Diyala Journal of Engineering Sciences

ISSN 1999-8716
Printed in Iraq

Vol. 07, No. 01, pp. 16-29, March 2014

CPU SCHEDULING VISUALIZATION

Taqwa Flayyih Hasan
College of Engineering/ Diyala University/ Iraq
E-mail: tota75m@yahoo.com
(Received: 18/9/2012; Accepted: 29/11/2012)

ABSTRACT: - Scheduling is a key concept in computer multitasking and multiprocessing operating system design, and in real-time operating system design. CPU scheduling is the basis of multiprogramming operating systems by switching the CPU among process; the operating system can make the computer more productive, scheduling algorithms are widely used in communications networks and in operating systems to allocate resources to competing tasks. In this paper, visual interfaces for CPU scheduling algorithms were designed by using Visual Basic6 language. They may use to learn users about this algorithms and how they work.

Keywords: Scheduling, Process, CPU.

1- INTRODUCTION

The Central Processing Unit (CPU) is an important component of the computer system; hence it must be utilized efficiently. This can be achieved through what is called CPU scheduling. The CPU scheduling can be defined as the art of determining which processes run on the CPU when there are multiple runnable processes. Also, it is the problem of deciding which computer process in the ready queue (in other words, which particular programs need some processing and are ready and waiting for it) is to be allocated to the CPU for processing. It is a fundamental problem in operating systems (OS) in terms of minimizing the wait for the user when he or she simply wants to execute a particular set of tasks. It is important because it has a big effect on resource utilization and the overall performance of the system [(1)].

The base idea in multiprogramming is that kept CPU always busy. For this we use scheduling. This is the method by which threads, processes or data flows are given access to system resources [(2)]. The assignment of physical processors to processes allows processors to accomplish work. The problem of determining when processors should be assigned and to

which processes is called processor scheduling or CPU scheduling. When more than one process is run able, the operating system must decide which one first. The part of the operating system concerned with this decision is called the scheduler, and algorithm it uses is called the scheduling algorithm [5] [8]. CPU Scheduling algorithms deal with the problem of deciding which process in ready queue should be allocated to CPU [6].

Scheduling: Refers to the way processes are assigned to run on the available CPUs, since there are typically many more processes running than there are available CPUs. This assignment is carried out by Software known as a scheduler and dispatcher. And there are many types of scheduler like short term scheduler, long term scheduler and midterm scheduler. The goal of scheduler is to achieve:

1. High processor utilization
2. High throughput: i.e. number of processes completed per time unit
3. Low response time: i.e. time elapse from the submission of a request to the beginning of the response [1] [5].

2- Some CPU Scheduling Concepts

Bursts

An executing process alternates between two cycles: CPU execution and I/O wait. The amount of time it takes for each CPU execution to be completed is called the CPU burst, and the time the process spends waiting for an I/O request to be fulfilled is the I/O burst. Historical data for the burst times of a process are taken into account when selecting a CPU scheduling algorithm. The point of this is to prevent a process with a very long burst time from taking control of the CPU before a process with a short burst time is able to run [4].

Dispatcher

The dispatcher is responsible for performing context switches. This means that it gives control of the CPU to the process that is to be run next. First, it saves the state of the currently running process. Next, the state of the selected process is loaded into the registers. Finally, the process begins executing at the location specified by the program counter [4].

3. The Scheduling algorithms categories

The Scheduling algorithms can be divided into two categories with respect to how they deal with clock interrupts.

3.1 Preemptive Scheduling

A scheduling discipline is preemptive if, once a process has been given the CPU can take away. The strategy of allowing processes that are logically run able to be temporarily suspended is called Preemptive Scheduling and it is contrast to the "run to completion" method [5].

3.2 Non preemptive Scheduling

A scheduling discipline is non-preemptive if, once a process has been given the CPU, the CPU cannot be taken away from that process. Following are some characteristics of non-preemptive scheduling:

1. In non-preemptive system, short jobs are made to wait by longer jobs but the overall treatment of all processes is fair.
2. In non-preemptive system, response times are more predictable because incoming high priority jobs cannot displace waiting jobs.
3. In non-preemptive scheduling, a scheduler executes jobs in the following two situations.

a. When a process switches from running state to the waiting state.

b. When a process terminates [5].

4. Scheduling Algorithms

Sensible scheduling strategies might be:

• Maximize throughput or CPU utilization

• Minimize average turnaround time, waiting time or response time [7].

