國立陽明交通大學
NATIONAL YANG MING CHIAO TUNG UNIVERSITY

Institute of Artificial Intelligence Innovation
Department of Computer Science

Operating System

Homework 04: CPU Scheduling (part2)

Shuo-Han Chen (陳碩漢)
shch@nycu.edu.tw

Wed. 10:10 - 12:00 EC115 +
Fri. 11:10 – 12:00 Online

Goal

1. The default CPU scheduling algorithm of Nachos is a simple round-robin scheduler for every 100 ticks.
2. The goal of HW4 is to replace it with a priority scheduling strategy.
3. Cancelling time slicing to employ a new scheduling strategy

Part 2 Prerequisite

- Before you start implement this homework feature, you need to ensure your previous NachOS assignments meet the following requirements.
  1. Complete the requirements for Homework 2 to ensure that your Nachos system is capable of running multiple processes correctly.
  2. Please refer to the initial file ‘HW04_init’ for implementation details
  3. ‘HW04_init’ is for sure to function correctly for HW02

Part 2 Modification

- To observe scheduling easily by PrintInt(), change ConsoleTime to 1 in machine/stats.h

```
const int ConsoleTime = 1;
```

- Comment out postOffice at Kernel::Initialize() and Kernel::~Kernel() in kernel.cc

```
// postOfficeIn = new PostOfficeInput(10);
// postOfficeOut = new PostOfficeOutput(reliability);
// delete postOfficeIn;
// delete postOfficeOut;
```

Part 2 Modification (cont'd)

- add test case for better debugging.

hw4_consoleIO_1

```c
#include "syscall.h"

int
main()
{
    int n;

    for (n=0; n < 4; n++) {
        PrintInt(1);
    }
    return 0;
}
```

hw4_consoleIO_2

```c
#include "syscall.h"

int
main()
{
    int n;

    for (n=0; n < 5; n++) {
        PrintInt(2);
    }
    return 0;
}
```

hw4_consoleIO_3

```c
#include "syscall.h"

int
main()
{
    int n;

    for (n=0; n < 12; n++) {
        PrintInt(3);
    }
    return 0;
}
```

hw4_consoleIO_4

```c
#include "syscall.h"

int
main()
{
    int n;

    for (n=0; n < 11; n++) {
        PrintInt(4);
    }
    return 0;
}
```

Part 2 Implementation

- Implement a **Priority Queue** Scheduler as described below:
    1. All processes must have a valid scheduling priority between 0 to 149. Higher values means higher priority. So 149 is the highest priority, and 0 is the lowest priority.
    2. In Homework 4, you are required to implement only the Priority Queue Strategy, meaning all threads will be in the same queue. ~~However, in the future, we plan to enhance this to include three different strategies, each with its own queue. Please keep this in mind while implementing your work.~~
    3. **Priority Queue** uses a non preemptive priority scheduling algorithm. A thread don't preempt other threads. If two threads enter the queue with the same priority, either one of them can execute first.
        1. Remember that even if it's non-preemptive, there's still a chance it can be preempted when transitioning from running to waiting.
        2. This may result in the order not being the same as its priority.

Part 2 Implementation (cont'd)

- Add a command line argument "-ep" for nachos to initialize priority of process.
- E.g., the command below will launch 2 processes: test1 with priority 40, and test2 with priority 80.

$ ../build.linux/nachos –ep test1 40 –ep test2 80

Part 2 Implementation (cont'd)

- Add a debugging flag "z" and use the DEBUG('z', expr) macro(defined in debug.h) to print following messages. Replace "{...}" to the corresponding value.
  1. Whenever a process is inserted into a priority queue

     [A] Tick [{current total tick}]: Process [{Process Name}] is inserted into queue
  2. Whenever a process is removed from a queue

     [B] Tick [{current total tick}]: Process [{Process Name}] is removed from queue
  3. Whenever a process changes its scheduling priority

     [C] Tick [{current total tick}]: Process [{Process Name}] is now selected for execution, Process [{Process Name}] is replaced.

