OS Project #1

ID: B07902070 Name: 陳昱妤

設計

Overall

對於一個 child（被排程的 process），會設計一個結構：

- name[32]: child 名稱
- ready: 何時會被 fork()
- exec: 這個 child 還要再經過多少 time unit 才結束
- pid: child 的 pid

一個 scheduler 會被指定在 CPU 0 上跑。
child（被排程的 process）會在對應的 ready time 被 fork() 出去，並將它 enQueue。
然後設定它們在 CPU 1 上跑，並且給予對應的優先序。
等到可以在 CPU 執行時，才從 waiting queue 中取出。

除了 FIFO 外，其它的排程方式都會採取以下策略：
設定 3 種優先序：
HIGH_PRIORITY (90)、MIDDLE_PRIORITY (50)、LOW_PRIORITY (10)。
正在執行的會給予最高的優先序，
在 waiting queue 中的第一個 child 會給予中等的優先序，
其它在 waiting queue 的 child 會給予最低的優先序。

再依照 scheduler 的時間，來決定是否 wait() 結束的 child，
或是改變正在執行的 child。

另外，避免 scheduler 和 child 間的時間誤差，
導致來不及動態調整正確的優先序，
會在排程前先 fork() 出一個 barrier 來跑 while(1) 迴圈。
它也會在 CPU 1 上執行，優先序設在 MIDDLE_PRIORITY 和 LOW_PRIORITY 之間。

FIFO

1. 先依照 ready time 排序
2. child 被 fork() 出去後，設定其在 CPU 1 上跑，
   優先序為 99 - [不含自己，已經 fork() 出去的 child 數]。
   由於前面提到的 barrier ，最多只能產生 79 個 child 才可正常運作。
3. 等全部的 child 都被 fork() 出去後，
   才會開始 wait() child。

RR

1. 先依照 ready time 排序，然後 scheduler 會跑迴圈計時。
   每經過一個 time unit，當時正在 CPU 中執行的 child，其 exec 會減少 1。
2. 如果此時 child 的 exec 為 0，表示該 child 已結束，
   需 wait() 它再繼續接下來的排程。
3. 設定 accumTime（在程式碼中寫成 accumQuantum）。
   如果此數值等於 500 且沒有 child 會在這個時間點結束，
   須強制從 waiting queue 中取出一個 child 來執行，
   並將原本執行的 child 放到 waiting queue 的最後面。
4. 如果要切換時恰好有要排進 waiting queue 內的 child，會先放入再切換
5. 一旦換上新的 child 來執行，accumTime 須歸零。

SJF

1. 先依照 ready time 排序，然後 scheduler 會跑迴圈計時。
   每經過一個 time unit，當時正在 CPU 中執行的 child，其 exec 會減少 1。
2. 如果此時 child 的 exec 為 0，表示該 child 已結束，
   需 wait() 它再繼續接下來的排程。
3. 在 waiting queue 中的順序，會依照其 exec 的大小排序，小的在前。

PSJF

和 SJF 的差別，只有在每次有新的 child 被 fork() 後，
會先比較它和正在執行的 child，何者 exec 較小。
較大者會放入 waiting queue 中，另一個則可以在 CPU 中執行。

