Proposed Fuzzy CPU Scheduling Algorithm (PFCS) for Real Time Operating Systems

Prerna Ajmani[1] and **Manoj Sethi**[2]

Submitted in March, 2013; Accepted in August, 2013

Abstract - *In the era of supercomputers multiprogramming operating system has emerged. Multiprogramming operating system allows more than one ready to execute processes to be loaded into memory. CPU scheduling is the process of selecting from among the processes in memory that are ready to execute and allocate the processor time (CPU) to it. Many conventional algorithms have been proposed for scheduling CPU such as FCFS, shortest job first (SJF), priority scheduling etc. But no algorithm is absolutely ideal in terms of increased throughput, decreased waiting time, decreased turnaround time etc. In this paper, a new fuzzy logic based CPU scheduling algorithm has been proposed to overcome the drawbacks of conventional algorithms for efficient utilization of CPU.*

Index Terms - CPU scheduling, fuzzy logic, Multiprogramming Operating System, process, turnaround time, and throughput.

1. INTRODUCTION

With the advancement in operating system, multiprogramming operating systems has evolved. In a multiprogramming environment, many processes are loaded into memory that competes for CPU time. CPU scheduling algorithms determines which process will be given processor time and which will wait. Some of the objectives that scheduling function should satisfy in order to be effective include fairness, efficient use of processor time, response time, turnaround and throughput [11].There are many scheduling algorithms such as FCFS, SJF, PRIORITY Scheduling etc., but none is efficient for real time tasks.

1. **FCFS**: - In FCFS algorithm the process is allotted processor time on First Come, First Serve basis. It is a non-preemptive scheduling in which the processes are being given CPU in the order of their arrival in ready queue. Advantage of FCFS is less context switching overhead. But the limitations are: - (i) Throughput can be low, since long processes can hold the CPU. (ii)Turnaround time, waiting time and response time can be high for the same reason. (iii)No prioritization occurs, thus this system has trouble to meet deadlines of the processes. (iv)Convoy Effect: - All the processes wait for one long process to get off CPU [11].
2. **SJF: -** To overcome the limitations of FCFS, Shortest Job First (SJF) algorithm was proposed. This algorithm selects the process with smallest burst time to execute next. The limitation of algorithm is: - it is very difficult to know the burst time of next CPU request. Although this algorithm is optimal but it cannot be implemented at the level of short-term CPU scheduling[11].
3. **SRTF:-**Shortest-Remaining-Time-First(SRTF) scheduling algorithm is preemptive version of SJF. This algorithm allows the next process with shorter burst to preempt the process already executing, if the burst of new arrived process is shorter than the remaining time for the running process.
4. **Priority Scheduling Algorithm (Pri):-**In this algorithm the process with highest priority is assigned CPU first and so on. The priorities are assigned to process by operating system. Low priority process gets interrupted by the incoming of higher priority process. The limitation of algorithm is indefinite blocking or starvation of lower priority process if there is large number of high priority process. Also, waiting time and response time depends on priority of process. To overcome the limitation of indefinite blocking aging technique was proposed which gradually increases the priority of processes waiting from long time.

None of the algorithms stated above is ideal with respect to scheduling objectives. Therefore, in this paper we proposed a t new algorithm which uses fuzzy logic to find the dynamic priority of the process.

2. RELATED WORK

Terry Regner & Craig Lacey[8] introduced the concepts and fundamentals of the structure and functionality of operating systems. The purpose of this article was to analyze different scheduling algorithms in a simulated system. This article has the implementation of three different scheduling algorithms: shortest process first, round robin, and priority sequence. Comparing the three algorithms, they find that the CPU utilization values indicate that the shortest process first has the highest throughput values with CPU utilization times comparable to those of the round robin. Ajit Singh[9] developed a new approach for round robin scheduling which helps to improve the CPU efficiency in real time and time sharing operating system. Alexander[10] stated that Multimedia applications have unique requirements that must be met by network and operating system components. In any multimedia application, we may have several processes running dependently on one another. Multimedia is a real-time

