Processes Scheduling Algorithms

Department of Electrical & Computer Engineering
BIRZEIT UNIVERSITY

Dr. Adnan Yahya
Ahmad Yahya (1150)
Hamed Hijazi (1151849)

Abstract

CPU scheduling plays a deep-seated role by switching the CPU among different tasks & processes. As processor is the important resource, CPU scheduling becomes very important in accomplishing the operating system design goals. The intention of the OS should allow the process as many as possible running at all the time in order to make best use of CPU. The high efficient CPU scheduler depends on design of the high quality scheduling algorithms which suits the scheduling goals. This project is a simulator for processes scheduling, simulating the ways in which five different scheduling algorithms behave.

1. Introduction

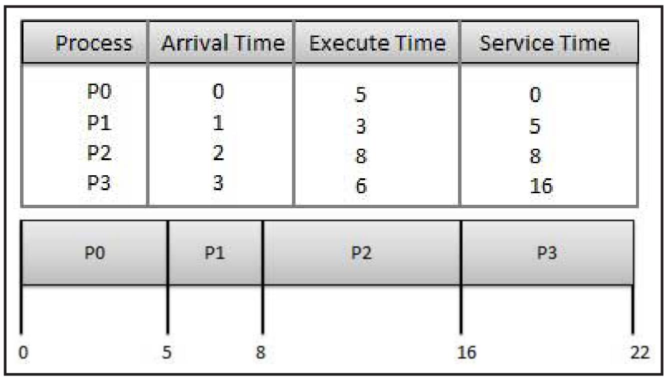
A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. Scheduling usually refers to selecting which process to run next - but can also refer to which input/output (or disk) operation to do next. For a long time, computers could literally do only one thing at a time. The illusion of doing many things was made possible by doing things in rapid rotation, so fast that the human being behind the screen (as it were) perceived everything happening at once. Today, with multi-core machines, the CPU can do multiple things at once - but each of these cores are performing the same fast process switching process, and the collection of cores add considerations to process scheduling.

This project deals with 5 Algorithms for scheduling which are:

- First-Come, First-Served FCFS Scheduling
- Shortest-Job-First SJF Scheduling
- Shortest Remaining Time
- Priority Scheduling
- Round RobinRR Scheduling

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 (the control can't be taken away from that process) until it completes its time, wheres preemptive scheduling is based on priority where a scheduler may preempt (the control can be taken away from that process) a low priority running process anytime when a high priority process enters into a ready queue.

FCFS (non preemptive) Algorithm, which scheduals processes by their arrival time, working as a First In First Out "FIFO" Queue, it's easy to implement and understand, but the bad thing of it is the high waiting time for the processes, where each process needs to wait for the whole current running process to finish, which also increases the total turnaround time.



Process	Arrival Time	Execute Time	Service Time
P0	0	5	0
P1	1	3	5
P2	2	8	8
P3	3	6	16

FCFS implementation

SJF could be preemptive or non-preemptive scheduling algorithm, it's much better than FCFS with lowering the wait time, The processer should know in advance how much time process will take. or otherwise the schedular can't know which process take less time to select.

SRTF is the preemptive version of the SJF 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 and It is often used in batch environments where short jobs need to give preference.

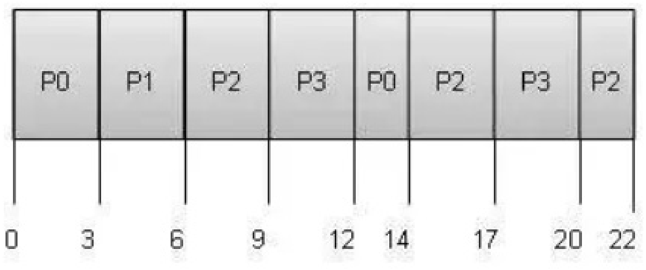
Priority may be preemptive or non-preemptive algorithm and one of the most common scheduling algorithms in batch systems, Each process is assigned a priority. Process with highest priority is to be executed first and so on.

Processes with same priority are executed on first come first served basis.

the priorities (higher to lower) are like:

1. System process(Highest priority)
2. Interactive Processs
3. Batch Process
4. User Process(Lowest priority)

Round Robin is a preemptive process scheduling algorithm, where each process is provided with fixed time to execute, it is called a quantum(Time Quantum), Once a process is executed for a given time period, it is preempted and other process executes for a given time period, and Context switching is used to save states of preempted processes.



2. Design Philosophy

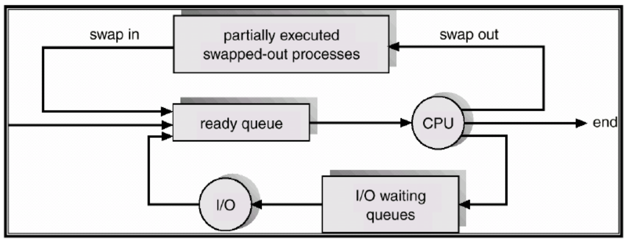
To implement the simulation of these scheduling algorithm, Java Programming Language was used with the help of its User Interface Design, The program has a main loop which is always running, calculating and changing values of processes which are generated randomly for the user.

Processes have properities, which are: arrival time, priority, and Array of bursts (which contains the type of burt "CPU/IO" and its execution time).

these properities are randomly generated from inputs of the user preferences, generating process was random in two different ways, gaussian random generator and binomial random generater, to ensure some specifications of the CPU bursts values and IO bursts values, and to exactly have an average of 75% of Processes CPU bounded, and the rest are IO bounded.

