Operating System Course Report - First Half of the Semester

B class

October 1, 2024

Contents

1 Introduction **3**

2 Course Overview **3**
2.1 Objectives 3
2.2 Course Structure 3

3 Topics Covered **4**
3.1 Basic Concepts and Components of Computer Systems 4
3.2 System Performance and Metrics 4
3.3 System Architecture of Computer Systems 4
3.4 Process Description and Control 4
3.5 Scheduling Algorithms 5
3.5.1 First-Come, First-Served (FCFS) 5
3.5.2 Shortest Job Next (SJN) 5
3.5.3 Round Robin (RR) 8
3.6 Process Creation and Termination 8
3.7 Introduction to Threads 8
3.8 File Systems 8
3.9 Input and Output Management 9
3.10 Deadlock Introduction and Prevention 9
3.11 User Interface Management 9
3.12 Virtualization in Operating Systems 9

4 Assignments and Practical Work **10**
4.1 Assignment 1: Process Scheduling 10
4.2 Assignment 2: Deadlock Handling 10
4.3 Assignment 3: Multithreading and Amdahl's Law 10
4.4 Assignment 4: Simple Command-Line Interface (CLI) for User Interface Management 10
4.5 Assignment 5: File System Access 10

5 Conclusion **11**

1 Introduction

This report summarizes the topics covered during the first half of the Operating System course. It includes theoretical concepts, practical implementations, and assignments. The course focuses on the fundamentals of operating systems, including system architecture, process management, CPU scheduling, and deadlock handling.

2 Course Overview

2.1 Objectives

The main objectives of this course are:

- To understand the basic components and architecture of a computer system.
- To learn process management, scheduling, and inter-process communication.
- To explore file systems, input/output management, and virtualization.
- To study the prevention and handling of deadlocks in operating systems.

2.2 Course Structure

The course is divided into two halves. This report focuses on the first half, which covers:

- Basic Concepts and Components of Computer Systems
- System Performance and Metrics
- System Architecture of Computer Systems
- Process Description and Control
- Scheduling Algorithms
- Process Creation and Termination

- Introduction to Threads
- File Systems
- Input and Output Management
- Deadlock Introduction and Prevention
- User Interface Management
- Virtualization in Operating Systems

3 Topics Covered

3.1 Basic Concepts and Components of Computer Systems

This section explains the fundamental components that make up a computer system, including the CPU, memory, storage, and input/output devices.

3.2 System Performance and Metrics

This section introduces various system performance metrics used to measure the efficiency of a computer system, including throughput, response time, and utilization.

3.3 System Architecture of Computer Systems

Describes the architecture of modern computer systems, focusing on the interaction between hardware and the operating system.

3.4 Process Description and Control

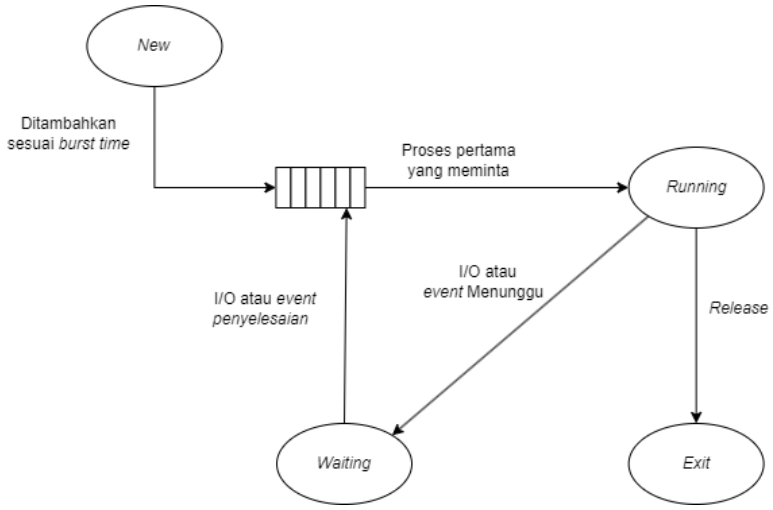
Processes are a central concept in operating systems. This section covers:

- Process states and state transitions
- Process control block (PCB)
- Context switching

3.5 Scheduling Algorithms

3.5.1 First-Come, First-Served (FCFS)

3.5.2 Shortest Job Next (SJN)



Dalam algoritma *Shortest Job Next* (SJN), saat memilih proses berikutnya yang akan dijalankan, kita melihat semua proses yang berada dalam keadaan siap dan memilih yang memiliki waktu eksekusi paling singkat. Dalam hal ini, kita perlu mengetahui waktu eksekusi setiap proses pada titik waktu tertentu. SJN adalah algoritma *non-preemptive*, artinya proses baru tidak akan mendapatkan giliran di CPU sampai proses yang sedang berjalan selesai, meskipun proses baru memiliki waktu eksekusi yang lebih singkat.

