DEVELOPING THE BEST SCHEDULING ALGORITHM FROM EXISTING ALGORITHMS FOR REAL TIME OPERATING SYSTEMS

Abdul Salam
University of Malakand,Chakdara,KPK,Pakistan
asalam.swat@gmail.com

Sohail Abbas
University of Malakand,Chakdara,KPK,Pakistan
sabbas@uom.edu.pk

Yousaf Khan
University of Malakand,Chakdara,KPK,Pakistan
yousafkhan792@gmail.com

Sanaul Haq
University of Malakand,Chakdara,KPK,Pakistan
sanaulhaq33@yahoo.com"

Saeed Ullah Jan
University of Malakand,Chakdara,KPK,Pakistan
saeedullah@uom.edu.pk

ABSTRACT

This paper is about the selection of scheduling algorithm for real time system. In this paper we compared different scheduling algorithms and from these comparison we get another algorithm which is good in performance as compared to the existing one. First we compare ACO and EDF but both have some merits and demerits. ACO is not good when system is under loaded, preemptive and single processor while the result of ACO is good in overloaded condition with a lot of execution time. From the comparison of ACO and EDF another algorithm is developed called adaptive algorithm which is best in both overloaded and under loaded condition. [8]

Then we compared EDF and GA based scheduling algorithm. EDF is used in under loaded condition and when the system become overloaded it changes to GA based algorithm. From the comparison of these two algorithms we get another algorithm which is also called adaptive algorithm. Performance of both algorithms is measured by using success rate, effective CPU utilization and execution time.

KEYWORDS

REAL TIME SCHEDULING, DEADLINE, SCHEDULER, EARLIEST DEADLINE FIRST (EDF), ANT COLONY OPTIMIZATION (ACO), GENETIC ALGORITHMS.

1. INTRODUCTION

1.1 REAL-TIME OPERATING SYSTEMS

Real time systems are those systems which depend on two factors. The one is the logical result of computation and another is the time at which it produces the results. A fixed time is given to a process to complete its execution and if the time expires for the process the system will fail. There are two techniques for real time scheduling: (1) static (2) dynamic. In static algorithm the priorities remain the

same for a task and priorities are assigned at the time of designing, while in dynamic assigns priority at run time [1, 6].

1.2 ANT COLONY OPTIMIZATION (ACO) ALGORITHM

ACO was bring together to solve hard combinatorial optimization problem. Ant Colony is an approximation algorithm which is used to provide solution to hard problems in the required time. ACO follows the principle of real ants. The ant first visit different areas randomly and as the food source is found it evaluates quality and quantity of food and guides other ant to the food source. [12]

1.3 EARLIEST DEADLINE FIRST (EDF) ALGORITHM

In EDF algorithm each process has a deadline. Priorities are given to processes according to their deadlines. Higher priority is given to those processes whose deadline is nearest and that process is selected for execution. This algorithm is good for a system with single processor and under loaded systems. The performance of EDF is not good when system is overloaded [4, 9].

1.4 GENETIC BASED SCHEDULING ALGORITHM

Genetic algorithm can either be produced randomly or based on various other algorithms. All individual is an encoding of a set of parameters that uniquely identifies a possible outcome of the problem. In genetic algorithm first we have to encode possible solution of a problem which are strings and we called these stings a set of chromosomes.

1.5 ADAPTIVE SCHEDULING ALGORITHM

To overcome the limitation of the above three algorithms we develop another technique which is called Adaptive scheduling Algorithm. By combining ACO and EDF we get adaptive algorithm. As, EDF is not good in overloaded situation and ACO is better than EDF when the system is overloaded [4]. The disadvantage of ACO is large execution time. Adaptive scheduling Algorithm automatically switch between Algorithms. [8]

By combining GA based algorithm and EDF we can also get adaptive algorithm. When the system is under loaded, the EDF algorithm is used by adaptive algorithm. During overloaded condition,GA algorithm is used [4].

2. COMPARISON OF DYNAMIC ALGORITHMS

2.1 EARLIEST DEADLINE FIRST (EDF) ALGORITHM

EDF is the dynamic scheduling algorithm which depends upon the deadline of the task. The task which has nearest deadline has highest priority [10].

