem. In the next subsection, we describe the algorithms for our problem.

3.2. Genetic Algorithm (GA)

GA is a stochastic search algorithmbased on the mechanism of natural selection and natural genetics.The basic concepts of GA were introduced by Holland [11]. With regards to the growing interest and simplicity of the GA and its ability for discovering goodsolutions fast, this metaheuristic is selected as one of the solving methodologies.In the next subsections, we present the required steps for solving the problem by a GA.

3.2.1. Chromosome Representation

Each possible solution to this problem is a collection of selected components, and n_i parts in parallel for each subsystem. n_i parts can be chosen only in one combination amongst the m_i available components. The solution encoding is $2 \times s$ matrix. The first and second rows demonstrate type of selected components, and thenumber of selected components, respectively. The columns represent subsystem. Fig.2 presents an example of encoding solution with 14 subsystems. Thismatrix represents a prospective solution with four of thethird component used in parallel forthe first subsystem; two of the second component used in parallel for the second subsystem, etc.

3.2.2. Initial Population

A GA requires a population of potential solutions of the given problem to be initialized. The initial population of individuals is randomly generated by a number of chromosomes (population size or pop size).

3.2.3. Constraint-handling and Fitness Function

This evaluation is achieved through the computation of the cost associated with each chromosome, using the fitness function. The offspring produced by the GA operators is likely to be infeasible. The most common approach in the GA community to handle constraint is to use penalties. The idea of this method is to transform a constrained optimization problem into an unconstrained one by adding or multiplying a certain value to/by the objective function based on the amount of constraint violation presented in a certain solution by Ozgur [21]. In this study, we use the multiplicative form of the penalty function (Pen(S)) and the fitness function (fitn(S)) with the following form:

$$\begin{aligned} &fitn(s) = f(s) \times Pen(s) \\ &Pen(s) = 0\ if\ \ f\ \ is\ \ feasible \\ &Pen(s) > 0\ otherwise \end{aligned} \quad (4)$$

Where $f(s)$ the objective is function in Eq. (1) and s represents a solution. In this approach, we search for the solution that maximize $fitn(s)$.

3.2.4. Selection Operator

In the next phase of the genetic algorithm, the chromosomes for the next generation are selected. In this paper, a 'roulette wheel selection" procedure has been applied for the selection operator.

3.2.5. Crossover Operator

The crossover operator is the basic operator of producing new chromosomes in a GA. It operates on two-parent solutions with probability p_c and generates offspring by recombining both parent solution features.This operator first generates a random crossover mask and then exchanges relative genes between parents according to the mask by Hou [12]. For instance, the crossover is performed as depicted in Fig.3.

Subsystem	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Type of component selected	3	2	3	2	2	1	1	2	1	1	2	3	3	4
Number of component	4	2	2	3	4	4	2	1	2	1	2	4	3	4

Fig. 2. chromosome representation

$$\begin{pmatrix} 2 & 1 & 2 & 3 & 1 & 1 & 1 & 1 & 4 & 1 & 1 & 2 & 2 & 3 \\ 4 & 3 & 4 & 4 & 3 & 6 & 6 & 5 & 6 & 1 & 3 & 2 & 2 & 5 \end{pmatrix} \quad \begin{pmatrix} 1 & 1 & 2 & 3 & 1 & 1 & 1 & 2 & 4 & 3 & 2 & 2 & 2 & 4 \\ 2 & 3 & 1 & 5 & 3 & 6 & 6 & 5 & 6 & 5 & 3 & 2 & 3 & 5 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 2 & 2 & 1 & 4 & 1 & 2 & 4 & 3 & 2 & 3 & 2 & 4 \\ 2 & 3 & 1 & 5 & 1 & 6 & 6 & 5 & 6 & 5 & 4 & 2 & 3 & 5 \end{pmatrix} \quad \begin{pmatrix} 2 & 2 & 2 & 2 & 1 & 4 & 1 & 1 & 4 & 1 & 1 & 3 & 2 & 3 \\ 4 & 3 & 4 & 4 & 3 & 6 & 6 & 5 & 6 & 1 & 4 & 2 & 2 & 5 \end{pmatrix}$$

Fig. 3. Example of crossover operator

3.2.6. Mutation Operator

Mutation is the second operation in a GA method thatexplores the solution spaces which are not explored by crossover operator. It operates on one parent solution with probability p_m . In this paper, the swap mutation operator was used. In swap operator, two position row matrixes are selected randomlyand their contents are swapped.For instance, the mutation is performed as depicted in Fig.4.

