資訊三甲 10827145 黃霈昕 作業系統 HW2 document

1. 開發環境

   CPU: Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz 1.99 GHz
   HDD:931GB
   RAM:8GB
   OS: Windows 10 - 21H1
   系統類型:64 位元作業系統，x64 型處理器
   程式編輯器: Visual Studio Code
   程式語言:python(3.9)

2. 實作方法和流程

   FCFS:讀入指定檔並記錄每個 process 的 ID、CPU Burst、Arrival Time、 Priority，再將 process 放到 list 中，再將 process 以 Arrival Time 排序，若 Arrival Time 相同則以 ID 排序。排序好的 process 依序執行並將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，處理完一個 process 才會計算該 process 的 Waiting Time 和 Turnaround Time 並換下一個 process 執行，一直到所有 process 處理完。最後將所有 process 依 ID 排序，畫出甘特圖再輸出各 process 的 Waiting Time 和 Turnaround Time。

   RR: 讀入指定檔並記錄每個 process 的 ID、CPU Burst、Arrival Time、 Priority，再將 process 放到 list 中，再將 process 以 Arrival Time 排序，若 Arrival Time 相同則以 ID 排序。當 curTime 與 process 的 Arrival Time 相同時，將該 process 放入 readyQueue 中等待被執行。正在執行的 process 若 Time Out，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中並放回 readyQueue 中，然後到 readyQueue 中取第一個 process 做執行。正在執行的 process 若執行完畢，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，計算該 process 的 Waiting Time 和 Turnaround Time，然後到 readyQueue 中取第一個 process 做執行。所有 process 都執行完畢後，將所有 process 依 ID 排序，畫出甘特圖再輸出各 process 的 Waiting Time 和 Turnaround Time。

   SRTF: 讀入指定檔並記錄每個 process 的 ID、CPU Burst、Arrival Time、 Priority，再將 process 放到 list 中，再將 process 以 Arrival Time 排序，若 Arrival Time 相同則以剩餘時間 remain 排序，若 remain 相同則以 ID 排序。當 curTime 與 process 的 Arrival Time 相同時，將該 process 放入 readyQueue 中等待被執行，若 readyQueue 中有 process 的 remain 比正在執行的 process 小，則進行奪取，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，並放回 readyQueue 中，開始執行奪取後的 process。正在執行的 process 若執行完畢，

將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，計算該 process 的 Waiting Time 和 Turnaround Time，然後到 readyQueue 中取第一個 process 做執行。所有 process 都執行完畢後，將所有 process 依 ID 排序，畫出甘特圖再輸出各 process 的 Waiting Time 和 Turnaround Time。

PPRR: 讀入指定檔並記錄每個 process 的 ID、CPU Burst、Arrival Time、 Priority，再將 process 放到 list 中，再將 process 以 Arrival Time 排序，若 Arrival Time 相同則以 priority 排序，若 priority 相同則以 ID 排序。當 curTime 與 process 的 Arrival Time 相同時，將該 process 放入 readyQueue 中等待被執行，並將 readyQueue 中的 process 依 priority 排序，若 priority 相同則放在相同 priority 者之後，若 readyQueue 中有 process 的 priority 比正在執行的 process 小，則進行奪取，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，並放回 readyQueue 中，開始執行奪取後的 process。正在執行的 process 若 Time Out 且 readyQueue 中有 process 的 priority 與正在執行的 process 相同，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中並放回 readyQueue 中，然後到 readyQueue 中取第一個 process 做執行。正在執行的 process 若執行完畢，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，計算該 process 的 Waiting Time 和 Turnaround Time，然後到 readyQueue 中取第一個 process 做執行。所有 process 都執行完畢後，將所有 process 依 ID 排序，畫出甘特圖再輸出各 process 的 Waiting Time 和 Turnaround Time。

HRRN: 讀入指定檔並記錄每個 process 的 ID、CPU Burst、Arrival Time、 Priority，再將 process 放到 list 中，再將 process 以 Arrival Time 排序，若 Arrival Time 相同則以 ID 排序。當 curTime 與 process 的 Arrival Time 相同時，將該 process 放入 readyQueue 中等待被執行，並計算 readyQueue 中 process 的 Response Ratio 當作 priority，以 priority 大者優先，若 priority 相同則以 Arrival Time 排序，若 Arrival Time 相同則以 ID 排序。正在執行的 process 若執行完畢，將執行中的 process 的 ID 依執行時間放入存放甘特圖的 list 中，計算該 process 的 Waiting Time 和 Turnaround Time，然後到 readyQueue 中取第一個 process 做執行。所有 process 都執行完畢後，將所有 process 依 ID 排序，畫出甘特圖再輸出各 process 的 Waiting Time 和 Turnaround Time。

