MỤC LỤC

Đề mục Trang

PREFACE ....................................................................................................................

CHAPTER 1: THEORETICAL FOUNDATIONS ......................................................

1.1 INTRODUCTION ................................................................................................. 2  
1.1.1 Scheduling Objectives ....................................................................................... 2  
1.1.2 Characteristics of Processes ............................................................................ 2

1.2 BASIC CONCEPTS ............................................................................................. 3  
1.2.1 Concept of CPU Time ....................................................................................... 3  
1.2.2 Process States Related to CPU Time .............................................................. 4  
1.2.3 Concept of CPU Scheduling ............................................................................ 5

1.3 SCHEDULING ALGORITHMS .......................................................................... 6  
1.3.1 First Come First Served (FCFS) ....................................................................... 6  
1.3.2 Round Robin (RR) ............................................................................................ 7  
1.3.3 Shortest Job First (SJF) .................................................................................... 8  
1.3.4 Shortest Remaining Time (SRT) ...................................................................... 9

CHAPTER 1: THEORETICAL FOUNDATIONS

1.1 INTRODUCTION

1.1.1 Scheduling Objectives

The scheduler does not provide mechanisms but makes decisions. Operating systems use different scheduling strategies to manage processes, but they all share common objectives as follows:

* Fairness: Processes share the CPU fairly so that no process waits indefinitely for CPU allocation.
* Efficiency: The system should maximize CPU utilization — ideally keeping the CPU busy 100% of the time.
* Reasonable response time: The system should minimize response time, especially for interactive users.
* Minimized turnaround time: Reduce the total time required to complete all batch jobs.
* Maximum throughput: Maximize the number of jobs processed per time unit.

However, it is often impossible to satisfy all these goals simultaneously. They sometimes conflict, and compromises must be made depending on system priorities.

1.1.2 Characteristics of Processes

Process scheduling is a complex task that requires considering several factors to meet system goals. Some important characteristics affecting scheduling include:

* I/O-bound vs. CPU-bound processes:  
  I/O-bound processes spend most of their time waiting for I/O operations and use the CPU only for short periods.  
  In contrast, CPU-bound processes spend most of their time performing computations and use the CPU for long durations.
* Interactive vs. batch processes:  
  Interactive processes require quick responses to user actions, whereas batch processes handle large volumes of tasks with acceptable delays.
* Process priority:  
  Each process can be assigned a priority level. Higher-priority processes (more important or urgent) should receive CPU service before lower-priority ones.
* CPU burst time:  
  Some scheduling methods prefer processes with shorter CPU bursts, expecting shorter waiting times and higher system responsiveness.
* Remaining execution time:  
  Systems may attempt to minimize average waiting time by predicting how long each process still needs to complete.

1.2 BASIC CONCEPTS

1.2.1 CPU Time (CPU Burst)

The CPU is one of the most critical resources in a computer. Every process needs CPU service to perform tasks such as calculations or data processing.  
The time during which a process uses the CPU is called CPU time (or a CPU burst).

At any given moment, only one process can be assigned to the CPU to execute its instructions.

1.2.2 Process States Related to CPU Time

In a multiprogramming environment, each process can be in one of three main states related to CPU use:

* Ready: The process has all necessary resources and is waiting for CPU allocation.
* Running: The process is currently using the CPU.
* Waiting: The process cannot continue execution because it is waiting for an event (e.g., I/O completion or additional resources).

During execution, a process repeatedly transitions between these states, as shown in the diagram:

A diagram of a diagram

AI-generated content may be incorrect.

The CPU is allocated and released multiple times as the process executes.  
A process in the running state may leave it for one of the following reasons:

* The process has completed execution and no longer needs CPU time.
* The process is waiting for an event (such as I/O).
* The process has used up its CPU time slice and must return to the ready queue.

Transitioning a process from waiting back to ready means it is again eligible to compete for CPU allocation.

