Design and Development of Genetic Algorithm for Test Interval Optimization of Safety Critical System for a Nuclear Power Plant

Molly Mehra, M.L. Jayalal, A. John Arul,
S.Rajeswari, K. Kuriakose, S.A.V. Satya Murty
Indira Gandhi Centre for Atomic Research
Kalpakkam, Tamil Nadu
India PIN:603102
mollymehra@igcar.gov.in

Abstract—**With the increase in Nuclear Power Plant (NPP) operating experience, the importance of effectively scheduling the maintenance activities has been recognized as it decreases the testing and maintenance costs without compromising the plant safety. Surveillance Tests are vital for safety critical systems of nuclear plants that need regular maintenance for ensuring reliable functioning. Deciding the value of Surveillance Test Interval forms an optimization problem where two separate cases can be considered. First one is the cost minimization while the performance or unavailability is constrained to be at a given level. The second case is the maximization of availability or performance, for the given cost level. Genetic Algorithm (GA) is applied to solve the model to get the global optimized maintenance strategy. The results obtained are validated with the reference study results.**

Keywords- Genetic Algorithm; Nuclear Power Plants; Safety Grade Decay Heat Removal System; Simple Genetic Algorithm; Steady State Genetic Algorithm; Prototype Fast Breeder Reactor

I. INTRODUCTION

Genetic Algorithm (GA) is adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetics. It represents an intelligent exploitation of a random search within a defined search space to solve optimization and search problems. Since the advent of GA in early 1970s it has been employed in many application domains. Changes to the implementation of the basic GA have been pursued by researchers to improve its performance. A particular method may have advantage over another for a particular domain of application. GA has been successfully employed for Test Interval optimization in many plants and continues to apply to existing and upcoming NPPs. Varied approaches have been adopted and suggested in work relating to this domain. Here, a general structure for GA is sought which is most effective in attacking such problems.

A 500 MWe capacity sodium cooled pool type Prototype Fast Breeder Reactor (PFBR) project has been designed by Indira Gandhi Centre for Atomic Research (IGCAR) which is under construction at Kalpakkam. For PFBR Test Interval data has been given from experience of previous fast reactors, this value can be improved with further cost reduction without affecting the unavailability.

Test Interval Optimization in Fast Reactor domain is being done for the first time and we have selected the Safety Grade Decay Heat Removal (SGDHR) system of Prototype Fast Breeder Reactor (PFBR) for our study. The system comprises of many components; with redundancy in design for increased component reliability. Test Intervals for all components are optimized such that the minimum cost is involved in testing without affecting the safety of plant. This is a multimodal and multiobjective problem that justifies the choice of Genetic Algorithm (GA) over other traditional search techniques like hill climbing that have the limitation of getting trapped in local optima [1]. The problem is complex and the solution space is very large. Normal GA techniques do not suffice in providing a global feasible solution. Therefore, different flavors of GA and different operator techniques have been implemented and compared. Here, two categories of GA are considered – Simple Genetic Algorithm (SGA) and Steady State Genetic Algorithm (SSGA). GA operators like- one point, two point and multipoint crossover are explored. Performances of different selection operators like- Roulette Wheel, Tournament Selection are compared. Coding is carried out in C++ using Object Oriented Methodology.

II. PROBLEM DESCRIPTION

SGDHR is a passive standby system of PFBR for decay heat removal which is called into action when the normal active heat removal path, through Operational Grade Decay Heat Removal (OGDHR) system is unavailable. For successful decay heat removal both Primary Heat Transport and SGDHR should work. So, the components in primary sodium circuit (like pump, motors), secondary sodium circuit (Direct Heat Exchanger, piping, valves) and air circuit (Air Heat Exchanger, dampers) need to undergo periodic testing and maintenance to guarantee their availability when actual demand on the system comes. The reliability analysis was done [2] and fault trees [3] for both Primary Circuit and Intermediate air circuit were considered.

III. Equation Formulation

Models for unavailability and cost of testing of individual components are established with Surveillance Test Interval as decision variable. Reliability parameters like- standby failure rate, per demand failure probability, mean time to test, testing cost per hour etc. are considered for modeling. The list of symbols used to represent reliability parameters and their meanings are given in Table 1. The cost of testing and maintenance of whole system is found by adding individual component costs and the system unavailability is found by adding the minimal cutsets derived from the fault tree analysis of the system. These models for system cost and unavailability serve as objective functions that have to be minimized by deciding on the values of test and maintenance intervals.

