

Project 4

Scheduling Algorithms

Introduction

The project implements five scheduling algorithms: First-Come-First-Serve (FCFS), Shortest-Job-First (SJF), Priority, Round-Robin (RR), and Priority with Round-Robin. The project simulates the scheduling of processes and calculates the average waiting time, turnaround time, and response time for each algorithm. The project also use atomic variables to ensure thread safety when allocating tid to processes.

Implementation

FCFS

The First-Come-First-Serve algorithm schedules processes based on the order they arrive. The algorithm is implemented using a queue to store the processes. The algorithm traverses the queue and runs each process in order.

SJF

The Shortest-Job-First algorithm schedules processes based on the burst time of the processes. The algorithm traverses the queue and finds the process with the shortest burst time to run next, and removes the process from the queue.

Priority

The Priority algorithm schedules processes based on the priority of the processes. The algorithm traverses the queue and finds the process with the highest priority to run next, and removes the process from the queue.

RR

The Round-Robin algorithm schedules processes in a circular queue. The algorithm runs each process for a fixed time quantum and removes the process from the queue if the process is completed. When it reaches the end of the queue, the algorithm starts from the beginning of the queue again, until all processes are completed.

Priority with RR

The Priority with Round-Robin algorithm schedules processes based on the priority of the processes. The processes have a maximum and a minimum priority. The algorithm traverses from the highest priority to the lowest priority and checks if there are any processes in the queue with the current priority. If there are processes with the current priority, the algorithm runs the processes in a round-robin fashion, as is introduced in the RR algorithm.

Atomic Variables

The project provides a `get_tid()` function to allocate a unique tid to each process. The Implementation is:

```
int get_tid()
{
    static atomic_int tid = 0;
    return atomic_fetch_add(&tid, 1);
}
```

All threads call the `get_tid()` function to get a unique tid. The `atomic_fetch_add()` function ensures that the tid is incremented atomically, so each process gets a unique tid.

Statistics

The project calculates the average waiting time, turnaround time, and response time for each scheduling algorithm. The statistics are updated in the `run()` function for each algorithm. The statistics are stored in arrays for each process, and the average is calculated at the end of the simulation.

```
static int *burst_time      = NULL;
static int *turnaround_time = NULL;
static int *waiting_time    = NULL;
static int *response_time   = NULL;
static int *response_flag   = NULL;
static int  time            = 0;
static int  task_count      = 0;

void init_stats(int tsk)
{
    // Initialization code
}

void print_stats()
{
    // Print statistics
}

void run(Task *task, int slice)
{
    printf("Running task = [%s] [%d] [%d] for %d units.\n",
           task->name, task->priority, task->burst, slice);

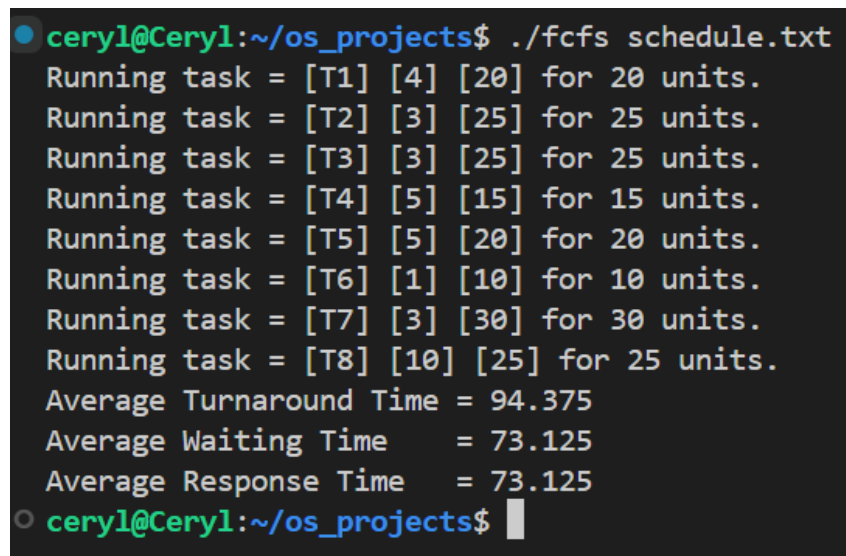
    if (response_flag[task->tid] == 0)
```

```
{
    burst_time[task->tid] = task->burst;
    response_time[task->tid] = time;
    response_flag[task->tid] = 1;
}
time += slice;

if (task->burst - slice == 0)
{
    turnaround_time[task->tid] = time;
    waiting_time[task->tid] = time - task->burst;
}
}
```

