Ho Chi Minh University of Technology

Falcuty of Computer Science and Engineering
Course: System Evaluation System (CO3007)

System Evaluation System Project:

CPU scheduler and memory allocation performance evaluation

Lecturer: Tran Van Hoai
Student: Phan Minh Toan - 1852798
Tran Nhut Quang - 1852037
Trinh Minh Trung - 1852825

Ho Chi Minh, March 2022

Contents

List of Figures

Listings

1 Introduction

The Project help to understand the scheduling algorithms and the operations of contiguous memory allocation in Operating System. We can get more knowledge about the principle of each scheduling algorithm in practicing. Through that, we can clearly understand the idea of each allocation algorithm in memory and the concept of fragmentation as well as using the most suitable algorithm in many cases.

2 Step in Performance Evaluation Project

In this assignment, the first 5 steps (out of 10 steps) of evaluating performance of a project will be walked through. The first 5 steps are:

- Step 1: Stating goals and defining the system.
- Step 2: List services and outcomes.
- Step 3: Selecting Metrics.
- Step 4: List Parameters.
- Step 5: List Factors to study.

The first step is to describe precisely the system boundaries by stating goals. Specific study goals can affect the definition of a system. Then, in second step, set of services of a provided system will be listed as well as all of its possible outcomes. In step 3, Metrics will be selected from all possible outcomes. These metrics are related to speed, accuracy and availability of services. In the fourth step, all possible parameters that affect the performance of the system will be listed, system parameters and workload parameters. In the fifth step, factor list and per-factor levels will be gradually extended. For this assignment, the first 5 steps in Performance Evaluation of **CPU Scheduling** and **Memory Allocation** will go under study to give out what case is the most suitable in specific occasion and how it affects the performance of the system.

2.1 CPU Scheduling

2.1.1 State goals and define the system

Goals of CPU Scheduling:

- To estimate the interval from the time of submission of a process to the time of completion of each scheduling algorithms, turnaround time.
- To estimate the waiting time of a process in the ready queue of each scheduling algorithms, waiting time.
- To estimate the response time of each process from the submission of a process.
- To maximize the throughput.
- To minimize waiting time, response time, turnaround time.
- To define which scheduling algorithms among **First in, First out (FIFO)**, **Shortest job first (SJF)**, **Priority Scheduling** and **Round-Robin Scheduling**, is the most suitable under each circumstances.
- To define the CPU Utilization.

Define the system:

- Define the algorithm the system uses.
- Define the processing speed of the CPUs.
- Define how many CPUs the system have.

2.1.2 List of services and outcomes

List of services:

- To schedule jobs/processes for CPU Execution.
- CPU Queue handling the process.

Outcomes:

- Processes have to wait a long time before finishing executing or Processes have long response time.
- Processes can not be finished under scheduling.
- Processes have long waiting time before executing.
- All of Processes finishes under a reasonable amount of time.
- Processes response immediately.
- All the CPU have a process in their queue.

2.1.3 Selected Metrics

Selected Metrics of the system:

- CPU Utilizes, rate of CPU under no service is low.
- The responsiveness of processes is fast.
- Time for a process to wait under ready queue is short.
- Rate of an uncompleted process is low.
- The turn around time of processes.
- The number of times that context switch happens.

2.1.4 List of Study Parameters

System Parameters:

- Speed of local CPUs.
- Speed of program handling the scheduling.

Workload Parameters:

- Number of processes that are under scheduling.
- Amount of CPUs handling process.
- Scheduling Algorithm.
- Quantum time in specific scheduling.
- **Round-Robin Scheduling**.
- Number of context switch.

2.1.5 List of Factors to study

List Factors to study of CPU Scheduling:

- Number of processes that are under scheduling.
- Quantum time in specific scheduling **Round-Robin Scheduling**.
- Scheduling Algorithm in specific cases and the most suitable case.

2.2 Memory Allocation

2.2.1 State goals and define the system

Goals:

- To allocate a memory slot that large enough for the process.
- To satisfy a request of size from a list of free memory holes.
- To solve the problem of the general dynamic storage allocation problem.
- To decide which memory allocation strategies from **First Fit**, **Best Fit** and **Worst Fit** are most commonly used to select a free hole from the set of available holes.
- To optimize available memory slot for other process.

Define the system:

- To define the size available of the memory.
- To define the process of allocating, de-allocating memory.

2.2.2 List of services and outcomes

List of services:

- To send the memory to perform a process to specific location in memory.
- To get the memory contains specific process.