CPU Scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated the CPU; the following are some scheduling algorithms we will study [5]:

- FCFS Scheduling.
- Round Robin Scheduling.
- SJF Scheduling.
- SRT Scheduling.
- Priority Scheduling.
- Multilevel Queue Scheduling.

a. First-Come-First-Served (FCFS) Scheduling

Other names of this algorithm are:

- First-In-First-Out (FIFO)
- Run-to-Completion
- Run-Until-Done

Perhaps, First-Come-First-Served algorithm is the simplest scheduling algorithm is the simplest scheduling algorithm. Processes are dispatched according to their arrival time on the ready queue. Being a non-preemptive discipline, once a process has a CPU, it runs to completion. The FCFS scheduling is fair in the formal sense or human sense of fairness but it is unfair in the sense that long jobs make short jobs wait and unimportant jobs make important jobs wait. FCFS is more predictable than most of other schemes since it offers time. FCFS scheme is not useful in scheduling interactive users because it cannot guarantee good response time. The code for FCFS scheduling is simple to write and understand. One of the major drawbacks of this scheme is that the average time is often quite long. The First-Come-First-Served algorithm is rarely used as a master scheme in modern operating systems but it is often embedded within other schemes [3] [5] as shown in figure (1).

b. Round Robin Scheduling

One of the oldest, simplest, fairest and most widely used algorithms is round robin (RR). In the round robin scheduling, processes are dispatched in a FIFO manner but are given a limited amount of CPU time called a time-slice or a quantum. If a process does not complete before its CPU-time expires, the CPU is preempted and given to the next process waiting in a queue. The preempted process is then placed at the back of the ready list. Round Robin Scheduling is preemptive (at the end of time-slice) therefore it is effective in time-sharing environments in which the system needs to guarantee reasonable response times for interactive users. The only interesting issue with round robin scheme is the length of the quantum. Setting the quantum too short causes too many context switches and lower the CPU efficiency. On the other hand, setting the quantum too long may cause poor response time and approximates FCFS. In any event, the average waiting time under round robin scheduling is often quite long [3] [5].

c. Shortest-Job-First (SJF) Scheduling

Other name of this algorithm is Shortest-Process-Next (SPN). Shortest-Job-First (SJF) is a non-preemptive discipline in which waiting job process with the smallest estimated run-time-to-completion is run next. In other words, when CPU is available, it is assigned to the process that has smallest next CPU burst. The SJF scheduling is especially appropriate for batch jobs for which the run times are known in advance. Since the SJF scheduling algorithm gives the minimum average time for a given set of processes, it is probably optimal. The SJF algorithm favors short jobs (or processors) at the expense of longer ones. The obvious problem with SJF scheme is that it requires precise knowledge of how long a job or process

will run, and this information is not usually available. The best SJF algorithm can do is to rely on user estimates of run times. In the production environment where the same jobs run regularly, it may be possible to provide reasonable estimate of run time, based on the past performance of the process. But in the development environment users rarely know how their program will execute. Like FCFS, SJF is non-preemptive therefore, it is not useful in timesharing environment in which reasonable response time must be guaranteed [3] [5].

d. Shortest-Remaining-Time (SRT) Scheduling

The SRT is the preemptive counterpart of SJF and useful in time-sharing environment.

- In SRT scheduling, the process with the smallest estimated run-time to completion is run next, including new arrivals.
- In SJF scheme, once a job begins executing, it run to completion.
- In SJF scheme, a running process may be preempted by a new arrival process with shortest estimated run-time.
- The algorithm SRT has higher overhead than its counterpart SJF.
- The SRT must keep track of the elapsed time of the running process and must handle occasional preemptions.
- In this scheme, arrival of small processes will run almost immediately. However, longer jobs have even longer mean [3].

e. Priority Scheduling

The basic idea is straightforward: each process is assigned a priority, and priority is allowed to run. Equal-Priority processes are scheduled in FCFS order. The shortest-Job-First (SJF) algorithm is a special case of general priority scheduling algorithm. An SJF algorithm is simply a priority algorithm where the priority is the inverse of the (predicted) next CPU burst. That is, the longer the CPU burst, the lower the priority and vice versa. Priority can be defined either internally or externally. Internally defined priorities use some measurable quantities or qualities to compute priority of a process [3].

f. Multilevel Queue Scheduling

A multilevel queue scheduling algorithm partitions the ready queue in several separate queues, in a multilevel queue scheduling processes are permanently assigned to one queues. The processes are permanently assigned to one another, based on some property of the process, such as:

- Memory size
- Process priority
- Process type

Algorithms choose the process from the occupied queue that has the highest priority, and run that process either:

- Preemptive or
- Non-preemptively

Each queue has its own scheduling algorithm [3].