Correctness

The following figures show the results of the input `schedule.txt` file for each scheduling algorithm.

A terminal window showing the execution of a program for the FCFS scheduling algorithm. The user runs './fcfs schedule.txt' in the directory ~/os_projects. The output lists the execution order of tasks T1 through T8, their burst times, and the number of units they run. It also calculates the average turnaround time, waiting time, and response time.

```
● ceryl@Ceryl:~/os_projects$ ./fcfs schedule.txt
Running task = [T1] [4] [20] for 20 units.
Running task = [T2] [3] [25] for 25 units.
Running task = [T3] [3] [25] for 25 units.
Running task = [T4] [5] [15] for 15 units.
Running task = [T5] [5] [20] for 20 units.
Running task = [T6] [1] [10] for 10 units.
Running task = [T7] [3] [30] for 30 units.
Running task = [T8] [10] [25] for 25 units.
Average Turnaround Time = 94.375
Average Waiting Time    = 73.125
Average Response Time   = 73.125
○ ceryl@Ceryl:~/os_projects$
```

Figure 1: FCFS

```
● ceryl@Ceryl:~/os_projects$ ./sjf schedule.txt
Running task = [T6] [1] [10] for 10 units.
Running task = [T4] [5] [15] for 15 units.
Running task = [T1] [4] [20] for 20 units.
Running task = [T5] [5] [20] for 20 units.
Running task = [T2] [3] [25] for 25 units.
Running task = [T3] [3] [25] for 25 units.
Running task = [T8] [10] [25] for 25 units.
Running task = [T7] [3] [30] for 30 units.
Average Turnaround Time = 82.500
Average Waiting Time    = 61.250
Average Response Time   = 61.250
○ ceryl@Ceryl:~/os_projects$
```

Figure 2: SJF

```
● ceryl@Ceryl:~/os_projects$ ./priority schedule.txt
Running task = [T6] [1] [10] for 10 units.
Running task = [T2] [3] [25] for 25 units.
Running task = [T3] [3] [25] for 25 units.
Running task = [T7] [3] [30] for 30 units.
Running task = [T1] [4] [20] for 20 units.
Running task = [T4] [5] [15] for 15 units.
Running task = [T5] [5] [20] for 20 units.
Running task = [T8] [10] [25] for 25 units.
Average Turnaround Time = 93.125
Average Waiting Time    = 71.875
Average Response Time   = 71.875
○ ceryl@Ceryl:~/os_projects$
```

