雲端系統與實務 2016

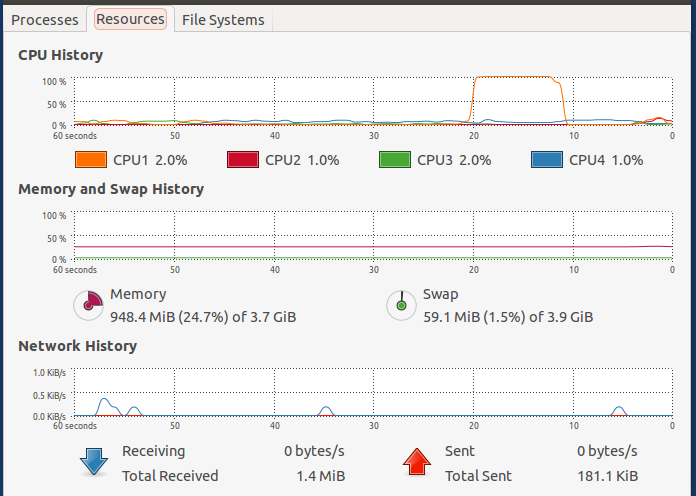
學號:0556148

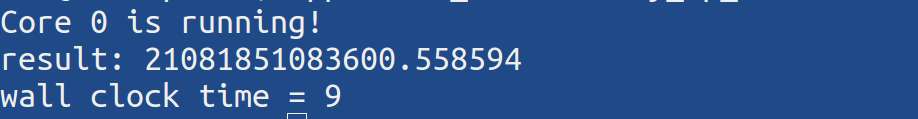
姓名:鄭偉聖

實驗環境:Ubuntu 14.04, 4顆CPU, 4G memory

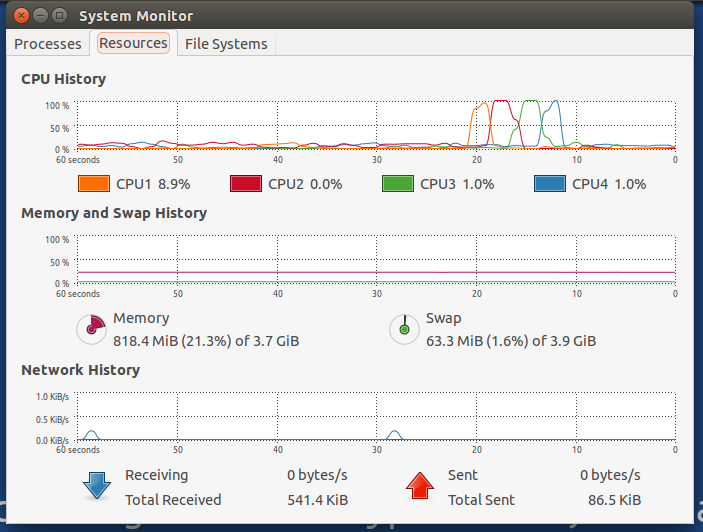
LAB1-1

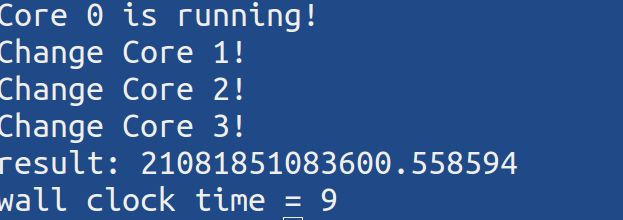
|  |  |
| --- | --- |
| 項目 | Time |
| a. 執行原始程式 pthread\_setaffinity\_np\_test.c | wall clock time = 8 |
| b. 修改原始程式以 create 一個 thread 並使用 pthread\_setaffinity\_np，讓 thread 在執行過程中切換 core 執行。 | wall clock time = 9 |
| c. 修改原始程式以 create 兩個 thread 並使用 pthread\_setaffinity\_np，讓兩個 thread 分別在不同兩個 core 上執 行。< pro\_2.c > | wall clock time = 5 |
| d. 修改原始程式以 create 兩個 thread 並使用 pthread\_setaffinity\_np，讓兩個 thread 強迫在同個 core 上執行。 < pro\_3.c > | wall clock time = 8 |