TABLE I. Reliability Related Symbols Used

Symbol	Meaning
Ti	Surveillance test interval
T	Mean time to test
λ	Standby failure rate
T_R	Mean time to repair
C_{ht}	Surveillance Testing cost per hour
C_{hr}	Cost of repair per hour

The unavailability equation for SGDHR system was formulated as:

$$u_i\ x\ = \lambda\,(T_i/2\ + T_R) \qquad (1)$$

where $u_i(x)$ represents unavailability of component that depends on the vector of decision variables x. Total unavailability was found from the cut-set equations obtained from the fault tree analysis. System unavailability is sum of j number of minimal cut sets and the product k extends to the number of basic events in the j^{th} cut set as:

$$U\ x\ = \ _{j}\ \ _{k}u_{jk}(x) \qquad (2)$$

where u_{jk} represents the unavailability associated with the basic event k belonging to minimal cut set number j. The cost model is established as:

$$c_i\ x\ = \frac{t}{T_i}C_{ht} + \lambda T_R C_{hr} \qquad (3)$$

The total yearly cost of the system having i number of components is given by:

$$C\ x\ = \ _{i}c_i\,(x) \qquad (4)$$

The problem is solved using GA for two cases: Case 1: Keeping the cost as objective function to be minimized and unavailability as constraint. That is represented as:

$$Min(\ _{i=1}^{n}c_i\,(x)) \qquad (5)$$

$$U(x) \le MaxRisk \qquad (6)$$

Case 2: Keeping the unavailability as objective function to be minimized and cost as constraint. That is represented as:

$$Min(U(x)) \qquad (7)$$

$$\ _{i=1}^{n}c_i\ x\ \le MaxCost \qquad (8)$$

For the high redundancy systems like SGDHR, data for large number of simultaneous failures does not exist. So, a common cause analysis is done with beta factor model, in which the unavailability of a single component is multiplied by some value of beta depending on the number of such redundant components. The approach followed in this study is that active components with levels of redundancy less than or equal to three, a beta of 5% is used. If redundancy is greater than or equal to four, a beta of 1% is used.

IV. Genetic Algorithm Design

In a genetic algorithm, many individual solutions are randomly generated to form an initial population. This population then evolves over successive generations to give better solutions. Each generation is comprised of various phases, the most important being – fitness evaluation, selection (competition), reproduction (cross-over) and mutation [4]. Fitness evaluation is the step in which the quality of an individual is assessed. Selection is an operation used to decide which individuals to use for reproduction and mutation in order to produce new search points. Reproduction is the process by which the genetic material in two or more parent individuals is combined to obtain one or more offspring. Mutation is normally applied to one individual in order to produce a new version of it where some of the original genetic material has been randomly changed.

An individual is represented as a string of numbers known as a chromosome. Chromosomes are composed of genes where each gene is a set of values called alleles that represents an encoded decision variable. The binary encoding scheme of the decision variables is used here for test interval optimization, due to its simplicity in mutation operation and also because the range constraint is automatically implicit [5].

Goldberg DE [4] suggested that good GA performance requires the choice of high cross over probability, low mutation probability and a moderate population size. For all the experiments, crossover rate of 0.6 and mutation rate of 0.03 was taken. Population size was taken as 100 for SGA and 80 for SSGA with a replacement size of 20. The GA is implemented for TI optimization with the following methods and operators:

A. *Fitness Scaling*

Fitness Scaling was introduced to improve the performance of GA by controlling the copies of individuals during the beginning of run and as the run matures.

B. Elitism

This was implemented to retain the best individual of a generation in the next generation so that highest fitness solutions are not lost in reproduction and mutation.

C. Penalization

Optimization of test schedules is done by taking either cost or unavailability as the objective function and the other as the constraint. Constraint Implementation has been done by converting this problem into a maximizing one by taking the fitness function as the reciprocal of the actual function value i.e. to be minimized. If a particular solution vector violates the constraint its fitness is reduced using the penalize function as suggested by Martorell[6].

D. Fitness evaluation

An individual is decoded and its cost and unavailability are found using the models in eq. 2. and eq. 4. Fitness is evaluated as the inverse of cost or unavailability depending on which one is taken as the objective function.

E. Selection

For the purpose of comparison the following selection schemes were considered:

1) *Roulette-wheel (RW):* It is a sampling method that picks the individuals by simulating the roulette-wheel for fitness proportionate selection.

2) *Tournament selection:* This involves running several "tournaments" among a few individuals chosen at random from the population. The winner of each tournament (the one with the best fitness) is selected for crossover.

3) *Hybrid selection:* It takes two individuals by fitness proportionate selection and then chooses the best one among them for crossover. This increases the selection pressure on individuals.

