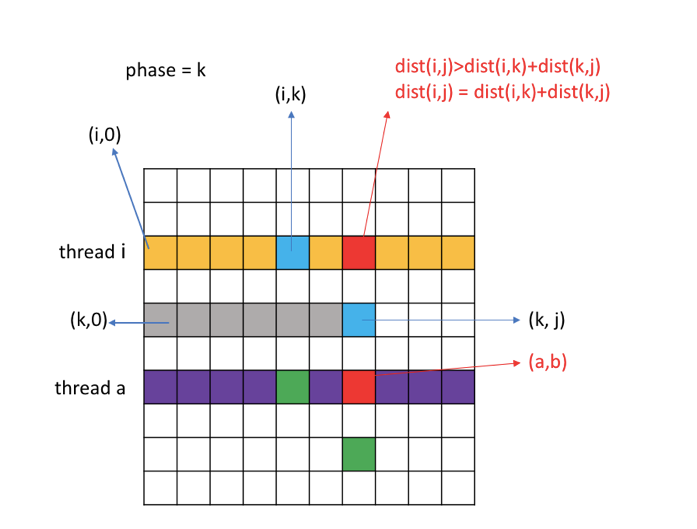
Parallel Programming hw3 ----APSP(CPU)

學號：109062639 姓名：葉哲欣

**Implementation:**

1. Which algorithm do you choose?  
   Algorithm : Foly-Warshall
2. How to implement?  
   Thread framework: OpenMP  
   採用老師上課所教的類似cuda平行處理方式，執行每個phase(k)時，由於每個階段是循序被執行的，並不會有data dependency，故以平行的方式每個thread處理一列下去比較。直到所有的row 被thread執行完成，換下一個phase。如右圖所示，k固定情況下，thread i 會跟每一個(i,j)之dist值做比較，當dist(i,j)> dist(i,k)+(k,j)則交換，跑完所有phase即可完成。
3. Time complexity  
   原本Foly-warshall algorithm time complexity: O(|V|3)  
   初值設定：  
   Edge讀取：O(3\*|E| / p )

無邊權重設定：O( |E|2/p)

Phase:O(|V|-1)=O(|V|)  
OpenMp :每個thread執行一row , 假設有p個threads,  
每個 thread 共要 n/p 次,故Time complexity: O(|V/p|\*|V|) = O(|V|2/p)

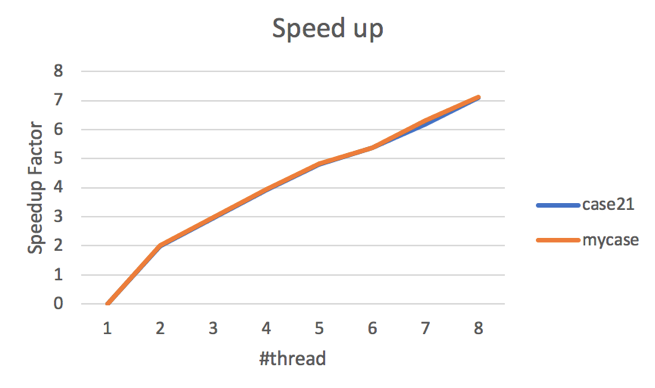
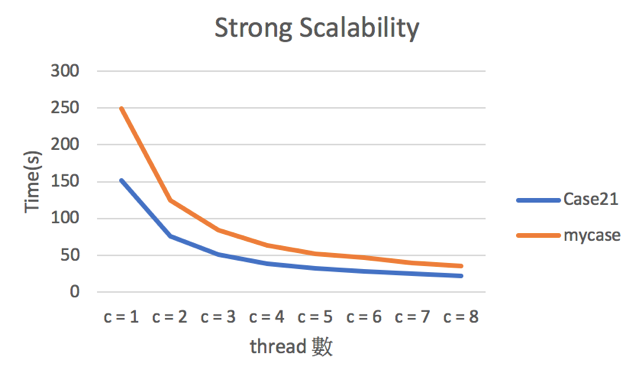
Time Complexity : O(3\*|E| / p ) + O( |E|2/p) + O(|V|)\*O(|V|3/p) = O(|V|3/p)

1. Design and generate testcase   
   採取complete graph方式，以random 方式生成：  
   |V| = 6000, |E|= |V|\*|V -1| E 之range : (500,1000)

