Iterative PSO Algorithms for RAP Problems with $k$-out-of-$n$:G Subsystems and Mixing of Components and Changing $k$ Values

Dongkui Li*, Wulantuya2, Yanlong Zhu2, Xuebao Li2, Liiping Yang2 and Yuqin Xia2

1 Editorial Department of Journal, Baotou Teachers College, Baotou 014030, China
2 Faculty of Information Science and Technology, Baotou Teachers College, Baotou 014030, China
*Email: 165380380@qq.com

Abstract. The subsystem is $k$-out-of-$n$: G configuration, and the subsystem has component mixing (that is, subsystem components can be selected from different kinds of components), and the minimum number of normal working elements ($k$ value) that can be selected as well as can change RAP problem. On the basis of giving a solution construction method and generating a new solution by symmetric bit transformation technology, an iterative DPSO (discrete particle swarm optimization) algorithm with fixed-compression coefficient and dynamic inertia weight is designed to solve the model. The calculate results indicate that the algorithm can effectively give the optimal solution for each test case (which is consistent with the existing optimal results in the literature). PSO algorithm can effectively solve the RAP problem of subsystems with mixed components and varying $k$ values. When the size of subsystems ($k$-out-of-$n$: G) is large, and the solution of the model is limited by microcomputers.

1. Introduction
The reliability redundancy allocation problem (RAP) can be divided into the reliability allocation problem with only the reliability change of elements, the reliability redundancy allocation problem with the change of redundancy number and the reliability redundancy allocation problem with the change of reliability and redundancy number. According to whether the subsystem allows component mixing, it can be divided into RAP problem which allows component mixing and RAP problems which does not allow component mixing. The general reliability redundancy allocation problem is NP-hard. The traditional mathematical methods, for instance Integer Programming, Dynamic Programming and Substitution Constraint Method, are often restricted. The GA (genetic algorithm), ACO (ant colony) algorithm, PSO (particle swarm optimization) algorithm and other meta-heuristic algorithms are paid attention to solve this problem. Therefore, it is of theoretical and practical significance to study and solve the RAP problem of a specific model [1-2].

In the existing literature on RAP, most of the system models are series-parallel systems or generalized redundancy allocation problem with complex network structure (GRAP) [3-31], and most of the subsystems are parallel structures, that is, the case of $k = 1$ in $k$-out-of-$n$: G structure. There are few studies on $k$-out-of-$n$: G structure and $k > 1$ [6, 32-33]. This is because when the subsystem is $k$-out-of-$n$: G structure, especially when the value of $n$ and $k$ is relatively large, the calculation difficulty of subsystem reliability is increased, and the general microcomputer algorithm is not very effective.
[34-37], so the research on reliability optimization becomes very complex. The scale of the subsystem studied in this paper is small, and it is a k-out-of-n: G system (k > 1), and MPSO (modified particle swarm optimization) algorithm is applied to solve the RAP optimization problem with component mixing and varying k value.

2. Assumptions and Models

2.1. Assumptions
(1) Components and systems are two states, i.e. normal operation state and failure state; (2) the properties of components, i.e. reliability, price and weight as well as volume etc., are known and determined; (3) different kinds of elements are allowed to be mixed in a subsystem; (4) Components are not repairable; failure of components does not damage the system; (5) The failure of components is statistically independent.

2.2. Model
The system composed of s k; out of n; G subsystems S, i = 1, 2,..., s; each subsystem has m; types of elements. The number of each element as well as the k; value of subsystem are selected to maximize the reliability of the whole system under the constraints of system weight and cost.

