School Engineering and Applied Science
(Ahmedabad University)
Course:*Operating Systems*

Project On

CPU Scheduling Algorithms

Instructor

Dr. Sanjay Chaudhary

Mentors

Purnima Shah
Mayank Jobanputra
Saumil Shah
Shashwat Sanghvi

Group Members

Akhil Vavadia (1401095)
Harsh Soni (1401121)
Pinak Divecha (1401098)
Vatsal Patel (1401123)

Abstract:

Effective scheduling is a basis to achieve system objectives in optimum way. The process manager is responsible for resource utilization in efficient manner and to execute various processes in a perfect way. In scheduling all processes are handled on the basis of particular strategy. The performance of system is primarily based on CPU scheduling. Fundamentally, scheduling is a matter of managing queues and to decide which of the process have to be executed next to achieve high efficiency level. Scheduling algorithms deals to minimize queuing delay and to optimize performance of queuing environment. In this paper, some common scheduling approaches like First Come First Serve, Shortest Job First, Priority Scheduling and Round Robin Scheduling are studied and reviewed on the basis of their working strategy. Scheduling techniques are analyzed on system objectives like waiting time and turnaround time and it is highlighted that which algorithm is appropriate for any particular situation.

***Keywords*:** Cpu Scheduling, First come First Serve, Shortest Job First , Priority Scheduling, Round Robin Scheduling,theads,inter process communication,mutual exclusion.

Introduction:

Operating system is a collection of system programs which control all operations of computer system and works as an interface between user and system. The fundamental objective of an operating system is to provide an atmosphere where various processes can be executed in convenient manner. It provides appropriate mechanism so that available resources can be used among all processes in an optimum way. It keeps each resource status and decides control over computer resources. It handles I/O programming and controls allocation of resources among various users and tasks and during execution it manages those resources also. As scheduling is a core function of any operating system hence almost all computer resources are scheduled before use. In our project we have implemented four algorithms for controlling the processes which are First Come First Serve, Shortest Job First, Priority, Round Robin. Based on the algorithm, results will be displayed on the terminal. For thread handling we have used signalling. We implemented thread using posix library. In our project we have analysed the scheduling performance on the following criteria

1. **Waiting Time**: It is the amount of time a process waits in the ready queue.
2. **Turnaround Time**: It is the amount needed for execution of a single process.
3. **CPU utilization**: The maximum use of CPU when it is busy
4. **Throughput**: It is the number of processes that complete there execution per unit time.
5. **Cpu burst Time**: It is the amount of time process spend in cpu till it completes.
6. **Response Time**: This is the amount of time takes from when a request was submitted until the first response is produced not output.

Technical Specifications:

In our project there are two main points we focused on are **Implementation of Cpu Scheduler** and **Analysis of various algorithms**.

Implementation:As shown in below Fig:1 there are total three types of scheduling is done in real systesms

Long term:Schedules the way processes are choosen from secondary memory to main memory for execution so it takes processes from new state to ready state.

Medium term: Schedules processes from suspended(block/ready) state to ready state.

Short term:Schedues processes which are in ready state and gives it to processor for execution.so it changes process state from ready to running.

In our project we implemented short term scheduler using three algorithms namely First Come First Serve,Shortest Job First,Priority base and Round Robin.We have made complete framework for cpu sheduler on user level.In real system what happens is we write a program in high level language(c,c++,java etc.) and we compile it using appropriate compiler and convert it in assembly language now when we want execute a program it is selected by long term scheduler and puts it in main memory where process image of program(now program becomes process) is created.Process image contains Process control block(PCB),stack,program,variables.In our part we have used only PCB which stores P_id,P_name,waiting time,Cpu burst time(from logs),response time.Now a ready queue is formed in accordance to choosen scheduling algorithm by OS.Now process from ready queue is gives to processor one by one(which is short term scheduling).In our project all assembly file which contains instruction that program need to execute are stored in 'programs' folder so it depicts the secondary memory in our project and when our program executes it takes program files from that folder and creates PCB for each of this program(now process) and creates array of PCB(which is ready queue) which is then set according to algorithm.Here in our project we made a parser program which does all arithmatic and logic operations and act as ALU inside processor.

In our project we implemented all things in user level mode so we can't take actual process inside cpu and schedule it but instead we created threads using posix library(pthread_create()) and applied mutual exclusion(using mutex lock) on processor so no other threads can interfere thread using cpu at a time. Instead of Inter process communication we implemented inter thread communication using signalling mechanism(pthread_kill).

Analysis of various algorithms: As our implemented cpu scheduler has restrictions we can't apply it for say 100 or 1000 processes as we have to give it that many instruction set also we have taken assumtion like every statement take one second to execute and setted time slice according so it doesn't provide reliablity for analysis.So for analysis part we have made combine program for all algorithms where we only give number of process and time slice as input and it generates analysis table for each algoritm and we can run it for different no of processes every time.

