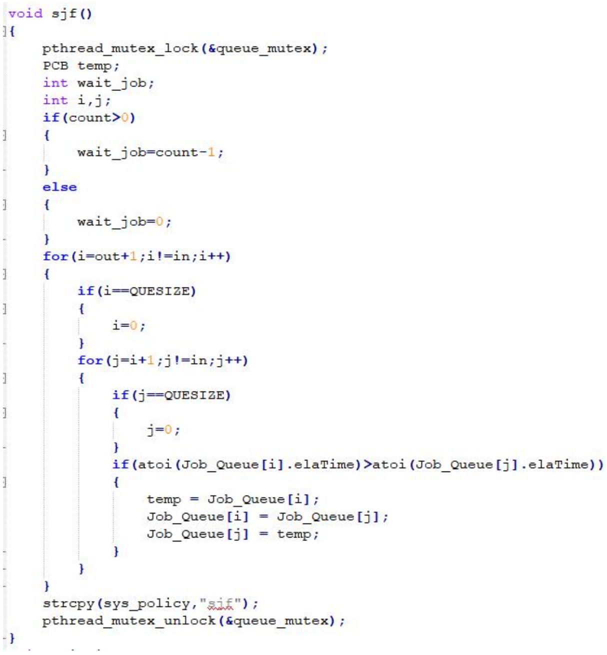
Project 3-AUbatch-A Batch Scheduling System

Gaoxiang Li gzl0034

1. Design:

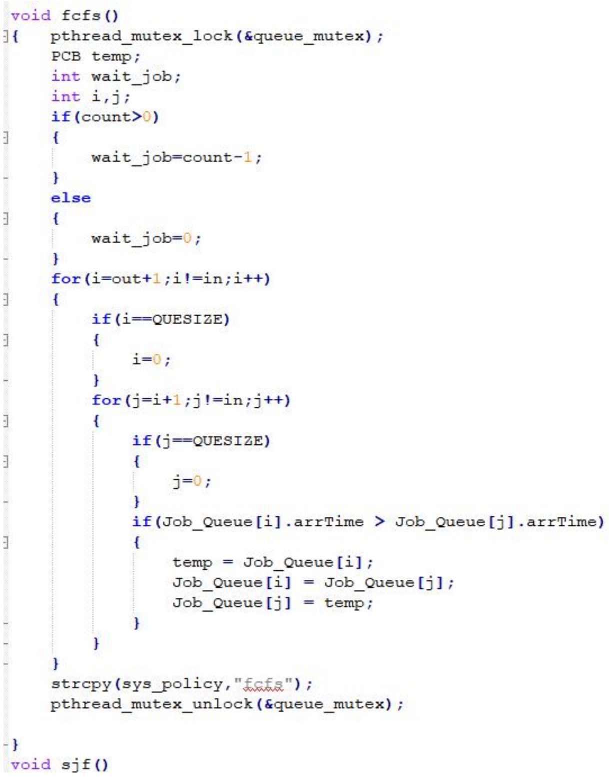
(1) three scheduling polices:design to use three policies to schedule the job queue here are the algorithm code.

SJF:



```
void sjf()
{
    pthread_mutex_lock(&queue_mutex);
    PCB temp;
    int wait_job;
    int i,j;
    if(count>0)
    {
        wait_job=count-1;
    }
    else
    {
        wait_job=0;
    }
    for(i=out+1;i!=in;i++)
    {
        if(i==QUESIZE)
        {
            i=0;
        }
        for(j=i+1;j!=in;j++)
        {
            if(j==QUESIZE)
            {
                j=0;
            }
            if(atoi(Job_Queue[i].elaTime)>atoi(Job_Queue[j].elaTime))
            {
                temp = Job_Queue[i];
                Job_Queue[i] = Job_Queue[j];
                Job_Queue[j] = temp;
            }
        }
    }
    strcpy(sys_policy,"sjf");
    pthread_mutex_unlock(&queue_mutex);
}
```

