International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

Research Article

CAPS: A Tool for Process Scheduling in Distributed Environment

Hifzan Ahmad[Ȧ*] and Neelendra Badal[Ȧ]

[Ȧ]Department of Computer Sci. & Engineering, K.N.I.T. SULTANPUR, U.P., INDIA

Accepted 01 March 2014, Available online 01 April 2014, **Vol.4, No.2 (April 2014)**

Abstract

A multiprogramming operating system allows more than one process to be loaded into the main-memory at a time and allows the loaded process to share the CPU using time-multiplexing. CPU scheduling is the method of determining when processors should be assigned and to which processes. CPU scheduling in a distributed system can be defined as allocating processes to processors so that total execution time will be minimized, utilization of processors and load balancing will be maximized. This paper presents a simulating behavior of CPU scheduling in distributed environment using the design and developed Computing Analyzer and Process Simulator (CAPS) tool. Symmetric multiprocessor scheduling technique has been implemented in the presented CAPS tool where a common ready queue is shared among the processors. The selection of the processes from the ready queue is done through the FCFS scheduling policy. Finally, a comparative study of the CAPS tool with other existing tools on the basis of execution time required to perform simulation has been presented. The comparative analysis shows that the execution time of CAPS tool is less than that of other existing tools for process scheduling.

Keywords: *CPU Scheduling, Distributed System, Multiprocessor Scheduling, Multiprogramming, Simulator.*

1. Introduction

A process is an instance of a computer program in execution. It contains the program code and its current activity (V. Singh *et al*, 2013). CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated the CPU. It can also be defined as the method of determining when processors should be assigned and to which processes (Maria Abur *et al*, 2011). In a distributed system it can be defined as allocating processes to processors so that total execution time will be minimized, utilization of processors will be maximized and load balancing will be maximized (V. Harsora *et al*, 2011). CPU scheduling is the basis of multiprogrammed operating system (Silberschatz. A. *et al*, 2002). It plays an important role in distributed systems in which it enhances overall system performance metrics such as process completion time and processor utilization (Chaptin *et al*, 2003). It will increase speed of the execution of the workload and executed more quickly with having the scheduling algorithm (Chow *et al*, 1997). The basic idea behind distributed process scheduling is to enhance overall system performance metrics (Stallings *et al*, 1998). In a multiprogramming operating system many processes are loaded into the main-memory at a time where they reside in a ready queue. A multiprogramming system also allows the loaded process to share the CPU using time-multiplexing (Maria Abur *et al*, 2011). Therefore, a tool is required which allow the users to simulate the behavior of CPU scheduling in distributed environment. The design and developed CAPS tool allows the user to make a distributed environment by adding multiple resources and then perform simulation in this environment.

The implementation of CAPS tool is done through Microsoft Visual Studio 9.0. There are two mechanisms to perform CPU scheduling in multiprocessor system. In asymmetric multiprocessing, all scheduling decisions, I/O processing, and other system activities handled by a single processor called the master processor. The other processors are involved in the execution of user codes only. In symmetric multiprocessing, all processes are in a common ready queue, or each processor may have its own private queue of ready processes. An attempt to keep processes running on the same processor instead migration of processes form one processor to another is called processor affinity. When it is possible for a process to migrate between processors, it is called soft affinity and when it is not possible for a process to migrate between processors, it is called hard affinity (Silberschatz. A. *et al*, 2002). Symmetric multiprocessing mechanism and hard affinity mechanism has been implemented in the presented CAPS tool.

2. Objectives of Process Scheduling

The main objective of CPU scheduling algorithms is to utilize the resources effectively and efficiently. It can be achieved by making CPU busy as much as possible. The criteria for CPU scheduling are as follows:

*Corresponding author: **Hifzan Ahmad**

- Utilization/ Efficiency: It keeps the CPU busy as much as possible. It must have maximum value.
- Throughput: The number of processes that complete their execution per unit time. It must have maximum value.
- Waiting time: The amount of time spent in ready queue. It must have minimum value.
- Turn-around time: The amount of time from submission to the completion of process. It must have minimum value.
- Response time: The amount of time it takes from when a request was submitted until the first response is produced. It must have minimum value.

3. Related Works

Process Scheduling Simulation, Analysis, and Visualization (PSSAV) is a simulation tool which allows the user to perform simulation on multiple jobs, and calculate their average waiting time and average turnaround time (Firmansyah A., 2010). It supports First-Come/First-Served (FCFS), Shortest Job First (SJF), Round Robin (RR) and Priority scheduling algorithms. It has fixed resources; means the user cannot add resources (processors, memories, files, and devices) as per the requirement of the simulation. It can run on any platform supporting Java runtime system. But, still there is a limitation that it does not calculate waiting time and turnaround time for each process. It does not perform scheduling in distributed environment as well as multiprocessor scheduling. It performs scheduling on a single processor system only.

