AN EFFICIENT HEURISTIC ALGORITHM FOR DETERMINING OPTIMAL REDUNDANCY ALLOCATION OF COMPLEX NETWORKS

Kumar Pardeep * and Chaturvedi D.K. **

•

Institute of Instrumentation Engineering,
Kurukshetra University, Kurukshetra, – 136 119, India
Fax: +91 1744 - 238 277, 238 191, 238 035.
E-mail: pardeep_kuk@rediffmail.com*, dhiraj_chaturvedi@yahoo.co.in**
*Corresponding author

Pahuja G.L.

•

Department of Electrical Engineering,
National Institute of Technology, Kurukshetra – 136 119, India
Fax: +91 1744 238 050, E-mail: pahuja_gl@yahoo.co.in

ABSTRACT

The paper presents a new heuristic algorithm for determining optimal redundancy allocation of complex networks. The present algorithm is an iterative method; all the paths of the network are first arranged in decreasing order of their priority determined using a path sensitivity factor, the highest priority path is optimized first by adding redundant components for subsystems of the path iteratively based on proposed subsystem selection factor. In case of availability of any residual resources next lower priority paths are considered for redundancy allocation. The proposed algorithm not only demonstrates improved performance in comparison with most of the existing heuristic algorithms but also leaves minimum slack of components without any further possibility of redundancy.

Keywords: constrained redundancy optimization; complex networks; heuristic algorithm.

1. INTRODUCTION

The problem of redundancy allocation generally has been solved as a single objective optimization problem to maximize system reliability subject to several constraints such as cost, weight, volume, etc. The solutions of such optimization problems have been obtained using mathematical models like dynamic programming [1-3], heuristic methods [4-23] and meta-heuristics such as genetic algorithms [24-26], tabu search [27], ant colony optimization [28], etc.

In recent works, major focus is on the development of heuristic and meta-heuristic algorithms for redundancy allocation problems to improve system reliability [4, 29]. Many heuristic algorithms have been proposed in the literature for solving redundancy reliability optimization problems which search for the solutions not only in feasible regions but also do excursion in infeasible/bounded infeasible regions for finding possibly improved solutions [30-

32]. For example, Shi method [19] searches solution only in feasible region and shows good temporal efficiency. The Nakagawa –Nakashima (N-N) method [15] also search solution in the feasible region and is superior to other heuristics in terms of optimality rates and relative errors, but have poor temporal efficiency [4,29]. The Agarwal - Gupta (A-G) algorithm [32] is the one of the recent methods for solving redundancy optimization of complex combinatorial problems and allows the search for optimal solution not only in the feasible region but also into the bounded infeasible region. Recently, Kumar *et al.* [22,23] presented heuristic algorithms which shows better performance.

The purpose of this paper is to present an efficient heuristic algorithm for determining optimal redundancy allocation of complex systems. The proposed algorithm [P-Alg] consists of: arranging all the path sets of the network in the decreasing order of path sensitivity factor value and then selecting the highest priority path set for redundancy allocation. A redundant parallel subsystem is added to the unsaturated subsystem of the chosen path set having maximum value of subsystem selection factor, if no constraints are violated. In case of violation of any constraint, the subsystem is excluded from further consideration and the next subsystem of the path set having highest value of subsystem selection factor is considered for redundancy allocation. The path is removed only after exhausting all the subsystems of the path set. The proposed algorithm not only demonstrates improved performance in comparison with most of the existing heuristic algorithms but it also leaves minimum slack of components without any further possibility of redundancy. The P-Alg not only utilizes a different formulation for subsystem selection factor but it also differs significantly from Kumar [22] the way the path sets are removed from further consideration in case of violation of any constraints.

The computational experiments are conducted on 4-, 5-, 7-, 10- and 15-unit complex systems with linear constraints. The numerical results obtained with P-Alg, Kumar [22], Shi [19], N-N [15] and A-G [31] methods are compared in terms of performance measures like average relative error (*A*), maximum relative error (*M*), optimality rate (*O*) and average execution time (*T*) [22, 30,31].

2. PROBLEM FORMULATION

2.1 Assumptions:

1. There are n subsystems in the system.
2. The system and subsystems are coherent. The subsystem structure is not restricted.
3. Subsystem states are mutually and statistically independent.
4. Constraints are separable and additive among components. Each constraint is an increasing function of x_i for the subsystem.
5. Redundant components can not cross subsystem boundaries.

2.2 Problem Definition:

A complex system consists of several components connected to each other neither in series nor in parallel. Figs. 1-5 show 5 such complex systems for 4-, 5-, 7-, 10- & 15- unit networks respectively. The problem of constrained redundancy optimization can be reduced to the following integer programming problem:

$$\text{Maximize: } R_s(x) \tag{1}$$

$$\text{Subject to: } \sum_{i=1}^{n} g_i^j(x_i) \le C_j, \quad j = 1,2,....,k$$

$x_i \geq 1$, for $i = 1, 2, \ldots$, n.

3. PROBLEM FORMULATION

3.1 Algorithm

Assuming that the system reliability expression $R_s(x)$ is known (Shi [1]), the proposed algorithm uses following steps for finding the solution.

