Project Report

Atal Bihari Vajpayee Indian Institute of Information Technology and Management, Gwalior

विश्वजीवनामृतं ज्ञानम्

Long term and Short term Scheduling

Submitted by :

Arpit Mishra	2018-IMT-020
Mithilesh Kumar	2018-IMT-052
Mohd Wasiuddin Junaid	2018-IMT-053
Nitya Chandra	2018-IMT-060
Pankaj Ahakey	2018-IMT-061
Saurav Kumar	2018-IMT-091
Shivam Soni	2018-IMT-095

1. INTRODUCTION

The project aims at designing a process scheduling technique. In brief, it is about how to assign processes to be executed by the processor in a way that meets system objectives, such as response time, throughput, and processor efficiency.

PROCESS SCHEDULING

The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.
Process scheduling is an essential part of a multi programming operating systems. Such operating systems allow more than one process to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

Long Term Scheduler

It is also called a job scheduler. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the queue and loads them into memory for execution. Process loads into the memory for CPU scheduling. The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. On some systems, the long-term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When a process changes the state from new to ready, then there is use of long-term scheduler.

Short Term Scheduler

It is also called a CPU scheduler. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.

Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers

MULTI THREADING TECHNIQUE

Multi threading is a specialized form of multitasking and a multitasking is the feature that allows your computer to run two or more programs concurrently. In general, there are two types of multitasking: process-based and thread-based.

Process-based multitasking handles the concurrent execution of programs. Thread-based multitasking deals with the concurrent execution of pieces of the same program.

A multi threaded program contains two or more parts that can run concurrently. Each part of such a program is called a thread, and each thread defines a separate path of execution.

C++ does not contain any built-in support for multi threaded applications. Instead, it relies entirely upon the operating system to provide this feature.

In this project, we have worked on Linux OS and we have written a multi-threaded C++ program using POSIX. POSIX Threads, or Pthreads provides API which are available on many Unix-like POSIX systems such as FreeBSD, NetBSD, GNU/Linux, Mac OS X and Solaris.

SCHEDULING ALGORITHMS

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. These algorithms are either non-preemptive or preemptive. Non-preemptive algorithms are designed so that once a process enters the running state, it cannot be preempted until it completes its allotted time, whereas the preemptive scheduling is based on priority where a scheduler may preempt a low priority running process anytime when a high priority process enters into a ready state.

Following are some of the important Process Scheduling Algorithms and their characteristics:

1. First Come First Serve (FCFS)

- Jobs are executed on a first come, first serve basis.
- It is a non-preemptive, preemptive scheduling algorithm.
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance as average wait time is high.

2. Shortest Job Next (SJN)

- This is also known as shortest job first, or SJF
- This is a non-preemptive, preemptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.
- Impossible to implement in interactive systems where required CPU time is not known.
- The processor should know in advance how much time the process will take.

3. Priority Based Scheduling

- Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems.

- Each process is assigned a priority. Process with the highest priority is to be executed first and so on.
- Processes with the same priority are executed on a first come first served basis.
- Priority can be decided based on memory requirements, time requirements or any other resource requirement.

4. Shortest Remaining Time

- Shortest remaining time (SRT) is the preemptive version of the SJN algorithm.
- The processor is allocated to the job closest to completion but it can be preempted by a newer ready job with shorter time to completion.
- Impossible to implement in interactive systems where required CPU time is not known.
- It is often used in batch environments where short jobs need to give preference.

5. Round Robin Scheduling

- Round Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called a quantum.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.

2. ISSUES AND CHALLENGES

Scheduling Issues :

Application Profile:

- A program alternates between CPU usage and I/O
- Relevant question for scheduling: is a program compute-bound (mostly CPU usage) or I/O-bound (mostly I/O wait)

Multi-level scheduling (e.g., 2-level in Unix):

- Swapper decides which processes should reside in memory
- Scheduler decides which ready process gets the CPU next

When to schedule

- When a process is created
- When a process terminates

- When a process issues a blocking call (I/O, semaphores)
- On a clock interrupt
- On I/O interrupt (e.g., disk transfer finished, mouse click)
- System calls for IPC (e.g., up on semaphore, signal, etc.)

Is preemption allowed?

- Non-preemptive scheduler does not use clock interrupts to stop a process

What should be optimized?

- CPU utilization: Fraction of time CPU is in use
- Throughput: Average number of jobs completed per time unit
- Turnaround Time: Average time between job submission (or command issue) and completion
- Waiting Time: Average amount of time a process is ready but waiting
- Response Time: in interactive systems, time until the system responds to a command
- Response Ratio: (Turnaround Time)/(Execution Time) -- long jobs should wait longer

Different applications require different optimization criteria

- Batch systems (throughput, turnaround time)
- Interactive system (response time, fairness, user expectation)
- Real-time systems (meeting deadlines)

Overhead of scheduling

- Context switching is expensive (minimize context switches)
- Data structures and book-keeping used by scheduler

What's being scheduled by the OS?

- Processes in Unix, but Threads in Linux or Solaris

Challenges in Scheduling :

Type of Job :

When the process is allocated to the CPU, Scheduling of processes starts. At present, long term scheduler simply takes the job in ready queue & short term scheduler selects the job from ready queue & finally job is given to CPU. But the current challenge is that before entertaining any job by scheduler, it doesn't look for the type of job it is. Basically, there are two types of job. One is computational intensive and other is communication intensive. If the job is communication intensive then it would require higher bandwidth for the transfer of data and if it is computation intensive then it would require higher CPU speed. The computational jobs are the jobs that require processing time more than communication time. However, communicative jobs are the jobs that require communication time more than processing time. In computational intensive, the job considers execution time for processing and it also takes more power.