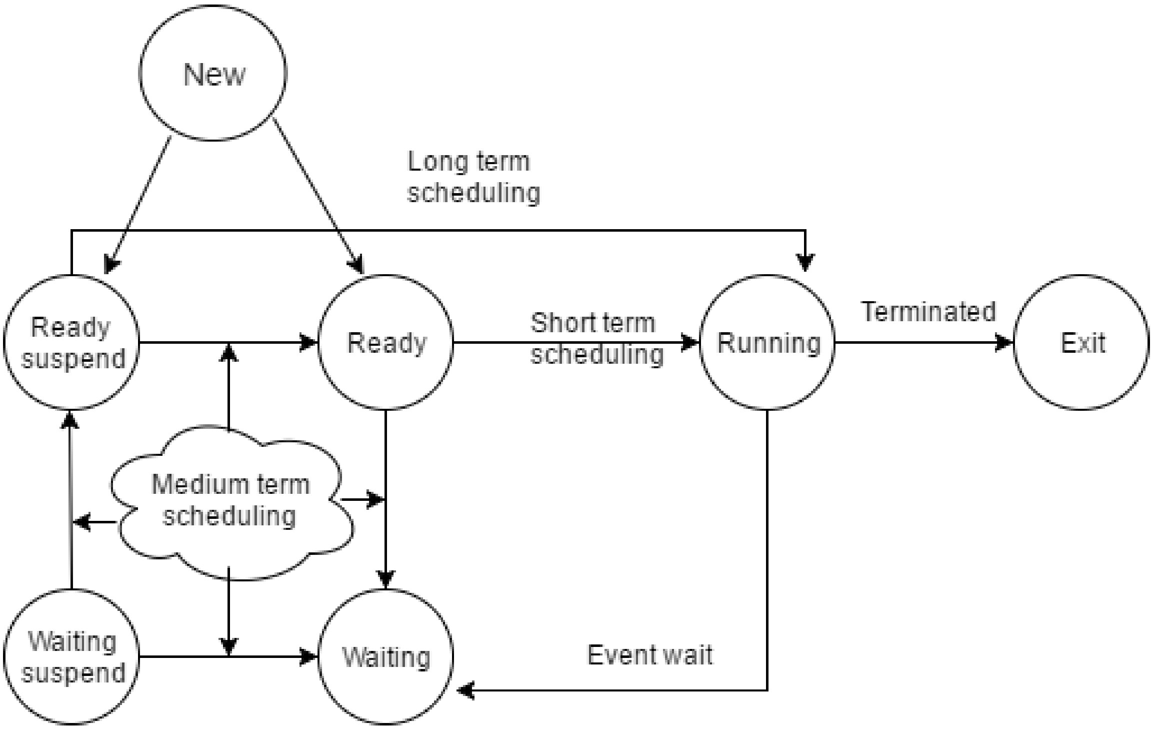


Fig:1(Process States)

Implementation: List of source codes

List of source codes developed, tested and implemented for this project

fcfsI.c: Contain c program to run First Come First Serve (FCFS) algorithm.

rrI.c: Contain c program to run Round Robin (RR) algorithm.

sjfI.c: Contain c program to run shortest job first (SJF) algorithm.

priorityI.c: Contain c program to run Priority scheduling algorithm.

parser.c: Contain c program to do arithmetic operation.

creatingprocess.c: Contains c program to create process by making process control block (pcb) of each program file(Parser) taken from 'programs' folder.

Combine.c:Contains c program which is combine of all four algorithms with all parameters generated randomly(this file we created for analysis purpose.)

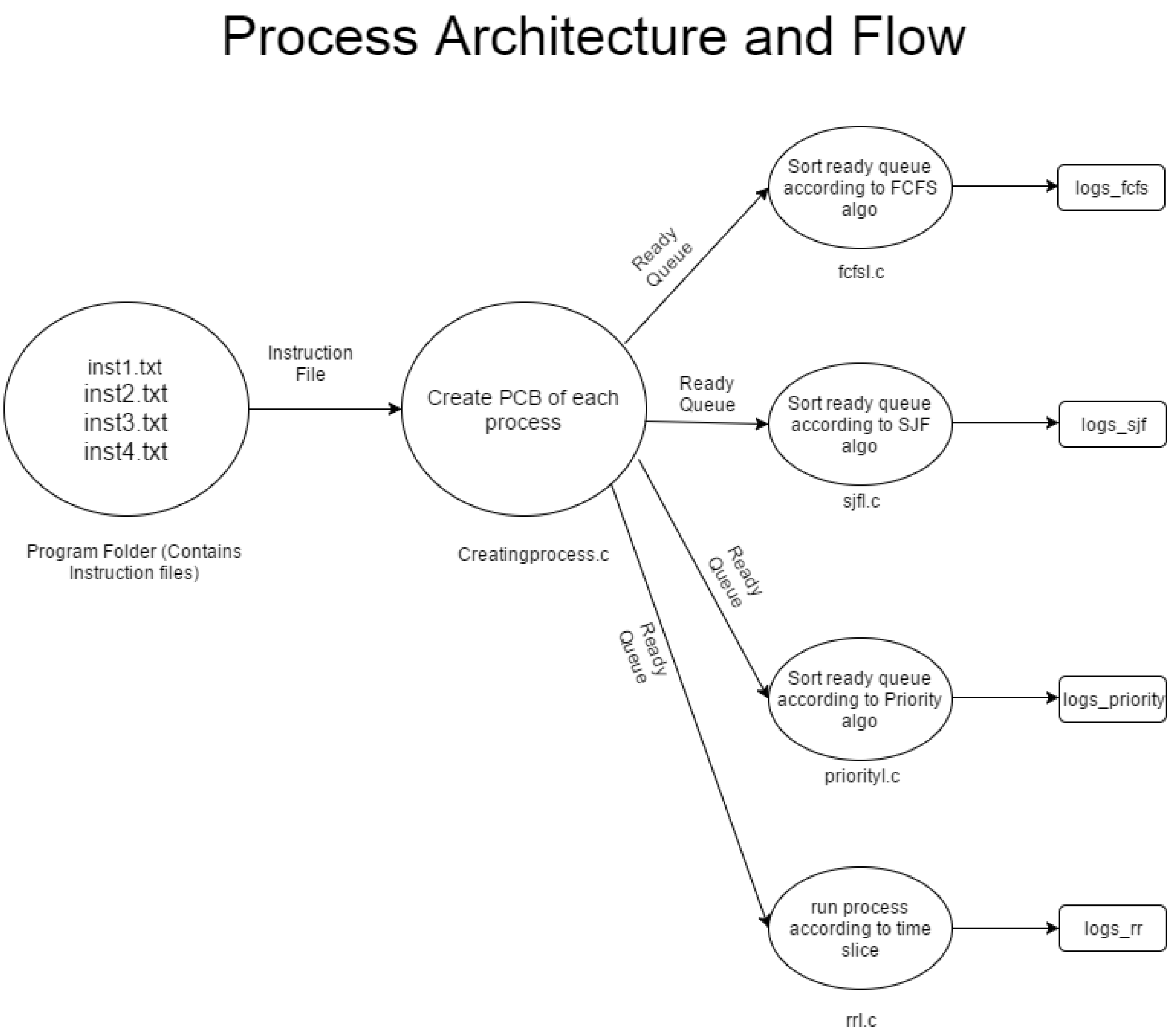


Fig:2(Flow chart)

Algorithms and Test data Sets:

General Working Algorithm:-creatingprocess.c

- Take instructions set of ready process from the 'programs' folder(secondary memory)
- Make PCB of each process. PCB contains pid, pname, arrival time, bust time, waiting time, priority.
- Fill each filed of PCB with initial values. Make an array of PCB which forms ready queue.