Outcomes:

- All the memory are located perfectly.
- A part of memory that need located can not be allocated.
- All the memory can not be allocated.
- Memories in memory slot can be taken correctly.
- Some memories in memory slot are taken mistakenly.
- Can not define a specific allocated memory in memory pool.

2.2.3 Selected Metrics

Selected Metrics of Memory Allocation:

- Time to handle/allocate the memory of the process.
- Rate of impossibility to allocate a memory.
- Time to de-allocate the memory of the process.
- Memory Proficiency.

2.2.4 List of Study Parameters

System Parameters:

- Speed to allocate/de-allocate a memory of a process.
- Speed of a memory registering an address.
- Size of given memory for allocation.

Workload Parameters:

- Memory size of a process.

2.2.5 List of Factors to study

List Factors to study of Memory Allocation:

- Size of given memory to handle memory allocation. Example memory size: 64Kb, 128Kb, 256Kb, 512Kb.
- Memory size needed for a given process.
- Memory Allocation Algorithm in specific cases and the most suitable case.

3 System performance Evaluation

3.1 Scheduler

We have run the scheduler with around 90 inputs with **quantum time** of 1, 2, 3, 5, and 8 to collect this statistic.

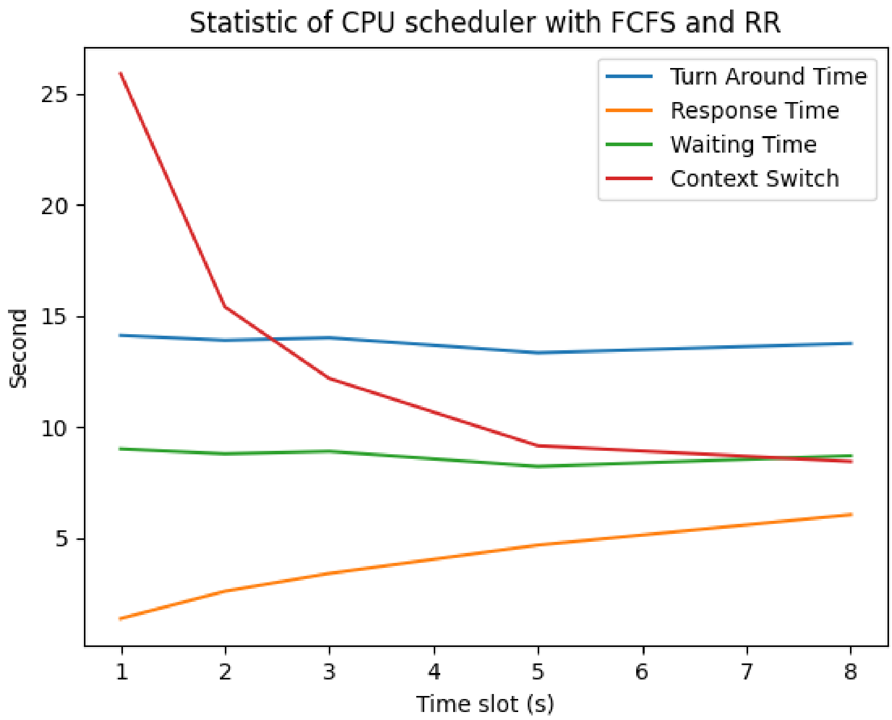

Figure 1: Statistic of CPU scheduler with FCFS and Round Robin

According to the figure Fig1, with small quantum time such as 1 or 2, the average **response time** is very fast. However, the number of **context switch** is very high. Assuming that each context switch takes one second, in general, we must need 25 additional second for the **context switch**.
We can improve the performance by increase the **quantum time** to decrease the number of **context switch**. Nevertheless, there is a consequence that the responsiveness of of each process would be much lower because the **response time** is increased now.
Therefore, based on the Fig1, we can balance between **context switch** and **response time** to decrease over-head time and meet system's requirements.

3.2 Memory Allocation

3.2.1 Time Consumption

In this aspect, we can see that the **best fit** algorithm need more time to find a memory segment to allocate compare to **first fit** approach.

3.2.2 Allocation Efficiency

In this section, the statistic is collected when we ran around 90 inputs with the total memory size is 128 bytes. The average number of processes in each input is 7 processes and each of them need average of 27 bytes to be executed.