Reliability :

The primary concern in systems is of security and resources. A proper safety mechanism is required for grid resources, because if a resource is affected by virus or some malicious code then it will not execute the application safely and degrades the performance of the system9. The reliability factor (RF) value of each system is calculated through its self-protection capability and reputation weightage obtained from the user community on its past behavior.

Location Dependency :

In our distributed operating system, communication jobs should be submitted at a nearby location. By this, there would be no network congestion problem and packet delay has dropped to considerable rate which results in low traffic. If communication jobs are to be submitted at a very far location from the system then it would make network congestion a problem. In this, network problems increase and also increase the failure rates.

3. DESIGN AND LAYOUT

- We stimulate the working of Long Term and Short Term Scheduler using Threads. So to do so we have used Pthread to implement Threads.

https://media.geeksforgeeks.org/wp-content/uploads/20190514222912/Untitled-Diagram-124.png

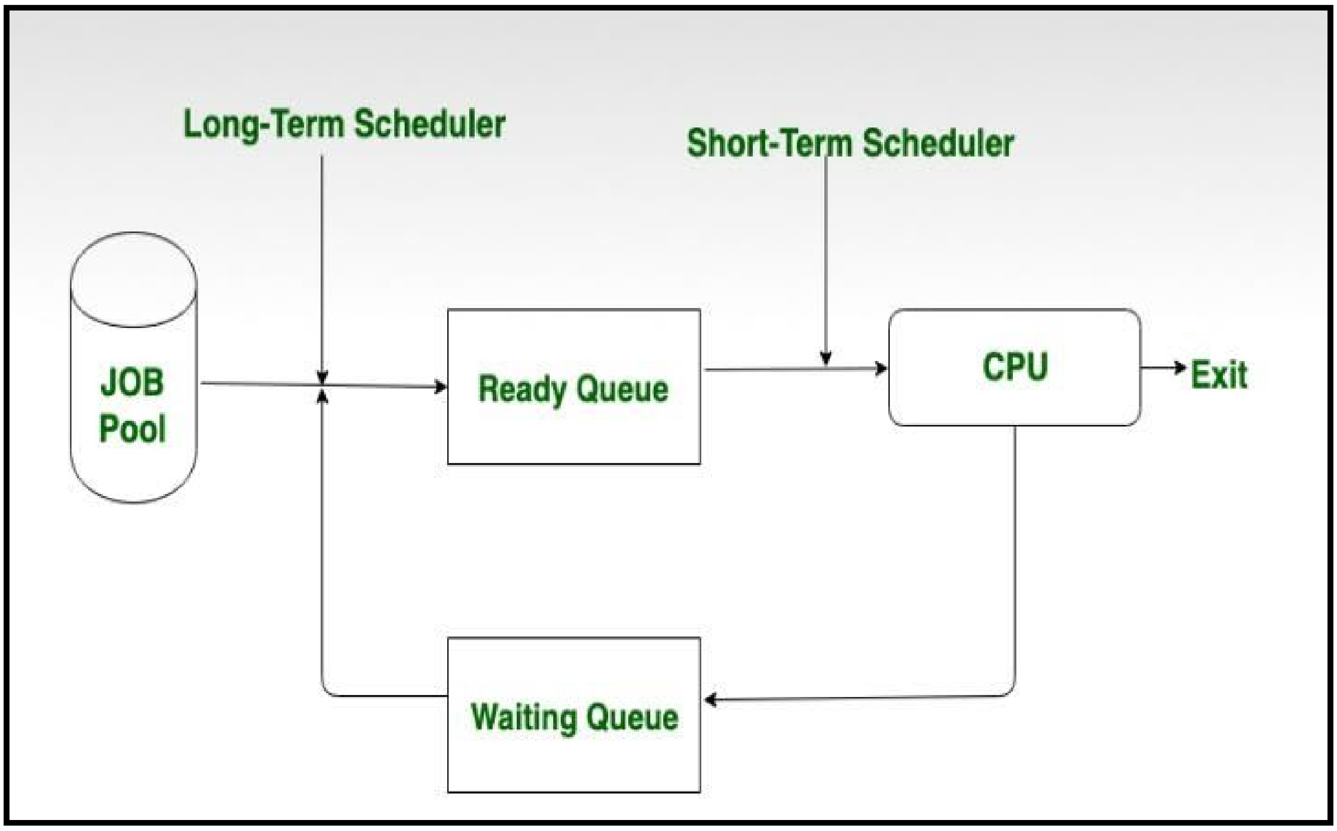


fig 0 : Flow diagram of our Project

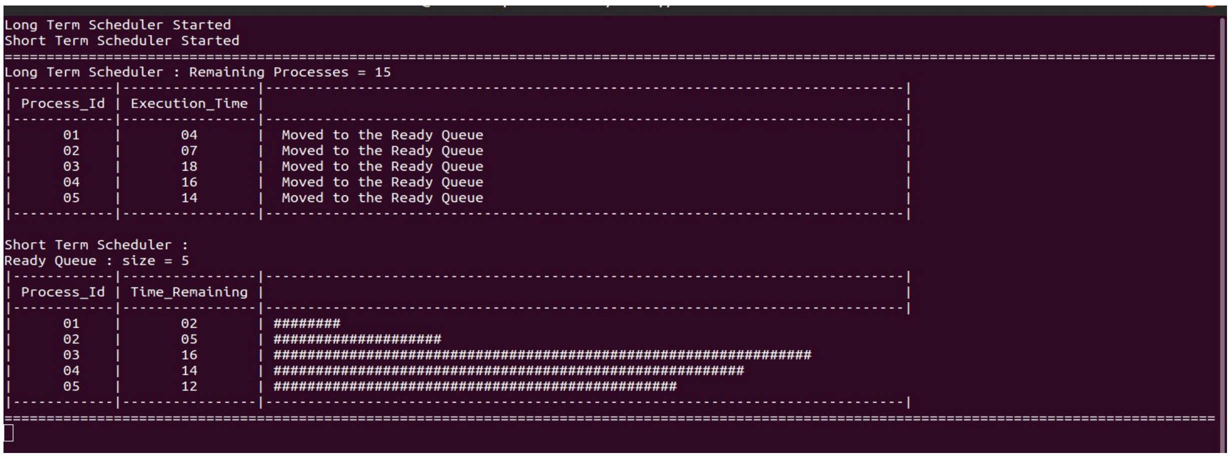
To start the Project we should compile the main file(secheduling.c) with -lpthread so that we can able to use pthread in our Project :

1. goto the folder containing project file
2. to compile type : **gcc scheduling.c -o scheduling -lpthread**
3. then type : **./scheduling** to start Project

- We have also created a web page to simulate the working of our project :
    1. First we have to start the rails server using command : **rails s**
    2. After opening the project , an index page is gonna show up where the program will fill up the Job pool.
    3. Next we have to click on the simulation button to start the main Program.
- Basically we have created two threads Named longScheduler andshortScheduler and we use mutex lock and check conditions to change the context between two. By this way we implement both Long Term and Short TermSchedulers in real time.
- All the processes have three main states : Waiting State, Ready State,Running State.
- Initially we have 15 Processes in our Job Pool. Now one by one all processes were enqueued to the processQueue - from where longScheduler picks the job.
- After this we start the longThread and shortThread. The size of our ReadyQueue is restricted to 5 and Long term scheduler keeps the readyQueue full until no remaining processes are left.
