多處理機平行程式設計 作業五 report

F74109032 陳均哲

Problem 1

1. 問題回答

1. 把a, n, temp設成shared，並把count設成private。
2. 沒有，因為count的結果只被a影響，而那個迴圈裡面a的資料都不變，因此也不會被其他thread的運算影響。
3. 可以改memcpy的起始位置跟要複製的size，讓不同thread同時複製陣列的不同區段。
4. 如作業繳交區所繳交的程式碼。
5. 執行時間: qsort < parallel count sort < serial count sort。

2. 時間比較

n = 10000

thread_count	qsort	Serial	2	4	8	16	32
time(sec)	0.001025	0.677710	0.441776	0.214574	0.151591	0.150473	0.212346

3. 結果分析

這些結果是在我自己電腦上的虛擬機跑的，總共有八個核心，因此測出來的結果thread數量在8~16之間跑起來會是最快的算是合理的，而超過16個thread則會增加執行時間。另外還可以看出qsort明顯比這些count sort的結果都快很多，因為兩種排序法的複雜度不是在同個層級。qsort的複雜度是O(n log n)，而count sort則是O(n^2)。

Problem 2

1. 程式說明

我寫的方法是建一個keyword.txt，裡面以空格為區間存放要記錄的keywords。而我建了一個叫做files的資料夾，裡面放了0.txt~15.txt，共16個文檔。而根據使用者跑執行檔時輸入的producer數量，用cyclic partition的方式切割每個producer處理那16個文檔。而也會根據使用者輸入的consumer則是會有while迴圈讓他busy waiting直到shared_queue裡面有資料可以處理再進入函式讀取句子進行tokenize。當shared_queue進入push或pop時都會設定critical section避免race condition來實作題目所要求的thread safe queue。

2. 時間比較

固定producer = 4，改變consumer (單位:sec)

1	2	4	8	16	32
0.001526	0.001415	0.001718	0.002217	0.084755	0.137691

固定consumer = 4，改變producer (單位:sec)

1	2	4	8	16	32
0.00059	0.000951	0.003018	0.003522	0.00603	0.00992

3. 結果分析

以實作一個平行程設的作業看到這樣的結果對我比較稀奇，因為幾乎都是多個producer或多個consumer的情況下處理速度比較慢，但仔細想過後其實出現這樣的結果還滿合理的。

1. producer的工作是讀取其中一個檔案分成一行一行丟進去queue裡面。而設定多個thread雖然可以減少讀檔案總共花的時間，但因為每個producer丟句子進去queue的時候thread safe queue的機制需要設定critical section，變成其他producer都會卡住等正在區間裡面的producer出去才能換下一個進去。加上文檔裡面我放的文字量都不大，頂多就分成三行或四行，因此感覺卡在citical section的時間可能會大於多個thread讀檔所減少的時間。
2. consumer的工作是把queue裡面的東西抓出來tokenize，但是從queue裡面pop出東西的過程就需要進入critical section，因此等於還是多個thread會卡在critical section，要等退出critical section才能開始tokenize。而tokenize所花的時間也不顯著，因此這樣設定多個consumer進行平行化也無法有效減少執行時間。

雖然在我測試的程式中多個producer跟多個consumer都無法有效增加程式效能，但如果要處理的文檔規模變得真的非常大，讓文檔的數量變比較多，或是分行讀檔案跟tokenize花的時間相對變得非常顯著，說不定就會真的有效果。