Computer and Information Science; Vol. 7, No. 3; 2014
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

Using Genetic Algorithm to Find the Optimal Shopping Policy for 1-out-of-n Active-Redundancy Series Systems under Budget Constraint

Saleem Z. Ramadan[1]

[1] Department of Mechanical and Industrial Engineering, Applied Science University, Shafa Badran, Jordan

Correspondence: Saleem Z. Ramadan, Department of Mechanical and Industrial Engineering, Applied Science University, Shafa Badran, 11931, Amman, Jordan. E-mail: s_ramadan@asu.edu.jo

Received: April 24, 2014 Accepted: June 10, 2014 Online Published: July 29, 2014

doi:10.5539/cis.v7n3p81 URL: http://dx.doi.org/10.5539/cis.v7n3p81

Abstract

The mathematical model to find the optimal shopping policy from many available manufacturers for 1-out-of-n active redundancy series systems under budget constraint was formulated and tested using GA. The study showed that the number of possible combinations for this problem can be very high and the majority of those combinations are infeasible. This renders the enumeration technique ineffective or even impossible in practice, the matter that calls for a solution through GA.

The results showed that the proposed genetic algorithm has high degree of robustness. Moreover, the results showed that the proposed algorithm is superior to the enumeration technique in terms of both computational time and quality of solution. Furthermore, the results showed that the convergence of the algorithm to the optimal solution is high.

Keywords: system reliability, GA, redundancy, position-based crossover, active redundancy, shopping policy

1. Introduction

System reliability can be defined as the probability that the system will conduct its intended functions satisfactorily at least for a given period of time when operated under normal operating conditions. Redundancy, in its both configurations: high and low redundancy level, can increase the reliability of the system. This increase in the system reliability is normally accompanied with increase in the system cost. Redundancy allocation problem (RAP) is considered an optimization problem involving the selection of components and their layout in the system to maximize the system reliability under certain constraints.

The problem of how to choose and mix between different manufacturers that supply similar components with different reliabilities and costs for a series system can be treated as a special case of series-parallel RAP.

The problem of optimal shopping policy considered in this study is as follows: Consider a system that consists of k different series subsystems such that each subsystem consists of a single different component than the other subsystems. For each component, there are number of different manufacturers available in the market for that component with different reliabilities and costs. The optimal shopping problem is to determine the optimal configuration of the system within the budget, i.e., the configuration that will have the best reliability within the given budget. The configuration of the system includes the determination of the manufacturer(s) along with the number of components that will be used from each manufacturer in each subsystem such that the best reliability possible is reached and the budget of the system is not exceeded.

2. Related work

The RAP problem was proved to be an NP-hard problem. The redundancy allocation problem had been studied extensively in the literature. Fyffe et al. (1968) developed goal programming to solve system reliability allocation problem. You and Chen (2005) proposed an efficient heuristic to solve the RAP. Tian and Zuo (2006) used multi-objective optimization to solve RAP where the model maximized the system performance and minimized the system cost and weight simultaneously. Kulturel and Coit (2008) used objective prioritization in a multi-objective frame work to optimize the RAP. Coit (2001) optimized the RAP for non-repairable systems using cold standby strategy for the subsystems. Taboada and Coit (2012) used multi-objective evolutionary

algorithm based on rank selection and elitist reinsertion to optimize the series-parallel systems. Mori et al. (2007) used simulated annealing along with genetic algorithm to solve three RAPs in series condition. Dao and Murat (2013) used a decomposition- based approach to have an exact solution for the RAP for series-parallel systems. Zia and Coit (2010) used column generation approach for solving RAP for series-parallel systems. Liang and Chen (2007) used Variable Neighborhood Search (VNS) to solve the RAP when reliability cost and weight are considered as objective functions. David and Alice (1996) used GA hybrid with neural network to solve the RAP under lower bound of reliability constraint when cost of the system has to be minimized. Coit and Smith (1994) showed that GA is one of the best methods for solving RAP and explained the advantages for such a methodology in solving the problem. In their GA they determined the reliability of the system directly during the GA search. Painton and Campell (1994) and Ida et al. (1994) also used GA to solve the RAP but they used simulation to determine the reliability of the system. Liang and Smith (2004) used ant colony meta-heuristic optimization method to solve the redundancy allocation problem (RAP). Feller (1968) solved the RAP as an occupancy problem as the author did not have a limit for the number of the components that can be used as redundant components. Other authors used nonlinear programming to identify optimal reliability levels at the component or subcomponent level (Tillman et al., (1980, 1977)). Others used integer and dynamic programming to solve the problem (Gen et al., 1990; Misra & Sharma, 1991).