1.2.3 Concept of CPU Scheduling

Because processes often switch between states, the system must manage these transitions efficiently.  
The CPU scheduler organizes processes in queues, assigns priorities, and decides which process should receive CPU next.

This is illustrated as follows:

A diagram of a process

AI-generated content may be incorrect.

In short, CPU scheduling means managing ready processes in a queue and allocating CPU time based on process priority to achieve the best CPU utilization.

Each process in the ready state is assigned a priority order, determined by factors such as process creation time, estimated CPU burst time, and completion deadlines.

1.3 SCHEDULING ALGORITHMS

1.3.1 First Come First Served (FCFS)

In this algorithm, the process that arrives first is served first — similar to the FIFO (First In, First Out) rule.  
Each process in the queue gets CPU service in the order they arrive until it finishes or is blocked.

A diagram of a computer

AI-generated content may be incorrect.

Advantages:

* Simple and easy to implement.
* CPU time is allocated without interruption (non-preemptive), so there is no overhead from context switching.

Disadvantages:

* The average waiting time can be very long if a long process runs before shorter ones (the convoy effect).
* Short processes may wait unnecessarily long.
* The system’s overall responsiveness is low for interactive users.

1.3.2 Round Robin (RR)

The Round Robin scheduling algorithm is specifically designed for time-sharing systems.  
It works similarly to FIFO but adds a time quantum (or time slice) that limits how long a process can hold the CPU before being preempted.  
Typical time quanta range from 10 to 100 milliseconds.

The ready queue is treated as a circular queue.  
The CPU scheduler assigns CPU time to each process in turn for one time slice.  
If a process finishes within that time, it leaves the queue; otherwise, it is placed back at the end of the ready queue.

This ensures all processes get a fair share of CPU time and prevents any single process from monopolizing the CPU.

A diagram of a list

AI-generated content may be incorrect.

Advantages:

* Each process is given a CPU time slice in turns, so the waiting time is reduced.
* Very effective for processes related to input/output (I/O) operations, which are frequently interrupted.
* Easy to implement, not too complicated.

Disadvantages:

* The average waiting time is usually longer compared to other scheduling policies.
* If the time quantum is too large, the RR algorithm becomes similar to FIFO.
* If the time quantum is too small compared to the execution time of a process, too much time will be wasted on context switching.
* The rule of determining the time quantum is important; if it exceeds 80% of the CPU cycle, efficiency decreases.

1.3.3 Shortest Job First (SJF)

Another approach to CPU scheduling is the Shortest Job First (SJF) algorithm. This algorithm associates each process with the length of its next CPU burst time. When the CPU becomes idle, it is assigned to the process that has the smallest next CPU burst.  
If two processes have the same CPU burst length, they are scheduled according to the FIFO (First In, First Out) rule.  
Although a more accurate term might be Shortest Next CPU Burst, we still commonly refer to this as SJF since it is widely used in textbooks and references.

Advantages:

* The algorithm is considered optimal because it minimizes the average waiting time.
* Maximizes CPU utilization.

Disadvantages:

* Complex to implement, requiring additional processing and management.
* Although optimal in theory, it cannot be implemented in real systems because the exact length of the next CPU burst is unknown.

The SJF algorithm can be either preemptive or non-preemptive, resulting in different variants. Whether it is optimal or not depends on whether preemption is allowed.

1.3.4 Shortest Remaining Time (SRT)

Similar to SJF, this scheduling algorithm gives higher priority to the process that has the shortest remaining execution time.  
It determines priority based on the estimated time required to complete each process (the total CPU burst time minus the time already used).  
Therefore, this algorithm must frequently update information about the remaining execution time of each process.  
Moreover, the CPU allocation policy must be handled properly to avoid loss of fairness among processes.

Advantages:

* Minimizes waiting time; the system remains efficient when most processes are short.
* Produces the most optimal scheduling results.

Disadvantages:

* Implementation is complex and requires frequent updates.
* Requires strict management to control process coordination.
* Must constantly track the remaining time of each process.