First Come First Serve (FCFS)-fcfsI.c

FCFS is a non preemptive scheduling algorithm. It uses First in- First out- FIFO strategy to assign the priority to processes in the order, that is same as the request made by process for the processor. The process or job that requests the CPU first is allocated the CPU first and other if in the queue has to wait until the CPU is free. Algorithm follows the given step:

Step1: Make an array(fcfs_array) from ready queue(p_queue).

Step2: Sort array according to arrival time of the process.

Step3: Take out process (thread) one by one and give it to CPU(parser.c)

Step4: Lock CPU till, process is completed.

Step5: Do step3 and step4 until queue is null.

Shortest Job First(SJF)-sjfI.c

In SJF technique the shortest amongst the entire ready queue job is executed first .The benefit if this is that waiting time is minimal for the shorter jobs. The SJF is especially appropriate for the batch jobs for which the run time(cpu_burst) are known in advance. Algorithm follows the given step:

Step1: Make an array(sjf_array) from ready queue(p_queue) .

Step2: Sort array according to burst time of the process.

Step3: Take out process (thread) one by one and give it to CPU(parser.c)

Step4: Lock CPU till, process is completed.

Step5: Do step3 and step4 until queue is null.

Priority-priorityI.c

In Priority scheduling algorithm each process is assigned priority by either an outer agency or as per their system requirements and as soon as each process hits the queue it is sorted in based on its priority so that process with higher priority are dealt with first. In case two processes arrive with same priority in different order then they are executed in FCFS order. The main advantage of Priority scheduling is that the important jobs can be finished first. Here algorithm follows following step:
(Note:In our case we have taken lower interger value as higher priority i.e 1 has higher priority than 2)

Step1: Make an array(priority_array) from ready queue(p_queue).

Step2: Sort array according to the priority of the process(highest first).

Step3: Take out process (thread) one by one and give it to CPU(parser.c).

Step4: Lock CPU till, process is completed.

Step5: Do step3 and step4 until queue is null.

Round Robin(RR)-rrI.c

In this approach a fixed time slot is defined before the execution of processes starts, which is a normally small unit of time. In each time slice (quantum) the CPU executes the current process only up to the end of time slice. If that process is having less burst time than the time slice then it is completed and is discarded from the queue and the next process in queue is handled by CPU. However, if the process is not completed then it is halted (preempted) and is put at the end of the queue and then the next process as per arrival time in line is addressed during the next time slice. Here algorithm follows following step:

Step1: Make a array(rr_array) from ready queue(p_queue).

Step2: Sort array according to arrival time of the process.

Step3: Take out thread (process) one by one and suspend it.

Step4: Give threads one by one send signal to resume it.

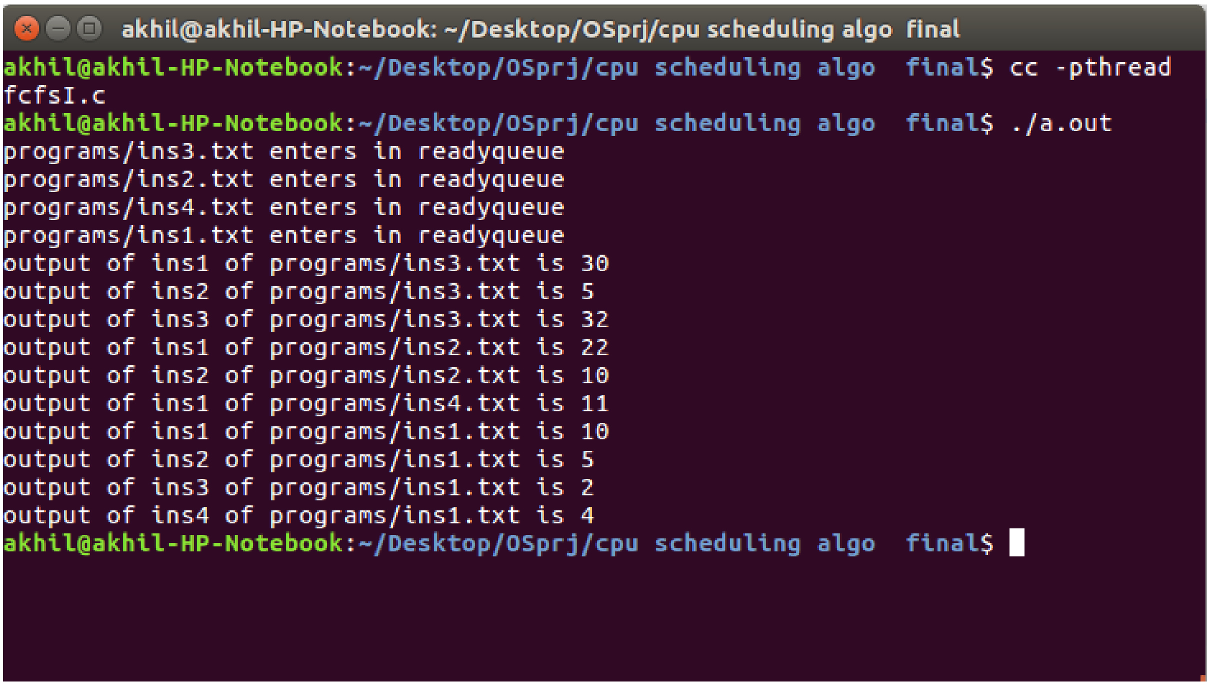
Step5: Give thread to CPU till time slice.

Step6: Send signal to pause (Preemtion)

Step7: Do step4, step5 and step6 till all thread completed.

