International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 18 (2018) pp. 13512-13516

Algorithms Comparison of Planning Processes, Through Proposed Metric

Vilma Gisela Garcia Garcia[1], Danilo Alfonso López Sarmiento[2,*], Nancy Yaneth Gelvez García [3]

[1] Social Science Graduate, Specialization in IT Management Universidad Libre, Cúcuta, Colombia.
[2,3] Faculty of Engineering, Full time professor at Universidad Distrital Francisco José de Caldas, Bogotá, Colombia.

*(*Corresponding Author)*

Abstract

The following article intends to perform a qualitative and quantitative study of the planning algorithms starting from a simulation, analyzing the resulting metrics. As a result of each simulation a respective metric is shown that measures the efficiency of each algorithm in order to make a general comparison regarding the performance of each studied algorithm.

Keywords: Appropriative, Planning, Round Robin, SRTF, FIFO

INTRODUCTION

Computers comprise a physical part, which are the interconnection components, and a functional part which is the interaction with each other through communication rules. The functional structure studies the organization and operation architectures of components, and determines the system`s functional behavior; the physical structure of the system is made up of the components as such that the computer has. A basic computer model was proposed by John Von Neumann made up of three main parts: input devices, processes central unit (arithmetic unit, control unit, main memory) and output peripherals.

The function of the control unit was to read the machine instructions stored in the main memory one by one, generating control signals for all machines to work and execute the instructions. The function of the arithmetic logic unit was to make a series of elementary operations and the main memory which comprises a set of cells of equal size on which reading and writing operations could be made [1].

Operating systems are the control software of a computer, coordinating and manipulating the system hardware; organize memory files and control any errors that may be generated. There are many operating systems that vary according to the devices and their features change involving various configurations. Some very famous OSs belong to the Windows and Linux brands which are the market leaders.

Operating systems can be classified in many ways, such as:

- **General purpose**. They are characterized by having a large number of users working with a diverse range of applications.
- **Specific purpose**. They are used in environments where a large number of events mostly external to the computer are accepted and processed in a short time. For example: industrial control, flight control, switched telephone equipment or real-time simulations.

Classification of systems by number of users:

- **Single-User**. Support a single user at a time.
- M**ultiuser**. Serve multiple users simultaneously.

By number of tasks:

- S**ingle-task**. User is allowed to perform only one task at a time. Multitask. User is allowed to perform multiple tasks simultaneously.
- Classification of operating systems by the way they offer services.
- **Centralized**. In this model a central computer handles all processing and users connect to it through terminals without memory and processor.
- **Distributed**. Enables distribution of jobs, tasks or processes by a set of processors, which may be on the same computer or on different computers.
- **Desktop**. It is commonly used for desktops, workstations and laptops.
- **Online.** They maintain unity between two or more computers through some communication media to share resources and system information.
- Classification of operating systems according to the response time:
- **Real-time.** Enable the computer to give an immediate response after launching a process.
- **Shared time**. Allow multiple individual users to interact simultaneously, so that each user feels that he is being served exclusively, although, in reality, each task has a priority level and is run in sequential order. **[**2]

There are also mobile operating systems comprised of: user interface and native applications, interface applications, libraries, kernel.

The main functions of operating systems are the processing time allocation policies, control and handling of input and output devices, control of resources, manage and maintain the file systems, manage and maintain users and groups, enable interaction between machine and user with a graphical interface, detect errors. The operating system has full access to all hardware and can execute any instruction that the machine

is capable of running; the remaining software runs in user mode which is a restricted mode.

Currently operating systems allow simultaneous execution of several programs where the concept of priority is used, a hierarchy that is assigned to processes for their administration at the time of execution.

THEORETICAL FRAMEWORK

The process scheduler uses an algorithm known as scheduling algorithm responsible for deciding what process in ready state will run first, if only one CPU is available.

Personal computers have two main features; one is that in most cases there is only one active process therefore the planner does not have to make an effort to determine what process to run; and second, thanks to the current speed of CPUs resources rarely become scarce. Scheduling is very important with network servers because the process to address first must be chosen.

CPU efficiency is relevant because

1. User mode must be changed to kernel mode.
2. State of the current process must be saved, including storing their records in the process table so that they can be reloaded later.
3. The memory map must be saved
4. A new process must be selected by running the scheduling algorithm.
5. The new process memory map must be reloaded in the MMU.

