UE-BASED SMALL CELL CONCEPT IN SUPPORT OF 5G: CLUSTERING ALGORITHM EVALUATION USING SIMULATION TOOLS

Giorgos Argyrides

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

May 2018

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

UE-based Small Cell concept in support of 5G: clustering algorithm evaluation using simulation tools

Giorgos Argyrides

Supervising Professor
Andreas Pitsillides

This thesis is submitted to the Undergraduate Faculty of University of Cyprus in partial fulfillment of the requirements for the Degree of Computer Science at Computer Science Department

May 2018

ACKNOWLEDGEMENT

I would first like to thank my thesis supervisor, Professor Pitsillides for his expert advice and encouragement throughout the duration of this study. I would also like to acknowledge Dr. Christoforou and Dr. Mylonas who assisted me with the network simulations. In addition, I would like to thank my colleague Maria Georgallidou for a wonderful collaboration. You supported me greatly and were always willing to help me. Finally, I must express my very profound gratitude to my family for their continuous support and understanding. This project would have been impossible without of them.

ABSTRACT

The anticipated fifth generation of mobile networks has set ambitious goals for higher capacity, higher data rate, lower latency, lower power consumption and ubiquitous high-speed connectivity for an ever-increasing number of users. The dense deployment of a massive number of small cells will have a pivotal role in 5G as one of the most promising technologies towards realizing its target specifications. This thesis investigates an approach where a subset of User Equipment (UEs) is dynamically selected to serve as the base stations of other users. These UEs are referred as UE-based Virtual Small Cell Base Stations (UE-VBSs) and they can be used in a targeted manner to effectively relieve traffic in hot spot areas, increase coverage, and spectral efficiency. UE-VBSs can remove the constraint of the static deployment of existing Small Cell technologies and bring renaissance to wireless communication networks as a major technological breakthrough.

The first chapter is an introduction in the 5G cellular mobile networks, where the technologies and challenges of 5G are introduced. Moreover, the concept of small cells along with their benefits and challenges is discussed. In the second chapter, the work related to my thesis is exanimated. Then the background of clustering theory is explored, and an overview of some popular clustering algorithms is presented. In the third chapter I evaluate the performance of the UE-VBSs technology. In addition, I compare different clustering techniques, trying to find the most suitable algorithm for clustering an ultra-dense network which utilizes UE-VBSs. The final chapter is a conclusion and comments on the work done.

Table of Contents

Table of figures

List of tables

Chapter 1

Introduction

1.1 Cellular network concept

Wireless mobile networks use radio waves (electromagnetic waves) to transmit data between devices (e.g. mobile devices and base stations). In a cellular network a geographical area is divided into smaller sub-areas called cells, each served by a base station (BS). A mobile device located in the cell's area joins the network through the BS. When combined, these cells provide radio coverage over a broad area. Continuous coverage is achieved by handover (i.e. the transfer of a connection from one BS to another as a mobile device crosses cell borders). The key feature of cellular networks is the capability to reuse frequency bands for increasing coverage and capacity.

1.2 Evolution of cellular networks

Every new generation of wireless networks delivers more functionalities and evolves in terms of data rate, capacity, coverage, quality of service (QoS), mobility, and spectral efficiency. Starting from the 1st generation in early 1980's we are now awaiting the arrival of the 5th generation (5G) of mobile networks with the ambitious timeline of the early 2020s for widespread launch.

The 1st generation (AMPS, TACS, NMT) mobile networks emphasised on speech-related services. They used analog signal, applied circuit switching and were based on FDMA (Frequency-Division Multiple Access) multiplexing scheme. 1G faced major problems like poor voice quality, poor battery life and no security. Moreover, 1G networks were incompatible with each other due to national specifications as they were developed with national scope.

The 2nd generation (GSM) was established in the 1990's and the main difference between its predecessor is that 2G used digital signals. 2G also applied circuit switching was based on TDMA/FDMA multiplexing scheme. 2G, besides the traditional speech service, provided various new features such as text (e.g. SMS), picture and multimedia messages.

Between the 2nd and the 3rd generation 2.5G (GPRS) was introduced. 2.5G applied packet switching along with circuit switching. It provided features such as web browsing and email services. However, it was unable to handle complex data such as videos, it required strong digital signals and it had very limited support for the Internet.

The 3rd generation (UMTS, HSPA) was introduced around 2000 and it was based on WCDMA (Wideband Code Division Multiple Access). 3G featured faster web browsing, video conferencing and 3d gaming. It used both circuit and packet switching until 3.5G. Since 3.5G, the following generations are only using packet switching. 3.5G further improved the data rates but it still had limited support for high-speed Internet.

4th generation (LTE, LTE-Advanced) launched around 2012 and was based on OFDMA (Orthogonal Frequency-Division Multiple Access). 4G provides data speeds of up to 100 Mbps, offers high security and high QoS. 4G features amended mobile web access, high-definition mobile TV and enhanced gaming services. Despite their potential, 4G networks face serious limitations as will be discussed in the next section.

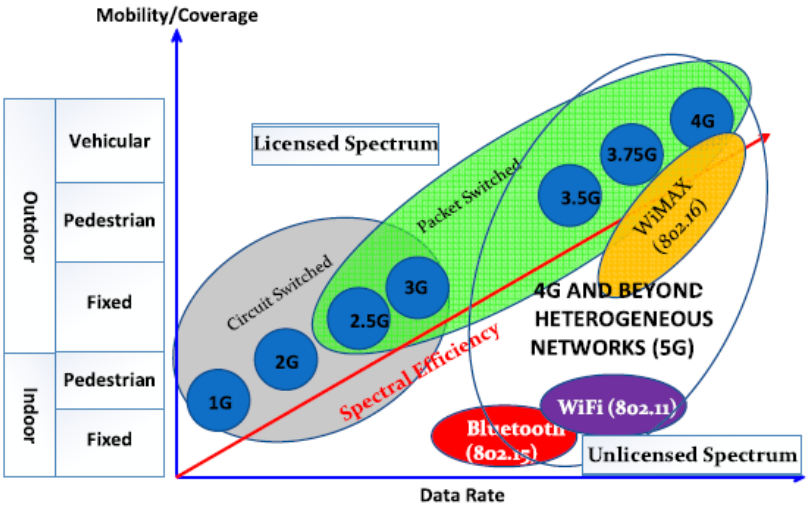


Figure 1. Evolution of cellular networks [1]

1.3 Motivation towards 5G

1.3.1 5G Vision

The vision of 50 billion connected devices by the end of 2020 [2] and the anticipated 1000x data traffic growth [3] necessitate a fifth-generation mobile network. Compelling services such as high-resolution video streaming (In 2022, video will account for around 75% of mobile data traffic [4]) and emerging technologies like autonomous cars demand higher capacity, higher data rate, lower end-to-end (E2E) latency, reliability and robustness on the network [5]. Moreover, consumers desire reduced cost, consistent quality of experience (QoE) and ubiquitous high-speed connectivity.

All these create unprecedented challenges to overcome and along with the unmanageable limitations of the current conventional systems are pushing towards migration to the sate-of-the-art 5G network. 5G will see a radical shift on how cellular networks are designed and used. Subsequently its capabilities will evolve over time and is forecast to cover around a third of the global population by 2025, with adoption reaching 1.1 billion connections [6].

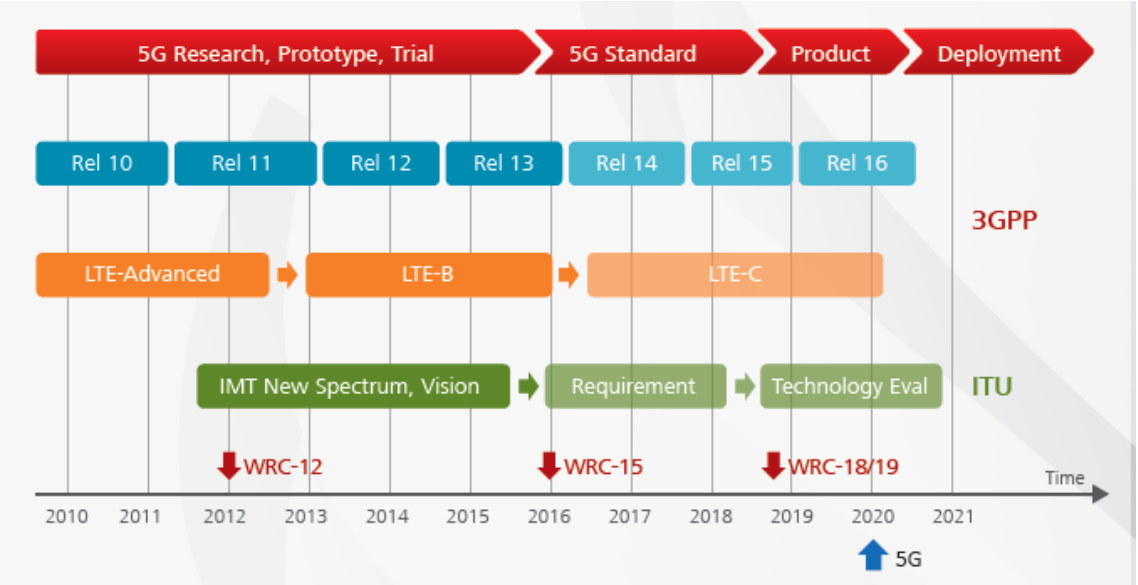


Figure 2. Roadmap and timeline for 5G [7]

1.3.2 Limitations of the conventional cellular systems

In previous generations of cellular networks, mobile phones were practically the only type of device expected to be supported. However, the introduction of a vast variety of devices and applications like Internet of Things (IoT), and tactile Internet [8] will lead us to the aforementioned 50 billion ubiquitous devices. The large amount and diversity of communicating machines creates new requirements and characteristics which the current technologies seem unable to manage efficiently.

For instance, extremely low latency is a primary concern for life-critical systems (e.g. vehicle-to-vehicle communications), real-time applications (e.g. e-commerce transactions, cloud-based gaming) and services with zero delay tolerance (e.g. e-health). However, all the existing technologies of cellular networks are far from achieving the zero latency.

The thousand-fold increase in total mobile traffic requires researchers to seek greater capacity and find new wireless spectrum beyond the 4G standard [9]. Furthermore, the utilization of current radio spectra needs to be improved to solve the spectrum shortage problem.

A BS (Base Station) in conventional cellular networks is designed to support peak time traffic and only its associated UEs (User Equipment) can use its processing power. Additionally, base stations consume a constant power, regardless of the traffic load. However, a heavy loaded BS on a business area during working hours becomes idle during nighttime. On the opposite side an over-subscribed BS in a residential area during weekends or holidays becomes lightly loaded in working days. This effect necessitates better utilization of BS processing power (e.g. when a BS is almost idle, its coverage should expand in a larger geographical area) [10].

Moreover, current cellular networks do not separate indoor and outdoor users although, more than 70 percent of data consumption occurs indoors; in homes, offices, malls and other public places [3]. In addition, most of the wireless users stay inside for approximately 80 percent and outside for approximately 20 percent of the time [11].

