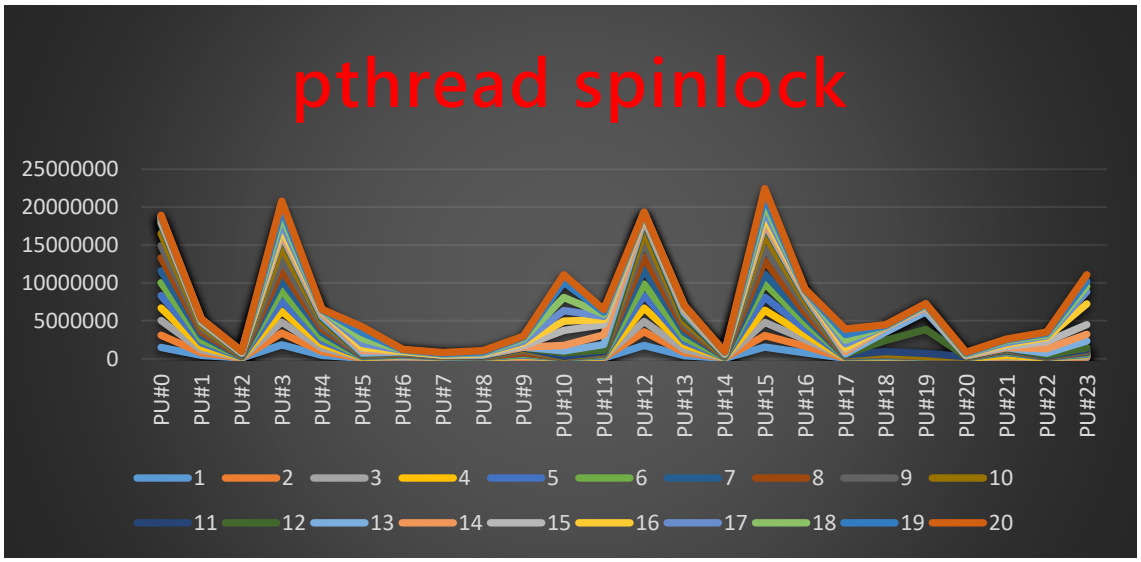
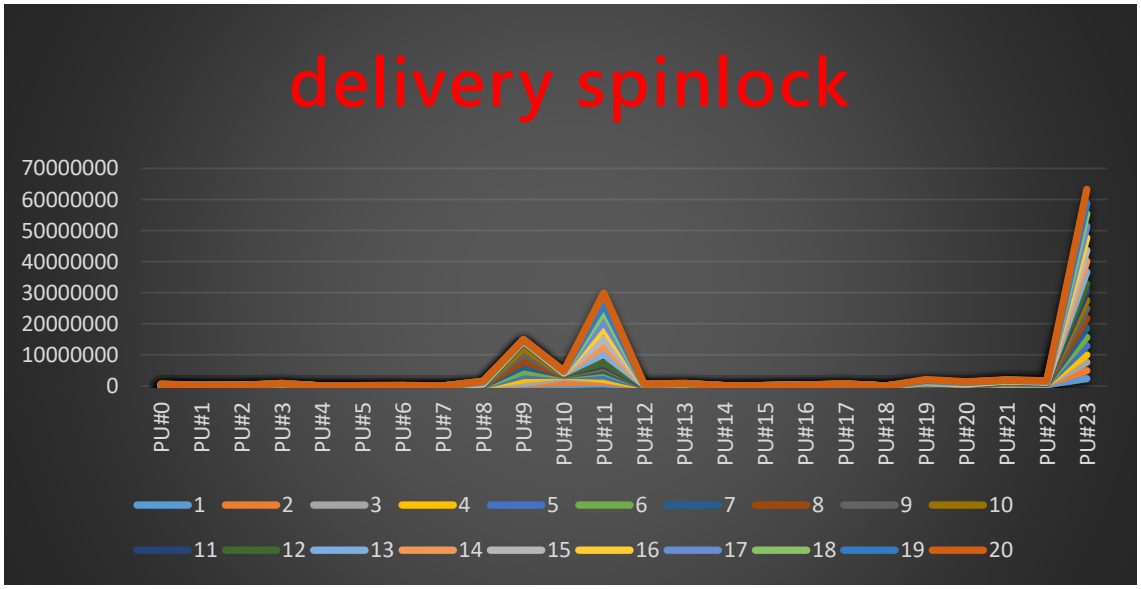
資工三 405235035 王博輝