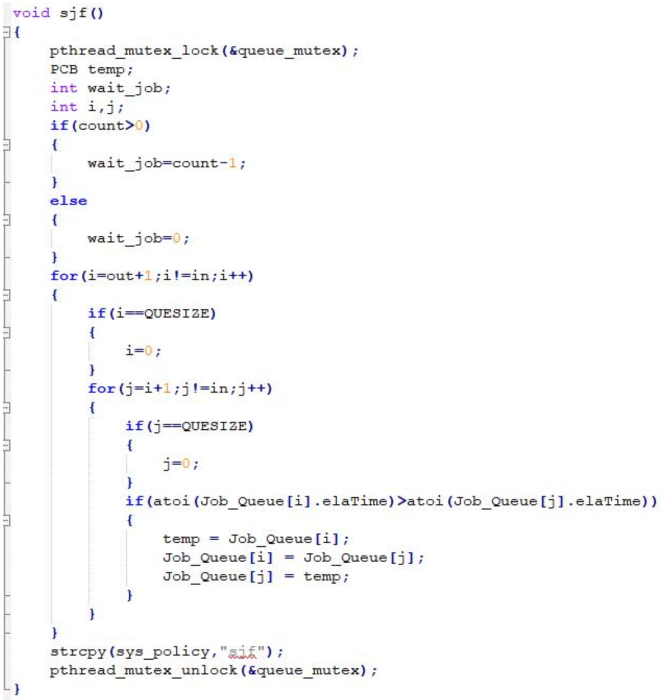
Fcfs:



```
void fcfs()
{   pthread_mutex_lock(&queue_mutex);
    PCB temp;
    int wait_job;
    int i,j;
    if(count>0)
    {
        wait_job=count-1;
    }
    else
    {
        wait_job=0;
    }
    for(i=out+1;i!=in;i++)
    {
        if(i==QUESIZE)
        {
            i=0;
        }
        for(j=i+1;j!=in;j++)
        {
            if(j==QUESIZE)
            {
                j=0;
            }
            if(Job_Queue[i].arrTime > Job_Queue[j].arrTime)
            {
                temp = Job_Queue[i];
                Job_Queue[i] = Job_Queue[j];
                Job_Queue[j] = temp;
            }
        }
    }
    strcpy(sys_policy,"fcfs");
    pthread_mutex_unlock(&queue_mutex);

}
void sjf()
```

Priority:



```
void sjf()
{
    pthread_mutex_lock(&queue_mutex);
    PCB temp;
    int wait_job;
    int i,j;
    if(count>0)
    {
        wait_job=count-1;
    }
    else
    {
        wait_job=0;
    }
    for(i=out+1;i!=in;i++)
    {
        if(i==QUESIZE)
        {
            i=0;
        }
        for(j=i+1;j!=in;j++)
        {
            if(j==QUESIZE)
            {
                j=0;
            }
            if(atoi(Job_Queue[i].elaTime)>atoi(Job_Queue[j].elaTime))
            {
                temp = Job_Queue[i];
                Job_Queue[i] = Job_Queue[j];
                Job_Queue[j] = temp;
            }
        }
    }
    strcpy(sys_policy,"sjf");
    pthread_mutex_unlock(&queue_mutex);
}
```

(2) Command line parser:design to implement “help” “quit” “list” “run” “test” and switch policy function.(more details in source code)

(3) Scheduler and Dispatcher: Scheduler get the submitted job and schedule the job, then put job into job queue. Dispatcher get and execv the job from job queue. Two processes use pthread to control synchronization between these two.(more details in

source code)