F. Mutation

Here, mutation is performed on a bit-by-bit basis. In binary encoding this simply means changing a 1 to a 0 and vice versa. By itself, mutation is a random walk through the string space. When used sparingly with reproduction and crossover, it is an insurance policy against premature loss of important notions.

G. Crossover

One point and multipoint crossover differ in the number of sites, single or many, chosen for exchanging information between two individuals. As the number of points (sites) increases for crossover the exchange of information between individuals takes place over the whole string length at various places. This increases the probability of finding better individuals by reproduction.

V. Flavors of Genetic Algorithm

For this study, the Genetic Algorithm has been implemented in two different ways –SGA and SSGA – to make a comparison of their performance.

A. Simple Genetic Algorithm (SGA):

An initial population of solutions is generated and evaluated. Then a selection process chooses individuals for crossover and mutation. A new population is formed by reproduction of the selected individuals. This new population has better solutions than those of previous generation. The process is repeated till an optimal solution is found.

B. Steady State Genetic Algorithm (SSGA):

In SSGA the whole population does not undergo transformation at each generation; instead an auxiliary population of size *nrepl* is generated and included in the base population. After this, the worst *nrepl* individuals are excluded from the population based on their fitness evaluation. In this way some part of the base- population is carried to next generation without being affected. Hence SSGA allows more exploration by retaining some low fitness solutions which would otherwise be lost in a single generation owing to high selective pressure and reproduction like in SGA.

VI. Results And Discussions

A. SGA and SSGA

The average fitness for a particular generation can be obtained from individual fitness score assigned in the fitness evaluation stage. The average fitness versus generation for SGA and SSGA were plotted and it was found that SGA converges very fast to some optimum value and the average fitness of the population does not change much after that (Fig. 1.). Whereas, for SSGA, the optimum value is found in later generations and average of population keep increasing making the probability of production of a better individual higher.

The Maximum Fitness value for each generation represents the individual that has the highest fitness score got in the fitness evaluation stage. As shown in Fig.2. for more number of generations SSGA converges to a better optimum than that of SGA in same number of generations. The SGA approaches to a higher value at the initial generations itself but unable to improve further as generations are evolving.

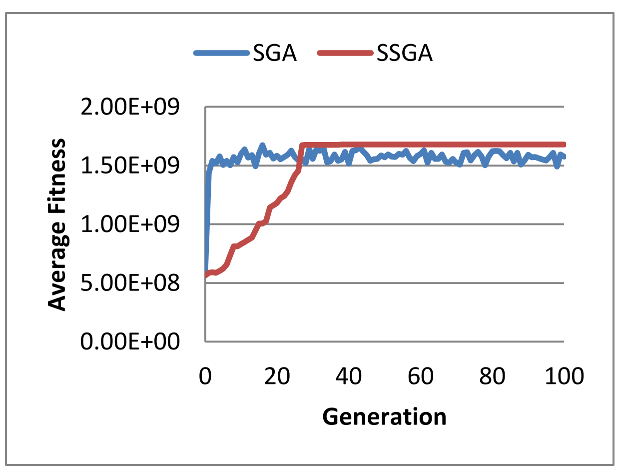

Figure 1. Average Fitness Vs Generations for SGA and SSGA.

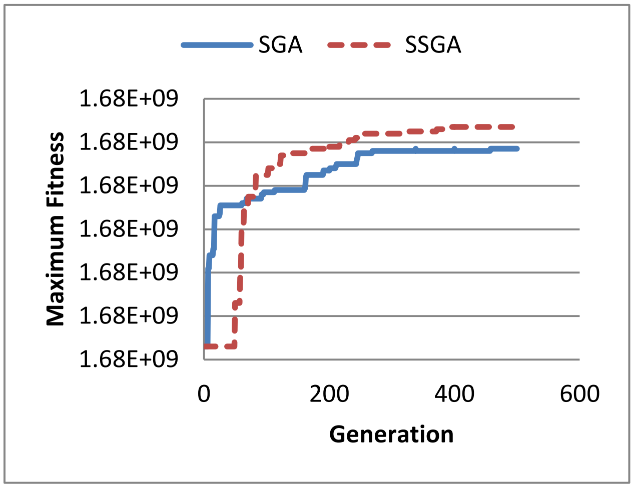

Figure 2. Maximum Fitness Vs Generations for SGA and SSGA.

This is due to more exploration and lesser exploitation in the earlier generations of SSGA where only a few individuals are allowed to undergo reproduction and most of the individuals are retained as it is, rather than replacing the whole population with super individuals. SSGA's effect on the dynamics of GA was analyzed in this study. SSGA improves the efficiency of GA for some problems like the one that we selected fro our study.