作業系統概論 作業四

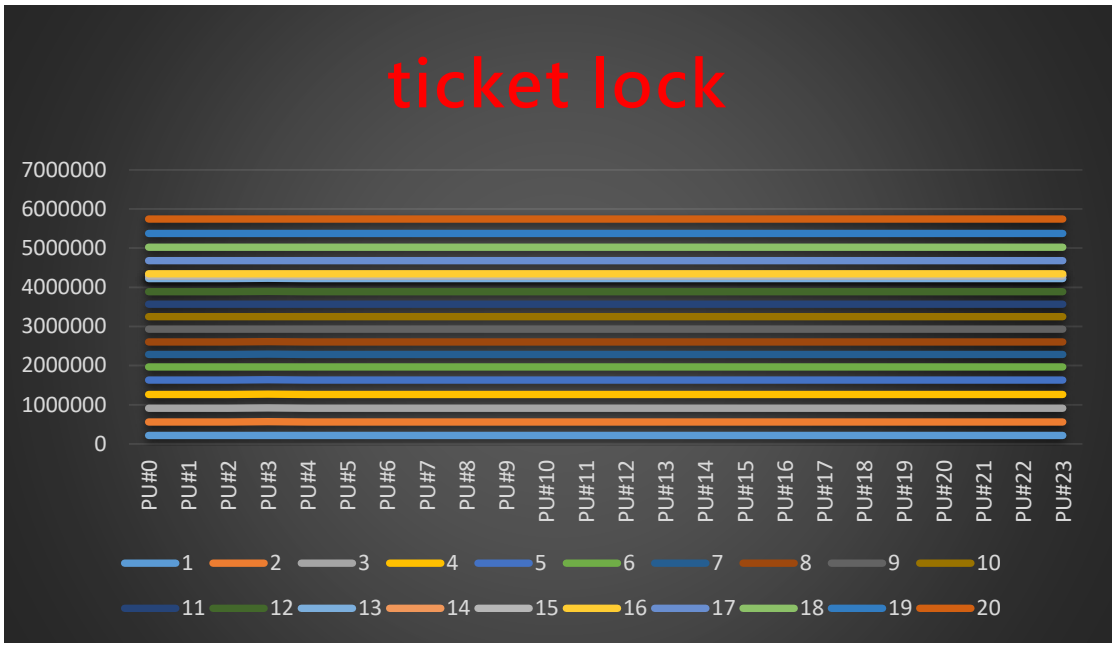
實驗環境：12 核心，24 硬體執行緒的電腦（AMD Threadripper）