1.4 5G Challenges

Ultra-high data rate and network capacity

New technologies, applications and architectures demand matching capacity and data transfer capabilities. Therefore, in order to provide 1000x capacity/km^2 and ultra-fast data transmissions of 10 Gbps peak data rate and 100 Mbps cell edge data rate (100 times faster than 4G LTE) [8], the 5G system will have to combine several innovative solutions.

Reduced Latency

It is mandatory for networks to have almost zero end-to-end latency (1 ms) with high availability and reliability to enable new real-time applications. For instance, remote controlled robots for medical applications, safety-critical applications built around vehicle-to-vehicle (V2V) and vehicle-to infrastructure (V2I) communication, augmented and virtual reality applications. All these applications require rapid feedback control cycles and very quick request-response cycles [5]. The need in latency reduction requires technological innovation in air interface, hardware, waveform design and a flexible architecture in the higher layers of the network [8].

QoS, mobility and coverage.

Users require guaranteed QoS (99.99% availability and 100% coverage [10]). Thus, 5G must provide the same device connectivity and uninterrupted communication services at home, in the office or on the move. In addition robust communication is essential in disaster areas, remote areas, local traffic demands and in places with potential poor coverage (blind spots) [12]. 5G should support high-speed mobility (up to 500 km/h) for railways and V2V communications. Another scenario is extreme user density like shopping malls, open air festivals and stadiums where great user experience must be maintained. A further challenge for 5G is to manage the mobility of the users with low battery consumption for their UE.

Energy efficiency, Cost Reduction

The intelligent use of energy is a major target for 5G (90% less energy). Many times, technologies that improve rate, capacity and coverage lead to energy efficiency drop.

When the energy spent by the infrastructure increases, operation expenditures (OpEx) increase for the network operator which indirectly affect the expenses of the consumers. Also, high computational communication strategies reduce the battery lifetime of UE [13].

Spectral efficiency, new spectrum allocation

Wireless systems are facing a bottleneck in spectrum resources due to the extremely scarce available spectrum. Hence, spectrum needs to be utilized more efficiently in order to provide higher system capacities in the radio interface of the wireless systems.

An additional vital issue of 5G is the allocation of new spectrum as the thousand-fold traffic growth cannot be managed by only enhancing the spectral efficiency. Qualcomm and NSN believe that 10 times more spectrum is needed to meet the demand [14]. Opportunities for more spectra include higher frequency bands (e.g. millimeter-wave, mmW), unlicensed spectrum, and aggregation of fragmented spectrum resources [5].

Improved security and privacy

Services like mobile payment and cloud storage introduce great challenges on data security and user privacy. The huge number of new types of all-time connected devices is threatened by several types of attacks like impersonation, denial-of-service (DoS),

eavesdropping, etc. In addition, the high-speed transfer of a vast volume of data in secure manners is critical while preventing malicious files to penetrate. Moreover, current authentication mechanisms use an authentication server that takes hundreds of milliseconds delay. However, due to the reduced latency requirement of 5G, authentication of network devices becomes extremely challenging [10].

Massive deployment of sensors and actuators

Small sensors and actuators are mounted to stationary or movable objects and enable a wide range of applications by monitoring, alerting or actuating. 5G networks must integrate the communication of ubiquitous things and manage the overhead created by the vast number of devices [12].

1.5 5G Technologies

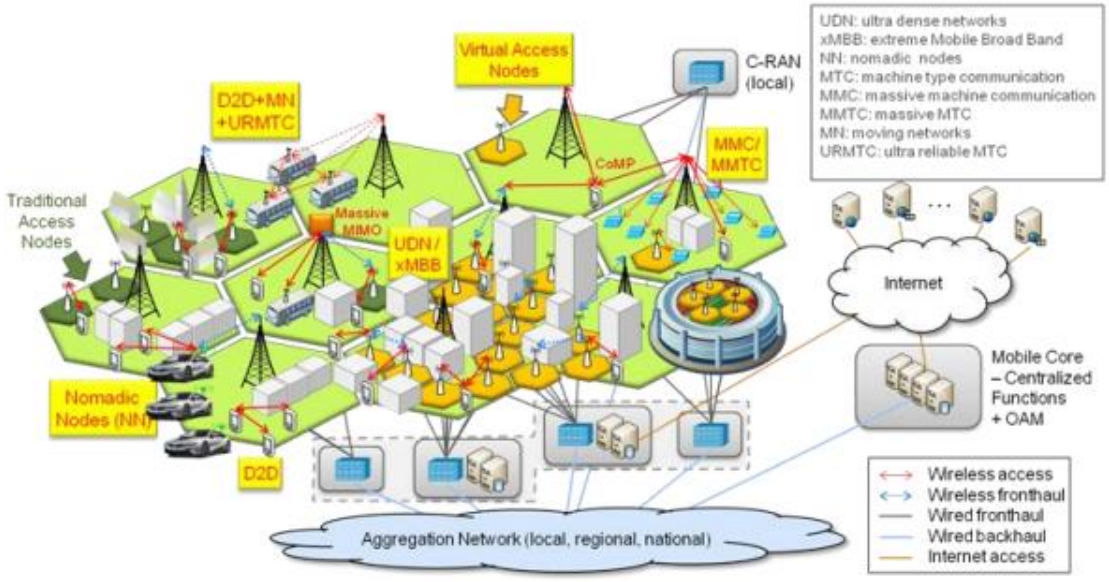


Figure 3. Generic 5G network architecture [15]

Millimeter wave spectrum

The current wireless bandwidth is not able to support a huge number of users in 5G networks. Due to the need for more capacity in mobile broadband, researchers are looking for a new wireless spectrum beyond the 4G standard. According to recent studies millimeter wave frequencies would be able to increase the radio spectrum band from 700 MHz to 2.6 GHz. Using the millimeter wave frequencies, we will achieve a higher

bandwidth, which also means higher data transfer rates. The great capacity that this technology offers would offer high-rate machine-type communications (MTC) applications, and the short range of mm-wave could benefit D2D situations. Moreover mm-wave would possibly extend the IoT (Internet of Things) devices' lifetime by saving a significant amount of energy. This happens by constructing very short over-the-air data packets, so this allows an even more aggressive duty cycling of MTC devices [16].

Wireless software-defined network (SDN)

SDN partitions the data plane functions (traffic forwarding between network devices) from the control plane functions (decisions about the routing of traffic). SDN features a logically centralized entity (i.e., the SDN Controller) which makes the network control functions programmable. This strategy provides network abstraction, therefore allows flexible network reconfiguration and adjustments [16]. Moreover, it reduces the need of complex hardware procedures and it makes the network cheaper to deploy and manage. In addition, SDN makes the infrastructure easier to adapt to the IoT and Cloud Computing [14] and enhances the scalability of the network.

Network function virtualization (NFV)

Network Function Virtualization (NFV) is a complementary technology of SDN, destined to impact future 5G networks. NFV decouples network functions from dedicated hardware and moves them into more general computing and storage platforms like servers [16]. This is achieved by virtualizing a set of network functions such as DNS, traffic load management, network address translation, firewalls etc. Moreover, NFV aims to centralize the base band processing within the RAN (Cloud Radio Access Network - CRAN) [13]. With the NFV, network operators reduce their OpEx and CapEx (Capital Expenditure) while requiring less time and effort to deploy new services.

The combination of NFV and SDN solutions will radically change the way network services are provided. For instance, when a given data center is unable to manage an extreme user density scenario (e.g. an open-air festival), supplementary capacity can be borrowed from other data centers. Also, when an application demands more processing power, it can dynamically allocate more recourses within a data center [17].

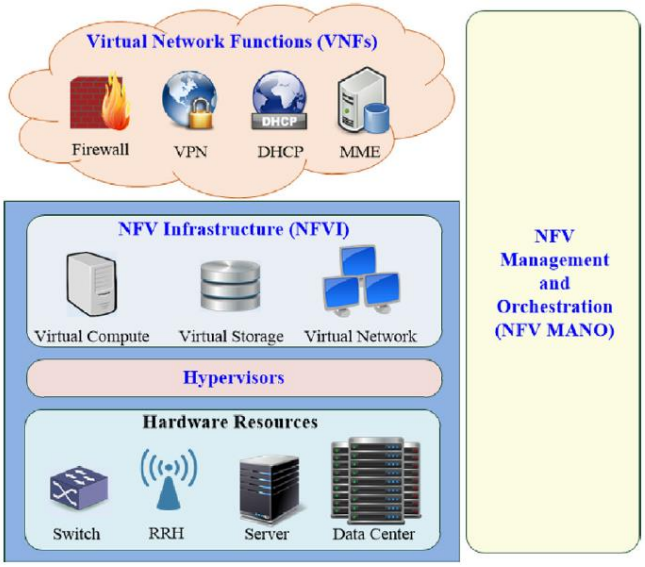


Figure 4. Overall architecture of NFV [8]

Massive MIMO

Massive MIMO (mMIMO) is the evolution of the current MIMO technology and aims to maximize the advantages of MIMO. Arrays (linear, rectangular or cylindrical arrays) with hundreds of antennas are mounted on a BS to simultaneously serve many UEs in the same time-frequency resource [10]. mMIMO improves the spectral efficiency because data transmission can be realized in multiple orthogonal spatial dimensions [17]. It decreases the latency on the air interface, it improves the radiated energy efficiency by 100 times and it increases the capacity by more than 10 times [1].In addition, mMIMO can be utilized to expand the coverage of high frequency bands by relying on beamforming gains [5].

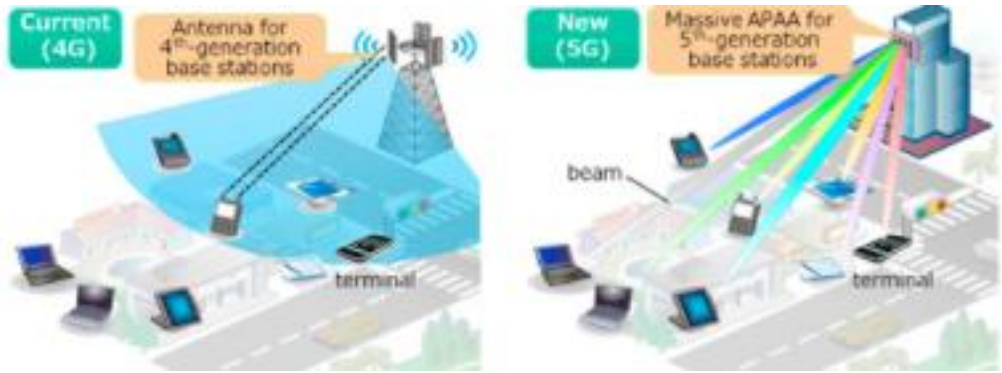


Figure 5. Evolution from 4G MIMO to 5G mMIMO [18]

Device-to-Device (D2D) communications

In conventional cellular networks, all the communications occur through a BS and devices cannot directly communicate in the licensed cellular bandwidth [19]. D2D is a direct communication between devices which allows exchange of user plane traffic without involving or with a controlled involvement of a BS [1]. The proximity of devices can enable reuse of radio resources, low power consumption, low delay and high data rate. Moreover, network-controlled device-to-device technique can achieve local offload and increase the system throughput [20]. Although D2D communications have great potential, they create new challenges such as more interference situations. D2D communications is in an early phase of research and their standards and frameworks are yet to be defined [10].