The mathematical expression of the model is as follows:

\[
R = \prod_{i=1}^{s} R_i = \prod_{i=1}^{s} \prod_{i=1}^{m_i} \frac{1}{\prod_{j=1}^{n_i} \prod_{i=1}^{m_j} (1 - y_{ij})^{X_{ij}} y_{ij}^{x_{ij}}}
\]

\[
\text{max } R
\]

s.t. \( \sum_{i=1}^{s} \sum_{j=1}^{m_i} c_{ij} x_{ij} \leq C_0 \) (3)

\( \sum_{i=1}^{s} x_{ij} = n_i \) (4)

\( \sum_{j=1}^{m_i} x_{ij} = n_i \) (5)

\( \prod_{j=1}^{m_i} y_{ij} = l, 0 \leq y_{ij} \leq x_{ij} \) (6)

\( 1 \leq k_i^1 \leq k_i \leq k_i^u \leq n_i \) (7)

\( \sum_{i=1}^{s} g_i (k_i) \leq b \) (8)

\( x_{ij} \in \{0,1,2,3,...\}, k_i \in \{1,2,3,...\} \) (9)

Where R is the system reliability, which is related to vectors X and K, and can also be written as R(X,K); R0, Ci, and Wj respectively indicate the reliability, price and weight of element j of subsystem i; Ci and Wj are the cost and weight constraints of the system, and ni is the maximum number of components allowed to be used in subsystem i; for constraint (8), for simplifying the problem, formula (10) is taken here:

\( k_1^2 + k_2^2 + ... + k_s^2 \leq b \) (10)

When calculating the reliability of k; out of n; G subsystem, select one of the formulas (1) or (2) according to the value of k; and the calculation complexity will be reduced.

3. Algorithm

3.1. Structure of Solution
The computing speed of PSO (particle swarm optimization) largely depends on the size of the search space of the solution, that is, it depends on the encoding of the solution; a good coding method can effectively shrink the size of the search space of the solution.

The coding method is as follows:
A particle (solution) is a row vector, which is composed of two parts. One part is the selection number of each element in the subsystem, the variable is placed on the left side of the vector, the other part is the minimum number of working elements of the subsystem, which is placed on the right
side of the row vector, which has the format $[X, K]$. $X$ sequence is composed of redundant variables of each elements in each subsystem, and $K$ is composed of the minimum number of normal working units of each subsystem.

For example (see test example 4.1 for specific data): a particle (solution) $[0 3 0 2 0 0 0 0 3 1 2 1 2 1 1 2]$, it means that: the first subsystem selects 3 kind 2 components, the second subsystem selects 2 kind 1 components, the third subsystem selects 3 kind 4 components, the fourth subsystem selects 1 kind 1 component, 2 kind 2 components, and 1 kind 3 component, and the minimum number of normal operation elements of the four subsystems are 2, 1, 1 and 2.

3.2. New Solution Generation Algorithm

In particle swarm optimization algorithm, a new solution is produced from the known solution $[X_0, K_0]$. According to the coding rules, it needs to be implemented in two parts:

Algorithm 1

Step 0 gives the initial solution $X_0$.

Step 1 for the $i$th bit of the initial solution, if the randomly generated random number $\geq 0.5$, add 0.5 to the $i$th bit of the initial solution and round it; otherwise, subtract 0.5 from the symmetric bit of the $i$th bit of the initial solution and round it.

Step 2 outputs a new solution.

Step 3 algorithm is terminated.

For the minimum number of working components of the subsystem, the new solution algorithm is as follows:

Algorithm 2

Step 0 gives the initial solution $K_0$.

Step 1 for the $i$th bit of the initial solution, if the randomly generated random number $\geq 0.5$, add 1 to $i$th bit of the initial solution; otherwise, subtract 1 from the symmetric bit of the $i$th bit of the initial solution.

Step 2 outputs a new solution.

Step 3 algorithm is terminated.

3.3. Fitness Function

In order to convert the constrained optimization problem as the unconstrained optimization problem, the fitness function is introduced as follows:

$$
\max f(X, K) = \begin{cases} 
R(X, K) & \text{the cost of system components } TC \leq C_0 \land \text{the weight } TW \leq W_0 \\
R(X, K) \ast \min \left( \left(\frac{C_0}{TC}\right)^{\alpha}, \left(\frac{W_0}{TW}\right)^{\beta} \right), & \text{otherwise}
\end{cases}
$$

(11)