First of all, unsaturated minimal path sets of the system are arranged in their decreasing order of priority using path sensitivity factor $a_l(x)$

$$a_l(X) = \frac{\prod_{i \in P_l} R_i(x_i)}{\sum_{i \in P_l} \sum_{j=1}^{k} (g_i^j(x_i)/k\,C_j)}, \qquad l = 1,2,..,m \tag{2}$$

From the above ordered set of minimal path sets, path set having highest sensitivity factor- $a_l(X)$ is considered for optimization and subsystem selection factor $b_i(x_i)$ of all the unsaturated subsystem is found using

$$b_i(x_i) = \frac{\Delta R_i}{\sum_{j=1}^{k} (g_i^j(x_i)/k\,C_j)}, \quad \text{for each } i \in P_l \tag{3}$$

where $\Delta R_i = R_i(x_i) - R_i(x_i - 1)$

ΔR_i is termed as increment in subsystem reliability when a unit is added in parallel to the subsystem. After finding the subsystem having highest value of subsystem selection factor $b_i(x_i)$ of the chosen path set, a redundant parallel component is added to the unsaturated subsystem if no constraint is violated. In case of violation of any constraint the subsystem is removed from further consideration for redundancy and the next subsystem of the path set having highest value of subsystem selection factor $b_i(x_i)$ is considered for redundancy allocation. The iteration continues either till all the subsystems are removed from further consideration or all the resources are consumed. If all the resources are exactly consumed the iteration stops giving the optimal solution. But in case of all the subsystems of the chosen path set are removed from further consideration and there are still some resources available, the minimal path set having next highest value of sensitivity factor $a_l(X)$ is considered for optimization and then the steps are repeated till optimal solution is reached.

3.2 Steps of the Proposed Algorithm

Step1: Find all minimal path sets P$_l$ (for all l = 1,2,…,m.) of the system using any method.

Step2: Let $x_i = 1$ for all i; $i = 1, 2, \ldots, n$ such that $X = (1, 1, \ldots, 1)$

Step3: All unsaturated minimal path sets P$_l$ (for all l = 1,2,…,m.) of the system are arranged in their decreasing order of priority Ps using sensitivity factor $a_l(X)$.

Step4: The path set $[P_{Ps(Q)}]_{(Q=1)}$ is selected for optimization.

Step5: For the above chosen path set, calculate $b_i(x_i)$ for all the subsystems of the path set and find i^*such that $b_{i*}(x_i) = max\,[b_i(x_i)]$.

Step6: Check, by adding one redundant unit to unsaturated subsystem i^*:

i. if no constraints are violated, add one redundant subsystem to unsaturated subsystem i^* by replacing x_{i^*} with x_{i^*}+1 and go to step 5.

ii. if at least one constraint is exactly satisfied and others are not violated, then add one redundant subsystem to unsaturated subsystem i^* by replacing x_{i^*} with x_{i^*}+1.

The $x^* = x$ is the optimal solution. Go to step 7.

iii. if at least one constraint is violated then exclude the subsystem i^* from further consideration and the next subsystem i^* of the same path set having highest value $b_{i^*}(x_i)$ is considered for redundancy allocation, go to step 6.

iv. if at least one constraint is violated and all other subsystems of the path $[P_{Ps(Q)}]$ are excluded; exclude the path $[P_{Ps(Q)}]$ from further consideration; consider the next path $[P_{Ps(Q+1)}]$ having next highest value of $a_l(X)$ and go to step 5.

v. if all the subsystems and/or all the minimal path sets are excluded from further consideration, than $x^* = X$ is the optimal solution; go to step 7.

Step7: Calculate the system reliability, $R_s(x^*)$.

4. COMPUTATION AND RESULT

The redundancy allocation problem for complex systems is formulated with the objective of maximization of the system reliability under constraint environment. In this paper, the test problems of computational experiments are generated for 4-, 5-, 7-, 10- and 15- unit complex networks shown in Figs. 1-5 respectively. As an illustration of P-Alg method two sets of bench mark examples consisting of 4- unit composite network (Fig. 1) and 5-unit bridge network (Fig. 2) with linear constraints are considered. The same examples are also solved by Kumar, Shi, N-N and A-G methods for comparison.

Example 1: 4-unit composite network (Fig. 1) with linear constraints (n = 4 and k = 2). The constraint redundancy optimization problem is expressed as the following integer programming problem:

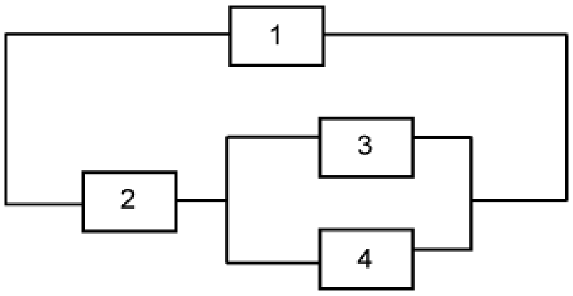

Figure 1. 4-Unit Composite Network

$$\text{Max.}\,R_s(x) = \begin{array}{l} R_1(x_1)+Q_1(x_1)R_2(x_2)R_4(x_4) \\ +Q_1(x_1)R_2(x_2)R_3(x_3)Q_4(x_4) \end{array} \tag{4}$$