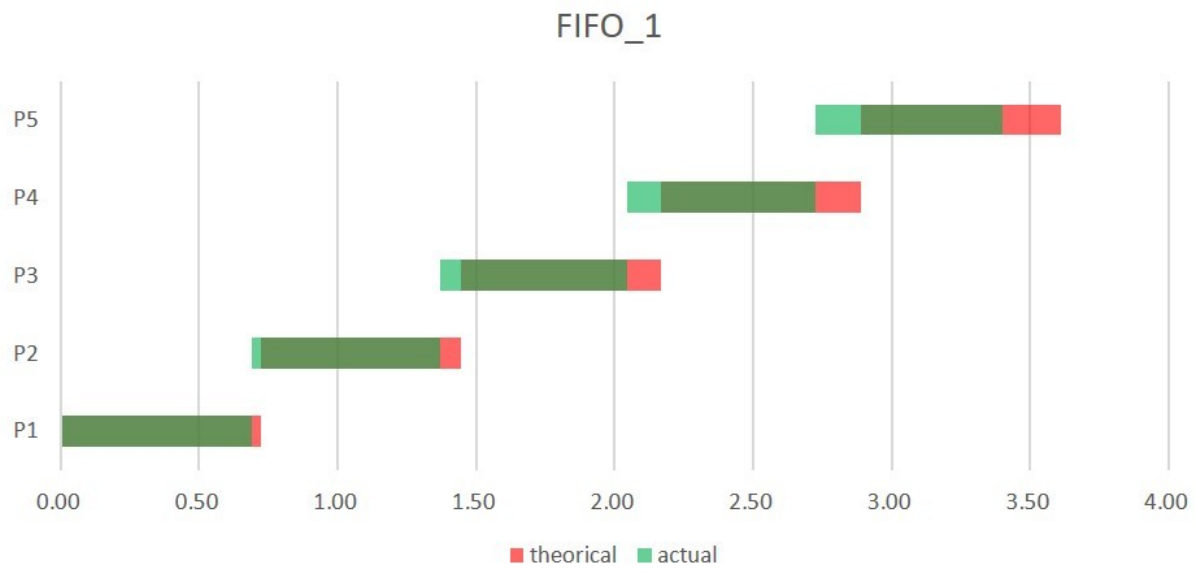
核心版本

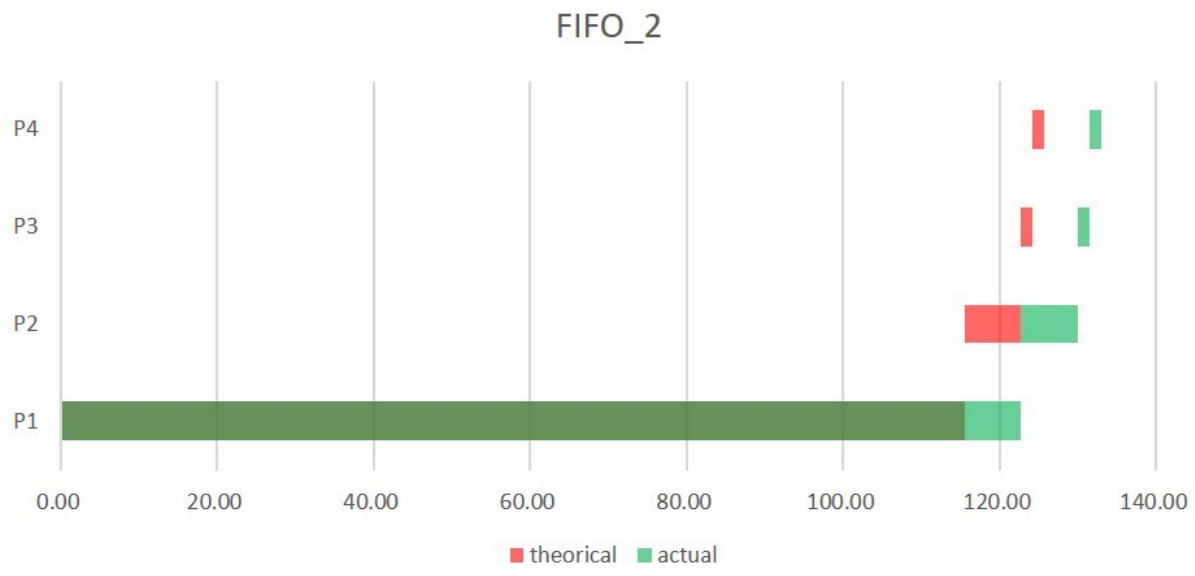
- `linux 4.14.25 x86_64`

比較實際結果與理論結果，並解釋造成差異的原因

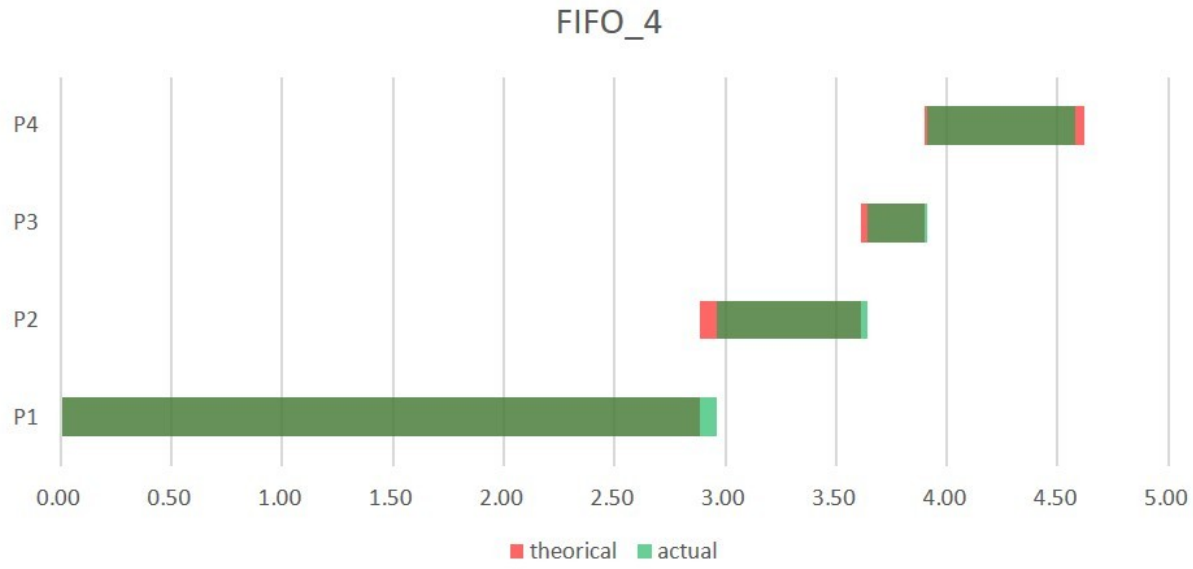
以 `TIME_MEASUREMENT` 當作基準，算出一單位時間為 0.001444009 秒。