The Process Simulator is a tool which allows the user to perform simulation with various process scheduling algorithms on a collection of processes and calculate the average execution time of all the (Alek Modi *et al*, 2012). It supports FCFS, SJF and RR scheduling policy. It is developed using VB.Net and can run on any platform supporting .Net framework. This simulator also has some limitations. The arrival time of the processes is fixed in this simulator; the arrival time for all the processes is zero second. It does not calculate waiting time and turnaround time, it calculate only average execution time for all the processes. It is also limited to perform scheduling on a single processer system. It does not perform multiprocessor scheduling and CPU scheduling in distributed environment.

4. Overview of the CAPS Tool

The proposed and developed Computing Analyzer and Process Simulator (CAPS) tool is a dotnet (.Net) application that can be run on any platform supporting a dotnet (.Net) framework. It is a simulator which allows the user to make a distributed environment by adding multiple resources (such as processors, memory modules, files and devices) and then perform simulation on various processes using FCFS scheduling policy. Symmetric multiprocessor scheduling technique has been implemented in the presented CAPS tool, where a common queue of ready processes is shared among all the processors (Silberschatz. A. *et al*, 2002). The selection of processes from this common ready queue is carried out by using First-Come, First-Serve (FCFS) scheduling policy. The hard affinity mechanism has been implemented in the proposed CAPS tool where processes are restricted to migrate among processors. Once a process has been allocated to a CPU, it will complete its execution on that processor only (Silberschatz. A. *et al*, 2002). The load balancing in the CAPS tool is performed by checking the status of the processor before assigning a job. Initially the job is assigned to the free processors only. But, in the case when all the processors are busy in jobs executions then the tool calculate the waiting time for all the processors and assigns the job to the processor which have minimum waiting time. It allocates other resources using the same mechanism as the processor. The arrival time and burst time of the processes and the resource assignment to each process is done by the user manually.

5. Model of the CAPS Tool

In the developed CAPS tool, three models has been implemented which are illustrated in Fig.1, Fig.2 and Fig.3.

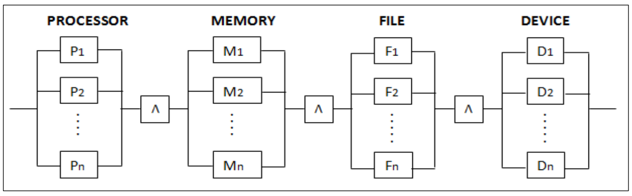

Fig.1: CAPS Model 1

The CAPS Model 1 as shown in Fig.1 explains that the scheduling in the distributed environment performs using the function given below in equation 1 for each job.

$$(P_1 \vee P_2 \vee - - \vee P_n).(M_1 \vee M_2 \vee - - \vee M_n).$$
$$(F_1 \vee F_2 \vee - - \vee F_n).(D_1 \vee D_2 \vee - - \vee D_n) \quad (1)$$

It states that from the available resources in the distributed environment each job gets one processor out of the assigned processors, one memory out of the assigned memories, one file out of the assigned files and one device out of the assigned devices.

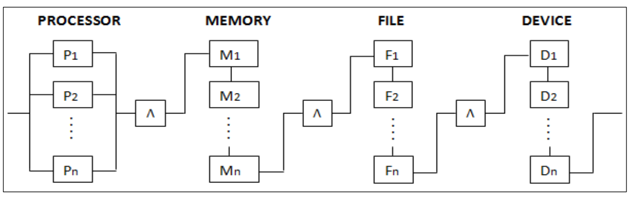

Fig.2: CAPS Model 2

The CAPS Model 2 as shown in Fig.2 explains that the scheduling in the distributed environment performs using the function given below in equation 2 for each job.

$$(P_1 \vee P_2 \vee - - \vee P_n).(M_1 \wedge M_2 \wedge - - \wedge M_n).$$

$$(F_1 \wedge F_2 \wedge - - \wedge F_n).(D_1 \wedge D_2 \wedge - - \wedge D_n) \qquad (2)$$

It states that from the available resources in the distributed environment each job gets one processor out of the assigned processors, all the assigned memories, all the assigned files and all the assigned devices.

