雲報專欄：雲端運算環境上的資料叢集分析—黃仁暐/成功大學電機工程學系助理教授、陳銘憲/技術專家委員會委員/中央研究院資訊科技創新研究中心主任/台灣大學電機工程學系教授

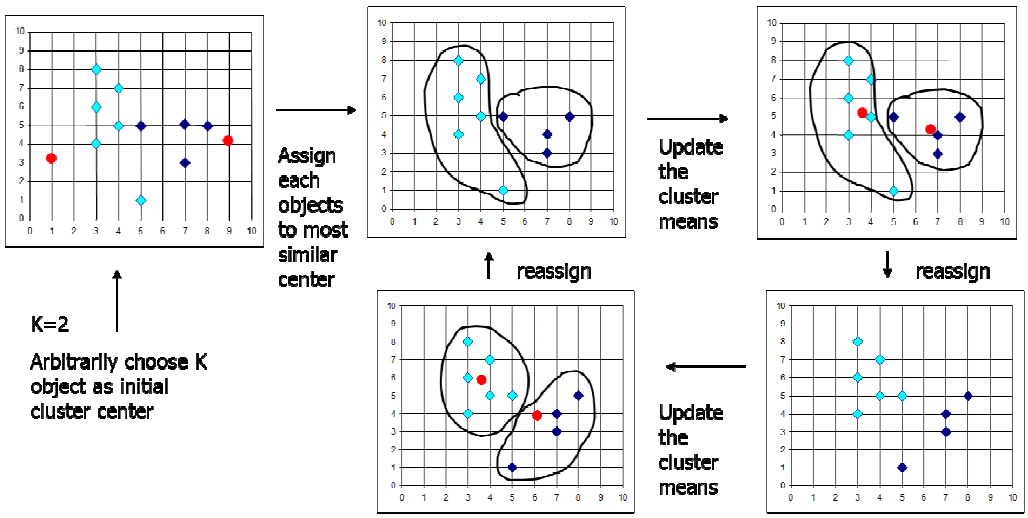
資料叢集(data clustering)技術旨在研究資料與資料之間的相似度，其主要目的為將相似的資料物件聚集在一起而形成叢集，在相同叢集中之資料物件彼此較為相似，而在不同叢集中之資料物件則與其他資料較為相異。資料叢集技術在資料探勘(data mining)領域已經被大量地討論，是資料探勘領域中最重要的研究之一[9]。其應用亦十分廣泛，如在近年來被廣大民眾使用的社群網站(Facebook, Twitter 等[註 1])中，藉由分析使用者的興趣或彼此之間的行為交流，可找出相似類型的使用者並形成群體，這類由相似使用者組成的叢集可提供網路行銷和市場分析許多有用的資訊。隨著數位化技術的演進，資料產生的速度和方式亦日新月異，除了一般使用者產生的資料之外，還有為了紀錄訊息而由機器產生的資料，甚至有因為在社群網站上使用者的互動而產生的大量資訊，這些大資料(Big data)的截取、儲存、分析和呈現都是一項困難的挑戰，在資料叢集演算法的設計中，不可避免地會需要計算資料物件彼此之間的相似度，這種動作將會需要大量的計算時間或空間，在單台機器上運算的傳統演算法可能會受到有限的資源限制，如記憶體不足或計算能力不夠的問題，而導致無法有效率地找出有用的資料叢集。雲端運算(cloud computing)的架構正好可以提供大量的計算資源和儲存空間，用來解決傳統資料叢集演算法碰到大資料時的記憶體不足或計算能力不夠的問題。本文將探討傳統資料叢集演算法設計上的概念，以及提供如何將傳統演算法實現在雲端運算環境的想法。

為了解決大量資料處理的問題，有些研究者提出採用平行化或分散式的架構來計算資料叢集中大量的資料。平行化的計算將資料切割成較小的區塊，由多個計算單元同步進行部分的運算，而分散式的運算則將資料傳送到不同的機器，每台機器藉由傳遞訊息與資料共享的方式完成各自的子任務，再將各部分合併成最終的結果[8][10]。然而，不論是過去的平行化和分散式的架構，演算法的設計者都必須要由自己控制資料的分配、資料同步、訊息的傳遞、工作的分配、負載的平衡和故障恢復的工作，這些任務使得傳統的平行化和分散式系統不能很快地擴展規模，且每當有系統狀態改變時都需要重建計劃以處理這些變化，而增加了開發高效率演算法的困難度和實用性。所幸，許多雲端運算的環境，如 Hadoop[註2]適時地解決了上述的問題，並且能同時達到平行化與分散式架構的優點，我們所需注意的就是減少訊息的傳遞與合併結果的成本，使得雲端環境中的資料叢集演算法變得較為容易設計與更為實用。