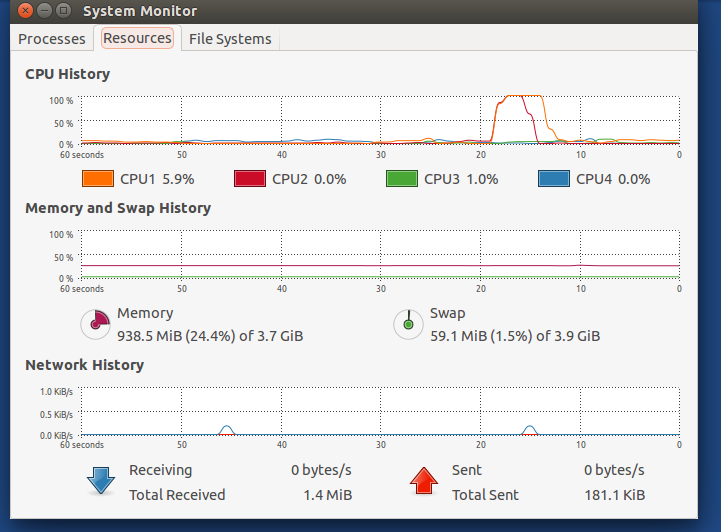


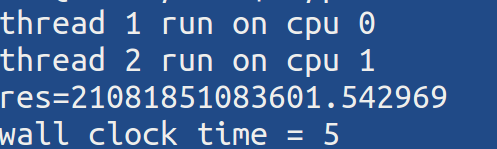
↑此圖為1-1a的實驗結果 可以看到thread在CPU0上run 最後的result是接下來實驗的依據,而只執行在一顆cpu上的時間為9



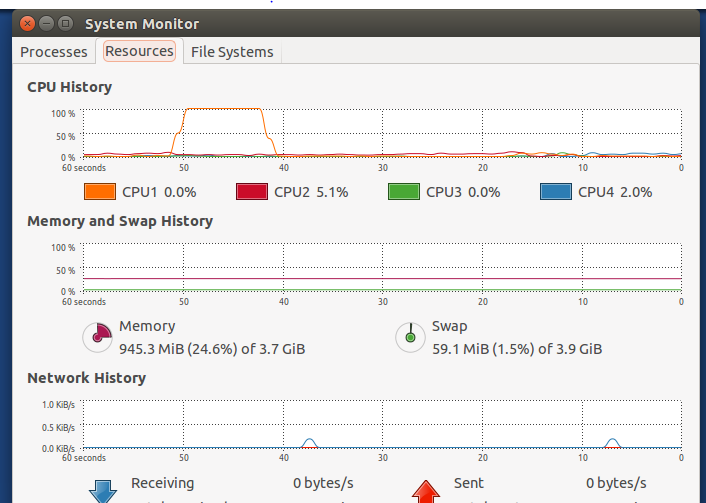


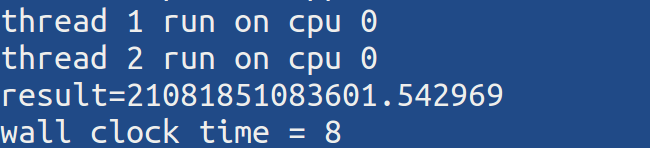
↑此圖為1-1b的實驗結果 可以看到thread在每顆CPU上都有run過,最後得出的result與實驗a的結果一樣,可得知的是時間與a時間變化不大,可能是因為只有執行三次的context switch。此實驗方法是透過直接在waste time上去做分割。





↑此圖為1-1c的實驗結果,將兩條thread 在兩顆cpu上,透過thread function的參數回傳來達到控制thread的行為,好讓他們去分工,(詳情請看CODE),最後會有兩個result 會發現加起來幾乎等於1-1a的result(值會除不盡,無法完全表示,因此會有小誤差),證明實驗分割無誤,也達到利用平行化讓時間縮短為5。





↑此圖為1-1d的實驗結果,將c的兩條cpu設在同一顆上就完成實驗了,由兩條thread result總和與a幾乎相同(誤差原因註解在c)可知實驗分割計算是正確的,時間上沒有比較快的原因應該是因為兩條thread一直在搶cpu資源,導致不斷的context switch固時間沒有比較快,甚至可能會變慢。