The CAPS Model 3 as shown in Fig.3 explains that the scheduling in the distributed environment performs using the function given below in equation 3 for each job.

$$(P_1 \vee P_2 \vee - - \vee P_n).(M_1 \vee M_2 \vee - - \vee M_m).$$
$$(M_{m+1} \wedge M_{m+2} \wedge - - \wedge M_n).\ (F_1 \vee F_2 \vee - - \vee F_m).$$
$$(F_{m+1} \wedge F_{m+2} \wedge - - \wedge F_n).\ (D_1 \vee D_2 \vee - - \vee D_m).$$
$$(D_{m+1} \wedge D_{m+2} \wedge - - \wedge D_n) \qquad (3)$$

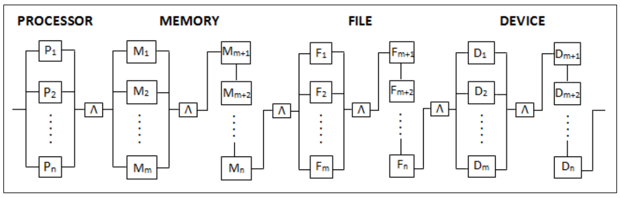

Fig.3: CAPS Model 3

It states that from the available resources in the distributed environment each job gets one processor out of the assigned processors, anyone of the memory from M_1 to M_m and all the memories from M_{m+1} to M_n out of the assigned memories, anyone of the file from F_1 to F_m and all the files from $_{Fm+1}$ to F_n out of the assigned files and anyone of the device from D_1 to D_m and all the devices from D_{m+1} to D_n out of the assigned devices.

6. Implementation of CAPS Tool Using Microsoft Visual Studio 9.0

The implementation of the presented CAPS tool is carried out with Microsoft Visual Studio 9.0. The CAPS tool allows the user to add four types of resources (processors, memories, files and devices) manually to create a distributed environment. The distributed environment created by the user is illustrated in Fig.4.

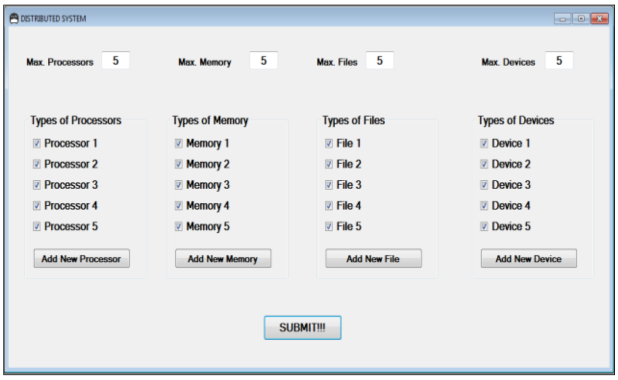

Fig.4: Creating distributed environment

Figure 4 shows the resources added by the user to create a distributed environment. The number of processes required to perform simulation is decided by the user. The arrival time and burst time of the processes and the resource assignment to each process is done by the user manually. The CAPS tool calculates the waiting time and turnaround time for each process and average waiting time and average turnaround time for all the processes. The simulation performed on the processes is illustrated in Fig.5.

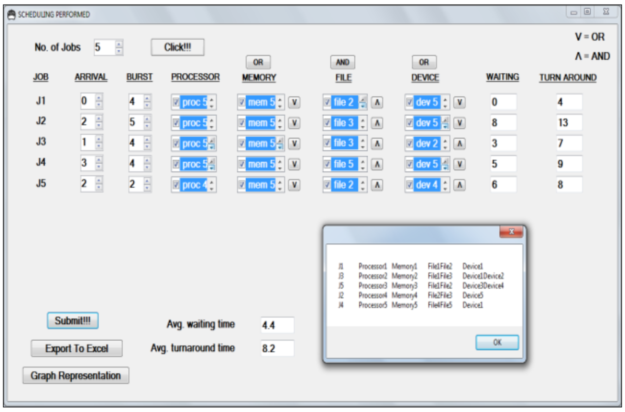

Fig.5: Simulation performed

Figure 5 shows the assignment of all resources to each process and the result determined by the CAPS tool after the simulation performed. The order in which processes are executed is also displayed in Fig.5.

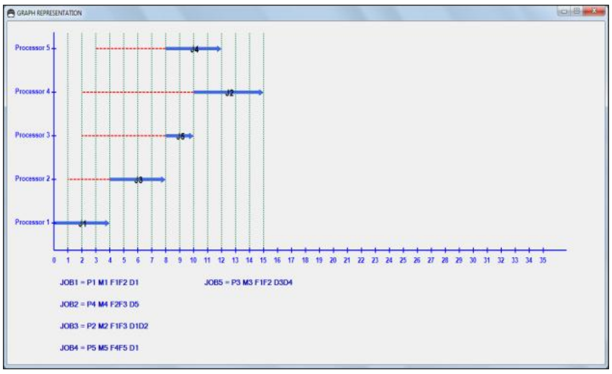

Fig.6: Graphical representation of process execution

It allows the user to log the simulation result into an excel file. It also allows the user to displays the result of the simulation in graphical representation. The graphical representation of the processes execution is illustrated in Fig.6. The dotted lines represent the waiting time of the processes, and the solid line represents the running/execution time of the processes. It also displays the resources assigned to each process.

