Hierarchical clustering On Spark for Wireless Communication Network

Team Member ： Siqi Huang ID: 800953935

- Overview of the project:

  In this project, I want to cluster the base stations in a certain area into multiple clusters. In this case, the base stations in different clusters can be managed separately. Thus, the network performance and system management efficiency can be improved. Since the volume of the original data is very large. Big data process platform is needed. I use Spark to implement a Hierarchical Clustering Algorithm (HCA) to clustering Base stations in wireless network.

  - Task1：Network data preprocessing

    Load the original data set, clear the conflict, missing and duplicated data records. And filter the privacy related data items.
  - Task2：Feature extraction

    Collecting the data records for the base stations in the certain area. Then filter out the factors which are used to calculate the distance between the base stations.
  - Task3：HCA algorithm design

    Implement the Hierarchical Clustering algorithm on Spark with additional packages.
  - Task 4：Performance evaluation

    Evaluate the performance of the clustering on Spark. The main factor is the speed of the data processing and clustering procedure.
  - Result：Network clustering solution.

- Context and motivation of the project

  The mobile network data are growing explosively with the proliferation of mobile devices and bandwidth-hungry applications, and have led to a continuous surge in capacity demand across mobile networks. As a result, base stations (BSs) are being densely deployed to provide high capacity, low latency, and seamless service coverage in mobile networks. The density of BSs is expected to be comparable to that of mobile devices, thus resulting in ultra-dense mobile networks. The network densification together with advanced transmission technologies alleviates spectrum shortage and promises high network capacity. Therefore, ultra-dense mobile networking is essential to 5G and future mobile networks.

  A major challenge in provisioning ultra-dense mobile network is the increased complexity of networking mechanisms such as the traffic load balancing and BS sleep control, which notably grow with the large number of BSs. Such rapid growth in complexity stifles existing networking mechanisms for ultra-dense mobile networks. For example, the traffic load balancing problem is NP-hard. With the large number of BSs, these suboptimal mechanisms

can incur long computing time and high communication overhead so that they cannot efficiently balance traffic loads in ultra-dense mobile networks. An effective solution to mitigate the complexity growth is partitioning the entire RAN into multiple autonomous sub-RANs. Each sub-RAN consists of a small number of BSs such that it can be efficiently managed by existing networking mechanisms.

Since the volume and the variety of data are very large and complex. An efficient data process platform is essential. Spark is worth to try.

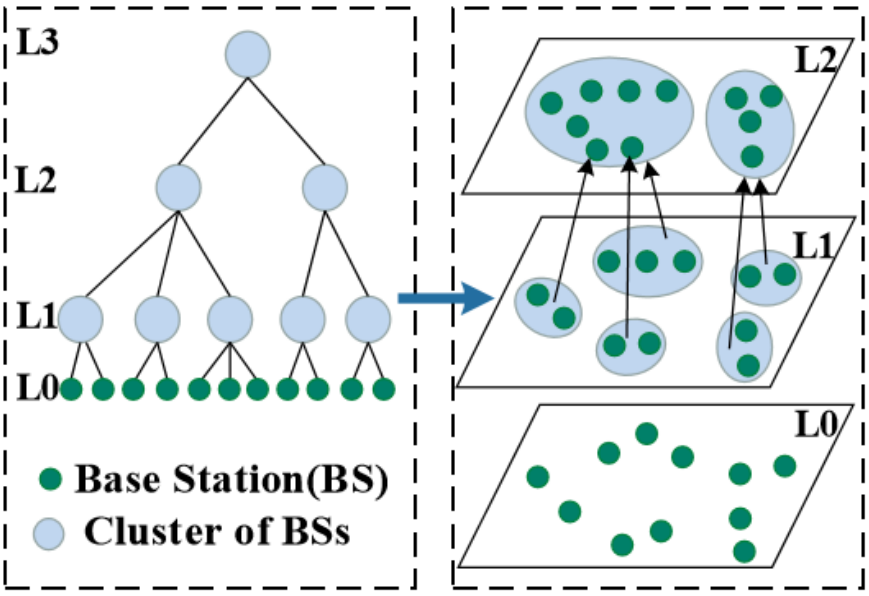
- Data sets

  The data sets are the real mobile network records which consist of two-week mobile traffic data (220 GB and 2.8 billion records) collected from an operating mobile network with about 10000 BSs and 500000 users. In this project, the geographical location of the base stations are the feature used to measure the distance between base stations.

- Algorithm

  I use the Hierarchical clustering algorithms to cluster the base stations. The advantage of HCA over other clustering algorithms such as K-means clustering is that it does not require a priori knowledge of the number of clusters. Besides, HCA is flexible to adjust the number of clusters for different network scenarios. In the network partitioning, the number of sub-RANs cannot be predefined due to mobile traffic dynamics.

  At the beginning, each BS is treated as an individual cluster. Based on the distance function, the tightness of the coupling between clusters can be evaluated. A shorter distance indicates a tighter coupling between the clusters. Based on the distances between clusters, the HCA algorithm iteratively merges clusters with the shortest distance into a new cluster. After each round of merging clusters, the distances between clusters are recalculated. The merging process continues until all BSs are in one cluster.