Dalam algoritma Shortest Job Next (SJN), beberapa istilah kunci perlu dipahami untuk memahami cara kerja algoritma ini. Berikut adalah istilah-istilah yang relevan:

- *Service time* adalah jumlah waktu yang dibutuhkan oleh sebuah proses atau job untuk menyelesaikan eksekusinya setelah mulai berjalan di CPU. *Service time* juga dikenal sebagai burst time.
- *Arrival time* adalah waktu ketika sebuah proses tiba dan siap untuk dieksekusi di CPU atau antrian penjadwalan.
- *Completion time* adalah titik waktu ketika sebuah proses menyelesaikan eksekusinya.

- *Turnaround time* adalah total waktu yang dibutuhkan oleh sebuah proses untuk menyelesaikan eksekusinya, dari saat proses tersebut tiba di antrian *ready* hingga selesai dieksekusi. *Turnaround time* dihitung sebagai *completion time* dikurangi *arrival time*.

- Algoritma penjadwalan *non-preemptive* adalah algoritma di mana sebuah proses yang telah mulai dieksekusi akan terus berjalan sampai selesai. Proses baru tidak diizinkan untuk mengganggu proses yang sedang berjalan.

Penggunaan algoritma SJN memiliki beberapa keuntungan yang signifikan. Berikut adalah beberapa keuntungan utama dari SJN:

1. Penjadwalan SJN meminimalkan waktu tunggu rata-rata untuk proses di antrian *ready* karena proses yang lebih singkat diberi prioritas lebih tinggi, sehingga mengurangi waktu tunggu mereka dan meningkatkan kinerja sistem dalam hal waktu respons dan penyelesaian (*turnaround time*).

2. Penggunaan algoritma *Shortest Job Next* (SJN) dapat meningkatkan pemanfaatan sumber daya CPU karena algoritma ini memprioritaskan proses yang lebih pendek. Hal ini memungkinkan lebih banyak proses untuk diselesaikan dengan lebih cepat.

Meskipun SJN memiliki keuntungan, terdapat juga beberapa kerugian yang perlu diperhatikan. Berikut adalah beberapa kerugian utama dari algoritma SJN:

- *Starvation* dapat menjadi masalah dalam penggunaan algoritma SJN. Dalam situasi ini, proses dengan durasi lebih lama mungkin tidak pernah mendapatkan kesempatan untuk dieksekusi jika ada kedatangan proses yang lebih pendek secara terus-menerus. Ini bisa menjadi masalah terutama dalam sistem yang mengutamakan keadilan.

- Memperkirakan durasi proses dengan akurat bisa menjadi tugas yang sulit dalam praktiknya. Jika perkiraan durasi proses ternyata tidak akurat, hal ini bisa memengaruhi kinerja SJN dan menyebabkan preemption yang sering.

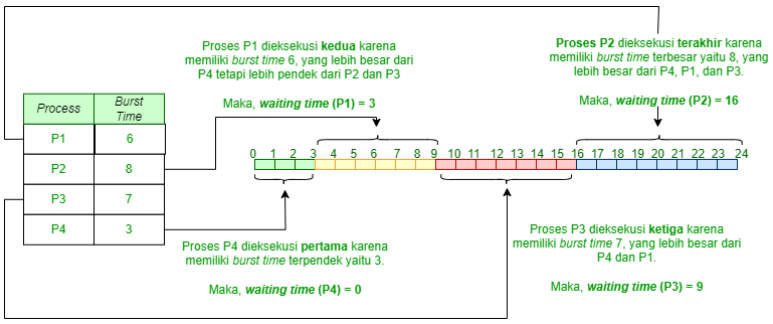
- Dalam situasi di mana respons cepat sangat penting, SJN mungkin bukan pilihan yang paling responsif untuk sistem interaktif atau lingkungan *real-time*. Ketika proses singkat mendominasi CPU, proses yang lebih panjang bisa mengalami penundaan yang lama, yang sering kali merugikan.