Generally speaking, the literature available for the RAPs differs from one to another in the constraints and the methodology used for solving them. Cost, weight, size, and reliability of the components were used as constraints sometimes and as single objective functions or multi-objective functions in another times. In this study, the problem of choosing and mixing between different available manufacturers to build a 1-out-of-n active redundancy series system within the budget available is studied. This distinguishes this study from other studies in this area as this study focuses on the shopping policy, i.e., "who to buy from and how many to buy".

The rest of the paper will be organized as follows: section 3 presents the problem formulation, section 4 presents the design of the proposed genetic algorithm, section 5 presents the experimentation and the results, and section 6 presents the conclusions.

3. Problem Formulation

Configuration of a system involves many factors such as total cost, reliability, size, and weight. If the objective is to maximize the reliability of a system within certain budget, then the configuration of the system requires the determination of the best combination of components that yields the highest possible reliability within the given budget. Depending on the budget available, this configuration may allow for redundancy. The determination of the best combination of the redundant components to use in each subsystem depends on the available similar components (different manufacturers) in the market and their costs and reliabilities. For this study purpose, the redundant components are those components with different manufacturers and similar function. Since those components have different manufacturers they usually have different reliabilities and costs.

If at least one component need to be operating in each subsystem that consists of *n* redundant components, the configuration is known as 1-out-of-n redundancy problem. The 1-out-of-n active-redundancy series system is a system that has K 1-out-of-n subsystems on series that has all of its redundant components operating simultaneously. Figure 1, shows a schematic diagram for a typical 1-out-of-n active-redundancy series system with K subsystems.

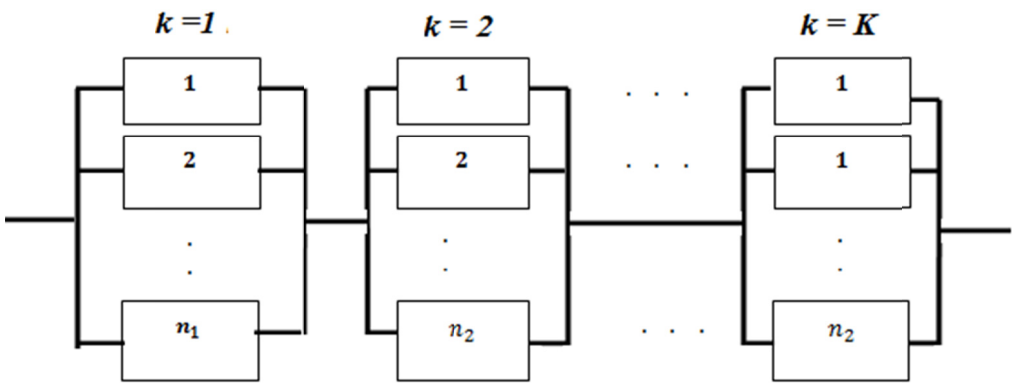

Figure 1. A schematic diagram for a typical 1-out-of-n active-redundancy series system with K subsystems

Consider a system consists of K different series components (subsystems) with n_k different manufacturers for each kth subsystem. Furthermore, assume that each subsystem's manufacturer has a different cost and reliability for the subsystem denoted by $C_{k,m_{k,i}}$ and $R_{k,m_{k,i}}$, respectively, where $m_{k,i}$denotes the ith manufacturer for the kth subsystem. The design variables for the model is the number of components used from each manufacturer in each subsystem, $N_{m_{k,i}}$, to reach the optimal system reliability, R_s, within the available budget, $maxcost$. The mathematical model for the problem can be expressed as follows:

$$max\ R_s = \prod_{k=1}^{K}\left(1-\prod_{i=1}^{n_k}\left(1-R_{k,m_{k,i}}\right)^{N_{m_{k,i}}}\right) \tag{1}$$

s.t

$$\sum_{k=1}^{K}\sum_{i=1}^{n_k} C_{k,m_{k,i}} \times N_{m_{k,i}} \leq maxcost \tag{2}$$

$$1 \leq N_{m_{k,i}} \leq floor\left(\frac{maxcost}{C_{k,m_{k,i}}}\right) \qquad \forall\ k = 1,2,\dots K\ and\ \forall i = 1,2\dots n_k \tag{3}$$

$$N_{m_{k,i}} \leq \frac{maxcost}{C_{k,m_{k,i}}} \qquad \forall k = 1,2,..,K\ and \forall\ i = 1,2,..n_k \tag{4}$$

$$N_{m_{k,i}}\ positive\ integers \tag{5}$$

Equation (1) is the objective function for this model which denotes the overall reliability of the 1-out-of-n active redundancy series system with K subsystems. Equation (2) guarantees that the overall cost of the system will not exceed the budget. Equation (3) limits the number of components for any subsystem that can be used from any manufacturer in the system and also guarantees that at least one component must be used in any subsystem, i.e., guarantees that the system must consist of a total of K subsystems. Equation (4) limits the number of components used in any subsystem for a certain manufacturer, and finally, equation (5) guarantees that only whole components can be used in the subsystems.

