

Research Report

by Naomi Jones

FILE	2872_NAOMI_JONES_RESEARCH_REPORT_95675_960753067.ODT (21.38K)		
TIME SUBMITTED	25-JAN-2018 10:40PM (UTC+0000)	WORD COUNT	1201
SUBMISSION ID	82392992	CHARACTER COUNT	7426

Research Report

Naomi Jones

p151933

HND Software Engineering

Table of Contents

1.0 Introduction	3
2.0 Processes	3
2.1 Scheduling algorithms	4
2.1.1 First Come First Serve	4
2.1.2 Round Robin	4
2.1.3 Shortest Job First	4
2.2 Inter-Process Communication	5
3.0 Threads	6
4.0 Summary	7
5.0 References	8

1.0 Introduction

Multi programming systems give the illusion of executing programs at the same time. This is done by interleaving the execution of the programs. So part of one program is executed, then another etc. This allows a single CPU system to provide the illusion of being able to run multiple programs at the same time, and minimises the amount of processing time wasted.[1]

The process by which instructions are executed on a computer is known as the instruction cycle or fetch-execute cycle. During this cycle, the address of an instruction is fetched from a place of memory, usually the program counter. The computer then executes the instruction stored at that address, then fetches the next instruction.[2, p. 14][3]

This report will first discuss processes and what they are. There will be a look into the importance of process scheduling and several of the algorithms that are used. Next, there will be a look at how processes can communicate with each other and the importance of synchronisation. The next section will look at threads.

2.0 Processes

A process is the name given to a program that is currently being executed.[2, p. 750], [4]

Processes are usually stored in a data structure known as the Process Control Block. A process control block contains information about a process, which can be broken down into three main sections. These sections are as follows:

- Process identification: Allows the operating system to identify the process.
- Processor state information: Keeps track of the state of the registers, so if the process is interrupted, the information can be restored when it continues processing.
- Process control information: Contains the state of the process and further information that determines when it should be scheduled.[2, pp. 128–130]

A process can be in one of several states. The main states it can be in are Running, where it is currently being processed. Ready, where the process is ready to go as soon as the CPU is free. The final state is Waiting, where it has to wait for something else to be processed before becoming ready.[2, p. 110], [5]

With multiprocessing, scheduling is necessary in order to maintain using the CPU's time as efficiently as possible. There are three main types of scheduling considerations that are made. Long-term, medium term and short term.

Long-term scheduling involves the moving of programs into the system so they are ready for processing. First it looks to see if the operating system has enough resources to take on any more processes. If it can take on more, then it will decide which program(s) to take and turn into processes. The more processes that are there, the less time each process can be executed. This means the long term scheduler is in charge of limiting how much or little multiprogramming can occur overall.

Medium-term scheduling deals in part with the swapping of processes in and out of main memory. These processes are put on disk, until a time where they can be brought back into main memory and then processed.

The short-term scheduler or dispatcher is the most frequently used. This scheduler looks at the processes that are in the “Ready” state and in memory then decides which one will be executed next.[1], [2, pp. 394–398], [6]

2.1 Scheduling algorithms

There are several different types of CPU scheduling algorithms. Three of them will be outlined here.

When one of them is used, there are several considerations that are taken into account. Amongst these considerations are:

- Utilisation/efficiency: Keep the CPU as busy as possible and so minimising time spent waiting.
- Throughput: Maximise the number of jobs that are processed over a certain time period.
- Fairness: Do all the processes get a fair share of processing time?

2.1.1 First Come First Serve

First come first serve scheduling executes processes in the order that they are scheduled. One of the problems with this type of scheduling is it can become inefficient if one large process is being executed, it can cause a significant delay in smaller processes that are later in the queue.

2.1.2 Round Robin

CPU time is split into slices known as time slices or quantum.. Each process uses the CPU for its time slice or until it is completed. If its time is up and it has not finished executing, it is interrupted and put back in the ready queue. The next process is then selected. Since each process is given the same length time slice, it is considered a fair algorithm. One of the important considerations of this algorithm is how long the slice of time is. If it is too short a period of time, efficiency can be reduced due to having to frequently swap out processes. However, if it is too long, the responsiveness i.e. how long it takes the system to react can become too high.