傳統的資料叢集演算法大致上可以分為以下七類：

1. Partitioning approach:

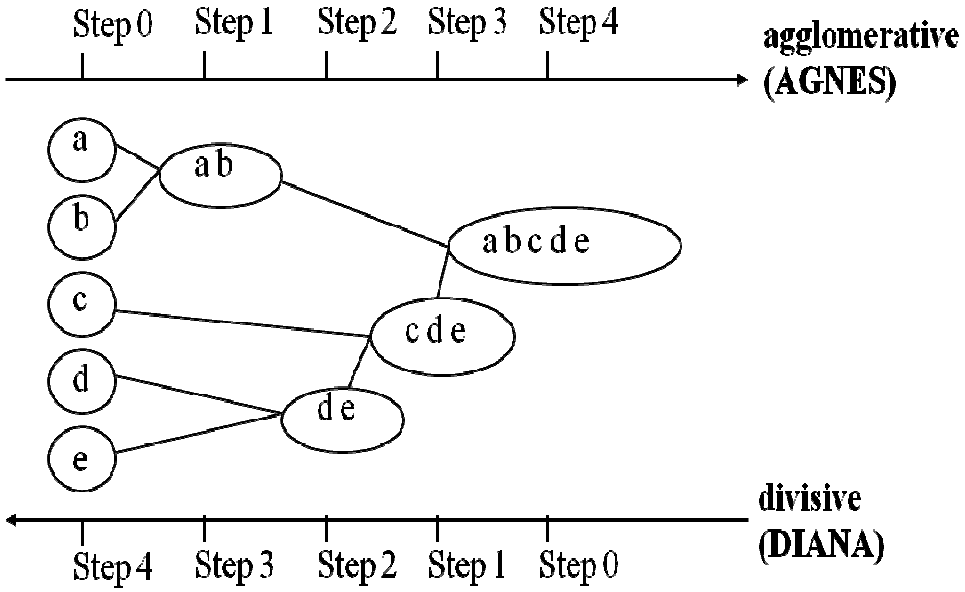
利用將資料集切割的概念，將資料物件以本身的屬性分佈來分割成多個叢集，接近的資料將被分割在同一個叢集中，分割完之後之叢集再作細部的調整，改變資料物件歸屬的叢集編號，以達成最佳化的分割方式，如圖一所示，此類方法中最為知名的為 K-means[4] & K-medoids[11]。在雲端環境中實現此類方法，主要著重在將運算量最大的資料物件相似度的比對，平均分散到各個運算單元去，且利用資料的切割方式將共同所需的叢集資訊傳到各運算單元，以便各單元進行獨立的運算，減少訊息傳遞的數量。在此方法中不需要合併結果的計算，雲端運算環境對於效能的改進十分有效[13]。



圖一　Partitioning approach [7]

2. Hierarchical approach:

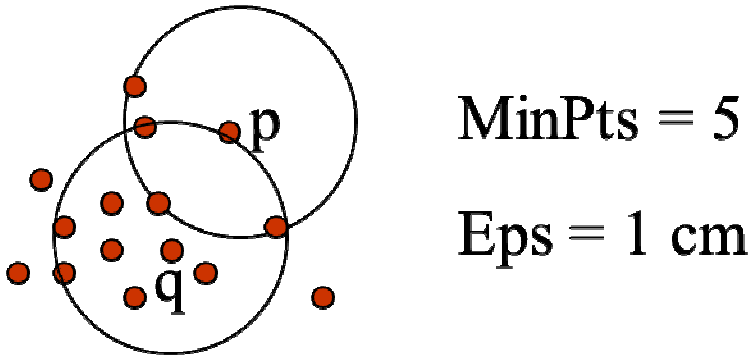
使用階層式的尋找叢集方法，有 agglomerative 與 divisive 兩種方式，如圖二所示。agglomerative 為由小到大將相似的資料物件一步一步聚集成較大的群組，直到所有資料物件皆屬於相同群組為止；而 divisive 則由大到小將整個資料集作切割，將相異的資料物件分至不同的群組當中，直到所有資料物件皆分開為止。在雲端環境中實現時，需考慮兩兩資料點間相似度的計算，儘可能將工作分配到各運算單元中，並減少傳輸資料的重覆率，也可使用多次的運算來達到相似度的計算[16]。



圖二　Hierarchical approach [7]

3. Density-based approach:

分析資料集中資料物件分佈的密度，將資料物件分佈密度較高之處標計為高密度區域，並將相連之高密度區域結合成完整的叢集。如圖三所示，先以 q 點為圓心設定區域範圍 Eps，檢查範圍內之資料密度是否達到門檻值 MinPts，若高於門檻則將其與相臨之高密度區域結合成叢集，此即為知名的 DBScan 演算法[5]。在雲端運算環境中需考慮的主要項目為：資料分割、局部叢集與結果合併。資料分割的目標是將原有的資料集分割成數個區塊，每個區塊包含差不多數量的資料，為了減少之後的合併成本，相類似的資料物件將被分入相同的區塊中，此外，通常在分割資料時，會將各區塊部分重疊，以減少作局部叢集動作時所需的訊息傳遞量，藉由這些重疊的資料物件，合併的成本亦可顯著地降低。在資料分割成數個區塊後，各區塊會被分送至雲端環境中之各運算單元，各運算單元獨立地執行局部的叢集方法，且不需要與其他運算單元進行溝通，由於相類似的資料物件被放入同一個區塊中，任何基於密度的資料叢集演算法都可以應用在局部叢集中。在各運算單元計算出局部叢集的結果後，將此部分結果傳回主系統中，主系統即可進行最後合併階段的分析，並得到最後的叢集結果[4]。