Test Results:

First Come First Serve: Fig 1a shows the working of the algorithm in the terminal.



```
akhil@akhil-HP-Notebook: ~/Desktop/OSprj/cpu scheduling algo  final
akhil@akhil-HP-Notebook:~/Desktop/OSprj/cpu scheduling algo  final$ cc -pthread
fcfsI.c
akhil@akhil-HP-Notebook:~/Desktop/OSprj/cpu scheduling algo  final$ ./a.out
programs/ins3.txt enters in readyqueue
programs/ins2.txt enters in readyqueue
programs/ins4.txt enters in readyqueue
programs/ins1.txt enters in readyqueue
output of ins1 of programs/ins3.txt is 30
output of ins2 of programs/ins3.txt is 5
output of ins3 of programs/ins3.txt is 32
output of ins1 of programs/ins2.txt is 22
output of ins2 of programs/ins2.txt is 10
output of ins1 of programs/ins4.txt is 11
output of ins1 of programs/ins1.txt is 10
output of ins2 of programs/ins1.txt is 5
output of ins3 of programs/ins1.txt is 2
output of ins4 of programs/ins1.txt is 4
akhil@akhil-HP-Notebook:~/Desktop/OSprj/cpu scheduling algo  final$
```

Fig:1a

Fig 1b shows the logs file of FCFS

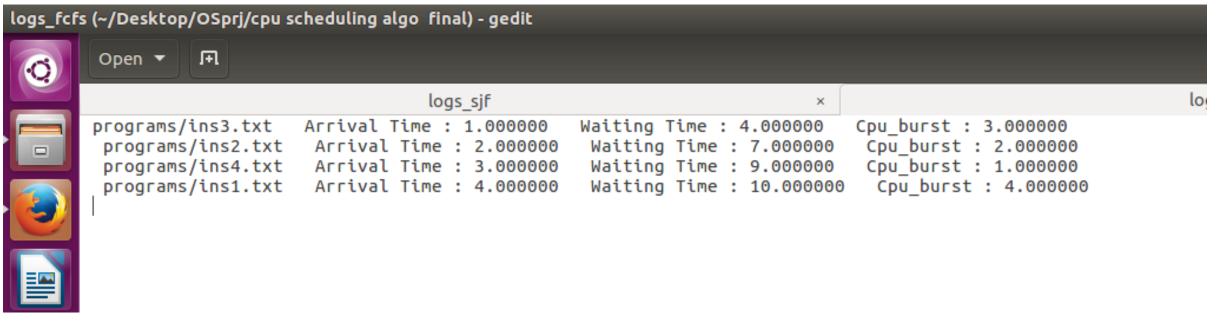


Fig:1b

Shortest Job First: Fig 2a shows the working of the algorithm in the terminal.

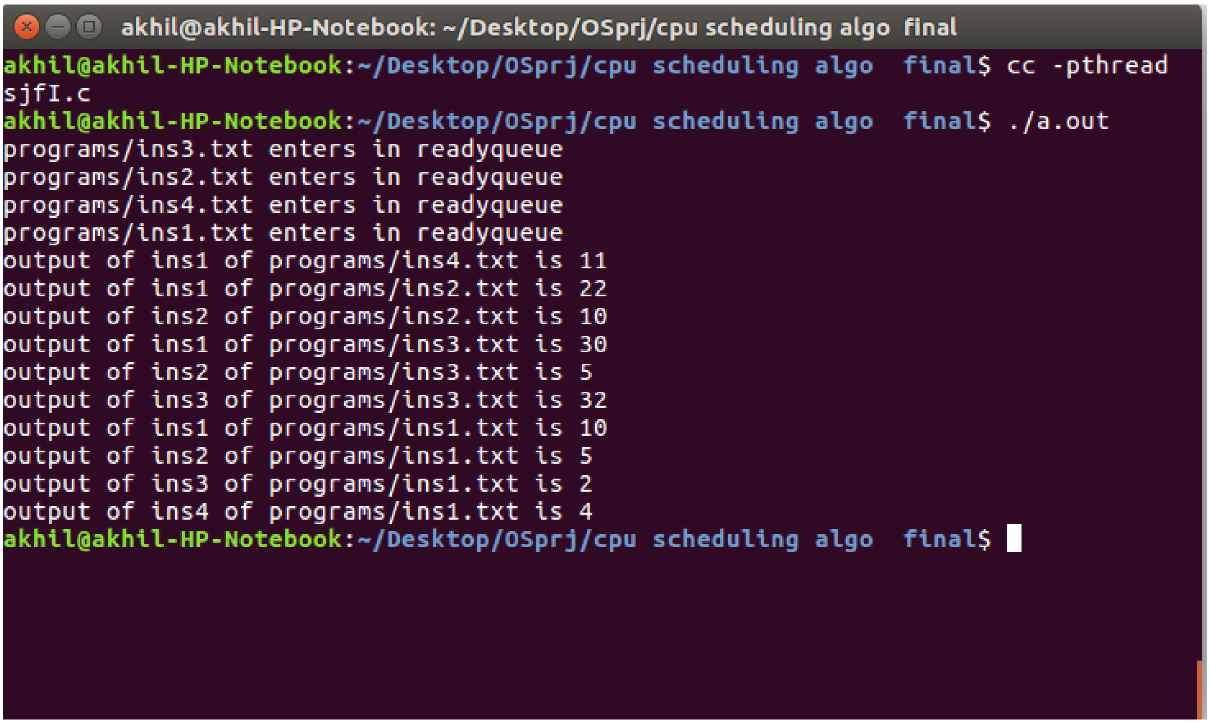


Fig:2a

Fig 2b shows the logs file of SJF

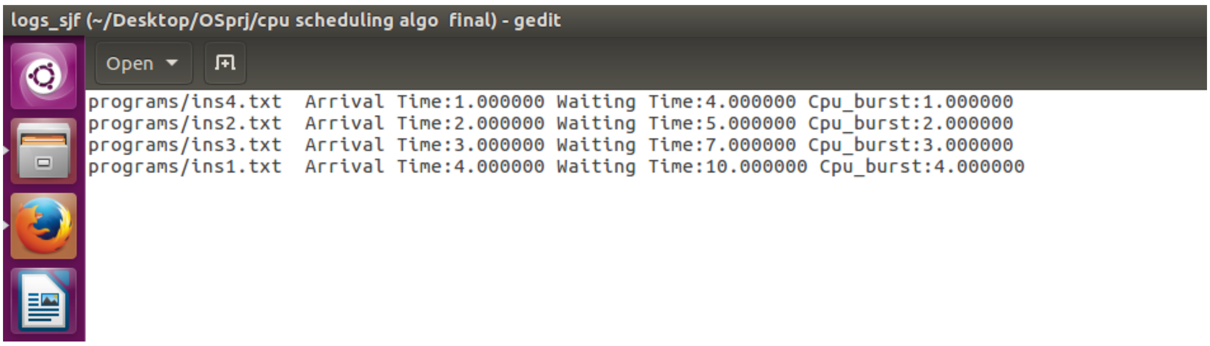


Fig:2b

Priority Algorithm: Fig 3a shows the working of the algorithm in the terminal.