3.4. Algorithm

3.4.1. Standard Particle Swarm Optimization Algorithm (PSO). Please refer to the description of the literature [38].

3.4.2. Iterative PSO Algorithms (MDPSO). Principle: for the sake of solve the above optimization problem (model), the PSO algorithm is designed as follows. When the particle velocity and position function are generated, the redundant number and the minimum $k$ value of the subsystem are iterated respectively. After updating the speed and position, the particle velocity and position are updated again for the particles that do not meet the constraint conditions, which is called the modified basic PSO algorithm. The method is a two-stage iterative PSO algorithm. The details are as follows:

Algorithm 3 (Pseudo code of Matlab) [39]

Step 0 (to initialize the system) gives the initial solution of the system $[X_0, K_0]$ ($X_0$ represents the number of initial redundancy of each elements in each subsystem, as well as $K_0$ represents the initial minimum number of normal working elements in each subsystem). $N= [n_1, n_2, ..., n_s]$, each component
sequence represents the number of normal working components of each subsystem. \( M = [m_1, m_2, \ldots, m_r] \), the each component in the \( M \) represents the number of kinds of units that can be chosen by each subsystem. The vector \( P \) represents the reliability of each component of each subsystem, and the vector \( C \) and \( W \) represent the price and weight of each component of each subsystem.

The compression coefficient \( c_1 = c_2 = 1.4962; \ b = b_0 \).

Let \( V = \text{zeros} (|X_0| + |K_0|, n_c) \), where \( |X_0| (|K_0|) \) is the cardinal number of \( X_0 \) (\( K_0 \)), the \( n_c \) is the particles number in the population.

Let \( A = B = \text{zeros} (|X_0| + |K_0|, n_c) \), \( CA = CB = \text{zeros} (1, n_c) \) store the fitness values of particles (solutions) in population \( A \) and \( B \) respectively, and each particle of \( B \) is the local optimal solution corresponding to particles in \( A \).

\[ E = X_0, \ \text{EK} = K_0, \ E, \ \text{EK} \] store the overall optimal value of \( X \) and \( K \).

Step 1 according to the new solution generation algorithm 1-2, \( 2^*n_c \) particles (solutions) satisfying the constraint conditions are generated. They are stored in matrix \( A \) and \( B \) respectively, and the fitness values calculated according to formula (11) are stored in \( CA \) and \( CB \) respectively.

Step2 for \( t = 1: n_c \) (\( n_c \) is the maximum number of cycles).

The dynamic inertia weight \( w = 0.9-0.5 \times \frac{(r-1)}{(n_c-1)} \), to calculate the cost \( TC \) and weight \( TW \) of the optimal solution, and the fitness value \( e \) of the optimal solution.

Step 2.1 for each particle in population \( A \), if the corresponding adaptive value is greater than the local optimal value, the value of particles in \( A \) is assigned to the corresponding particles in \( B \); if the adaptive value of particles in \( B \) is greater than the global optimal value \( e \), the values of particles in \( B \) are assigned to \( E \) and \( \text{EK} \).

Step 2.2 for each particle in population \( A \), update the speed and position according to the following formulas (12) - (17) (segmentation):

\begin{equation}
V(i,j)=w\times V(i,j)+c_1 \times \text{rand} \times (B(i,j)-A(i,j)) + c_2 \times \text{rand} \times (E(1,i)-A(i,j))
\end{equation}

(12)

\[ A(i,j)=\text{round}(A(i,j)+V(i,j)) \]

(13)

if \( A(i,j) < 1 \), then \( A(i,j) = 0 \)

(14)

\begin{equation}
V(i,j)=w\times V(i,j)+c_1 \times \text{rand} \times (B(i,j)-A(i,j)) + c_2 \times \text{rand} \times (EK(1,i)-X_0 \times A(i,j))
\end{equation}

(15)

\[ A(i,j)=\text{round}(A(i,j)+V(i,j)) \]

(16)

if \( A(i,j) < 1 \), then \( A(i,j) = 0 \)

(17)

Step 2.3 for particles that do not meet the constraint conditions, the equations (12) - (17) are applied again to update the velocity and position until the constraints are met.