3.2.7. Stopping Criteria

In this research, the stopping criteria are defined as the number of generations. The algorithm will be stopped, when that reaches a predefined number of generations.

3.3. Simulated Annealing (SA)

SA is a well-known local search metaheuristic, as presented by Aarts et al. [1]. SA is based on the Monte Carlo method introduced by Metropolis et al. [18]. This idea was originally used to simulate a physical annealing process and was applied to combinatorial optimization for the first time in the 1980s independently by Kirkpatrick et al. [14], and Cerny [4]. The pseudo code of SA algorithm is presented in Fig.5, where the following notation is used:

s =The current solution, s^* =The best solution, s_n =Neighboring solution, $f(s)$=The value of objective function at solution s , n =Repetition counter, T_0 =Initial temperature, L =Number of repetition allowed at each temperature level, p =Probability of accepting s_n when it is not better than s .

For Applying SA to the problem under consideration, some requirements should be defined including a solution representation, fitness function, and the neighborhood identification of the current solution. In what follows we present the requirements of the SA algorithm for this problem.

$$\begin{pmatrix} 2 & 3 & \mathbf{4} & 1 & 3 & 2 & 1 & 1 & 2 & 4 & 3 & \mathbf{3} & 3 & 4 \\ 2 & 3 & 1 & 3 & 2 & 1 & 2 & 3 & 1 & 2 & 2 & 3 & 1 & 3 \end{pmatrix} \quad \begin{pmatrix} 2 & 3 & \mathbf{3} & 1 & 3 & 2 & 1 & 1 & 2 & 4 & 3 & \mathbf{4} & 4 & 4 \\ 2 & 3 & 1 & 3 & 2 & 1 & 2 & 3 & 1 & 2 & 2 & 3 & 1 & 3 \end{pmatrix}$$

Fig. 4. example of mutation operator

```
Initialize the SA control parameter (T0,L)
Select an initial solution, s0
Set T = T0 ; Set s = s0; Set s* = s0; Calculate f(s0);
While the stop criterion is not reached do:
   Set n = 1;
   While n < L do:
 generate solution sn in the neighborhood of s0; Calculate Δ= f(sn) − f(s);
      If Δ≤ 0
         s = sn
      Else
         generate a random number, r ∈ (0,1)
         if (r ≤ p = e^(−Δ/T));
            s = sn; n = n + 1;
         End
      End
      if (f(s) < f(s*))
         s* = sn;
      End
   End
   reduce the temperature T;
End
```

Fig. 5. Pseudo-code SA

3.3.1. Chromosome Representation

The solution representation in SA algorithm is the same as one in genetic algorithm.

3.3.2. Neighbour Generation

Neighbosolutionfrom the current solution performs according to pseudo-code presented in Fig.6.

```
Generate a random number, r ∈ (0,1)
If r ≥ 0.5
first row is selected, Generate a random number, b ∈ (0,1)
If b ≥ 0.5
two elements are selected and swapped
else
one element are selected and its value replaced with a random
numberbetween1and m_i
end
else
second row is selected, Generate a random number, q ∈ (0,1)
If q ≤ 0.5
one element are selected and its value replaced with a random
number between 1and n_{Max,i}
else
two elements are selected and swapped.
End
End
```

Fig. 6. Pseudo-code of neighbor solution

3.3.3. Constraint-handling and Fitness Function

This section is like above mentioned the section 4.2.3 genetic algorithm.

