Simulation of Schedulers

February 5, 2015

In this assignment, we will simulate a scheduler.

1 Input files

You MUST use exactly the formats described here, because we will be using an automated script for evaluation.
Two files named PROCESS_SPEC and SCHEDULER_SPEC will be used.

PROCESS_SPEC would contain specification of all processes that are to be run under the scheduler. The specification of each process is as follows:

```
PROCESS
(process id,starting priority,admission time)
(#iterations,CPU burst,Duration of I/O)
(#iterations,CPU burst,Duration of I/O)
.....
(#iterations,CPU burst,Duration of I/O)
END...
```

Where process id and starting priority are both integers, with a larger number implying a higher priority, and all times are in milli-seconds. A process contains several phases in its lifetime. Each phase is described by a triple giving the number of iterations and details of the CPU burst and duration of I/O operation in each iteration. A process may contain any number of phases, and any number of processes can be admitted in the system.

SCHEDULER_SPEC would contain

```
SCHEDULER
# of scheduling levels
```

```
(scheduling level,priority,time slice)
(scheduling level,priority,time slice)
.....
(scheduling level,priority,time slice)
END
```

where scheduling levels are integers starting with 1, priority is an integer, and time slice is either an integer indicating milliseconds or the character “N” indicating that there is no time slice associated with that scheduling level.

2 Simulator schematic

The schematic is given in a separate pdf file. You are supposed to analyze and understand the schematic and do the design accordingly.

The basic idea is that there is a clock. The clock is advanced to the next significant event that occurs in the system. Various parts of the set-up decide which events will occur when and they put the information in the Event table. The event manager picks the next event. This is how the simulator runs.

The code which is provided to you contains implementation of three parts of the schematic:

- **Process information table :** function **handling_PROCESS_SPEC_file()** processes the process specification input file i.e PROCESS_SPEC and stores all the data in the structure defined for a process in the given code.

- **Scheduler information table :** function **handling_SCHEDULER_SPEC_file()** processes the scheduler specification input file i.e SCHEDULER_SPEC and stores all the data in the structure defined for a scheduler in the given code.

- **Event Manager :** The basic structure of event manager is defined in the file ”event_manager.h”. The event manager picks the next event from the event table.
  The entries of event table has following 3 attributes

  - event occurrence time

- Event type
- Process_id

In the given code “type” field is of integer type.
Here “event type” stands for different types of events that may occur during execution of the processes. The events are named below:

- Process admission
- End of time-slice
- IO start
- IO end

NOTE :
1. You can make changes in the schematic or use or develop your own schematic. You can also add more attributes to the event table if needed.
2. In the code, “event type” field is of integer type. You must change this type to a non-integer type of your own choice.

3 Scheduler report

The scheduler should print an information summary of its operation containing atleast following kind of informations.

```
PID :: 1  TIME :: 3  EVENT :: Process Admitted
PID :: 1  TIME :: 3  EVENT :: Process dispatched
PID :: 2  TIME :: 4  EVENT :: Process Admitted
PID :: 1  TIME :: 5  EVENT :: CPU burst completed
PID :: 1  TIME :: 5  EVENT :: IO started
PID :: 2  TIME :: 5  EVENT :: Process dispatched
PID :: 2  TIME :: 8  EVENT :: CPU burst completed
.....
.....
```

4 Originality of design and implementation

Any similar designs or code will be taken to be instances of dishonesty. You are advised to use “original” names to avoid this possibility.

5 Schedulers you should implement:

You will implement the simulator for a variety of schedulers. To begin with, you should implement the following 3 schedulers.

- **A multiprogramming scheduler**
- **A time-sharing scheduler using simple round robin scheduling:** This scheduler will have only one scheduling level.
- **A multi-level scheduler with promotions and demotions:** This scheduler will have many scheduling levels. It works as follows:

  a. Each scheduling level has a different priority and a different time slice.
  b. A higher priority level has a smaller time slice and a lower priority level has a larger time slice.
  c. Many processes may have the same priority. Such processes would exist at the same priority level and would be scheduled in a round-robin manner.
  d. Processes at a scheduling level would be considered for scheduling only if no ready processes exist at scheduling levels with higher priorities.
  e. To begin with, a process will have the priority given in its specification.
  f. **Demotion:** A process is demoted to the next lower priority level (if one exists) if, when scheduled, it uses up the time slice completely, i.e., it does not start an I/O operation before the time slice elapses.
  g. **Promotion:** A process is promoted to the next higher priority level (if one exists) if, when scheduled, it starts an I/O operation before the time slice elapses.

6 Submission

- Submit a single tar file containing a README file and code files.
- The README file should mention the event names being used and should have a log of output.
- You will submit only some specified schedulers at 5.00 pm and submit all of the specified schedulers at the specified deadline.

7 Right to fix errors and/or make changes is reserved.