7. Comparative Studies

The CAPS tool allows the user to make a distributed environment to perform scheduling on multiple jobs. The other tools like PSSAV and Process Simulator do not allow the user to make a distributed environment. The comparative analysis of CAPS tool with PSSAV and Process Simulator has been discussed in Table 1.

Table 1 Comparison of CAPS with Other Tools

Attributes	**Simulators**		
	CAPS	**PSSAV**	**Process Simulator**
Distributed environment	Yes	No	No
Multiprocessor scheduling	Yes	No	No
First-Come/First-Served	Yes	Yes	Yes
Shortest job first	No	Yes	Yes
Round-Robin	No	Yes	Yes
Waiting time	Yes	No	No
Turnaround time	Yes	No	No
Average waiting time	Yes	Yes	No
Average turnaround time	Yes	Yes	No
Export simulation result	Yes	No	No
Graphical representation	Yes	No	No

The comparative analysis of CAPS tool with PSSAV tool and Process Simulator tool on the basis of execution time required to perform simulation is illustrated in Table 2.

Table 2 Comparison of CAPS with other Tools where iterations are fixed

No. of Jobs	No. of Iterations	Execution Time (sec)		
		CAPS	PSSAV	Process Simulator
1	5	0.8	0.8	0.9
2	5	0.9	0.8	0.9
3	5	0.9	0.9	1
4	5	1	1.1	1.2
		Execution Time (sec)		
No. of Jobs	No. of Iterations	CAPS	PSSAV	Process Simulator
5	5	1.2	1.3	1.5
6	5	1.3	1.5	1.7
7	5	1.5	1.8	2
8	5	1.6	2.1	2.3
9	5	1.9	2.3	2.5
10	5	2.2	2.6	2.8

Fig.7: Execution Graph with fixed no. of iterations

It is depicted in Fig. 6 that as the number of jobs increased the execution time for all the simulators are also increased. The average execution time of CAPS tool is less than that of PSSAV and Process Simulator.

8. Conclusions

The CPU scheduling on a single system is much easier as compare to distributed system. The tools are available to perform the CPU scheduling on a single system, but no such a tool is available which allow the user to make a distributed environment and then perform CPU scheduling. The developed CAPS tool allows the user to make a distributed environment and then perform scheduling on number of processes. It calculates the waiting time and turnaround time for each process. It also calculates the AWT and ATT for all the processes. It allows the user to map the execution time and waiting time for each process on the graph. The user can export the simulation result into an excel file. Finally, the comparative analysis shows that the developed CAPS tool takes less execution time for the same number of jobs as compare to other existing tools.

9. Future Scope

The future scopes of the CAPS tool include adding various scheduling policies which will allow the user to make a comparative analysis on various scheduling policy. The number of resources (processors, memory modules, files and devices) can be increased or decreased as per the requirement.

References

V. Singh, T. Gabba (2013), Comparative study of processes scheduling algorithms using simulator, *Int'l Journal of Computing and Business Research (IJCBR)*, vol. 4

Maria Abur, A. Mohammed, S. Danjuma, S. Abdullahi (2011), A critical simulation of CPU scheduling algorithm using exponential distribution, *Int'l Journal of Computer Science Issues (IJCSI)*, vol. 8, Issue 6, No. 2

V. Harsora, A. Shah (2011), A modified genetic algorithm for process scheduling in distributed system, *Int'l Journal of Computer Application*, special issue on Artificial Intelligence Techniques – Novel Approaches & Practical Applications (AIT)

Silberschatz. A., P.B. Galvin, G. Gagne (2002), Operating System Concept, 6th edition

Chaptin, S.J. (2003), Distributed and Multiprocessor Scheduling, University of Minnesota

Chow, R. and T. Johnson (1997), Distributed Operating Systems and Algorithms, Addison-Wesley

Stallings, W. (1998), Operating Systems, 3rd edition, Prentice-Hall

Firmansyah Adiputra (2010), http://code.google.com /p/pssav/ visited on 20-May-13

Alek Modi, Abdulkarim Almayouf, Young Yang (2012), http://magicmadeincode.blogspot.in/2012/06/operating-system-process-scheduling.html visited on 20-May-13