5. Software Construction

Software design of scheduling algorithm represent visualization programmable design for the most known algorithms, it content several selections each one performed in separate form. The obtained results, related to the three types of scheduling are done by varying the arrival time, burst time for the processes, and calculate the waiting time for each process in the ready queue before allocate this process to the CPU. From the results we get the better algorithm is the (Short job first preemptive), therefore this algorithm is the best especially when we want to execute several programs efficiently. The software consists of the following VB- forms, each from has its own particular design.

When the program runs, the first window (main window) appears and contains several selections for three algorithms of CPU scheduling performed by press the commands which are considered as start addresses to algorithms in the next windows, the bottom commands used for exit from program as shown in figure (2). When the command FCFS pressed the window of the first algorithm will appear, it consists of lists, number of commands and number of hidden labels. The list is used to draw the selected processes that chosen by the user and burst time process to execute and calculate waiting time to each process and calculate turnaround time for each process.

Frames are used to put lists inside it and write the names of lists, and to put label in it to calculate waiting time and turnaround time. Use (text) to enter number of process and using label to rename this text .the (combo Box) is used to select the names of processes and burst time for each process store number of name and burst time inside combo to let users select it. Command use in design command (Enter Bottom) to enter number of process and burst time for each process from combo Box to list depends on the number of process entering in text .The Command (compute) to computation wait time and turnaround time and

draw the Chan chart by use number of label hidden .command (Graph) to draw simple figures of process and moving them to CPU by use line instruction inside image. Command (exit) to view message (thank you for use) and end program. Command (back) to return back to the main form, Label used to show the name of algorithm, as shown in figure (3)(4)(5)(6)(7).

When the command SJF in the main window pressed, the window of the second algorithm will appear, it consist of the same proprieties of FCFS window with some addition proprieties related with this algorithm such as combo and list of arrival time for each process. After entering data, the calculation of waiting time and turnaround time will done. Also, when pressing Preemptive or Nonpreemtive commands according to user selection as shown in figures (8), (9).

When command priority pressed in main window, the third algorithm priority will appear and it consist the same proprieties of the previous two algorithms(FCFS ,SJF) with some differences, such as priority combo Box and priority list for each process. After entering data in the same way as the previous algorithms, the computation of the time will done after pressed compute command, as shown in figure (10).

6. Comparison and Conclusions

1- Based on performance, the shortest job first (SJF) algorithm is recommended for the CPU scheduling problems of minimizing either the average waiting time or average turnaround time.

2- The first come first serve (FCFS) algorithm is recommended for the CPU scheduling problems of minimizing either the average CPU utilization or average throughput.

3- In RR algorithm the major problem is the selection of time quantum. Setting the quantum too short causes too many context switches and lower the CPU efficiency; setting the quantum too long may cause poor response time and approximates FCFS. In any event, the average waiting time under round robin scheduling is often quite long.

4- The SRT is the preemptive counterpart of SJF, and it has higher overhead than its counterpart SJF.

5- Multilevel Queue algorithm allow different algorithms to be used for various classes of processes.

7- References

1- Abraham Silberschatz, A. and Galvin, P.B., (1994), “Operating System Concepts “, 4th ed .Addison-Wesley, MA.

2- Abdulla Shaik, (4, July-Aug. 2012), "Shortest Time Quantum Scheduling Algorithm", International Journal of Engineering Research (IJMER), Vol. 2, Issue, pp-1548-1551.

3- E.O. Oyetunji and A. E. Oluleye, (August 29, 2009)"Performance Assessment of Some CPU Scheduling Algorithms", Research Journal of Information Technology 1(1).

4- Kevin Mellott, (February 3, 2003) "CPU Scheduling, With an Emphasis on Algorithms " .

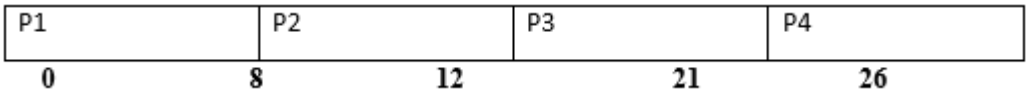
5- Gerard Tel, (2000), "Introduction to Distributed Algorithms", (2nd edition), Cambridge University Press.