$$\text{Subject to: } \sum_{i=1}^{4} c_{1i}(x_i) \le 132$$

$$\sum_{i=1}^{4} c_{2i}(x_i) \le 341 \tag{5}$$

$$x_i \ge 1, \text{ for i = 1, 2, 3, 4.}$$

$$L(x) = (1, 1, 1, 1), \quad U(x) = (6, 1, 13, 4)$$

The problem is solved for P-Alg, Kumar, Shi, N-N and A-G methods with the randomly generated values of parameters r_i, c_{1i}, c_{2i}, C_1 and C_2 shown in Table 1. It is interesting to note that all the methods yield same solution x^* = (3, 1, 2, 1) for $R_s(x^*)$ = 0.989612, which is also the global optima. The time taken by P-Alg (0.00413 sec.) is comparable with Kumar (0.00342 sec.) and Shi (0.00264 sec.) methods and is much smaller than N-N (0.01963 sec.) and A-G (0.90486 sec.) methods.

Table 1. Data for Figure 1

I	1	2	3	4
r_i	0.6984	0.625	0.8464	0.7536
c_{1i}	2	64	3	4
c_{2i}	48	74	23	74
C_1		132		
C_2		341		

Example 2*:* 5-unit complex network (Fig. 2) with linear constraint (n = 5 and k = 1). The problem is defined as:

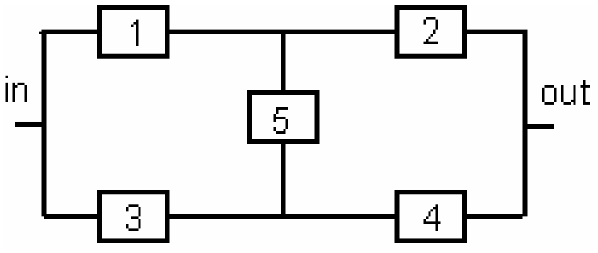

Figure 2. 5-Unit Bridge Network

$$\text{Maximise } R_s(X) = \begin{array}{l} R_1(x_1)R_2(x_2)+R_3(x_3)R_4(x_4)(Q_1(x_1) \\ +R_1(x_1)Q_2(x_2)) \\ +R_1(x_1)Q_2(x_2)Q_3(x_3)R_4(x_4)R_5(x_5) \\ +Q_1(x_1)R_2(x_2)R_3(x_3)Q_4(x_4)R_5(x_5) \end{array} \tag{6}$$

Subject to: $$\sum_{i=1}^{5} c_{1i}(x_i) \le 290 \tag{7}$$

$$x_i \ge 1, \quad \text{for } i = 1, 2, 3, 4, 5.$$

$$L(x) = (1, 1, 1, 1, 1),\ U(x) = (4, 146, 19, 3, 5)$$

The problem is solved for all the above methods with randomly generated values of parameters given in Table 2.

Table 2. Data for Figure 2

i	1	2	3	4	5
r_i	0.8106	0.6940	0.6974	0.8068	0.633 1
c_{1i}	45	1	8	56	35
C_1			290		

The results obtained for $x^*, g_1(x^*),\ R_s(x)$ and T with various methods are compared in Table 3. From the table it is evident that P-Alg and A-G methods obtain best optimal solution x^* = (4, 11, 1, 1, 1) for $R_s(x)$ = 0.999546 without any slack of components, but the execution time of P-Alg is much lower than that of A-G. The A-G method takes more time by an order of 25 in comparison with P-Alg method. Though both Shi and N-N methods also do not leave any slack of components but their solution quality is inferior in comparison with P-Alg and A-G methods. Although, Kumar method takes minimum execution time but the quality of solution is poorest of all and it leaves maximum slack of components. To illustrate the working of different iterations of the algorithm, an example of 7- unit bridge network (Fig. 3(a)) has been solved for random generated data set in Appendix B.

Table 3. Comparison of Figure 2 Complex Network

Methods	x^*	$g_1(x^*)$	$R_s(x^*)$	T (sec.)
P-Alg	(4,11,1,1,1)	290	0.999546	0.04992
Kumar	(3,7,3,1,1)	257	0.999293	0.00326
Shi	(2,61,6,1,1)	290	0.997432	0.24205
N-N	(3,16,6,1,1)	290	0.999514	0.05007
A-G	(4,11,1,1,1)	290	0.999546	1.26784

5. PERFORMANCE MEASURES

In addition to the above, each of the n = 4- and 5- unit (Fig. 1 and 2 respectively) complex networks is further solved for 9 additional sets of randomly generated parameters r_i , c_{ji} and C_j for the comparison of performance measures of different methods.

The performance in terms of computational efficiency and solution quality of P-Alg, Kumar, Shi, N-N and A-G methods, defined as u =1, 2, 3, 4, 5 respectively, is illustrated with number of examples. The test problems of computational experiments are generated for 4-, 5-, 7-, 10-, and 15- unit examples of complex systems (shown in Figs. 1- 5) used by Kumar [22, 23] with linear constraints. The algorithms of different methods for the test problems are compared through performance measures such as average relative error (A_u), maximum relative error (M_u), optimality rate (O_u) and average execution time (T) for ten randomly generated initial data (v = 1, 2, …., 10), defined as:
Average relative error for method u,

$$A_u = \frac{1}{10}\sum_{v=1}^{10}(R_v^* - R_{uv})/R_v^* \tag{8}$$

where

R_{uv} is system reliability obtained by method u for test problem v; and R_v^* is the best system reliability obtained by any of the four methods or by complete enumeration for the test problem v.

