DEPARTMENT OF COMPUTER SCIENCE, HANYANG UNIVERSITY

MLFQ and Stride scheduling

Operating Systems: Three Easy Pieces

Programming Project with xv6

Due date: April 23, 2017

1 PROCESS SCHEDULER

CPU scheduling is the basis of multiprogrammed operating systems. The CPU scheduler chooses the best candidate among available processes, and this leads the operating system to make best use of resources efficiently, thus being more productive. There is an operating system module that selects the next jobs to be admitted into the system and the next process to run. We call the module a process scheduler.
In this project, our goal is improving xv6 process scheduler by implementing the process scheduler as a combination of multi-level feedback queue and stride scheduling method.

1.1 MULTILEVEL FEEDBACK QUEUE SCHEDULING

Unlike multilevel queue scheduling algorithm where processes are permanently assigned to a queue, multilevel feedback queue scheduling allows a process to move between queues. This movement is facilitated by the characteristic of the CPU burst of the process. If a process uses too much CPU time, it will be moved to a lower-priority queue. This scheme leaves I/O-bound and interactive processes in the higher priority queues. In addition, a process that waits too long in a lower-priority queue may be moved to a higher priority queue. This form of aging (or priority boosting) also helps to prevent starvation of certain lower priority processes.

1.2 STRIDE SCHEDULING

Stride scheduling is a deterministic resource proportional-share algorithm. By interpreting the stake of a resource using the concept of stride the algorithm overcomes unfairness of lottery algorithm which is the basic proportional share scheduling method.

2 PROJECT MILESTONES

1. **The First Milestone**. Design new scheduler with MLFQ and Stride on the abstraction level. Follow steps below, because it should be implemented incrementally.
   a) First step: MLFQ
      - 3-level feedback queue
      - Each level of queue adopts Round Robin policy with t, 2t, 4t time quantum respectively
      - To prevent starvation, priority boost is required
   b) Second step: Combine the *stride scheduling* algorithm with MLFQ
      - Make the system call (i.e. cpu_share) that requests the portion of CPU and guarantees the calling process to be allocated that CPU time.
      - Total stride processes are able to get at most 80% of CPU time. Exception handling is needed for exceeding request.
      - The rest 20% of CPU time should run for the MLFQ scheduling which is the default scheduling in this project

   - **Due Date**: April 4

2. **The Second Milestone**. Implement the newly designed scheduler. Observe its behaviour and make a report.

   IMPORTANT: Never proceed to this milestone unless your design gets confirmed by TAs or the professor.

   - For implementation detail, the additional specification will be uploaded soon.
   - **Due Date**: April 23

3 GENERAL REQUIREMENTS

1. Every documentation should be written in your Gitlab Wiki.
   - Wiki: documentation and explanation storage. For example, analysis report, design documents, background information, dev note, etc. can be here.
2. Follow the appropriate coding convention.
   - In xv6 kernel source code, follow the xv6 style.

- In your own source code, follow the HYU style.

3. Do brainstorming with your class mates via on/offline.
    - Piazza can be a good dev community. Share what you think with your friends.
4. Do NOT share the code.
    - Please keep the programming ethics.