INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

# Processor Scheduling Algorithms in Environment of Genetics

**Randeep**

Department of Computer Science and Engineering
R.N. College of Engg. & Technology
Haryana, India
randeepravish@gmail.com

***Abstract*— *The efficiency of the operating system of a computer depends on many contributing factors. The process scheduling or task scheduling for processor or CPU is the main factor that mostly affects the efficiency of CPU. Many types of scheduling algorithm is used for CPU scheduling. Each scheduling algorithm, allocates the processor to processes on the basis of some type of scheme to increase the efficiency of system. In this paper genetic algorithm is used at the places of other scheduling algorithms to increase the efficiency of system. In genetic algorithm different types of encoding, crossover and mutation operations are used. Firstly, we find out the minimum waiting time for any number of processes using genetic algorithm than compare it to the waiting time of First Come First Serve and Shortest Job First scheduling algorithm.*

***Keywords*—*NP Hard Problems, CPU scheduling, Genetic Algorithm, Encoding, Mutation, Crossover, Selection.***

## 1. INTRODUCTION

Processor Scheduling Problems are a classical combinatorial problem that is problems on the selections and arrangements of elements of a finite set. These problems fall under the category of NP-complete [2] problems. A set of finite elements is called Search Space. Various types of algorithms are used for scheduling the processor. With the help of these scheduling algorithms we find the optimal result. Using these algorithms waiting time, burst time etc. is calculated. Among these First-Come First-Served (FCFS) Scheduling, Shortest-Job-First (SJF) Scheduling, Priority Scheduling, Round Robin (RR) Scheduling, Multilevel Queue Scheduling are of much importance and are widely used for scheduling of jobs in a processor. Evolutionary algorithms (EA" s) are population-based optimization algorithms that uses biology-inspired mechanisms and terms like encoding, mutation, crossover, natural selection, and survival of the fittest in order to refine a set of solution candidates iteratively [BAC97, BAC96]. Genetic algorithm is one of the most used techniques of evolutionary algorithm. In genetic algorithm different type of population, operators like selection, crossover, mutation and reproduction are used. These operators are the very basic elements of genetic algorithm.

This paper is an effort to develop a simple general algorithm (genetic algorithm based) for obtaining optimal or near optimal schedules for Single Processor Scheduling Problems with minimum computation effort even for large sized problems and comparing the minimum average waiting time using different scheduling algorithms (like FCFS, SJF and scheduling using genetic algorithm

## 2. OPTIMIZATION WITH GENETIC APPROACH

John Holland first proposed genetic algorithms in the 1960s. The genetic algorithm is a search algorithm based on the mechanics of natural selection and natural genetics [6]. As summarized by Tomassini [5], the main idea is that, in order for a population of individuals to adapt to some environment, it should behave like a natural system. Using Tomassini's terms [5], genetic algorithms (GA" s) consider an optimization problem as the environment where feasible solutions are the individuals living in that environment.

### A. *BASICS OF OPTIMIZATION*

In the field of mathematics and computer science, optimization (alternatively, optimization or mathematical programming) is the selection of a best element (with regard to some criteria) from some set of available alternatives. This means solving problems in which one want to minimize or maximize a real valued function by systematically choosing the values of real or integer variables from within a possible set. Using a scalar or real-valued objective function is the simplest example. It means to find the best available values of some objective function given a well-defined domain, including a variety of different types of objective functions and different types of domains.

An optimization problem can be shown in the following way:

Given*:* a function $f$: $A \longrightarrow$R from some set $A$ of some elements to the real numbers.

Sought*:* an element $x_0$ in $A$ such that $f(x_0) \le f(x)$ for all $x$ in $A$ ("minimization") or such that $f(x_0) \ge f(x)$ for all $x$ in $A$ ("maximization").

This type of representation is called an optimization problem or a mathematical programming problem.

Many real-world and theoretical based problems can be solved in this type of framework. Problems formulated using this technique in the fields of management and computer vision may refer to the technique as cost minimization and profit maximization. In cost minimization, minimized cost of a product or function is calculated. On the other hand, maximized profit is calculated in profit maximization.