Step 2.4 updated the fitness values \( CA \) and \( CB \) of \( A \) and \( B \).

Step 3 outputs the optimal solution (\( E, \ EK, \ ER, \ TC, \ TW \)), and draws the iterative curve.

Step4 End.

4. Simulation

The Simulation is carried out on microcomputer that is configured with an double Intel (R) core (TM) CPU i5-6500@3.20Ghz 3.20Ghz, it’s memory is 8GB and HD is 600GB; the OS is wins10P; the programming system software is Matlab R2015b.

4.1. Simplified FYFFE Problem (SFYFFE)

For the SFYFFE [3] problem with four subsystems, the known subsystem element kinds and elements reliability, price as well as weight data are listed in Table 1. The known system price constraint is no more than 40 units, and the weight constraint is no more than 70 units.
Table 1. SFYFFE problem known data 

| Subsystem | Selectable component types |
|-----------|---------------------------|
|           | r   | c   | w   | r   | c   | w   | r   | c   | w   | r   | c   | w   |
| 1         | 0.90| 1   | 0.93| 1   | 0.91| 2   | 0.95| 2   | 5   |
| 2         | 0.95| 2   | 0.94| 1   | 0.93| 1   | 9   | 4   | 4   |
| 3         | 0.85| 2   | 0.90| 3   | 0.87| 1   | 6   | 0.92| 4   | 4   |
| 4         | 0.83| 3   | 0.87| 4   | 0.85| 5   | 4   | 5   | 5   |

* Indicates that the data does not exist

The parameters of the algorithm are set to as follows: $n = 4, c_1 = c_2 = 1.4962; C_0 = 40, W_0 = 70, n_c = 100, n_t = 1000, \alpha = \beta = 2; P = [0.9, 0.93, 0.91, 0.95, 0.95, 0.94, 0.93, 0.85, 0.90, 0.87, 0.82, 0.83, 0.87, 0.85]; C = [1, 1.2, 2, 2.2, 2, 1, 2, 3, 1.4, 3, 4, 5]; W = [3, 4, 2, 5, 8, 10, 9, 7, 6, 4, 5, 6, 4];$ take the initial solution $X_0 = [0.9, 0.93, 0.91, 0.95, 0.95, 0.94, 0.93, 0.85, 0.90, 0.87, 0.82, 0.83, 0.87, 0.85]; K_0 = [1, 1, 1, 1]; M = [4, 3, 4, 3].$ The algorithm is executed 10 times, and the optimal solution is listed in Table 2. If the system and subsystem scale is small, the execution’s efficiency of the algorithm is very high, and the convergence curve is also very good, as shown in Figures 1-3.

Table 2. The optimal solution of the algorithm is executed for 10 times.

| TC | TW | N  | Optimal solution | Reliability of optimal solution | Cost of optimal solution | Weight of optimal solution | K   | b  |
|----|----|----|-----------------|-------------------------------|-------------------------|----------------------------|-----|----|
| 40 | 70 | 3,2,3,4 | 030020000003121 | 0.86429                       | 35                       | 61                         | 2,2,2,2 | 1  |
| 40 | 70 | 3,2,3,4 | 030020000003121 | 0.95527                       | 35                       | 61                         | 2,1,2,2 | 2  |
| 40 | 70 | 3,2,3,4 | 030020000003121 | 0.97246                       | 35                       | 61                         | 2,1,1,2 | 3  |
| 40 | 70 | 3,2,3,3 | 030020000003030 | 0.99622                       | 35                       | 61                         | 1,1,1,1 | 4  |
| 40 | 70 | 3,2,3,3 | 030020000003030 | 0.83322                       | 31                       | 58                         | 2,2,2,2 | 1  |
| 40 | 70 | 3,2,3,3 | 030020000003030 | 0.92093                       | 31                       | 58                         | 2,1,2,2 | 2  |
| 40 | 70 | 3,2,3,3 | 030020000003030 | 0.96352                       | 31                       | 58                         | 2,1,2,1 | 3  |
| 40 | 70 | 3,2,3,3 | 030020000003030 | 0.99446                       | 31                       | 58                         | 1,1,1,1 | 4  |
| 40 | 70 | 3,3,3,3 | 030011100111021 | 0.89909                       | 28                       | 70                         | 2,2,2,2 | 1  |
| 40 | 70 | 3,3,3,3 | 030011100111021 | 0.94484                       | 28                       | 70                         | 2,2,2,2 | 2  |
| 40 | 70 | 3,3,3,3 | 030011100111021 | 0.97236                       | 28                       | 70                         | 2,2,1,1 | 3  |
| 40 | 70 | 3,3,3,3 | 030011100111021 | 0.99588                       | 28                       | 70                         | 1,1,1,1 | 4  |
Figure 1. SFYFFE problem, when \( n = [3, 2, 3, 4] \), \( b = 1, 2, 3, 4 \), the optimal solution convergence curve of the algorithm. The number of iterations is the \( x \)-axis, the \( y \)-axis means reliability.