This model is clearly a non-linear integer model that belongs to the NP-hard class, therefore solving it with exact methods is mathematically intractable. Evolutionary algorithms can be used to solve such problem efficiently. Hence, in this study, a genetic algorithm will be proposed and used to solve this problem. Figure 2, shows a schematic diagram for the problem in hand.

The proposed genetic algorithm will be explained in the next section.

4. Design of the Proposed Genetic Algorithm

Genetic algorithm is one type of heuristic optimization search based on Darwin's natural selection theory. Genetic algorithm simulate the natural systems in solving problems. It enhances the initial population (initial solutions) through continuous evolution over generations through successive application of exploration and exploitation operators to reach a superior population that contains solutions (chromosomes) superior to the initial population. The genetic algorithm starts by encoding the solutions (phenotype) into chromosomes (genotype) using certain vector string template. This template is used to generate the initial population. The fitness value for each individual in the population is calculated, using a fitness function, to determine the next generation's parents through selection operator. Then, exploration (crossover) operator is applied on the parents to produce the offspring after which the exploitation (mutation) operator is applied on the offspring to generate the individuals of that generation. This evolutionary process, reproduction and selection, continues until certain level of convergence or a predetermined termination criterion is satisfied.

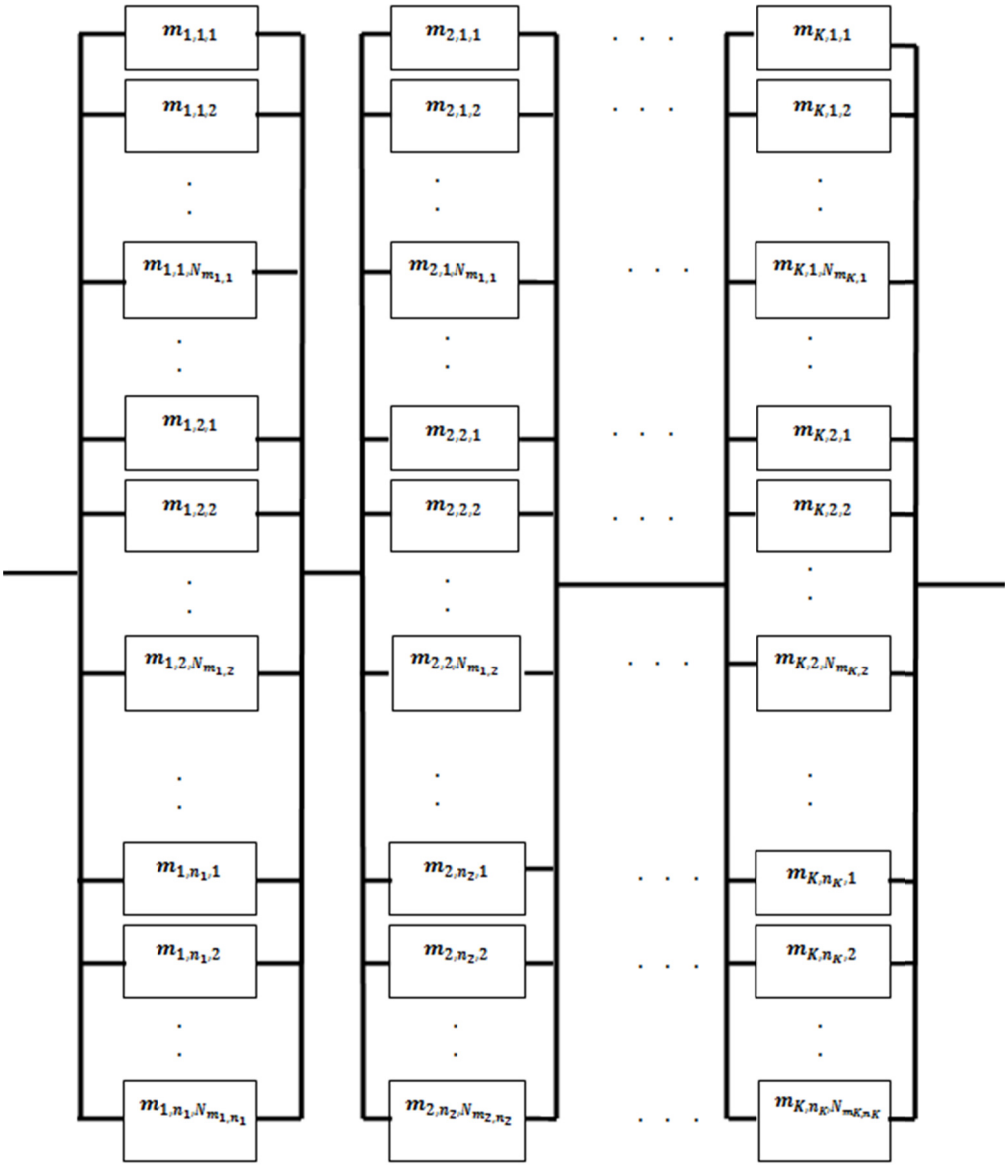

Figure 2. A schematic diagram for the problem in hand

The genetic algorithm used in this study is discussed in the next subsections.