Three main queues were involved in the design, Jobs Queue, Ready Queue, and a list of Devices for Input/Output operations (which were assumed as there are infinite devices for IO).

The Process of scheduling were as in the next Figure, A bunch of Processes come from the Jobs Queue to the Ready queue, taking in consideration the Degree of multiprogramming, which is, how many processes can the CPU handle at the same time.



Path of Processes

Sorting Ready queue for processing depended on the used algorithm, where for FCFS the sort was on the arrival time, for Priority Algorithm the sort was on the priority of each process, and so on.

The main parameters to be calculated are: cpu utilization which is the amount of work handled by the CPU, Throughput which is how many processes can work in a time unit, Turnaround time and CPU waiting time.

4. Theorotical Ovrview

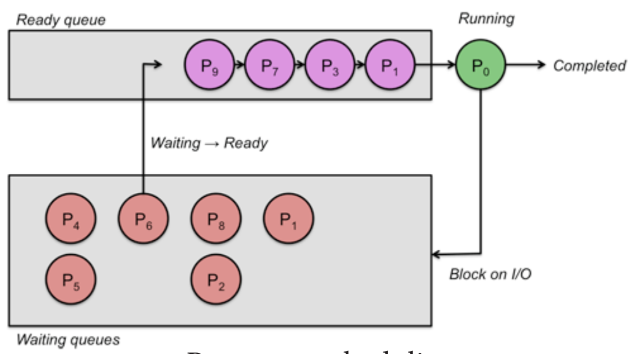
4.1 Choosing a scheduling algorithm

When designing an operating system, a programmer must consider which scheduling algorithm will perform best for the use the system is going to see. There is no universal “best” scheduling algorithm, and many operating systems use extended or combinations of the scheduling algorithms above.

For example, Windows XP/Vista uses a multilevel feedback queue, a combination of fixed-priority preemptive scheduling, round-robin, and first in, first out algorithms. In this system, threads can dynamically increase or decrease in priority depending on if it has been serviced already, or if it has been waiting extensively. Every priority level is represented by its own queue, with round-robin scheduling among the high-priority threads and FIFO among the lower-priority ones. In this sense, response time is short for most threads, and short but critical system threads get completed very quickly. Since threads can only use one time unit of the round-robin in the highest-priority queue, starvation can be a problem for longer high-priority threads.

4.2 OS scheduler implementations

The algorithm used may be as simple as round-robin in which each process is given equal time (for instance 1ms, usually between 1ms and 100ms) in a cycling list. So, process A executes for 1 ms, then process B, then process C, then back to process A.



Processes scheduling

More advanced algorithms take into account process priority, or the importance of the process. This allows some processes to use more time than other processes. The kernel always uses whatever resources it needs to ensure proper functioning of the system, and so can be said to have infinite priority. In symmetric multiprocessing systems, processor affinity is considered to increase overall system performance, even if it may cause a process itself to run more slowly. This generally improves performance by reducing cache thrashing.

4.3 Simulating before real work

Simulation is the imitation of the operation of a real-world process or system. The act of simulating something first requires that a model be developed, this model represents the key characteristics, behaviors and functions of the selected physical or abstract system or process. The model represents the system itself, whereas the simulation represents the operation of the system over time.

Simulation is used in many contexts, such as simulation of technology for performance optimization, safety engineering, testing, training, education, and video games. Often, computer experiments are used to study simulation models. Simulation is also used with scientific modelling of natural systems or human systems to gain insight into their functioning, as in economics. Simulation can be used to show the eventual real effects of alternative conditions and courses of action. Simulation is also used when the real system cannot be engaged, because it may not be accessible, or it may be dangerous or unacceptable to engage, or it is being designed but not yet built, or it may simply not exist.

Key issues in simulation include acquisition of valid source information about the relevant selection of key characteristics and behaviours, the use of simplifying approximations and assumptions within the simulation, and fidelity and validity of the simulation outcomes. Procedures and protocols for model verification and validation are an ongoing field of academic study, refinement, research and development in simulations technology or practice, particularly in the field of computer simulation.

The simulation of Scheduling Algorithms in this Project comes in hand when wanting to implement an algorithm in some new or edited operating system, where the statistics and outcomes of the simulator tells which algorithm to use and when.

comparing those algorithms is hard using a real system, where processes my not always be as we want to, simulators help in building virtual data and processes as needed from users to simulate and compare each algorithm with other algorithms, to test and use it in the real-world systems as a final imlementation.

5. Conclusion & Summary

There is nothing such as pure-perfect scheduling algorithm, each algorithm has its pros and cons, using them wisley is the main key, CPU scheduling will always be the main topic of creating a new Operating System or editing old ones, to make the best use of CPU it may requier to combine algorithms togather to minimize the bad outcomes of each one, and here where simulators come in handy and help to reach the best use and best way of scheduling Processes for the CPU and having the best of each component of a computer.

6. Refrences

[1] Operating Systems Concepts (9th edition), Greg Gagne.
[2] Operating systems lecture notes, Birzeit university.
[3] Informit.com/articles/article.aspx?p=101760
[4]cs.rutgers.edu/~pxk/416/notes/07-scheduling.html
[5] cs.bgu.ac.il/~os142/wiki.files/Scheduling.pdf