- For Short Term Scheduling we use Round Robin CPU scheduling with time quantum of 2. The main goal of short term scheduler is to boost the system performance according to set criteria
- We Represent the Ready Queue to illustrate the real time simulation of CPU on Console.

4. FINAL OUTPUT

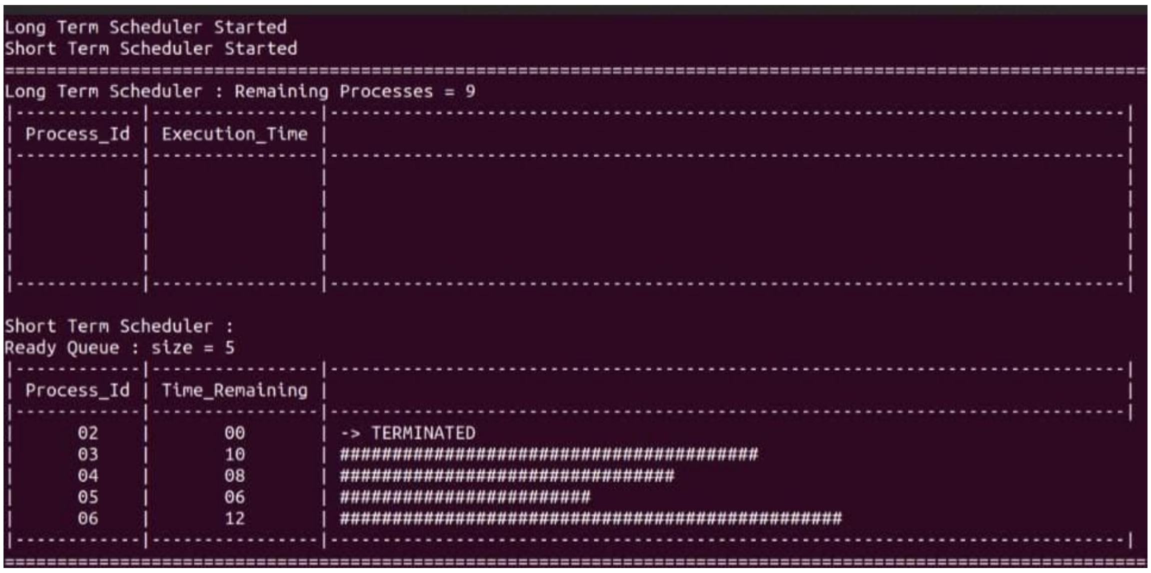
Output on LINUX Console :



```
Long Term Scheduler Started
Short Term Scheduler Started
===========================================================================================================================================================
Long Term Scheduler : Remaining Processes = 15
|------------|----------------|------------------------------------------------------------------------------|
| Process_Id | Execution_Time |                                                                              |
|------------|----------------|------------------------------------------------------------------------------|
|     01     |       04       |  Moved to the Ready Queue                                                    |
|     02     |       07       |  Moved to the Ready Queue                                                    |
|     03     |       18       |  Moved to the Ready Queue                                                    |
|     04     |       16       |  Moved to the Ready Queue                                                    |
|     05     |       14       |  Moved to the Ready Queue                                                    |
|------------|----------------|------------------------------------------------------------------------------|

Short Term Scheduler :
Ready Queue : size = 5
|------------|----------------|------------------------------------------------------------------------------|
| Process_Id | Time_Remaining |                                                                              |
|------------|----------------|------------------------------------------------------------------------------|
|     01     |       02       | ########
|     02     |       05       | ####################
|     03     |       16       | ################################################################
|     04     |       14       | ########################################################
|     05     |       12       | ################################################
|------------|----------------|------------------------------------------------------------------------------|
===========================================================================================================================================================
```