實驗一：pthread spinlock 結果



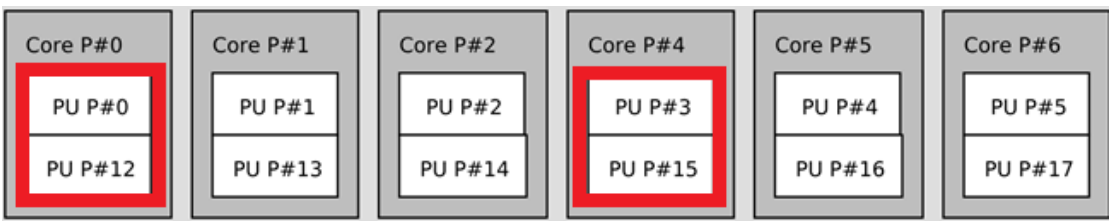
實驗二：delivery spinlock 結果



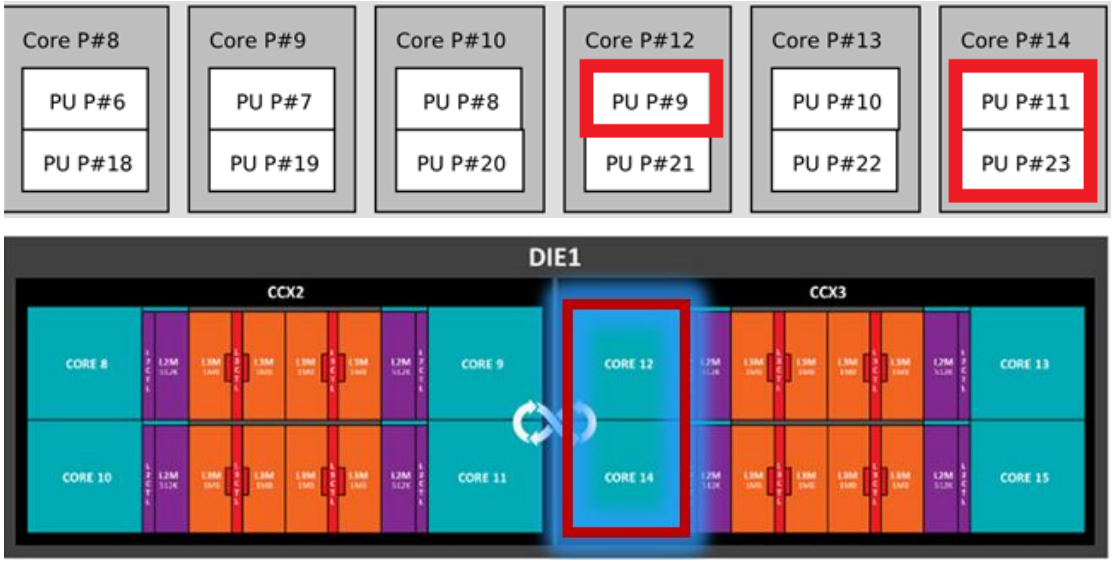
實驗三：ticket lock 結果

1. 重複實驗（spinlock[1~3]），並解釋為什麼每個 thread 拿到 lock 的機率並不一樣。申論。

實驗一：由圖上可發現，thread 0、3、12、15 拿到 lock 的機率比其他 thread 來得高，而這四個 thread 正好分別被封裝在 Core P#0 和 Core P#4 兩個核心中，所以我覺得造成每個 thread 拿到 lock 的機率不一樣的原因是因為資訊傳遞速度的差異造成，如果在同一顆核心上的 thread 做 unlock 的動作的話，那處於同一顆核心上的另一個 thread 就能比其他核心的 thread 早接收到 unlock 的資訊，進而可以比其他的 thread 早取得 lock 的權力。



實驗二：由圖上可發現，thread 9、11、23 拿到 lock 的機率較高，而 23 又遠比其他 thread 來得高，會產生這樣的結果的原因我想也是因為計算機架構的關係，因為每次 unlock 都要由 thread 23 來負責，所以 23 拿到 lock 拿到的次數是最多的並不意外，而在 23 unlock 之後，又因為資訊傳遞速度的關係，同處於同一核心上的 11 和相鄰於隔壁核心的 9 能夠更快的得知 unlock 的資訊，所以能夠比其他的 thread 更早 lock 進去 critical section。



實驗三：每個 thread 拿到 lock 的機率幾乎相同，因為我們在程式碼中利用取號碼牌的概念，適當的分配每個 thread 能夠拿到 lock 的順序，與其它的 spinlock 不同，ticket lock 就像是字面上的意思，是按照申請使用的順序來取得 ticket 進而取得 lock 的權力，而不是每一次 lock 釋出時都是先搶先贏。

2. 測試效能（spinlock[1~3]），請問這三種 spinlock 的效能比較，說明你的比較方法。申論。

名稱	總 lock次數	百分比
pthread spinlock	172849550	100.00%
delivery spinlock	126497825	73.18%
ticket lock	137744209	79.69%

將每種 lock 方法的第 20 次 lock 總次數加總得到圖上的數值 (20 輪共 100 秒)

可以發現 pthread spinlock 在執行時間相同的情況下的總 lock 次數最多，可見效率最高，但從上題可以得知負擔集中在某幾顆核心上，大部分的 thread 都無事可做，所以這是一個不太好的方法。

delivery spinlock 的效率最差，因為每一次 unlock 都一定要交由 P#23 來處理，之後再給其它的 thread lock 的機會，所以這是很沒有效率的，實驗出來的結果也是如此，同時間下總 lock 的次數最低，而且負擔集中的情形更為明顯。

ticket lock 的效率雖然只有 pthread spinlock 的八成，但資源分配非常平均，每個 pthread 都負擔差不多的工作量，所以我認為雖然效率在這三個 spinlock 中不是最好的，但我覺得這是最好的 lock 方法，因為每個 thread 都有在做事。

3. 雖然 ticketlock 看起來很公平，但 while loop 執行的太密集，請你加入「 asm("pause") 」讓 CPU 更有效率。程式碼。

請見 ticketlock.c

在 ticketLock_acquire 中的 while 迴圈中加入 asm("pause");

4. 你是否可以使用 atomic_compare_exchange 實作出具有 FIFO 功能的 spinlock。程式碼。可參考 delivery_spinlock.c

無實作