4.1 Chromosome Representation

The chromosome consists of $K \times \max\left(N_{m_{k,i}}\right)$ positive integer genes. Each gene will carry three pieces of information: the subsystem number (k), the manufacturer number (i), and the number of components used in the kth subsystem from the manufacturer i $\left(N_{m_{k,i}}\right)$. The location of the gene in the chromosome gives the first two pieces of information, k and i, while the value of the gene gives the third piece of information $\left(N_{m_{k,i}}\right)$. This type of chromosome representation is known in literature as position-based chromosome representation.

To explain the chromosome representation used in the proposed GA, consider a system consists of 3 series subsystems for which the first subsystem has 4 different manufacturers, the second subsystem has 5 different manufacturers, and the third subsystem has 2 different manufacturers. In this problem $K = 3$ and $\max\left(N_{m_{k,i}}\right) = \max\{4, 5, 2\} = 5$. Hence, the chromosome consists of $3 \times 5 = 15$ genes. The first set of 5 genes represents the first subsystem. These genes can have values between 0 and 4 with at least one gene of a value between 1 and 4 and at least one gene of a value of 0 (since there are only 4 available manufacturers). The second set of 5 genes

represents the second subsystem. These genes have values between 0 and 5 with at least one gene of a value between 1 and 5. The third set of 5 genes represents the third subsystem. These genes have values between 0 and 2 with at least one gene of a value between 1 and 2 and at least 3 genes with values of 0 (since there is 2 available manufacturers). Figure 3 shows a possible chromosome for this specific problem.

0	2	1	5	0	2	1	1	2	0	0	0	2	0	0

Figure 3. Chromosome representation for one possible solution

This chromosome suggests the use of 8 parallel components for subsystem 1 as follows: 2 components from manufacturer 2, 1 component from manufacturer 3, and 5 components from manufacturer 4. No components will be used from manufacturer 1. Note that gene number 5 must have a value of 0 since there are only 4 available manufacturers for this subsystem. The chromosome also suggests the following for subsystem 2: 2 components from manufacturer 1, 1 component from manufacturer 2, 1 component from manufacturer 3, and 2 components from manufacturer 4. No components will be used from manufacturer 5. Furthermore, the chromosome suggests the use of only 2 components from manufacturer 3 for subsystem 3.

This chromosome can be represented as a reliability block diagram as shown in Figure 4 from which the overall reliability of the system can be calculated. In this figure m_i represents the manufacturer number i.

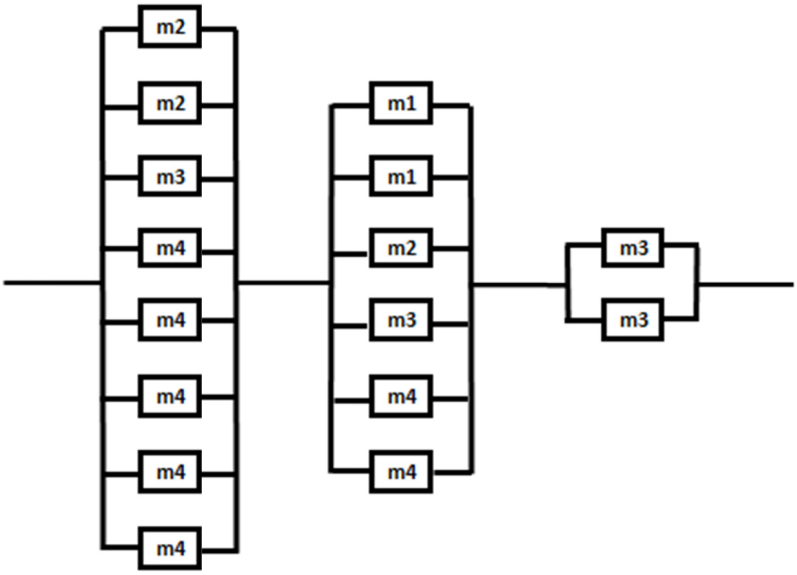

Figure 4. Reliability block diagram for Figure 3

4.2 Fitness Function

The fitness function for the proposed GA is the function that calculates the overall system reliability. This function is given by equation (1). For convenience, it is repeated her.

$$R_s = \prod_{k=1}^{K}\left(1 - \prod_{i=1}^{n_k}\left(1 - R_{k,m_{k,i}}\right)^{N_{m_{k,i}}}\right)$$

The fitness values calculated the different chromosomes are used as the base for elitist selection in the selection stage of the algorithm.

