ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

Optimized Processor Scheduling Algorithms using Genetic Algorithm Approach

Mohita Gupta[1], Savita Gupta[2]

CSE Department, SKIET, Kurukshetra, India [1]

CSE Department, SKIET, Kurukshetra, India [2]

Abstract: Processor Scheduling implies that jobs or tasks are to be assigned to a particular processor for execution at a particular time Problem of finding an optimal schedule for a set of jobs is NP-complete. Algorithm that implement scheduling requires exponential and polynomial time to reach an optimized solution. Various optimization techniques can be used to find optimized solutions for process scheduling. Genetic Algorithm is one of the optimized solutions for process scheduling. This paper contains the review of processor scheduling algorithms and genetic algorithm to provide efficient scheduling algorithm.

Keywords: CPU Scheduling, Optimization, Genetic Algorithm, Crossover, Selection

I. INTRODUCTION

Scheduling is a fundamental operating system function. Almost all computer resources are scheduled before use [3].Process scheduling in an operating system can be defined as allocating processes to the processor so that efficiency of the system will be improved. Various scheduling problems are considered as NP Hard problems [2]. There are different scheduling algorithms among them First-Come First-Served (FCFS) Scheduling, Shortest–Job-First (SJF) Scheduling, Round Robin (RR) Scheduling and Priority Scheduling are mostly used for scheduling of tasks of processor [4].The algorithm in evolutionary computation research called Genetic Algorithm to find an optimal solution for scheduling.

II. GENETIC ALGORITHMS(GA)

Genetic Algorithms are the meta heuristics search and optimization techniques that mimic the process of natural evaluation. Genetic Algorithms are invented by the Holland in 1960[6]. The algorithm work by evolving a population of individuals over a number of generations.

The genetic algorithm is very simple. It starts with a population of random individuals, each corresponding to a particular candidate solution to the problem to be solved. Then, the best individuals survive, mate, and create offspring, originating a new population of individuals. This process is repeated a number of times, and usually leads to better and better individuals.

III. KEY ELEMENTS of GA

A. Individuals

An individual is a single solution. Individual groups together two forms of solutions as given below [6]:

1) *The chromosome*: which is the raw genetic information (genotype) that the GA deals with.
2) *The phenotype:* which is the expressive of the chromosome in the terms of the model.

A chromosome is subdivided into genes. Genes code the characteristics of an individual. Possibilities of the genes for one characteristic are called alleles and a gene can take different value of alleles [1]. Each factor in the solution set corresponds to gene in the chromosome.

B. Fitness

The fitness of an individual in a genetic algorithm is the value of an objective function for its phenotype. For calculating fitness, the chromosome has to be first decoded and the objective function has to be evaluated. The fitness not only indicates how good the solution is, but also corresponds to how close the chromosome is to the optimal one.

C. Populations

A population is a collection of individuals. A population consists of a number of individuals being tested, the phenotype parameters defining the individuals and some

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

information about search space. The two important aspects of population used in Genetic Algorithms are:

The initial population generation.

The population size.

D. Encoding

Encoding is a process of representing individual genes. The process can be performed using bits, numbers, trees, arrays, lists or any other objects. The encoding depends mainly on solving the problem. For example, one can encode directly real or integer numbers.

1) *Binary Encoding:* Each chromosome encodes a binary (bit) string. Each bit in the string can represent some characteristics of the solution. Every bit string therefore is a solution but not necessarily the best solution. Another possibility is that the whole string can represent a number

2) *Octal Encoding:* This encoding uses string made up of octal numbers (0–7).

3) *Hexadecimal Encoding:* This encoding uses string made up of hexadecimal numbers (0–9, A–F).

4) *Tree Encoding:* This encoding is mainly used for evolving program expressions for genetic programming. Every chromosome is a tree of some objects such as functions, routines and commands of a programming language. And the respective genes are placed as the branches of the tree.

5) *Value Encoding:* In value encoding, every chromosome is a string of some values. Values can be anything connected to problem, form numbers, real numbers or chars to some complicated objects [6].

The basic genetic algorithm is as follows:

1) [start]: Genetic random population of n chromosomes (suitable solutions for the problem).

2) [Fitness]: Evaluate the fitness f(x) of each chromosome x in the population.

3) [New population]: Create a new population by repeating following steps until the new population is complete,

4) [selection]: select two parent chromosomes from a population according to their fitness (the better fitness, bigger are the chances to get selected).

5) [crossover]: With a crossover probability, cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.

6) [Mutation]: With a mutation probability, mutate new offspring at each locus (position in chromosome).