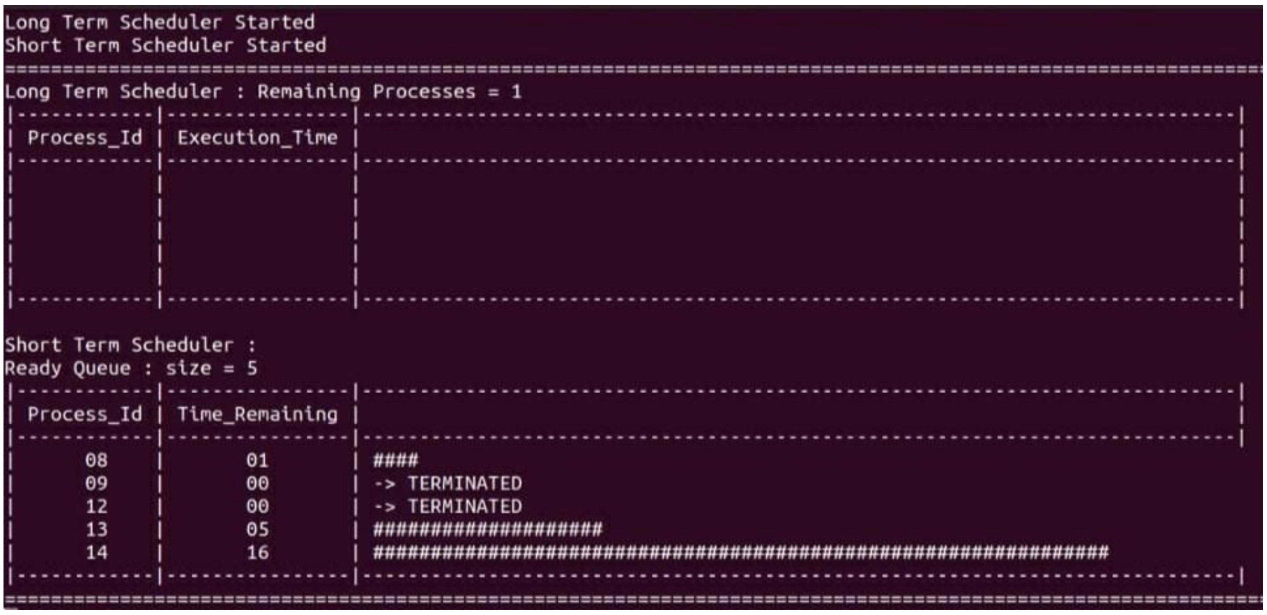
fig 1 : Long Term and Short Term Scheduler Started and 5 Processes were moved to Ready Queue



```
Long Term Scheduler Started
Short Term Scheduler Started
=================================================================================================================
Long Term Scheduler : Remaining Processes = 9
|-----------|---------------|-----------------------------------------------------------------------------|
| Process_Id | Execution_Time |                                                                           |
|-----------|---------------|-----------------------------------------------------------------------------|
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|-----------|---------------|-----------------------------------------------------------------------------|

Short Term Scheduler :
Ready Queue : size = 5
|-----------|---------------|-----------------------------------------------------------------------------|
| Process_Id | Time_Remaining |                                                                           |
|-----------|---------------|-----------------------------------------------------------------------------|
|     02    |      00       | -> TERMINATED
|     03    |      10       | ########################################
|     04    |      08       | ################################
|     05    |      06       | ########################
|     06    |      12       | ################################################
|-----------|---------------|-----------------------------------------------------------------------------|
=================================================================================================================
```