### *INTRODUCTION TO GENETIC ALGORITHMS*

A genetic algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems [3]. GA is described as a heuristic method which is based on the survival of the fittest. It is a particular class of evolutionary algorithm (EA). Genetic algorithm uses its basic term from biology science like mutation, genes, chromosomes etc. GA is used to search the search space for the best solution like minimum or maximum. It is discovered as a useful tool for search and optimization

### B. *CPU Scheduling algorithms and its types*

CPU scheduling is used for multiprogramming operating system. In multiprogramming operating system, more than one process is loaded into the memory for processing. Each process shares the same memory and same processor for processing. Here only single processor is taken into consideration. Sharing of CPU and processor is based on different types of scheduling algorithms. Some of these algorithms are following:-

a. *First-Come First-Served Scheduling(FCFS)*

Processes are executed on the basis of their arriving.

b. *Shortest-Job-First Scheduling(SJF)*

Processes are executed on the basis of their execution time.

c. *Priority Scheduling*

Processes are executed on the basis of some type of given priority.

d. *Round Robin Scheduling*

Processes are executed on the basis of time quantum fixed by operating system.

### A Simple Genetic Algorithm

An algorithm is a process of step by step for solving a given problem. A genetic algorithm is a problem solving method that uses genetics as its model of problem solving. It's a search technique to find nearby solutions to optimization and search problems [7]. In GA, a population of given solutions is used. With the help of chromosome, each solution is represented. First of all, all possible solutions are coded into the chromosomes. Then a set of reproduction operators are used. Reproduction operators are implanted directly on the chromosomes, and are used to perform mutations. Appropriate representation and reproduction operators are really something determinant, as the behaviour of the GA is extremely dependant on it [1].

Selection is done by applying a fitness function. Each chromosome has a related value corresponding to the fitness of the solution it represents. The fitness value describes how good the candidate solution is. The optimal solution is the one, which describes the maximize value of the fitness function. Once the reproduction and the fitness function have been appropriately defined, a Genetic Algorithm is evolved according to the same basic structure. It begins by generating an initial population of chromosomes. The gene pool should be as large as possible so that it can take all solutions of the search space. Some best chromosomes are copied into new off springs and the applied different operators. It is called Elitism. The basic cycle of genetic algorithms is shown in Figure 1.
It shows that the genotypes are used in the reproduction operations whereas the values of the objective functions $F_i$ are computed on basis of the phenotypes in the problem space X which are obtained via the genotype-phenotype mapping "gpm".
The basic genetic algorithm is as follows:

• [start]: Genetic random population of n chromosomes (suitable solutions for the problem).
• [Fitness]: Evaluate the fitness f(x) of each chromosome x in the population.
• [New population]: Create a new population by repeating following steps until the new population is complete,
• [selection]: select two parent chromosomes from a population according to their fitness (the better fitness, bigger are the chances to get selected).
• [crossover]: Along with a crossover probability, cross over the parents to form new offspring (children). If crossover was not performed, offspring is the same copy of parents.
• [Mutation]: With a mutation probability, mutate new offspring at each locus (position in chromosome)
• [Accepting]: Place new offspring in the new population.
• [Replace]: Use newly generated population for a further sum of the algorithm.
• [Test]: If the end conditions are satisfied, stop and return with the best solution in current population.
• [Loop]: Go to step2 for fitness evaluation.
This algorithm is stopped when the population meets towards the optimal solution.