Maximum Relative Error for method u,

$$M_u = \max_v\{(R_v^* - R_{uv})/R_v^*\} \tag{9}$$

Optimality rate for method u, O_u = number of times (out of 10 problems) method u yields the best system reliability.
T is average execution time of 10 test problems (sec.)

Following section illustrate the solution of 3 complex networks (Figs. 3-5). For n = 7 (Fig. 3(a)) network, 4 problems are formed by taking two constraints k = 1, 5 and two different values of parameter C_j as '*small*' and '*large*' defined as

$$\{ C_j \} = w_j * \sum_{i=1}^{n} g_i^j \tag{10}$$

here w_j denotes random uniform deviates from 1.5 to 2.5 for 'small', and from 2.5 to 3.5 for 'large'. Data for the parameters of the problems are generated randomly by taking $\{g_i^j(x_i)\}$ a random uniform deviates from 0 to 100 and random uniform deviates $\{r_i\}$ from 0.6 to- 0.85. Another $n = 7$ (Fig. 3(b)), 10- (Figs 4) and 15- (Figs 5) unit networks are solved for two different data sets of parameters $k = 5$, C_j = *'small'* and $k = 5$, C_j = *'large'*. Thus, in total 12 test problems are solved for 6 different networks (Figs. 1-5) with various methods using MATLAB on a Pentium(R) D, 3.4 GHz CPU based computer. Each of the test problems is then solved for 10 randomly generated data sets for r_i , c_{ji} and C_j.

To obtain optimal solutions, P-Alg, Kumar and Shi methods use single initial solution (1, 1,…, 1) whereas A-G method uses 10 initial solutions generated randomly by a 2-phase procedure of Kim and Yum [30]. For N-N method, each problem is solved by taking initial solution (1, 1, …, 1) and 10 values of the balancing coefficient α as 0.1, 0.2,…., 1.0. Thus, out of 10 such solutions obtained by N-N and A-G methods best solution is selected for comparison with other methods. Thus in total 1320 test problems are solved (120 test problems by P-Alg, Kumar and Shi methods and 1200 test problems by N-N and A-G methods).

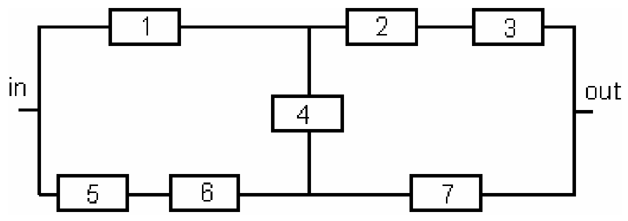

Figure 3(a). 7-Unit Complex System (n = 7)

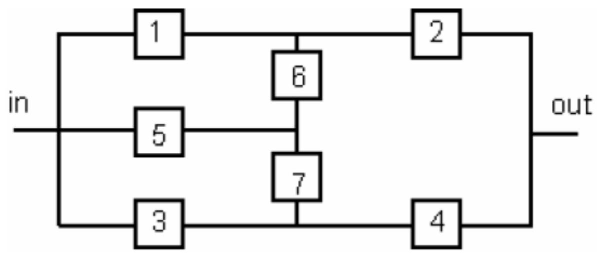

Figure 3(b). 7-Unit Complex System (n = 7)

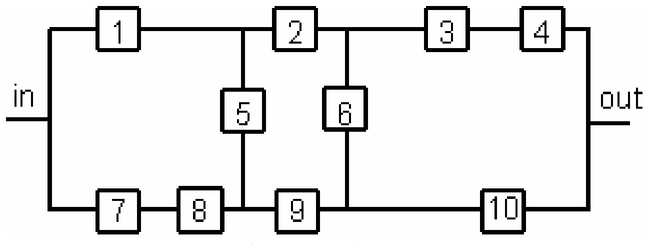

Figure 4. 10-Unit Complex System (n = 10)

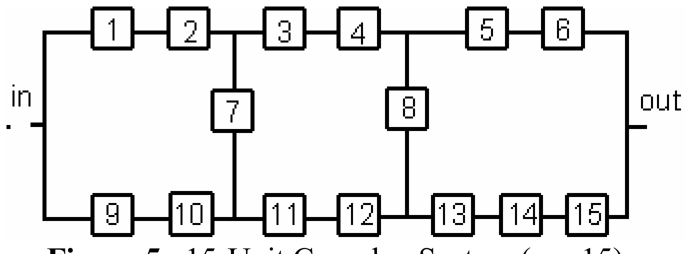

Figure 5. 15-Unit Complex System (n = 15)

The results of performance measures, i.e., average relative error (*A*), maximum relative error (*M*), optimality rate (*O*) and average execution time (*T)* obtained with P-Alg, Kumar, Shi, N-N, and A-G methods, are compared in Table 4. The performance measures *A, M* and *O* of P-Alg are consistently better than all other methods, except in case of 4,7x5-'large*'* and 8,10x5-*'small'* network examples where N-N method shows better performance. The average execution time (*T*) of P-Alg is also better or comparable with Kumar in most of the cases.