4. A Numerical Example

To evaluate the performance of the GA and SA, 33 test problems were provided by varying incrementally the available weight from 100 to 132 while fixing the available cost = 75 in order to maximize system reliability at $t = 250$ hours. This example is an adapted version of an example provided by Fyffe et al. [5].In these problems, there is series-parallel system with 14 subsystems.Each subsystem has two, three or four components of choice and the number maximum of component within a subsystem has been defined to be three.The switch operates and fails as perfect switchingis used. The data test problems are given in Table 1.All the test problems are coded MATLAB 7.8.0 (R2009a). Due to the stochastic nature of the proposed algorithms, for each of the test problems it is run 4 times and the best solution amongst them is considered as the final solution. The computational results for two algorithms are shown in Tables2and 3.The results show that there is no significant difference among the standard deviation of the two algorithms solutions. To compare the results of the best solution, we performed a paired T-test. Fig.8. shows the results of T-test. Because of the p-value (0.187) greater than $\alpha(0.5)$ so, two algorithms have similar results and no one is better than the other one. Fig.7 shows the standard deviation of the two proposed algorithms.

Table 1
Component data for example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13	**14**
Choice(j=1)	λ_{ij}	0.0022	-	0.0003	0.0013	0.0009	-	-	0.0002	0.0019	-	-	-	0.0017	**0.0020**
	α_{ij}	-	0.0007	-	-	-	0.0008	0.0003	-	-	0.0002	0.0004	0.0009	-	-
	β_{ij}	-	0.0025	-	-	-	0.0075	0.0042	-	-	0.0068	0.008	0.0046	-	-
	c_{ij}	5	1	5	2	2	1	3	5	1	6	3	1	2	**5**
	w_{ij}	3	2	7	8	7	2	3	8	10	8	5	3	8	**7**
	pdf	E	H	E	E	E	H	H	E	E	H	H	H	E	**E**
Choice(j=1)	λ_{ij}	-	0.0009	0.0014	-	-	0.0001	0.0016	0.0020	0.0010	-	0.0010	0.0020	-	-
	α_{ij}	0.0001	-	-	0.0005	0.0009	-	-	-	-	0.0004	-	-	0.0008	**0.008**
	β_{ij}	0.0068	-	-	0.0011	0.0023	-	-	-	-	0.0006	-	-	0.0006	**0.0044**
	c_{ij}	5	6	3	5	5	1	1	2	5	4	5	1	2	**4**
	w_{ij}	10	2	10	10	7	7	7	10	9	5	5	4	5	**5**
	pdf	H	E	E	H	H	E	E	E	E	H	E	E	H	**H**
Choice(j=3)	λ_{ij}	-	0.0012	-	-	0.0014	-	-	0.0011	0.0022	0.0006	-	0.0016	0.0011	**0.0007**
	α_{ij}	0.0007	-	-	0.0003	-	-	-	-	-	-	-	-	-	-
	β_{ij}	0.0031	-	-	0.0065	-	-	-	-	-	-	-	-	-	-
	c_{ij}	6	5	-	1	5	-	-	4	2	2	-	1	6	**4**
	w_{ij}	8	9	-	4	8	-	-	10	9	3	-	2	8	**7**
	pdf	H	E	-	H	E	-	-	E	E	E	-	E	E	**E**
Choice(j=4)	λ_{ij}	-	-	-	0.0005	-	-	-	-	0.0004	-	-	0.0014	-	-
	α_{ij}	0.0001	0.0007	-	-	-	-	-	-	-	0.0009	-	-	0.0005	-
	β_{ij}	0.0022	0.0022	-	-	-	-	-	-	-	0.0039	-	-	0.0065	-
	c_{ij}	4	3	-	2	-	-	-	-	2	3	-	5	3	-
	w_{ij}	3	7	-	10	-	-	-	-	4	2	-	7	4	-
	pdf	H	H	-	E	-	-	-	-	E	H	-	E	H	-