B. *Roulette Wheel, Tournament and Hybrid Selection*

Roulette Wheel, Tournament and Hybrid selection were implemented with SSGA. The results do not vary much in terms of maximum fitness solutions produced by the three selection techniques (Fig. 3.).

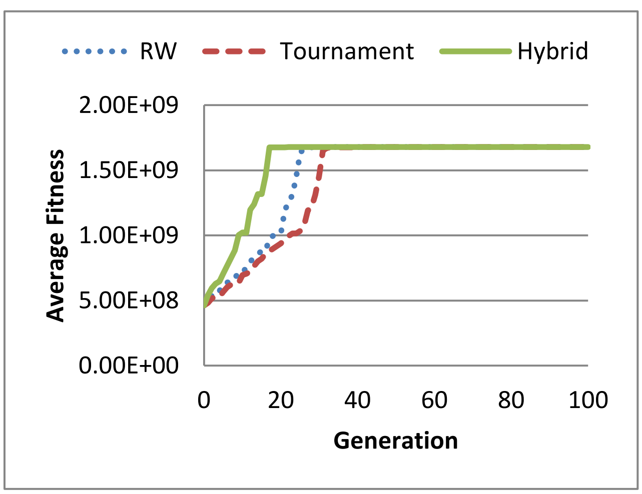

Figure 3. Maximum Fitness Vs Generations for Roulette Wheel, Tournament and Hybrid Selection.

In the case of average fitness, the evolution of fitness is faster with hybrid selection than other two selections i.e. it gives better result in early stages (Fig. 4.). Hence we can say that the overall performance of hybrid selection method is better for our problem.

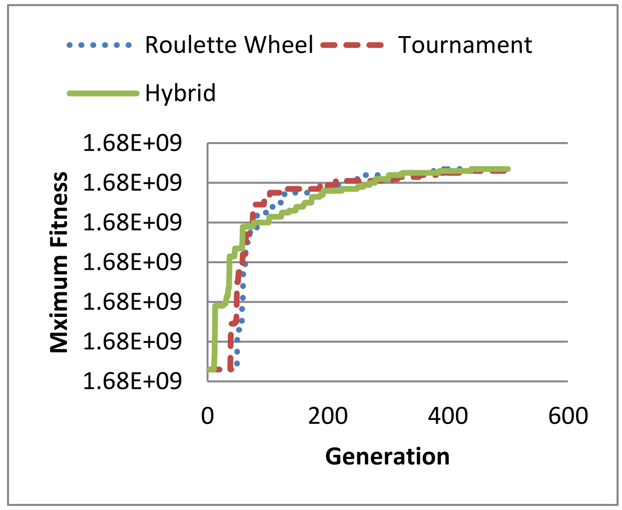

Figure 4. Average Fitness Vs Generations for Roulette Wheel, Tournament and Hybrid Selection.

VII. Conclusion

Here, we have considered the problem of deciding test intervals for a safety critical system of PFBR, wherein the test strategy for the plant is improved such that unnecessary testing burdens are reduced without compromising the plant safety. The reliability parameters values were taken from an internal report [2] and serve as input data for solving the unavailability and cost equations (eq. (2) and (4)). Two separate optimization cases are considered namely cost minimization and availability maximization. The optimization is done using Genetic Algorithms which takes cost or availability as the objective function and solves for the set of best test interval values for all components. We have done a comparative study on two different implementations of GA namely SGA and SSGA. From the performance comparison, it is found that SSGA is suitable for the selected problem. Then different selection techniques of GA are evaluated with in the selected SSGA. SSGA with hybrid selection is found to perform better than other techniques in this complex problem domain with large population size.

Although the current problem considers only SGDHR of PFBR, this study can be extended for other safety critical systems of Nuclear Power Plants and also can be extended to include other Technical Specifications.

References

[1] Haupt R.L and Haupt S.E, "Practical Genetic Algorithms," John Wiley & Sons, 1998.

[2] Confidential Internal Report: "Probabbilistic Safety Assesment, Level 1: Internal Events for PFBR, System Reliability Analysis" Volume II, April 2011.

[3] Confidential Internal Report: "Probabbilistic Safety Assesment, Level 1: Internal Events for PFBR, Event Tree and Cutsets," Volume III & Systems Basic Events Volume IV, April 2011.

[4] Goldberg D.E, “Genetic Algorithms in Search Optimization and Machine Learning,” Addison-Wesley Publishing Company, 1989.

[5] Michalewicz Z, “Genetic Algorithm + Data Structure = Evolution Programs,” Spinger-Verlag, New York, 1994.

[6] Martorell S, Carlos S, Sanchez A, Serradell V. “Constrained optimization of test intervals using steady-state genetic algorithms: