Memory Management Unit Simulator Report

Omar Alfarouq Bouhadi - Basma Arnaoui

May 16, 2024

Contents

1 Introduction

The Memory Management Unit (MMU) Simulator is a versatile application designed to illustrate different memory allocation strategies used in operating systems. This tool allows users to interactively simulate the process of allocating and deallocating memory using various strategies such as First Fit, Next Fit, Best Fit, and Worst Fit, providing insights into their operational efficiencies and limitations.

2 Project Overview

2.1 Purpose

The primary purpose of this simulator is to provide an educational tool that helps students and professionals alike understand the complexities involved in dynamic memory management in operating systems.

2.2 Features

- Multiple allocation strategies including First Fit, Next Fit, Best Fit, and Worst Fit.
- An interactive command-line interface for real-time memory management simulation.
- A user-friendly web interface for visual interaction with the memory management system.
- Real-time visualization of memory allocation and deallocation.

3 Technical Details

3.1 Implementation

The simulator is implemented using Python, taking advantage of its powerful libraries to manage the backend logic and Flask to handle the web interface.

3.2 File Structure

```
- memory_manager.py: Contains the core logic for memory management strategies.
- mmu.py: The main entry point of the CLI-based application.
- routes.py: Manages the web interface routing and Flask setup.
```

4 How the App Works

4.1 Command-Line Interface

The CLI allows users to start the simulation by specifying the total memory and the strategy. Users can then execute commands to allocate and deallocate memory, simulate processes, and print the memory map.

4.2 Web Interface

The web interface provides a graphical user interface to interact with the MMU simulator. It allows users to perform all operations available in the CLI in a more intuitive way.

5 Usage

To run the CLI application:

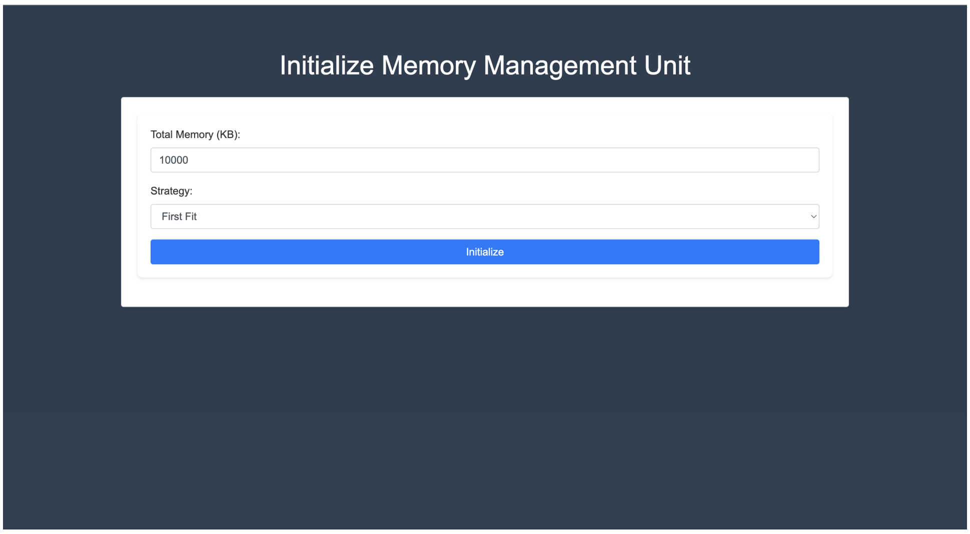
```
python mmu.py <total_memory_in_KB> <strategy_number>
```

To start the web interface:

```
python routes.py
```

Access the web interface at `http://localhost:5000`.

6 Screenshots

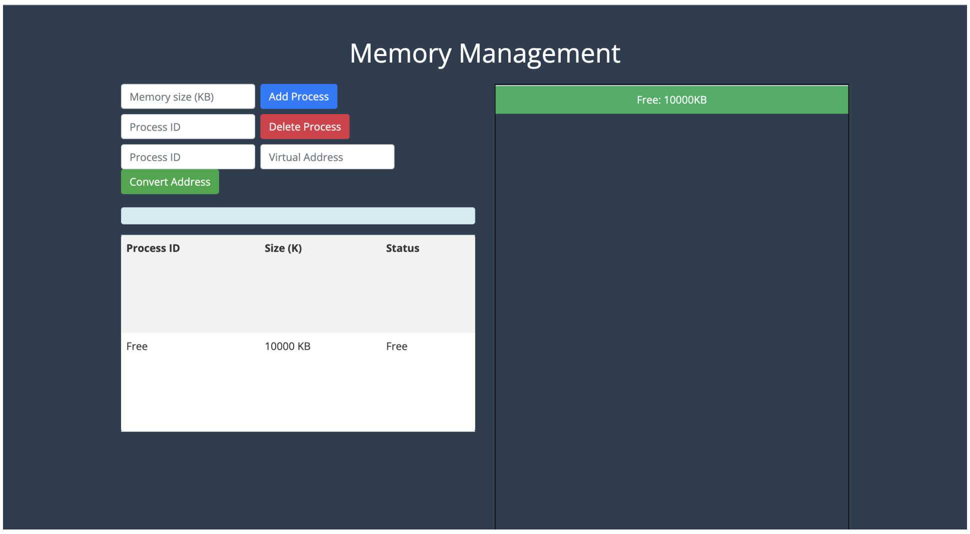


Initialize Memory Management Unit

Total Memory (KB): 10000

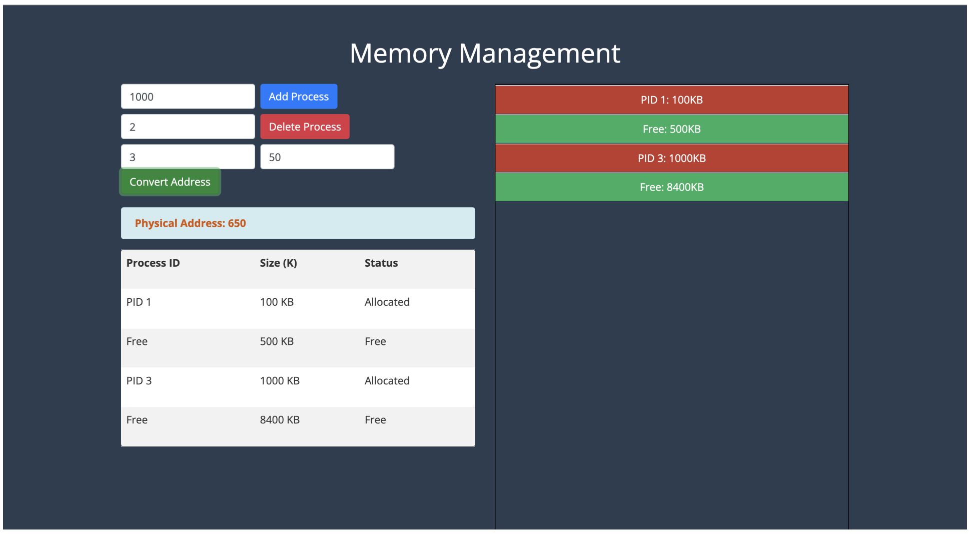
Strategy: First Fit

Figure 1: Landing Page - Choice of Total Memory Size + Algorithm



Process ID	Size (K)	Status
Free	10000 KB	Free

Figure 2: Memory Visualization



1000

2

3

50

Physical Address: 650

Process ID	Size (K)	Status
PID 1	100 KB	Allocated
Free	500 KB	Free
PID 3	1000 KB	Allocated
Free	8400 KB	Free

Figure 3: Simulation of First Fit algorithm