Data-Driven Network Optimization in Ultra-Dense Radio Access Networks

Siqi Huang†, Qiang Liu†, Tao Han†, and Nirwan Ansari‡

†Department of Electrical and Computer Engineering, The University of North Carolina at Charlotte
‡Department of Electrical and Computer Engineering, New Jersey Institute of Technology
†{shuang9, qliu12, tao.han}@uncc.edu, ‡nirwan.ansari@njit.edu

***Abstract*—The complexity of networking mechanisms will increase significantly because of the dense deployment of radio base stations in ultra-dense mobile networks. As a result, the existing networking mechanisms may be unable to efficiently manage ultra-dense mobile networks. To solve this problem, we propose a data-driven network optimization framework which integrates the big data analysis methods with networking mechanisms. In the proposed framework, we adopt big data analysis methods to divide densely deployed base stations into groups. Then, each group of base stations are managed with networking mechanisms independently. In this way, the complexity of the networking mechanisms is reduced. The key challenge in designing the framework is to optimally group base stations into clusters in realtime. Addressing this challenge, the proposed framework consists of an offline machine learning module and an online base station clustering and network optimization module. The offline machine learning module predicts the optimal number of base station groups in the next time interval based on the historical data. The online base station clustering and network optimization module clusters base stations and optimize the network in realtime. The performance of the proposed data-driven network management framework is validated through network simulations with real network data traces.**

I. Introduction

The mobile network data are growing very fast with the proliferation of mobile devices and applications [1]. In order to carry the ever-increasing mobile data, radio base stations are being densely deployed to provide high network capacity [2], [3]. Although the network densification promises high network capacity, it increases the complexity of networking mechanisms such as the traffic load balancing and BS sleep control because of the large number of base stations [4], [5].

The rapid growth in complexity stifles existing networking mechanisms for ultra-dense mobile networks [6]. For instance, the computational complexity of the suboptimal traffic load balancing mechanisms can be estimated to be $O(M^a N^b)$ for some nonnegative integer a and b, where M and N are the numbers of users and BSs, respectively [7]. When base stations are densely deployed, these networking mechanisms can incur long computing time and high communication overhead [4], [8], [9]. Therefore, there is an urgent need to design efficient networking mechanisms for ultra-dense networks.

This work is partially supported by the US National Science Foundation under Grant No. 1731675.

An effective method to mitigate the complexity growth is to divide base stations into groups and manage each group of base stations independently [7], [10]. However, the realtime base station grouping is challenging for a few reasons. First, the relationships among base stations cannot be accurately quantified based on the geographical distances [7]. Hence, it is non-trivial to define the distance function that quantifies the relationships among base stations for the grouping. Second, owing to the dynamic mobile traffic, it is difficult to learn the optimal number of base station groups at different time intervals. Third, the tightly coupling of the base station grouping and the network performance requires an integrated base station grouping and network optimization.

Big network data analysis and machine learning are being applied to manage and optimize communications networks [7], [11]–[13]. However, most of research works on big-data-driven mobile networks focus on developing big mobile traffic data monitoring and analyzing systems [11], [12] and discovering network usage and user behavior patterns through analyzing big mobile traffic data [13]. We have designed a network partitioning method based on the analysis of a mobile traffic dataset containing billions of mobile traffic records generated from an operating mobile network [7]. In designing the network partitioning method, we have validated that the complexity of the networking mechanisms can be significantly reduced at the cost of a small network performance decline [7]. The network partitioning method, however, is an offline method which is not appropriate for realtime network management and optimization.