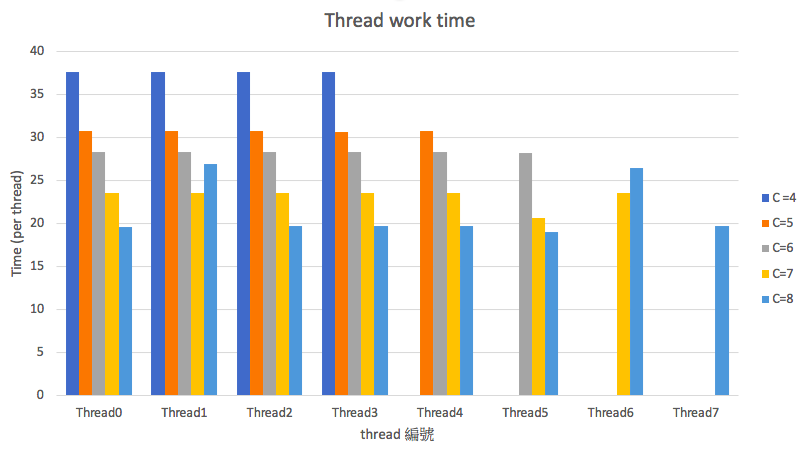
**Experiment & Analysis**

1. System Spec:   
   使用apollo system
2. Strong Scalabilitys

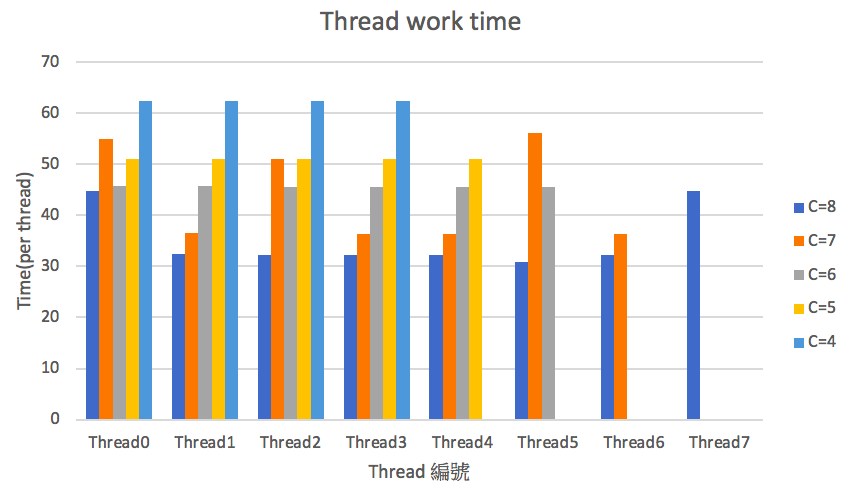
(1.) case: (case21,mycase)



(2.) Thread work time (case:21)

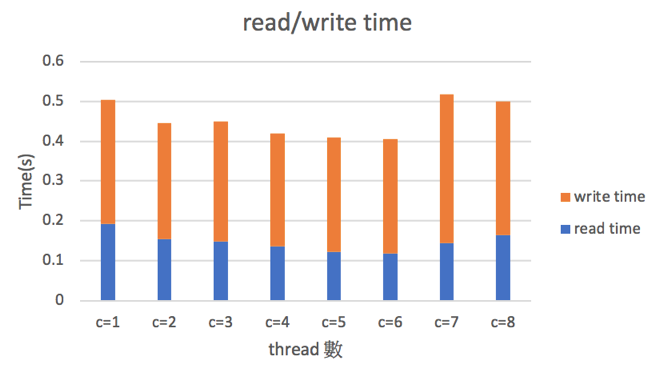
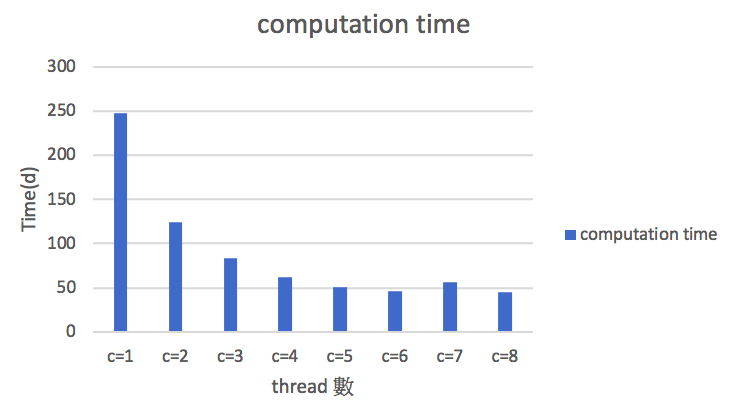


Thread work time (mycase)

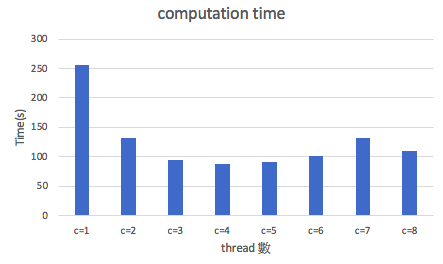
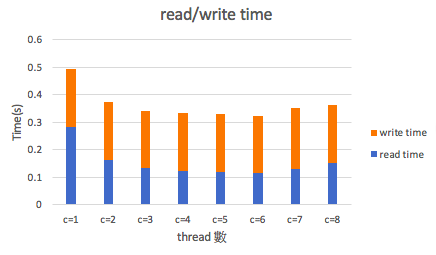


從上圖(2.)我們可以發現：此方法並不能有效地平均分配工作，thread執行時間不一，造成有些thread執行時間過長，而其他的thread必須等待。

(3.)case:mycase



(4.) method: collapse

**Experiment & Conclusion** a.  
 本次作業我學習到Folyd-warshall 如何實現平行運算，採取openMp 實作方式，初期以為 collapse 版本效能會比 一般 parallel for 效能好;但實際實驗並非如此。Collapse 將迴圈攤平 給不同的thread執行，反而增加了thread溝通上與傳遞資料的問題，造成效能不如預期。

另外此次作業也學習到如果想要多個phase同時進行，必須解決data dependency問題， 也是hw4會時做到的如何將distance matrix 拆成多個 Block運算。