Mawlana Bhashani Science and Technology University

Lab -Report

Report No:09
Course code: ICT-3110
Course title: Operating System Lab
Date of Performance:09-09-2020
Date of Submission:16-09-2020

Submitted by

Name: Ruku Shikder
ID: IT-18057
3th year 1st semester
Session: 2017-2018
Dept. of ICT
MBSTU.

Submitted To

Nazrul Islam
Assistant Professor
Dept. of ICT
MBSTU.

Experiment No :09
Experiment Name: Implementation of Priority Scheduling Algorithm

- What is Priority Scheduling Algorithm?
- How to implement in c?

Priority Scheduling Algorithm:

In priority scheduling algorithm each process has a priority associated with it and as each process hits the queue, it is stored in based on its priority so that process with higher priority are dealt with first. It should be noted that equal priority processes are scheduled in FCFS order.

To prevent high priority processes from running indefinitely the scheduler may decrease the priority of the currently running process at each clock tick (i.e., at each clock interrupt). If this action causes its priority to drop below that of the next highest process, a process switch occurs. Alternatively, each process may be assigned a maximum time quantum that it is allowed to run. When this quantum is used up, the next highest priority process is given a chance to run.

Limitations

The problem occurs when the operating system gives a particular task a very low priority, so it sits in the queue for a larger amount of time, not being dealt with by the CPU. If this process is something the user needs, there could be a very long wait, this process is known as "Starvation" or "Infinite Blocking".

Algorithm:

Step 1: Start the process

Step 2: Accept the number of processes in the ready Queue

Step 3: For each process in the ready Q, assign the process id and accept the

CPU burst time

Step 4: Sort the ready queue according to the priority number.

Step 5: Set the waiting of the first process as _0 and its burst time as its turnaround time

Step 6: Arrange the processes based on process priority

Step 7: for each process in the Ready Q calculate

Waiting time(n)= waiting time (n-1) + Burst time (n-1)

Turnaround time (n)= waiting time(n)+Burst time(n) Step 8: Calculate

Average waiting time = Total waiting Time / Number of process

Average Turnaround time = Total Turnaround Time / Number of process Print the results in an order

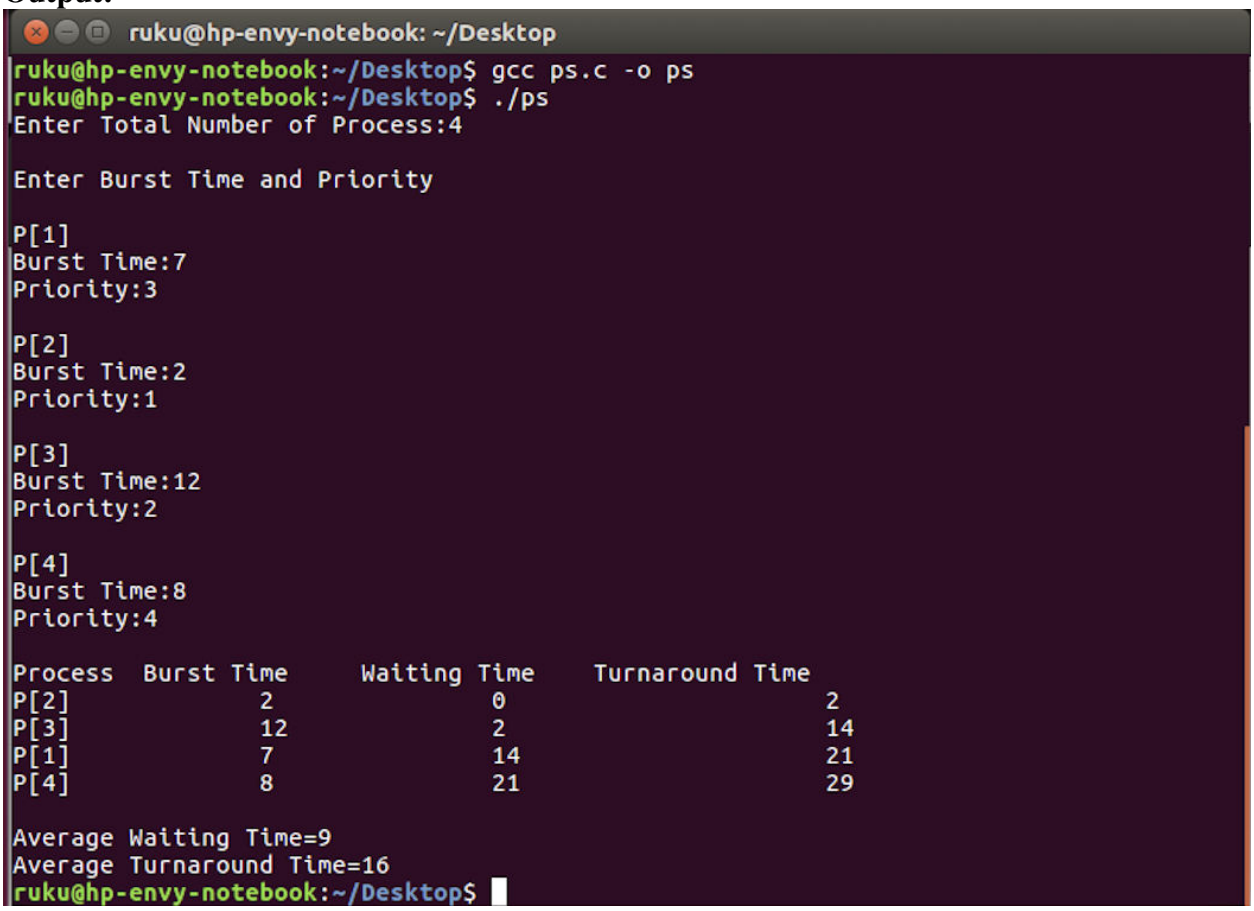
Step 9: Stop the process

Implement in c:

```
#include<stdio.h>

int main()
{
   int bt[20],p[20],wt[20],tat[20],pr[20],i,j,n,total=0,pos,temp,avg_wt,avg_tat;
   printf("Enter Total Number of Process:");
   scanf("%d",&n);

   printf("\nEnter Burst Time and Priority\n");
   for(i=0;i<n;i++)
   {
      printf("\nP[%d]\n",i+1);
      printf("Burst Time:");
      scanf("%d",&bt[i]);
      printf("Priority:");
      scanf("%d",&pr[i]);
      p[i]=i+1;          //contains process number
   }

   //sorting burst time, priority and process number in ascending order using selection sort
   for(i=0;i<n;i++)
   {
      pos=i;
      for(j=i+1;j<n;j++)
      {
```

```
            if(pr[j]<pr[pos])
                pos=j;
        }

        temp=pr[i];
        pr[i]=pr[pos];
        pr[pos]=temp;

        temp=bt[i];
        bt[i]=bt[pos];
        bt[pos]=temp;

        temp=p[i];
        p[i]=p[pos];
        p[pos]=temp;
    }

    wt[0]=0;              //waiting time for first process is zero

    //calculate waiting time
    for(i=1;i<n;i++)
    {
        wt[i]=0;
        for(j=0;j<i;j++)
            wt[i]+=bt[j];

        total+=wt[i];
    }

    avg_wt=total/n;      //average waiting time
    total=0;

    printf("\nProcess\t    Burst Time    \tWaiting Time\tTurnaround Time");
    for(i=0;i<n;i++)
    {
        tat[i]=bt[i]+wt[i];     //calculate turnaround time
        total+=tat[i];
        printf("\nP[%d]\t\t  %d\t\t    %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);
    }

    avg_tat=total/n;     //average turnaround time
    printf("\n\nAverage Waiting Time=%d",avg_wt);
    printf("\nAverage Turnaround Time=%d\n",avg_tat);

            return 0;
}
```

Output:



Discussion:

In this priority scheduling algorithm each process has a priority associated with it and as each process hits the queue, it is stored in based on its priority so that process with higher priority are dealt with first. It should be noted that equal priority processes are scheduled in FCFS order.