In this paper, we propose the data-driven network optimization (DINO) framework which integrates data analysis and machine learning methods with network optimization algorithms to enable realtime and efficient ultra-dense network optimization. The proposed framework is composed of an offline machine learning module and an online base station clustering and network optimization module. The offline module mines mobile traffic dataset and performances base station clustering analysis using the hierarchical clustering analysis (HCA) method [14]. The analysis results are utilized to train an artificial neural network (ANN) for predicting the optimal number of

978-1-5090-5019-2/17/$31.00 ©2017 IEEE

base station groups at different time intervals. Based on the prediction, the online module adopts the K-medoids clustering algorithm to partition base stations into groups and then optimize the performance of each group of base stations independently. To evaluate the performance of the DINO framework, we set up data-trace driven network simulations. The simulation results show the performance of the offline prediction and the online clustering and optimization and reveal the impact of the base station grouping on the computation complexity and performance of the network optimization.

The rest of the article is organized as follows. Section II presents the overview of the data-driven network management framework. Section III presents the offline machine learning module in detail. Section IV presents the online base station clustering and network optimization module. Section V shows the performance evaluation. We conclude the paper in Section VI.

II. Framework Overview

In this section, we present an overview of the data-driven network optimization (DINO) framework as illustrated in Fig. 1. In order to reduce the complexity of the network optimization mechanisms, the DINO framework partitions base stations into groups and optimizes the performance of base stations in each group independently. The DINO framework consists of an offline machine learning module and an online base station clustering and network optimization module.

A. Offline Machine Learning

Since the coupling of the base station grouping and the network performance, it is highly complex to obtain optimal base station grouping. To enable the optimal base station grouping in realtime, the offline machine learning module aims to learn the relationship between the optimal number of base station groups and the aggregated mobile traffic in individual base stations. Based on the learned relationship, the offline module predicts the optimal number of base station groups at different intervals. The offline machine learning module consists of four major components: mobile traffic data collection, networking features extraction, network performance aware clustering analysis, and ANN-based cluster number prediction. The mobile traffic data in each base station are collected and stored in a traffic record database. These data are used for the clustering analysis and the ANN-based cluster number predictor. In order to perform the clustering analysis, three networking features are extracted from the traffic record database. Based on these networking features, we define a distance function to quantify the relationships between base stations. The hierarchical clustering analysis (HCA) is adopted to obtain a clustering tree which contains the optimal base station clusterings for all possible cluster numbers. For example, if the optimal cluster number is K, the clustering solution in the K level of the tree is the optimal base station clustering solution.

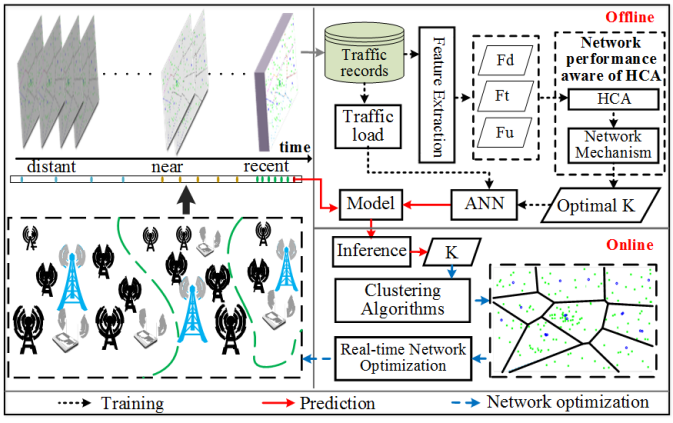

Fig. 1: The data-driven network optimization framework.

To determine the optimal base station clustering, we adopt a traffic load balancing algorithm[1] as the networking management mechanism to evaluate the performance of different base station clustering strategies. Denote $\mathcal{B}$ and $\mathcal{U}$ as the set of BSs and users in the network, respectively. Let R_i and $\hat{R}_i$ be the ith user's data rate before and after the base station clustering, respectively. With a given traffic load balancing algorithm, the networking performance of the base station clustering solution is defined as $\mathcal{P} = \frac{\sum_{i\in\mathcal{U}} U(\hat{R}_i)}{\sum_{i\in\mathcal{U}} U(R_i)}$. Here, $U(\cdot) = log(\cdot)$ is the utility function and the R_i is the ith user's data rate [4]. Denote τ and $\hat{\tau}$ as the computing time of the traffic load balancing algorithm before and after the base station clustering, respectively. The computation performance of the base station clustering solution is defined as $\mathcal{V} = \frac{\hat{\tau}}{\tau}$. Considering both the networking and computation performance, we define $\psi = (1-\omega)\mathcal{P}+\omega(1-\mathcal{V})$ as the performance function of the base station clustering solution. Here, $0 < \omega < 1$ is a parameter which can be selected by the network operator. Given ω, the clustering solution that maximizes ψ is the optimal base station clustering. Then, the optimal number of base station clusters under given traffic conditions are learned.