6- Gursharan Singh Tatla, (15-Feb-2011), "Scheduling Algorithms", IET Bhaddal, Ropar Punjab, INDIA. www.eazynotes.com/.../scheduling-algorithms.pps

7- Hartley, Stephen .J, (March 30, 1995), "Operating System Programming ".Publisher: Oxford University Press, USA.

8- Steven Hand, (2010) "Operating Systems", Michaelmas Term, 12 lectures for CST IA. www.cl.cam.ac.uk/teaching/1011/.../os1a-slides.pdf

Process	***burst time***
P1	***8***
P2	***4***
P3	***9***
P4	***5***



Average waiting time= (0-0) + (8-0) + (12-0) + (21-0)/4=41/4=10.25

Figure (1): Example data of FCFS algorithm.

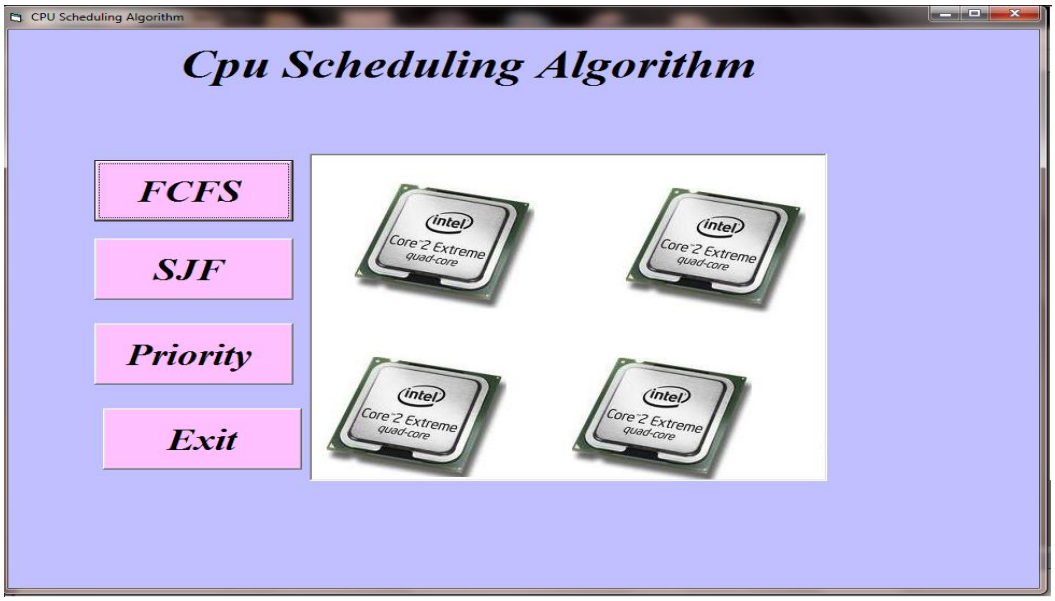


Figure (2): main window.