Table 2
GA performance for example

Problem	Weight	cost	Trial 1			2			3			4				
			R	W	C	R	W	C	R	W	C	R	W	C	B	SD
1	100	75	0.8920	99	59	0.8918	91	71	0.8874	99	68	0.8920	99	59	0.8921	0.0022
2	101	75	0.9028	101	71	0.9028	101	63	0.9028	101	63	0.8983	101	60	0.9028	0.0022
3	102	75	0.8866	100	75	0.8901	102	66	0.8830	100	63	0.9066	102	64	0.9066	0.0104
4	103	75	0.9064	103	72	0.9045	103	69	0.9030	103	65	0.9030	103	65	0.9064	0.0016
5	104	75	0.9045	103	69	0.9042	104	63	0.9131	104	65	0.9087	103	64	0.9131	0.0041
6	105	75	0.9062	105	75	0.9019	105	75	0.9084	104	74	0.9085	104	73	0.9085	0.0030
7	106	75	0.9148	106	71	0.8925	106	62	0.9148	106	71	0.9105	106	73	0.9148	0.0106
8	107	75	0.9069	107	65	0.9107	106	61	0.9071	107	62	0.9051	106	65	0.9107	0.0023
9	108	75	0.9218	108	65	0.9175	108	67	0.9022	107	74	0.9155	108	66	0.9218	0.0084
10	109	75	0.9256	109	67	0.9257	109	66	0.9154	108	67	0.9209	109	75	0.9257	0.0048
11	110	75	0.9057	109	67	0.9173	110	67	0.9137	110	65	0.9220	102	67	0.922	0.0068
12	111	75	0.9127	110	75	0.9173	110	67	0.9173	110	67	0.9209	109	75	0.9209	0.0033
13	112	75	0.9109	112	73	0.9062	108	75	0.9064	111	73	0.9164	111	74	0.9164	0.0048
14	113	75	0.9190	112	73	0.9341	113	73	0.9344	113	73	0.9257	113	65	0.9344	0.0073
15	114	75	0.9384	114	68	0.9209	113	74	0.9265	114	69	0.9211	114	73	0.9384	0.0082
16	115	75	0.9164	114	69	0.9362	115	73	0.9384	114	68	0.9242	115	68	0.9384	0.0103
17	116	75	0.9450	116	69	0.9298	115	69	0.9298	115	68	0.9364	116	75	0.945	0.0072
18	117	75	0.9450	116	69	0.9286	117	69	0.9281	116	75	0.9324	116	67	0.945	0.0078
19	118	75	0.9387	117	75	0.9084	117	75	0.9476	118	71	0.9383	118	67	0.9476	0.0171
20	119	75	0.9355	118	73	0.9280	118	75	0.9341	118	74	0.9429	118	70	0.9429	0.0061
21	120	75	0.9496	120	76	0.9497	120	71	0.9496	120	71	0.9383	119	75	0.9497	0.0056
22	121	75	0.9427	120	70	0.9416	121	73	0.9497	120	71	0.9275	115	75	0.9497	0.0093
23	122	75	0.9417	122	64	0.9288	118	75	0.9469	118	75	0.9463	122	62	0.9469	0.0084
24	123	75	0.9414	123	73	0.9447	123	64	0.9605	123	73	0.9499	122	73	0.9605	0.0083
25	124	75	0.9519	123	70	0.9510	124	74	0.9460	124	62	0.9499	122	73	0.9519	0.0025
26	125	75	0.9513	122	74	0.9485	125	61	0.9571	125	64	0.9351	121	75	0.9571	0.0093
27	126	75	0.9611	126	75	0.9455	126	70	0.9544	125	72	0.9525	126	72	0.9611	0.0064
28	127	75	0.9540	126	73	0.9455	126	70	0.9565	127	72	0.9525	126	70	0.9565	0.0047
29	128	75	0.9652	127	75	0.9436	126	73	0.9581	127	74	0.9604	127	74	0.9652	0.0092
30	129	75	0.9582	129	75	0.9964	129	75	0.9495	129	75	0.9563	129	72	0.9964	0.0211
31	130	75	0.9656	130	74	0.9675	130	74	0.9649	129	75	0.9617	129	66	0.9675	0.0024
32	131	75	0.9482	130	74	0.9675	130	74	0.9675	130	74	0.9615	131	66	0.9675	0.0091
33	132	75	0.9640	132	65	0.9581	132	75	0.9640	132	65	0.9675	130	74	0.9675	0.0038