The clustering analysis and the clustering performance evaluation have very high computation complexity and are not appropriate for realtime clustering and network optimization. In order to achieve the optimal base station clustering in realtime, we have to predict the optimal number of base station clusters based on the traffic conditions. Hence, we use the derived optimal base station clustering solution and the corresponding traffic conditions as input data to train an artificial neural network and obtain the model for predicting the optimal number of base station

[1]Note that different networking mechanisms may be implemented to evaluate the clustering performance. The optimal clustering solution may be different when the clustering performance is evaluated under different networking mechanisms. Owing to the page limit, we only discuss the clustering solution under the traffic load balancing scheme.

clusters under a given traffic condition. The online clustering and network optimization module use this prediction model for the realtime base station clustering.

B. Online Clustering and Network Optimization

Given the ANN-based prediction model, the optimal number of base station clusters can be derived based on the current traffic condition. Then, base stations can be clustered into groups based on low-complexity clustering algorithms such as K-means and K-medoids [15]. Once the base stations are clustered into groups, each group of base stations can be optimized independently by networking mechanisms, e.g, the traffic load balancing.

III. Offline Machine Learning

In this section, we present the design of the offline learning module consisting of the mobile traffic data collection, networking feature extraction, network performance aware clustering analysis, and the ANN-based cluster number predictor.

A. Mobile Traffic Data Collection

In the DINO framework, mobile traffic records are continuously collected and stored in the traffic record database. A mobile traffic record include the identification number of a base station, the geographical location of the base, the identification number of a user, the start and end time of the user's traffic in the base station.

B. Networking Feature Extraction

To cluster base stations into groups, a distance function is required to quantify the relationship between the base stations. Therefore, we extract three networking features from the mobile traffic record data to define the distance function the clustering analysis.

1) F_d: The geographical distance between two base stations can reflect whether they are tightly coupled. When the base stations are close to each other, their coverage areas will be overlapped. A larger overlap area may indicate a tighter coupling. We define x_i and y_i as the latitude and longitude of BS_i, respectively. φ_i and λ_i are the corresponding radians of the latitude and longitude. R is the radius of the earth. The geographical distance between the ith and jth base stations is expressed as:

$$d_{ij} = \arccos(\sin\varphi_i \sin\varphi_j + \cos\varphi_i \cos\varphi_j \cos\Delta\lambda)R, \quad (1)$$

where $\varphi_i = \frac{x_i \pi}{180}$ and $\Delta\lambda = \frac{y_i \pi}{180} - \frac{y_j \pi}{180}$.

2) F_t: The mobile traffic trend measures the mobile traffic diversity over time in a base station. If two base stations are tightly coupled, they will have similar traffic trends. We define the duration of a time slot to be 10 minutes. Then, a day consists of 144 time slots. To derive the traffic trend, we calculate the accumulative traffic load in a base station in each time slot. Denote $\gamma_{i,j}$ as the accumulative traffic load in the ith base station in the jth time slot. Then, $\Gamma_i = \{\gamma_{i,1}, \gamma_{i,2}, \cdots, \gamma_{i,144}\}$ represents the traffic load sequence in the ith base station. The traffic trend similarity between base stations can be quantified by using the discrete Fréchect (DF) distance [16], which can calculate the similarity of two curves regardless of the shift, scale, and rotation of the curves. Denote $\delta_{i,j} = FrechetDist(\Gamma_i, \Gamma_j)$ as the DF distance between the ith and jth base stations. A small $\delta_{i,j}$ indicates that the ith and jth base stations have the similar traffic trend.