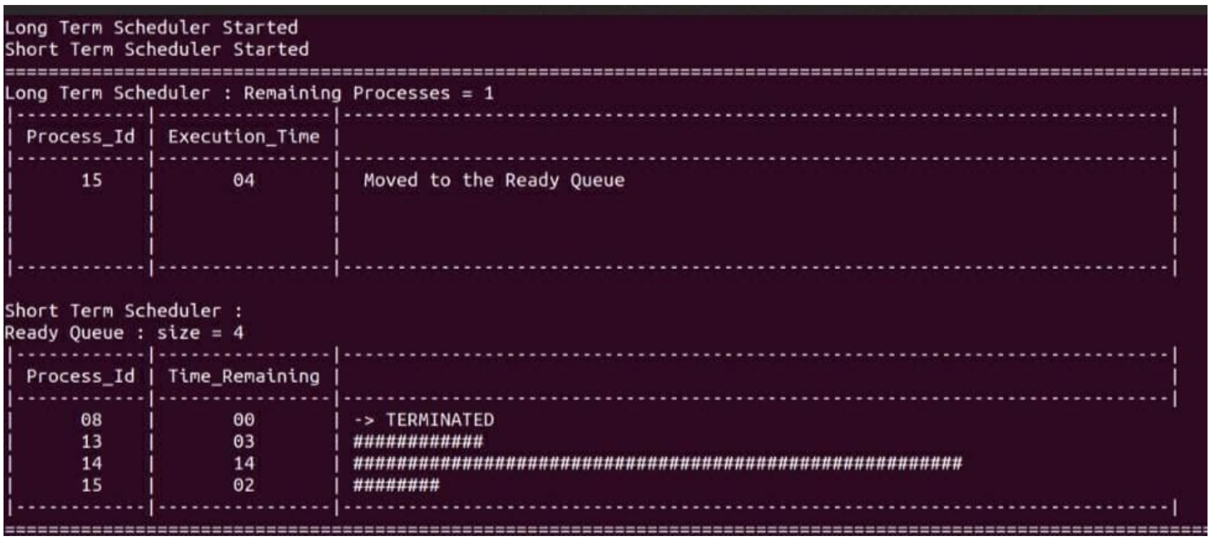
fig 2 : 9 Processes Remaining in the Job pool and 1 process is executed successfully



```
Long Term Scheduler Started
Short Term Scheduler Started
=================================================================================================================
Long Term Scheduler : Remaining Processes = 1
|-----------|---------------|-----------------------------------------------------------------------------|
| Process_Id | Execution_Time |                                                                           |
|-----------|---------------|-----------------------------------------------------------------------------|
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|-----------|---------------|-----------------------------------------------------------------------------|

Short Term Scheduler :
Ready Queue : size = 5
|-----------|---------------|-----------------------------------------------------------------------------|
| Process_Id | Time_Remaining |                                                                           |
|-----------|---------------|-----------------------------------------------------------------------------|
|     08    |      01       | ####
|     09    |      00       | -> TERMINATED
|     12    |      00       | -> TERMINATED
|     13    |      05       | ####################
|     14    |      16       | ################################################################
|-----------|---------------|-----------------------------------------------------------------------------|
=================================================================================================================
```

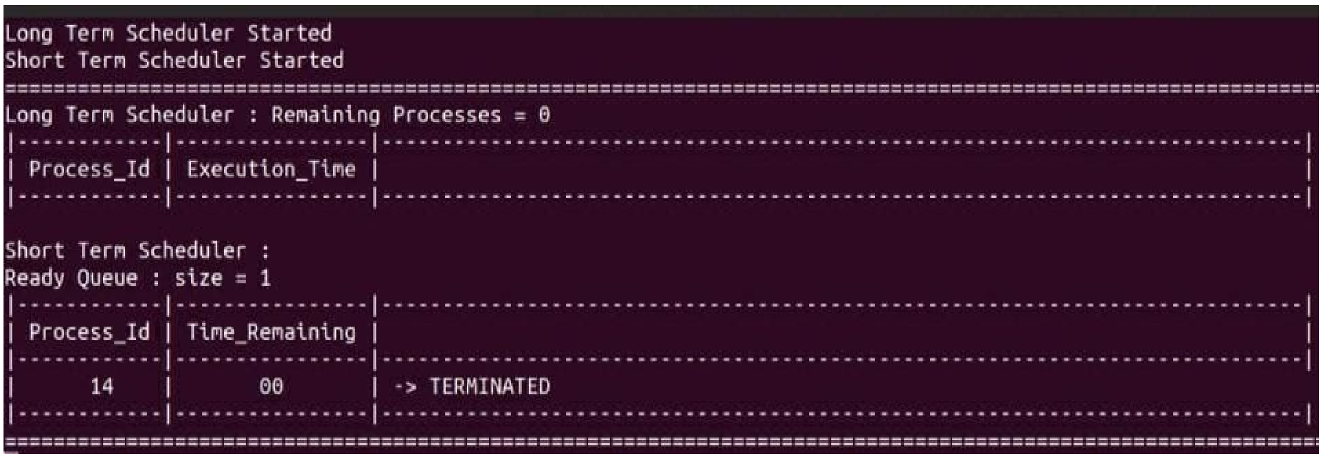
fig 3 : 1 Process Remaining in Job Pool and 2 Processes are executed successfully



```
Long Term Scheduler Started
Short Term Scheduler Started
=================================================================================================================
Long Term Scheduler : Remaining Processes = 1
|-----------|---------------|-----------------------------------------------------------------------------|
| Process_Id | Execution_Time |                                                                           |
|-----------|---------------|-----------------------------------------------------------------------------|
|     15    |      04       |  Moved to the Ready Queue                                                   |
|           |               |                                                                             |
|           |               |                                                                             |
|           |               |                                                                             |
|-----------|---------------|-----------------------------------------------------------------------------|

Short Term Scheduler :
Ready Queue : size = 4
|-----------|---------------|-----------------------------------------------------------------------------|
| Process_Id | Time_Remaining |                                                                           |
|-----------|---------------|-----------------------------------------------------------------------------|
|     08    |      00       | -> TERMINATED
|     13    |      03       | ############
|     14    |      14       | ########################################################
|     15    |      02       | ########
|-----------|---------------|-----------------------------------------------------------------------------|
=================================================================================================================
```