圖三　Density-based approach [7]

4. Grid-based approach:

此類方法將資料物件分佈的空間直接切割成一定大小的格子，再計算每一個格子空間中所有的資料物件數量，並將相連的高密度格子合併成叢集。此類切割格子方法較上述方式節省許多時間，但叢集的邊界則被限制為只能落在格子的邊界，降低叢集的準確性。較著名的方法有 STING[14]及 CLIQUE[1] 。此方法在雲端環境中亦需要資料分割、局部叢集與結果合併三個步驟，資料分割時將原始資料集中的資料分散到各個格子點中，並將同一格子點之資料傳給同一運算單元計算密度，將高於密度門檻之小叢集傳回主系統，最後主系統合併相鄰之格子點即可得到結果。

5. Frequent pattern-based approach:

將資料物件出現的空間位置轉換成交易資料中的項目，再使用頻繁樣式探勘(frequent pattern mining)[2]的方法為基礎，找出常常一起出現的資料的空間位置，並將此頻繁出現的空間視為一個小叢集，再連結相鄰的小叢集形成完整的叢集，如 pCluster[15]。要將此方法實作在雲端環境中，只需採用雲端頻繁樣式探勘的方法[12]即可簡單達成，另可多加入分散式資料轉換的設計，可進一步提升其效能。

6. Model-based approach:

使用某些特定數學運算式的模型，在資料集與模型之間找出最適合的對應，基於所對應到的模型，可藉由資料的機率分佈找出叢集所在的位置。EM[3]與COBWEB[6]為較著名之方法。在雲端環境中要實現此類方法，需取決於模型是否容易分散，或是否可將各資料點分散式與模型比對。

7. Hybrid approach and others:

另外尚有些資料叢集演算法會採用混合式的模型或非上述所列的方式。而其雲端環境上之實作方式則需針對個別演算法作特別設計。

未來隨著資料量的增長，資料叢集分析的需求將會益趨重要，而雲端運算環境的進步也能提供更多我們設計資料叢集演算法時的概念，相信此議題尚有許多等待我們研究發展的空間。

[註 1] Facebook: http://www.facebook.com

Twitter: http://twitter.com

[註 2] Hadoop: http://hadoop.apache.org

References:

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications", In Proceedings of the ACM SIGMOD Conference, 94–105, 1998.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases", In Proceedings of the 20th International Conference on Very Large Data Bases, 478–499, 1994.

[3] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm". Journal of the Royal Statistical Society. Series B (Methodological) 39 (1), 1–38, 1977.

[4] B.-R. Dai and I.-C. Lin, "Efficient Map/Reduce-Based DBSCAN Algorithm with Optimized Data Partition", IEEE International Conference on Cloud Computing, 59–66, 2012.

[5] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise", In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, 226–231, 1996.

[6] D. H. Fisher "Knowledge Acquisition Via Incremental Conceptual Clustering", Machine Learning vol.2 issue2, 139–172, 1987.

[7] J. Han, M. Kamber, and J. Pei. "Data Mining: Concepts and Techniques, 3/e", 2011.

[8] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks", In Proceedings of the European Conference on Computer Systems, ACM, 59–72, 2007.

[9] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review", ACM Computer Survey, 31:264–323, September 1999.

[10] E. Januzaj, H.-P. Kriegel, and M. Pfeifle, "Scalable Density-Based Distributed Clustering", In The 15th European Conference on Machine Learning and the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, vol. 3202, 231–244, 2004.

[11]L. Kaufman and P.J. Rousseeuw, "Clustering by means of Medoids", In Statistical Data Analysis Based on the L1–Norm and Related Methods, 405–416, 1987.

[12] KW Lin and YC Luo, "Efficient strategies for many-task frequent pattern mining in cloud computing environments", In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 620–623, 2010.

[13] A. Mahendiran, N. Saravanan, N. Venkata Subramanian and N. Sairam, "Implementation of K-Means Clustering in Cloud Computing Environment", Research Journal of Applied Sciences, Engineering and Technology, 4(10):1391–1394, 2012.

[14]W. Wang, J. Yang, and R. Muntz, "STING : A Statistical Grid Approach to Spatial Data Mining", 186–195, 1997.

[15]H. Wang, W. Wang, J. Yang, and P.S. Yu, "Clustering by pattern similarity in large data sets", In Proceedings of the ACM SIGMOD Conference, 394–405, 2002.

[16] S. Wang, H. Dutta, "PARABLE: A PArallel RAndom-partition Based HierarchicaL ClustEring Algorithm for the MapReduce Framework", 6th Annual Machine Learning Symposium at the New York Academy of Science (NYAS), 2011