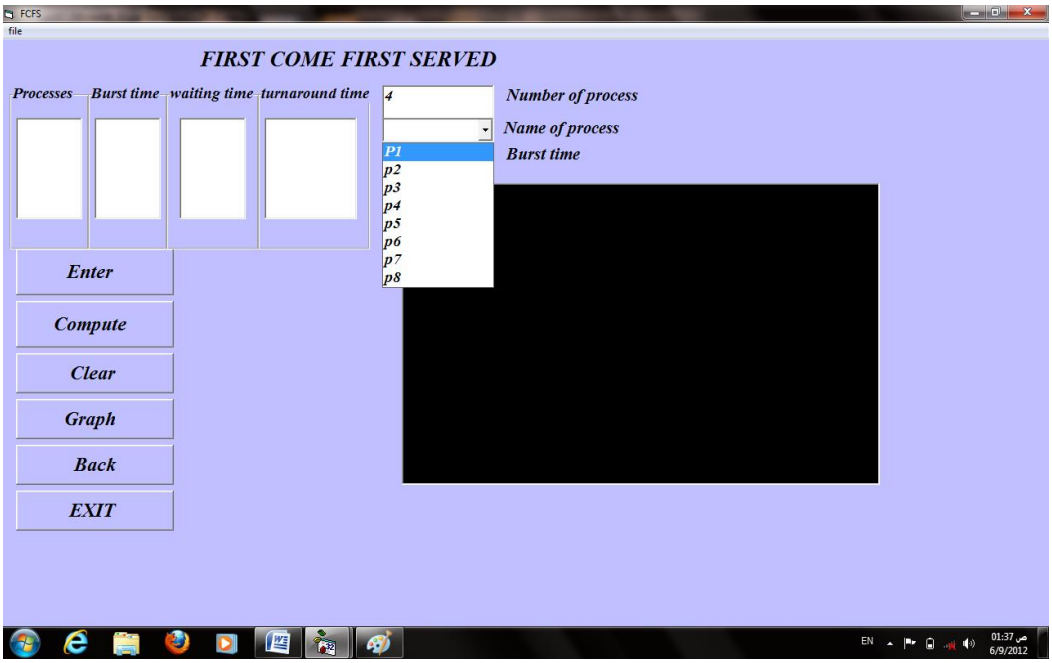


Figure (3): FCFS algorithm.

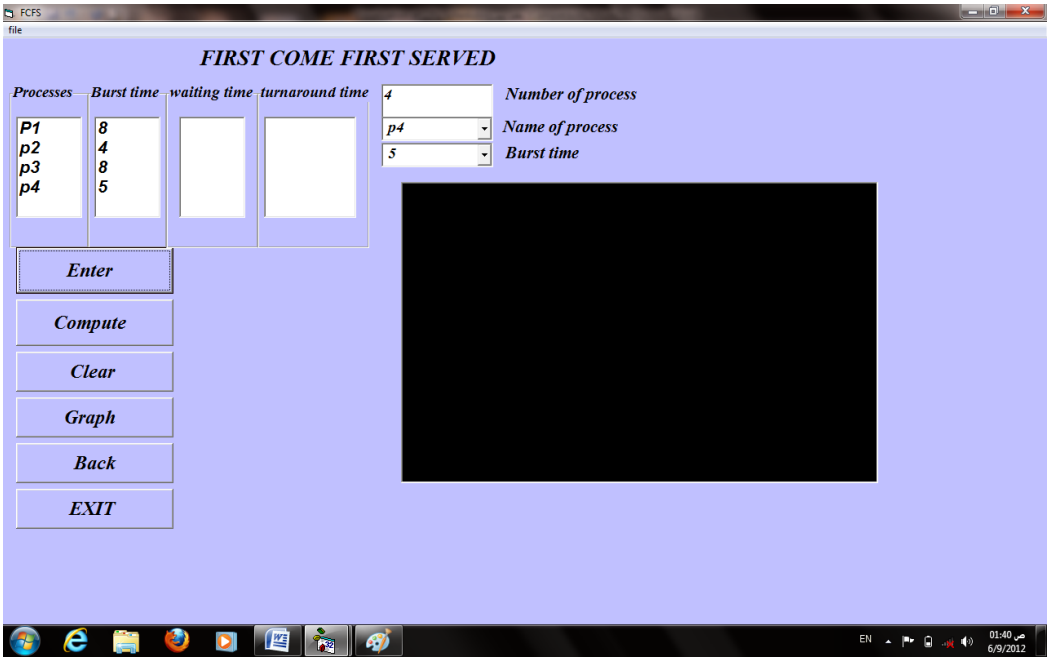


Figure (4): FCFS Algorithm entry data for process.