There are four main types of D2D communication [1]:

1. Device relaying with base station-controlled link formation
2. Direct device to device communication with base station-controlled link formation
3. Device relaying with device-controlled link formation
4. Direct device to device communication with device-controlled link formation

Big data-driven network intelligence

The upcoming 5G cellular infrastructure with its support for Big Data will enable various smart improvements. Excessive amounts of data (user-centric, network-centric and context-centric) will be generated everywhere by both people and machines [14]. Then useful data like people's preferences, traffic conditions, eNB configuration information, interference information, handover reports, fault status, link utilization, call drop ratio etc. will be collected and analyzed in real-time by the network cloud to infer valuable insights [21]. The amassed information can give helpful contribution for network planning, resource management, traffic routing, mobility management and offload decisions [5].

Ultra-densification

Current dense wireless networks are suggested as a supplement for cellular networks and are deployed in hotspot and indoor situations. However, the 5G ultra-dense cellular network, with the help of the innovations of mMIMO and mmWave is proposed to deploy in general cellular situations [22]. The BSs that are used for 4G are unable to hold on with the estimated mobile data traffic that is 15 exabytes by the time of 2018. Hence, to

offload the congested data flow in base stations, spatial densification is proposed. By using this method more layers of cells will be created and with the intelligent handling of shared spectrum resources the capacity will further increase [8]. Nevertheless, researches show that densification is limited by the backhaul network capacity and backhaul energy efficiency restrictions [22]

Cloud Radio Access Network (C-RAN)

The main idea of C-RAN is a network topology where most of the processing of a macro base station (MBS) is executed in the cloud [10].

The three main components of C-RAN as shown in Figure 6 are:

- Distributed network consisting of a Remote Radio Unit (RRU, performs radio functions) and an antenna.
- Optical transmission network which connects the RRU and the Bandwidth-Based Unit (BBU, implements baseband processing using baseband processors).
- Concentrated base band processing pool comprising of high performance general processor and real-time virtual technology.

The dense deployment of RRUs shortens the distance between a UE and an RRU, thus, reduces the transmission power consumption. Consequently, UE battery life will be prolonged, and the power consumption of wireless access networks will be reduced. Conclusively, C-RAN provides a dynamic, scalable architecture, accelerates the speed and reduces the cost of the operational network development [23].

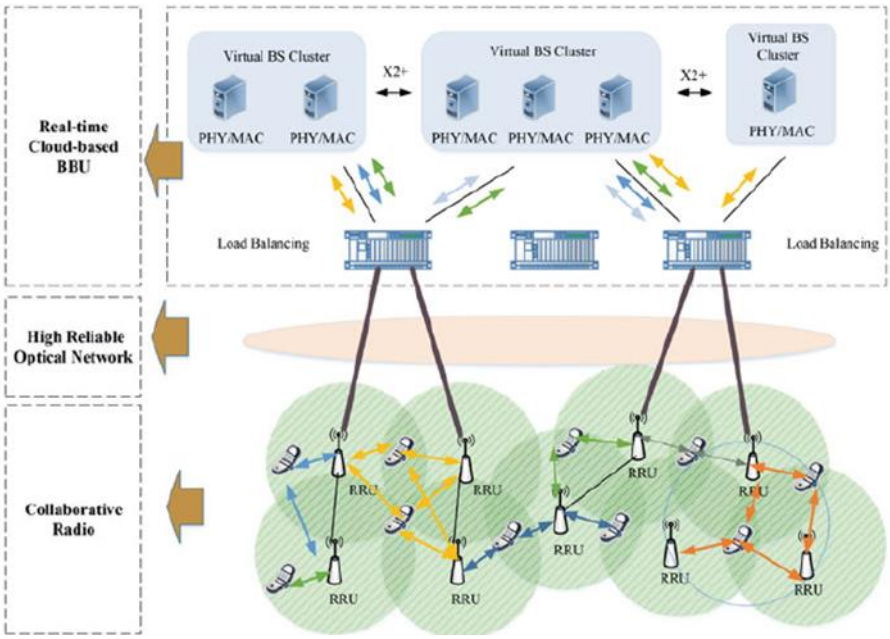


Figure 6. C-RAN architecture [23]

Requirements	Enabling solutions
Ultra-high data rate and network capacity 10 Gbps peak data rate; 100 Mbps cell edge data rate; 1000x capacity/km2	mmWave mMIMO Ultra-densification C-RAN
Reduced latency 1 ms E2E latency	D2D, Big data NFV, mmWave
Spectral efficiency	mmWave, mMIMO Ultra-densification, D2D
Energy efficiency 1000 times decrease in energy consumption per bit;	Ultra-densification D2D
High scalability	mMIMO, SDN, NFV, C-RAN
QoS, mobility and coverage. 99.99% availability 100% coverage high-speed mobility (up to 500 km/h)	Ultra-densification D2D SDN, NFV C-RAN
Improved security and privacy	SDN, NFV, Big data

Table 1. Summarized requirements and corresponding enabling solutions for 5G

1.6 Small cells

A small cell is a low-powered, low-cost, self-organizing base station or access point. The range of a small cell varies between 10 meters and few kilometers. They are placed in opportunistic positions to fill coverage gaps and to provide access to the users in densely-populated regions. The key features of small cells for the consumers are: improved QoS especially for indoor environments, better coverage and capacity, extended battery life. From the operator perspective the key features are reduced CAPEX and OPEX. In legacy network systems, the coverage was provided mostly by macrocells. However, to achieve high throughput to an ever-increasing number of users, the areal reuse must be exploited (i.e. utilize small cells)[17].

Network Densification

Network densification refers to the dense deployment of a massive number small cells.

1.6.1 Types of small cells

Small cells is an umbrella term which includes femtocells, picocells, and microcells.

Femtocells are user-deployed access points with less than 50 meters range. They can operate in closed, open, or hybrid access mode. In closed access mode (usually used in residential scenarios), only a set of enlisted users who belong in the Closed Subscriber Group (CGS) are permitted to connect to the femtocell. In hybrid access mode (usually used in small business scenarios), all users can access the femtocell, but the subscriber group is prioritized. They open access mode is mostly used in public places like airports. [24]

Picocells are deployed by operators for covering small areas, such as street corners and indoor areas (e.g. shopping malls, offices, metro stations) where the macrocell penetration is inadequate. They have less than 500 meters radius, they serve few tens of users and their access is open to all UEs.

Microcells are outdoor base stations with a few hundred meters. They are often installed temporarily during sporting events and other occasions where extra capacity is known to be needed at a specific location in advance. Sometimes they are deployed indoors to provide coverage in areas above the range of a picocell [25].

Relay Nodes (RNs) are low-power base stations, deployed at cell edges for enhanced coverage and capacity. When UEs at cell edge fail to access the macrocell eNodeB, they connect with a RN which in turn is connected with its donor eNodeB (DeNB) via a radio interface [25]. In particular, a RN receives the signal from UEs and retransmits it over the wireless backhaul link between the RN and the DeNB.

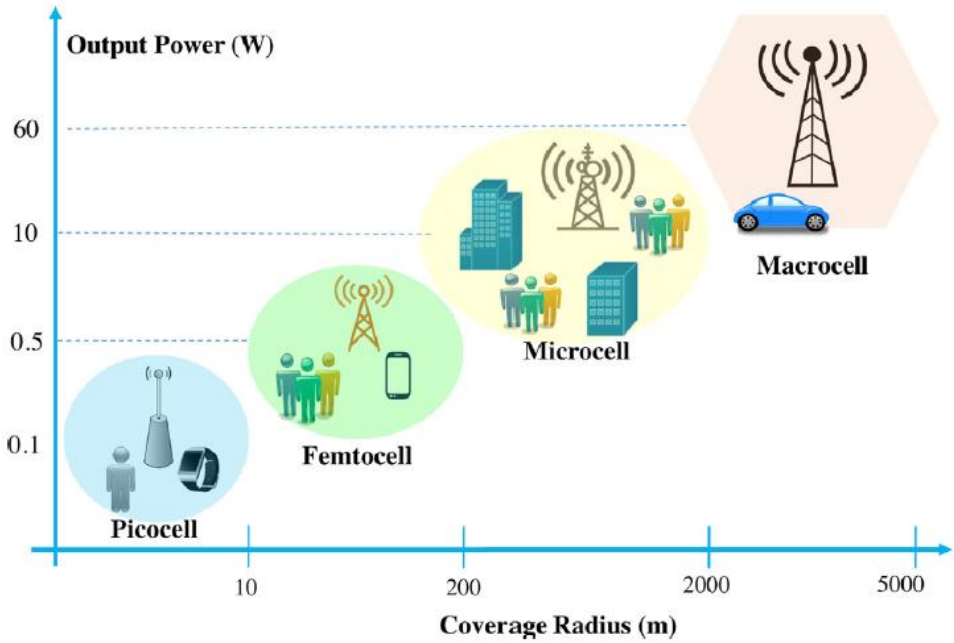


Figure 7. Macrocell and different types of small cells with their corresponding output power and coverage radius [8].

1.6.2 Benefits

As the macrocell network capacity reached very close to their optimal theoretical limits, the next performance leap was achieved by adopting small cells. LTE networks advanced to incorporate small cells such as femtocells, picocells and relay nodes [26].

Small cells brought the mobile network close to the user, thus reduced the distance between the transmitter and the receiver. This results in a higher-quality link and

improved spectral efficiency. Moreover, due to the proximity of UEs and BSs there is reduced power consumption at the BSs and battery savings at the UEs. In addition, lower transmit power results in less adjacent channel and co-channel interferences. Small cells exploit high frequencies, as the high attenuation they suffer is not a disadvantage, but a facilitator to allow effective separation between neighboring cells [5]. Small cells allow operators to follow the traffic requirements by deploying or activating base stations on demand, thus support the users in extremely crowded situations like football stadiums, open air festivals and shopping malls. Also, by dynamically switching small cells on and off network operators can save energy without downgrading network performance [5]. Furthermore, by utilizing small cells, the cellular systems are more robust against the failure of single components. Finally, small cells achieve enhanced in-building and cell-edge coverage.