EDF depends on some parameters which are Start time, Execution Time, Deadline of the process, Release time and the Load of each process. As the process is created, it is stored in a queue and its priority depends upon the deadline [9].

The processes are added one by one in a EDF queue. Let ? = (? 1,p2,? 3…….? ?) denotes a set of process. Suppose for process ? 1, p1= (?*i*,e*i*,? *i*) is characterized by its release time ?*i*, execution time ?? and deadline of the process ? 1. The process is added in the system and the process with highest priority is switched to EDF algorithm and the switching depends upon deadline of the process.

For process p1 the periodic task τi= (c*i*,p*i*) is by two parameters: one is execution time c1 and another is period p1. The utilization of periodic task τi is defined as follow [11].

? i=c*i*/p*i* . . . . . *(1)*

A task can be feasibly scheduled using EDF algorithm if the total utilization of a task set ? is

$$U(\tau) = \sum_{i=1}^{n} u_i \qquad (2)$$

In the under loaded process, the EDF algorithm executes but CPU usage for that process is minimum . In case of overloaded some processes fail during execution that is the disadvantage of EDF algorithm [4]. To minimize this failure adaptive algorithm is utilized [5, 7].

2.2 ANT COLONY OPTIMIZATION (ACO) ALGORITHM

This algorithm follows the behavior of real ants each one makes a path and there are many ants which are active at the same time concurrently [3].

In this algorithm, there are many ants and each ant represent a node. Each ant initiates their journey from various nodes after applying scheduling algorithm in ACO each node is a task whose probability depend upon the pheromone value ? and heuristic value ? [2, 12].

It is given by;

$$p_i(t) = \frac{(\tau_i(t))^{\alpha} * (\eta\eta_i(t))^{\beta}}{\sum_{l \in R_1} (\tau_l(t))^{\alpha} * (\eta\eta_l(t))^{\beta}}$$

Where,

(?) is the probability at time t of *i*th node.

?? is the value pheromone on ??*h* node at time ?.

?? is a heuristic value of ??*h* node at time ? and is calculated by,

□?=?? ?–?

2.3 GENETIC ALGORITHMS

A genetic algorithm copies the natural evolution process and it generally starts with the initial populations of individual, which could be generated randomly or it depends on some other algorithm. Each member of a set uniquely identifies a solution for problem. The populations go through different stages in order to create new individuals which replace individuals from which it creates. In crossover, part of the two individuals of the populations are exchanged to create two entirely new individuals which replace the individual from which they evolve. Each individual is chosen for crossover with a probability of crossovers rate. Mutations alter one or extra gene in chromosomes with a chance of mutations rate. For example, if the individual is an encoding of a schedule, two tasks are picked randomly and their positions are interchanged. A fitness function calculates the fitness of each individual, i.e., it decides how good a particular solution is. In the selection process, each individual of the current population is selected into the new population with a probability proportional to its fitness. The selection process ensures that individuals with higher fitness values have a higher probability to be carried onto the next generation, and the individuals with lower fitness values are dropped out. The new population created in the above

manner constitutes the next generation, and the whole process is terminated either after a fixed number of generations or when a stopping criteria is met. The population after a large number of generations is very likely to have individuals with very high fitness values which imply that the solution represented by the individual is good; it is very likely to achieve an acceptable solution to the problem. There are many variations of the general procedure described above. The initial population may be generated randomly, or through some other algorithm. The search space, i.e., the domain of the individuals, can be limited to the set of valid individuals, or extended to the set of all possible individuals, including invalid individuals. The population size, the number of generations, the probabilities of mutation and crossover are some of the other parameters that can be varied to obtain a different genetic algorithm

The height varying point of the tasks are taken for cross over as shown in fig 1.After cross over the task arrangement is shown in fig 2. And their Gantt chart is shown in Fig 3.