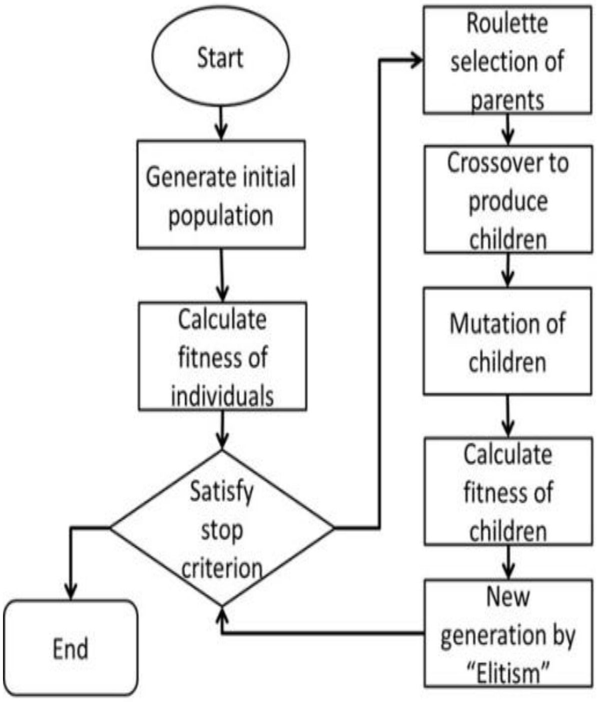

Start
Generate initial population
Calculate fitness of individuals
Satisfy stop criterion
End
Roulette selection of parents
Crossover to produce children
Mutation of children
Calculate fitness of children
New generation by "Elitism"

**Figure1:** Basic Cycle of Genetic Algorithm

## 3. PROBLEM STATEMENT

In this part, all discussion is about the problem. In scheduling of single processor system, each process is processed by the processor. There are a number of processes which are in waiting queue. Each process is processed completely till all processes are processed by the processor. Only a single process is processed by the processor and other processes are in waiting queue. Scheduling of processes follows following rules in single processor system:-

- Processor has to process each and every job, without leaving any process unprocessed.
- After a process is executed completely, processor is allocated to next process. Each process should be processed only once.
- No pre-emption is permitted means processor can't shift between processes before completion of currently executing process.
- The waiting time for all those processes which are in waiting queue should be minimum.

Here the discussion is all about simple processor scheduling algorithms like FCFS, SJF and algorithm which are used in genetic environment. Problem is how came to know about the better scheduling algorithm. It can be found with the help of comparison of scheduling algorithms. As already talked about previously, genetic algorithm is one of the best optimization techniques that can be used to solve the problems of function maximization. It can be said as a search procedure inspired by principles from natural selection and genetics [4]. The operation of the genetic algorithm is very simple. It starts with a population of random individuals. The best individuals survive, mate, and create new offspring, originating a new population of individuals. This process is repeated a number of times, and it leads to better and better individuals.

## 4. GA RELATING OUR PROBLEM DOMAIN

The steps of applying GA relating our problem are following:-

1. Choosing an Encoding scheme.
2. Choosing Fitness function.
3. Choosing Operators.
4. Choosing Parameters.
5. Choosing an Initialization method and Stopping criteria.

A. ***Encoding Schemes***

Genetic algorithm's application starts with encoding. There are many types of encoding schemes which are used in genetic algorithm. Various types of encoding schemes are:

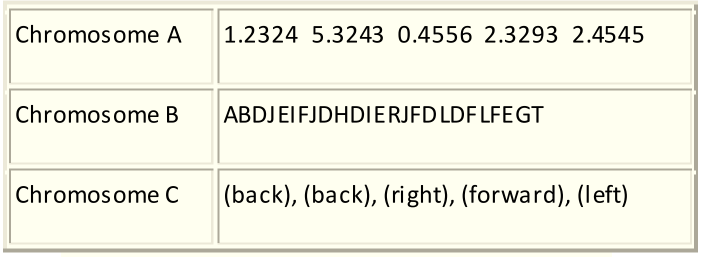
1. Binary encoding
2. Permutation encoding
3. Value/Real encoding
4. Tree encoding
5. Octal encoding &
6. Hexadecimal encoding.

Value encoding is required for our problem.

**Real/Value encoding**

Every chromosome describes a string of values and the values can be anything related to the problem. This encoding makes best results for some special problems. Direct value encoding can be used in some complicated value problems, such as real numbers. Use of binary encoding for this type of problems would be very difficult.