Table 3
SA performance for example

Problem	Weight	cost	Trial 1			2			3			4				
			R	W	C	R	W	C	R	W	C	R	W	C	B	SD
1	100	75	0.8899	100	64	0.8899	100	99	0.8964	99	62	0.8961	99	72	0.8964	0.0036
2	101	75	0.8981	101	72	0.8866	100	75	0.9028	101	63	0.8981	101	72	0.9028	0.0068
3	102	75	0.9028	101	63	0.9028	101	63	0.9028	101	63	0.8948	101	68	0.9028	0.0040
4	103	75	0.9030	130	65	0.9045	103	69	0.9002	103	69	0.9045	103	69	0.9045	0.0020
5	104	75	0.9084	104	74	0.9045	103	69	0.9005	103	59	0.8997	103	75	0.9084	0.0040
6	105	75	0.9022	105	69	0.9069	105	60	0.9084	104	74	0.9084	104	74	0.9084	0.0029
7	106	75	0.9115	106	65	0.9148	106	71	0.9148	106	71	0.9150	106	74	0.915	0.0016
8	107	75	0.9190	107	66	0.9086	107	66	0.9007	107	66	0.9148	106	71	0.9194	0.0079
9	108	75	0.9117	108	65	0.9154	108	67	0.9117	108	65	0.8932	108	75	0.9154	0.0100
10	109	75	0.9019	109	71	0.9218	108	65	0.9046	108	69	0.9108	108	75	0.9218	0.0088
11	110	75	0.9137	110	65	0.9256	109	67	0.9171	110	73	0.9257	102	66	0.9257	0.0060
12	111	75	0.9220	110	67	0.9278	111	66	0.9220	110	67	0.9235	110	71	0.9278	0.0027
13	112	75	0.9323	111	67	0.9238	112	73	0.9194	112	66	0.9238	112	73	0.9320	0.0053
14	113	75	0.9302	113	68	0.9242	113	67	0.9140	112	67	0.9302	113	68	0.9302	0.0076
15	114	75	0.9257	114	67	0.9257	113	74	0.9211	114	73	0.9202	113	74	0.9257	0.0029
16	115	75	0.9319	115	75	0.9369	115	69	0.9153	115	68	0.9369	115	69	0.9369	0.0102
17	116	75	0.9362	115	73	0.9450	116	69	0.9282	116	75	0.9369	115	69	0.945	0.0068
18	117	75	0.9179	116	67	0.9366	117	69	0.9387	117	75	0.9366	117	69	0.9387	0.0097
19	118	75	0.9384	118	72	0.9476	118	71	0.9469	118	75	0.9391	118	68	0.9476	0.0049
20	119	75	0.9275	115	75	0.9391	118	68	0.9310	119	75	0.9211	118	75	0.9391	0.0075
21	120	75	0.9497	120	71	0.9371	117	71	0.9496	120	71	0.9370	120	69	0.9497	0.0072
22	121	75	0.9185	117	75	0.9404	121	75	0.9537	121	72	0.9347	120	69	0.9537	0.0145
23	122	75	0.9303	122	72	0.9393	120	71	0.9513	122	74	0.9499	122	73	0.9513	0.0098
24	123	75	0.9430	122	74	0.9447	123	64	0.9320	123	74	0.9519	123	70	0.9519	0.0082
25	124	75	0.9519	123	70	0.9439	124	72	0.9477	123	71	0.9430	122	74	0.9519	0.0040
26	125	75	0.9497	125	71	0.9305	124	74	0.9332	125	73	0.9453	124	73	0.9497	0.0092
27	126	75	0.9217	126	71	0.9409	125	74	0.9545	126	75	0.9455	126	70	0.9545	0.0138
28	127	75	0.9404	125	74	0.9334	126	73	0.9566	127	72	0.9501	127	73	0.9566	0.0102
29	128	75	0.9444	127	75	0.9620	128	73	0.9541	128	75	0.9511	128	66	0.962	0.0072
30	129	75	0.9613	128	75	0.9652	127	75	0.9613	128	75	0.9607	128	73	0.9652	0.0020
31	130	75	0.9368	130	73	0.9675	130	74	0.9675	130	74	0.9446	129	75	0.9675	0.0157
32	131	75	0.9453	131	75	0.9588	131	74	0.9534	131	65	0.9675	130	74	0.9675	0.0093
33	132	75	0.9675	130	74	0.9640	132	65	0.9640	132	65	0.960	132	65	0.9675	0.0030