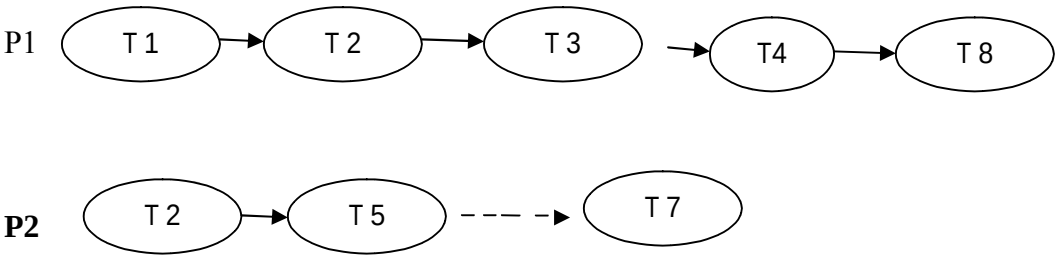

Figure 1: Before Cross Over Point

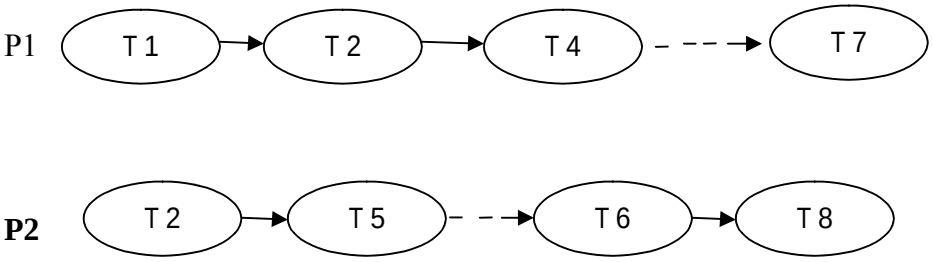

Figure 2: After Cross Over

T1	**T3**	**T4**	**T6**	**T8**
T2	**T5**		**T7**	

Fig 3 Gantt chart of tasks after Cross over.

2.4 ADAPTIVE ALGORITHM

Adaptive algorithm is used to improve the load balancing. An Adaptive algorithm follows the following principles:

When the system is under loaded it uses EDF algorithm and priority of the job is decided according to the deadline of the task [6]. When the system is overloaded it switches to ACO algorithm, calculates the total execution time and minimum time required for each process. Designing steps of adaptive algorithm are as follow:

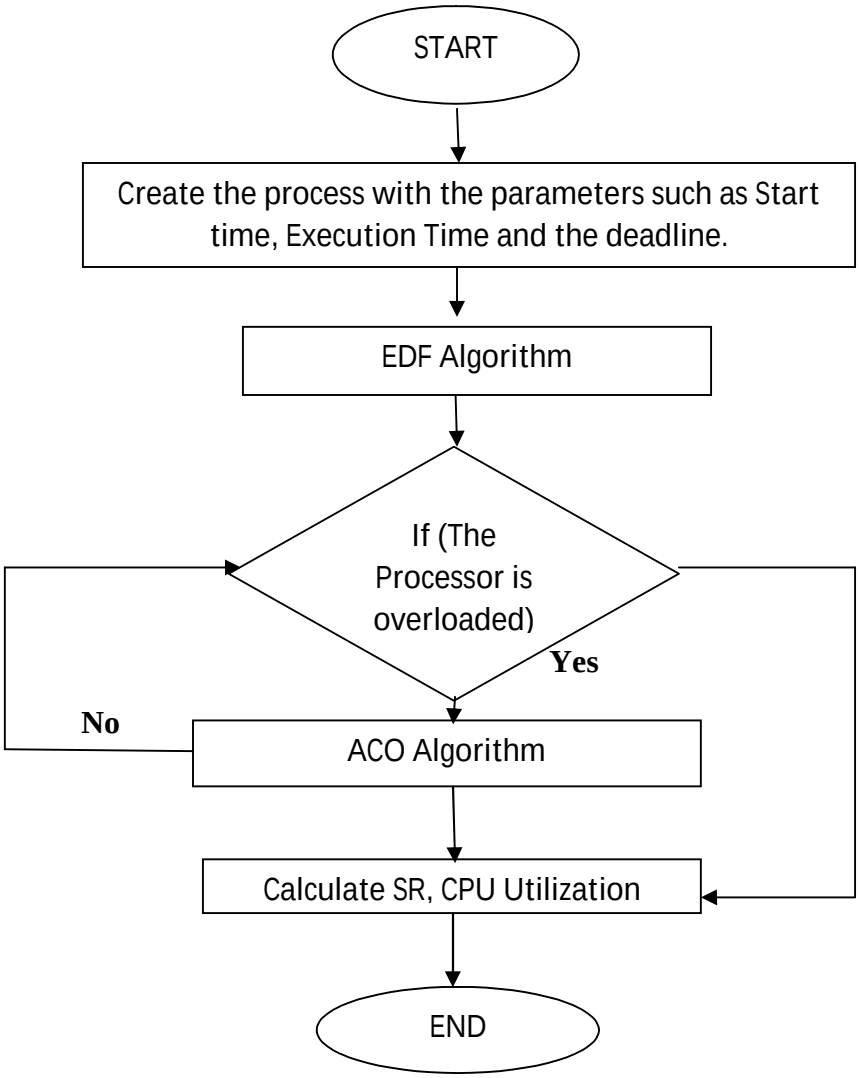

Figure 4: Flowchart of an Adaptive Scheduling Algorithm

1. Create the process and initialize all the parameters such as Execution time, start Time, Deadline, load of process and release time.
2. First switch process to EDF.
3. The process with nearest deadline [6] is given the highest priority.
4. For switching to ACO algorithm, if the load of the process is greater than CPU load, then the process is transferred from EDF to ACO algorithm.

5. In ACO, it will calculate the total time required for all the processes and the process which contains minimum execution time is executed first. In short execution depends upon the execution time.
6. Success ratio is used for performance measurement, CPU Utilization and it also calculates the total number of processes which are executed using EDF and the ACO algorithm with the deadline missed process [6].
7. Finally the result is calculated.