**Figure 2: Value encoding [9]**



| | |
|---|---|
| Chromosome A | 1.2324 5.3243 0.4556 2.3293 2.4545 |
| Chromosome B | ABDJEIFJDHDIERJFDLDFLFEGT |
| Chromosome C | (back), (back), (right), (forward), (left) |

Some examples of encoded chromosomes relating to our problem are shown below:

[1 7 3 2 4 5 6 10 9 8]

[12 10 8 11 6 4 3 2 5 1 7 9]

After this fitness function is selected.

B. ***Fitness function***

The next step is to specify a function that can assign a score to any possible solution. The score is a numerical value that shows how well a particular solution solves the problem. The score describes the fitness of the individual solution .The task of the GA is to discover solutions that have higher fitness values among the set of all possible solutions. For a solution (Chromosome), the fitness function must return a single numerical value, which is proportional to the "fitness" of that solution. Our aim is to find individual having minimum average waiting time. So, the fitness function of a Solution Si is given by

Fitness (Si) = ———

Where wj shows the waiting time of the process j.

n is the total no. of processes.

*C. **OPERATORS***

Once the encoding and the fitness function are defined, the implementor has to choose selection and genetic operators to evolve to new solutions to the problem being solved. The selection operator simulates "Survival of the fittest". There are many way to implement this type of operator.

**Selection**

Selection is a process in which two parents are selected from population for crossing. This selection is done on the basis of their evaluation function and two chromosomes are selected. Selection has to be balanced in respect to variation form crossover and mutation. Too strong selection signifies sub optimal highly fit individuals will take over the population, and it reduces the diversity needed for change and progress, too weak selection will result in too slow evolution. The Figure 3 shows the basic selection process.

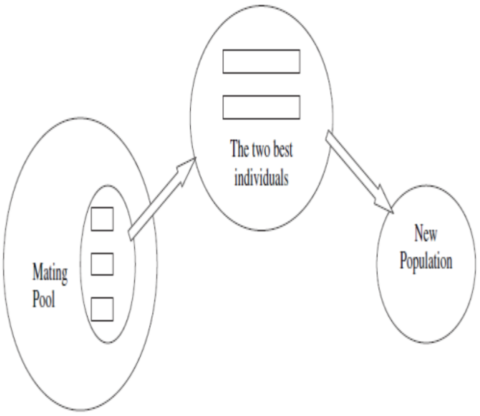


**Figure 3:** Selection [7]

**Roulette wheel selection**

Various selection methods are used. Roulette wheel selection method is used here. The principle of roulette wheel selection is a linear search through a roulette wheel with the slots in the wheel weighted in proportion to the individual's fitness values. A target value is set, which is a random proportion of the sum of the finesses in the population. The population is stepped through until the target value is reached. The expected value of an individual is that fitness divided by the actual fitness of the population. A slice of the roulette wheel is assigned to each individual, the size of the slice being proportional to the individual's fitness. Here N is the number of individuals in the population so the wheel is spun N times. On each spin, the individual under the wheel's marker is selected to be parents for the next generation.

This method is implemented as follows:

1. Total expected value of the individuals in the population is summed. Let it be T.
2. Repeat N times:
    - A random integer 'r' between o and T is Chosen.
    - Loop through the individuals in the population, summing the expected values, until the sum is greater than or equal to 'r' the individual whose expected value puts the sum over this limit is the one selected.

Roulette wheel selection is easier to implement but is noisy. The rate of evolution depends on the variance of fitness is in the population.

**Crossover**

Crossover is the process of choosing two parent chromosomes and producing from them a number of off spring is [7].

Crossover is a recombination operator that works in three steps:

i. The reproduction operator selects at random a pair of two individual strings for the mating/crossover.

ii. A cross site or cross point is selected at random along the string length.

iii. Finally, the position values are swapped between the two strings following the cross site.