2. Sample input and output:

(1) priority policy

Input: test job priority 5 5 1 10

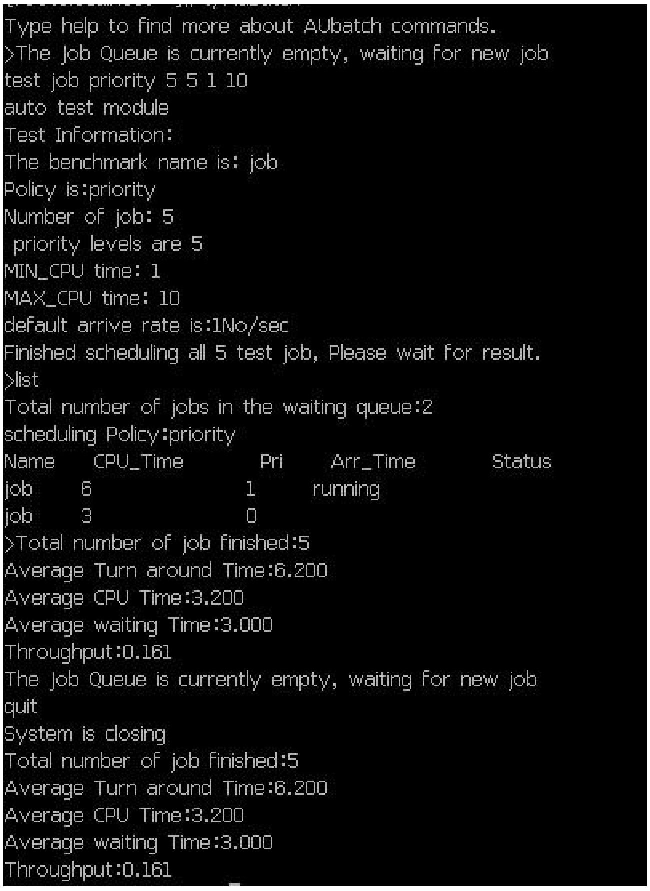
Benchmark:job policy:priority number of job:5 priority level:5

MIN_CPU TIME:1 MAX_CPU TIME 10

>list to show job queue

>quit to exit and show performance data

Here are the output screenshot:



```
Type help to find more about AUbatch commands.
>The Job Queue is currently empty, waiting for new job
test job priority 5 5 1 10
auto test module
Test Information:
The benchmark name is: job
Policy is:priority
Number of job: 5
 priority levels are 5
MIN_CPU time: 1
MAX_CPU time: 10
default arrive rate is:1No/sec
Finished scheduling all 5 test job, Please wait for result.
>list
Total number of jobs in the waiting queue:2
scheduling Policy:priority
Name    CPU_Time        Pri     Arr_Time        Status
job     6               1      running
job     3               0
>Total number of job finished:5
Average Turn around Time:6.200
Average CPU Time:3.200
Average waiting Time:3.000
Throughput:0.161
The Job Queue is currently empty, waiting for new job
quit
System is closing
Total number of job finished:5
Average Turn around Time:6.200
Average CPU Time:3.200
Average waiting Time:3.000
Throughput:0.161
```

(2)FCFS

Input: test job fcfs 5 5 1 10

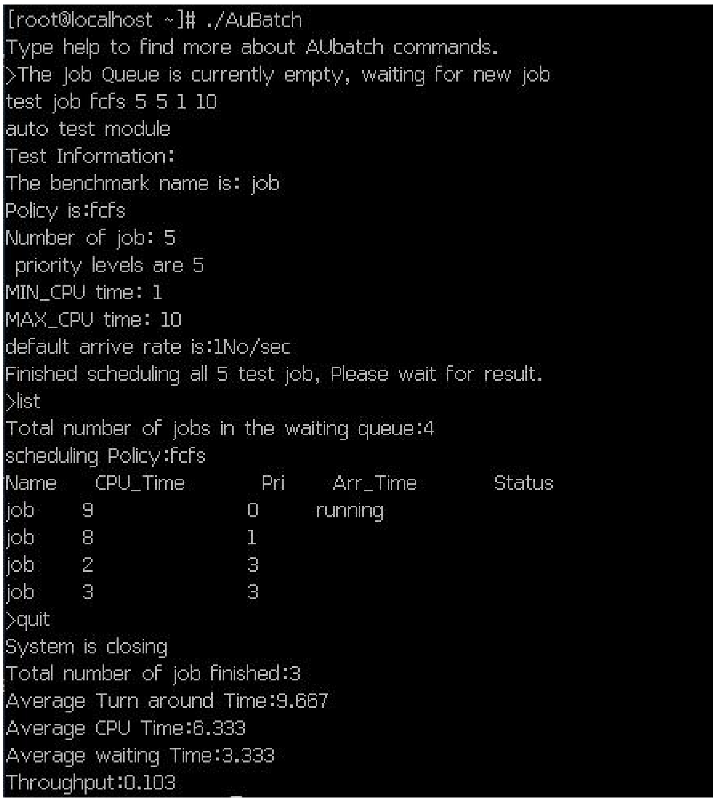
Benchmark:job policy:fcfs number of job:5 priority level:5