4.3 Crossover Operator

The Frequency Crossover operator (FC) proposed in Ramadan (accepted 2012) will be used in this GA. In this operator, chromosomes will be selected randomly in groups consist of Y chromosomes. The chromosomes in each group will be sorted ascending based on their fitness values. The best half of the chromosomes in each group will form the crossover group. Using 100% crossover rate, the best chromosome in the crossover group

will crossover with every other chromosome in the crossover group to form $\frac{Y}{2}-1$ offspring. The frequency of the corresponding genes in both parents will be evaluated. Frequency can be either one or two. A frequency of two means that both corresponding genes carry the same number of components for the same manufacturer of the subsystem, while a frequency of one means that the two corresponding genes have different numbers of components for the same manufacturer of the subsystem. The genes with frequency of two will be determined and their values will be copied into the offspring with their relative positions preserved. The remaining genes, those that have a frequency of one, will have their values assigned randomly. This randomly assigned values will eliminate the need for mutation operator.

The new offspring will be subjected to a feasibility check to ensure that the budget constraint is not violated. In case that the chromosome is not feasible (budget constraint is violated), a penalty will be imposed on the chromosome's fitness value that will change it to zero, hence, the offspring will never be selected and thus will be extinguished. Figure 5 shows a schematic diagram for the FC.

Note that in the offspring of Figure 5, the genes numbers 4, 6, and 12 are assigned randomly as they have a frequency of 1 while the other genes (1, 2, 3, 5, 7, 9, 10, 11, 13, 14, 15) will have the values from their parents as they have a frequency of 2. This crossover strategy can be seen as a heuristic mutation for the best chromosome as the important genes (those having a frequency of two) preserve their values and positions in the offspring while the rest of the genes (those having a frequency of one) are mutated. This can eliminate the need for mutation operator.

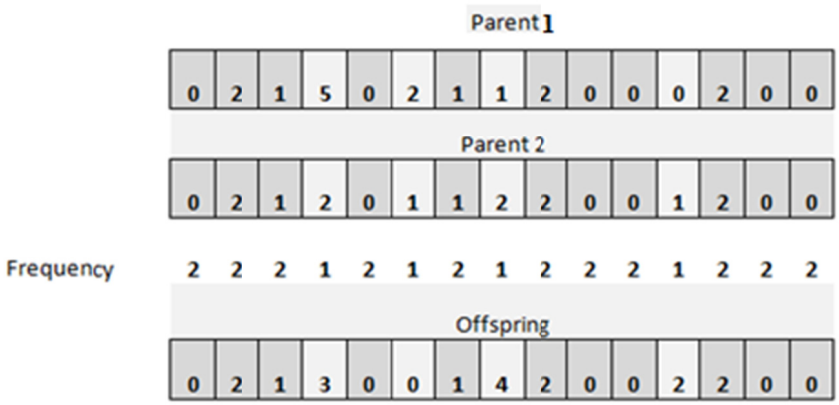

Figure 5. Schematic diagram for the FC

5. Experimentation and Results

In this section two systems (one large and one small) will be designed using the proposed GA . For the large problem, the system consists of 10 subsystems with an available budget of $2000. The relevant data is shown in Table 1 where C and R stands for cost and reliability of the component respectively.

Table 1. Relevant data for the large problem

Manuf	Subsys1		Subsys2		Subsys3		Subsys4		Subsys5		Subsys6		Subsys7		Subsys8		Subsys9		Subsys10	
	C	R	C	R	C	R	C	R	C	R	C	R	C	R	C	R	C	R	C	R
1	10	0.52	30	0.70	20	0.87	26	0.79	65	0.70	10	0.51	69	0.43	26	0.53	5	0.4	45	0.62
2	13	0.56	39	0.59	21	0.75	28	0.70	71	0.60	15	0.63	75	0.45	29	0.54	9	0.49	49	0.65
3	15	0.59	42	0.61	21	0.87	28	0.65	75	0.61	18	0.64	79	0.51	31	0.59	10	0.52	53	0.69
4	18	0.62	42	0.63	26	0.82	30	0.71	76	0.61	21	0.64	90	0.62	34	0.6	13	0.59	62	0.7
5	25	0.85	49	0.68	29	0.83	32	0.73	78	0.64	21	0.65	95	0.75			15	0.57		
6	26	0.84			31	0.84	35	0.79	79	0.65	23	0.67	120	0.83						
7	28	0.81			34	0.85			85	0.70	25	0.72	123	0.84						
8	30	0.84									26	0.73								
9	32	0.83									30	0.75								
10	34	0.84																		

By varying the size of the population and the number of generations and taking 100 replications, different results were obtained as shown in Table 2.