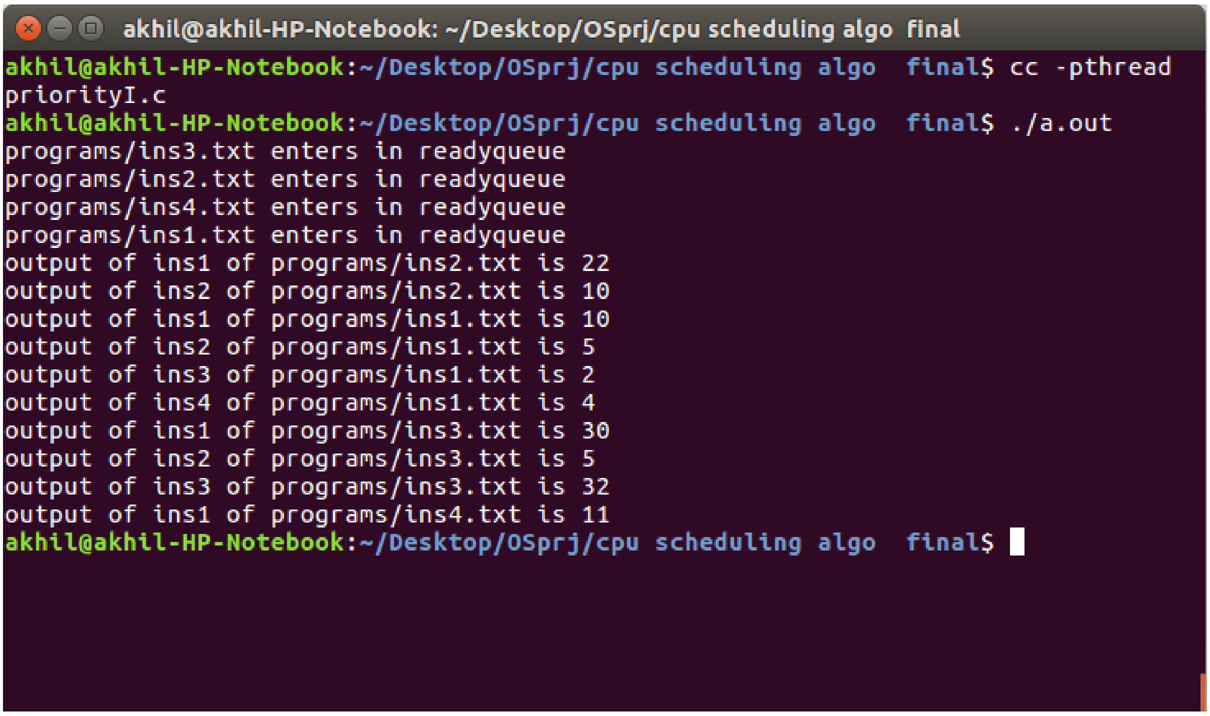


Fig:3a

Fig 3b shows the logs file of priority algorithm

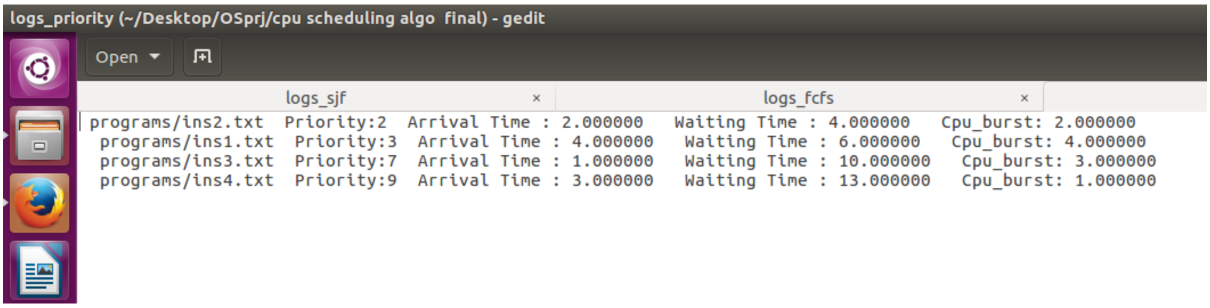


Fig:3b

Round Robin: Fig 4a shows the working of the algorithm in the terminal.