Figure 2. SFYFFE problem, when \( n = [3, 2, 3, 3] \), \( b = 1, 2, 3, 4 \), the optimal solution convergence curve of the algorithm. The number of iterations is the \( x \)-axis, the \( y \)-axis means reliability.
4.2. Generalized FYFFE Problem (GFYFFE)

The GFYFFE [3] problem with 14 subsystems, the type and reliability, price and weight data of subsystem components, please refer to Table 1 of Ref. [33]. It is known that the system price constraint is no more than 130 units, and the weight constraint range is 170 to 205 units. Because the algorithm takes too long to execute (especially when \( b = 5 \), the microcomputer usually needs to run for more than 24 hours), we take the best results of random execution one time of the algorithm as shown in Table 3-4. We can see from Tables 3-4, when \( b = 14 \), the optimal solution is consistent with that obtained by ONISHI J. et al. [11], and SOOKTIP T’ results obtained by genetic algorithm [33] are inconsistent, the optimal reliability of their results is lower.

Table 3. The best solution, reliability and \( N \) value corresponding to different \( b \) values when the algorithm is executed once at random.

| TC  | TW  | \( N \)                     | Optimal solution       | Reliability of optimal solution | \( b \) | Population particle number | Population particle number and number of iterations |
|-----|-----|-----------------------------|------------------------|--------------------------------|-------|---------------------------|-----------------------------------------------------|
| 130 | 205 | 3,3,3,4,3,2,3,4,2,2          | 0030 300 0003 202 030 0200 300 400 1100 003 300 400 020 0011 | 0.991182 | 14   | 100, 1000                 |                                                     |
|     |     |                             | 0030 300 0003 202 030 0100 300 400 1100 003 300 400 020 0011 | 0.954942 | 12   | 10, 300                  |                                                     |
| 130 | 205 |                             | 0030 300 0003 202 030 0101 300 400 1100 003 300 400 020 0020 | 0.893897 | 10   | 10, 300                  |                                                     |
Table 4. The algorithm is executed once randomly, and different $b$ values correspond to the $k$ value, the cost as well as weight of the optimal solution.

| $TC$ | $TW$ | Reliability of optimal solution | Cost of optimal solution | Weight of optimal solution | $K$ | $b$ | Population particle number and number of iterations |
|------|------|----------------------------------|--------------------------|----------------------------|-----|-----|-----------------------------------------------|
| 130  | 204  | 3.3,3,4,3,2,3,4,2,3,4,2,2        | 0.808987                 | 7                          | 10,  | 300  |
|      |      |                                   |                          |                            |      |      |                                                                 |
| 130  | 202  | 3.2,3,4,3,2,3,4,2,3,5,2,2        | 0.990548                 | 14                         | 100, | 1000 |
|      |      |                                   |                          |                            |      |      |                                                                 |
| 130  | 201  | 3.2,3,4,3,2,3,4,2,3,5,2,2        | 0.990350                 | 14                         | 100, | 1000 |
|      |      |                                   |                          |                            |      |      |                                                                 |
| 130  | 170  | 3.2,3,3,3,2,2,3,2,3,4,2,2        | 0.970760                 | 14                         | 5,   | 300  |
|      |      |                                   |                          |                            |      |      |                                                                 |
| 130  | 205  | 0.991182                          |                          |                            |      |      |                                                                 |
|      |      | 130 205                           | 0.954942                 | 10                         | 10,  | 300  |
|      |      |                                   |                          |                            |      |      |                                                                 |
|      |      | 0.893897                          | 126 205                 | 12                         | 10,  | 300  |
|      |      |                                   |                          |                            |      |      |                                                                 |
|      |      | 0.808987                          | 130 205                 | 7                          | 10,  | 300  |
|      |      |                                   |                          |                            |      |      |                                                                 |
|      |      | 0.746729                          | 130 203                 | 5                          | 5,   | 300  |
|      |      |                                   |                          |                            |      |      |                                                                 |
4.3. Comparison and Discussion

