COMP 530 Introduction to Operating Systems

Fall 2017  
Kevin Jeffay

Worksheet 5, September 18

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Your Name: |  | You worked with: |  | +1/blank/-1: |  |
|  | Aaron Zhang |  | Daniel Estrada |  | +1 |  |
|  |  |  | Jonathan Ng |  | +1 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

1. These question all concern Shortest-Job-First (SJF) scheduling.

*a*) As presented in lecture, was SJF scheduling a preemptive or a non-preemptive scheduling policy? In practice, wow would the execution of a non-preemptive SJF scheduling policy differ from a preemptive SJF scheduling policy?

Preemptive because at any point, the shortest job will be at the head of the job queue to be executed. For non-preemptive the program is not interrupted if there is a shorter job inserted in the queue. Difference is allowing current execution to occur or not.

The lecture example you can not tell if it is preemptive or non-preemptive because there are no examples of jobs arriving. If something arrives after a scheduling position in a preemptive scheduling policy, if there exists a shorter job a longer job will never be executed. Essentially the difference is that you can have arrivals with lower execution time in non-preemptive and the job may still execute.

*b*) Explain how you could use a priority scheduler to implement an SJF scheduling policy.

Use a dynamic priority scheduler and as new jobs come in sort them and bump jobs up based on their priority.

If runtimes can be mapped, then a scheduler can assign a priority with respect to the runtime. Can use a filter to estimate the priority’s can find on the slides… Dynamic and preemptive scheduler

2. What is the “problem” for which multi-level feedback scheduling is the solution?

Starvation, as it has bigger quantum sizes if a high priority job bubbles down.

The convoy effect is limited because the execution time for CPU bound jobs are limited. Round robin does not take into effect the time each quantum is. With multi-level feedback, execution time needs are taken into account. It also allows short jobs to be executed first.

3. These question all concern the real-time scheduling policies of rate-monotonic (RM) scheduling and earliest deadline first (EDF) scheduling.

*a*) Explain how RM and EDF scheduling policies can be implemented with a priority scheduler. Does the scheduler have to be preemptive or non-preemptive scheduling?

Dynamic priority with RM giving priority to the smallest execution time and EDF giving priority to earliest deadline. Both are preemptive

EDF wants to execute things nearest to you in time. Priority is when a job is released + its period, priority is a point in time. Effectively computing a point on the time and whichever is nearest gets highest priority. Dynamic scheduling scheme, with time increasing release time gets larger. Every jobs priority changes with each release.

Rate monotonic is just the jobs period, this is static.

*b*)Rate-monotonic scheduling tends to be a more popular and hence more widely used real-time scheduler even though EDF can provide better real-time guarantees (can accommodate workloads with higher processor utilization). Knowing what you now know about scheduling mechanisms, speculate why this is the case.

Rate-monotic can guarantee that jobs occur within their timing constraints, while an EDF can give priority to a process with a lower deadline but may fall outside of the time constraint. With streaming video and etc this could cause issues.