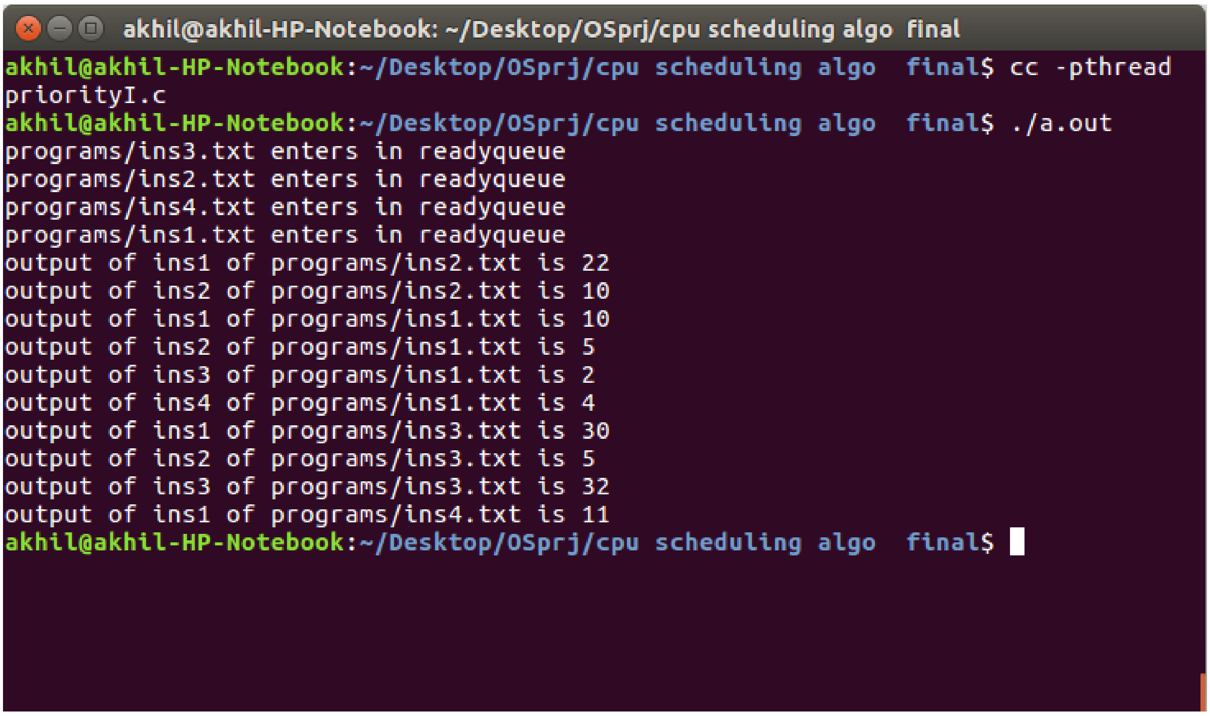


Fig:4a

Fig 4b shows the logs file of round robin algorithm

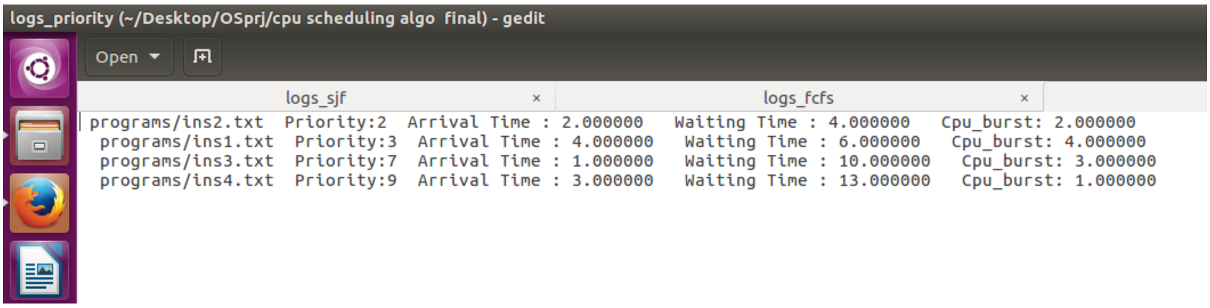


Fig:4b

Combine Output Of All Algorithms:

First come first serve:

```
Enter total number of processes: 7

Enter Time Quantum: 2

-----------------------FCFS---------------------------
Process          Burst Time       Waiting Time     Turnaround Time
P[1]             207              0                207
P[2]             3545             207              3752
P[3]             1508             3752             5260
P[4]             1468             5260             6728
P[5]             770              6728             7498
P[6]             4239             7498             11737
P[7]             3558             11737            15295

Average Waiting Time:5026
Average Turnaround Time:7211
------------------------------------------------------
```

Shortest Job First:

```
--------------------------SJF----------------------
Process          Burst Time       Waiting Time     Turnaround Time
p[1]             207              0                207
p[5]             770              207              977
p[4]             1468             977              2445
p[3]             1508             2445             3953
p[2]             3545             3953             7498
p[7]             3558             7498             11056
p[6]             4239             11056            15295

Average Waiting Time=3733
Average Turnaround Time=5918
-----------------------------------------------------
```

Priority:

```
-----------------------Priority--------------------------
Process          Priority         Burst Time       Waiting Time     Turnaround Time
P[2]             395              3545             0                3545
P[7]             1174             3558             3545             7103
P[3]             1190             1508             7103             8611
P[1]             1208             207              8611             8818
P[6]             2313             4239             8818             13057
P[5]             3239             770              13057            13827
P[4]             4411             1468             13827            15295

Average Waiting Time=7851
Average Turnaround Time=10036
------------------------------------------------------
```

Round Robin:

```
-----------------------RR--------------------------
Process          Burst Time       Waiting Time     Turnaround Time
P[1]             207              1236             1443
P[5]             770              4053             4823
P[4]             1468             6845             8313
P[3]             1508             6965             8473
P[2]             3545             11041            14586
P[7]             3558             11056            14614
P[6]             4239             11056            15295

Average Waiting Time= 7464
Avg Turnaround Time = 9649
------------------------------------------------------
```

Output Analysis:

We have taken four graphs for analysis as follows,

1.Each process Vs waiting time of that process for all algorithms

2.Each process Vs turnaround time of that process for all algorithms

3.No. of Process Vs avg. Waiting time of all algorithms

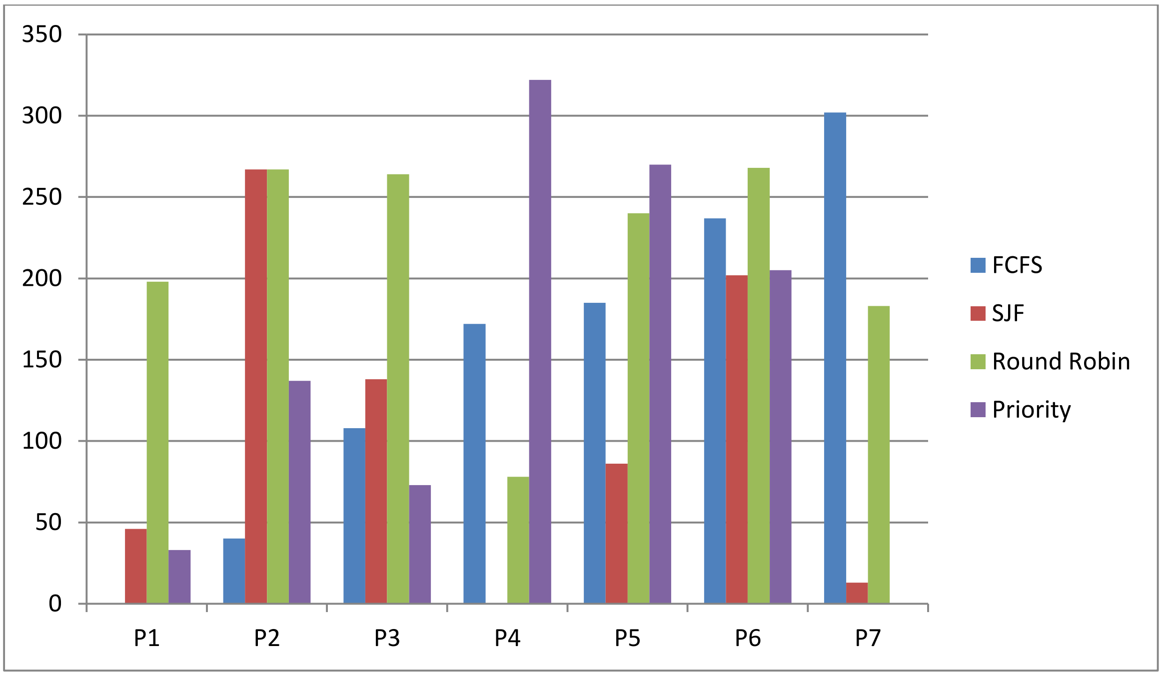
4.No. of Process Vs avg. Turnaround time of all algorithms.