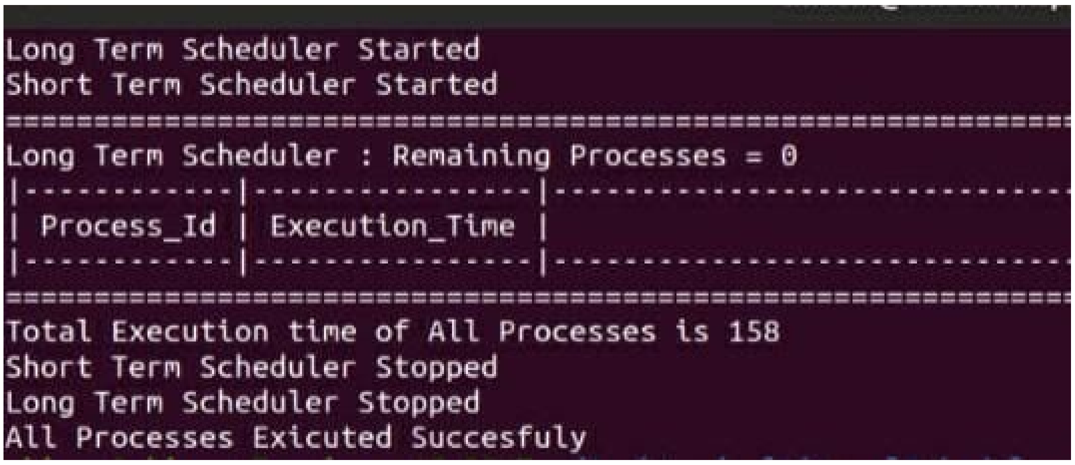
fig 4 : size of Ready queue is reduced to 4 as no more processes remain in Job Pool



```
Long Term Scheduler Started
Short Term Scheduler Started
==================================================================================================
Long Term Scheduler : Remaining Processes = 0
|-----------|--------------|------------------------------------------------------------------|
| Process_Id | Execution_Time |                                                                  |
|-----------|--------------|------------------------------------------------------------------|

Short Term Scheduler :
Ready Queue : size = 1
|-----------|--------------|------------------------------------------------------------------|
| Process_Id | Time_Remaining |                                                                  |
|-----------|--------------|------------------------------------------------------------------|
|     14    |      00      | -> TERMINATED
|-----------|--------------|------------------------------------------------------------------|
==================================================================================================
```

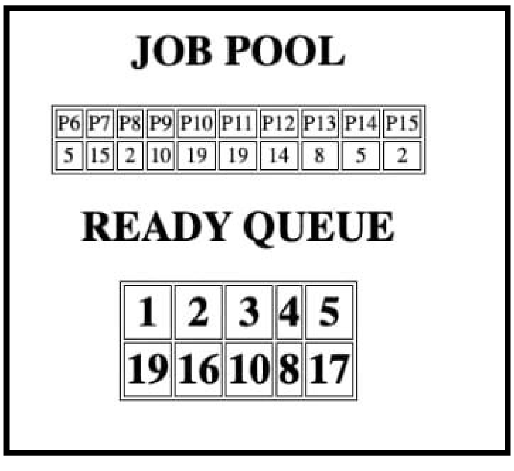
fig 5 : Now the last Process also executed successfully



```
Long Term Scheduler Started
Short Term Scheduler Started
=========================================================
Long Term Scheduler : Remaining Processes = 0
|-----------|--------------|----------------------------
| Process_Id | Execution_Time |
|-----------|--------------|----------------------------
=========================================================
Total Execution time of All Processes is 158
Short Term Scheduler Stopped
Long Term Scheduler Stopped
All Processes Exicuted Succesfuly
```

fig 6 : Long Term and Short Term Schedulers Stopped and all Processes executed successfully

Output on WebPage :



