IJCSE International Journal of Computer Sciences and Engineering Open Access
Survey Paper Vol.-7, Special Issue-14, May 2019 E-ISSN: 2347-2693

Effectiveness of Teaching and Learning CPU Scheduling Algorithms: A Survey

Sudhir K. Pandey[1*], Gopal Krishna[2]

[1,2] School of C and IT, School of Computing and Information Technology REVA University, Bangalore, India

Corresponding Author: ssddpp39@gmail.com

DOI: https://doi.org/10.26438/ijcse/v7si14.146151 | Available online at: www.ijcseonline.org

Abstract— CPU scheduling algorithms are integral part of learning operating system. Over the years, it has been experience that initially CS students face a lot of problems in understanding and further implementing the various Scheduling algorithm. Also generating and regenerating Gantt Charts is faced with difficulties by many CS students. However, teaching and learning CPU scheduling algorithms using conventional lectures and textbooks is faced with difficulties by many students. First, textbooks illustrate the CPU scheduling algorithms in an incomplete and unclear manner. Second, students solve problems manually. They don't receive any immediate feedback on their solutions. Third, due to time restriction, the teacher has to select a few small problems. To overcome these problems, this can be used as an efficient tool for teaching and learning CPU scheduling algorithms. The tool is also capable of doing calculations different effectiveness criteria of an algorithm like waiting time of each process, average waiting time and turnaround time.

Keywords- CPU utilization and system throughput.

I. INTRODUCTION

Short term scheduling is called as CPU is scheduling or process scheduling or memory scheduling. CPU scheduling a process which is loaded in memory (ready state) to run in CPU is known as dispatching. CPU scheduling decided which ready process to run next in CPU and which running process to time out. Process may used both CPU and I/O. if process need more CPU execution and less I/O service then such process is called as CPU bound process or compute bound process if process need less CPU execution and more I/O service then such process is called I/O bound process. CPU bound process keep the CPU busy and I/O bound process keep the disk busy CPU scheduling is used in multiprogramming system by swatting the process called as context switch or process which during context switch there is no used full work done by CPU to user. The type of process in this system will affect the performance of scheduling algorithm. A short term CPU scheduling decision is needed.

In multiprogramming system, maximum CPU utilization this is the fundamental function of any operating System and is called as CPU scheduling[1][5] In sections II, III we present the theory of Popular CPU scheduling algorithms, which are implemented in the proposed tool. In section IV the details of the conclusion. The details are spread into different subsections, each highlighting the various features available with the application.

II. THEORY

(A) Some basic term O.S use full of the CPU Scheduling

(B) CPU bond process

The process required more amount of time are called as CPU bound process the process will spend more amount of time is running state

(C) I/O bound process

The process which required more amount of I/O time or called as I/O bound process the process is spend more amount time in wait or block state

(D) degree of multiprogramming

Number of process in the main memory at any point of time is called as degree of multiprogramming each and every time is process is moving from one state to other state , then the context of the process will we change

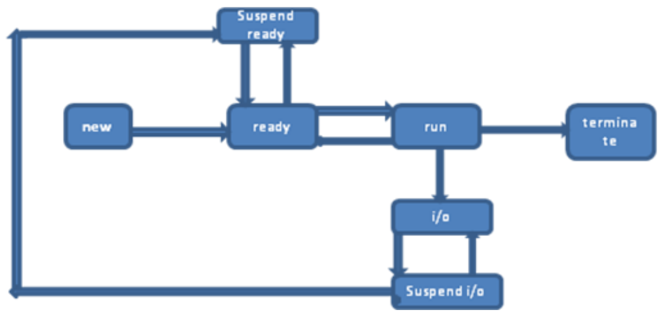
(E) context switching

Saving the context of the one process and loading of context of another processor is called as context switching when the context of process is more than the context switching is increasing when is undesirable context switching time is consider as overhead. There is a choice, however in circumstances 2 and 3[4].

When Scheduling takes place only under circumstances 1 and 4, we say the scheduling scheme is non-preemptive; otherwise the scheduling scheme is preemptive.