3) F_u: A common active user in base stations is defined as the user whose data traffic appears in all these base stations during a certain time period, e.g., 10 minutes. A large number of common active users in base stations may indicate that active users frequently handover among these base stations, implying that these BSs are tightly coupled. Therefore, we analyze the mobile traffic data and derive the number of common active users of any two base stations.

Based on the three network features, the distance function that quantifies the relationship between the ith and jth BSs is expressed as $D_{i,j} = \sum_{k=1}^{N} \alpha_k d_{i,j}^k$. Here, N is the number of features used in the distance calculation; α_k is the weight of the kth feature; $d_{i,j}^k$ is the distance calculated based on the kth feature. The optimal value of α is derived in the clustering analysis.

C. Network Performance Aware Clustering Analysis

The objective of the network performance aware clustering analysis is to derive the optimal base station clusterings under different mobile traffic conditions such that the analysis results can be utilized for training the ANN-based cluster number predictor. The cluster analysis is performance in two steps.

First, given the distance function, we adopt the hierarchical clustering analysis method to generate a cluster tree which contains the optimal base station clustering solutions for all possible cluster numbers [14]. At the beginning of the HCA method, each base station is treated as an individual cluster. Define $\mathcal{D}(C_m, C_n) = \min_{i \in C_m, j \in C_n} D_{i,j}$ as the distance between the base station cluster C_m and C_n. Hence, the distance between two base station clusters reflects the minimum inter-base stations distance for base stations in different clusters. 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 iteration, the distances between clusters are recalculated. The merging process continues until all BSs are in one cluster. Once the clustering tree $\mathcal{C}_T$ is derived, each level of the clustering tree represents a base station clustering solution. If the Kth level of $\mathcal{C}_T$ is selected, the base stations are clustered into K groups.

Second, we use the mobile traffic load balancing mechanism to evaluate the performance of the clustering solutions in terms of both the network and computation

performance [4], [7]. We execute the traffic load balancing mechanism on the entire radio access network without the base station clustering and obtain the network utility $\sum_{i\in\mathcal{U}} U(R_i)$ and computation time τ, respectively. For a clustering solution, we perform the traffic load balancing mechanism on individual groups of base stations independently. The corresponding network utility and computation time are derived as $\sum_{i\in\mathcal{U}} U(\hat{R}_i)$ and $\max(\hat{\tau})$, respectively. Heres, since the networking mechanism is concurrently executing on individual groups of base stations, we use the maximum computation time required in base station groups to represent the computation time after the base station clustering. Then, the network and computation performance of a base station clustering solution can be calculated as $\mathcal{P} = \frac{\sum_{i\in\mathcal{U}} U(\hat{R}_i)}{\sum_{i\in\mathcal{U}} U(R_i)}$ and $\mathcal{V} = \frac{\max(\hat{\tau})}{\tau}$, respectively. The overall performance of a base station clustering solution can be evaluated by $\psi = (1-\omega)\mathcal{P} + \omega(1-\mathcal{V})$. In the clustering analysis, we traverse the entire clustering tree to obtain the maximum ψ and the corresponding number of clusters. These analysis results are transferred to the ANN-based cluster number predictor for training the ANN.

D. ANN-based Cluster Number Predictor

The ANN-based cluster number predictor aims to predict the optimal number of base station clusters in realtime based on the traffic condition. In order to accurately predict the optimal number of base station clusters, we train the ANN with the traffic record data and the corresponding clustering analysis results. As illustrated in Fig. 2, the ANN-based cluster number predictor is composed of two parts: the input generation and the artificial neural network.

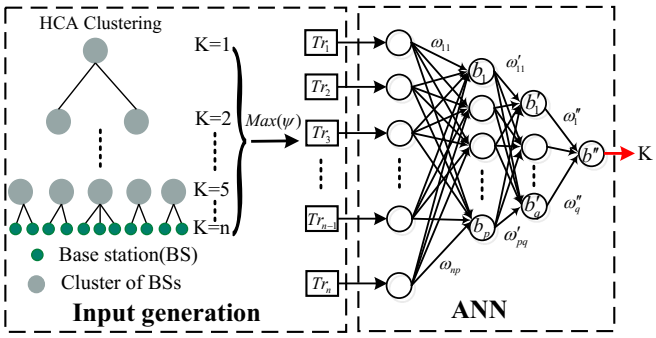

Fig. 2: Structure of Neural Network Predictor.

Input Generation: We define the duration of a time slot to be 10 minutes. Denote Tr_i as the accumulative traffic load in the ith BS in one time slot. Assume there are n base stations in the network. Then, we use $\Gamma = \{Tr_1, Tr_2, \cdots, Tr_n\}$ to represent the network traffic load in one time slot. The normalized Γ is one of the inputs to the ANN. Then, we use the network performance aware clustering analysis method to derive the optimal base station clustering solution and the corresponding cluster numbers at different time slots. The optimal cluster numbers act as the ground truth for the ANN training.

Artificial Neural Networks: We adopt the back-propagation neural network as the prediction algorithm. Our neural network consists of the input layer, two hidden layer and the output layer. We assign a sigmoid function $\varphi(x)$ for the neurons in the hidden layers and a purelin function $I(x)$ for the output layer. The output of the predication is expressed as

$$K = I(\sum_q \omega_q^{''} \varphi(\sum_p \omega_{pq}^{'} \varphi(\sum_n Tr_n \omega_{np} + b_p) + b_q^{'}) + b^{''}), \quad (2)$$

where $b^{''}$, $b_q^{'}$ and b_p are the biases associated with the neuron in different layers. $\omega_q^{''}$, $\omega_{pq}^{'}$ and ω_{np} donate the weights associated with the input of different layers.

The performance of the base station clustering depends on the distance function which quantifies the relationships between base stations. Since mobile traffic is highly dynamic, the distance functions should be revised according to mobile traffic conditions. The ANN-based cluster number predictor is trained with data derived based on the distance function (clustering analysis). Therefore, the prediction model should be retrained according to the mobile traffic conditions. Leveraging the diurnal pattern of the mobile traffic, we periodically validate our predication model to ensure the accurate prediction [12].

When we validate the predictor, the ground truth for the cluster numbers K_g obtained from the clustering analysis will be compared with the predicted cluster numbers K. The prediction model will be retrained if the performance of the base station clustering with K clusters is less than 90% of that with K^g. During the retraining process, the latest mobile traffic record data will be integrated to the dataset for re-training the prediction model.

IV. Online Clustering and Network Optimization

The computation complexity of clustering algorithms is vital to achieve the realtime network optimization. As compared with the HCA method, the K-medoids algorithm has lower computation complexity[2]. In addition, since the optimal number of clusters has been predicted by the ANN-based predictor, there is no need to generate the clustering tree. Hence, we adopt the K-medoids algorithm as our online base station clustering algorithm. The pseudo code of the K-medoids algorithm is presented in Algorithm. 1. Once the base stations are divided into clusters, each cluster of base stations are optimized independently with the networking mechanism, e.g, the traffic load balancing.

[2]The complexity of HCA is $o(n^2 * logn)$ [17], where n is the total number of base stations in the network. The complexity of the K-medoids is $o(ikn)$. Here, k is the number of clusters and i is the number of iterations required for the convergence.

Algorithm 1: The K-medoids Clustering Algorithm

Input : A feature set$\{F_g, F_t, F_u\}$, number of clusters k, a cluster set $\mathcal{C}$;
Output: A cluster $\mathcal{C}_T$;
Randomly select k points in the cluster set as k medoids;
while k *medoids changes* **do**
  Associate each point to the closest medoid and formulate k clusters $\mathcal{C}_k$;
  for i=1:k **do**
    Find the point x_i in $\mathcal{C}_i$ that minimizes $\sum_{j\in\mathcal{C}_i}^{|\mathcal{C}_i|} \|x_i - x_j\|_2^2$;
    Set the point as the new medoid;
return $\mathcal{C}_T$;

V. Performance Analysis

In this section, we evaluate the performance of the data-driven network optimization (DINO) framework through data trace-driven network simulations.

