2020 OS Project 1 _ Process Scheduling

資工二 B07902069 李哲宇

1. 設計

使用2顆 CPU，0執行 Scheduler；1執行 children processes。

a. kernel_files
   - get_time.c system call based on the function getnstimeofday
   - printkk.c show the information

b. main.c
   - cmp() qsort() 所需要的 cmp compare 函式
   - main()
   - unit() 產生 unit time

c. schedule.c
   - scheduling() 具有三個參數的排程
     (i) opt 判別 scheduling policy
     (where FIFO is 0; RR is 1; SJF is 2; PSJF is 3）
     (ii) 排程的 process
     (iii) number of process，N。
   - create() 使用 fork() 產生 children process 並用 system call 知道執行的時間
   - get_next_id()
     判斷是否不能在當前程序執行時轉換為其他程序進行（non-preemptive-policy）
     其餘依照以下方式排程

FIFO(opt=0)	行程中 ready time 最小的執行。
RR(opt=1)	當沒有行程在執行時，執行 index 最小的。 當時間來到500(time quantum)，換下一個執行。
SJF(opt=2) PSFF(opt=3)	從所有 ready 且尚未執行完畢的行程中挑選剩餘時間最小的行程執行。若剩餘時間最小的行程超過一個，則挑選 index 最小的執行。

d. setting.h
   - 宣告上述函式
   - 定義結構體 Process

name	the name of the process
rt	ready time of the process
et	execution time of the process
pid	process id

2. 核心版本
    - Kernel Version: 4.14.25 (From Homework1)
    - Platform: Ubuntu 18.04.2

3. 比較實際結果與理論結果
    a. 在 output 檔案中，可以見到實際排程與理論排程中行程執行順序相同。
    b. 然而透過 dmesg 可以知道，當不同 kernel 競爭 CPU 時，實際執行結果與理論不盡相同。
    c. 至於延遲的原因則推測可能為
        (1) Scheduler 與 children processes 的邏輯判斷及運算
        (2) context switch
        (3) 虛擬機本身的不穩定性