A. Non-Preemptive Scheduling

Under non preemptive scheduling, once the CPU has been allocated to a process the process keeps the CPU until it releases the CPU either By switching to the waiting state Non preemption scheduling is also called as Cooperative scheduling



A. Preemption scheduling

Whenever a process switches from the running state to the ready state or waiting state to the ready state and if scheduling takes place in these cases then the scheduling is called preemptive scheduling A scheduling is preemptive if once a process has been given the CPU can taken away Preemptive Scheduling in cases are:

- **CPU Utilization**
  CPU should be as busy as possible
- **Throughput**
  The number of process that time are completed per until time is called throughput
  Throughput=number of process/max(C.T)-min(A.T)
- **Turnaround Time**
  The time difference between completion time and Arrival time is called as turn around time
  T.A.T=C.T-A.T
- **Waiting Time**
  Waiting time is the sum of the period spend waiting in the ready queue or the time difference between T.A.T and arrival time is called as waiting time of process
  W.T=T.A.T-B.T
- **Response Time**
  The time from the submission of a requested until the first response is produced is called response time or the time difference between first response and arrival time is called as response time

III. SCHEDULING

B. Algorithm

To decide which process to execute first and which process to execute last to achieve maximum CPU utilization, computer scientists have defined some algorithms, and they are

C. First Come First Serve Scheduling

Criteria- Arrival time

Mode-non preemptive

The process that requested the CPU first is allocated to the CPU first its implemented using first come first serve Queue First come first serve scheduling algorithm is non preemptive the average waiting time in first come first serve is often quite long

Note- If the Arrival of the process or matching then scheduling the process which has lowest process id.

- A perfect real life example of FCFS scheduling is buying tickets at ticket counter.

Calculating Average Waiting Time

For every scheduling algorithm, Average waiting time is a crucial parameter to judge it's performance.

process	Arrival time	Burst time
P1	0	2
P2	1	3
P3	2	5
P4	3	4
P5	4	6

Gantt chart

P1	P2	P3	P4	P4

0 2 5 10 14 20

T.A.T for P1 =2 – 0=2

T.A.T for P2 =5-1=4

T.A.T for P3 =10-2=8

T.A.T for P4 =14-3=11

T.A.T for P5 =20-4=16

The Average T.A.T =2 + 4 + 8 + 11 + 16/5

Average T.A.T =8.2ms

Average waiting time

W.T for P1 =2-2=0

W.T for P2 =4-3=1

W.T for p3 =8-5=3

W.T for P4 =11-4=7

W.T for P5 =16-6=10

Average waiting time=0 + 1 + 3 + 7 + 10/5

Average waiting time =4.2 ms

The Gantt chart above perfectly represents the waiting time for each process.

1) *What is Convoy Effect?*

The first come first serve in the first process having CPU bound process then it will have major effect On Average waiting of the process this effect is called convoy effect.

D. Shortest Job First (SJF)

Shortest job first scheduler a process based on the selection of smallest burst time of process. Shortest job first assign a process with smallest burst time to the CPU also shortest job first is Non-preemptive algorithm Shortest job selection first because it finishes earliest shortest job first is also called as shortest time to completion first if all job have same service time then shortest job first work same as first come first serve

Non Pre-emptive Shortest Job First

All processes available in the ready queue for execution

Process	Arrival time	Burst time
P1	1	5
P2	2	3
P3	3	4
P4	4	1
P5	5	2

In Shortest job first scheduling The shortest Process is executed first hence the Gantt chart will be following

Ideal	P1	P4	P5	P2	P3

0 1 6 7 9 12 16

Find Average T.A.T

T.A.T for P1 =6-1=5

T.A.T for P2 =12-2=10

T.A.T for P3 =16-3=13

Turn Around Time for P4 =7-4=3

Turn Around Time for p5 =9-5=4

Average Turn Around Time =5 + 10 +13 +3 +4/5

Average Turn Around Time =7ms

Find Average waiting time

Waiting time for p1 =5-5=0

Waiting time for p2 =10-3=7

Waiting times for p3 =13-4=9

Waiting times for p4 =3-1=2

Waiting times for p5 =4-2=2

Average waiting time =0 + 7 + 9 + 2 + 2 /5

Average waiting time =4ms

Note

If burst time of the process or matching then

Then scheduling the process which lowest Arrival

Time

A. Pre-emptive Shortest Job First

The Gantt chart for preemptive shortest JFS scheduling

process	Arrival time	Burst time
P1	3	4
P2	4	2
P3	5	1
P4	2	6
P5	1	8
P6	2	4

	P5	P6	P6	P6	P6	P3	P2	P1	P4	P5

0 1 2 3 4 5 6 7 9 13 19 26

Average T.A.T find

T.A.T for p1 =10

T.A.T for p2 =5

T.A.T for p3 =2

T.A.T for p4 =17

T.A.T for p5 =25

T.A.T for p6 =4

Average T.A.T =63/6

Average T.A.T =10.5ms

Average waiting time

W. t for p1 =6

W .t for p2 =3

W. t for p3 =1

W .t for p4 =11

W. t for p5 =17

W. t for p6 =0

Average waiting time =38/6

Average waiting time =6.3ms

E. Priority Scheduling

It selected the highest priority process to run each process is assigned a priority of all processes ready to run the one with the highest priority gets to run next. Priorities may be internal or external internal priorities are determined by the system.

External priorities are assigned by administrator and priorities may also be statics or dynamic a process with statics priority keeps that priority for the entire life of the process.