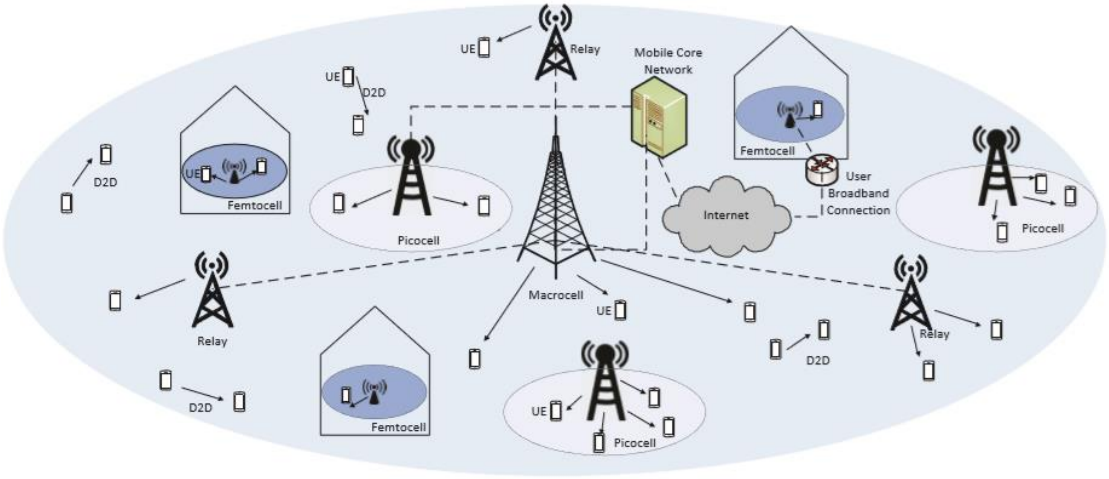


Figure 8. A multi-tier network composed of macrocells, picocells, femtocells and relay nodes

1.6.3 Challenges in ultra-dense small cell deployment

Although small cells are beneficial to cellular networks, their densification and randomness result in a completely different operational environment. In this environment many challenges emerge and need to be resolved in order to exploit the maximum benefits of small cells.

Interference

As the number of cells increases, there is greater potential for interference [25]. In addition, the various transmit powers, the CSG access of femtocells, and the ad hoc deployment of access points further complicate the interference mitigation mechanisms. Particularly, frequencies and power levels need to be carefully adjusted to balance interference and coverage of the different cells in the multi-tier cellular network. Interference can be categorized in co-tier interference and cross-tier interference. Cross-tier interference occurs among network nodes that belong in different tiers, for example picocells and macrocells. On the other hand, co-tier interference appears between nodes that belong in the same tier.

Mobility

A heterogeneous network (HetNet) consists of macrocells and small cells. While small cells improve capacity and coverage, macrocells can support high mobility. A mobility management mechanism is required, where UEs connect to the proper cells depending on their speed. Static UEs should be serviced by the ultra-dense small cells and mobile UEs should be handed over to the macrocells [27].

Backhaul

5G ultra-dense networks are anticipated to achieve throughput of gigabit levels [28]. This traffic must be effectively forwarded by reliable and low latency backhaul connections to the core network. Mobile operators consider small cell backhaul to be more challenging than macrocell backhaul because since small cells are frequently placed in hard-to-reach spaces near streets instead of clear area [14]. The backhaul requirements vary according to the position of small cells, the traffic supported by the cell, the target QoS, and the cost of installing a backhaul connection. Hence there is no best solution for the backhaul of small cells [29]. Most operators must use a mix of wired and wired solutions with cautious planning and administration to avoid restricting capacity by inadequate backhaul connections.

Other important challenges of the ultra-dense small cell depoloyment are Complexity (computational and implementation complexity), network topology control, congestion

control and radio resource management (RRM). 5G needs to address all the aforementioned challenges in order to realize the potential of the small cells technology.

Chapter 2

Background Knowledge

2.1 Related work

2.1.1 CelEc framework for reconfigurable small cells

The Cella Ecosystem (CelEc) [30] is a proposal to support the upcoming ultra-dense 5G networks. Conventional small cells follow a static deployment, which cannot deal effectively with the short term changing dynamics of mobile traffic. However, the CelEc framework establishes a mobile small cell, the Cella cell, to cope with the mobile traffic patterns. The Cella cell is the smallest, portable and reconfigurable cell, which can be deployed in real time, in an effortless and adjustable way. The Cella cell utilizes UEs which are already densely distributed as an integrated part of the mobile network infrastructure. The UEs used in a Cella cell are recruited by network operators under a "leasing service contract" and are called CelEc Devices (CelDes). A CelDe has macro base station's functionalities and can provide services to other UEs on demand, according to the network requirements. The massive deployment of Cella cells, has the ability to fill coverage holes, improve network capacity, increase data rates, spectral and energy efficiency.

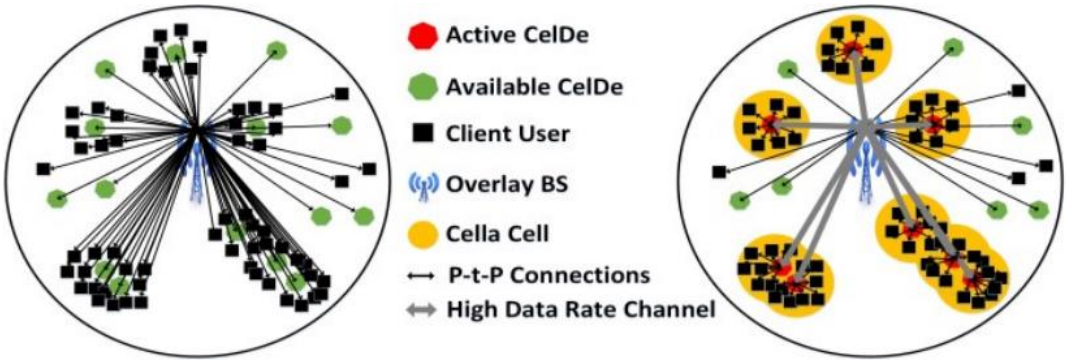


Figure 9. Left: All connections are handled by the BS. Right: Most of direct connections are handled by CelDes [30].

2.1.2 Virtual Small Cells Formation in 5G Networks

The virtual small cell formation approach for 5G networks is proposed in [31]. In this approach, a subset of qualified UEs is selected to serve as the base stations of virtual small cells in connecting other UEs to the macrocell base stations. With this technique, the number of required communication links to the macrocell base stations for connecting all users directly or through small cells is reduced. More specifically, the number of connections to the macrocell base stations becomes proportional to the logarithm of the user density. Thus, utilization of virtual small cells can significantly increase the overall network capacity. The realization of conventional small cells is mainly achieved in the form of increasing the cellular infrastructure. This approach implies significant costs and complexity. Moreover, such static networks would not be able to follow the dynamic traffic flow of the small cells cost-effectively. On the other hand, a UE-based small cell formation is an almost costless, dynamic, and ready-to-use solution.

2.1.3 Selection of UE-VBS using Affinity Propagation Clustering

The paper *"Selection of UE-based Virtual Small Cell Base Stations using Affinity Propagation Clustering"* proposes the dynamic clustering of virtual small cells and the activation of UE-VBSs (UE Virtual small cell Base Stations) using the affinity propagation clustering technique. The objective is to choose the ideal number of UE-VBSs for activation and then cluster the UEs. Every cluster represents a Virtual Small

Cell and every cluster center represents the UE-VBS which will server all the UEs in that cluster. The original parameters and procedures of affinity propagation are modified to meet the specific requirements of forming the virtual small cells and choosing the active UE-VBSs. The Received Signal Strength (RSS) is the similarity parameter between the eligible UE-VBSs and UEs. The UEs measure the RSS from every eligible UE-VBS in their proximity. Each UE will connect with the UE-VBS from which it receives the maximum RSS. In contrast with the original Affinity Propagation algorithm which passes messages between all points, the modified algorithm passes messages only between the UEs and the eligible UE-VBSs.

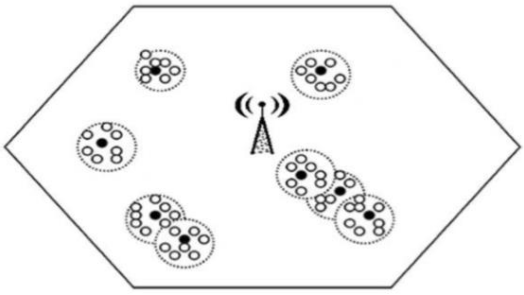


Figure 10. Network that uses UE-VBSs (Black Nodes)

2.1.4 Network-assisted Outband D2D-clustering in 5G Cellular Networks

In this approach [32] UEs form clusters where only the UE with the best channel condition (i.e. the cluster head) is connected directly with the eNodeB. This way, a cluster can be seen as a UE whose SNR is the highest of the SNR of cluster members. The rest of the UEs in the cluster, utilize Outband D2D connections to relay their traffic. Specifically, in Outband D2D, UEs can use both Cellular and WLAN technologies. The eNodeB communicates with the cluster head via cellular connection (e.g. LTE) while the rest of the UEs use unlicensed spectrum such us mm-Wave and WiFi Direct for intra-cluster communications. Simulations prove that by clustering UEs this way, cellular networks can achieve higher spectral and energy efficiency as well as lower signaling overhead. Consequently, clustering is suitable for the 5G ultra dense networks where energy and spectral efficiency is essential. Moreover, in this paper, the analysis outcomes indicate that joining a cluster not only is advantageous for the general system performance but also for all individual members. Precisely, their proposal boosts the network's capacity

by up to 76% with clusters of just five UEs. Additionally, network operators can …. users to support the cellular network. Users will be rewarded according to their contribution. In addition, network operators can motivate their users to relay each other's traffic by rewarding them according to their contributions.

Figure 11. Example of D2D clustering [32]

2.2 Cluster analysis

Cluster analysis or clustering is the process of organizing objects into groups (i.e. clusters) whose members are similar in some way. Given a set of objects, we can use a clustering algorithm to classify each object into a specific group. Theoretically, objects that are in the same group should have similar properties, while objects in different groups should have highly dissimilar properties.

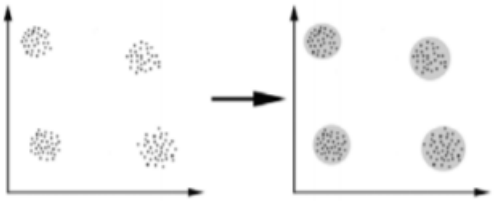


Figure 12. An example of clustering [33]

A good quality clustering will produce clusters where intra-cluster similarity is high wile inter-cluster similarity is low [34]. In a classic clustering problem, we have a set of n input objects from an arbitrary metric space and want to divide them into k clusters [35]. Typical applications of clustering concern low-dimensional Euclidean spaces (i.e. *"a space in any finite number of dimensions, in which points are designated by coordinates (one for each dimension) and the distance between two points is given by a distance formula"*). Nevertheless, modern applications involve high-dimensional Euclidean

spaces and spaces that are not Euclidean. Clustering problems arise in many different scientific areas such as data mining, machine learning, image processing, pattern recognition etc.