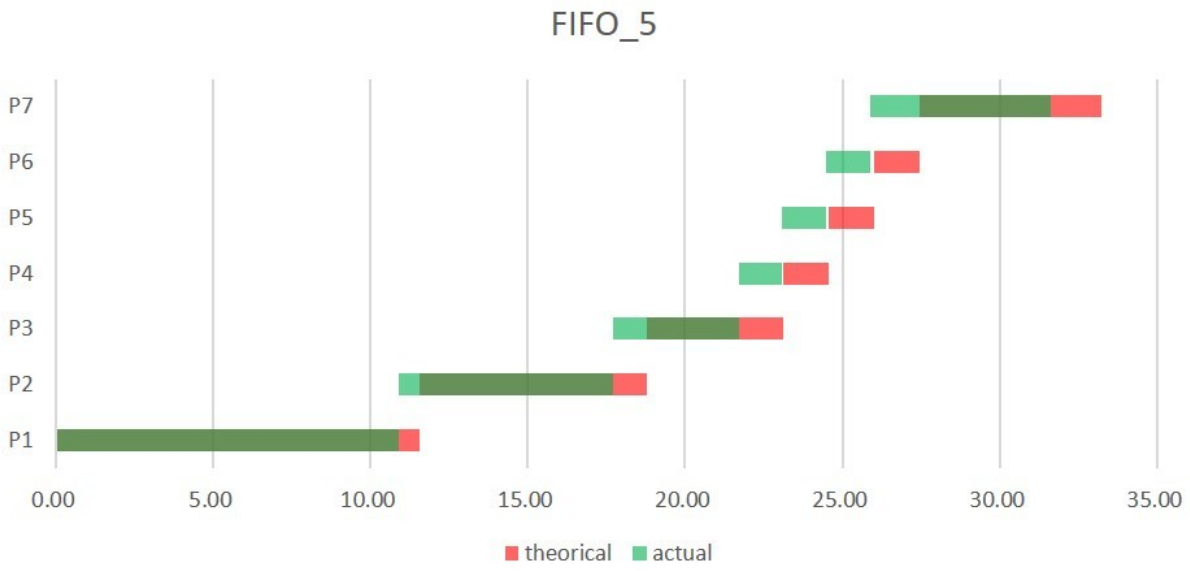
FIFO



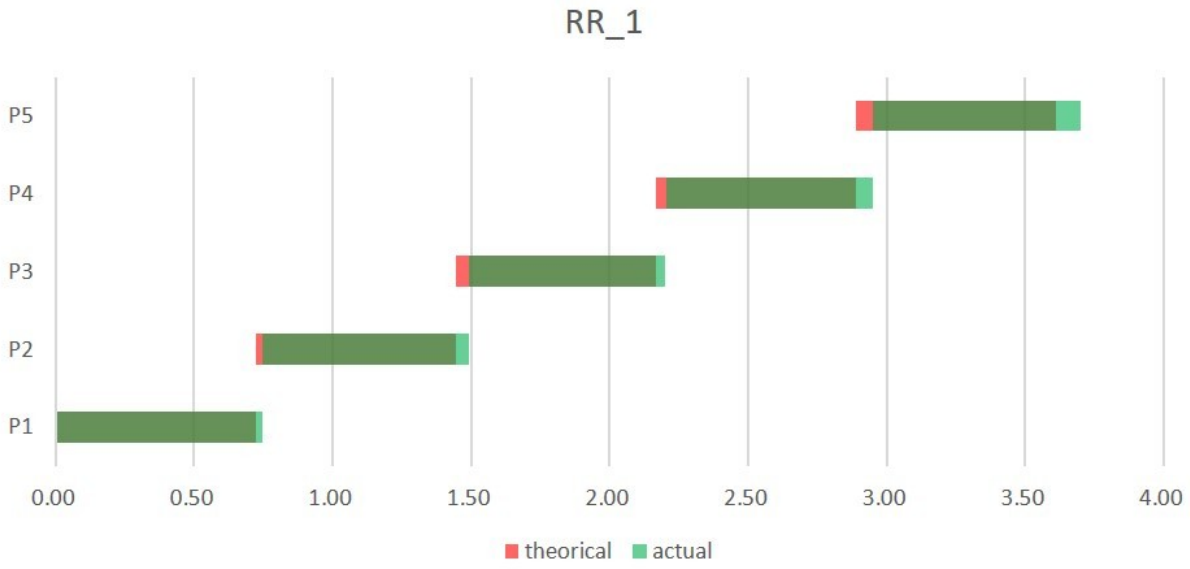


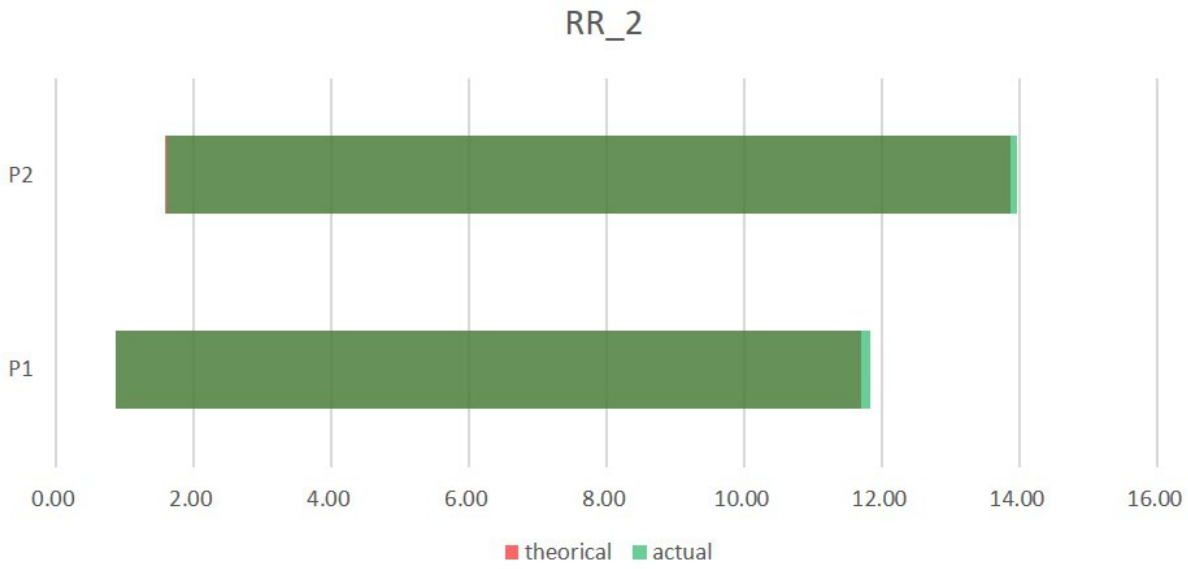

FIFO_3
P7
P6
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