[1]*SGTBIM&IT, GGSIPU, Delhi*
[2]*Delhi Technological University, Delhi*
E-mail: [1]pahwapre@gmail.com and [2]ms2690@rediffmail.com

application. In context of multimedia applications, the CPU scheduler determines quality of service rendered. The more CPU cycles scheduled to a process, the more data can be produced faster, which results in a better quality, more reliable output. Many Researchers have tried to implement fuzzy logic to schedule the processes. A fuzzy-based CPU scheduling algorithm is proposed by Shata J. Kadhim et. al[1]. Round robin scheduling using neuro fuzzy approach is proposed by Mr. Jeegar A Trivedi et. al[2]. Soft real-time fuzzy task scheduling for multiprocessor systems is proposed by Mahdi Hamzeh et. al[3]. Efficient soft real-time processing is proposed by C. Lin et. al[4]. An Improved fuzzy-based CPU Scheduling(IFCS)algorithm for real time systems is proposed by H.S. Behera[5].

3. FUZZY LOGIC TERMINOLOGY USED

3.1 Fuzzy Logic:- A Fuzzy logic is a generalization of standard logic, in which a concept can possess a degree of truth anywhere between 0 and 1. It allows intermediate values to be defined between conventional evaluations. A Fuzzy logic system is nonlinear mapping of an input data to output data. A Fuzzy logic system consists of components: fuzzier, rules, inference engine and de-fuzzier. The process of fuzzy logic is to first collect the crisp set of inputs and convert it to the fuzzy sets using fuzzy linguistic variables, fuzzy linguistic terms and membership functions. This is known as Fuzzification. Afterwards an inference is made on the basis of set of rules. Finally, the resulting fuzzy output is mapped to a crisp output using the membership functions, in the defuzzification step.

3.2 Fuzzy Logic Terminology :-

(i) **Linguistic Variables**: - It is the input or output variables of the system whose values are non-numeric.

The values may be words or sentences derived from natural language.

(ii) **Membership Functions**: - Membership functions are used in the Fuzzification and defuzzification steps of a FLS, to map the non-fuzzy input values to fuzzy linguistic terms and vice-a-versa. A membership function is used to quantify a linguistic term. It is denoted by µ.In our proposed algorithm we consider two memberships, one of burst time (µb) and other of priority (µp).

3.3 Fuzzy Inference System (FIS):- Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basic from which decisions can be made[5]. An FIS consists of an input stage, a processing stage, and an output stage. The input stage maps the inputs, such as deadline, execution time, and so on, to the appropriate membership functions and truth values. There are two common FIS:-(i)Mamdani's fuzzy inference method proposed in 1975 by Ebrahim Mamdani[6]. (ii)Takagi-Sugeno-Kang, or simply Sugeno, method of fuzzy inference introduced in 1985[7].

These two methods are similar as the procedure of fuzzifying the inputs and fuzzy operators. The difference between the two is that the Sugeno’s output membership functions are either linear or constant but Mamdani’s inference expects the output membership functions to be fuzzy sets.

In our proposed algorithm we use Mamdani’s inference system.

3.4 Fuzzy Based CPU Scheduling Algorithms:-

Improved Fuzzy CPU Scheduling Algorithm (IFCS):- Scheduling is very critical for real time processes. The processes priorities keep on changing in real time. This algorithm is based on dynamic priorities (dpi) rather on static priorities (pti). It considers membership function of priority, burst time and response ratio to find dynamic priority and schedule the process according to it. The algorithm was proposed by H.S. Behera et al.[6]. This algorithm ensures reduced waiting time and turnaround time.

Proposed Fuzzy CPU Scheduling Algorithm (PFCS):- This algorithm also calculates dynamic priorities and schedules the process according to it but it doesn’t include membership function based on response ratio while calculating dynamic priority. For calculation of dynamic priorities it relies on membership function of priorities and burst time. The algorithm is further evaluated to see the performance in terms of turnaround time and waiting time.