Priority scheduling can be either pre-emptive or non-pre-emptive

Process	Arrival time	Burst time	priority
P1	4	6	4
P2	6	3	7
P3	3	4	6
P4	2	2	6
P5	1	3	1
P6	2	2	3

Gantt chart

I deal	5	4 p	2 p	3 p	1 p	6 p

0 1 4 6 9 13 19 21

AVG T.A.T FIND

TAT for p1 =19– 4=15

TAT for p2 =9– 6=3TAT for p3 =13 – 3=10

TAT for p4 =6 – 2=4

TAT for p5 =4 -1=3

TAT for p6 = 21 -2=19

AVG turn around time =4 + 3 +10 +19 +15 +3/6

AVG turn around time =9ms

Round robin scheduling

There is no non pre-emptive version of round robin every process has fixed time quantum, OS assigns a fixed time quantum (q) to each process for execution. If a process complete within given time quantum then immediately next process is scheduled by short term Scheduler for execution. If process need more time then given time quantum then after executing given time quantum then it will again go back to the ready queue If time quantum is grater then all burst time then round robin work as a first come first serve.

If time quantum is more small then more context switching Happened If the time quantum is long then context switching will decreasing and response time will we more Time quantum=2ms

Process	Arrival time	Burst time
P1	0	4
P2	1	5
P3	2	2
P4	3	1
P5	4	6
P6	6	3

Ready queue:

P1,p2,p3,p1,p4,p5,p2,p6,p5,p2,p6,p5

1	2	3	1	4	5	2	6	5	2	6	5

0 2 4 6 8 9 11 13 15 17 18 19
21

Find AVG turn around time

TAT for p1 =8 – 0=8

TAT for p2 =18 – 1=17

TAT for p3 =6 – 2=4

TAT for p4 =9 – 3=6

TAT for p5 =21 -4=17

TAT for p6 = 19 -6=13

Avg TAT =8 + 17 + 4 + 6 +17 + 13/6

Avg TAT =10.8ms

Find Avg Waiting time

Waiting time for p1 =8 – 4=4

Waiting time for p2 =17 -5=12

Waiting time for p3 =4 – 2=2

Waiting time for p4 =6 -1 =5

Waiting time for p5 =17 – 6=11

Waiting time for p6 =13-3=10

Avg waiting time =4 + 12 + 2 + 5 + 11 + 10/6

Avg waiting time =6.3ms

Higher response ratio next

Which higher response ratio next scheduling select a process which has highest response ratio to schedule as next process into CPU.

Highest response ratio next scheduler is non pre-emptive algorithm, also it is minimum the average turnaround time.

Highest response ratio next scheduling comparison between first come first serve and shortest job first.

It is favours both longest process and short process the process are waiting from longer will also get scheduled by highest response ratio next scheduler

Response ratio =response time/service time =turnaround time/service time

Response ratio= w+s/s

W=waiting time

S=burst time

process	Arrival time	Burst time
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2

P1	P2	P3	P5	P4

0 3 9 13 15 20

Waiting time=completion time – arrival time

W3=completion time of process p2 – arrival time of process p3

W3 =9 – 4 =5

Response ratio=w+s/s=5+4/4=2.25

W4 =9 – 6=3

Response ratio=3 + 5/5=1.6

W5=9 – 8=1

Response ratio= 1 + 2/2=1.5

Here p3 process response time grater then p4 and p5

Then first excuite p3 process.

Then again calculate

W4=13 -6=7

Response ratio =7 + 5/5=2.4

W5=13 -8=5

Response ratio =5 + 2/2=3.3

Here also p5 process response time grater then p4.

Multilevel Queue Scheduling

Depending on the priority of process in which particular ready in process. Has two place will we decided.

Height priority process will we place in top level ready queue and low priority process will we place in the bottom level ready queue.

Only of the completion of all process from the top level process

If this strategy following then the process which or place in the bottom level ready queue will suffer from starvation

Each queue has its own scheduling algorithm for example separate queue might be used for foreground and background process and these process might have separate scheduling among fixed priority pre-emptive scheduling

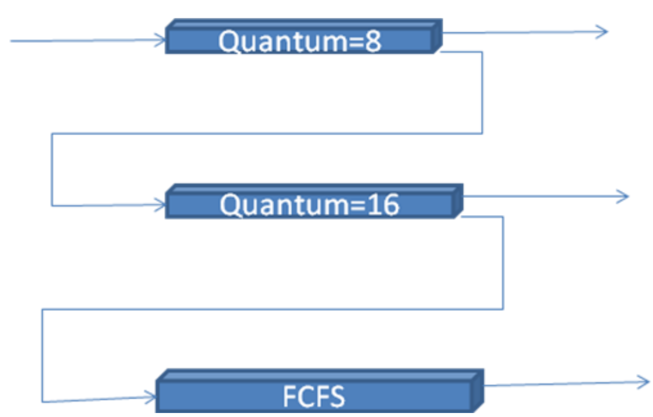
SYSTEM PROCESS

INTERACTION PROCESSES
INTERACTIVE EDITING PROCESS
BATCH PROCESS
STUDENT PROCESS

A. Multilevel feedback queue scheduling

It is allows a process to move between queue. It uses many ready queue and associates with Priority with each queue.