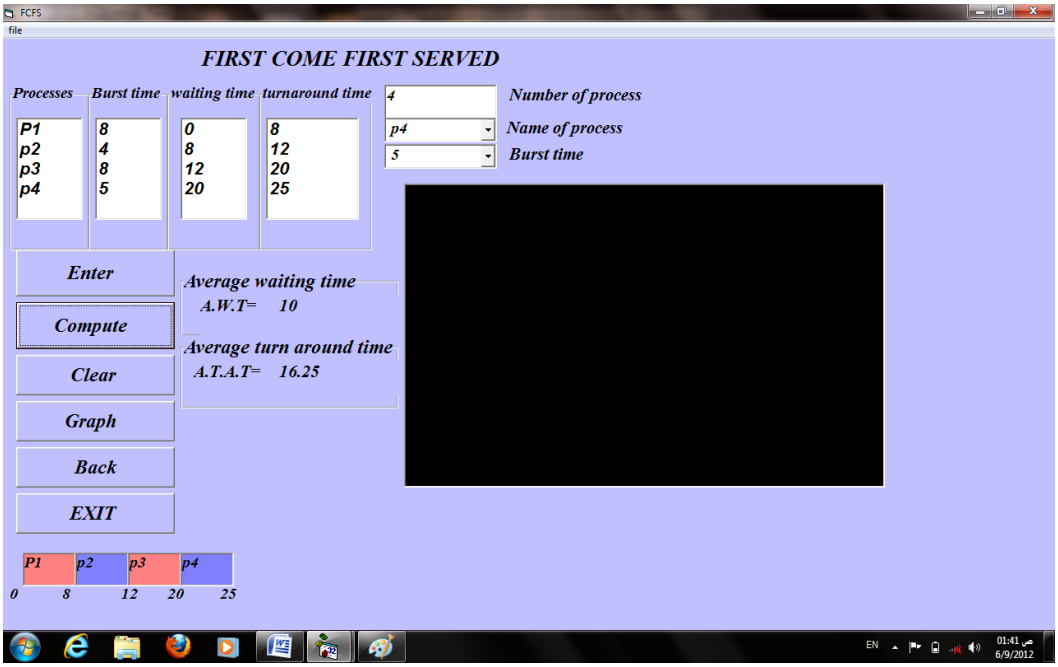


Figure (5): FCFS Algorithm compute time.

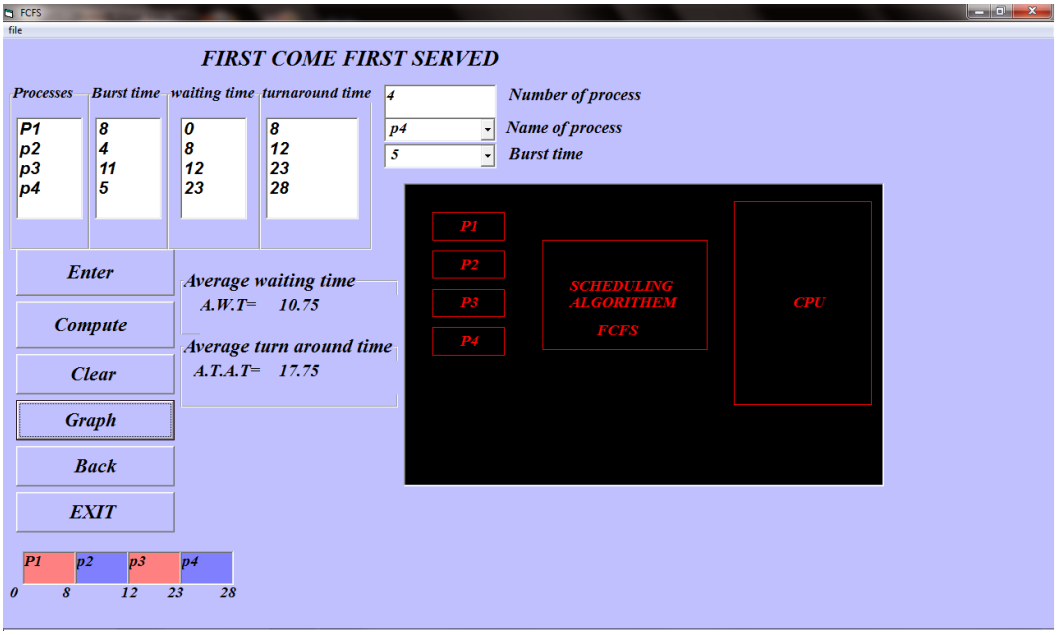


Figure (6): FCFS Algorithm graph before computing.