The comparison results of the raised algorithm (MDPSO) with ISC [11] and GA [33] are listed in Table 5. Through the observation Table 5, it is found that: in the given model test example, the improved surrogate constraint method only provides the calculation results of $b = 14$, and the optimal solution obtained is the best at present, and the CPU time is also small; GA [33] algorithm provides the reliability of the system optimal solution under four conditions of $b = 14, 12, 7, 5$, and does not specify the specific time; MDPSO algorithm provides $b = 14, 12, 7$, and, 5. The reliability of the optimal solution in four cases is obviously better than that provided by GA [33], and in the case of $b = 14$, is consistent with the reliability given by ISC method is achieved.

| TC  | TW  | Optimal solution reliability of ISC method[11] | Optimal solution reliability of GA[33] | Optimal solution of MDPSO reliability | Cost | Weight | b  |
|-----|-----|---------------------------------------------|----------------------------------------|----------------------------------------|------|--------|----|
| 0.990548 | 130 204 | 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1 | 0.991182 | 0.9892 | 0.991182 | 130 205 | 14 |
| 0.953069 | 130 204 | 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1 | 0.991182 | 0.9892 | 0.991182 | 130 205 | 14 |
| 0.905034 | 130 204 | 1,2,2,2,2,2,1,1,2,2,1,1,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.803525 | 130 204 | 2,2,2,2,2,2,2,1,2,2,2,1,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.747070 | 130 204 | 2,2,2,2,2,2,2,2,2,2,2,2,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.990548 | 130 204 | 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.953069 | 130 204 | 1,1,1,1,1,1,1,1,1,1,1,1,1,1,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.905034 | 130 204 | 1,2,2,2,2,2,1,1,2,2,1,1,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.803525 | 130 204 | 2,2,2,2,2,2,2,1,2,2,2,1,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |
| 0.747070 | 130 204 | 2,2,2,2,2,2,2,2,2,2,2,2,1 | 0.9892 | 0.965 | 0.954942 | 126 205 | 12 |

Table 5. Comparison of calculation results.

- TC: Test Case
- TW: Test Weight
- ISC: Improved Surrogate Constraint Method
- GA: Genetic Algorithm
- MDPSO: Modified Differential Evolution PSO
5. Conclusion
We studied the MDPSO algorithm for RAP problem for the series system composed of $k_i$ out of $n_i$: G subsystem, which allows the components to mix and has variable $k$ value. When the scale of the subsystem is small and the $k$ value is in a reasonable interval, the effectiveness of the design algorithm is verified by test cases. It is found that when the scale of subsystem becomes larger, such as more than 10 components, more than 2 subsystems, and $k$ value is not 1, it will become very complex to calculate the RAP problem of the system on a microcomputer, and more efficient algorithms or different coding techniques are needed to deal with the reliability of the system, and the related problems are worthy of further study.

Acknowledgments
This research was financially supported by the Inner Mongolia Natural Science Foundation Project (2018MS06031), Inner Mongolia University Science Research Project (NJZY17290).