4. SIMULATOR USED

4.1 PSSAV (Process Scheduling Simulation, Analyzer, and Visualization):-It is an application for CPU scheduling algorithm which provides customizable comparison between each scheduling algorithm. We have used to analyze our algorithm in this simulator.

4.2 Emulated turbo C++:-It is an integrated development environment (IDE) which has C compiler. We have developed the code corresponding to our algorithm in this IDE.

5. PROPOSED ALGORITHM (PFCS):-

5.1 Calculate dynamic priority(dpi):-

1) For each process Pi in ready queue fetch its parameters burst time (bti), static priority (pti), and arrival time (ati) and give them as input to FIS.

2) For each process Pi; evaluate membership function of priority i.e. µp

µp=pti/max (pti) +1; where 1<=i<=n

3) For each process Pi; evaluate membership function of burst time i.e. µb

µb=1-(bti/max (bti) +1); where 1<=i<=n

4) For each process Pi in ready queue find minimum priority process.

5) To calculate dynamic priority (dpi)

If process Pi has minimum priority then

dpi= (µp+µb)

Else

dpi= max {µp, µb}

where 1<=i<=n

(A) Pseudo code:-
1) Set dynamic priority (dpi) to the output of FIS.
2) Schedule the process Pi with the highest value of dpi for execution where 1<=i<=n.
3) If the scheduled process finishes and no new process arrives go to step 2.
   Else if new process arrives go to step 1.
4) Exit.

6. RESULTS & PERFORMANCE EVALUATION

Different Case Studies are being taken to evaluate the performance of proposed fuzzy CPU Scheduling Algorithm (PFCS).

CASE STUDY 1:-

Process ID	Arrival Time(ati)	Burst Time(bti)	Static Priority (pti)
P1	0	3	6
P2	2	6	1
P3	4	4	5
P4	6	5	2
P5	8	2	7

Table 1: Case Study-1 Data Set [5]

Comparison Table:-

Algorithm	Waiting Time average (ms)	Turnaround Time average (ms)	Completion Time (ms)
Priority Algorithm(Pri)	4.8	8.8	20
IFCS	3.8	7.8	20
PFCS	3.8	7.8	20

Table 2: Comparison between various algorithms for Case Study-1

CASE STUDY 2:-

Process ID	Arrival Time(ati)	Burst Time(bti)	Static Priority(pti)
P1	0	18	11
P2	0	2	6
P3	0	1	7
P4	0	4	3
P5	0	3	5
P6	0	12	2
P7	0	13	1

Table 3: Case Study-2 Data Set [5]

Comparison Table:-

Algorithm	Waiting Time average(ms)	Turnaround Time average(ms)	Completion Time(ms)
Priority Algorithm(Pri)	23.86	31.43	53
IFCS	14.86	22.43	53
PFCS	14.86	22.43	53

Table 4: Comparison between various algorithms for Case Study-2

Comparative Analysis:-

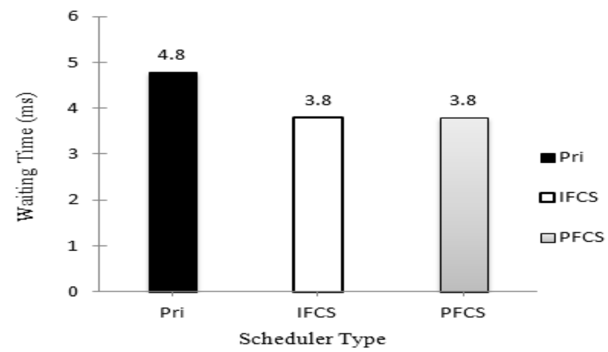

Figure 1: Waiting time comparison between various algorithms for Case Study-1.