3. 不同排程法的比較

Average Waiting Time

	Input1(TimeSlice=1)	Input2(TimeSlice=3)	Input3(TimeSlice=10)
FCFS	14.33	8.4	6.67
RR	18.4	6.4	11.67
SRTF	8.07	3	6.67
PPRR	14.67	9.4	12.5
HRRN	11.6	8.2	6.67

Average Turnaround Time

	Input1(TimeSlice=1)	Input2(TimeSlice=3)	Input3(TimeSlice=10)
FCFS	18.2	13.2	24.17
RR	22.27	11.2	29.17
SRTF	11.93	7.8	24.17
PPRR	18.53	14.2	30
HRRN	15.47	13	24.17

FCFS 是先到的先做，若有 process 正在執行，就必須一直等待正在執行的 process 執行結束才能換下一個 process 執行。若以 Average Waiting Time 和 Average Turnaround Time 評估效率，FCFS 的效率與 ReadyQueue 中等待的 process 數量和 process 的 CPU Burst 有關。若遇到前面有較長 CPU Burst 的 procees 或前面排隊的 process 很多時，效率會不高，如 input1、input2。若每個 process 需佔用 CPU 的時間點剛好錯開，就不會有那麼差的效率，如 input3。

RR 是依 TimeSlice 輪流依序執行，若 TimeSlice 用完須回到 ReadyQueue 中排隊。若以 Average Waiting Time 和 Average Turnaround Time 評估效率，RR 的效率與 TimeSlice 有關。若 TimeSlice 大小不適當，如 input1、input3，每個 process 執行一下就又要回到 ReadyQueue 中排隊，process 會花不少時間在重新排隊上，因此效率不高。若 TimeSlice 大小定義的適當，如 input2，效率就不會那麼差。

SRTF 是剩餘時間最短的先執行，而且是 preemptive 的。若以 Average Waiting Time 和 Average Turnaround Time 評估效率，SRTF 的效率是最佳的。由於剩餘時間最短的 process 會先執行，process 在 ReadyQueue 中排隊不會太久，因此有較高的效率。

PPRR 是高優先等級先執行，並對同優先等級者做 RR，而且是 preemptive 的。

若以 Average Waiting Time 和 Average Turnaround Time 評估效率，PPRR 的效率不高。由於 PPRR 是以優先等級排程，優先等級低者會需要一直在 ReadyQueue 中等待被執行，所以效率不高。

HRRN 是最高反應時間先執行，在 ReadyQueue 中等待越久優先權也會越高。若以 Average Waiting Time 和 Average Turnaround Time 評估效率，HRRN 照顧到了 Waiting Time，改善了 FCFS 因前面有較長 CPU Burst 的 procees 或前面排隊的 process 很多時效率會不高的問題，雖然效率不如 SRTF，卻比其他排成法高。

以 Average Waiting Time 和 Average Turnaround Time 比較這五個排程法，SRTF 效率最高，HRRN 次之，RR 若 TimeSlice 設定的好效率會比 FCFS 和 PPRR 高，若設定的不佳，效率會比 FCFS 和 PPRR 差，而 PPRR 效率又比 FCFS 差。

4. 結果與討論

在使用 SRTF 排程時較有效率，但實際上我們很難預估 CPU Burst，且若一直有新增的 process CPU Burst 長的 process 可能會出現 starvation 的狀況。因此，在能預估 CPU Burst 且新增的 process 固定時，可以選擇 SRTF 排程追求效率。HRRN 效率沒有 SRTF 高，但因考慮了 Waiting Time 方面的問題，改善了 SRTF 可能出現 starvation 的狀況，因此若要避免 starvation 可選擇 HRRN 排程。RR 若 TimeSlice 設定的好可以有比 FCFS 高的效率，但所謂好的 TimeSlice 很難設定，因此在無法找到好的 TimeSlice 的情況下，選擇 FCFS 排程較容易，而在可以找到適當 TimeSlice 的情況下，可以選擇 RR 排程。PPRR 效率雖然比 FCFS 差，但在需使用優先權做排程時也可以選擇 PPRR 排程。