theorical
actual

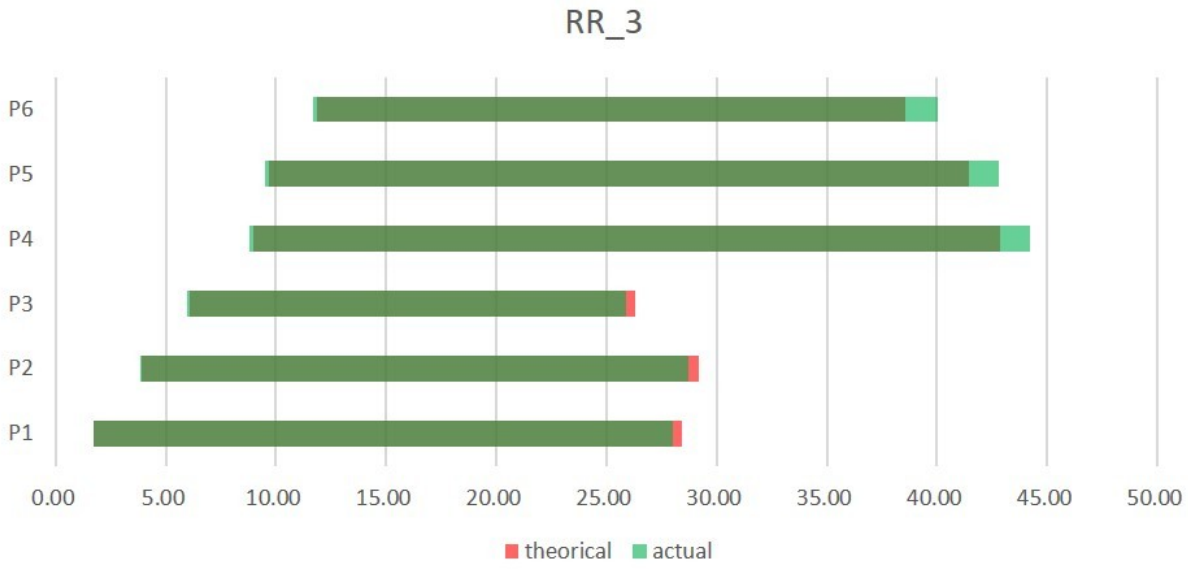

FIFO_4
P4
P3
P2
P1
0.00
0.50
1.00
1.50
2.00
2.50
3.00
3.50
4.00
4.50
5.00
theorical
actual

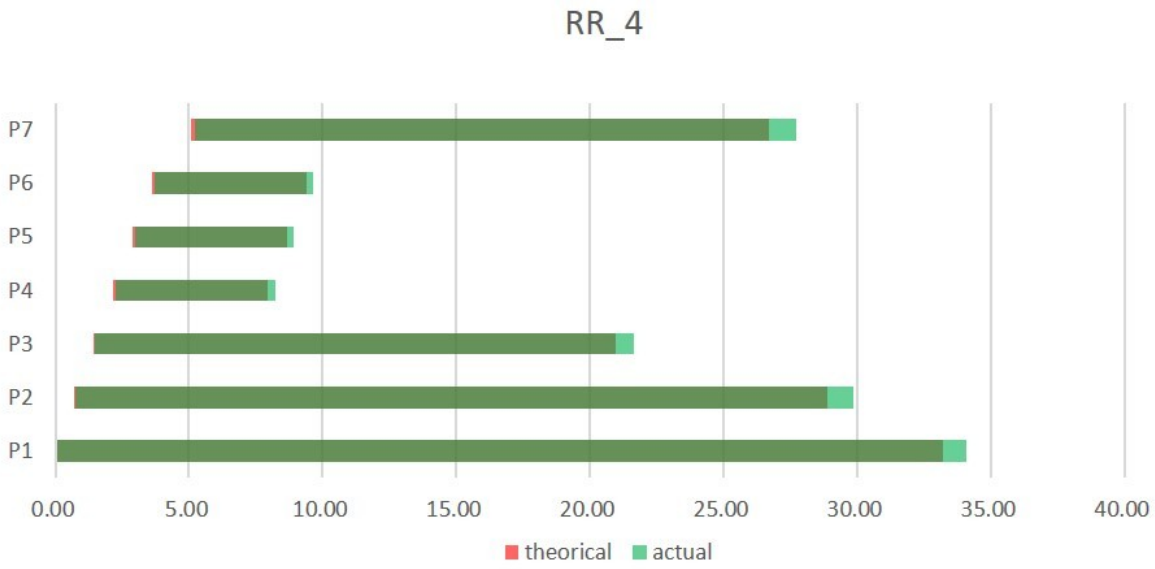

FIFO_5
P7
P6
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
theorical
actual

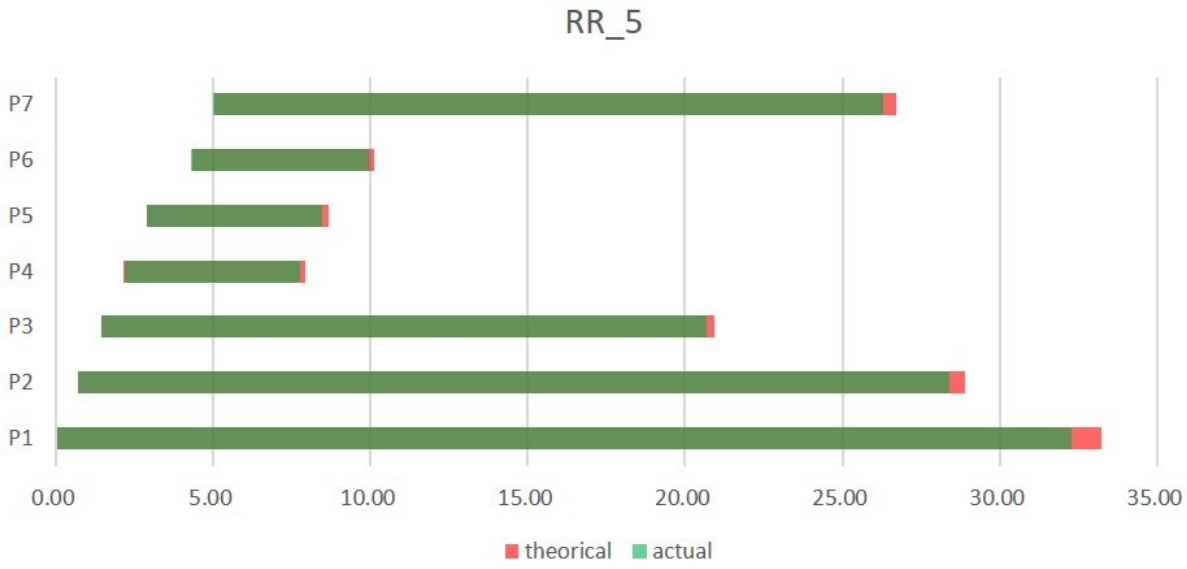
RR


RR_1
P5
P4
P3
P2
P1
0.00
0.50
1.00
1.50
2.00
2.50
3.00
3.50
4.00
theorical
actual

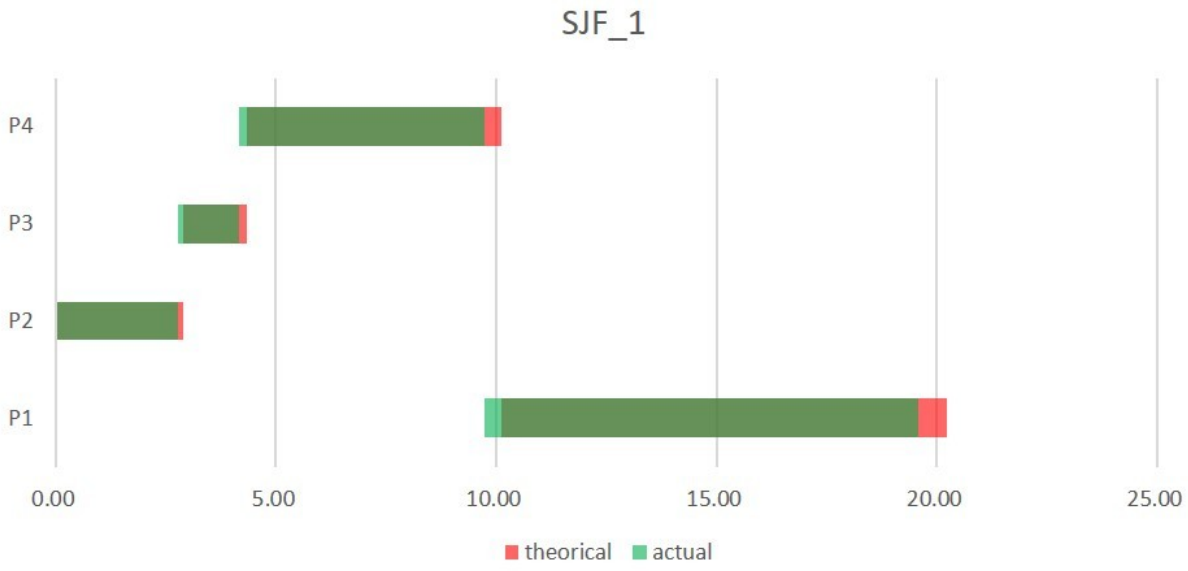

RR_2
P2
P1
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
theorical
actual


RR_3
P6
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
40.00
45.00
50.00
theorical
actual

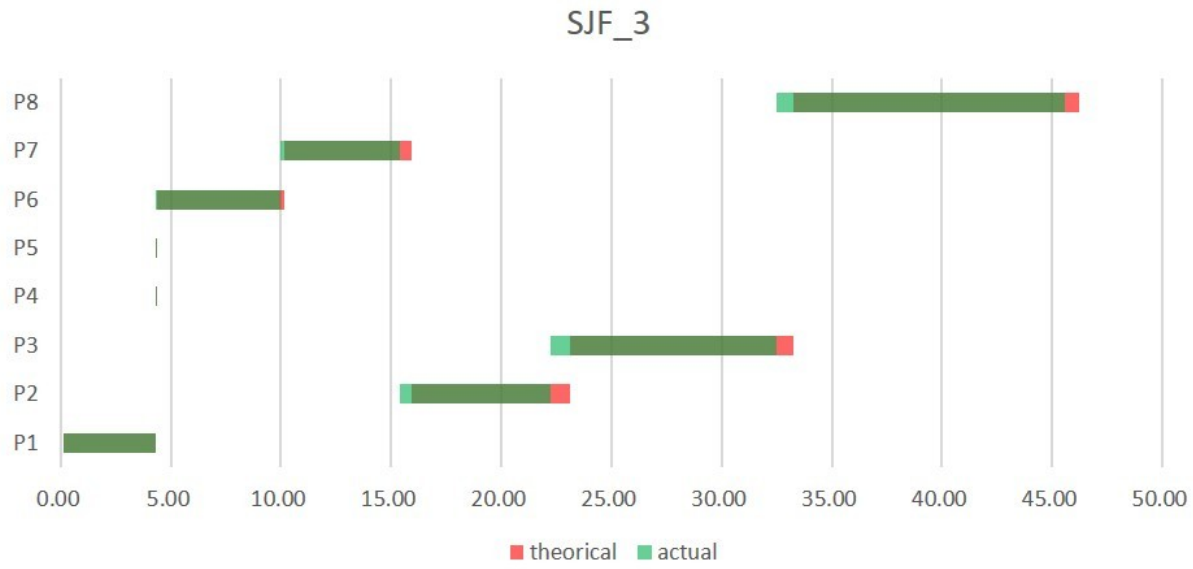

RR_4
P7
P6
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
40.00
theorical
actual


RR_5
P7
P6
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
theorical
actual