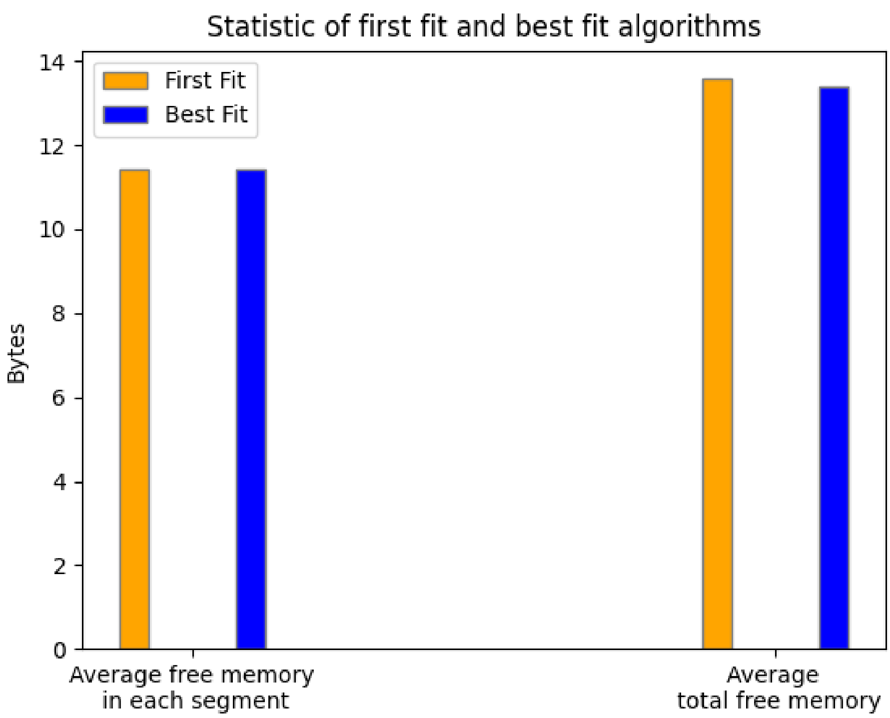

Figure 2: Statistic of memory allocation with First fit and Best fit algorithms

With the figure Fig2, each time that we cannot allocate memory, the average free memory in each segment is roughly equal with 2 algorithms. However, the average total free memory of **first** is a bit higher. This means that the **best fit** approach can make use of free memory segments in a better way.

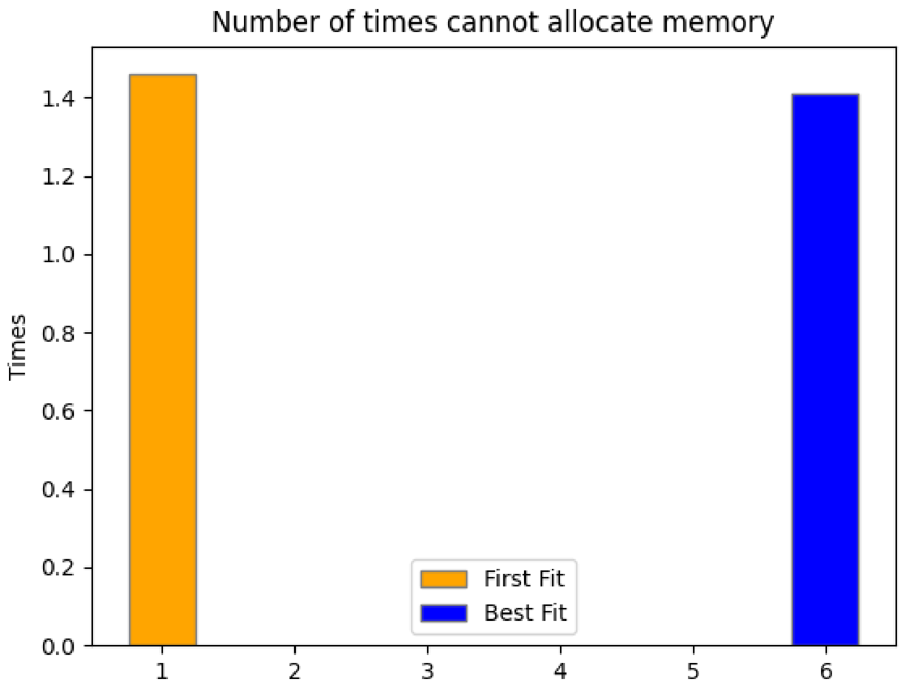

Figure 3: Number of times that cannot allocate memory for processes

In the figure Fig3, the number of times that we cannot allocate memory for processes of **First fit** is also slightly higher. Therefore, in our experiments, with the same size of memory and the same number of memory requests, the **Best fit** algorithm has a better performance compare to the [First fit] approach.

3.3 Source code

The whole source code, inputs and statistic of our work is in this link. https://github.com/quang3103/SPE_PROJ1