Student Name: Ashish Lepcha

Student Id:11603969

Roll:10

Email Address:<http://lepchahalpern@gmail.com>

GitHub Link: <https://github.com/vortex0ash/operatingsystem>

\*Problem: The traditional UNIX scheduler enforces an inverse relationship between priority

numbers and priorities: The higher the number, the lower the priority. The scheduler

recalculates process priorities once per second using the following function:

Priority = (Recent CPU usage / 2) + Base

where base = 60 and recent CPU usage refers to a value indicating how often

a process has used the CPU since priorities were last recalculated.

Assume that recent CPU usage for process P1 is 40, process P2 is 18, and process

P3 is 10. What will be the new priorities for these three processes when priorities are

recalculated? Based on this information, does the traditional UNIX scheduler raise or

lower the relative priority of a CPU-bound process?

Solutions:

|  |
| --- |
| formula: |
|  | Priority = (Recent CPU usage / 2) + Base. |
|  | Base = 60. |
|  | Recent CPU usage: |
|  | p1=40. |
|  | p2=18. |
|  | p3=10. |
|  |  |
|  | New priorities: |
|  | Priority for process 1: |
|  | Priority=(40 /2) + 60 |
|  | =80. |
|  | Priority for process 2: |
|  | Priority=(18 / 2) + 60. |
|  | = 69. |
|  | Priority for process 3: |
|  | Priority = (10 / 2) + 60. |
|  | = 65. |
|  |  |
|  | The priorities assigned to the processes are 80, 69, and 65 respectively. |
|  | The scheduler lowers the relative priority of CPU-bound process.  Code:  #include<stdio.h>    int main()  {  int n, cpuUsage[20],i,j,pit[20];  printf("Enter total number of processes:");  scanf("%d", &n);    printf("\nEnter new Cpu Usage\n");  for(i=0;i<n;i++)  {  printf("p[%d]:",i+1);  scanf("%d", &cpuUsage[i]);  }  pit[0]=0;  //calculating new priorities;  for(i=0;i<n;i++)  {  pit[i]=((cpuUsage[i] / 2)+ 60);      }    printf("\nProcess\t\t\t\tNewPriority");  for(i=0;i<n;i++){  printf("\np[%d]\t\t\t\t%d\n",i+1,pit[i]);  }  return 0;  } |

Description:

In the above problem we are calculating new priorities using the given formulae in the problem, we are given the value of CPU usage in problem and calculating the new priorities.

If the process usage more of the CPU it is called CPU bound process where we have less I/O.

The schedulers fall into one of two general catagories:

-Non-preemptive scheduling: when the currently executing process gives up the CPU voluntarily.

-Preemptive scheduling: when the operating system decides to favor another process, preempting the currently executing process .The Scheduler can lower or raise the priority of a CPU-bound process on the basis of CPU usage and I/O for the process , and age of the process.

Schedulers helps to select process from the ready queue, there are algorithms to select process from the queue .The main functional components of the scheduler are:

-Enqueuer : add a pointer or reference to the process.

-Dispatcher : implements scheduling algorithm to pick the next process to run.

-Context Switcher : loads the selected process onto the CPU as the running process.

Execution of the process depend on these scheduler metrics:

1.CPU Utilization .

2.Throughput.

3.Service time.

4.Turnaround time for a process.

5.Response time.

Algorithms used are:

-First come, First serve scheduler

-Shortest job first scheduler.

-Round Robin scheduler.

-multi-level priority queue scheduler.

Test cases:

-should calculate any number of priorities using CPU usage value.

-Input for the number of process given by the user.

Output:

