OS Project1 Report

B06902061, 林采鋒

1. 設計

單核心實做、選 SCHED_FIFO 排程策略、用 named fifo 溝通

(1)根據四種不同的演算法將 task 切開，以 struct SEQ 的開始執行時間排序好，其中 struct SEQ 包含以下資訊：

- 這段 sequence 開始時間
- 這段 sequence 執行時間
- 這段 sequence 是哪一個 task 執行

(2)以單核心實做，先將 parent process 設成特定 CPU 執行 (set affinity)，保證之後 fork 出去的 child process 也會繼承此特性，由同一顆 CPU 執行。

(3)以 SCHED_FIFO 作為排程策略，其中 priority 最高為 99, 最低為 1。
我讓所有的 processes 中，只有一個是 99，其他都是 98。
例如有兩個 processes P1, P2，優先做P1，再做 P2 (ex: P1 是 parent process，P2 是即將執行 seq 的 child process)，當 P1 要讓與 priority 的時候，就把自己設成 98，把 P2 設成 99，再執行 sched_yield()，就能成功讓給 P2 執行。

(4)首先將 parent_priority 設為 98，這樣 fork 出去的 child_priority 也是 98。
對於每個 seq，先檢查 pid 是否存在，不存在則 fork，child process 將 task name 和 task time 一併傳入 exclp 開啟。這時先把優先權給新的 child process 呼 system call 紀錄開始時間、開啟 reading named fifo，再將優先權還給 parent process。

(5)parent process 開啟 writing named fifo。寫入 seq 的執行時間，把優先權給執行 seq 的 child process。child process 跑完更新剩餘時間，如果剩餘時間為 0，就表示 task 已經完成，呼叫 system call 紀錄結束時間、輸出至 kernel，並結束。若剩餘時間不為 0，就再把優先權交還給 parent process

2. 核心版本

   Linux 4.14.25

3. 實驗結果

   以下排程時間定義為「第一個 child 放上 CPU 到 最後一個 child 離開 CPU」的時間，這樣理論的 task time (從 seq 陣列)、實際的 running time (從 dmesg.txt) 都比較好計算。

   (1)TIMEMEASURE:

   理論：5000 units

   實際：8.536261 s

   定義：u = 8.536261 / 5000 = 1.7072522 ms / unit

   (2)FIFO:

   FIFO_1:

   理論：2500 units = 4.268 s

   實際：4.248 s

   誤差：-0.47%

   FIFO_2:

   理論：87000 units = 148.531 s

   實際：154.811 s

   誤差：4.23%

   FIFO_3:

   理論：23000 units = 39.267 s

   實際：38.801 s

   誤差：-1.19 %

FIFO_4:

理論：3200 units = 5.463 s

實際：5.453 s

誤差：-0.18%

FIFO_5:

理論：23000 units = 39.267 s

實際：39.032 s

誤差：-0.60%

(3)RR:

RR_1:

理論：2500 units = 4.268 s

實際：4.497 s

誤差：5.37%

RR_2:

理論：16000 units = 27.316 s

實際：29.200 s

誤差：6.90%

RR_3:

理論：118500 units = 202.309 s

實際：212.366 s

誤差：4.97%

RR_4:

理論：83000 units = 141.702 s

實際：149.548 s

誤差：5.54%

RR_5:

理論：83000 units = 141.702 s

實際：150.391 s

誤差：6.18%

(4)SJF:

SJF_1:

理論：14000 units = 23.902 s

實際：25.223 s

誤差：5.53%

SJF_2:

理論：15300 units = 26.121 s

實際：27.745 s

誤差：6.22%

SJF_3:

理論：32020 units = 54.662 s

實際：57.763 s

誤差：5.67%

SJF_4:

理論：11000 units = 18.780 s

實際：21.052 s

誤差：12.09%

SJF_5:

理論：3500 units = 5.975 s

實際：6.308 s

誤差：5.57%

(5)PSJF:

PSJF_1:

理論：51000 units = 87.070 s

實際：92.178 s

誤差：5.87%

PSJF_2:

理論：15000 units = 25.608 s

實際：26.661 s

誤差：4.11%

PSJF_3:

理論：5000 units = 8.536 s

實際：9.235 s

誤差：8.20%

PSJF_4:

理論：15000 units = 25.608 s

實際：27.045 s

誤差：5.61%

PSJF_5:

理論：15300 units = 26.121 s

實際：27.878 s

誤差：6.72%

4. 分析討論

FIFO:

average error : 0.35%

因為 FIFO 是 non-preemptive， context switch 次數相對較少，大部分測資的 task 數又少於 TIME_MEASUREMENT，所以會有實際時間少於理論時間的情形。

其中 FIFO_2 因為 P1 時間遠大於 TIME_MEASUREMENT 的 500 units，所以會放大 time units 的時間，造成誤差較大。

RR、PSJF:

因為 RR 有 time quantum 的限制、PSJF 是 preemptive，而且大部分測資時間都是好幾千，所以 context switch 次數會增加許多，導致誤差放大。

其中 RR_4 RR_5 的總 unit 數一樣，但是 RR_5 較長，猜測是因為 RR_5 的 P5 P6 是一前一後進入 ready queue，所以中間會又會插入其他 process，導致 context switch 增加。