A. Mobile Traffic Dataset and Simulation Settings

The simulation uses the real mobile network data which consist of two-week mobile traffic records. These traffic records are collected from an operating mobile network with about 10000 BSs and 500000 users. We define the duration of a time slot to be 10 minutes. Then, we have 144 time slots per day. Γ_i is defined as the traffic load set of the network at ith time slot. In the simulation, 80% of mobile traffic data are used to train the prediction model, and the remaining 20% is used to evaluate the predictor.

In the simulation, we set $\omega = 0.5$ and $\alpha = \{0.65, 0.3, 0.05\}$. Transmit powers of macro and small cell base stations are $P1 = 46$dBm and $P2 = 35$dBm, respectively. On modeling the propagation environment, we adopt a path loss $L(d) = 34 + 40log(d)$ and $L(d) = 38 + 30log(d)$ for macro and small cell base stations, respectively. We adopt the log normal shadowing fading with a standard deviation $\sigma_s = 8$dB. The thermal noise power is assumed to be $\sigma^2 = -104$dBm.

B. DINO Performance

Fig. 3 shows the computation and networking performance of the proposed DINO framework. Fig. 3(a) presents the computation performance of the DINO framework. It shows that the computation time of the DINO framework is less than 10% of that of the optimal networking mechanism in about 95% of the simulated time slots. Here, the optimal networking mechanism is defined as the network optimization mechanism executing on the entire radio access network without the base station clustering. This result proves that the DINO framework can effectively improves computation performance of the networking mechanism. Fig. 3(b) presents the networking performance of the proposed DINO framework. It shows that the DINO framework can achieve at least 85% of the optimal network performance in all simulated time slots. In 80% of the simulated time slots, the DINO framework can achieve 90% of the optimal network performance. Therefore, the DINO framework can significantly reduce the computation time of the networking mechanism at a low cost of the networking performance.

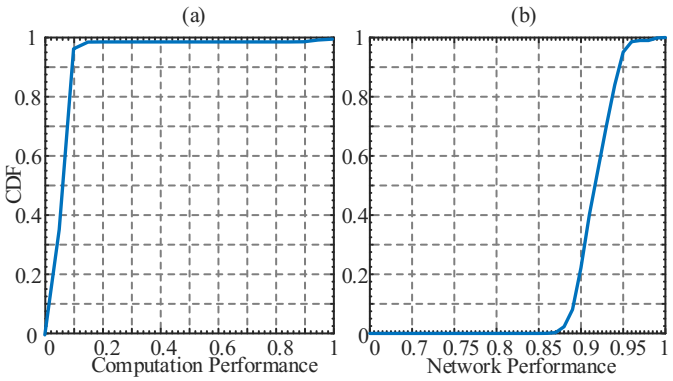

Fig. 3: The CDF of DINO performance.

C. Prediction Model Performance

In order to evaluate the performance of the ANN-based cluster number predictor, we first analyze the impact of the cluster number on the performance of the network optimization. Fig. 4 shows the performance of the DINO framework under three different traffic cogitations. 3:20 AM, 8:40 AM and 11:50 AM represent low, medium and high traffic load level, respectively. We can observe that the optimal cluster number, K^*, with the maximum ψ for different time slots are 8, 15, 25 respectively. Denote $\hat{K}$ as the number of cluster predicted from the ANN-based prediction model, we can see that ψ with the prediction errors $|\hat{K} - K^*| = 1$, $|\hat{K} - K^*| = 2$, and $|\hat{K} - K^*| = 3$ will only averagely reduced by 0.5%, 1.5%, and 2% respectively. Fig. 5 shows the prediction errors of

