Introduction to Massive Data Analysis

Homework 3 Report

Student ID: 105062635
Name: 吳浩寧

1. INTRODUCTION

這次的作業要使用 MapReduce 來實作常用的分群演算法 K-Means。首先要決定將所有的點分成幾(k)群，此時，有幾種選擇初始點的方式，分別為從所有點完全隨機選 k 個；或先選隨機選一點，之後每輪從剩下的點中，找到離目前已選擇的點最短距離最大者；課本中還提到，可以先取出一部分的樣本來作分群，再選出各群 centroid 最為所有資料點的初始點。而這次提供的測資 c1、c2 分別對應到前兩種方式，k 都等於 10。

接著便將每個點分配到離自己最近的 cluster，最近的定義為點與 cluster centroid 的距離最小，距離的計算則必須分別使用 Euclidean distance 與 Manhattan distance 來計算。全部的點分配完即可更新 cluster centroid 的位置，並反覆的執行以上步驟，基本上從大致的走向可以看出衡量分群效果的 cost 漸漸降低。

	Euclidean	Manhattan
distance	$\sqrt{\sum_{i=1\sim d}(a_i - b_i)^2}$	$\sum_{i=1\sim d}\lvert a_i - b_i\rvert$
cost	$\sum_{x\in X}\min_{c\in C}(x-c)^2$	$\sum_{x\in X}\min_{c\in C}\lvert x-c\rvert$

2. IMPLEMENTATION

可以指定的相關變數如下：

POINTS	點的數量	**DIM**	點的維度
MAX_ITER	執行幾輪		

1.) Job 1

Map Function:

接收兩個 input 檔案，分別為所有點的位置，和目前所有 clusters 的位置。如果資料來自所有點，則送出(pid, position)，pid 為自己這點的編號，從第一行編到最後一行，為了和來自 cluster 的資料區分，將 position 前面加上一個字元 D 以供辨識；若資料來自 cluster，則必須對每個 pid 送出(pid, cid + position)，cid 為 cluster 編號，也是從檔案第一行開始編。

input		output	
(point file)	(cluster file)	0 D 0 1 0	1 0 1 2 1
0 1 0	1 2 1	1 D 1 2 1	1 1 4 5 7
1 2 1	4 5 7	2 D 4 5 6	2 0 1 2 1
4 5 6		3 D 4 5 7	2 1 4 5 7
4 5 7		0 0 1 2 1	3 0 1 2 1
		0 1 4 5 7	4 1 4 5 7

Reduce Function:

主要計算距離的地方，根據收到的資料是否有'D'判斷是 centroid 還是 point，先把所有的 centroids 存到 ArrayList 中，才一一比較該點與所有 centroids 的距離，在這裡只須比大小，所以即使計算 Euclidean distance 也不需進行開根號的動作。找出距離最短的 centroid 後，會以屬於哪個 cluster 作為 key，以該點位置作為 value，送出(cid, position)供第二階段更新 centroid 位置用，送出("COST", distance)用來計算每一輪的 cost。

output (Euclidean distance)	
0 0 1 0	COST 3
0 1 2 1	COST 0
1 4 5 6	COST 1
1 4 5 7	COST 0

2.) Job 2

Map Function:

純粹將每一行切成(key, value)送到 reducer

Reduce Function:

若收到的 key 為"COST"，則將所有的值加起來計算最終的 cost；若收到的 key 為 cluster id 則計算收到的所有 pairs 數量，並將所有維度的值分別作平均已得到更新後的 cluster centroid。