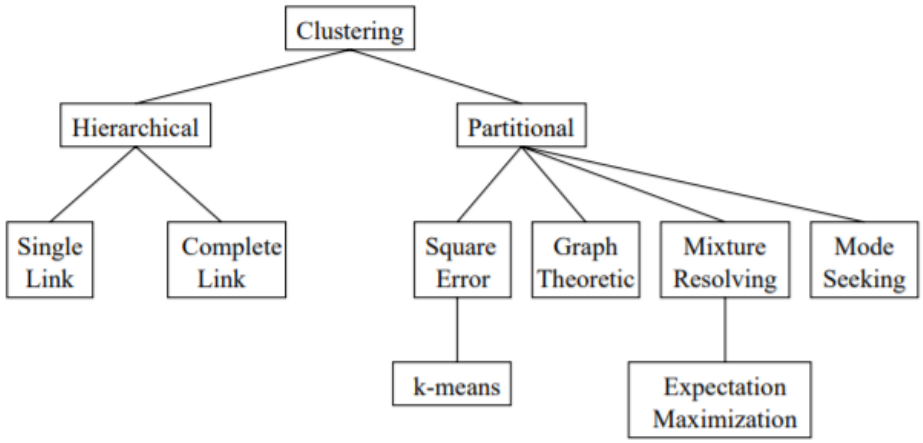


Figure 13. Taxonomy of clustering approaches [36].

As shown in the figure above, clustering can be divided in hierarchical and partitional. A partitional or flat clustering algorithm (e.g. k-means) is essentially a division of the data set into clusters (subsets), so that each data point is in precisely one subset. Given a set of data objects and the number k, it finds a partition into k clusters that optimizes the given partitioning principle. On the other hand, hierarchical clustering does not assume a particular value k. It is a set of nested clusters that can be visualized using a tree structure (dendrogram) as seen in the figure below.

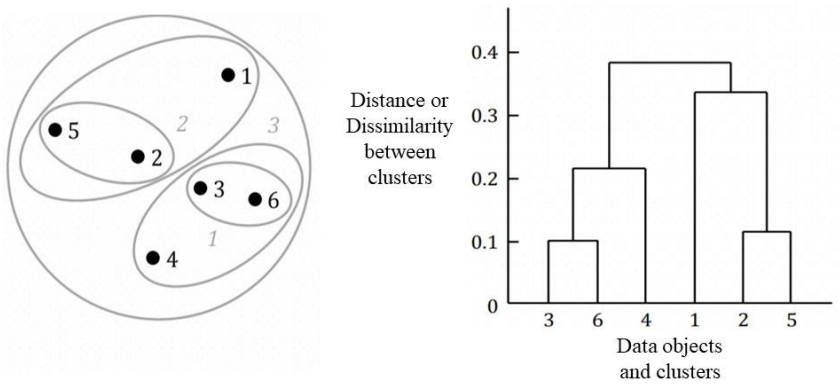


Figure 14. Nested cluster diagram and hierarchical tree diagram.

From this dendrogram it is possible to view partitions at different levels of granularities. The key operation of hierarchical clustering is repetitive combination (each combination is represented by a horizontal line in the dendrogram) of two nearest clusters. There are two types of hierarchical clustering, Agglomerative ("Bottom-up") and Divisive ("Top-down"). Agglomerative algorithms start by treating each data point as a single cluster and then continuously merge (or agglomerate) pairs of clusters until all clusters have been combined into a single cluster that contains all data points. Divisive algorithms start with one, all-inclusive cluster and then recursively removes the "outsider" points from the least cohesive cluster until all points belong in their own singleton cluster. Divisive algorithms are more computationally intensive, so they are less widely used than agglomerative methods.

Partitional clustering results in a single partitioning whereas hierarchical clustering can give different partitioning according on the level-of-resolution that the underlying application requires. Partitional clustering necessitates the number of clusters to be given in advance, but hierarchical clustering doesn't. Lastly, partitional clustering is often more efficient in terms of computational and storage while hierarchical clustering can get slow and consume a lot of storage as it must make a lot of merge and split decisions. Nonetheless there is no clear agreement on which of the two approaches achieves the better clustering.

2.2.1 K-Means clustering

K-means clustering algorithm was first proposed by Macqueen in 1967 [37]. It is an uncomplicated partitioning algorithm. Which means, it partitions the given dataset into a predefined number (k) of groups. K-Means determines a set of k points, called centres, which may not be necessarily a point in the dataset. The centres are chosen, such that the mean squared distance from each data point to its nearest centre is minimized. In multiple real applications the clustering technique that is used is the k-means algorithm with multiple restarts [36]. With this method the algorithm can be run and measured with many different k values and initializations. This performs well when the number of clusters is small and chances are great that at least one initialization is near to a good solution. However, the fact that k-means algorithm's performance heavily depends on the initial

starting conditions is a significant disadvantage for large datasets. Also, K-means is unable to represent density-based clusters.

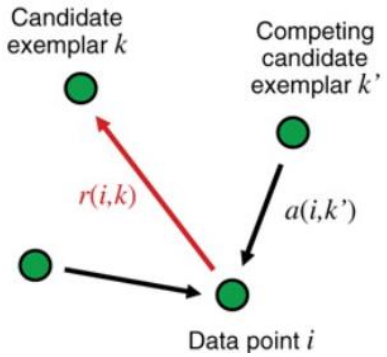
```
Initially choose k points that are likely to be in
    different clusters;
Make these points the centroids of their clusters;
FOR each remaining point p DO
    find the centroid to which p is closest;
    Add p to the cluster of that centroid;
    Adjust the centroid of that cluster to account for p;
END;
```

Figure 15. Outline of K-Means algorithms [34]

2.2.2 Affinity Propagation

The Affinity Propagation algorithm was published by Brendan Frey and Delbert Dueck in 2007. In Affinity propagation (AP) all the datapoints are potential cluster centres (exemplars). The result of AP is a set of exemplars, from which clusters can be derived by assigning each point to the cluster of its nearest exemplar. In contrast with K-means, in AP the number of clusters is not required as input. AP takes as input a set (n x n) of similarity values between all pairs of data points. The similarity value *S(i,k)* indicates how well the data point k is suited to be the exemplar for data point *i*. The AP algorithm works by exchanging the following messages between data points [38]:

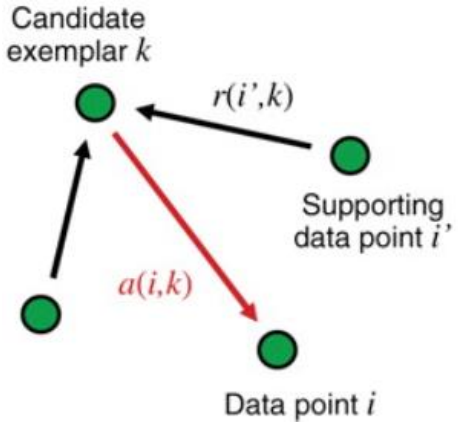
1) "Responsibilities" *r*(*i, k*) are sent from data point *i* for how well-suited point *k* is to serve as the exemplar for point *i*. Responsibility values are computed as shown in figure below. The availability values are initially zero.



$$r(i,k) \leftarrow s(i,k) - \max_{k' \text{ s.t. } k' \neq k} \{a(i,k') + s(i,k')\}$$

Figure 16. (left) visual representation of sending responsibility, (right) the rule for computing the responsibility values [39].

2) "Availabilities" *a(i, k)* are sent from candidate exemplar point k for how appropriate it would be for point *i* to choose point *k* as its exemplar. Availability values are computed as shown in figure below. Self-availabilities *a(k,k)* are computed differently.



$$a(i,k) \leftarrow \min\left\{0, r(k,k) + \sum_{i' \text{ s.t. } i' \notin \{i,k\}} \max\{0, r(i',k)\}\right\}$$

$$a(k,k) \leftarrow \sum_{i' \text{ s.t. } i' \neq k} \max\{0, r(i',k)\}$$

Figure 17. (left) visual representation of sending availability, (right) the rules for computing the availability values [39].

These messages are iteratively transmitted until an unchanging set of m exemplars emerge (convergence) or until a predefined maximum number of iterations. The corresponding clusters of the m exemplars are the solution of AP clustering. The major disadvantage of AP is its complexity which makes it suitable for small to medium sized datasets. It has time complexity of $O(k \times N^2)$, where N is the number of data points and k is the number of iterations. Furthermore, the memory complexity is $O(N^2)$ if a dense similarity matrix is used.

2.2.3 Mean-Shift

Mean shift clustering attempts to find dense areas of data points. Like K-Means and affinity propagation, it is a centroid based algorithm (its goal is to locate the centre point of each cluster). However, it can return clusters instead of partitions. Mean shift detects the centre points of each cluster by updating candidates to be the mean of the points within the sliding-window. Again, unlike K-means it's not needed to specify the number of

clusters. Another advantage of Mean-shift is that the cluster heads converge near the points with the maximum density. The density is specified by a probability density function and mean-shift tries to place the centroids of the clusters at the maxima of this function. Moreover, the most significant parameter of this algorithm is the bandwidth of the kernel used in the kernel density estimation method. The kernel is being shifted repetitively to a higher density area on every iteration until convergence. Nevertheless, mean shift outcomes can differ when the bandwidth parameter is changed, which makes its selection difficult but not as difficult as guessing the number of clusters.

2.2.4 Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

Like Mean-Shift, DBSCAN [40] is a density-based clustering algorithm, but with some noteworthy privileges. It extracts the dense clusters while the sparse (with density below a given threshold) data points are left as noise. The density threshold is the minimum number of data points (MinPts) in a sphere of radius epsilon (ε). All points within distance ε are neighbourhood points. The algorithm starts with an arbitrary point and visits all points iteratively. If there are enough (>= MinPts) points within the neighbourhood of the current point, then the clustering process begins. All the points within distance ε are added in the cluster and then this is repeated for all the new points in the cluster. Afterwards, a new unvisited point is selected, and a new cluster or noise points are detected. In the end of this procedure all points will either belong in a cluster or will be marked as noise. DBSCAN is a deterministic algorithm if the same dataset is given in the same order, but the results may vary when the same dataset is given in a different order [41]. Moreover, it can give arbitrarily shaped and sized clusters, as well as identify points as outliers instead of just throwing them into a cluster. However, the algorithm becomes unstable when detecting border objects of adjacent clusters [40] and there are algorithms that perform better when clusters are of varying density.