2.1.3 Shortest Job First

This algorithm looks at the expected processing time and selects the shortest one from the ready queue. This allows for a lot of small processes to be executed, however it can mean that long processes may not get the CPU time needed.

[1], [2, pp. 403–410], [3], [6]

2.2 Inter-Process Communication

Sometimes the need arises for data to be shared between processes. The mechanisms through which an operating system provides this is called Inter-Process Communication (IPC).[7]

3.0 Threads

A thread is a lightweight process that shares memory with other threads.[8] Threads have many similarities with processes, but use less resources when created. Threads exist within processes, so every process has at least one thread, however there can also be multiple threads within a process. [9] This is also how they require less resources than processes, since a process will have its own execution space, whereas a thread shares memory and execution space with other threads.

4.0 Summary

5.0 References

- [1] N. Goel and R. B. Garg, 'A Comparative Study of CPU Scheduling Algorithms', *Int. J. Graph. Image Process.*, vol. 2, no. 4, pp. 245–251, 2012.
- [2] W. Stallings, *Operating Systems*, 4th ed. Alan Apt, 2001.
- [3] H. Arora, D. Arora, and B. Goel, 'An Improved CPU Scheduling Algorithm', *Int. J. Appl. Inf. Syst.*, vol. 6, no. 6, 2013.
- [4] 'Definition of Multiprogramming Operating System'. [Online]. Available: <http://ecomputernotes.com/fundamental/disk-operating-system/multiprogramming-operating-system>. [Accessed: 22-Jan-2018].
- [5] S. Suranauwarat, 'A Visual and Interactive Learning Tool for CPU Scheduling Algorithms', *Int. J. Comput. Sci.*, vol. 10, no. 2, pp. 509–518, 2013.
- [6] I. Qureshi, 'CPU Scheduling Algorithms: A Survey', *Int. J. Adv. Netw. Appl.*, vol. 5, no. 4, pp. 1968–1973, 2014.
- [7] Z. Spasov Ana Madevska Bogdanova, 'INTER-PROCESS COMMUNICATION, ANALYSIS, GUIDELINES AND ITS IMPACT ON COMPUTER SECURITY', 2010.
- [8] J. Yang, H. Cui, J. Wu, Y. Tang, and G. Hu, 'Making parallel programs reliable with stable multithreading', *Commun. ACM*, vol. 57, no. 3, pp. 58–69, Mar. 2014.
- [9] 'Processes and Threads (The Java™ Tutorials > Essential Classes > Concurrency)'. [Online]. Available: <https://docs.oracle.com/javase/tutorial/essential/concurrency/procthread.html>. [Accessed: 15-Jan-2018].

Research Report

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

31 /100

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

GRADING FORM: SMU GENERIC_GJ

NAOMI JONES

31

RESEARCH-INFORMED LITERATURE



Your source list is very good - the majority are academically credible, peer-reviewed sources, used and cited correctly. This work, however, is clearly unfinished - and this, really, is the only cause of this work not receiving what would otherwise have been an extremely good mark. This is a real shame - your

15

work is great, but simply unfinished.

KNOWLEDGE AND UNDERSTANDING



Some good knowledge and understanding displayed, but again, this is severely limited by the work being incomplete.

5

ANALYSIS



Some evidence of analysis, with sources appropriately synthesised. Again, you clearly understood what was required, but did not sufficiently complete it. We'll have a chat about this at some point next week, to see what can be done to ensure that whatever limited you in this work can be mitigated in future.

3

PRACTICAL APPLICATION AND DEP.



N/A - no marks assigned to this section.

SKILLS FOR PROFESSIONAL PRAC.



You write really very well - a good written style, and a nicely-presented document. You might consider a few figures to explain abstract concepts where appropriate.

8