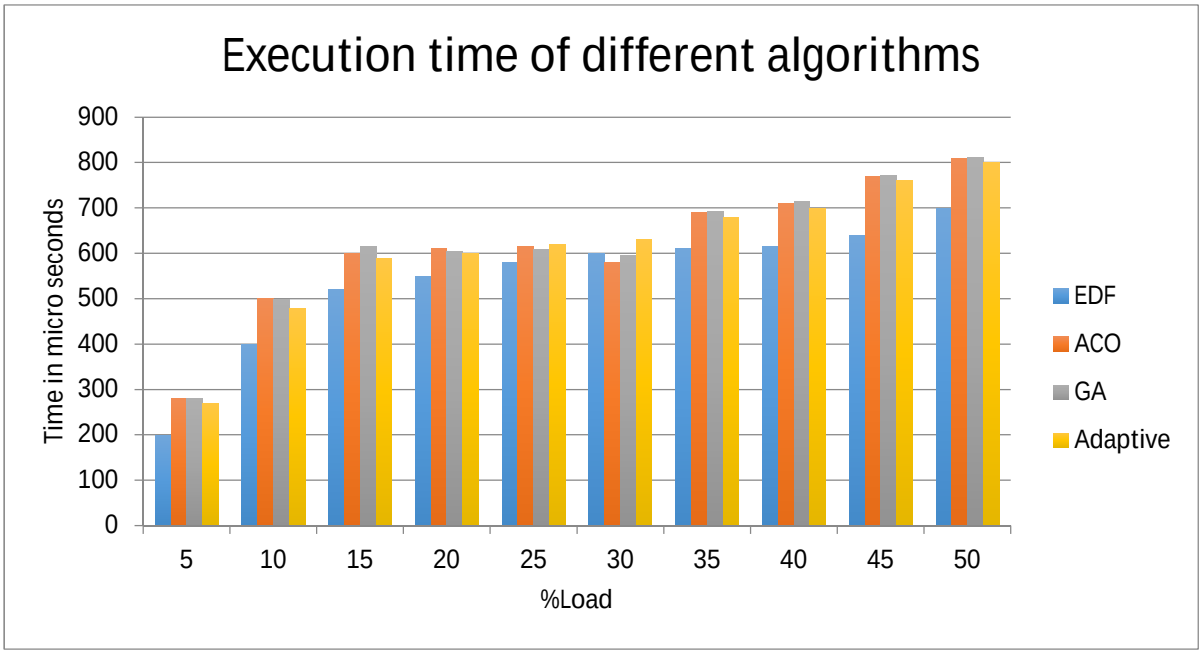

Figure 5: Comparison of Execution time of EDF, ACO, GA and Adaptive Algorithm

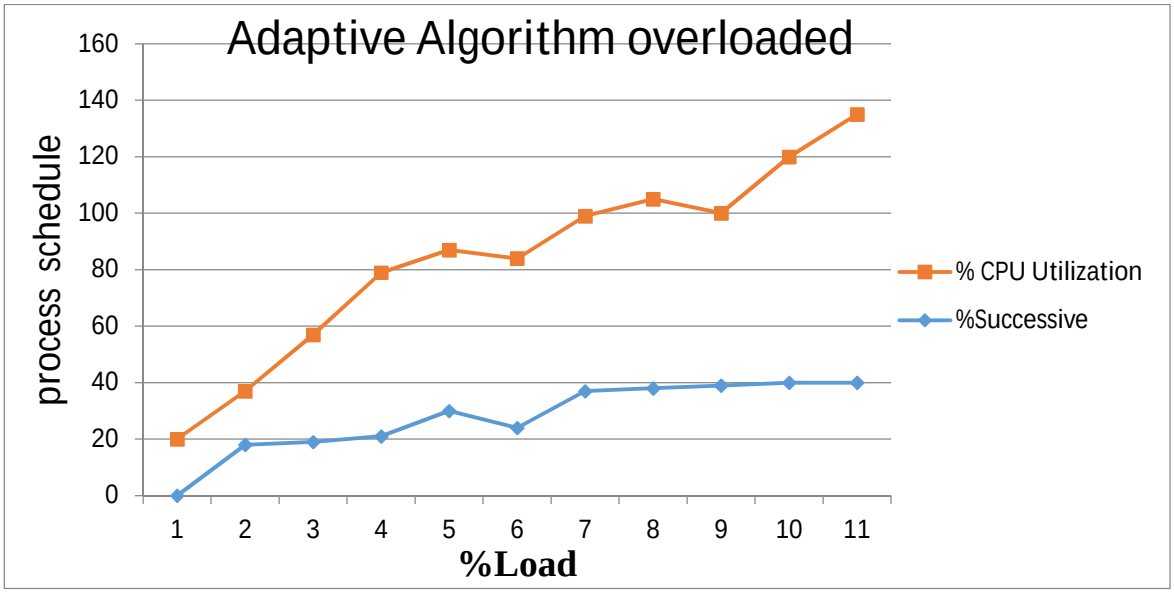

Figure 6: Successive rate and CPU utilization for Adaptive Algorithm when over loaded

