

Part 1 report
Aziz Hamad 101232108
Ismael Ouadria 101284947

Introduction

In this report, we design, test, and analyse the behavior of three CPU scheduling algorithms: External Priority (EP), Round Robin (RR), and External Priority and Round Robin combined (EP_RR) under different workloads. To explore the behavior of these algorithms, we have designed 20 different test workloads consisting of CPU-bound programs, programs that are I/O heavy, bursts of CPU and I/O in different combinations, staggered programs, and workloads that involve memory pressure and fragmentation.

For each workload we analyzed them using Four metrics:

- Turnaround time
- Waiting time
- Response time
- Throughput

Description of each scheduling algorithms

External Priority (EP)

In External Priority (EP) the ready process that has the highest static priority is always selected. If a new task with a higher priority is added to the system, a context switch occurs.
The Impact: This may cause lower-priority processes to wait for a long time, especially if there are many high-priority processes present in the system.

Round Robin (RR)

In Round Robin (RR) scheduling, the ready processes are scheduled in a cyclic manner and each process is given a time quantum of 5.

The Impact: This process scheduling method is fair does not starve any process

EP_RR

EP_RR sorts processes by their priority and then executes them in a round-robin fashion within each priority level.

It is more balanced than EP by itself, but it has more context switches because it uses round robin inside each priority group.

Scenario design

The 20 scenarios includes:

- CPU bound workloads (t2, t7)
- I/O heavy workloads (t6)
- Workloads of mixed bursts (t3, t4, t9)

- Workloads with staggered arrivals (t5, t8)
- Memory pressure or fragmentation (t13, t16, t18)
- Priority stress tests (t10 through t12)
- RR fairness tests (t14, t15)

We covered a wide variety of scenarios so we can see how each of the algorithms would respond to them.

Execution summary

each of the scenarios acts differently, however we there are a few consistent trends that we observed.

CPU bound workloads

- Time is allocated evenly to each process by round robin.
- When priorities do not differ, both EP and EP_RR behave the same.
- The waiting and response times are both low.

I/O Heavy workloads

Frequent transitions to the WAITING state create long periods of idle CPU time. All three schedulers behave similarly, and differences in ready-queue ordering rarely change the results.

Mixed workloads

- A high-priority process may dominate CPU time under EP.
- RR improves preemption fairness.
- EP_RR incurs a small amount of extra context-switch overhead.

These differences highlight clearly the key functions of each algorithm.

Memory pressure/fragmentation

It is common for processes to wait before they can even enter the ready queue, These delays are due to memory access and not scheduling. All algorithms are equally affected by fragmentation.

Metrics summary

The overall scheduler performance

Schedular	Avg throughput	Avg turnaround	Avg waiting	Avg response
EP	0.08539	552.17	84.69	44.23
EP_RR	0.08539	551.57	84.52	44.12
RR	0.08539	551.40	84.44	44.09

Scenario comparisons

Scenario for CPU-Bound(t2)

Schedular	turnaround	waiting	response
EP	11	0	0
EP_RR	11	0	0
RR	11	0	0

I/O-bound scenario (t6)

Schedular	turnaround	waiting	response
EP	300	100	100
EP_RR	300	100	100
RR	300	100	100

Scenario with mixed workload(t9)

schedular	Avg turnaround	Avg waiting	Avg response
EP	811.33	39.67	0
EP_RR	818.00	46.33	0
RR	811.33	39.67	0

memory pressure scenario (t13)

Schedular	turnaround	waiting	response
EP	247.5	150	150
EP_RR	247.5	150	150

RR	247.5	150	150
----	-------	-----	-----

The result

We observed that RR performs the best in scenarios where fairness and response time matter because of its preemption pattern. However for EP and EPRR they Sometimes behave similarly to each other, although EPRR slightly loses out due to the extra context switches.