OS Project1

B06902030 邱譯

設計

利用兩個核心，一個核心用來做排程，一個核心拿來執行process。

主要流程如下：

得到input後呼叫schedule()，此時該process會設定自己在0號cpu上運行，並在迴圈中判斷：

1. input中有沒有process ready了，如果有就fork出一個process，並設定child process在1號cpu且低權限。
2. 根據policy計算下一個應該執行的process為何，將他設定為高權限，並把原本在執行的process設定為低權限。
3. 如果有process執行完，就wait它。

直到所有process執行完畢。

而child process在被create時以及結束後利用system call my_get_time得到時間，並將這些資訊利用system call my_printk放到dmesg裡。

核心版本

4.14.25

測試結果與討論

（我的start time為process被create的時間）

TIME_MEASUREMENT.txt

(0.929285+0.937498+0.994876+0.956142+0.953980+0.948934+0.966160+0.975392+0.970865+0.916968)/10=0.955

平均每 500 unit 花費 0.955 秒

FIFO_1.txt

實際執行時間 = 0.995374+1.971296+2.935679+3.949329+4.871082=14.72276

理論執行時間 = (500+1000+1500+2000+2500)/500*0.955 =14.325

誤差 = (14.72276-14.325)/14.325=**0.027**

PSJF_2.txt

實際執行時間 = 1.923705+7.710387+3.873751+1.935884+17.272606=32.716333

理論執行時間 = (1000+4000+2000+1000+9000)/500*0.955=32.47

誤差 = (32.716333-32.47)/32.47=**0.007**

RR_3.txt

實際執行時間 = 28.315907+33.938513+32.497858+43.290599+48.128670+50.456020=236.627567

理論執行時間 = (14600+17500+16800+22400+25000+26400)/500*0.955=234.357

誤差 = (236.627567-234.357)/234.357=**0.009**

SJF_4.txt

實際執行時間 = 5.943480+5.915917+11.722586+3.849421+11.567529=38.998933

理論執行時間 = (3000+3000+6000+2000+6000)/500*0.955=38.2

誤差 = (38.998933-38.2)/38.2=**0.02**

測試結果

1. 可以觀察到實際執行時間都大於理論執行時間，我在下方的誤差原因會討論這件事。
2. 基本上誤差都在2%以內，算是蠻可以接受。

誤差原因

我認為有以下幾個可因素會造成誤差：

1. 實際執行時context switch是需要時間的，理論中沒有計算到。
2. 實際執行時，排程、fork()等等額外的花費，也需要花時間，理論中沒有計算到。
3. 實際執行時，cpu的執行效能並非維持固定，因此會有影響。
4. cpu可能還需要處理電腦中的其他process。
5. 透過兩個核心進行實作，但兩個核心的時間軸不一定會同步，因此造成誤差。

根據上述1,2點，實際執行時間大於理論執行時間是合理的結果。