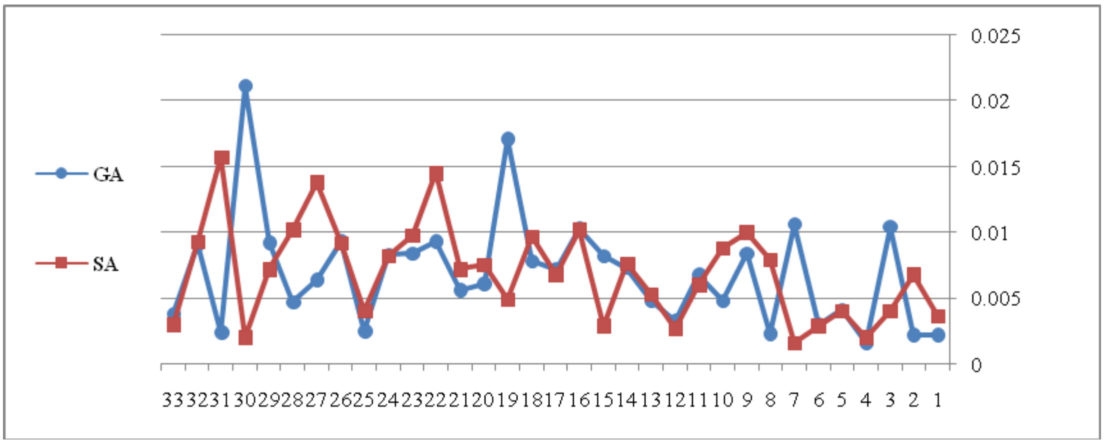

Fig. 7. Standard deviation of the fitness functions for two proposed algorithms.

```
Paired T for GA - SA

             N       Mean     StDev   SE Mean
GA          33   0.937909  0.024248  0.004221
SA          33   0.936139  0.021573  0.003755
Difference  33   0.001770  0.007538  0.001312


95% CI for mean difference: (-0.000903; 0.004443)
T-Test of mean difference = 0 (vs not = 0): T-Value = 1.35  P-Value = 0.187
```

Fig. 8. Result of the paired T-test

5. Conclusion

This paper proposes a new mathematical model for redundancy allocation problem for the series-parallel system with redundancy cold-standby strategy. In the proposed formulation,two typesof time-to-failure including exponential and hypo-exponential are investigated.To solve the model, two metaheuristic algorithms including GA and SA are provided.The computational results indicated that the quality of solutionsof two algorithms is similar. Considering the paired T-test outputs, both algorithms are efficient for this type of reliability optimization problem.

6. References

[1] E. H. L. Aarts, J. H. M. Korst, P. J. M. Laarhoven, Simulated annealing. Local Search in Combinatorial Optimization. Wiley, Chichester, 91–120, 1997.

[2] N. Beji, B. Jarboui, M. Eddaly, H. Chanbchoub, Ahybrid particle swarm optimization algorithm for the redundancy allocation problem. Journal of Computational Science, 1, 159-166, 2010.

[3] R. L. Bulfin, C. Y. Liu, Optimal allocation of redundant components for large systems. IEEE Transactions on Reliability, 34, 241–248, 1985.

[4] V. Cerny, Thermo dynamical approach to the traveling salesman problem: An efficient simulation algorithm. Journal of Optimization Theory and Applications, 45, 41–51, 1985.