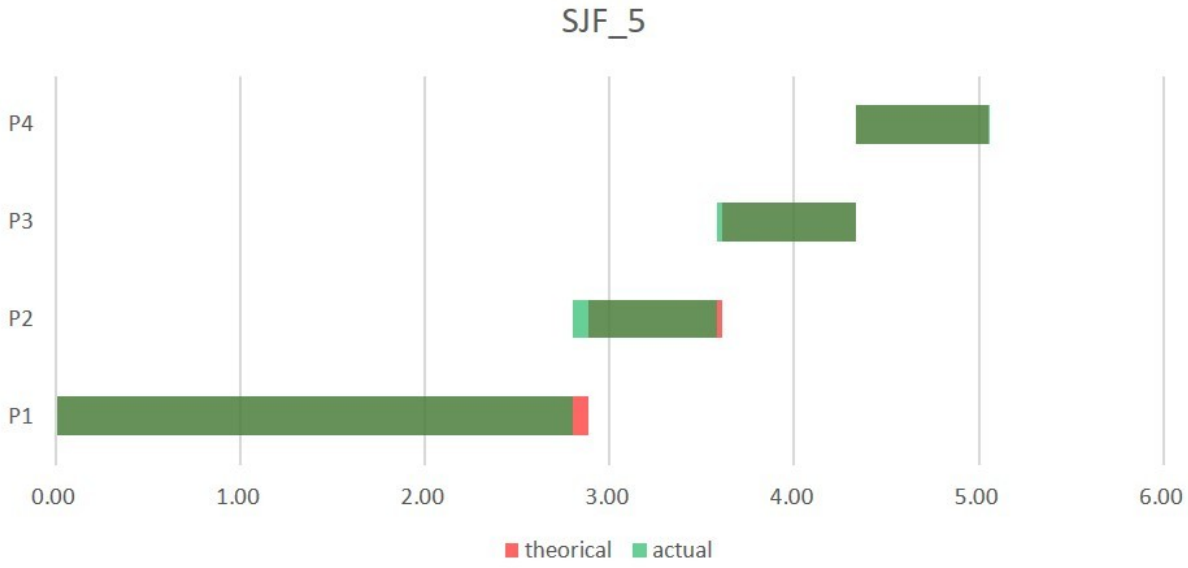
SJF


SJF_1
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
theorical
actual

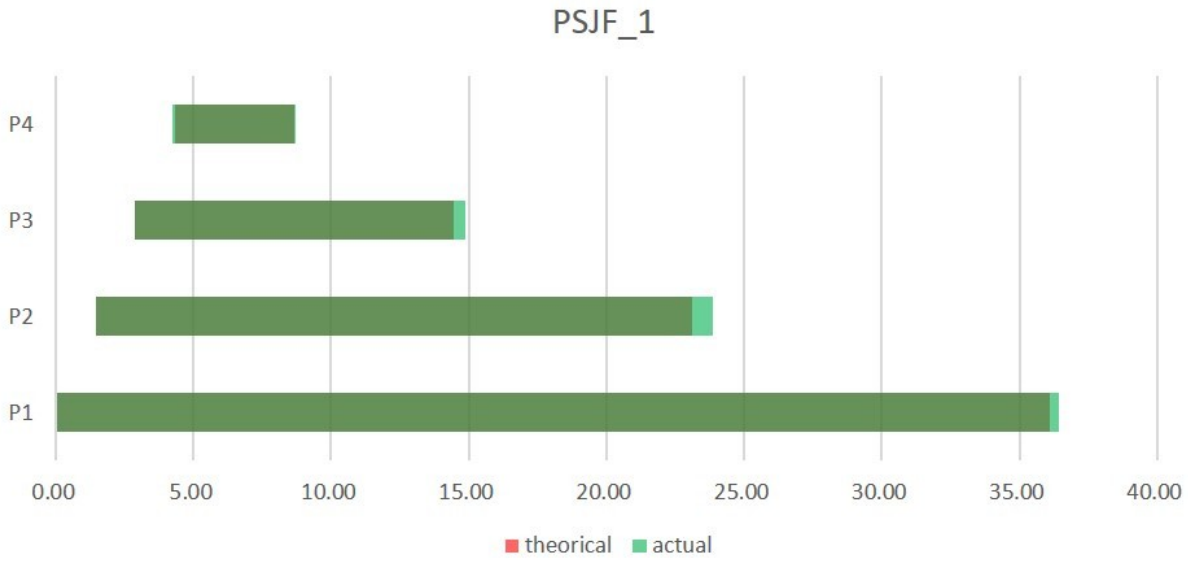

SJF_2
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
theorical
actual


SJF_3
P8
P7
P6
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
40.00
45.00
50.00
theorical
actual


SJF_4
P5
P4
P3
P2
P1
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
18.00
theorical
actual