Table 4. Comparison of Performance Measures
for 4-, 5-, 7-, 10-, 15-Unit Networks (Fig. 1-5 respectively)

Example (m, *n* x *k*)	Methods	Performance Measures			
		A	M	O	T (sec.)
3, 4 x 2 Figure 1	P-Alg	0	0	10	0.06020
	Kumar	0.000353	0.00186	5	0.053242
	Shi	0.000344	0.00186	6	0.27828
	N-N	9.5E-06	9.5E-05	9	0.02385
	A-G	0.000296	0.00173	1	12.38190
4, 5 x 1 Figure 2	P-Alg	4E-07	4E-06	9	0.04992
	Kumar	0.001382	0.003364	1	0.003262
	Shi	0.003845	0.007585	0	0.24205
	N-N	3.6E-06	3.6E-05	9	0.05007
	A-G	0.000593	0.002059	2	1.26784
4, 7 x 1 'small' Figure 3(a)	P-Alg	7.95E-05	0.000795	9	0.05076
	Kumar	0.001152	0.004727	2	0.05165
	Shi	0.048417	0.182504	0	0.98615
	N-N	0.000486	0.003217	8	0.08194
	A-G	0.003395	0.010776	0	33.03050
4, 7 x 5 'small' Figure 3(a)	P-Alg	0	0	10	0.05164
	Kumar	0.003966	0.010224	4	0.050893
	Shi	0.011981	0.06485	3	0.61064
	N-N	0.000895	0.005394	8	0.05421
	A-G	0.001741	0.009337	1	7.80162
4, 7 x 1 'large' Figure 3(a)	P-Alg	0	0	10	0.04938
	Kumar	0.000217	0.000444	0	0.05124
	Shi	0.005614	0.025473	0	1.41909
	N-N	2.6E-06	1.4E-05	8	0.14222
	A-G	0.000112	0.000215	0	14.51660
4, 7 x 5 'large' Figure 3(a)	P-Alg	0.008177	0.021486	4	0.04226
	Kumar	0.001184	0.004195	0	0.05186
	Shi	0.005277	0.019515	0	1.07131
	N-N	0.000274	0.000788	5	0.09980
	A-G	0.000436	0.001459	1	8.62698
6, 7 x 5 'small' Figure 3(b)	P-Alg	0	0	10	0.05200
	Kumar	0.006507	0.015626	0	0.03064
	Shi	0.012012	0.040473	0	0.66712
	N-N	0.005372	0.008265	0	0.05283
	A-G	0.004888	0.00917	0	12.40320
6, 7 x 5 'large' Figure 3(b)	P-Alg	0.001255	0.010007	8	0.05080
	Kumar	0.000235	0.001405	0	0.05107
	Shi	0.000231	0.001419	0	1.16433
	N-N	3.22E-05	0.000131	2	0.09677
	A-G	9.84E-05	0.000724	0	13.93550
8, 10 x 5 'small' Figure 4	P-Alg	0.01371	0.058709	3	0.04383
	Kumar	0.010458	0.058121	5	0.05611
	Shi	0.043408	0.080705	0	1.95416
	N-N	0.001176	0.010479	7	0.15770
	A-G	0.005039	0.013028	0	3.62803

Example (m, *n* x *k*)	Methods	Performance Measures			
		A	M	O	T (sec.)
8, 10 x 5 'large' Figure 4	P-Alg	0.003071	0.02704	8	0.04433
	Kumar	0.02487	0.041681	1	0.05511
	Shi	0.044704	0.080265	0	1.95288
	N-N	0.015437	0.038307	0	0.15536
	A-G	0.014913	0.038306	1	14.09680
8, 15 x 5 'small' Figure 5	P-Alg	0	0	10	0.04407
	Kumar	0.041613	0.064621	0	0.08765
	Shi	0.088131	0.174326	0	3.70830
	N-N	0.033755	0.059137	0	1.08656
	A-G	0.037845	0.059135	0	15.34900
8, 15 x 5 'large' Figure 5	P-Alg	0	0	10	0.05097
	Kumar	0.067226	0.091703	0	0.08684
	Shi	0.112797	0.178473	0	3.65437
	N-N	0.059578	0.07733	0	1.07747
	A-G	0.063573	0.077328	0	62.50630

CONCLUSIONS

In this paper, an efficient heuristic algorithm for determining optimal redundancy allocation of complex systems has been proposed. It has been shown that quality of solution in P-Alg is better than all other methods in most of the cases. The computational time of P-Alg is either better or comparable with Kumar and Shi methods. The method also leaves minimum slack of components without any further possibility of redundancy. Therefore, the P-Alg method finds its greater utility for solving redundancy allocation problems where both the solution quality and computational time are of prime importance.