Table 2. Results for the large problem

Size of Pop.	Num. of Gene.	Max. Rel.	Min. Rel.	Ave. Rel.	S. D.	Coe. Var.
50	1000	0.8809	0.6363	0.7736	0.0842	0.1088
50	2000	0.8911	0.6403	0.8034	0.0679	0.0845
100	1000	0.8997	0.6487	0.8159	0.0557	0.0683
100	2000	0.8586	0.6548	0.8327	0.0329	0.0395
200	1000	0.9111	0.8138	0.8443	0.0156	0.0184
200	2000	0.903	0.8237	0.8477	0.0130	0.0153
300	1333	0.9131	0.8300	0.8492	0.0123	0.0145

The best chromosome found is shown in Figure 6. This solution has a reliability of 0.9131 and a cost of $2000.

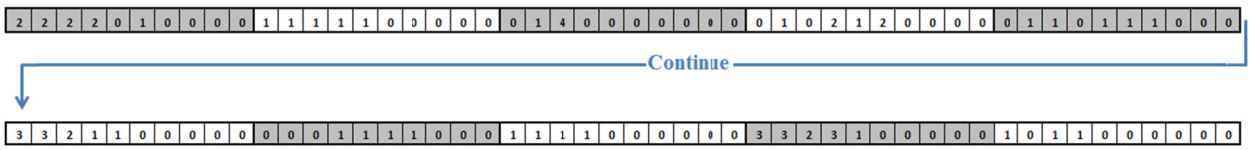

Figure 6. Best chromosome found

The reliability block diagram corresponding to this chromosome is shown in Figure 7.

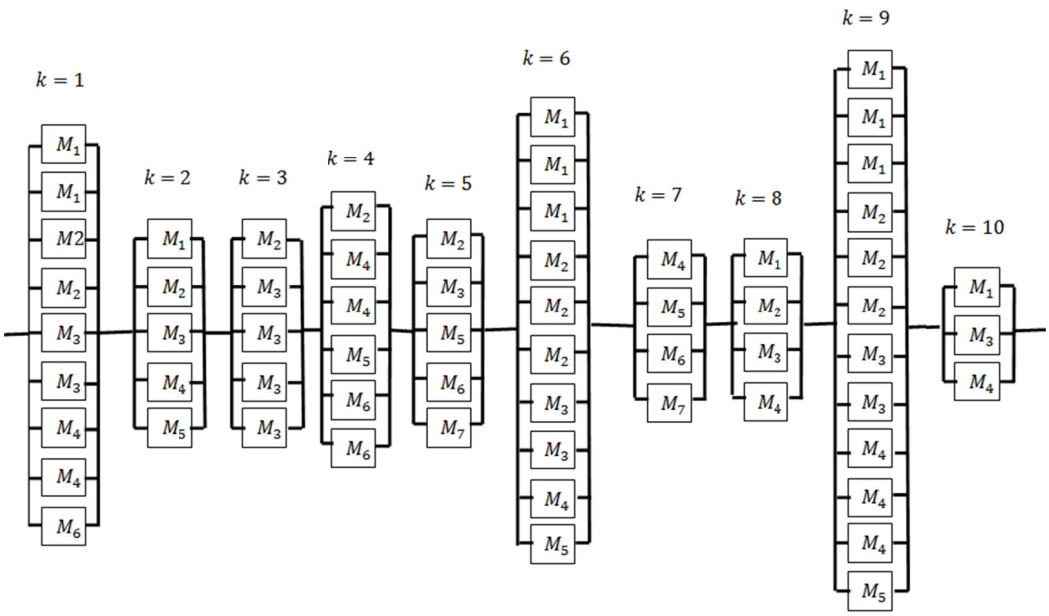

Figure 7. Reliability block diagram for Figure 6

The solution calls for the shopping policy shown in Table 3.

Table 3. Shopping policy for the system

anuf.	Units for Subsy1	Units for Subsy2	Units for Subsy3	Units for Subsy4	Units for Subsy5	Units for Subsy6	Units for Subsy7	Units for Subsy8	Units for Subsy9	Units for Subsy10
1	2	1	0	0	0	3	0	1	3	1
2	2	1	1	1	1	3	0	1	3	0
3	2	1	4	0	1	2	0	1	2	1
4	2	1	0	2	0	1	1	1	3	1
5	0	1	0	1	1	1	1	N/A	1	N/A
6	1	N/A	0	2	1	0	1	N/A	N/A	N/A
7	0	N/A	0	N/A	1	0	0	N/A	N/A	N/A
8	0	N/A	N/A	N/A	N/A	0	N/A	N/A	N/A	N/A
9	0	N/A	N/A	N/A	N/A	0	N/A	N/A	N/A	N/A
10	0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