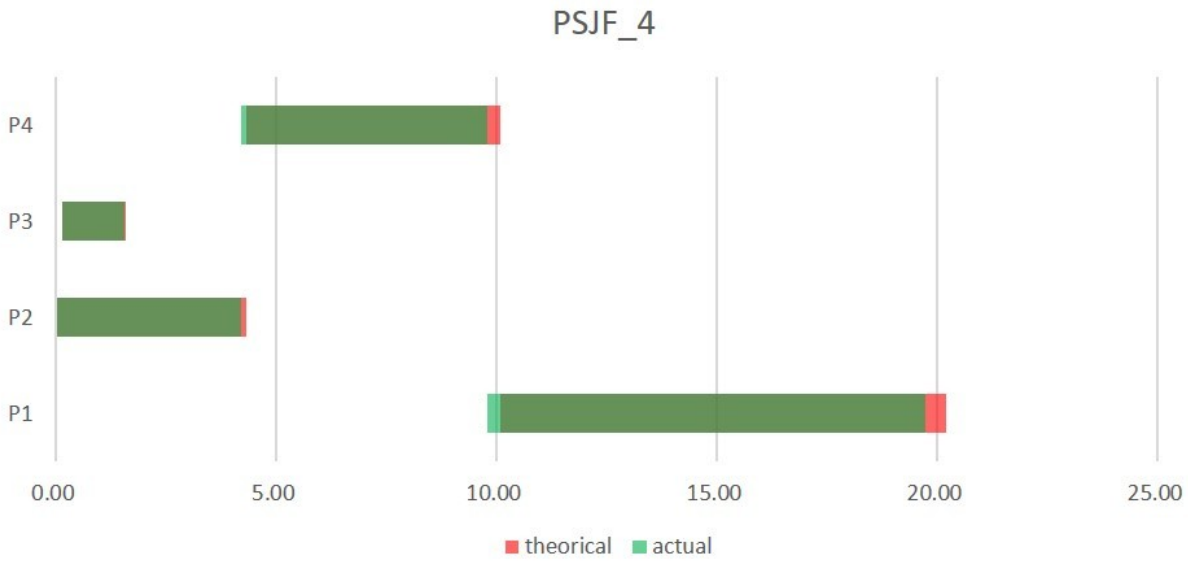

SJF_5
P4
P3
P2
P1
0.00
1.00
2.00
3.00
4.00
5.00
6.00
theorical
actual

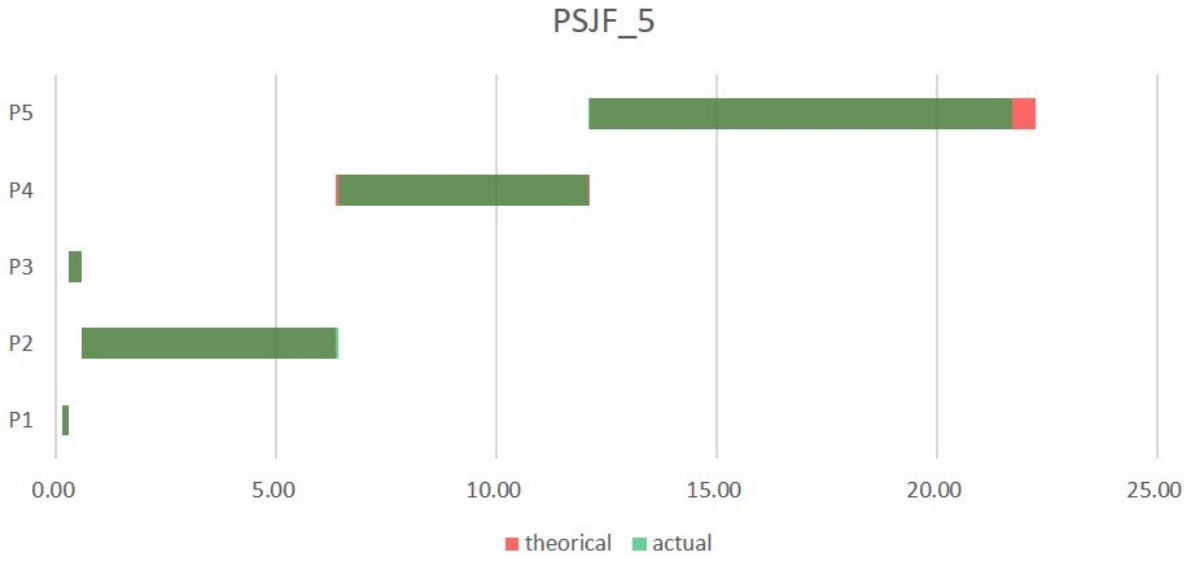
PSJF


PSJF_1
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
30.00
35.00
40.00
theorical
actual


PSJF_2
P5
P4
P3
P2
P1
0.00
2.00
4.00
6.00
8.00
10.00
12.00
14.00
16.00
18.00
theorical
actual


PSJF_3
P4
P3
P2
P1
0.00
1.00
2.00
3.00
4.00
5.00
6.00
theorical
actual


PSJF_4
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
theorical
actual


PSJF_5
P5
P4
P3
P2
P1
0.00
5.00
10.00
15.00
20.00
25.00
theorical
actual

- 註 1：橫軸為 scheduler 執行的秒數。
- 註 2：如果最小的 ready time 不為 0，由於沒有觀測數據，故以理論值當作開始的秒數。
- 註 3：以 Excel 長條圖作圖。

差異原因

- TIME_MEASUREMENT 本身即有誤差（當時工作環境可能使量出時間較大，因此大部分測資都有理論值大於實際值的現象）
- 實際值大於理論值的測資，大部分為可能暫停正在執行的 child 之排程方法（PSJF 、RR），因為要較頻繁的 context switch ，使工作量增加。
- 由於 child 之排程完全依照 scheduler 的計算時間來調控，可能在 child 結束時，scheduler 仍不知道要決定下一個跑的行程，造成除了 context switch 以外的誤差。

Reference

- wangyenjen/OS-Project-1: 2019 Operating System Project 1 - Process Scheduling https://github.com/wangyenjen/OS-Project-1
- B07902084 鄭益昀
- B07902110 張漢芝