APPENDIX A

Notation

$a_l(X)$	Sensitivity factor of l^{th} minimal path set
$b_i(x_i)$	Subsystem selection factor for i^{th} subsystem with x_i components
c_{ji}	Cost of subsystem i for k^{th} constraint.
C_j	Total amount of resource j available
$g_i^j(x_i)$	Amount of resources-j consumed in subsystem-i with x_i components
k	Number of constraints, $j = 1, 2,\dots, k$
$L(x)$	$(L_{x_1}, L_{x_2}, ..., L_{x_n})$, Lower limit of each of subsystem i,
m	Number of minimal path sets, $l = 1, 2,., m$
n	Number of subsystems, $i = 1, 2,\dots, n$
P_l	l^{th} minimal path set of the system
Ps	$(l^1, l^2, ..., l^{min})$: priority vector s.t. l^1 and l^{min} are the number of minimal path sets respectively having maximum and minimum value of path selection parameter $a_l(X)$.
$Q_i(x_i)$	Unreliability of subsystem i with x_i components.
r_i	Reliability of a component at subsystem i.
$R_i(x_i)$	Reliability of subsystem i with x_i components.
R_r	Residual resources $C_j - \sum c_{ij} x_i$
$R_s(x)$	System reliability
T	Average execution time of 10 test problems (sec.)
$U(x)$	$(U_{x_1}, U_{x_2}, ..., U_{x_n})$, Upper limit of each of subsystem i,
x^0	Initial feasible solution
x^*	Optimal solution
x_i	Number of components in subsystem i; $i = 1,2,\dots.n$
X	$(x_1,\dots\dots\dots x_n)$
α	Balancing coefficient for N-N method

APPENDIX B

This section presents how the P-Alg is applied for constraint redundancy reliability optimization problem of a 7- unit bridge network with n = 7, k = 1 (Fig. 3(a)) for data given in Table 5 and the procedure is illustrated below:

Table 5. Data for Figure 3(a)

i	1	2	3	4	5	6	7
r_i	0.7321	0.6109	0.7963	0.7013	0.6247	0.7104	0.6631
c_{1i}	54	48	95	24	68	9	99
C_1				794			

Illustration

Step1: Initialize i = 7, m =4, k=1, r_i = [0.7321, 0.6109, 0.7963, 0.7013, 0.6247, 0.7104, 0.6631], c_{1i} = [54, 48, 95, 24, 68, 9, 99], $R_r = C_j$ = 794, and P$_l$ (for l = 1, 2, 3, 4.) of the system are P$_1$ = [1, 2, 3], P$_2$ = [1, 4, 7], P$_3$ = [5, 6, 7] and P$_4$ = [2, 3, 4, 5, 6] .

Step2: Let x_i = 1 for all i; i = 1, 2,…,7 $i.e$ X = (1,1,1,1,1,1,1) and for this value of X, R_r = 397.

Step3: The sensitivity factor $a_l(X)$ i.e. [a_1, a_2, a_3, a_4] = [1.435, 1.527, 1.328 0.493], hence, Ps = [2, 1, 3, 4].

Step4: The path set $[P_{Ps(Q)}]_{(Q=1)} = [P_{Ps(1)}] = P_2$ is selected for optimization.

Step5: For P_2 all values of $b_i(x_i)$ for each $i \in P_2$ = [10.765, 23.201, 5.318] $_{(i=1,4,7)}$, i^*= 4 as $b_4(x_4)$ = 23.201 is the maximum.

Step6(i): By incrementing $x_4 = x_4 +1$, X = (1,1,1,2,1,1,1) and R_r= 373 (i.e. a +ve number), no constraints is violated.

Step5: For P_2, $b_i(x_i)$ = [10.765, 6.930, 5.318] $_{(i=1,4,7)}$, i^*= 1 as $b_1(x_1)$ = 10.765,

Step6(i): Increment $x_4 = x_4 +1$, X = (2,1,1,2,1,1,1) and R_r = 319, no constraint is violated. Repeat the Step5 and Step6(i) until X = (3,1,1,4,1,1,3) and R_r = 19, $b_i(x_i)$ = [0.773, 0.618, 0.604] $_{(i=1,4,7)}$, i^*= 1 as $b_1(x_1)$ = 0.773 is the maximum.

Step6(ii): Increment $x_1 = x_1 +1$, X = (4,1,1,4,1,1,3) and R_r = -35, constraint is violated, X is reinstated at its previous step value i.e. (3,1,1,4,1,1,3), subsystem i^* = 1 of P_2 is removed from further consideration, and $b_i(x_i)$ = [--, 0.618, 0.604] $_{(i=1,4,7)}$, next highest value of $b_i(x_i)$ is $b_4(x_4)$ = 0.618 for i^*= 4.

Step6(ii): Increment $x_4 = x_4 +1$, X = (3,1,1,4,1,1,3) and R_r = -5, constraint is violated, X is reinstated at its previous step value i.e. (3,1,1,4,1,1,3), subsystem i^* = 4 of P_2 is removed from further consideration, and $b_i(x_i)$ = [--, --, 0.604] $_{(i=1,4,7)}$, next highest value of $b_i(x_i)$ is $b_7(x_7)$ = 0.604 i^*= 7. Repeat the Step6 until all the subsystems of path P_2 are removed from further consideration and X = (3, 1, 1, 4, 1, 1, 3), $b_i(x_i)$ = [--, --, --] $_{(i=1,4,7)}$, $a_l(X)$ i.e. [a_1, a_2, a_3, a_4] = [1.435, ++, 1.328 0.493].