It is worth to mention here that the number of different configurations (feasible and infeasible) for the system is enormous as the number of combinations that can be used in any subsystem is huge, the matter that makes the enumeration of different configurations intractable. Considering the constraint given by equation (4), the total number of configuration combinations can be calculated by the following equation:

$$\prod_{k=1}^{K}\prod_{i=1}^{n_k}\left(1+floor\left(\frac{MaxCost}{C_{k,m_{k,i}}}\right)\right) \tag{6}$$

In equation (6), 1 was added to count for the case of having zero components from the prospective manufacturer. For the large problem, the number of possible combinations is calculated using equation (6) as:

$(201\times154\times134\times112\times81\times77\times72\times67\times63\times59)\times(67\times52\times48\times48\times41)\times(101\times96\times96)\times(77\times72\times72\times67\times63\times58)\times(31\times29\times27\times27\times26\times26\times24)\times(201\times134\times112\times96\times96\times87\times81\times77\times67)\times(29\times27\times26\times23\times22\times17\times17)\times(77\times69\times65\times59)\times(401\times223\times201\times154\times134)\times(45\times41\times38\times33)=\mathbf{1.56\times10^{108}}$. This enormous number renders the enumeration technique intractable.

To assess the effectiveness of the proposed GA, ten millions random chromosomes where generated and the percentage of the feasible chromosomes was less than 0.0002%. This indicates clearly that the problem is fairly constrained to the extent that only about 2 in a million randomly generated chromosomes are feasible (the budget is tight). The average reliability of the 19 feasible random chromosomes found from the ten millions randomly generated chromosomes was 0.4232 which is significantly lower than the worst average reliability found by the proposed GA (population size of 50 and generation number of 1000). Also the S.D. for these randomly feasible chromosomes was 0.1904 with a coefficient of variation of 0.4500 compared to 0.082 and 0.1088 for the worst case found by the proposed GA respectively. Moreover, the time for generating and evaluating the ten millions random chromosomes was 383720 seconds (using a machine with the following specifications: Manufacturer HP, Model HPE-500f, Processor AMD phenon (tn) IIX6 1045T processor 2.70GHz, RAM 8.0 GB, system 64-bit operating system). compared with 93 seconds per replication for the GA on the same machine. This clearly indicates that the proposed GA is very effective in terms of solution quality and computational time compared to enumeration.

Table 2 clearly indicates that as the number of generations and the size of the population increases, the average reliability increases and the S.D decreases and hence the coefficient of variance decreases. Furthermore, the table also shows that the population size is more important than the number of generations for the performance of the proposed GA. For example, using 200 chromosomes and 1000 generations gives better results in terms of average reliability and coefficient of variation than using 100 chromosomes and 2000 generations. In addition, the table also indicates that at the same population size, as the number of generations increases the average reliability and coefficient of variations deceases.

To assess the convergence of the algorithm, a small problem was solved by enumeration and GA. Table 3 shows the relevant information for the problem. The budget for this problem was $280.

Table 4. Relevant information for the small problem

Manuf.	Subsys1		Subsys2		Subsys3	
	C	R	C	R	C	R
1	11	0.55	12	0.58	17	0.49
2	13	0.59			22	0.53
3	15	0.62				

The number of possible combinations for this problem evaluated by equation (6) was 5.7×10^7. The problem was solved by GA using population size of 50 and 1000 generations with 100 replications. The results obtained for the reliability of the system by enumeration was 0.9409 while for the average 100 replications of the proposed GA was 0.9405. The S.D for the 100 replications was 0.0017 with coefficients of variance of 0.0018. Out of the 100 replications, GA found the optimal solution 88 times. This means that GA was capable to converge to the optimal value 88% of the time in this problem.

Figure 8 shows the shopping policy (chromosome) corresponds to the optimal solution found for this problem.

2	3	3	8	0	0	2	2	0

Figure 8. Optimal shopping policy for the small problem

6. Conclusions

The problem of selecting the optimal shopping policy of products from many available manufacturers for 1-out-of-n active redundancy series systems under budget constraint was formulated and tested using GA. The study showed that the number of possible combinations for this problem can be very high from which the majority of the possible solutions are infeasible, the matter that renders the enumeration technique ineffective or even practically impossible.

The results showed that the proposed algorithm has high degree of robustness. Moreover, the results showed that the proposed algorithm is superior to the enumeration technique in terms of both computational time and quality of solution. Furthermore, the results showed that convergence of the algorithm to the optimal solution is high.

Acknowledgments

The author is grateful to the Applied Science Private University, Amman, Jordan, for the financial support granted to this research (Grant No. DRGS-.)