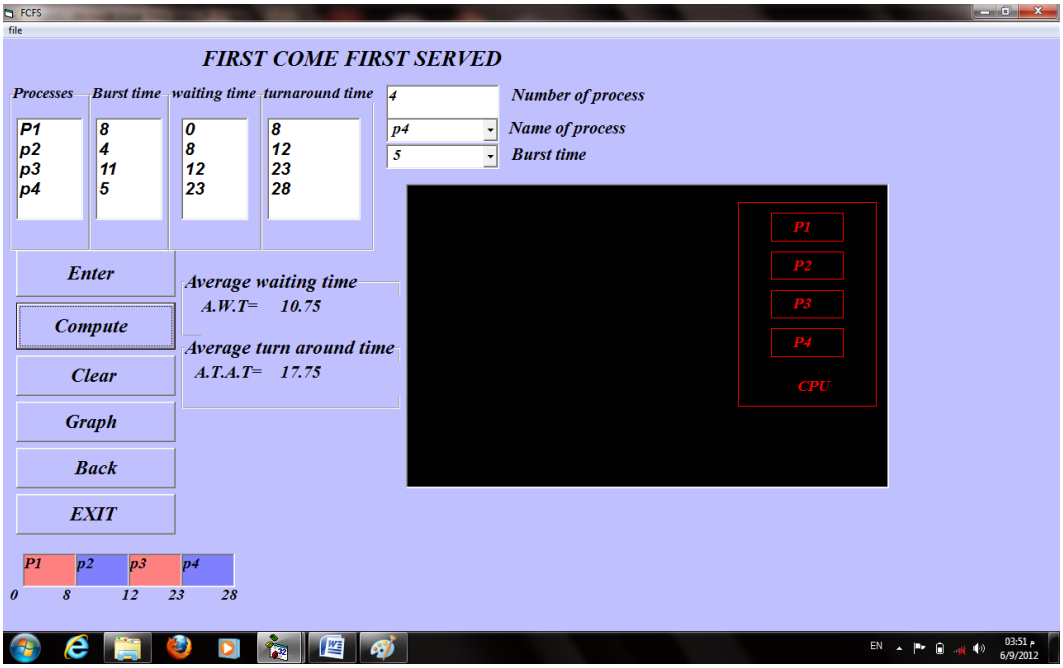


Figure (7): FCFS Algorithm graph after computing.

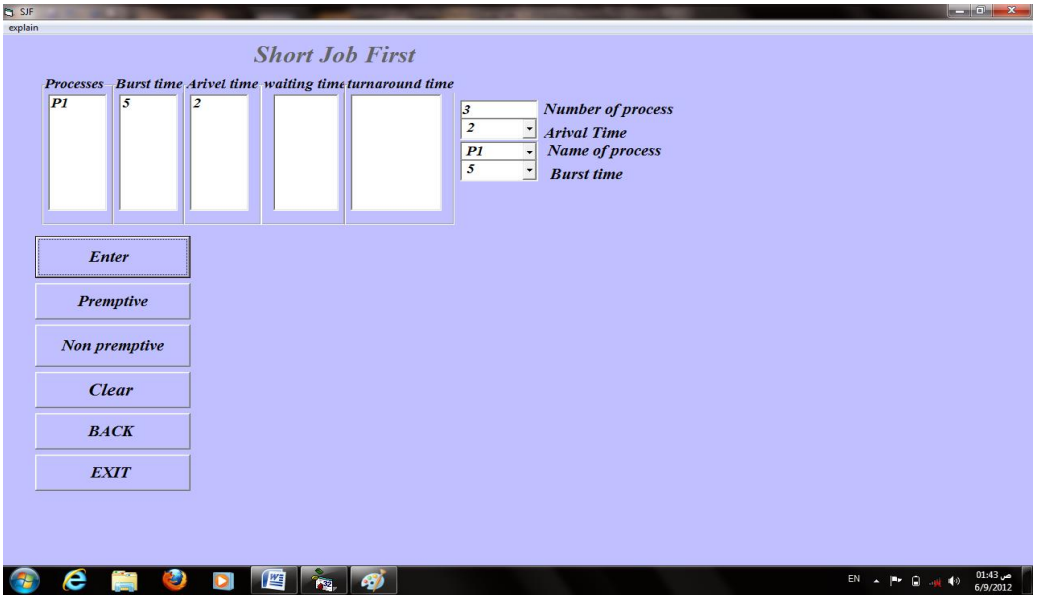


Figure (8): SJF Algorithm entering data.