Step6(iii) $Q = Q +1$, $[P_{Ps(Q)}]_{(Q=2)} = [P_{Ps(2)}] = P_1$, for P_1 all values of $b_i(x_i)$ for each $i \in P_1$ = [--, 10.105, 6.555]$_{(i=1,2,3)}$, as subsystem i = 1 has already been optimized hence not considered further.

Step5: The next highest value of $b_i(x_i)$ is $b_2(x_2)$ = 10.105 for i^*= 2.

Step6: By repetitively checking the various conditions for the sub steps in similar manners as described above, it is found that there is no possible of redundancy for any of the subsystem of the path set P_1, hence all the subsystems the path set are removed from further consideration, hence the path set is removed. X = (3,1,1,4,1,1,3), $b_i(x_i)$ = [--, --, --] $_{(i=1,2,3)}$, $a_l(X)$ i.e. [a_1, a_2, a_3, a_4] = [++, ++, 1.328 0.493].

Step6(iii) $Q = Q + 1$, $P_{Ps(Q)}]_{(Q=3)} = [P_{Ps(3)}] = P_3$, for P_3 all values of $b_i(x_i)$ for each $i \in P_3$ = [7.294, 62.673, -- $]_{(i=5,6,7)}$, as subsystem $i = 7$ has already been optimized hence not considered further.

This way, the above Step5 and Step6 are again repeated for the selected path set P_3 until all the subsystems and/or all the minimal path sets are excluded from further consideration, till optimal solution $x^* = X = (3,1,1,4,1,3,3)$ is obtained for $R_r = 1$. All the intermediate steps and the values of different parameter during different steps are given in Table 6.

Step7: System reliability, $R_s(x^*) = 0.971495$ is determined for the optimal solution $x^* = X = (3, 1, 1, 4, 1, 3, 3)$ and $T = 0.05076$ sec.

Table 6. Procedure of P-Alg for Figure 3 Complex Network

Allocation	Residual resources	Minimal path set sensitivity factor				Components selection factor						
(x_1, x_2, x_3, x_4, x_5, x_6, x_7)	$C_j - \sum c_{ij} x_i$	(a_1	a_2	a_3	a_4)	(b_1	b_2	b_3	b_4	b_5	b_6	b_7)
1,1,1,1,1,1,1	397	1.435	1.527*	1.328	0.493	10.765			23.201#			5.318
1,1,1,2,1,1,1	373					10.765#			6.930			5.318
2,1,1,2,1,1,1	319					2.884			6.930#			5.318
2,1,1,3,1,1,1	295					2.884			2.070			5.318#
2,1,1,3,1,1,2	196					2.884#			2.070			1.792
3,1,1,3,1,1,2	142					0.773			2.070#			1.792
3,1,1,4,1,1,2	118					0.773			0.618			1.792#
3,1,1,4,1,1,3	19					0.773#			0.618			0.604
4,1,1,4,1,1,3	-35$					--			0.618#			0.604
3,1,1,5,1,1,3	-5$					--			--			0.604#
3,1,1,4,1,1,4	-80$					--			--			--
3,1,1,4,1,1,3	19	1.435*	++	1.328	0.493	--	10.105#	6.655	--			--
3,2,1,4,1,1,3	-29$					--	--	6.655#	--			--
3,1,2,4,1,1,3	-76$					--	--	--	--			--
3,1,1,4,1,1,3	-19$	++	++	1.328*	0.493	--	--	--	--	7.294	62.673#	--
3,1,1,4,1,2,3	10					--	--	--		7.294	18.150#	--
3,1,1,4,1,3,3	1					--	--	--	--	7.294#	5.256	--
3,1,1,4,2,3,3	-67$					--	--	--		--	5.256#	--
3,1,1,4,1,3,3	1	++	++	++	0.493%	--	--	--	--	--	--	--

* This minimal path set has the highest value of the sensitivity factor.
# A redundant component is to be added to this subsystem for highest selection factor.
$ Constraint violation.
-- Subsystem is removed from further consideration.
++ Minimal path set removed from further consideration.
% All the subsystems of the path have already been optimized, hence no further possibility of redundancy.

REFERENCES

1. **Bellman, R. E.** and **Dreyfus, E.** Dynamic programming and reliability of multi component devices, *Operations Research*, 1958, .6, 200-206.
2. **Misra, K. B.** Dynamic programming formulation of redundancy allocation problem, *International Journal of Mathematical Education in Science and Technology*, 1971, 2, 207-215.
3. **Fyffe, D. E., Hines, W. W.** and **Lee, N. K.** System reliability allocation and a computational algorithm, *IEEE Trans. Reliability*; 1968, 17,.64–69.
4. **Kuo, W.** and **Wan, R.** Recent advances in optimal reliability allocation, *IEEE Trans. on Systems, Man, and Cybernetics-Part A: Systems and Humans,* 2007, 37(2),143-156.
5. **Misra, K. B.** and **Sharma, U.** An efficient algorithm to solve integer programming problems arising in system reliability design, *IEEE Trans. Reliability*, 1991,40,81-91.
6. **Tillman, F. A., Hwang, C. L.** and **Kuo, W.** Optimization techniques for system reliability with redundancy, a review, *IEEE Trans. Reliability,* 1977, R-26(3),147–155.