The new process must be started:

1. When a process is created and a choice must be made as to which should run first, the father or the son.
2. Making a scheduling decision when a process ends, and picking another that is in ready state and if there is none, choose one that is in idle state.
3. When a process is blocked for some reason, another process must be chosen to be executed.
4. When a resource is interrupted the process must be reset to ready state, and this is where the planner should decide whether to serve it immediately or wait for its corresponding turn.

Operations of processes can be:

- Creation
- Destruction
- Suspension
- Resumption
- Ownership Change
- Wake up

Scheduling algorithms can be divided into two categories.

Non appropriative: selects a process to run and then allows it to run until it crashes or until it releases the CPU voluntarily (it will not forcibly suspended).

Appropriative: selects a process and allows it to run for a pre-set time. "If it is still running at the end of the set time interval, it is suspended and the planner selects another process to execute (if there is one available). To carry out appropriative planning, it is necessary that the clock be interrupted at the end of the set time interval for the CPU to return control to the planner. If a clock is not available, non-appropriative planning is the only option."[3].

Scheduling algorithm goals, (all systems).

- Equity - Give each process a fair share of the CPU.
- Policy implementation: Verify that established policies are carried out
- Balance: Keeping all parts of the system busy

Performance "is the number of jobs per hour that the system completes" [3].

Processes are assigned times during their life cycle and are the following:

- Waiting time: is the time at which a process remains in ready.
- End time: time that it lasted in ready plus the execution time.
- Penalty: is the response time equal to time of completion on CPU time.

Processes priorities can be:

- Assigned by the operating system
- Assigned by the owner
- Static (cannot be modified)
- Dynamic (can be modified)

Possible failures are related to the process during its execution by the system:

- Catastrophic: preclude operation and are not recoverable by the system.
- Unrecoverable: do not affect the system.
- Recoverable: The process can be solved.

PROCEDURE FOR ALGORITHMS SIMULATION

For simulation processes have the following characteristics, as shown in Figures 1 and 2:

- Id (identifier)
- Name
- Runtime (Rt)
- Quantum (Round Robin)
- Lifetime (Fed Back Multiple Queues)
- Priority

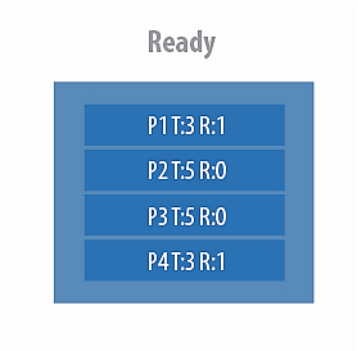

Figure 1. Processes in the ready queue simulator and their Features

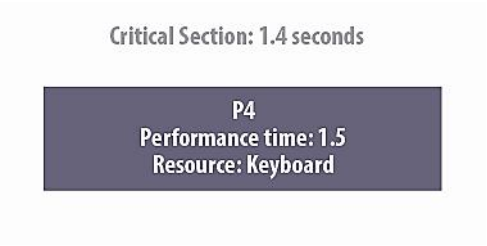

Figure 2. Critical Section

For the various algorithms, a graphic basis of each process statements is shown in a Gantt chart, where each color represents a possible state as shown in Figure 3.

Figure 3. Gantt Chart

The functionality of the Round Robin algorithm "Each process is assigned a time interval, known as quantum, during which it is allowed to run.

If the process is still running at the end of the quantum, the CPU It is suitable to be assigned to another process. If the process crashes or terminates before the quantum expires, the CPU switching is done when the process crashes of course. It's easy to implement the circular shift algorithm [3].

The main feature of this algorithm is the quantum function which should be calculated so that it is optimal for the CPU, it cannot be a very long or very short time; if the quantum is too short too many commutations processes occur and reduce CPU efficiency, but if it is too long, a poor response to short interactive requests can occur.

Default priority planning makes all processes equally important; each process is assigned a priority, and the runnable process with the lowest priority (most important) is the one that can be run.

METRICS ANALYSIS

For metrics analysis, both types of algorithms were considered as for each type a different analysis is performed (with and without priority), which are determined by equation 1,

$$\%Efficiency = \frac{Average\ Run\ Time}{Average\ Total\ Time} \cdot 100$$

Equation 1. Processes without priority

Where the percentage of efficiency was considered because it gives us a rough idea of the performance of each processor, and thus can determine the most efficient.

Round Robin

For the Round Robin simulation the following results were obtained considering the previous equation and performing

tests on processes with relatively long run times (30 sec. Approx.); simulation results are shown below in Figure 4.