**Ordered crossover**

Ordered two-point crossover is applied when the problem is based on order, for example in U-shaped assembly line balancing [7]. Two random crossover points are selected for partitioning from two parent chromosomes and divided into a left, middle and right portion. The ordered two-point crossover is done in the following way: left and right part of the first child is same to the parent 1. Only middle part is opposite to parent 1. The same process is repeated for child 2 with parent 2[7].

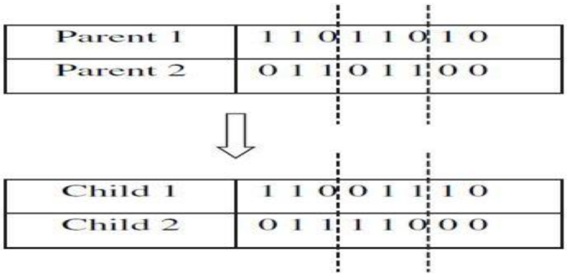


**Figure 4:** Ordered crossover

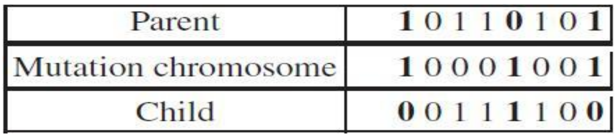
**Mutation**

The benefit of mutation is to recover the lost genetic materials as well as for randomly distributing genetic information. Mutation of a bit is done by flipping of bit, changing 0 to 1 and vice-versa. There are a number of mutation schemes such as:

- Flipping
- Interchanging
- Reversing
- Uniform mutation &
- Non-uniform mutation.
-

**Flipping**

Flipping of a bit is done by changing 0 to 1 and 1 to 0 based on a mutation chromosome generated. The Figure 5 below explains mutation-flipping concept.

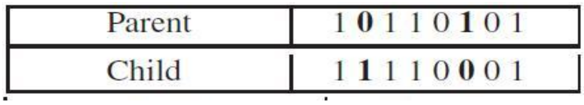


| | |
|---|---|
| Parent | 10110101 |
| Mutation chromosome | 10001001 |
| Child | 00111100 |

**Figure 5:** Mutation Flipping [7]

**Interchanging**

Two random positions of the string are picked and the bits corresponding to those positions are interchanged. This is shown below in Figure 6.

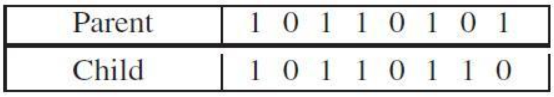


| | |
|---|---|
| Parent | 10110101 |
| Child | 11110001 |

**Figure 6:** Mutation Interchanging [7]

**Reversing**

A random position is picked and the bits next to that position are reversed and child chromosome is produced. This is shown below in Figure 7.



| | |
|---|---|
| Parent | 1 0 1 1 0 1 0 1 |
| Child | 1 0 1 1 0 1 1 0 |

**Figure 7:** Reversing [7]

**For our problem, interchanging is used.**

*D. Parameters*

On the basis encoding, a fitness function, and operators, the GA is ready to enter in execution. But before doing that, the user has to specify a number of parameters such as population size, no. of processes etc. The population diversity is maintained is maintaining by operator probabilities.

The probability of crossover operation is more than to mutation operation. Crossover operation has a probability of 100% whereas mutation operation mutation is done after every five consecutive algorithm runs [4].

*E. Initialization method & stopping criteria*

The genetic algorithm is generally initialized with a population of random individuals. After initialization, each individual is measured according to the user's specified fitness function. Thereafter, the GA simulates evolution on the artificial population of solutions using operators that imitate the survival-of-the-fittest and principles of natural genetics such as recombination and mutation [4].

A number of criterions can be used for this purpose.For our problem, stopping criteria used is no. of iterations and initialization method is random generation of population.

## 5. RESULT

Three algorithm of CPU scheduling are used. FCFS and SJF are normal processor scheduling algorithms. Genetic algorithm is also used which is a part of EA. GA starts with the random selection of population from search space. After this encoding of input population is used. Selection technique and crossover is performed after encoding. Mutation operation is executed after every five iterations and repeated for required no. of iteration.