7. **Tillman, F. A., Hwang, C. L.** and **Kuo, W.** Determining component reliability and redundancy for optimum system reliability, *IEEE Trans. Reliability*, 1997,26(3),162–165.
8. **Tillman, F. A., Hwang, C. L.,** and **Kuo, W.** *Optimization of system reliability*, 1985 (Marcel Dekker, New York).
9. **Tillman, F. A., Hwang, C. L.,** and **Kuo, W.** A note on heuristic methods in optimal system reliability, *IEEE Trans. Reliability,* 1978, R-27(5), 320-324.
10. **Nakagawa, Y.** and **Miyazaki, S**. An experimental comparison of the heuristic methods for solving reliability optimization problems, *IEEE Trans. Reliability,* 1981, R-30,181-184.
11. **Kuo, W.** and **Prasad, V. R.** An Annotated Overview of System-Reliability Optimization, *IEEE Trans. Reliability,*2000, 49(2),176-187.
12. **Sharma, J.** and **Venkateswaran, K. V.** A direct method for maximizing the system reliability, *IEEE Trans. Reliability,* 1971,R-20,256-259.
13. **Aggarwal, K. K., Gupta, J. S.** and **Misra, K. B.** A new heuristic criterion for solving a redundancy optimization problem, *IEEE Trans. Reliability,* 1975, R-24, 86-87.
14. **Aggarwal, K. K.** Redundancy optimization in general systems, *IEEE Trans. Reliability,* 1977, R-25, 330-332. (Corrections, 1977, R-26(Dec),345).
15. **Nakagawa, Y.** and **Nakashima, K**. A heuristic method for determining optimal reliability allocation, *IEEE Trans. Reliability,* 1977, R-26,156-161.
16. **Gopal, K., Aggarwal, K. K.** and **Gupta, J. S.** An improved algorithm for reliability optimization, *IEEE Trans. Reliability,* 1978,R-27(5),325-328.
17. **Gopal, K., Aggarwal K. K.** and **Gupta, J. S.** A new method for solving reliability optimization problem, *IEEE Trans. Reliability,* 1980, R-29, 36–38.
18. **Chen, H. C., Shi, D. H.** and **Xu, W.X.** A new algorithm for network system reliability, *Microelectron. & Rel.,* 1985, 25(1),35-40.
19. **Shi, D. H.** A new heuristic algorithm for constrained redundancy optimization in complex system, *IEEE Trans. Reliability,* 1987, R-36(5), 621-623.
20. **Sharma, U.** and **K. B. Misra,** An efficient algorithm to solve integer programming problems in reliability optimization, *Int'l J. Quality & Reliability Management,*1990, 7(5), 44–56.
21. **Prasad, V. R., Aneja, Y. P.** and **Nair, K. P. K.** A heuristic approach to optimal assignment of components to a parallel-series network, *IEEE Trans. Reliability,*1991, 40, 555–558.
22. **Kumar, P., Chaturvedi, D. K.** and **Pahuja, G. L.** A heuristic algorithm for constrained redundancy reliability optimization and performance evaluation, *Proc. IMechE, Part O:J. of Risk and Reliability,* 2009, 223, 381-386.
23. **Kumar, P., Chaturvedi, D.K.** and **Pahuja, G.L.** Heuristic Methods for Solving Redundancy Allocation in Complex Systems, *Int. J. of Reliability and Safety,* 2010, 4(2/3),285-298.
24. **Coit, D. W.** and **Smith, A. E.** Reliability optimization of series-parallel systems using a genetic algorithm, *IEEE Trans. Reliability*, 1996,45,254–260.
25. **Ida, K., Gen M.** and **Yokota, T.** System reliability optimization of series-parallel systems using a genetic algorithm, *Proceedings of the 16th International Conference of Computers and Industrial Engineering*, 1994,.349–352.
26. **Painton L.** and **Campbell, J.** Genetic algorithms in optimization of system reliability', *IEEE Trans. Reliability*, 1995,44, 172–178.
27. **Kulturel-Konak, S., Smith, A. E.** and **Coit, D. W.** Efficiently solving the redundancy allocation problem using tabu search, *IIE Trans. Reliability*, 2003,35(6), 515-526.
28. **Liang, Y. C.** and **Smith, A. E.** An ant colony optimization algorithm for the redundancy allocation problem (RAP), *IEEE Trans. Reliability*, 2004,53(3),417-423.
29. **Kuo, W., Prasad, V. R., Tillman. F. A.** and **Hwang, C. L.** *Optimal Reliability Design Fundamentals and Applications*, 2001 (Cambridge: Cambridge University Press).
30. **Kohda, T.** and **Inoue, K.** A reliability optimization method for complex systems with the criterion of local optimality, *IEEE Trans. Reliability,* 1982,R-31, 109-111.

31. **Kim J. H.** and **Yum, B. J.** A heuristic method for solving redundancy optimization problems in complex systems, *IEEE Trans. Reliability,* 1993, 42(4),572–578.
32. **Agarwal, M.** and **Gupta, R.** Penalty function approach in heuristic algorithms for constrained redundancy reliability optimization, *IEEE Trans. Reliability,* 2005,R-54(3),549-558.