International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 17 (2018) pp. 13043-13045

Analysis of Effectiveness of Planning Algorithms Processes, Through Time Performance

Vilma G. García [1], Nancy Y. Gelvez [2], Danilo A. López [3*]

[1]Social Science Graduate, Specialization in IT Management Universidad Libre, Cúcuta, Colombia (South America)

[2]Department of System Engineering, Universidad Distrital Francisco José de Caldas, Bogotá, Colombia (South America)

[3]Department of Electric Engineering, Universidad Distrital Francisco José de Caldas, Bogotá, Colombia (South America)

**Corresponding Author*

Abstract

In this paper we show parts of an implementation that runs the following planning algorithms: Round robin, Short Remaining Time First (SRTF), Shortest Job First (SJF), preemptive priority, non-preemptive priority, multiple queues and multiple fed back queues; to make an analysis of the metrics of each in order to determine which algorithm is the optimum.

Keywords: Processes, round robin, SRTF, SJF, multiple queues.

INTRODUCTION

Scheduling algorithms fulfill the function of defining the resources that a process uses for execution, highlighting the importance of good management to maximize CPU performance, because otherwise a process could cover all CPU resources, decreasing its effectiveness.

THEORETICAL FRAMEWORK

State of processes

Processes have several models to define their states; in this case we will use the five states model that comprises: [3]

Execution: The process is running.

Ready: The process is in memory, ready to move to running state.

Blocked: The process cannot be executed because it is in conflict with another operation.

Suspended: The process is waiting to move to the ready state.

Terminated: The process has already completed its running state.

Scheduling algorithm

It is an Operating System (OS) module, responsible for controlling the system tasks, managing the process execution state, and deciding at each step, which task should run in each moment. The purpose of planning is to ensure extensive use of the CPU, attempting to achieve equity in runtime access distribution [1].

Types of planning

Appropriative Planning: The process "appropriates" the CPU, this means that the task assigned, remains at runtime until completion.

Non appropriative Planning: In this type of planning, tasks are assigned an execution time, and when it ends, or a given condition occurs, the running process can be "expelled", in order to serve another process in queue [2].

Scheduling Policies

There are several policies to allocate the CPU to system processes.

- First in, first out (FIFO) (preemptive): Based on the guideline that the first process to enter the ready queue is the first to leave.
- Round Robin (Non Preemptive): It is assigned a period of time (Quantum), that the process will occupy the CPU. If the process is not completed, it is sent to the suspended queue, which after a period of time sends the process to the ready queue.
- Shortest process first (SJF) (preemptive): Selects the process with the shortest execution. A variation of this algorithm is its non-preemptive version, which chooses the process with the least running time remaining, expelling one with a greater runtime.
- Multiple queues: The ready queue is divided into several queues, each with a scheduling algorithm; and there is planning between queues, managed as levels. A variation of this algorithm is the queue feedback, which is essentially changing queue, or level, when the process has fulfilled a given waiting time in a given level queue [5].

STATE OF THE ART

In doing a literature review, the planning process was found to include areas such as:

- Quantitative analysis [6].
- Communications networks [7].
- Simulation [8].

- Computer industry [9].
- Benchmarking [10].

In these fields we observe significant presence of related disciplines such as software development, mathematical modeling and analysis of graphs [11].

ANALYSIS OF PLANNING ALGORITHMS

Analysis criteria

- Response time: The time needed to complete a p process outstanding work, including idle time waiting for execution.
- Standby time (lost time): CPU time, how long it is ready and waiting to run.
- Penalty ratio: Fraction of the response time during which p was waiting.
- CPU utilization: Percentage of time the CPU is doing useful work [4].

Simulation

The simulation is done by dividing the analysis into two sections, by the number of ready queues that each processor has:

- One queue: Round Robin, SJF, SRTF, preemptive priority, Non preemptive Priority.
- Several queues: multiple queues, multiple fed back queues.

The following parameters have been defined for all simulations, with the aim of analyzing the performance of scheduling algorithms.

Resources:

- Ram - 128 mb.
- Printer – 1 port.
- Speakers – 1 port.

Processor 1:

- P1 - time=10 - priority=User - resource= ram-64 – queue Priority=1
- P2 - time=20 - priority=System - resource= speakers-1 - queue Priority =1

Processor 2:

- P3 - time=20 - priority=User - resource= speakers-1 - queue Priority =2
- P4 - time=15 - priority=System - resource=printer-1 - queue Priority =2
- P5 - time=5 - priority=E/S - resource= ram-12 - queue Priority =1

- Processor 3:
- P6 –time040 –Priority=System –resource= ram 64 – queue Priority =1
- P7 –time=18 –priority=user –resource=printer-1 – queue Priority =1
- P8 –time=41 –priority=E/S –resource=printer-1 – queue Priority =1

Simulation results

The results of the Round Robin algorithm simulation are shown in Table 1.

Table 1. Round Robin Algorithm Metric.

Metrics									
processor	average response time (T)	Average waiting time (ready row) t	CPU time	CPU time (free)	Number of procceses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	20.5	5.5	2194	1082	2	15	4	0.25	49.31617228805836
2	27.8	13.8	2184	1072	5	14	2.076923076923077	0.48148148148148145	49.08424908424909
3	42.375	21.25	2125	1013	8	21	2	0.5	47.67058823529412
system	30.224999999999996	13.516666666666666	2167.666666666666	1055666666666667	15	16.666666666666668	2.692307692307692	0.4104938271604938	48.690384849449686

The results of the SJF algorithm simulation are shown in Table 2.

Table 2. SJF algorithm metric.