Hint

- The following files “may” be modified...
  - threads/kernel.*
  - threads/thread.*
  - threads/scheduler.*
  - threads/alarm.*
  - lib/debug.*

Jenkins verification

- The TA's job will involve running four tests.
  1. hw4_console_1 60 hw4_console_2 70

     2 2 2 2 2 1 1 1 1 …
  2. hw4_console_1 70 hw4_console_2 60

     1 1 1 1 2 2 2 2 2 ….
  3. hw4_console_1 70 hw4_console_3 80 hw3_console_2 50

     3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 2 2 2 2 2
  4. -d z hw4_console_1 60 hw4_console_2 70

     (Testing debug messages)

     Test case 3 is affected by the fact that the higher priority queue is sleeping. This causes the scheduler, when seeking the next thread from the ready queue, to select a lower-priority thread for execution.

More Hints in this Homework

- The Implementation of HW2 on file ‘HW04_init’
  - So you won’t get blocked if HW02 does not work out for you.
- More hints in source code
  - You may see **“// TODO”** in the files to modify
  - TAs will provide another package of source code

Part2 Verification Example

```
machine@machine:~/code/test$ timeout 1 ../build.linux/nachos -ep conso
leIO_test1 70 -ep consoleIO_test3 80 -ep consoleIO_test2 50
consoleIO_test1 with priority 70
consoleIO_test3 with priority 80
consoleIO_test2 with priority 50
3
3
3
3
3
3
3
3
3
3
return value:0
1
1
1
1
return value:0
2
2
2
2
2
return value:0
```

```
machine@machine:~/code/test$ timeout 1 ../build.linux/nachos -d z -ep c
onsoleIO_test1 60 -ep consoleIO_test2 70 -ep
[A] Tick [10]: Process [mp3_test1] is inserted into queue.
[A] Tick [20]: Process [mp3_test2] is inserted into queue.
[B] Tick [30]: Process [mp3_test2] is removed from queue.
[C] Tick [30]: Process [mp3_test2] is now selected for execution, thread [main] is replaced.
2
[B] Tick [68]: Process [mp3_test1] is removed from queue.
[C] Tick [68]: Process [mp3_test1] is now selected for execution, thread [mp3_test2] is replaced.
[A] Tick [78]: Process [mp3_test2] is inserted into queue.
[B] Tick [96]: Process [mp3_test2] is removed from queue.
[C] Tick [96]: Process [mp3_test2] is now selected for execution, thread [mp3_test1] is replaced.
[A] Tick [106]: Process [mp3_test1] is inserted into queue.
2
[B] Tick [142]: Process [mp3_test1] is removed from queue.
[C] Tick [142]: Process [mp3_test1] is now selected for execution, thread [mp3_test2] is replaced.
[A] Tick [143]: Process [mp3_test2] is inserted into queue.
[B] Tick [143]: Process [mp3_test2] is removed from queue.
[C] Tick [143]: Process [mp3_test2] is now selected for execution, thread [mp3_test1] is replaced.
[A] Tick [153]: Process [mp3_test1] is inserted into queue.
/...
1
[A] Tick [481]: Process [mp3_test1] is inserted into queue.
[B] Tick [481]: Process [mp3_test1] is removed from queue.
[C] Tick [481]: Process [mp3_test1] is now selected for execution, thread [mp3_test1] is replaced.
return value:0
done
```

Grading

- Part2 (Implementation) – 98%
    1. Priority Queue Correctness – 86%
    2. Debug Flag – 12%
- Report Format – 2%
- Deadline: 12/31 23:59
- No late submission is accepted as the deadline has been postponed.

Report Format

- Please follow the word file to form your report for HW04
- Format guide
  - Content format: 12pt front, ~~16pt row height~~, and align to the left.
  - Caption format: 18pt and Bold font.
  - ~~Font format: Times New Roman, 標楷體~~
  - ~~Figure: center with single line row height.~~
  - Upload pdf file with the filename:
    - OS_HW04_GROUP_XX.pdf (change XX to your Group Number)

Reminder

- 0 will given to cheaters. Do not copy & paste!
  - TA will check your repository
- Feel free to ask TA questions
  - The TA has created a channel named '作業討論'. If you have any questions about the homework, please ask and discuss them in the channel. The records in the channel may help classmates who have similar issues.
  - The TA will not help you debug your code.
- No late submission is accepted as the deadline has been postponed.

Q & A

Thank you for your attention