High priority queue often have a short time slice associated with them. If a process reaches the end of its time slice rather then blocking the scheduler demote it to the next lower priority queue



Lower priority queue are then assign increasing longer time slice. the process will get knocked to a lower priority level each time it run for the full quantum

It never choose a thread in queue I if there are threads in any queue k<i.

Thread in queue I use quantum qi<qk if I<k

Scheduling algorithm may lead to starvation list

Algorithm	starvation
First come first serve	No
Np-shortest job first	yes
Shortest remaining time first	yes
Round robin	No
Np-longest job first	yes
Longest remaining time first	No
Np-priority	yes
Preemptive priority	yes
Multi level queue	yes
H.R.R.N	No

Comparison table of scheduling algorithms

	FCFS	R.R	SPN	SRTP	HRRN	LRJF
Selection Function	Max	Contain	Min	Min S-C	Max(W +S/S)	-
Decision Mode	Non Preemptive	Pre-Emptive	Non Preemptive	Pre-Emptive	Non Preemptive	Pre-Emptive
Throughput	Non Preemptive	May Be Low	High	High	High	Low
Response Time	May Be High	Good Response Time For Short Process	Good Response Time For Short Process	Good Response Time	Good Response Time	Good
Overhead	Minimum	Minimum	Can Be High	High	Can Be High	High
Effect On Process	I/O Bound Process	Fair Treatment	Long Process	Long Process	Good Balance	High CPU Bound Process
Starvation	No	No	Possible	Possible	No	No

IV. CONCLUSION

Operating system is a mandatory course taught to cs student. CPU scheduling is a important part of designing an operating system. Student fined it difficult to grasp the basic concept and difference between various scheduling algorithms. Learning from text book has been made interesting using visual tools.

Future work includes: (1) finding out how effectively the tool can prove to be beneficial to the teachers and students , (2) improving the tool to cover deadlock avoidance algorithms,(3) modifying the tool to support multiprocessor scheduling.(4) tools support CPU-CPU scheduling.(5) multilevel queue scheduling (6) thread scheduling (7) Virtualization and scheduling (8) we also plan to add more features of operating system mainly the concept of memory management like paging, segmentation and fragmentation.

REFERENCES

[1] S. Abraham, Peter B. Galvin and Greg Gagne, “Operating System Concepts, ” 9th ed., John Wiley & Sons, 2012.

[2] S. William, “Operating systems: internal and design principles,” 8th ed., Prentice Hall, Person Education, 2015.

[3] William Stallings Operating Systems ISBN 0-13-031999-63241 Prentice Hall

[4] Gen M., and Cheng, R., "Genetic Algorithms and population. Out of the above final Design" John Wiley & Sons Inc. 2000.

[5] S. Grissom, M. McNally, and T. Naps, “Algorithm visualization in CS education: comarison levels of student engagement,” Proceedings of the 2003 ACM Symposium on Software Virtualization, pp. 87-94, 2003.

[6] ufDmoarnnn, Jp. pa.n6d29-Gi6r5s4c.h, M. (1994) Genetic Operators Based on Constraint Repair, [6] uf Dmoarnnn, Jp. pa.n6d29-Gi6r5s4c.h, M. (1994) Genetic Operators Based on Constraint Repair, ECAI'94 Workshop on Applied Genetic and other Evolutionary Algorithms, Amsterdam, August 9.

Authors Profile

Mr. Sudhir pandey pursed Bachelor of Engineering from I.E.T Dr R.M..L..A University, Faizabad(U.P) in 2011 and perusing Master of Technology in Computer Science from Reva University.

Dr. Gopal Kirshna Shyam received BE and PhD in Computer science and engineering from VTU, Belagavi His research interest includes Cloud Computing, Grid computing, High performance computing etc.
He has published about 10 papers in highly reputed National/International Conferences like IEEE, Elsevier etc. and 5 papers in Journals with high impact factor like Elsevier Journal on Network and Computer Applications and International Journal of Cloud computing (INDERSCIENCE). His research articles on Cloud computing co-authored by Dr. Sunilkumar S. Manvi have been cited by several researchers. He is a lifetime member of CSI and is actively involved in motivating students/faculties to join CSI/IEEE/ACM societies.