References
[1] Coit D W and ZIO E 2018 Reliab. Eng. Syst. Safe. doi:https://doi.org/10.1016/j.jress.2018.09.008
[2] Mitsuo G and Young S Y 2006 Reliab. Eng. Syst. Safe. 91 1008-26
[3] Fyffe D E, Hines W W and Lee N K 1968 IEEE Trans. Reliab. 17 64-9
[4] Nakagawa Y and Miyazki S 1981 IEEE Trans. Reliab. 30 175-80
[5] Misra K B and Sharma 1991 IEEE Trans. Reliab. 40 81-91
[6] Coit D W and Smith A E 1996 IEEE Trans. Reliab. 45 254-60
[7] Xu Z J, Ma C W, Mei Q Z and Xi S R 1998 J. Tsinghua Univ. (Technol.& Sci.) 38 54-7
[8] Zh T Z, Teng C X and Han Z G 2002 Contr. Deci. 17 378-84
[9] Liang Y C 2004 IEEE Trans. Reliab. 53 417-23
[10] LIANG Y C and WU C C 2005 IEMS 4 94-101
[11] Onishi J, Kimura S, James R J W and Nakagawa Y 2007 IEEE Trans. Reliab. 56 94-101
[12] Cheng S J, Lu W, He P 2009 Comput. Eng. Appl. 45 64-6
[13] Coelho L D S 2009 Reliab. Eng. Syst. Safe. 94 830-7
[14] Mohamed O, Mustapha N and Michel G 2009 CIRRELT-2009-05
[15] Chen Q and Ma X 2010 Modu. Mach. Tool Auto. Manu. Techn. 49-51+55
[16] Chen S J, Lu W and He P 2010 J. Mech. Syst. 32 64-7
[17] Phakhapong T, Worawat S, Apinan A and Pothisa S 2013 Int. J. Comput. Inf. Sci. Eng. 7 130-5
[18] Sahoo L, Banerjee A, Bhunia A K and Chattopadhyay S 2014 Swarm Evolu. Comput. 9 43-51
[19] Pant S, Anand D, Kishor A, Singh S B 2015 Int. J. Perfo. Eng. 11 33-42

* Indicates that the data does not exist
[20] Yeh W C 2017 Inf. Sci. 408 182-97
[21] Gholinezhad H and Hamadani A Z 2017 Reliab. Eng.Syst. Safe. 164 66-73
[22] Chang K H and Kuo P Y 2018 Eur. J. Oper. Res. 265 1094 -101
[23] Yeh W C 2009 Expert Syst. Appl. 36 9192-200
[24] Chern M S 1992 Oper. Res. Lett. 11 309-15
[25] Walter G, Pflug C and Andrzej R 1993 Working Paper WP-93-6
[26] Bellman R E, Dreyus E 1958 Oper. Res. 6 200- 6
[27] Beji N, Jarboui B, Eddaly M, Chabchoub H 2010 J. Comput. Sci. 1 159-67
[28] Jahromi A E and Feizabadi M 2017 Comput. Ind. Eng. 108 111-23
[29] Li D K 2018 Modern Electron. Technol. 41 89-92
[30] Li D K and Dong H 2016 Yinshan Acad. J. (Natu. Sci. Edit.) 30 35-41
[31] Li D K, Wulantuya, Zhu Y L 2015 Electron. Commer. 188 58-60+74
[32] Coit D W and Liu J C 2000 Int. J. Reliab., Qual. Safe. Eng. 7 129-42
[33] Tipwimol S,Naruemon W and Coit D W 2011 9th Int. Conf. Reliab., Main. Saf. ( IEEE ) 1382-7
[34] Li D K and Lu F 2018 Yinshan Acad. J. (Natu. Sci. Edit.) 32 49-52
[35] Li D K 2010 Comput. Simu. 27 125-8
[36] Li D K 2010 Comput. Simu. 27 362-5
[37] Li D K 2009 Commun. Technol. 42 149-51
[38] Wulantuya,Li D K,Li X B and Yang L P 2012 Commun. Comput. Inf.Sci.345 582-591
[39] Li D K 2021 Adv. Intell. Syst.Comput. 1143 23-33