Algoritma SJN memiliki karakteristik unik yang membedakannya dari algoritma penjadwalan lainnya. Berikut adalah beberapa karakteristik dari SJN:

- *Shortest Job Next* memiliki keuntungan berupa waktu tunggu rata-rata yang minimum dibandingkan dengan semua algoritma penjadwalan lainnya.
- Algoritma ini merupakan *Greedy Algorithm*, yang berarti mengambil keputusan yang terbaik saat ini tanpa mempertimbangkan konsekuensi di masa depan.
- Algoritma ini dapat menyebabkan *starvation* jika proses yang lebih pendek terus menerus muncul. Masalah ini dapat diatasi dengan menggunakan konsep *ageing*, yaitu memberikan prioritas kepada proses yang telah menunggu lebih lama.
- Dalam praktiknya, SJN sulit diterapkan karena sistem operasi mungkin tidak mengetahui *burst time* dan karena itu tidak dapat mengurutkannya. Meskipun tidak mungkin untuk memprediksi waktu eksekusi dengan akurat, beberapa metode dapat digunakan untuk memperkirakan waktu eksekusi suatu *job*, seperti menggunakan rata-rata tertimbang dari waktu eksekusi sebelumnya.
- SJN dapat digunakan di lingkungan khusus di mana perkiraan waktu *running* yang akurat tersedia.

Berikut adalah langkah-langkah dalam algoritma *Shortest Job Next* (SJN):

1. Urutkan semua proses berdasarkan *arrival time*.
2. Pilih proses yang memiliki *arrival time* terkecil dan *burst time* terkecil.
3. Setelah proses tersebut selesai, buat kumpulan proses (*pool*) yang tiba setelahnya hingga proses sebelumnya selesai.
4. Dari kumpulan tersebut, pilih proses yang memiliki *burst time* terkecil untuk dieksekusi selanjutnya.



3.5.3 Round Robin (RR)

3.6 Process Creation and Termination

Details how processes are created and terminated by the operating system, including:

- Process spawning
- Process termination conditions

3.7 Introduction to Threads

This section introduces the concept of threads and their relation to processes, covering:

- Single-threaded vs. multi-threaded processes
- Benefits of multithreading

3.8 File Systems

File systems provide a way for the operating system to store, retrieve, and manage data. This section explains:

- File system structure
- File access methods
- Directory management

3.9 Input and Output Management

Input and output management is key for handling the interaction between the system and external devices. This section includes:

- Device drivers
- I/O scheduling

3.10 Deadlock Introduction and Prevention

Explores the concept of deadlocks and methods for preventing them:

- Deadlock conditions
- Deadlock prevention techniques

3.11 User Interface Management

This section discusses the role of the operating system in managing the user interface. Topics covered include:

- Graphical User Interface (GUI)
- Command-Line Interface (CLI)
- Interaction between the user and the operating system

3.12 Virtualization in Operating Systems

Virtualization allows multiple operating systems to run concurrently on a single physical machine. This section explores:

- Concept of virtualization
- Hypervisors and their types
- Benefits of virtualization in modern computing

4 Assignments and Practical Work

4.1 Assignment 1: Process Scheduling

Students were tasked with implementing various process scheduling algorithms (e.g., FCFS, SJN, and RR) and comparing their performance under different conditions.

4.2 Assignment 2: Deadlock Handling

In this assignment, students were asked to simulate different deadlock scenarios and explore various prevention methods.

4.3 Assignment 3: Multithreading and Amdahl's Law

This assignment involved designing a multithreading scenario to solve a computationally intensive problem. Students then applied **Amdahl's Law** to calculate the theoretical speedup of the program as the number of threads increased.

4.4 Assignment 4: Simple Command-Line Interface (CLI) for User Interface Management

Students were tasked with creating a simple **CLI** for user interface management. The CLI should support basic commands such as file manipulation (creating, listing, and deleting files), process management, and system status reporting.

4.5 Assignment 5: File System Access

In this assignment, students implemented file system access routines, including:

- File creation and deletion
- Reading from and writing to files
- Navigating directories and managing file permissions

5 Conclusion

The first half of the course introduced core operating system concepts, including process management, scheduling, multithreading, and file system access. These topics provided a foundation for more advanced topics to be covered in the second half of the course.