Metrics									
processor	average response time (T)	Average waiting time (ready row) t	CPU time	CPU time (free)	Number of proccesses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	20.5	5.500	105	75	2	15	4.000	0.2500	28.57
2	25.00	11.67	105	65	3	14	2.273	0.4400	38.10
3	62.33	29.33	105	6	3	33	2.138	0.4677	94.29
system	35.94	15.50	105.0	48.67	8	20.67	2.804	0.3859	53.65

The SRTF algorithm simulation results are shown in Table 3.

Table 3. SRTF algorithm metric.

Metrics									
processor	average response time (T)	Average waiting time (ready row) t	CPU time	CPU time (free)	Number of proccesses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	NaN	NaN	5	1	0	NaN	NaN	NaN	80.00
2	23.00	9.667	739	699	3	14	2.556	0.3913	5.413
3	43.50	14.00	739	677	2	29	3.071	0.3256	8.390
system	NaN	NaN	494.3	459.0	5	NaN	NaN	NaN	31.27

The results of the non-preemptive priority algorithm simulation are shown in Table 4.

Table 4. Non-preemptive priority Algorithm Metrics.

Metrics									
processor	average response time (T)	Average waiting time (ready row)	CPU time	CPU time (free)	Number of proccesses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	26.50	11.50	445	415	2	15	2.364	0.4231	6.742
2	24.20	10.20	445	405	5	14	2.400	0.4167	8.989
3	43.00	21.88	445	346	8	22	2.048	0.4884	22.25
system	31.23	14.53	445.0	388.7	15	17.00	2.270	0.4427	12.66

The results of preemptive priority algorithm simulation are shown in table 5.

Table 5. Preemptive priority algorithm metric.

Metrics									
processor	average response time (T)	Average waiting time (ready row) t	CPU time	CPU time (free)	Number of proccesses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	25.50	10.50	797	767	2	15	2.500	0.4000	3.764
2	25.80	11.80	797	757	5	14	2.273	0.4400	5.019
3	44.00	22.88	797	698	8	22	2.000	0.5000	12.42
system	31.77	15.06	797.0	740.7	15	17.00	2.258	0.4467	7.068

The results of the multiple queues algorithm simulation are shown in table 6.

Table 6. Multiple queue algorithm metrics.

Metrics									
processor	average response time (T)	Average waiting time (ready row) t	CPU time	CPU time (free)	Number of proccesses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	21.50	6.500	281	251	2	15	3.500	0.2857	10.68
2	21.67	8.333	281	241	3	13	2.625	0.3810	14.23
3	66.00	33.00	281	182	3	33	2.000	0.5000	35.23
system	36.39	15.94	281.0	224.7	8	20.33	2.708	0.3889	20.05

The results of the Fed Back Multiple Queues simulation algorithm are in Table 7.

Table 7. Fed Back Multiple Queues Algorithm metrics.

Metrics									
processor	average response time (T)	Average waiting time (ready row) t	CPU time	CPU time (free)	Number of proccesses served	Average waiting tme E= T-t	average ratio penalization P = T/t	Reply ratio average R= t/T	CPU use %
1	28.00	13.00	127	97	2	15	2.154	0.4643	23.62
2	23.33	10.00	127	87	3	13	2.300	0.4348	31.50
3	84.33	51.33	127	28	3	33	1.647	0.6071	77.95
system	45.22	24.78	127.0	70.67	8	20.33	2.034	0.5021	44-36

When making the simulation, the generated metrics (Tables 1, 2, 3, 4 and 5) provide information about the behavior of algorithms, regarding CPU usage, the algorithm that least used it, was SRTF, with 31% usage, while the lowest waiting time employed, was 10.50 seconds. As to the average response rate, the algorithm with the highest ratio is Preemptive Priority, with 0.4467.

It can be said that in terms of performance, for a scheduling algorithm with a ready single queue, the best alternative is SRTF, as it has the best processor usage, and better waiting time management in processes.

The metrics generated in multiple queues algorithms (Tables 6 and 7) show that the multiple queues algorithm responds better than the fed back multiple queues, with 20% CPU usage and 115.94 seconds waiting time, and an average response ratio of 0.2889.

CONCLUSIONS

Scheduling algorithms enable optimizing CPU runtime according to the resources being handled at any given time.

Optimizing the system resources depends directly on how the scheduling algorithm organizes tasks to be carried out.

There may be combinations among the methods to schedule processes which are likely to increase performance, in terms of CPU usage.

REFERENCES

[1] Francisco J. Aliaga.Simuladores de planificadores de sistemas en tiempo real. Universidad de córdoba España. 2011.

[2] M. Barrionuevo. El Planificador de procesos a través de un simulador. Universidad de San Luis. Argentina

[3] Miranda Oscar. Kernel de Tiempo Real para Control de procesos. CINVESTAV-IPN, México D.F. 2011.

[4] Wolf Gunnar. Planificación de Procesos. IIEc-UNAM.

[5] Juan Domínguez. Simulador de Algoritmos de planificación de la CPU. Universidad de Cádiz.2010

[6] Barrionuevo Mercedes; Apolloni Rubén; Piccoli, F. El planificador de procesos a través de un simulador. XV Congreso Argentino de Ciencias de la Computación. 2009.

[7] Milenkovic Milan. Sistemas operativos: conceptos y diseños. McGraw-Hill, 1999.

[8] Tanenbaum Andrew S; Guerrero Gabriel. Sistemas operativos distribuidos. México; Prentice Hall, 1996.

[9] Catalinas Enrique Quero. Sistemas operativos y lenguajes de programación. Editorial Paraninfo, 1999.

[10] Vega Miguel A-, et al. SMPCache: Simulador de sistemas de memoria caché en multiprocesadores simétricos. XI Jornadas de Paralelismo, Granada, 2000.

[11] M.J. Bach. The Design of the UNIX Operating System. Prentice Hall, 1986.