1. 1.Each process Vs waiting time of that process for all algorithms

Table -1 Waiting Time(in ms)

Sr. No	Process	FCFS	SJF	Round Robin	Priority Algorithm
1	P1	0	46	198	33
2	P2	40	267	267	137
3	P3	108	138	264	73
4	P4	172	0	78	322
5	P5	185	86	240	270
6	P6	237	202	268	205
7	P7	302	13	183	0
Avg waiting time		150	108	214	149

Graphical Analysis (For waiting time)



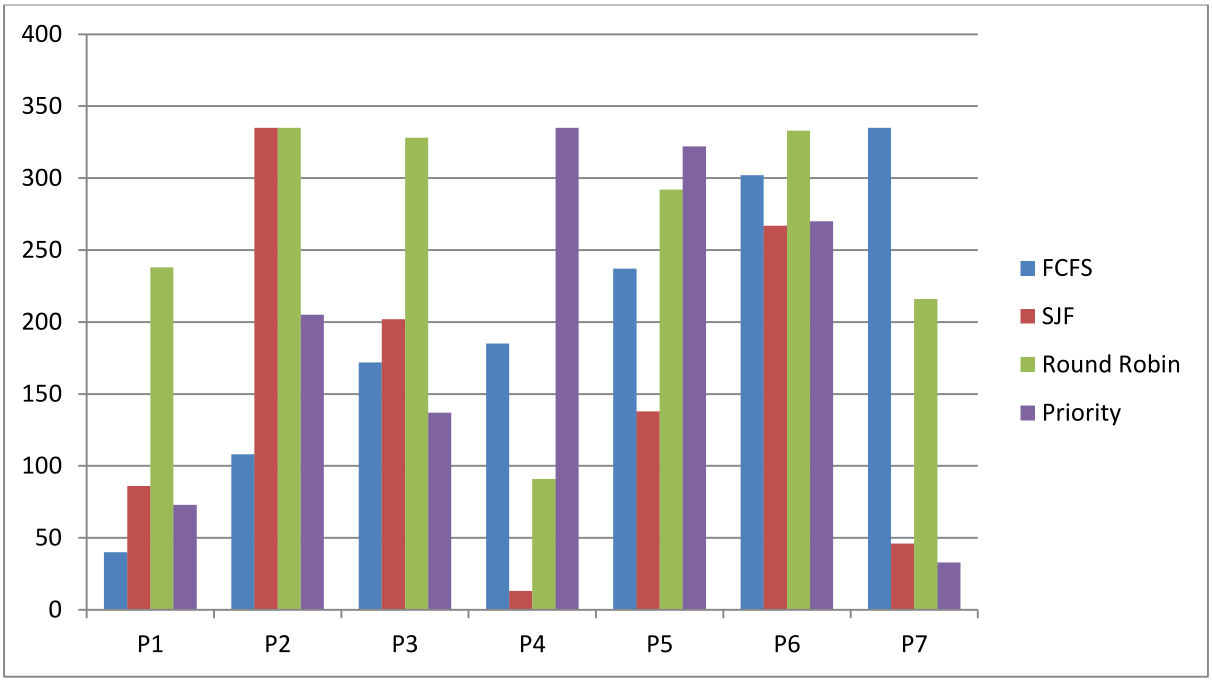
X->processes,Y->waiting time Graph:1

2.Each process Vs turnaround time of that process for all algorithms

Table-2 Turnaround Time(in ms)

Sr. No	Process	FCFS	SJF	Round Robin	Priority Algorithm
1	**P1**	40	86	238	73
2	**P2**	108	335	335	205
3	**P3**	172	202	328	137
4	**P4**	185	13	91	335
5	**P5**	237	138	292	322
6	**P6**	302	267	333	270
7	**P7**	335	46	216	33
Avg turn aroound time		197	156	262	197

Graphical Analysis (For Turnaround time)



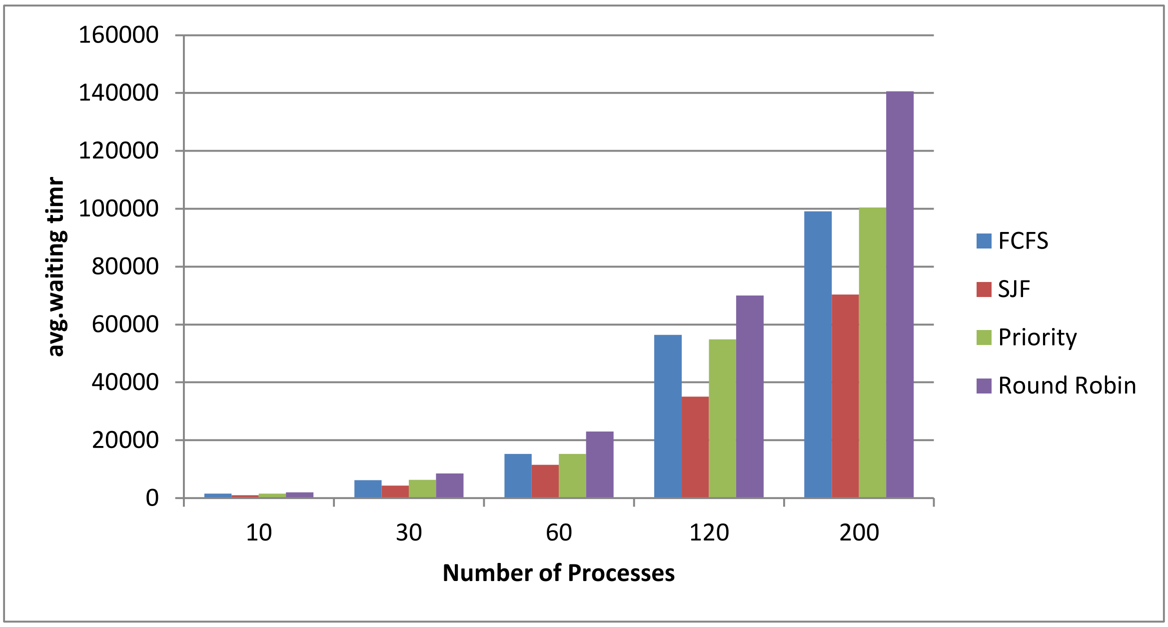
X->processes,Y-turnaround time   Graph:2