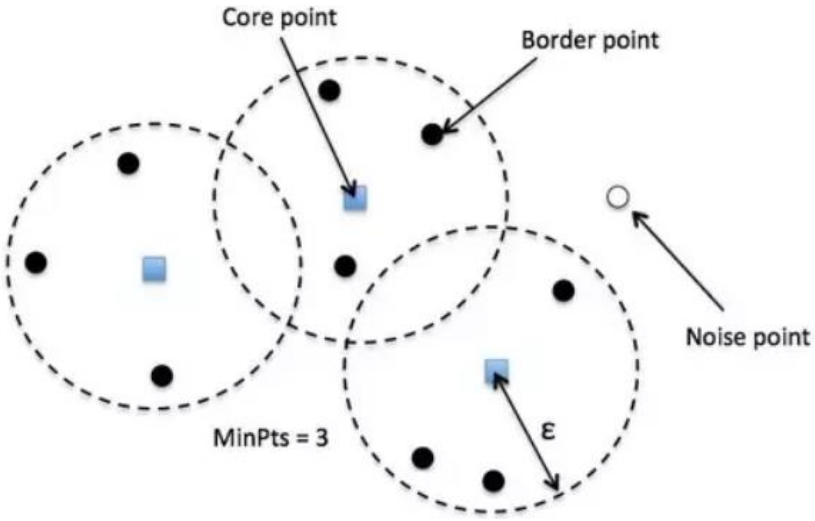


Figure 18. DBSCAN example

2.2.5 Hierarchical DBSCAN (HDBSCAN)

HDBSCAN [42] was published in 2015. A part of the people who published the aforementioned DBSCAN algorithm, also worked on HDBSCAN. It extends DBSCAN by inheriting its advantages and overcoming its limitations with varying density clusters. HDBSCAN takes as an input the parameter M_{pts} which is also inherited from DBSCAN (MinPts). The algorithm works by firstly transforming the space according to the density (with respect to the parameter M_{pts}). Then it constructs the Minimum Spanning Tree (MST) of the distance weighted graph. Afterwards it builds a dendrogram with the cluster hierarchy. The next step is where DBSCAN uses the epsilon (ε) value as a cut level of the dendrogram, which leads to a single fixed density level. However, HDBSCAN does not require the epsilon value as input. Instead it uses a procedure that allows the dendrogram to be cut at different levels, thus achieve varying density clusters. This procedure takes a parameter that defines the minimum cluster size ($M_{ClusterSize}$). This procedure condenses the dendrogram by using the $M_{ClusterSize}$ to find points that will be marked as noise or will split to form a new cluster. In the final step, the algorithm extracts the clusters from the condensed tree based on cluster stability. The $M_{ClusterSize}$ is not a hard to choose parameter, as giving the minimum cluster size is practically rational. On the other hand, the epsilon value of DBSCAN is an unintuitive parameter. In conclusion, HDBSCAN overcomes its predecessor by achieving varying density clusters and by trading an unintuitive input

parameter with an easier to choose parameter. Nevertheless, it still requires the M_{pts} parameter that again is not instinctive.

ALGORITHM 1: HDBSCAN* main steps

1. Compute the core distance w.r.t. m_{pts} for all data objects in **X**.
2. Compute an MST of $G_{m_{pts}}$, the Mutual Reachability Graph.
3. Extend the MST to obtain MST_{ext}, by adding for each vertex a "self edge" with the core distance of the corresponding object as weight.
4. Extract the HDBSCAN* hierarchy as a dendrogram from MST_{ext}:
   - 4.1 For the root of the tree assign all objects the same label (single "cluster").
   - 4.2 Iteratively remove all edges from MST_{ext} in decreasing order of weights (in case of ties, edges must be removed simultaneously):
     - 4.2.1 Before each removal, set the dendrogram scale value of the current hierarchical level as the weight of the edge(s) to be removed.
     - 4.2.2 After each removal, assign labels to the connected component(s) that contain(s) the end vertex(-ices) of the removed edge(s), to obtain the next hierarchical level: assign a new cluster label to a component if it still has at least one edge, else assign it a null label ("noise").

Figure 19. Main steps of the HDBSCAN algorithm [42]

2.2.6 Agglomerative Clustering

Agglomerative Clustering is a set of closely associated clustering techniques [43]. It is a "Bottom-up" hierarchical clustering algorithm. It begins with each point being a cluster on its own. Then clusters are recursively merged (agglomerated) according to how "close" they are, until all clusters have been combined into a single cluster. In the dendrogram visualization, the root is the cluster with all the data points and the leaves are the clusters with only one data point. The merging of clusters stops when further merging results to undesirable clusters. For instance, there may be a measure of how condense the clusters are and merging two clusters surpasses the measure. Alternatively, there may be a predefined number of clusters. There are many possible definitions of the cluster proximity (i.e. how "close" two clusters are) and each definition discriminates the various agglomerative algorithms. Notably, the computation of the cluster proximity (or linkage distance) is the key operation of agglomerative clustering and it is not an easy task.

Proximity methods:

Single-link (or single-linkage or MIN) distance of clusters C_1 and C_2 is the minimum distance between any data point in C_1 and any data point in C_2. Single-linkage can handle

non-elliptical shapes. However, it is sensitive to noise and outliers and results in chaining, which means clusters can get very large.

Complete-link (or complete-linkage or MAX) distance of clusters C_1 and C_2 is the maximum distance between any data point in C_1 and any data point in C_2. Complete-linkage produces more balanced clusters and is less susceptible to noise. Nonetheless, it can break big clusters and merge small clusters with larger ones, thus favors globular shapes (all clusters tend to have the same diameter).

Group average distance between clusters C_1 and C_2 is the average distance between any data point in C_1 and any data point in C_2. It is an intermediate approach between complete and single linkage. Group average is less susceptible to noise and outliers, but it is biased towards globular cluster.

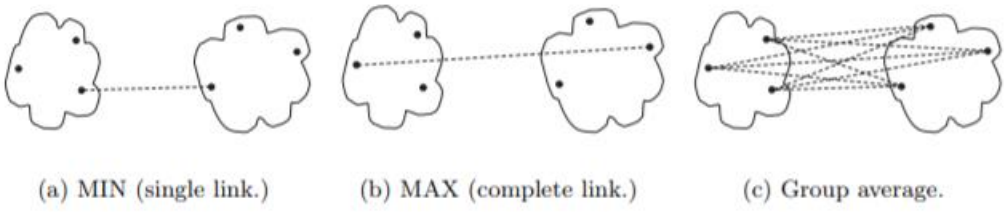


Figure 20. Graph-based definitions of cluster proximity [43]

Single-link, Complete-link and Group average are graph-based proximities. There is also Ward's method which is a prototype-based proximity. The prototype of a cluster is often the centroid (i.e. the average of all data points in the cluster). If a centroid has no meaning in the dataset then the prototype can be a medoid (i.e. the most representative data point of the cluster) [43].

Ward's distance between clusters C_1 and C_2 is the difference between the Sum of Squared Error (SSE) of the result from merging the two clusters and the total SSE of the two clusters separated ($d_{C1,\,C2} = SSE_{C1 \cup C2} - SSE_{C1} - SSE_{C2}$). Similar to group average, it is less susceptible to noise and outliers, but it minimizes the variance of the clusters being merged.

Agglomerative methods are typically used when the underlying application requires a hierarchy. Moreover, some studies suggest that they can produce higher-quality clusters. On the other hand, agglomerative clustering algorithms are computational and in terms of storage expensive. Also, the fact that all merges are final and cannot be undone can generate problems for high-dimensional and noisy data.

Agglomerative$(S = \{x_1, \ldots, x_n\})$ **returns** $Dendrogram_k$ for $k = 1$ to $|S|$.

1. $C_i = \{x_i\}, \forall i.$
2. **for** $k = |S|$ **down to** 1
   $Dendrogram_k = \{C_1, \ldots, C_k\}$
   $d(i,j) = D(C_i, C_j), \forall i, j; \quad l, m = argmin_{a,b}\, d(a,b).$
   $C_l = Join(C_l, C_m); \quad Remove(C_m).$
   endloop

Figure 21. Standard Agglomerative Clustering [44]

2.2.7 Spectral Clustering

The central idea of Spectral Clustering is to utilize the spectrum of the similarity matrix of the data and perform dimensionality reduction [45]. Then standard clustering is used in fewer dimensions. First, Spectral Clustering forms an $n \times n$ affinity matrix A, where n is the total number of data points. Then it builds the Laplacian matrix L from the normalized matrix of A [45]. Afterwards, Spectral Clustering looks at the eigenvectors of the Laplacian matrix L and tries to find a good (low dimensional) embedding of the graph into Euclidean space [46]. Finally, a standard clustering algorithm like k-means is run. Spectral Clustering is efficient if the affinity matrix A is sparse and when the number of clusters is relatively small. It is useful in hard non-convex clustering problems and it is extensively used in image segmentation.

2.2.8 Ordering Points To Identify the Clustering Structure (OPTICS)

OPTICS [47] is another density-based clustering algorithm. Just like HDBSCAN it addresses the DBSCAN's problem with varying density clusters. However, like DBSCAN, OPTICS requires the two parameters, ε (in OPTICS ε is only an upper limit for the neighborhood size used to reduce computational complexity) and MinPts. OPTICS

does not create one explicit clustering. Instead, it produces a cluster-ordering of the data points with respect to its density-based clustering structure containing the information about every clustering level of the data set up to a generating distance ε [48].

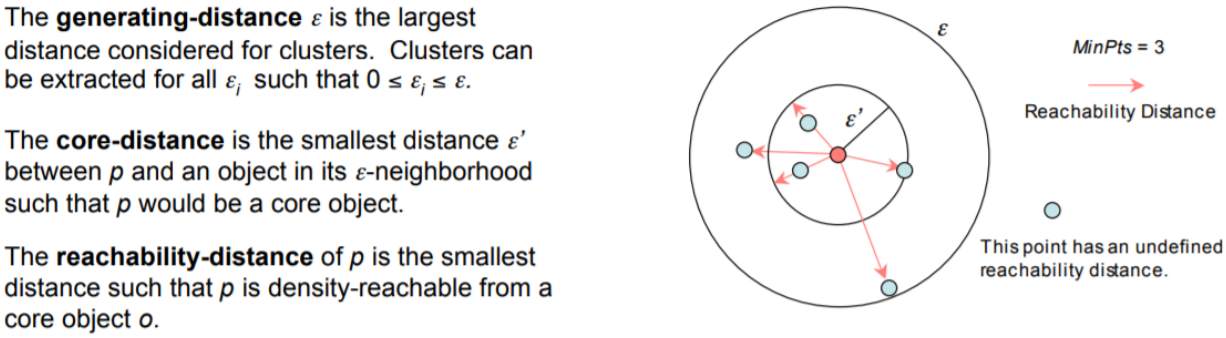


Figure 22. OPTICS distances

OPTICS defines a reachability-distance value for each data point. This value specifies the distance to the next data point in the ordering. Depending on what reachability threshold is used, a larger or smaller number of clusters is generated. Moreover, large reachability values represent the boundaries between clusters. The result of OPTICS (i.e. the cluster ordering) is visualized graphically by a reachability plot to support the analysis of the cluster structure [49]. In this plot the clustered points are ordered along the x-axis according to the cluster ordering computed by OPTICS and the reachability values assigned to each point are plotted along the abscissa. Data points having a small reachability value are closer, thus more similar to their predecessor points [50]. Valleys in the reachability plot correspond to clusters, which can be hierarchically nested [48].

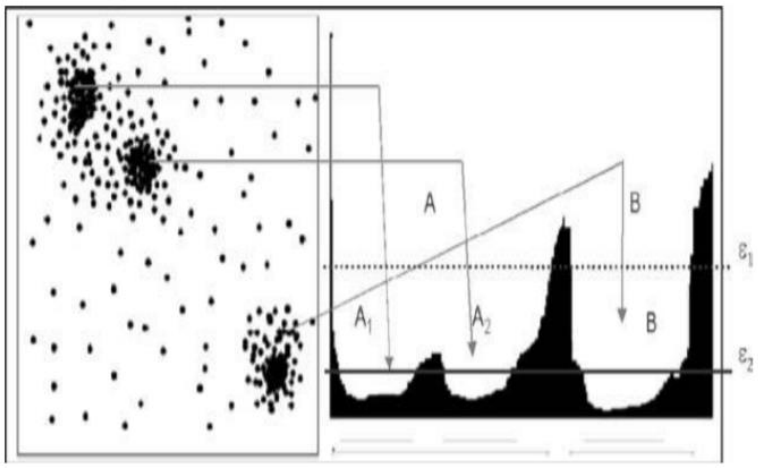


Figure 23. Reachability plot (right) computed by OPTICS for a sample data set (left) [50].