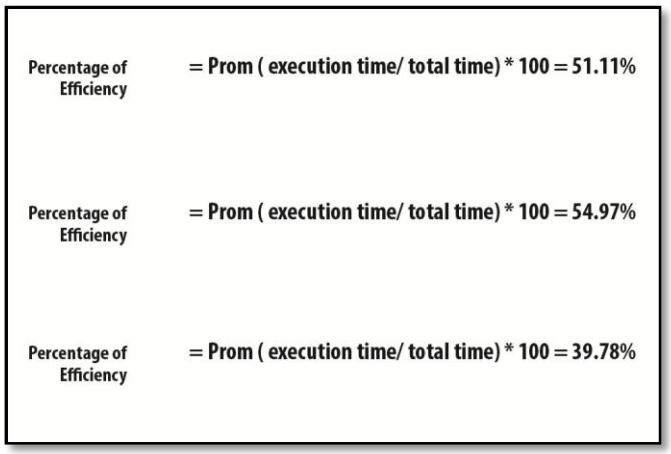

Figure 4. Efficiency percentages of each CPU in Round Robin

As can be seen percentages show low performance levels, and even more so for processes with high run times.

SJF (Appropriative and non-appropriative)

For the SJF algorithm we took into account the percentage of efficiency general formula, reaching similar results regarding the Round Robin algorithm; these results are shown in Figure 5.

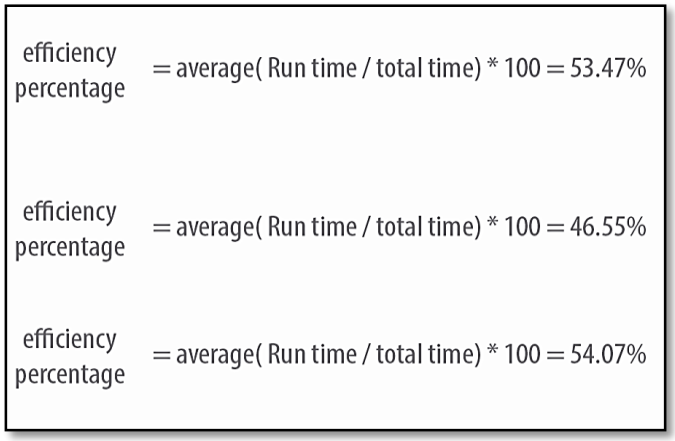

Figure 5. Percentages of the SJF Algorithm

Even with high process execution times, the algorithm shows percentages of low efficiency very close to the previous.

SRTF (Appropriative and Non Appropriative)

For the SRTF analysis we took into account that the processes with priority, i.e. where the percentage of efficiency formula is amended according to priority, which means that the average number of processes is made by priority as follows (equation 2):

$$\%Efficiency = \frac{(Average\ Run\ Time\ Priority\ (n)}{(Average\ Total\ Time\ Priority\ (n)} \cdot 100$$

Equation 2. Processes with Priority

For this case the following results were obtained with high runtime processes (figure 6):

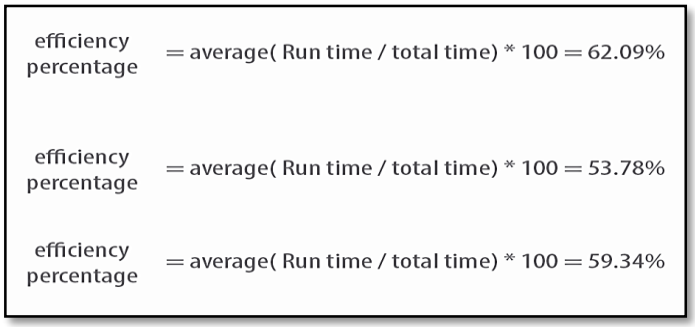

Figure 6. SRTF Efficiency percentages

Colas Retroalimentadas

Para este algoritmo se usó la ecuación 2, el número de procesos utilizados fueron pocos procesos en la cola de Round Robin y así poder evidenciar la eficiencia y el rendimiento con las demás colas (SRTF, y FIFO). En este caso también se usaron procesos con altos tiempos de ejecución obteniendo los siguientes que se observan en la figura 7.