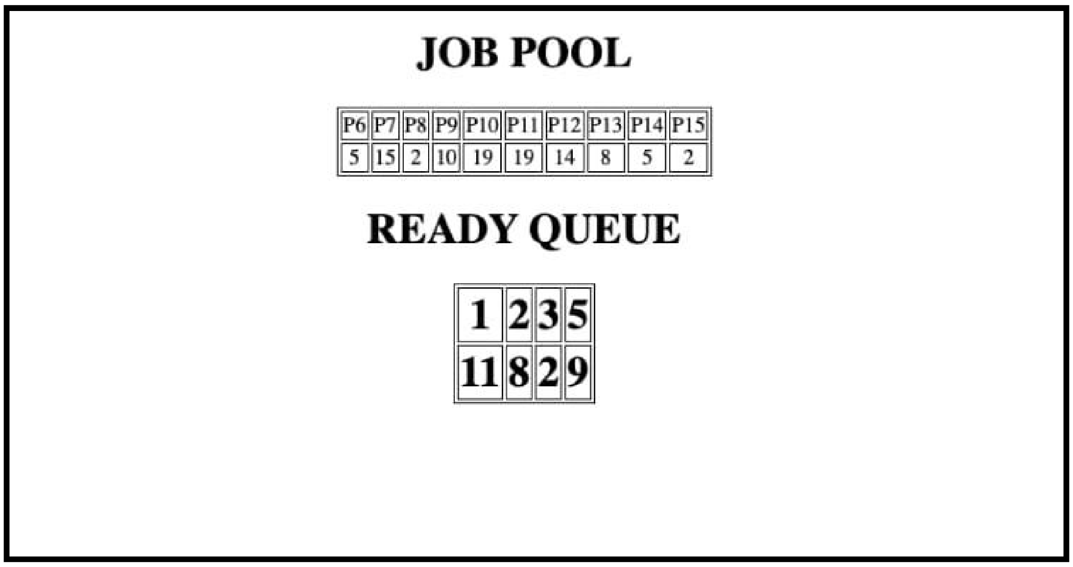
JOB POOL

P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
5	15	2	10	19	19	14	8	5	2

READY QUEUE

1	2	3	4	5
19	16	10	8	17

fig 7 : Schedulers has Started



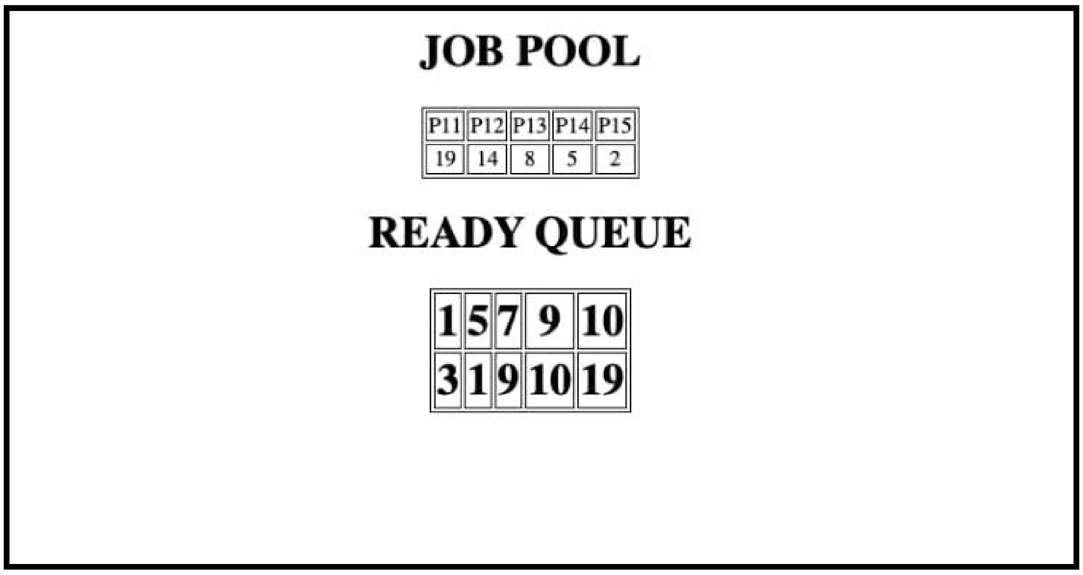
JOB POOL

P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
5	15	2	10	19	19	14	8	5	2

READY QUEUE

1	2	3	5
11	8	2	9

fig 8 : 1 Process executed successfully



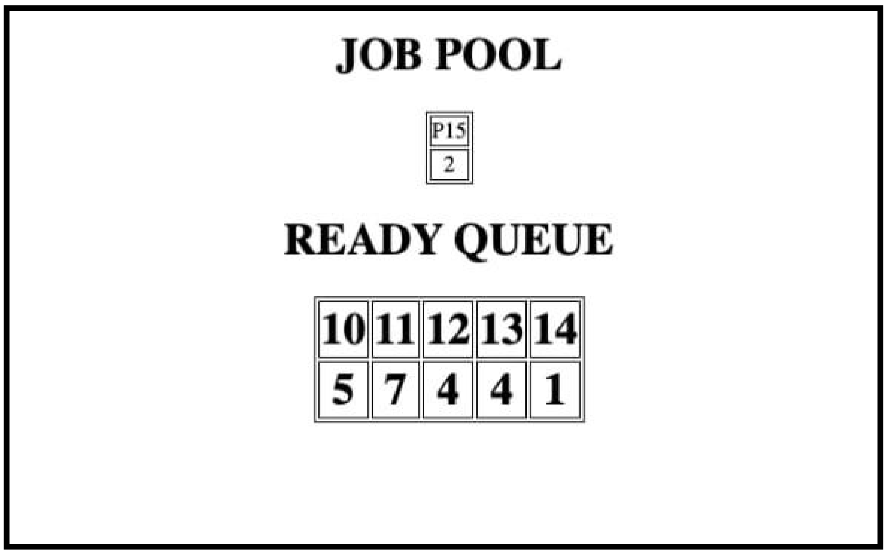
JOB POOL

P11	P12	P13	P14	P15
19	14	8	5	2

READY QUEUE

1	5	7	9	10
3	1	9	10	19

fig 9 : 5 Processes were remaining in Job pool



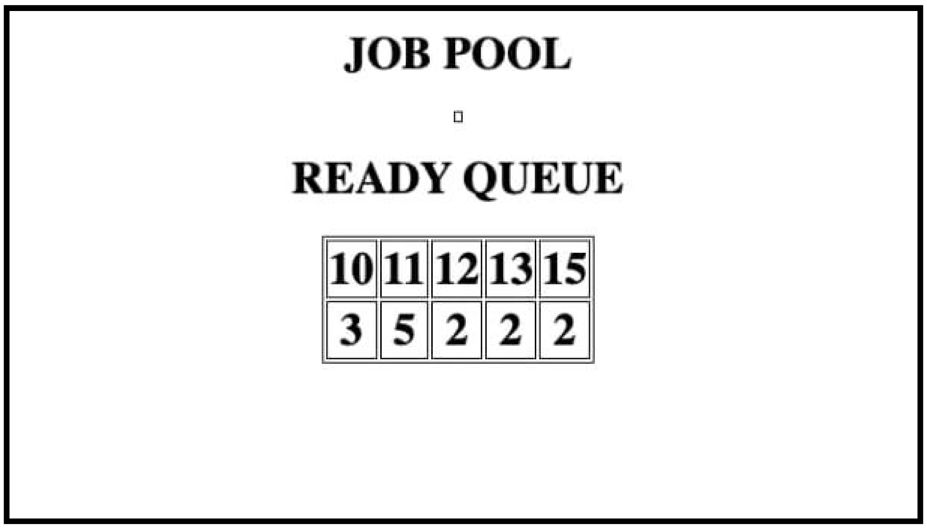
JOB POOL

P15
2

READY QUEUE

10	11	12	13	14
5	7	4	4	1

fig 10 : 1 Process remain in Job Pool

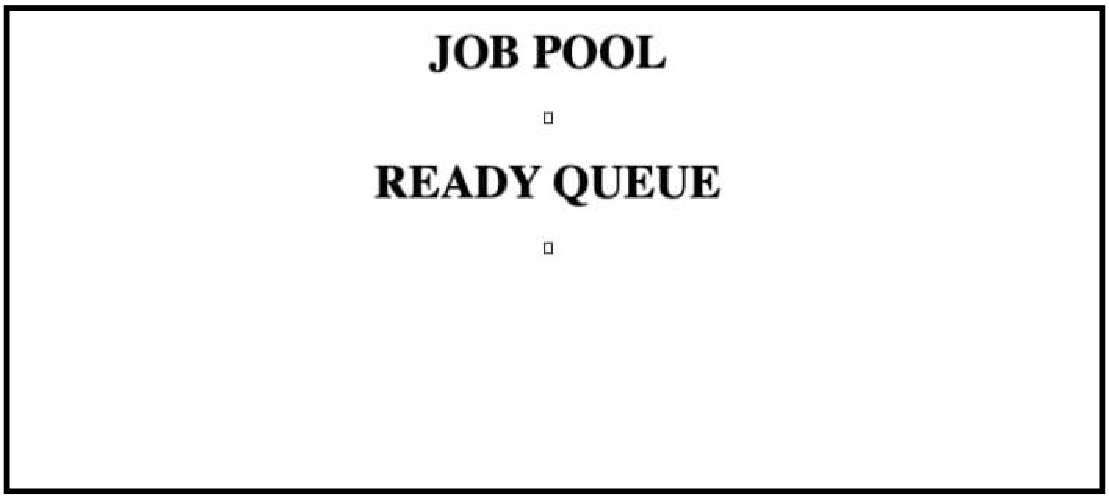

JOB POOL

□

READY QUEUE

10	11	12	13	15
3	5	2	2	2

fig 11 : no more Processes remain in Job Pool


JOB POOL

□

READY QUEUE

□

fig 12 : All Processes were executed successfully

5. CONCLUSION

In this paper, we have studied the Scheduling algorithms & find the challenges encountered during the scheduling of jobs. Scheduling is a major area in the operating system. We determined that during scheduling, the operating system doesn’t consider the types of job it is. It simply applies the algorithm on the basis of execution time, no of processes, etc. We can minimize the average waiting time more effectively if we are known of the type of job, i.e.,communication or computational. location dependency is also one of the major factors deciding the efficiency of scheduling algorithms. The effective and efficient exploitation of grid computing facilities needs highly advanced and protected resource management systems. Efficient resource sharing and accessing cannot go without the assurance of high trustworthiness.