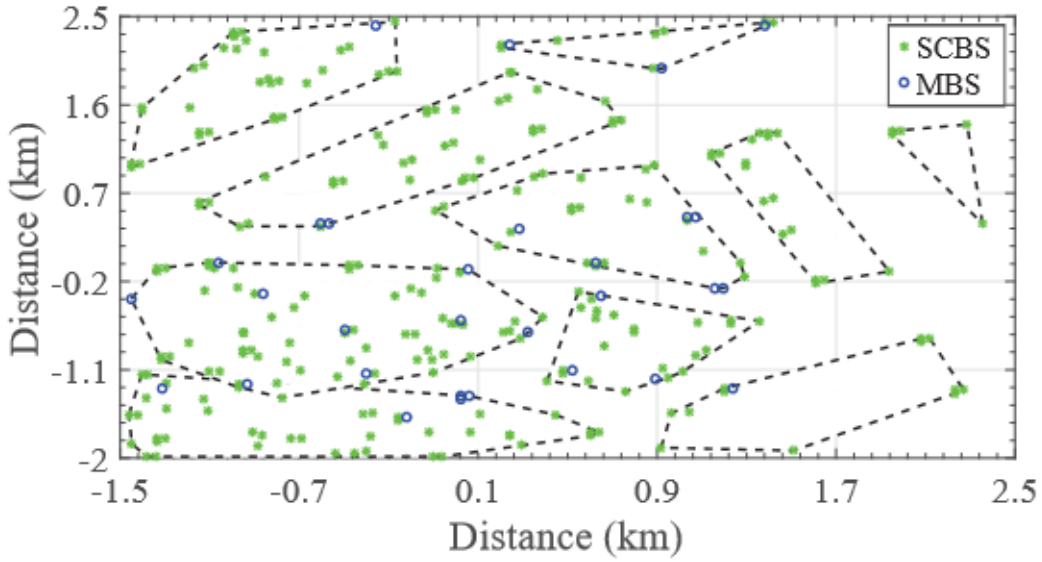


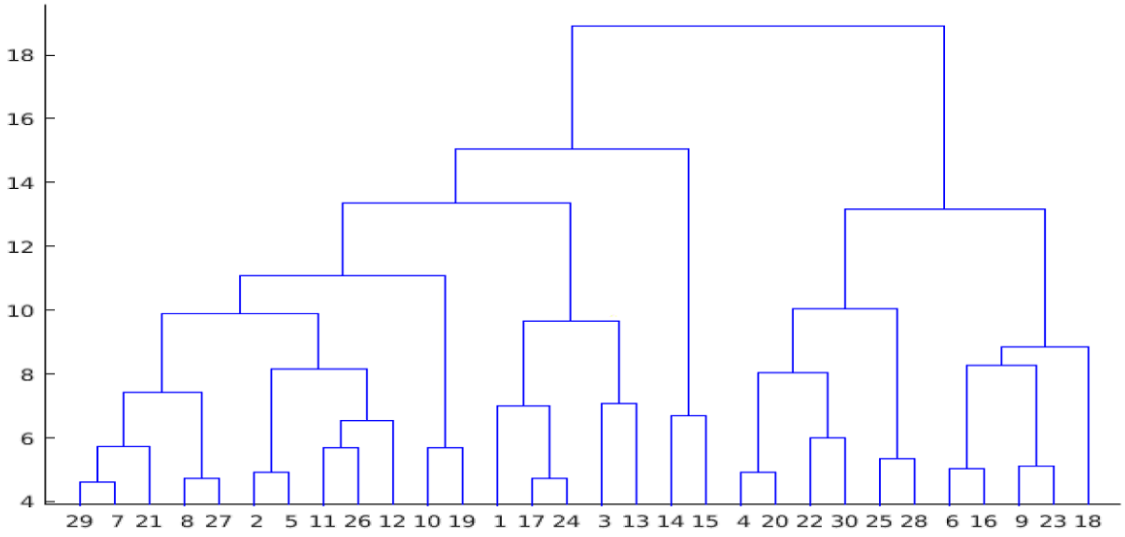
- External tools and software packages.

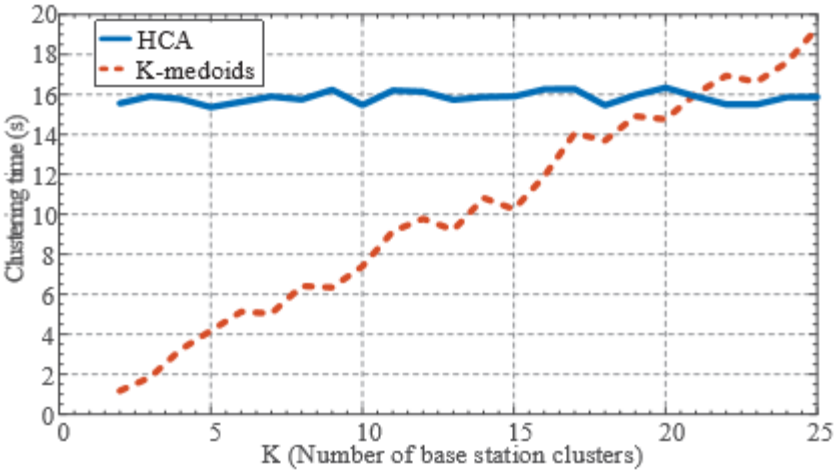
  Three packages are used to provide basic math calculation and graph plot functions.

  ```
  numpy
  matplotlib
  scipy
  ```

- Result






HCA
K-medoids
Clustering time (s)
K (Number of base station clusters)
0
2
4
6
8
10
12
14
16
18
20
0
5
10
15
20
25