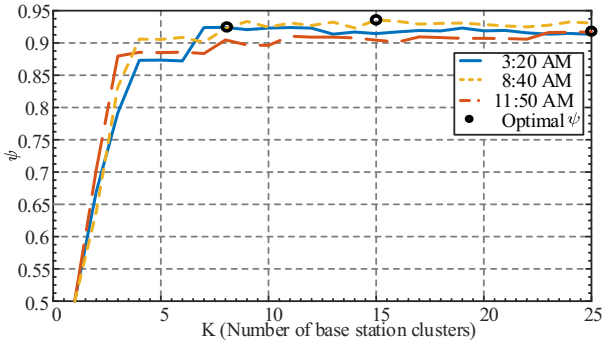

Fig. 4: The DINO performance under different traffic conditions.

the ANN-based cluster number predictor. Although only about 36% of the simulated time slots have no prediction error, there are about 85% of the simulated time slots in which the prediction errors are smaller that 3. This means the network optimization based on the predictions can achieve at least 98% of the optimal performance in 85% of the simulated time slots.

D. Clustering Algorithm Performance

Fig. 6 shows the comparison of the computation time of HCA and K-medoids algorithms. The computation time for clustering base stations with the HCA algorithm is almost stable. This is because the HCA algorithm has

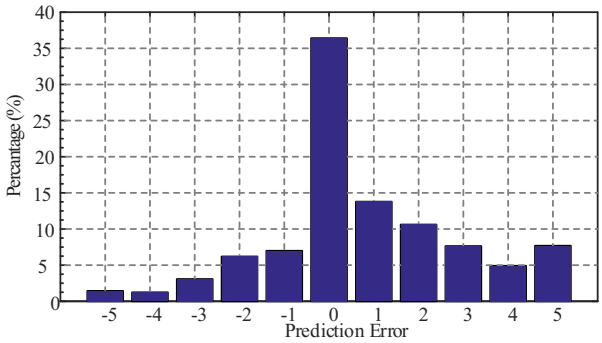

Fig. 5: The prediction error of the ANN predictor.

to generate the entire clustering tree for the base station clustering. The computation time of the K-medoids algorithm is linearly proportional to the number of base station clusters. When the number of base station clusters is small, the computation time of the K-medoids algorithm is much less than the HCA algorithm.

Fig. 7 show the performance comparison of different clustering algorithms. The performance of the HCA algorithm is only slightly better than that of the K-medoids algorithm. The HCA algorithm has a better performance because the HCA algorithm merges the clusters with the minimal inter-clusters distance through building the clustering tree. However, the computation complexity of the K-medoids algorithm is significantly lower than the HCA algorithm. Therefore, the K-medoids algorithm is selected in the DINO framework.

VI. Conclusion

In this paper, we have proposed the data-driven network optimization (DINO) framework which integrates data analysis and machine learning methods with network optimization algorithms to enable realtime and efficient ultra-dense network optimization. The proposed framework consists of an offline machine learning module and an online base station clustering and network optimization module. The offline module predicts the optimal number of base station groups while the online module clusters base stations and optimize the network in realtime. We have validated the performance of the DINO framework through mobile data trace-driven network simulation.

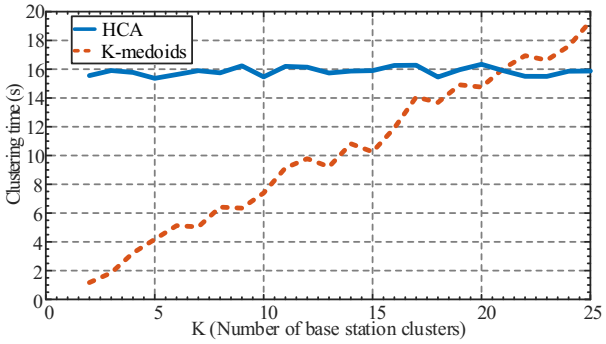

Fig. 6: The computation time of clustering algorithms.

