CS 423 MP2: Rate-Monotonic CPU Scheduling

G3: Weijie Liu (wliu43), Hongwei Wang (hwang172)

1. Overview

This MP requires developing a Rate Monotonic Scheduler for Linux using Linux Kernel Modules. Round-base Admission Control and basic kernel API are used. Slab allocator is also utilized to improve the performance. We also implement a test application to test our real-time scheduler.

2. Important Implementation Decision

Basically, we follow the instructions step by step with the same architecture as the instruction of MP2.

(1) The scheduler is implemented as a Linux Kernel Module with proc file system which provides write callback and read callback function for the program in user space to communicate with. Every task is with a wake-up timer, whose handler will change the state of the task and wake up the Dispatching Thread.

(2) Slab allocator is used to improve the performance of the module.

(3) A Dispatching Thread is implemented as a kernel thread to schedule the tasks in user space. It is also a two-halves mechanism where the top half is the wake-up timer handler and the bottom half is the dispatching thread.

(4) We use spinlock inside timer handler to protect the shared valuable and use read-write semaphore in the list data structure to provide read-write protection.

(5) For the consideration of performance, the bound-base admission control is implemented as integer arithmetic as (1000 * Sum of processing time)/Sum of period <= 693.

(6) Users can input the number of rounds of calculate a factorial, which can change the processing time of a job in a task. Then user can input a number, which representing the ratio of period over processing time. The number of jobs in a task is also the input of the user. This can change the parameters of the test application.

3. Compile and Test

Source file:

mp2_rms.h/mp2_rms.c -- Give the kernel module implementation

mp2_list.h -- the list data structure and some functions, like change a state of a task, travel the list

mp2_dispatch.h -- dispatching thread to schedule

mp2_test.c -- test application

To compile the module and the test application, simply type "**make**".

To install the module, type "**sudo insmod mp2_rms.ko**".

To unload the module, type "**sudo rmmod mp2_rms**".

To test the module, you can run multiple test applications by typing "**./mp2_test**", and run "**cat /proc/mp2/status**" to observe the process list. You can input different parameters in several test applications to see the result.