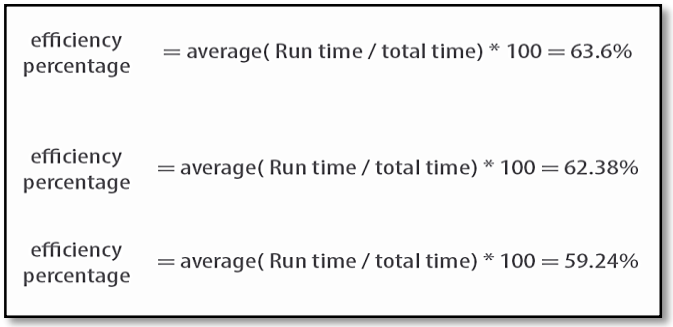

Figure 7. Efficiency Percentages

Fed Back Multiple Queues

Equation 2, was used for this algorithm, performing tests with few processes in the Round Robin queue to demonstrate the change of processes and an expected lifetime as a parameter for its execution time, making it "optimal", the results are those shown in Figure 8.

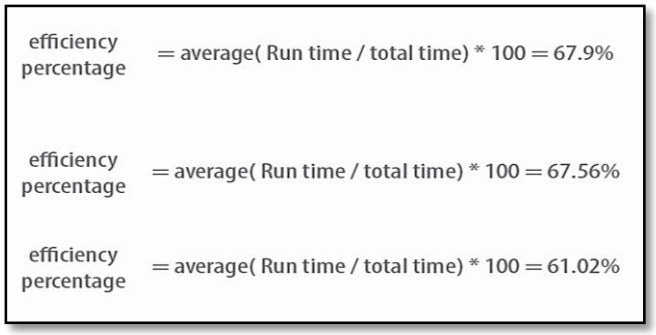

Figure 8. Fed Back Multiple Queues Efficiency Percentages

High execution times were also assigned in the FIFO and SRTF queue in order not to saturate the Round Robin queue.

CONCLUSION

The various algorithms offer advantages depending on the processes that are running and each has advantages and disadvantages depending on the type of process and handling of each processor.

According to results obtained in the metrics analysis, the algorithm with the highest efficiency percentage is the Fed Back Multiple Queues algorithm, having relatively acceptable percentages and being the non-appropriative algorithms the most efficient.

According to the literature, there are algorithms that can improve efficiency and it also depends on the type of processing (parallel); new technologies are intended to minimize the time that they last in the critical section and minimize the chances that any of these processes end up crashing.

According to Tanemboum, "In batch processing systems there are no users waiting impatiently in their terminals for a rapid response to a short request". Consequently, the non-appropriative algorithms are acceptable (or appropriative algorithms with long periods for each process).

This method reduces process switching and therefore improves performance. In fact, batch processing algorithms are quite general and can often be applied to other situations also, which makes them worth while studying even for people who are not involved in computing with corporate mainframes.

In an environment with interactive users, ownership is essential to prevent a process from monopolizing the CPU and deny service to others. Even if there were a process that ran indefinitely and intentionally, there could be a process that would disable the others indefinitely due to an error in the program. Appropriation is needed to prevent this behavior, "this became evident in various simulations.

REFERENCES

[1] Moreno Pérez, Juan Carlos, And Ramos Pérez, Arturo Francisco. Sistemas Operativos Y Aplicaciones Informáticas. España: RA-MA Editorial, 2014. Proquest Ebrary. Web.21 De Noviembre 2015. Pp 57-70

[2] Muñoz López, Francisco Javier. Sistemas Operativos Monopuesto. España: Mcgraw-Hill España, 2013. Proquest Ebrary. Web. 21 Noviembre 2015.

[3] Tanenboum, Andrew S. "Sistemas Operativos Modernos" 3 Ed. Pearson. Prentice Hall Pag 148, 157, 161

[4] Víctor Manuel Alonso Domínguez, José Antonio Villarino Fernández." Sistemas Operativos: Para Micros, Minis Y Mainframes, Con Ejercicios Prácticos De MS-DOS, UNIX Y OS/MVS /"Ed Madrid: Alhambra Longman, 1994.

[5] Barron, David William. "Sistemas Operativos. Editorial:México : Bogotá : Mcgraw-Hill, 1986

[6] Jesús Carretero Pérez. "Sistemas Operativos: Una Visión Aplicada". Madrid: Mcgraw Hill, 2001.

[7] Enrique Monsalve. "Sistemas Operativos Y Software De Base".

[8] Alcalde Lancharro, Eduardo. "Introducción A Los Sistemas Operativos: (MS/DOS, UNIX, OS/2, VMS, OS/400)" ED Madrid, Bogotá: Mcgraw-Hill, 1992.

[9] Mclever Mchoes, Ann. "Sistemas Operativos". Ed 6a. Instituto Politecnico Nacional. Mexico DF 2011

[10] Catalinas, Enrique Quedo. "Sistemas Operativos Y Lenguajes De Programación". Madrid 2003