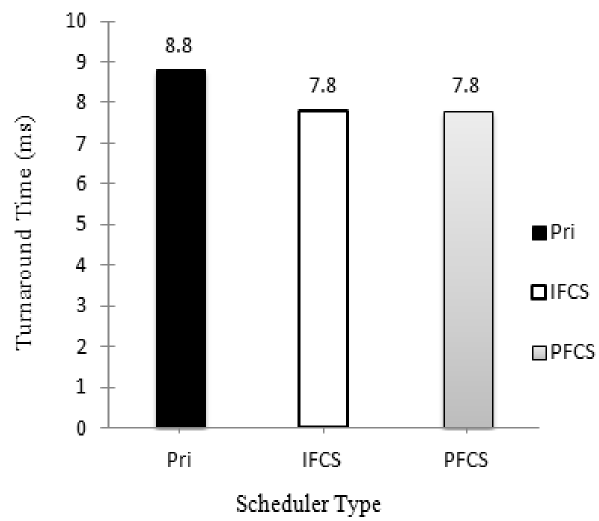

Figure 2: Turnaround time comparison between various algorithms for Case Study-1.

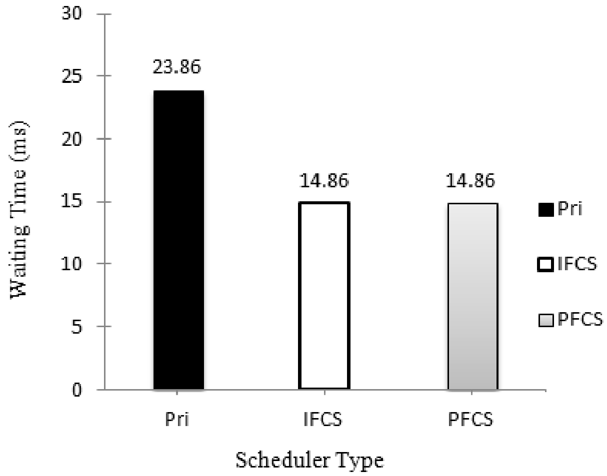

Figure 3: Waiting time comparison between various algorithms for Case Study-2.

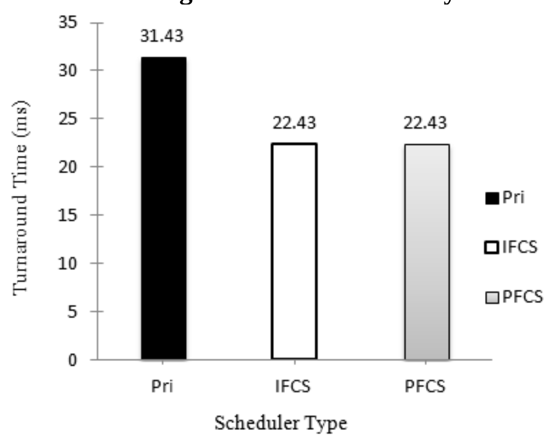

Figure 4: Turnaround time comparison between various algorithms for Case Study-2.

Comparison Table:-

Algorithm	Waiting Time average(ms)	Turnaround Time average(ms)	Completion Time(ms)
Priority Algorithm(Pri)	22.44	30.33	71
IFCS	20.78	28.67	71
PFCS	20.78	28.67	71

Table 5: Comparison between various algorithms for Case Study-3

Comparative Analysis:-

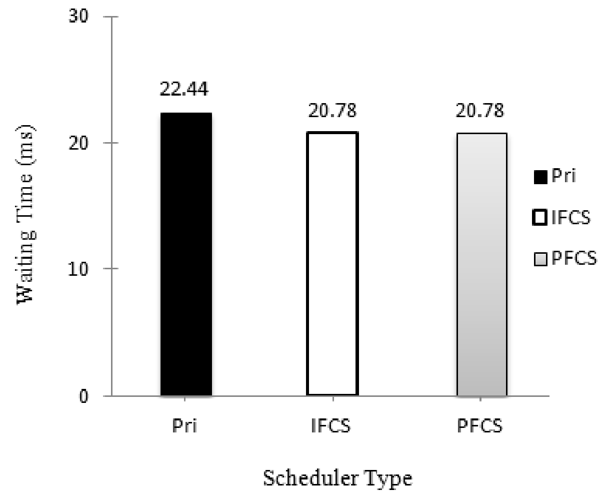

Figure 5: Waiting time comparison between various algorithms for Case Study-3.