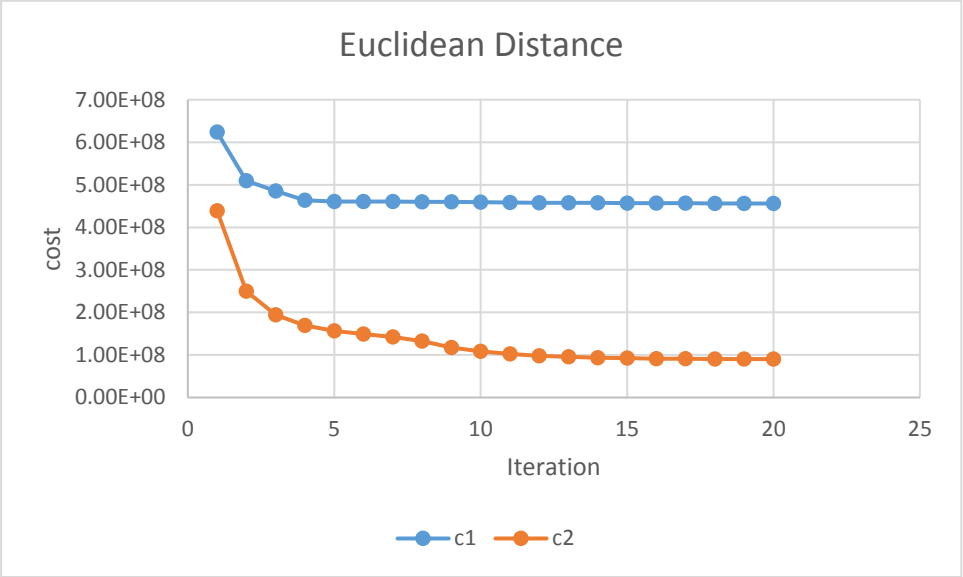
output
0.5 1.5 0.5
4.0 5.0 6.5
COST 4

main()

在主程式裡除了循環執行 job 1~2，基本上和之前大同小異，我每一輪會另外讀出書出檔的 COST，存到 ArrayList，最後再一起輸出。也根據作業要求，讀取最後一輪的輸出，並計算每個 centroids 兩兩間的距離。

3. RESULT

(a) Initialization Strategies with Euclidean Distance

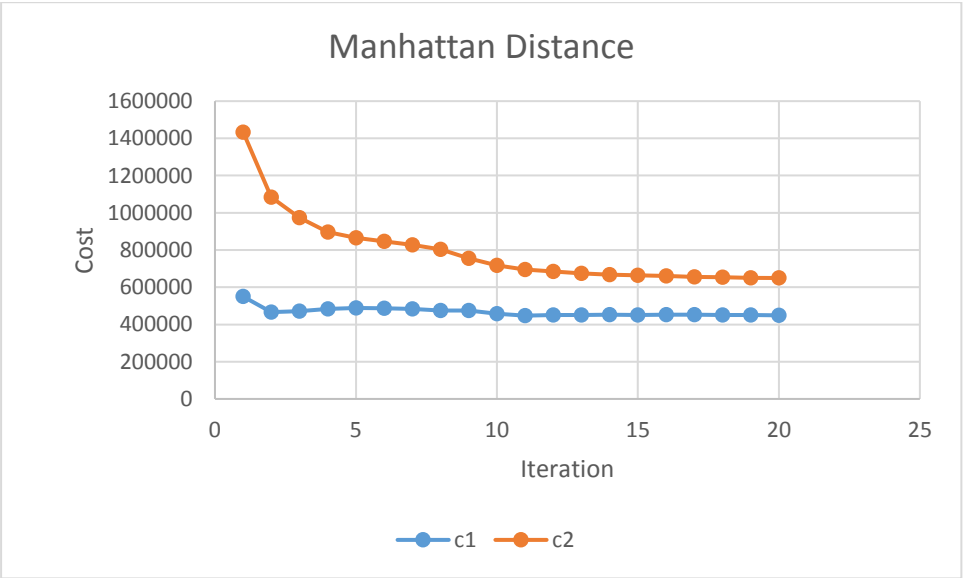


10 輪後 c1 improvement: (6.2366-4.5849)/6.2366=26.48%

10 輪後 c2 improvement: (4.3875-1.0224)/4.3875=76.69%

從圖形中可以看出，兩者的曲線皆是嚴格遞減的，而 c2 無論在 cost 降低的比例和 cost 本身的大小都比 c1 佳，由於 c2 之間每個點彼此初始距離就非長遠，因此很大的機率彼此屬於不同的 cluster，不會產生像隨機選點時，2 個 cluster centroids 可能非常近，導致所有點與他們的距離都差不多，因此在每一輪都不斷分到另一個 cluster，容易遇到瓶頸收斂不到較低的 cost。

(b) Initialization Strategies with Manhattan Distance



10 輪後 c1 improvement: (55012-44749)/55012=18.65%

10 輪後 c2 improvement: (14337-6946)/14337=51.55%

從圖形中可以看出，兩條曲線的收斂程度都較 Euclidean Distance 低。此外 c1 的 cost 都比 c2 低，且一直維持在差不多的值，甚至在中間稍微上升，c2 的 cost 下降的比例仍較多。雖然作業中沒說明，但 c2 找初始點用的距離，應該是 Euclidean Distance，雖然差不多可能間接導致

本次作業其它結果於附檔中：
result.txt 內容為各種組合每一輪的 cost，和最後 10 個 cluster centroids 兩兩之間的距離；Euclidean_c1, Euclidean_c2, Manhattan_c1, Manhattan_c2 則分別為各種組合最後得到的 10 個 cluster centroids。

4. EXPERIENCE

這次作業比較沒用到什麼新的技巧，主要須思考怎麼把作業要求的資料，透過程式整理出來，和想一些簡單的測資確認自己計算的結果。