IN THE NAME OF ALLAH

Operating Systems

SEYED MAHDI MAHDAVI MORTAZAVI III 40030490

FINAL PROJECT – THE ELEVATOR

WINTER 2024

Introduction

This is about implementing an elevator, with some preemptive and non-preemptive scheduling algorithm; FCFS and SJF (Shortest job first) as PREEMPTIVE algorithms and Round-Robin and SRTF (Shortest-remaining time first) as NON-PREEMPTIVE algorithms;

In this project, I implemented this four scheduling algorithms with Javascript and Python; Using GUI and TUI both; GUI for FCFS and Shortest-Job first, TUI for Round Robin and SRTF (of course they may not work :*) ...

key points of non-preemptive (and also it can be for preemptive) algorithms is **saving them in a queue to response them when they need (as noted above, also it (queue) can be used for preemptive algorithms;**

Non-Preemptive algorithms

- for **FCFS** algorithms we don't need to do any special things; it is only need to implement a queue to put passengers (as processes) in it to response their requests …
  We response their requests, in any order that they arrived into the queue; (Principle of FCFS scheduling algorithm :); FIRST COME, FIRST SERVE …
- To implement the **Shortest-job first (SJF)** algorithm, we can use a same queue (like FCFS), to sort passengers (as our processors), in that queue and using a benchmark to choose them (like a *GREEDY* choice ☺ in algorithm course based on their delay (input time – arrival time)). With sorting passengers, we can response them, iff they have lower time to response; in the Elevator, we can sort them in a queue and response them from the first index of this queue (array of passengers);

Preemptive algorithms

- Maybe the main different between preemptive and non-preemptive scheduling algorithms, is their generously (to get CPU to the other processes) ☺ …
- To implement **Round-Robin (RR)** algorithm, we can consider a queue (here we go again ;) (noted in previous slides) to put the passengers (processes) in this queue based on their arrival time (like FCFS) or sort them based on their total execution time (like SJF) and use them (actually get PC to them) in a certain time (that we call Quantum time); we can have an iteration on this queue and run the processes (get elevator to passengers) in a Quantum time (and continue this strategy to response all passengers in different floors (equal time for each passenger (process) to totally response them (until they be terminated).

Preemptive algorithms

- Round Robin is the fairest algorithm ….
- To implement the **Shortest-remaining-time first** (SRTF → that is the preemptive model of SJF); Unlike SJF, I think using a *sorted queue*, cannot have a significant benefit for us; Why? Because we want to choose passengers (processes) to response them when they comes with a lower remaining time (to replace with the current process that is using CPU now); we can consider that normal (unsorted) queue and have an iteration on this queue to response them it order that they arrived; BUT selected passenger (process) must have the minimum remaining time, at once that it arrives; Of course using a sorted array may be good (this is just my idea ☺).

The END

SEYED MAHDI MAHDAVI MORTAZAVI | 40030490

FINAL PROJECT – THE ELEVATOR

WINTER 2024