7) [Accepting]: Place new offspring in the new population.

8) [Replace]: Use newly generated population for a further sum of the algorithm.

9) [Test]: If the end conditions are satisfied, stop and return the best solution in current population.

10) [Loop]: Go to step2 for fitness evaluation.

IV. SCHEDULING CRITERIA

Different CPU scheduling algorithms have different properties, and the choice of a particular algorithm may favor one class of processes over another. In choosing which algorithm to use in a particular situation, we must consider the properties of the various algorithms [7].

Many criteria have been suggested for comparing CPU scheduling algorithms. Which characteristics are used for comparison can make a substantial difference in which algorithm is judged to be best. The criteria include the following:

1) *CPU Utilization:* We want to keep the CPU as busy as possible.

2) *Throughput:* If the CPU is busy executing processes, then work is being done. One measure of work is the number of processes that are completed per time unit, called throughput. For long processes, this rate may be one process per hour; for short transactions, it may be 10 processes per second.

3) *Turnaround time:* From the point of view of a particular process, the important criterion is how long it takes to execute that process. The interval from the time of submission of a process to the time of completion is the turnaround time. Turnaround time is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O.

4) *Waiting time:* The CPU scheduling algorithm does not affect the amount of the time during which a process executes or does I/O; it affects only the amount of time that a process spends waiting in the ready queue. Waiting time is the sum of periods spend waiting in the ready queue.

5) *Response time:* is the time it takes to start responding, not the time it takes to output the response. The turnaround time is generally limited by the speed of the output device. [5].

6) So, we can conclude that an efficient scheduling algorithm is that which can optimize the following performance measures:

Maximize CPU utilization
Maximize throughput
Minimize turnaround time
Minimize waiting time
Minimize response time

The scheduling criteria are optimization problems. Different optimization techniques are designed to optimize average times of throughput.

V. COMPARISON of GA with OTHER OPTIMIZATION TECHNIQUES

Genetic algorithm is one of the best techniques to optimize scheduling algorithms. Genetic algorithm differs from conventional optimization techniques in following ways [6]:

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

- GA's operate with coded versions of the problem parameters rather than parameters themselves i.e., GA works with the coding of solution set and not with the solution itself.
- Almost all conventional optimization techniques search from a single point but GA's always operate on a whole population of points (strings) i.e., GA uses population of solutions rather than a single solution for searching. This plays a major role to the robustness of genetic algorithms. It improves the chance of reaching the global optimum and also helps in avoiding local stationary point.
- GA uses fitness function for evaluation rather than derivatives. As a result, they can be applied to any kind of continuous or discrete optimization problems. The key point to give stress here is to identify and specify a meaningful decoding function.
- GA's use probabilistic transition rules while conventional methods for continuous optimization apply deterministic transition rules i.e., GA's does not use deterministic rules.

VI. CONCLUSION

Processor Scheduling Problems are a classical combinatorial problem that is problems on the selections and arrangements of elements of a finite set. Different techniques or algorithms are developed to find an optimal solution, Genetic algorithm is one of the optimization techniques that can be used to solve the problems of function maximization. Genetic algorithms can provide easy to understand and user friendly optimized solutions to the sequencing problems. In this paper, a survey on genetic algorithm to optimized processor scheduling algorithms and comparison with other optimization techniques is presented. The future work is intended to an efficient proposed genetic algorithm that will be efficient for process scheduling problems having less average waiting time.

REFERENCES

[1] Davis, L. "Handbook of Genetic Algorithm". Von Nostrand Reinhold, New York, 1991.
[2] Dr. Rakesh Kumar. "Genetic Algorithm approach to Operating system process scheduling problem". International Journal of engineering Science and Technology, pp. 4247-4252, 2010.
[3] Silberschatz,A. and P.B. Galvin,2003. Operating System concepts, Sixth Edition, John Wiley & Sons, Inc.
[4] Vikas Gaba,Anshu Parashar. "Comparison of processor scheduling algorithms using genetic approach". International Journal of Advanced Research in Computer Science and Software Engineering 2 (8), August-2012, pp. 37-45
[5]http://en.wikipedia.org/wiki/Scheduling_ (computing)
[6] Sivanandam, S. N. & Deepa, S. N. Introduction to
Genetic Algorithms. Springer.2008.
[7] Rakesh Kumar Yadav. "An Improved Round Robin Scheduling Algorithm for CPU scheduling".International Journal on Computer Science and Engineering,Vol. 02, No. 04, 2010, 1064-1066