Analysis From above Graphs – 1 & 2,

- SJF gives minimum avg .waiting time.
- FCFS and Priority gives almost equal in both avg waiting and turnaround time.
- RR in general takes more avg waiting time and avg turn around time.

3.No. of Process Vs avg. Waiting time of all algorithms

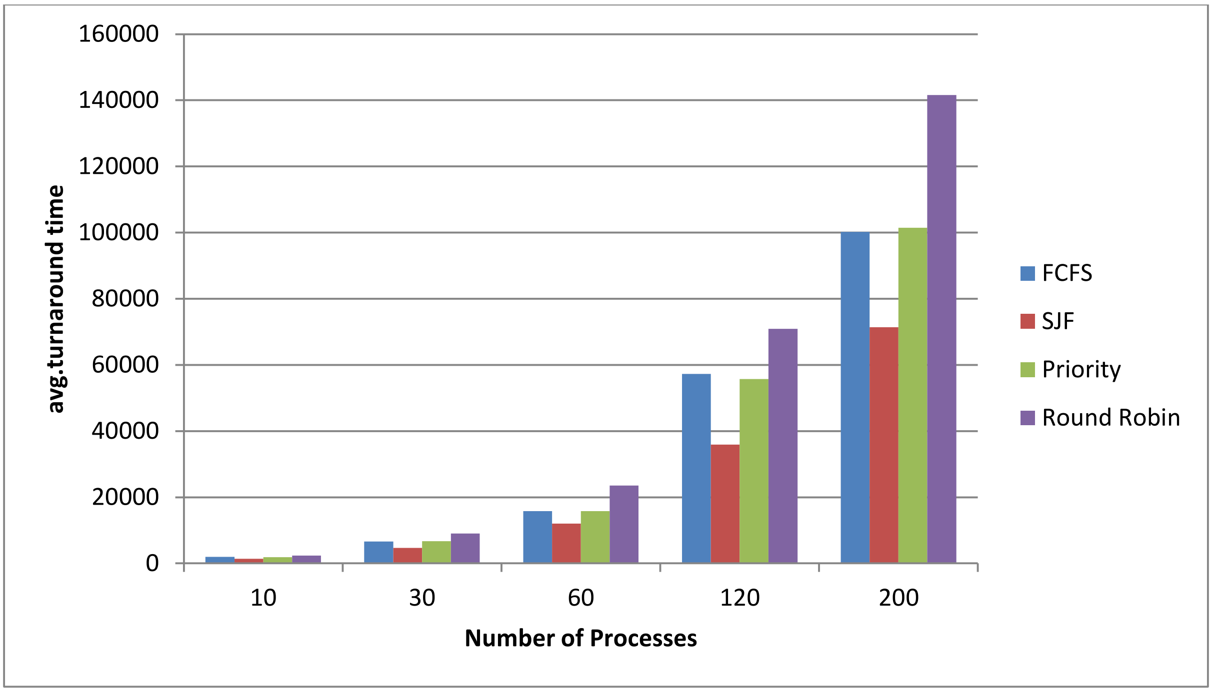
No. Of Processes	FCFS	SJF	Priority	RR
10	1587	1001	1553	1999
30	6207	4285	6294	8564
60	15280	11515	15282	23016
120	56379	35020	54833	70012
200	99139	70324	100478	140596



Graph:3

4.No. of Process Vs avg. Turnaround time of all algorithms.

No. Of Processes	FCFS	SJF	Priority	RR
10	1957	1371	1923	2370
30	6649	4728	6737	9006
60	15817	12051	15819	23553
120	57315	35956	55769	70948
200	100165	71354	101507	141625



Graph:4

Analysis from Graphs -3 & 4

- We can see as No. Of process increase avg. Waiting time and avg. Turnaround time increases.
- SJF has lowest avg. Waiting time and avg. Turn around time in any No. Of processes.
- When threre is large No. Of processes there is significant amount of difference between SJF and RR in both the cases.
- FCFS and Priority have nerly same value of avg.waiting time and avg.turnaround time which is increasing at nearly same rate with process.

Conclusion:

From the above calculative discussion and analysis it is infer that,

- First Come First Serve is easy to implement and mostly favourable for batch processing systems where waiting time is large.
- Short Job First works with optimum scheduling criteria and gives minimum average waiting time.
- Round robin scheduling is preemptive and based on policy of fair dealing of CPU to each process evenly. It works with interactive time sharing system.
- Round Robin reduces the penalty that short jobs suffer with FCFS by preempting running jobs periodically, and also saves starving of longer jobs and scheduling effort in case of SJF.

The main advantage of Round Robin Scheduling is that every process gets the CPU and thus there is no starvation.

- The priority scheduling algorithm is based on the priority criteria of execution from highest priority to lowest.

At last,It is recommended that any kind of simulation for any CPU scheduling algorithm has limited accuracy. The only way to evaluate a scheduling algorithm to code it and has to put it in the operating system, only then a proper working capability of the algorithm can be measured in real time systems.

References:

http://www.studytonight.com/operating-system/cpu-scheduling

Stalling, W. (2004): Operating Systems, fifth Ed., Pearson Education, Singapore, Indian Ed., New Delhi