The below bar graph shown the result of three algorithms which are FCFS, SJF and GA in the term of average **waiting time.** The result which came from GA is very near to the result of SJF and somewhere is less good than to SJF.

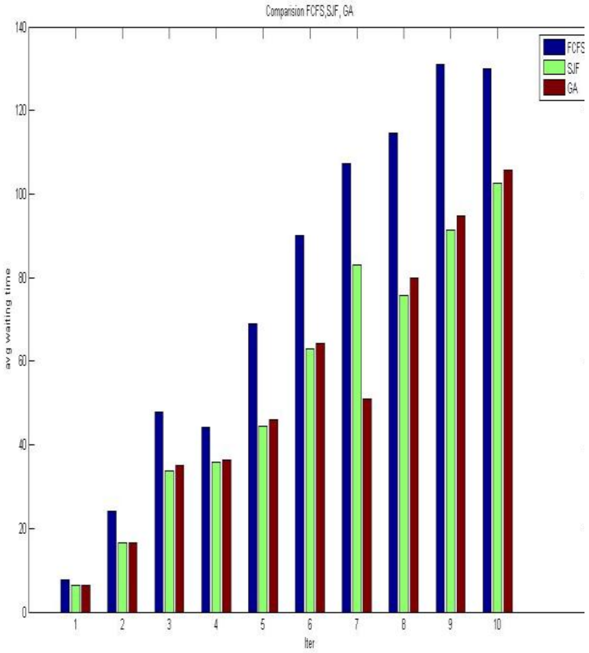


**Figure 8:** Results of various scheduling algorithms (Average Waiting Time)

## 6. CONCLUSION

The main motive of this paper is to optimize or minimize the waiting time. The main focus of this paper is on the scheduling algorithm which works in genetic environment and to show that GA is better than to FCFS and SJF in term of waiting time after a number of different iterations. Basically GA is one of the better function optimization methods generally employed now days. Population is randomly generated in GA. The encoding scheme for this problem of function maximization is value/real encoding. After this crossover is done on chrosomes for getting new offsprings.At last interchanging mutation is used while implementing this algorithm.

In this dissertation, we mainly solve the problem of processor scheduling in the environment of genetics. The algorithm of GA gives nearby same result as SJF gives. After all the experimentation and implementation, results that came out are like, best one is genetic algorithm.

## REFERENCES

**[1] Davis, L. (1991). Handbook of Genetic Algorithm. Von Nostrand Reinhold, Newyork**

**[2] Garey, M. & Johnson, D. (1979). Computers and Intractability: a theory of NP-Completeness. W.H.Freeman, San Francisco Proceedings of the Second International Conference on Genetic Algorithms, pp. 252-256**

**[3] Back, T., Fogel, David B. & Michalewicz, Z. (Eds.) (1997). Handbook of Evolutionary Computation. Computational Intelligence, Library Oxford University Press in cooperation with the Institute of Physics Publishing / CRC Press, Bristol, New York, ring bound edition. ISBN: 0-7503-0392-1, 978-0-75030-392-7, 0-7503-0895-8, 978-0-75030-895-3. [Also available online at:http://books.google.de/books?id=n5nuiIZvmpAC]**

**[4] Lobo, Fernando Miguel (2000).The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic algorithm operation. International Conference on Genetic Algorithms, in Lisboa.**

**[5] Tomassini, M. (1999). Parallel and Distributed Evolutionary Algorithms: A Review. In Miettinen, K., Makela, M., & Periaux, J. (Eds.), Evolutionary Algorithms in Engineering and Computer Science (pp. 113 - 133). Chichester: J. Wiley and Sons.**

**[6] Gldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Boston: Addison-Wesley**

**[7] Sivanandam, S. N. & Deepa, S. N. (2008). Introduction to Genetic Algorithms. Springer**