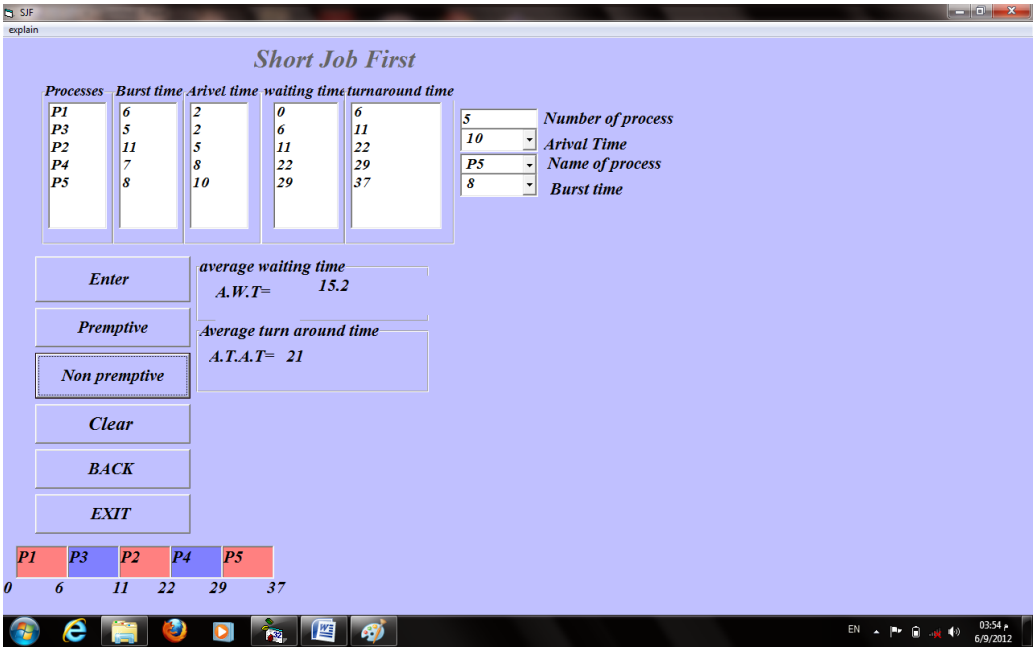


Figure (9): SJF Algorithm performoing Nonpreemtive selection.

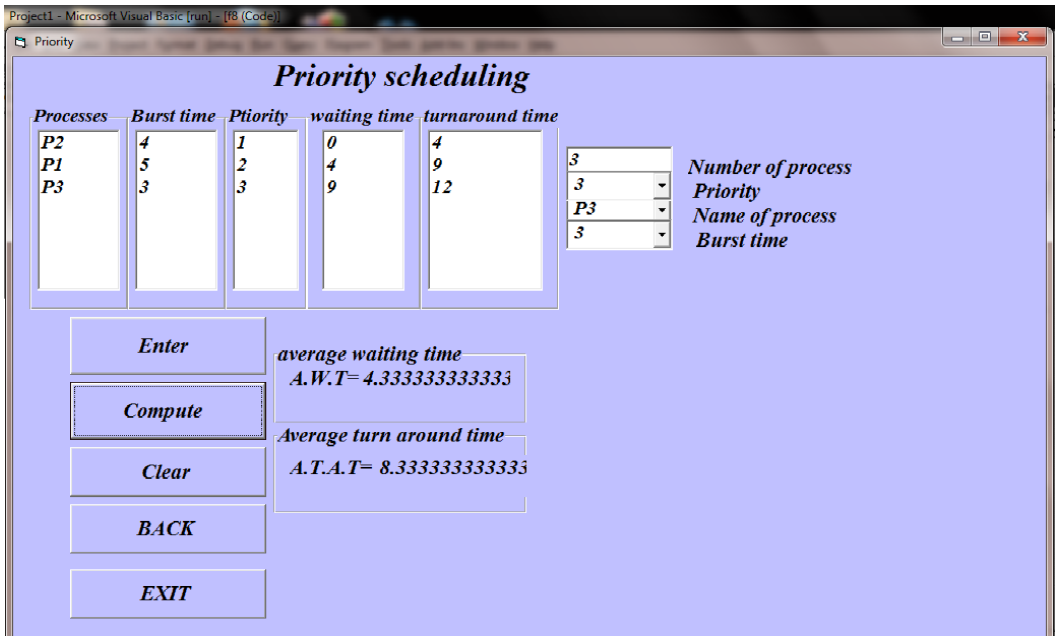


Figure (10): Priority Algorithm.

الجدولة المرئية لوحدة المعالجة المركزية

تقوى فليح حسن
مدرس مساعد
قسم هندسة الحاسبات والبرمجيات/ كلية الهندسة / جامعة ديالى

الخلاصة:

الجدولة هي مفتاح مفهوم تعدد المهام في الحاسوب وتعدد العمليات لتصميم انظمة التشغيل , وفي تصميم انظمة التشغيل ذات الوقت الحقيقي. جدولة وحدة المعالجة المركزية CPU Scheduling هي اساس انظمة التشغيل متعددة البرامج فبتحويل CPU بين العمليات تجعل انظمة التشغيل الكمبيوتر اكثر انتاجية, تستخدم خوارزميات الجدولة على نطاق واسع في شبكات الاتصالات وأنظمة التشغيل بتخصيص الموارد للمهام المتنافسة. في هذا البحث تم تصميم واجهات مرئية لخوارزميات جدولة وحدة المعالجة المركزية باستخدام لغة الفيجول بيسك6 , وممكن استخدامها لتعليم المستخدمين حول هذا الخوارزميات وكيفية عملها.