Operating System Course Report - First Half of the Semester

B class

October 1, 2024

Contents

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

This section covers:

- First-Come, First-Served (FCFS)
- Shortest Job Next (SJN)
- Round Robin (RR)

It explains how these algorithms are used to allocate CPU time to processes.

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

Seperti yang terlihat pada Gambar **??**, inilah cara menambahkan gambar dengan keterangan.

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

3.10.1 Deadlock prevention

Pencegahan terhadap deadlock adalah teknik yang digunakan untuk mencegah sistem memasuki keadaan deadlock. Berbeda dengan deadlock avoidance yang dimana mengatasi masalah setelah deadlock terjadi, pencegahan ada agar memperkecil kemungkinan terjadinya deadlock. Pencegahan deadlock dapat dilakukan dengan cara menghindari keadaan-keadaan yang dapat menyebabkan deadlock. Beberapa metode pencegahan deadlock adalah:

1. Resources Ordering

   Resources Ordering adalah pengurutan sumber daya dimana setiap sumber daya dalam sistem diberi nomor atau urutan unik. Proses harus meminta sumber daya dalam urutan yang konsisten dengan nomor tersebut. Artinya, jika suatu proses sudah memegang satu sumber daya dengan nomor lebih rendah, maka ia hanya boleh meminta sumber daya dengan nomor lebih tinggi.

   Salah satu contoh kasus pencegahan deadlock dengan metode resources ordering yaitu jika ada dua sumber daya, R1 dan R2, dan proses P1 memegang R1 dan ingin meminta R2, maka metode ini memastikan bahwa proses P1 tidak dapat memegang R2 tanpa mengikuti urutan yang telah ditentukan.

2. Resource allocation denial

   Metode ini dikenal juga sebagai Banker's Algorithm atau safety algorithm, di mana sistem memeriksa apakah pemberian sumber daya tertentu akan menyebabkan kondisi unsafe atau potensi deadlock. Jika pemberian sumber daya dianggap tidak aman, maka permintaan sumber daya tersebut akan ditolak.

   Metode resources allocation denial adalah ketika proses P1 meminta sumber daya R1. Sebelum mengalokasikan sumber daya, sistem mengevaluasi apakah proses lain bisa menyelesaikan eksekusi dengan sumber daya yang tersisa. Jika tidak, permintaan P1 ditolak sementara waktu dan akan dilanjutkan ketika semua sumberdaya yang dibutuhkan sudah lengkap.

3. Timeouts

   Dalam metode ini, sistem menggunakan batas waktu (timeout) untuk menunggu sumber daya. Jika proses menunggu terlalu lama, ia akan dipaksa gagal atau restart. Timeout dapat digunakan untuk mencegah proses berlama-lama dalam kondisi menunggu, sehingga menghindari deadlock.

   Proses P1 sedang menunggu sumber daya R1 selama 5 menit. Jika setelah batas waktu ini R1 belum tersedia, proses P1 akan dibatalkan atau dijadwalkan ulang untuk mencoba lagi.

4. Avoidance of hold-and-wait conditions

   Teknik avoidance of hold-and-wait conditions adalah teknik pencegahan deadlock dengan kondisi saling tunggu sumber daya antara suatu proses. Dalam teknik ini, proses tidak diperbolehkan menahan sumber daya sambil menunggu sumber daya lain. Proses harus meminta semua sumber daya yang dibutuhkan di awal (sebelum memulai eksekusi) atau melepaskan sumber daya yang sedang dipegang sebelum meminta sumber daya tambahan.

   Dalam penerapannya, Proses P1 harus meminta sumber daya R1 dan R2 sekaligus. Jika R2 tidak tersedia, maka proses tidak diperbolehkan memegang R1 dan harus melepaskannya sampai kedua sumber daya tersedia. Setiap proses yang belum memiliki sumber daya yang cukup untuk berjalan maka akan terus menunggu hingga sumber daya yang dibutuhkan terpenuhi semua.

5. Resource preemption

   Teknik ini memungkinkan sistem untuk merebut atau mengambil kembali sumber daya dari proses yang sedang berjalan jika diperlukan oleh proses lain yang lebih prioritas atau jika diperlukan untuk mencegah deadlock.

   Dalam kasus seperti proses P1 harus meminta sumber daya R1 dan R2 sekaligus. Jika R2 tidak tersedia, proses tidak diperbolehkan memegang R1 dan harus melepaskannya sampai kedua sumber daya tersedia.

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.

4.1.1 Group 1

```
    class Process:
    def __init__(self, pid, arrival_time, burst_time):
    self.pid = pid
    self.arrival_time = arrival_time
    self.burst_time = burst_time
    self.completion_time = 0
    self.turnaround_time = 0
    self.waiting_time = 0
```

Header 1	Header 2	Header 3
Row 1, Column 1	Row 1, Column 2	Row 1, Column 3
Row 2, Column 1	Row 2, Column 2	Row 2, Column 3

Table 1: Your table caption

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.