6. REFERENCES

1. https://techdifferences.com/difference-between-long-term-and-short-term-scheduler.html

2. Panda, Sanjaya Kumar.; Dash, Debasis.; and Rout, Jitendra Kumar, A group based time quantum round robin algorithm using Min-Max spread measure. *Inter. J. of Comp. App*. 64(10), 2013.

3. http://en.wikipedia.org/wiki/Scheduling_ (computing).

4. noon, Abbas.; Kalakech, Ali.; & Kadry, Seifedine A new round robin based scheduling algorithm for operating systems: Dynamic quantum using the mean average. *IJCSI Inter. J. Comp. Sci.,* 8(3), 2011.

5. Gaba,Vikas, and Prashar,Anshu, Comparison of processor scheduling algorithms using genetic approach *IJARCSSE*, 2(8), 2012.

6. https://www.geeksforgeeks.org/difference-between-long-term-and-short-term-scheduler/

7. Silberschatz ,Galvin and Gagne. Operating systems concepts, 8th edition, Wiley, 2009.

8. Bawa ,Rajesh Kumar, and Sharma , Gaurav,. Reliable resource selection in grid environment. *Inter. J. Grid Com. App. IJGCA,* 2012, 3(1).

9. Sindhu, M.; Rajkamal, R. & Vigneshwaran ,P.An optimum multilevel CPU scheduling algorithm. IEEE, 2010. 978-0-7695- 058-0/10.

10. S i n g h , P u s h p r a j ; S i n g h , Vi n o d ; a n d Pandey, Anjani. Analysis and comparison of CPU scheduling algorithms. *IJETAE*, 4(1), 2014

11. https://media.geeksforgeeks.org/wp-content/uploads/20190514222912/Untitled-Diagram-124.png