Figure 3: Priority

```
● ceryl@Ceryl:~/os_projects$ ./rr schedule.txt
Running task = [T1] [4] [20] for 5 units.
Running task = [T2] [3] [25] for 5 units.
Running task = [T3] [3] [25] for 5 units.
Running task = [T4] [5] [15] for 5 units.
Running task = [T5] [5] [20] for 5 units.
Running task = [T6] [1] [10] for 5 units.
Running task = [T7] [3] [30] for 5 units.
Running task = [T8] [10] [25] for 5 units.
Running task = [T1] [4] [15] for 5 units.
Running task = [T2] [3] [20] for 5 units.
Running task = [T3] [3] [20] for 5 units.
Running task = [T4] [5] [10] for 5 units.
Running task = [T5] [5] [15] for 5 units.
Running task = [T6] [1] [5] for 5 units.
Running task = [T7] [3] [25] for 5 units.
Running task = [T8] [10] [20] for 5 units.
Running task = [T1] [4] [10] for 5 units.
Running task = [T2] [3] [15] for 5 units.
Running task = [T3] [3] [15] for 5 units.
Running task = [T4] [5] [5] for 5 units.
Running task = [T5] [5] [10] for 5 units.
Running task = [T7] [3] [20] for 5 units.
Running task = [T8] [10] [15] for 5 units.
Running task = [T1] [4] [5] for 5 units.
Running task = [T2] [3] [10] for 5 units.
Running task = [T3] [3] [10] for 5 units.
Running task = [T5] [5] [5] for 5 units.
Running task = [T7] [3] [15] for 5 units.
Running task = [T8] [10] [10] for 5 units.
Running task = [T2] [3] [5] for 5 units.
Running task = [T3] [3] [5] for 5 units.
Running task = [T7] [3] [10] for 5 units.
Running task = [T8] [10] [5] for 5 units.
Running task = [T7] [3] [5] for 5 units.
Average Turnaround Time = 133.125
Average Waiting Time     = 128.125
Average Response Time    = 17.500
○ ceryl@Ceryl:~/os_projects$
```

Figure 4: RR

```

● ceryl@Ceryl:~/os_projects$ ./priority_rr schedule.txt
Running task = [T6] [1] [10] for 5 units.
Running task = [T6] [1] [5] for 5 units.
Running task = [T2] [3] [25] for 5 units.
Running task = [T3] [3] [25] for 5 units.
Running task = [T7] [3] [30] for 5 units.
Running task = [T2] [3] [20] for 5 units.
Running task = [T3] [3] [20] for 5 units.
Running task = [T7] [3] [25] for 5 units.
Running task = [T2] [3] [15] for 5 units.
Running task = [T3] [3] [15] for 5 units.
Running task = [T7] [3] [20] for 5 units.
Running task = [T2] [3] [10] for 5 units.
Running task = [T3] [3] [10] for 5 units.
Running task = [T7] [3] [15] for 5 units.
Running task = [T2] [3] [5] for 5 units.
Running task = [T3] [3] [5] for 5 units.
Running task = [T7] [3] [10] for 5 units.
Running task = [T7] [3] [5] for 5 units.
Running task = [T1] [4] [20] for 5 units.
Running task = [T1] [4] [15] for 5 units.
Running task = [T1] [4] [10] for 5 units.
Running task = [T1] [4] [5] for 5 units.
Running task = [T4] [5] [15] for 5 units.
Running task = [T5] [5] [20] for 5 units.
Running task = [T4] [5] [10] for 5 units.
Running task = [T5] [5] [15] for 5 units.
Running task = [T4] [5] [5] for 5 units.
Running task = [T5] [5] [10] for 5 units.
Running task = [T5] [5] [5] for 5 units.
Running task = [T8] [10] [25] for 5 units.
Running task = [T8] [10] [20] for 5 units.
Running task = [T8] [10] [15] for 5 units.
Running task = [T8] [10] [10] for 5 units.
Running task = [T8] [10] [5] for 5 units.
Average Turnaround Time = 101.875
Average Waiting Time     = 96.875
Average Response Time    = 63.125
○ ceryl@Ceryl:~/os_projects$

```

Figure 5: Priority with RR

The average time is shown below:

Algorithm	Average Waiting Time	Average Turnaround Time	Average Response Time
FCFS	94.375	73.125	73.125

Algorithm	Average Waiting Time	Average Turnaround Time	Average Response Time
SJF	82.500	61.250	61.250
Priority	93.125	71.875	71.875
RR	133.125	128.125	17.500
Priority with RR	101.875	96.875	63.125

Conclusion

The project implements five scheduling algorithms and calculates the average waiting time, turnaround time, and response time for each algorithm.

- SJF has the lowest average waiting time and turnaround time. RR has the lowest average response time.
- RR has the highest average waiting time and turnaround time. FCFS has the highest average response time.