[5] M. S. Chern, On the computational complexity of reliability redundancy allocation in a series system. Operation Research Letters, 11,309–315, 1992.

[6] D. W. Coit, Cold-standby redundancy optimization for non repairable systems. IIE Transactions on Reliability, 33(6), 471–479, 2001.

[7] D. W. Coit, L. Liu, System reliability optimization with k-out-of-n subsystems. International Journal on Reliability and Quality Safety Engineering, 7 (2), 129 – 143, 2000.

[8] D. W. Coit, A. E. Smith, Reliability optimization of series–parallel systems using a genetic algorithm. IEEE Transactions on Reliability, 45(2), 254 – 260, 1996.

[9] E. D. Fyffe, W. W. Hines, N. K. Lee, System reliability allocation and a computational algorithm. IEEE Transactions on Reliability, 17,64 – 69, 1968.

[10] S. V. Gurov, L. V. Utkin, Cold-standby systems with imperfect and non-instantaneous switch-over mechanism. Micro Electronics and Reliability, 6(10), 1425-1438, 1993.

[11] J. H. Holland, Adoption in neural and artificial systems. Ann Arbor, Michigan (USA): The University of Michigan Press, 1975.

[12] C. I. Hou, C. Y. Lo, J. H. Leu, Use genetic algorithm in production and inventory strategy. In: Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM) Conference, 963–967, 2007.

[13] K. Ida, M. Gen, T. Yokota, System reliability optimization with several failure modes by genetic algorithm, in Proceedings of the 16th International Conference on Computers and Industrial Engineering, 349–352, 1994.

[14] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by Simulated Annealing Science, 220, 671–680, 1983.

[15] W. Kuo, V. R. PrasadAn annotated overview of system-reliability optimization. IEEE Transactions on Reliability, 49(2), 176–187, 2000.

[16] W. Kuo, V. R. Prasad, F. A. Tillman, C. Hawang, Optimal Reliability Design Fundamental and Application. London: Cambridge University Press, 2001.

[17] Y. C. Liang, Y. C. Chen, Redundancy allocation of series–parallel systems using variable neighborhood search algorithms, Reliability Engineering and System Safety, 92, 323–331, 2007.

[18] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller, Equation of state calculations by fast computing machines. Journal of Chemical Physics, 21, 1087–1092, 1953.

[19] Y. Nakagawa, S. Miyazaki, Surrogate constraints algorithm for reliability optimization problems with two constraints. IEEE Transactions on Reliability, 30(2), 175–180, 1981.

[20] M. Ouzineb, M. Nourelfath, M. Gendreau, Tabu search for the redundancy allocation problem of homogenous series–parallel multi-state systems, Reliability Engineering and System Safety, 93, 1257–1272, 2008.

[21] Y. Ozgur, Penalty function methods for constrained optimization with genetic algorithm. Mathematical Computation and Application, 10, 45–56, 2005.

[22] D. G. Robinson, M. F. Neuts, Standby redundancy in reliability: a review. IEEE Transactions on Reliability, 38(4), 430 – 435, 1989.

[23] B. K. Shankar, M. Gururajan, Two-unit cold-standby system with imperfect repair and excessive availability period. Microelectronics and Reliability, 33(4), 509-512, 1993.

[24] M. Sharifi, M. Hamedanian. M. Naghizadeh, Real Time Redundancy study of a exponential time dependent model. Proceedings of the 2010 international conference on computer design, 46-55, 2010.

[25] R. Tavakkoli- Moghaddam, J. Safari, F. Sassani, Reliability optimization of series-parallel systems with a choice of redundancy strategies using a genetic algorithm, Reliability Engineering and System Safety, 93, 550-556, 2008.

[26] F. A. Tillman, C. L. Hwang, W. Kuo, Optimization techniques for system reliability with redundancy—a review. IEEE Transactions on Reliability, 26(3), 147–155, 1977.

[27] A. Yalaoui, C. Chu, E. Chatelet, Reliability allocation problem in series-parallel system. Reliability Engineering System Safety, 90, 55–61, 2005.