MIN_CPU TIME:1 MAX_CPU TIME 10

>list to show job queue

>quit to exit and show performance data

Here are the output screenshot:



```
[root@localhost ~]# ./AuBatch
Type help to find more about AUbatch commands.
>The Job Queue is currently empty, waiting for new job
test job fcfs 5 5 1 10
auto test module
Test Information:
The benchmark name is: job
Policy is:fcfs
Number of job: 5
 priority levels are 5
MIN_CPU time: 1
MAX_CPU time: 10
default arrive rate is:1No/sec
Finished scheduling all 5 test job, Please wait for result.
>list
Total number of jobs in the waiting queue:4
scheduling Policy:fcfs
Name    CPU_Time        Pri     Arr_Time        Status
job     9               0      running
job     8               1
job     2               3
job     3               3
>quit
System is closing
Total number of job finished:3
Average Turn around Time:9.667
Average CPU Time:6.333
Average waiting Time:3.333
Throughput:0.103
```

(3)SJF

Input: test job sjf 5 5 1 10

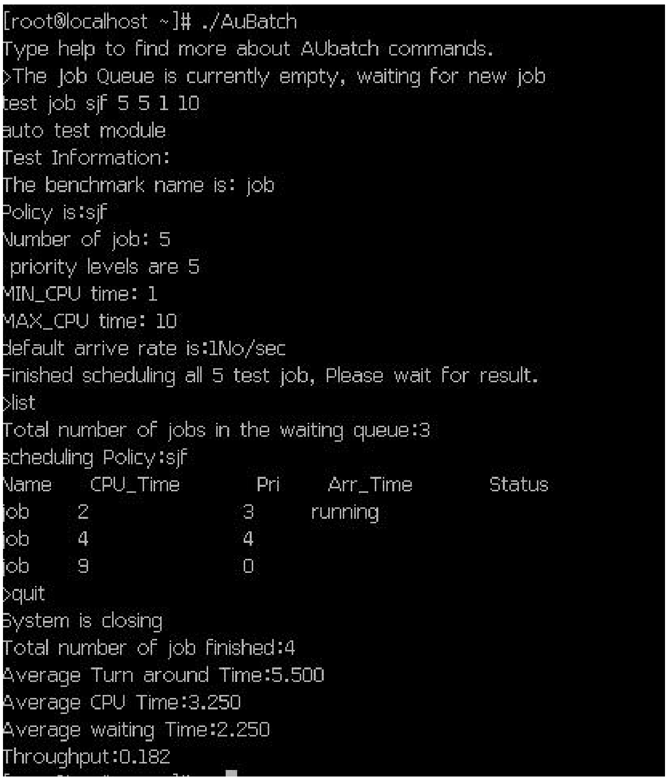
Benchmark:job   policy:sjf   number of job:5   priority level:5

MIN_CPU TIME:1   MAX_CPU TIME 10

>list to show job queue

>quit to exit and show performance data

Here are the output screenshot:



```
[root@localhost ~]# ./AuBatch
Type help to find more about AUbatch commands.
>The Job Queue is currently empty, waiting for new job
test job sjf 5 5 1 10
auto test module
Test Information:
The benchmark name is: job
Policy is:sjf
Number of job: 5
 priority levels are 5
MIN_CPU time: 1
MAX_CPU time: 10
default arrive rate is:1No/sec
Finished scheduling all 5 test job, Please wait for result.
>list
Total number of jobs in the waiting queue:3
scheduling Policy:sjf
Name    CPU_Time        Pri     Arr_Time        Status
job     2               3     running
job     4               4
job     9               0
>quit
System is closing
Total number of job finished:4
Average Turn around Time:5.500
Average CPU Time:3.250
Average waiting Time:2.250
Throughput:0.182
```

Reference: Source code and slides provided by Dr.Qin
Scheduling policy algorithm from stack overflow (idea only)