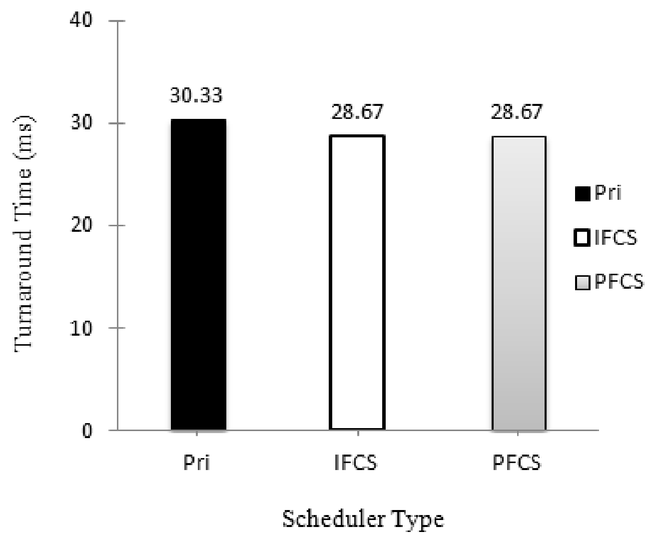

Figure 6: Turnaround time comparison between various algorithms for Case Study-3.

CASE STUDY 4:-

Process ID	Arrival Time(ati)	Burst Time(bti)	Static Priority (pti)
P1	0	10	3
P2	0	1	1
P3	0	2	4
P4	0	1	5
P5	0	5	2

Table 6: Case Study-4 Data Set [11]

Comparison Table:-

Algorithm	Waiting Time average(ms)	Turnaround Time average(ms)	Completion Time(ms)
Priority Algorithm(Pri)	8.2	12.0	19
IFCS	3.2	7.0	19
PFCS	3.2	7.0	19

Table 7: Comparison between various algorithms for Case Study-4

Comparative Analysis:-

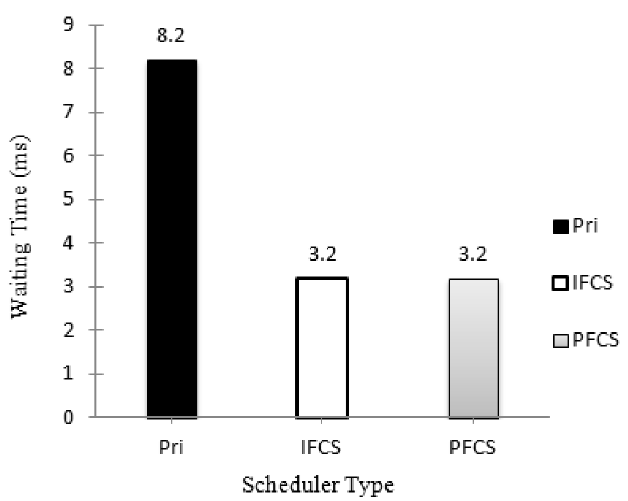

Figure 7: Waiting time comparison between various algorithms for Case Study-4.

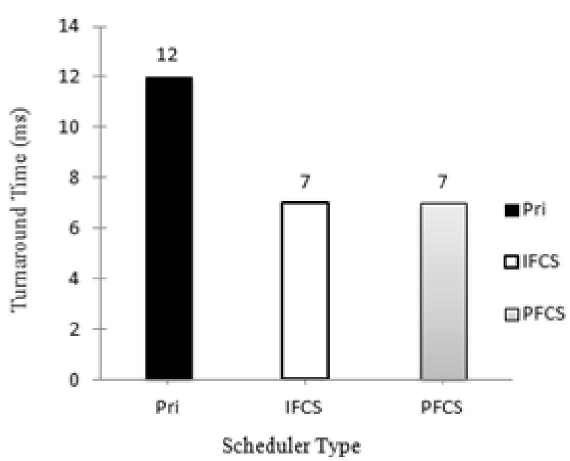

Figure 8: Turnaround time comparison between various algorithms for Case Study-4.