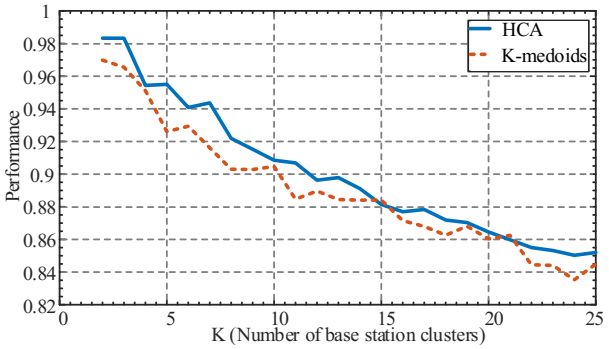

Fig. 7: The performance of clustering algorithms.

References

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2015–2020," Feb. 2016, white Paper.

[2] M. K. Weldon, *The Future X Network: A Bell Labs Perspective*. Crc Press, 2016.

[3] X. Ge, S. Tu, G. Mao, C. X. Wang, and T. Han, "5G ultra-dense cellular networks," *IEEE Wireless Comm.*, vol. 23, no. 1, pp. 72–79, 2016.

[4] Q. Ye, B. Rong, Y. Chen, M. Al-Shalash, C. Caramanis, and J. G. Andrews, "User association for load balancing in heterogeneous cellular networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 6, pp. 2706–2716, 2013.

[5] J. Wu, Y. Zhang, M. Zukerman, and E. K.-N. Yung, "Energy-efficient base-stations sleep-mode techniques in green cellular networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 803–826, 2015.

[6] Z. Mlika, M. Goonewardena, W. Ajib, and H. Elbiaze, "User-base station association in HetSNets: Complexity and efficient algorithms," *IEEE Transactions on Vehicular Technology*, no. 99, 2016.

[7] S. Huang, T. Han, and N. Ansari, "Big-data-driven network partitioning for ultra-dense radio access networks," in *IEEE International Conference on Communications (ICC)*, Paris, France, May 2017.

[8] J. G. Andrews, S. Singh, Q. Ye, X. Lin, and H. S. Dhillon, "An overview of load balancing in HetNets: Old myths and open problems," *IEEE Wireless Communications*, vol. 21, no. 2, pp. 18–25, 2014.

[9] T. Han and N. Ansari, "A traffic load balancing framework for software-defined radio access networks powered by hybrid energy sources," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 1038–1051, April 2016.

[10] J. Dai and S. Wang, "Clustering-based interference management in densely deployed femtocell networks," *Digital Communications and Networks*, vol. 2, no. 4, pp. 175 – 183, 2016.

[11] J. Liu, F. Liu, and N. Ansari, "Monitoring and analyzing big traffic data of a large-scale cellular network with hadoop," *IEEE Network*, vol. 28, no. 4, pp. 32–39, 2014.

[12] K. Zheng, Z. Yang, K. Zhang, P. Chatzimisios, K. Yang, and W. Xiang, "Big data-driven optimization for mobile networks toward 5G," *IEEE Network*, vol. 30, no. 1, pp. 44–51, 2016.

[13] Z. Su, Q. Xu, and Q. Qi, "Big data in mobile social networks: A QoE-oriented framework," *IEEE Network*, vol. 30, no. 1, pp. 52–57, 2016.

[14] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[15] H.-S. Park and C.-H. Jun, "A simple and fast algorithm for k-medoids clustering," *Expert systems with applications*, vol. 36, no. 2, pp. 3336–3341, 2009.

[16] H. Alt and M. Codau, "Computing the Fréchect distance between two polygonal curves," *International Journal of Computational Geometry and Applications*, vol. 05, no. 01n02, pp. 75–91, 1995.

[17] U. Rupapara and G. Mulchandani, "Cancer diagnosis using clustering technique: A literature survey," *Data Mining and Knowledge Engineering*, vol. 8, no. 2, pp. 44–47, 2016.