The time complexity of OPTICS algorithm is O (n log n).

OPTICSxi

The OPTICSxi algorithm has an extra parameter, xi, which is the contrast parameter that established the relative decrease in density. This parameter directly controls the number of clusters we will obtain. In this algorithm the ε parameter becomes less important but still needs to be given a value. The idea of varying densities is implemented by having a range of ε parameters. By setting the xi parameter we set the density variation we accept to consider that group a cluster.

2.2.9 Locally Scaled Density Based Clustering (LSDBC)

The LSDBC paper [51] introduces the notion of local scaling in density based clustering, that decides the density threshold according to the local statistics of the data. A k-nearest-neighbor density estimation is used to discover the local maxima of density which are used as cluster centers. Each cluster is expanded until the density falls below a predefined ratio of the center point's density. This clustering technique can produce clusters of arbitrary shape with noisy backgrounds that include density gradients. LSDBC has two input parameters, k, which is the order of nearest neighbor to consider for each data point for density calculation and α, that controls the boundary of the current cluster expansion based on its density. LSDBC uses the idea of local scaling. The local scaling technique is also successfully used by spectral clustering which is discussed above. The idea is to scale every data point with a factor proportional to its distance to its k_{th} neighbor. By ordering points according to their distance to their k_{th} neighbor, a measure of how dense the area around each data point is given. Afterwards, starting from higher density points, densely populated areas are cluster together. This clustering technique does not require fine tuning of its parameters and is more robust.

Chapter 3

Implementation and Evaluation

3.1 Problem statement

I consider an ultra-dense 5G network where a subset of UEs will be selected to act as virtual small cell base stations (UE-VBSs) on a dynamic schedule. The purpose of the UE-VBSs is to aid the BS when it is congested, by forming clusters of UEs served by a UE-VBS. As mentioned in the topics 2.1.1 and 2.1.2, a UE-based small cell formation is an almost costless, dynamic, and ready-to-use solution. Furthermore, it can improve network capacity, increase data rates, and enhance spectral and energy efficiency.

Firstly, I evaluate the performance of UE-VBSs using a network simulation tool. Then I compare different clustering methods which will be used to divide the UEs into groups where only one UE (i.e. the UE-VBS) will be directly connected with the main BS while the rest UEs in the cluster will communicate through their UE-VBS. The goal is to find the optimal way to cluster the UEs in order to maximize the benefits of virtual small cells.

3.2 Virtual small cell's performance evaluation using OPNET modeler

3.2.1 OPNET Modeler

OPNET (Optimized Network Engineering Tools) Modeler is a powerful tool for network modeling and simulation, which is used by universities and researchers worldwide. I worked with the version 17.5 of OPNET Modeler that supports protocols of LTE which were implemented conforming to the 3GPP Rel. 8 specification. The LTE model in OPNET provides high flexibility and supports different scenarios and configurations. Moreover, the LTE model takes advantage of powerful statistical evaluations tools, together with the graphical user interface of the OPNET simulator. The basic nodes of LTE architecture in OPNET are UE, eNodeB and Evolve Packet Core (EPC). In the uplink direction the data flows form the UE to the eNodeB, then through the EPC and ends up at the server. In the downlink direction the data flows in the opposite direction. The eNodeB is connected with the UEs via the radio bearers and with the EPC via backhaul connection.

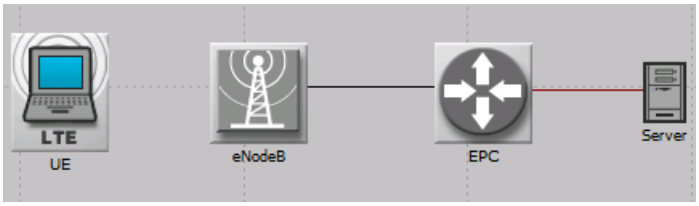


Figure 24. LTE nodes in OPNET Modeler

3.2.2 Simulation description

The following simulation aims to evaluate the gains of the utilization of VBSs (Virtual Base Station - a UE playing the role of a base station) in an LTE environment. I compare two scenarios with the same topology. One of these scenarios uses VBSs, in contrast with the other that deploys conventional communication links between the UEs and the eNodeB.

3.2.3 Simulation topology

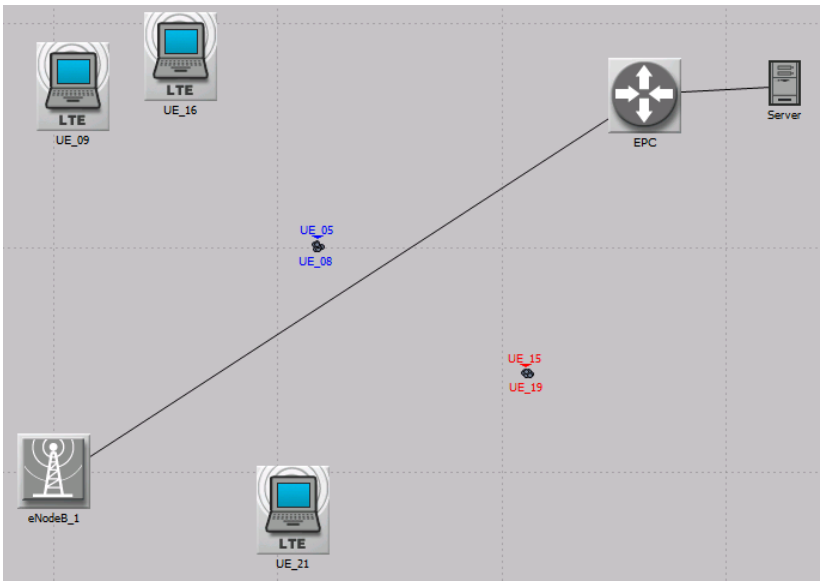


Figure 25. OPNET Simulation topology

The simulation consists of the following two scenarios:

1. Virtual Base Station Disabled Scenario
2. Virtual Base Station Enabled Scenario

There is one eNodeB and 24 UEs in the coverage area of the eNodeB. Three UEs are scattered randomly within the range of the eNodeB, the rest 18 UEs are forming two clusters of 9 UEs (blue and red, as seen in the figure). The UEs belonging in the same cluster are close to each other. In the VBS Enabled Scenario, in each cluster, a UE is manually selected to serve as the VBS of the remaining UEs of the cluster. My scenario topology follows the D2D cluster model (as seen in the figure below) proposed in [52] where UEs are normally distributed around each cluster center. All the UEs have a distance of up to 10 meters from their serving VBS. All the UEs are sending/receiving 20 Kbps traffic to/from the server. The pathloss model that is used is Urban Microcell (3GPP). The transmission bandwidth configurations for the LTE air interface are set to LTE 20 MHz FDD for both UL SC-FDMA and DL OFDMA channels.

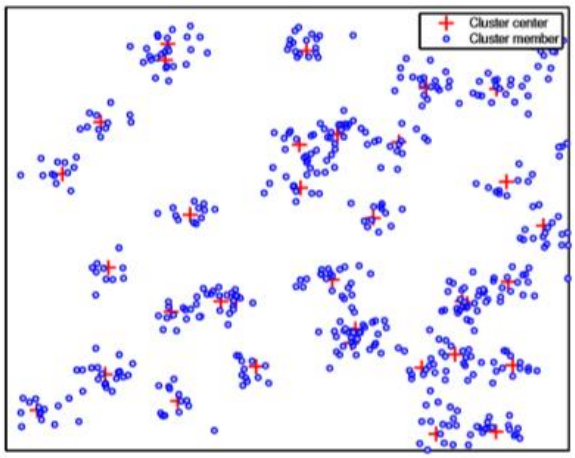


Figure 26. D2D cluster model where UEs are normally distributed around each cluster center [52].

Antenna Gain (dBi)	2.0
Battery Capacity	11
Downlink MIMO Transmission Tech...	Use Serving eNodeB Setting
Maximum Transmission Power (W)	1.0
Modulation and Coding Scheme Ind...	9
Multipath Channel Model (Downlink)	LTE OFDMA ITU Pedestrian A
Multipath Channel Model (Uplink)	LTE SCFDMA ITU Pedestrian A
Number of Receive Antennas	2
Number of Transmit Antennas	2
Pathloss Parameters	Urban Microcell (3GPP)
Receiver Sensitivity (dBm)	-200dBm

Antenna Gain (dBi)	15 dBi
Battery Capacity	Unlimited
MIMO Transmission Technique	Spatial Multiplexing 2 Codewords 2 Layers
Maximum Transmission Power (W)	5.0
Number of Receive Antennas	2
Number of Transmit Antennas	2
Operating Power	20
PHY Profile	LTE 20 MHz FDD Giorgos 1
Pathloss Parameters	Urban Microcell (3GPP)
Receiver Sensitivity (dBm)	-200dBm