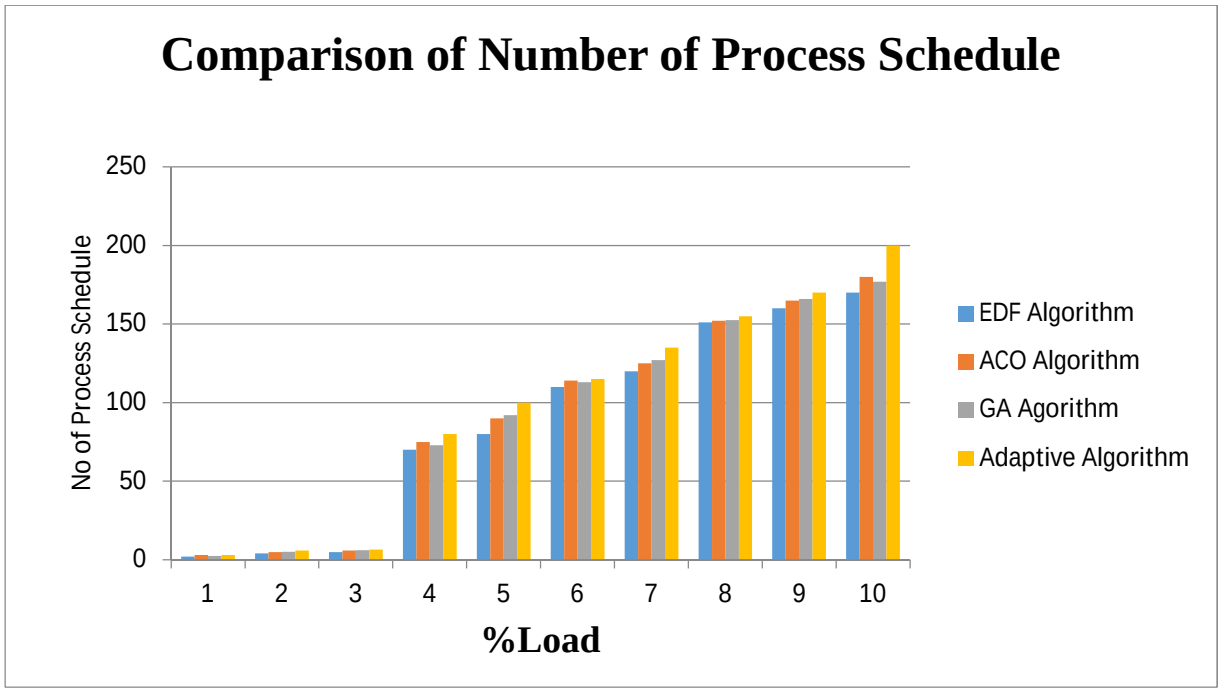

Figure 7: Comparison of all the processes scheduled for all Algorithms

3. CONCLUSIONS

In this paper, we compared EDF, ACO and GA based algorithm. All of the algorithms have some advantages and disadvantages. EDF performs well when the system is under loaded but ACO does not perform well in that situation. On the other hand EDF does not perform well when the system is overloaded while ACO and GA based algorithm take more execution time in that type of situation.

From the comparison of EDF, ACO and GA we develop an algorithm which is called adaptive algorithm. This algorithm works well when future workload of the system is not known. An Adaptive Algorithm schedules the process on single processor when it is preemptive. This algorithm automatically switches between the EDF and ACO algorithm and overcome the limitation of both the algorithms (EDF and ACO) algorithms. Adaptive algorithm takes less execution time than ACO, EDF and GA. when load of process is increased system usage also increased.

4. REFERENCES

[1] Jane W.S. Liu," *Real-Time Systems"* , Pearson Education, India,pp. 121 & 26, 2001.

[2] K. Kotecha and A. Shah,"*ACO based dynamic scheduling algorithm for real-time operating system*", *Sent to AIPR-08, Florida*, 2008.

[3] Bank, M., Honig, U., and Schiffmann, W.: "*An ACO-based approach for scheduling task graphs with communication costs*□. Proc. Int. Conf. on Parallel Processing (ICPP□05), 2005, pp. 623–629

[4] Hyeonjoong Cho, Binoy Ravindran & E. Douglas Jensen " *Optimal Real-Time Scheduling Algorithm for Multiprocessors* " Proceedings of the 27th IEEE International Real-Time Systems Symposium (RTSS'06)- 2006.

[5] M.Kaladevi and Dr.S.Sathiyabama, "*A Comparative Study of Scheduling Algorithms for Real Time Task International Journal of Advances in Science and Technology"*, Vol. 1, No. 4, 2010.

[6]A. Silberschatz, P.B. Galvin and G. Gagne, (2001)," *Operating Systems Concepts*", Sixth edition, John
Wiley Publishers.

[7] Lalatendu Behera Durga Prasad Mohapatra,"*Schedulability Analysis of Task Scheduling in Multiprocessor Real-Time Systems Using EDF Algorithm",* 2012 International Conference on Computer Communication and Informatics Coimbatore, INDIA

[8] Jashweeni Nandanwar, Urmila Shrawankar *,"An Adaptive Real Time Task Scheduler IJCSI International Journal of Computer Science Issues"*, Vol. 9, Issue 6, No 1, November 2012.

[9] C. Liu and J. Layland, "*Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*", J. ACM, vol. 20, pp. 46–61, 1973.

[10] Giorgio C. Buttazzo, Marko Bertogna and Gang Yao, "*Limited Preemptive Scheduling for Real-Time Systems IEEE Transactions On Industrial Informatics*", Vol. 9, no. 1, February 2013.

[11] Xuefeng Piao, Sangchul Han, Heeheon Kim, Minkyu Park, Yookun Cho "*Predictability of Earliest Deadline Zero Laxity Algorithm for Multiprocessor Real-Time Systems*" ,Proceedings of the Ninth IEEE International

[12] Symposium on Object and Component-Oriented Real-Time Distributed Computing 2006