References

Cao, D., Murat, A., & Babu, R. (2013). Efficient exact optimization of multi-objective redundancy allocation problems in series-parallel systems. *Reliability Engineering and System Safety, 111*, 154-163. http://dx.doi.org/10.1016/j.ress.2012.09.013

Coit, D. (2001). Cold standby redundancy optimization for nonrepairable systems. *IE transactions, 33*, 471-478. http://dx.doi.org/10.1080/07408170108936846

Coit, D., & Smith. A. (1994). Reliability optimization of series-parallel systems using a genetic algorithm. *IEEE Trans. Reliability, 45*(2), 254-260. http://dx.doi.org/10.1109/24.510811

David, W., & Alice, E. (1996). Solving the redundancy allocation problem using a combined neural network/genetic algorithm approach. *Computers Ops Res., 23*(6), 515-526. http://dx.doi.org/10.1016/0305-0548(95)00056-9

Feller, W. (1968). An introduction to probability theory. Wiley, New York.

Fyffe, D., Hines, W., & Lee, N. (1968). System reliability allocation and a computational algorithm. *IEEE Trans. Reliability, R17*, 64-69. http://dx.doi.org/10.1109/TR.1968.5217517

Gen, M., Ida K., & Lee, J. (1990). A computational algorithm for solving 0-1 goal programming with GUB structures and its applications for optimization problems in system reliability. *Electron. Commun. Jap., 73*, 88-96. http://dx.doi.org/10.1002/ecjc.4430731210

Ida, K., Gen, M., & Tokota, T. (1994). *System reliability optimization with several failure modes by genetic algorithm*. Proc. 16th Int. Conf. Computers and Industrial Engineering, 349-352.

Kulturel-Konak, S., Coit, D., & Baheranwala, F. (2008). Pruned pareto-optimal sets for the system redundancy allocation problem based on multiple prioritized objectives. *Journal of Heuristics, 14*(4), 335–357. http://dx.doi.org/10.1007/s10732-007-9041-3

Liang, Y., & Chen, Y. (2007). Redundancy allocation of series – parallel systems using a variable neighborhood search algorithm. *Reliability engineering and system safety, 92,* 323-331. http://dx.doi.org/10.1016/j.ress.2006.04.013

Liang, Y., & Smith, A. (2004). An ant colony optimization algorithm for the redundancy allocation problem (RAP). *IEEE Transactions on Reliability, 53*(3), 417-423. http://dx.doi.org/10.1109/TR.2004.832816

Misra, K., & Sharma, U. (1991). An efficient algorithm to solve integer programming problems arising in system reliability design. IEEE Trans. *Reliability, 40*, 81-91. http://dx.doi.org/10.1109/24.75341

Mori, B., Castro, H. De., & Cavalca, K. (2007). Development of hybrid algorithm based on simulated annealing and genetic algorithm to reliability redundancy optimization. *International Journal of Quality and Reliability Management, 24*, 972-987. http://dx.doi.org/10.1108/02656710710826225

Painton, L., & Campbell, J. (1994). *Identification of components to optimize improvements in system reliability.* Proc. DRA PSAM_II Conf. System-based Methods for the Design and Operational of Technological Systems and Processes, 10-15 to 10-20.

Ramadan, S. (2013). *Reducing premature convergence problem in genetic algorithm: Application on travel salesman problem*. Accepted July 12, 2012 in Computer and Information Science, will be published in Vol. 6, No. 1.

Taboada, H., & Coit, D. (2012). A new multiple objective evolutionary algorithm for reliability optimization of series-parallel systems. *International Journal of Applied Evolutionary Computation, 3*(2), 1-18. http://dx.doi.org/10.4018/jaec.2012040101

Tian, Z., & Zuo, M. (2006). Redundancy allocation for multi-state systems using physical programming and genetic algorithms. *Reliability Engineering & System Safety, 91*, 1049–1056. http://dx.doi.org/10.1016/j.ress.2005.11.039

Tillman, F., Hwang, C., & Kuo, K. (1980). *Optimization of System Reliability*. Marcel Dekker.

Tillman, F., Hwang, C., & Kuo, W. (1977). Optimization techniques for system reliability with redundancy-a review. IEEE Trans. *Reliability, R-26*, 148-155. http://dx.doi.org/10.1109/24.510811

You, P., & Chen, T. (2005). Efficient heuristic for series – parallel redundant reliability problems. *Computers & operations research, 32*, 2117-2127. http://dx.doi.org/10.1016/j.cor.2004.02.003

Zia, L., & Coit, D. (2010). Redundancy allocation for series-parallel systems using a column generation approach. *IEEE Transactions on Reliability, 59*, 706-717. http://dx.doi.org/10.1109/TR.2010.2085530