6.1 DISCUSSION ON SIMULATION RESULTS

It is clear from various case studies presented above that IFCS and PFCS show almost same result in terms of turnaround time and waiting time. These results show that system performance is far improved when compared to priority scheduling. Thus, when process is scheduled according to dynamic priority (dpi), it helps in reducing average turnaround time and average waiting time, so system performance is improved.

But PFCS is more advantageous than IFCS as it reduces the extra burden of calculation of response ratio.

7. CONCLUSION AND FUTURE SCOPE

The Proposed Fuzzy CPU Scheduling Algorithm reduces cost of calculating response ratio. In order to obtain an efficient scheduling algorithm, two membership functions μp and μb deals with both task priority and its execution time are evaluated to find dynamic priority so as to overcome the shortcomings of well-known scheduling algorithms. This Proposed Fuzzy CPU Scheduling Algorithm can further be improved by improving Fuzzification process. A new Fuzzy neuro based CPU scheduling algorithm can be generated. This Proposed Fuzzy CPU Scheduling Algorithm can be further improved by choosing more accurate formula for evaluating fuzzy membership value which may further reduce the waiting time and turnaround time.

ACKNOWLEDGEMENT

First of all, let me thank the almighty GOD, my parents and my In- Laws who are the most graceful and merciful for their blessings that contributed to the successful completion of this research. I take the opportunity to offer sincere thanks and deep sense of gratitude to my learned supervisor Mr. Manoj Sethi for expressing his confidence in me. I am thankful to my husband Mr. Rahul Ajmani for his moral support all the time; he has been always around to cheer me up in the odd times.

REFERENCES

[1]. Shata J. Kadhim, Kasim M. Al-Aubidy, "Design and Evaluation of a Fuzzy Based CPU scheduling Algorithm", *Springer-verlag Berlin Heidelberg 2010, CCIS 70*, pp. 45-52, 2010.

[2]. Jeegar A Trivedi and Dr.Priti Srinivas Sajja, "Improving efficiency of round robin scheduling using neuro fuzzy approach", *IJRRCS*, Vol.2, No. 2, April 2011.

[3]. Mahdi Hamzeh, Sied Mehdi Fakhraie and Caro Lucas, "Soft real time fuzzy task scheduling for multiprocessor systems", *International Journal of Intelligent Technology*, Vol.2, No. 4, 2007.

[4]. C. Lin and S. A. Brandt, "Efficient soft real-time processing 1in an integrated system", *in Proc. 25th IEEE Real-Time S1ystems Symp,* 2004.

[5]. H. S. Behera, Ratikanta Pattanayak and Priyabrata Mallick, "An Improved Fuzzy-Based CPU Scheduling (IFCS) Algorithm for Real Time Systems", *International Journal of Soft Computing and Engineering (IJSCE)*, Volume-2, Issue-1, March 2012.

[6]. Mamdani E.H., Assilian S., "An experiment in linguistic synthesis with a fuzzy logic controller", *International Journal of Man-Machine Studies* ,Vol.7, No. 1, 1975.
[7]. Sugeno, M., "Industrial applications of fuzzy control", *Elsevier Science Inc.*, New York, NY, 1985.
[8]. Terry Regner, craig lacy, "An Introductory Study of Scheduling Algorithms", *CPSC 321: A Project Assignment*, Feb 2005.
[9]. Ajit Singh, Priyanka Goyal,"An Optimized Round Robin Scheduling Algorithm for CPU Scheduling", *(IJCSE) International Journal on Computer Science and Engineering*, Vol. 02, No. 07,2010.
[10]. Alexander Taranovsky, "CPU Scheduling in Multimedia Operating Systems", *Research Report*, 1999.

Book References
[11] Silberschatz, A., Peterson, J. L., and Galvin, P.B. - *Operating System Concepts*; 7th Edition; Addison Wesley, 2006.