Figure 27. UE and eNodeB PHY (Physical layer) parameters

3.2.4 Results

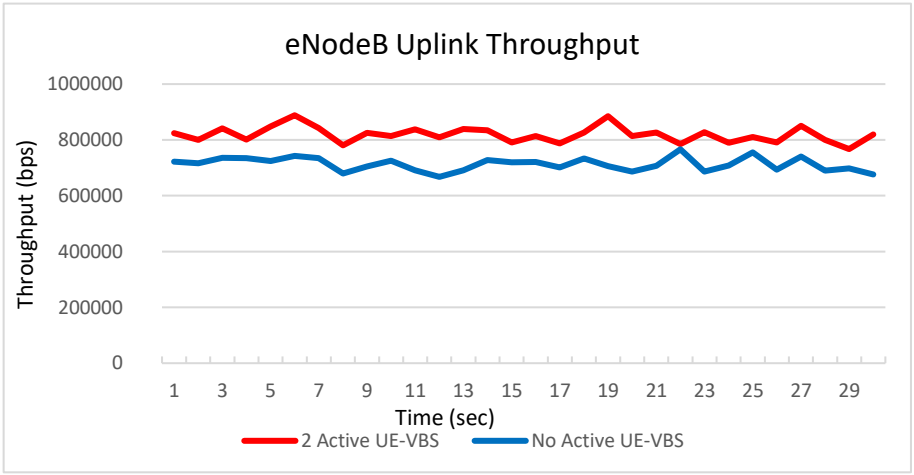


Figure 28. eNodeB Uplink Throughput

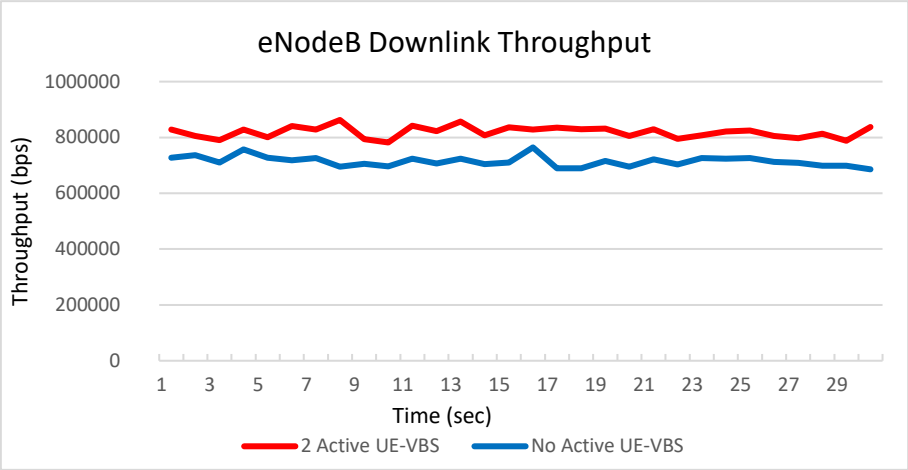


Figure 29. eNodeB Downlink Throughput

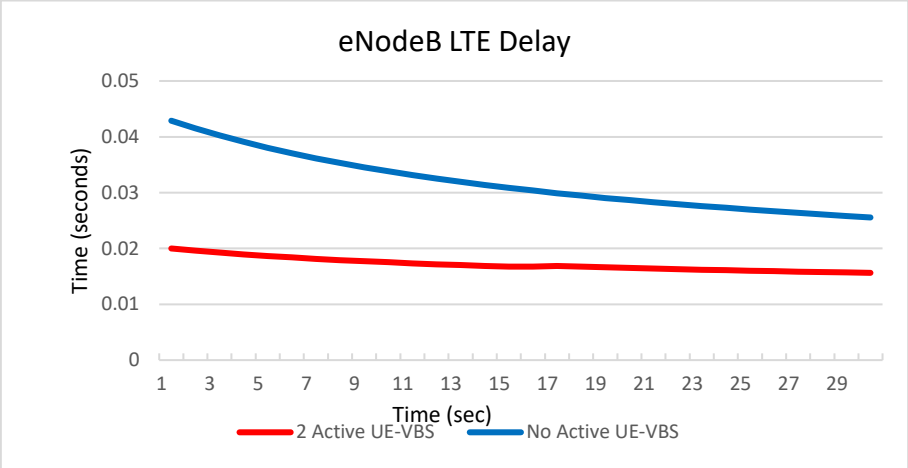


Figure 30. eNodeB LTE Delay

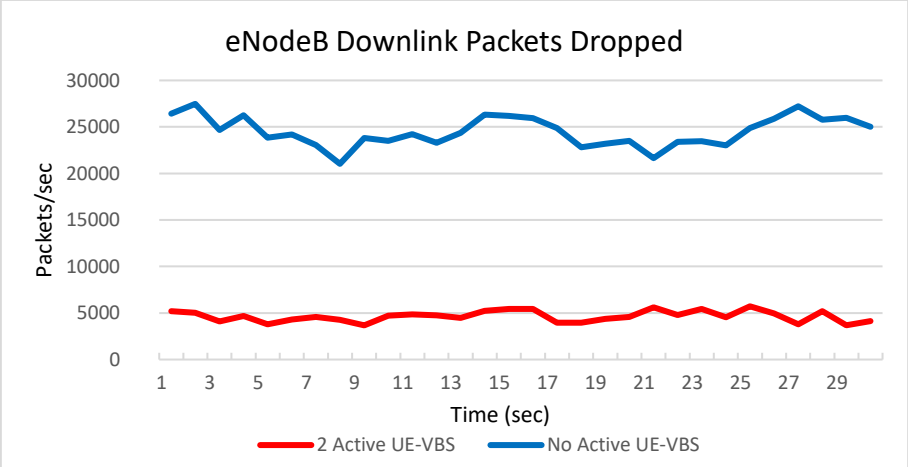


Figure 31. eNodeB Downlink Packets Dropped

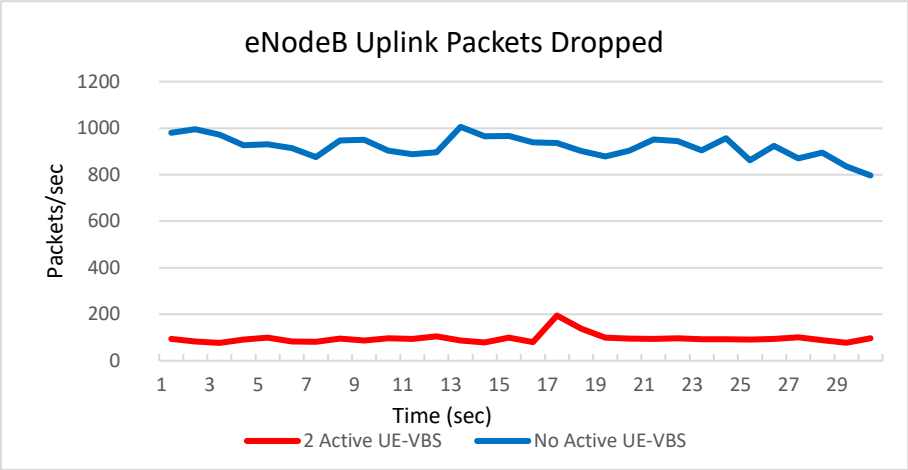


Figure 32. eNodeB Uplink Packets Dropped

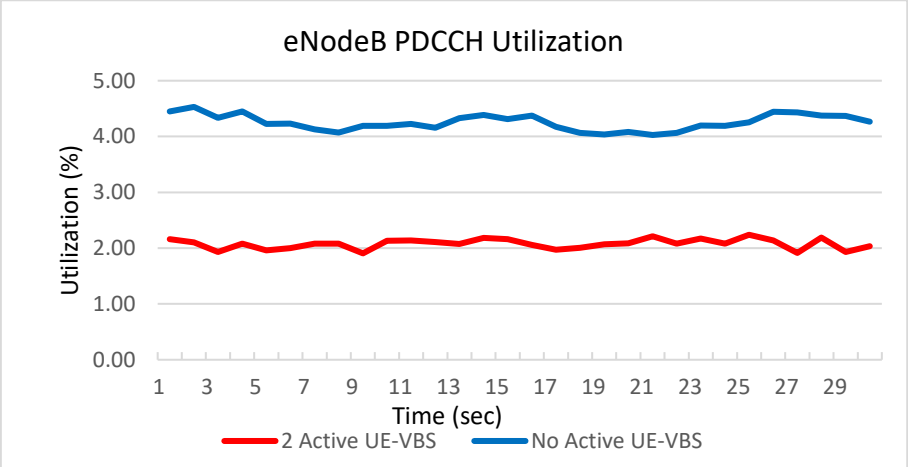


Figure 33. eNodeB PDCCH Utilization

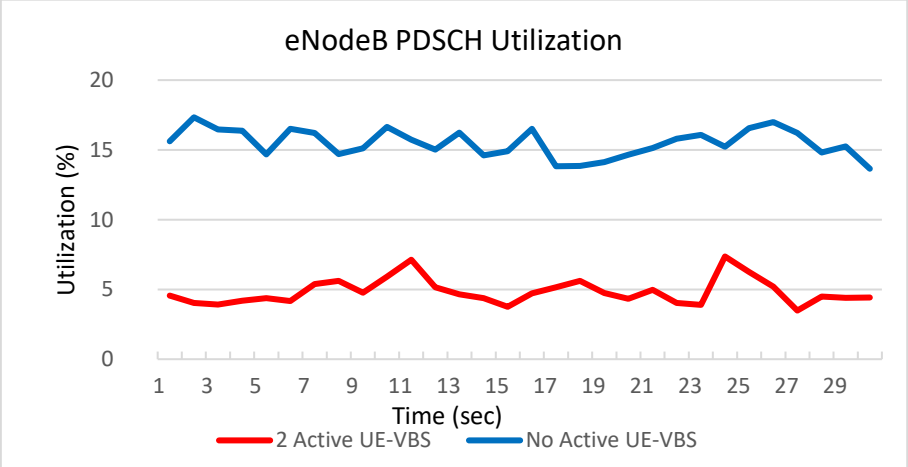


Figure 34. eNodeB PDSCH Utilization

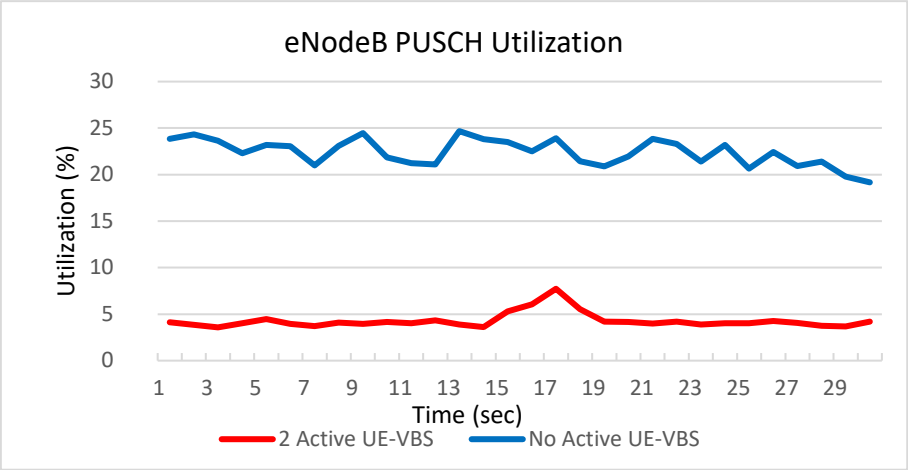


Figure 35. eNodeB PUSCH Utilization

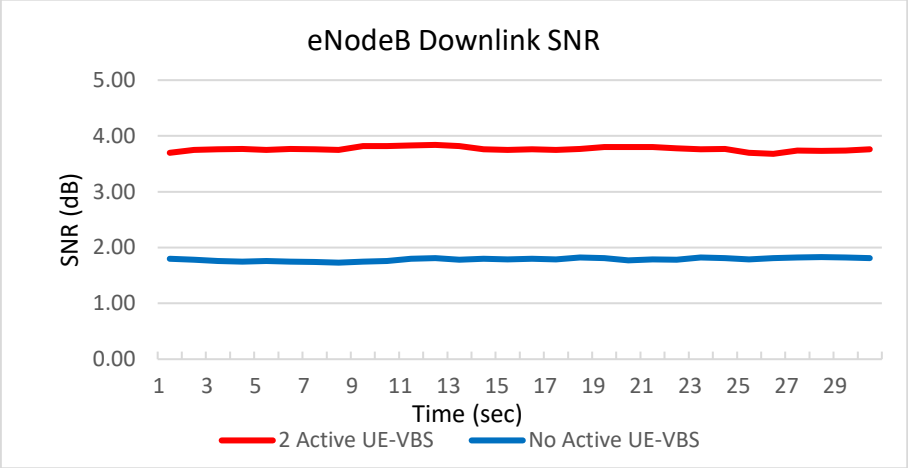


Figure 36. eNodeB Downlink SNR

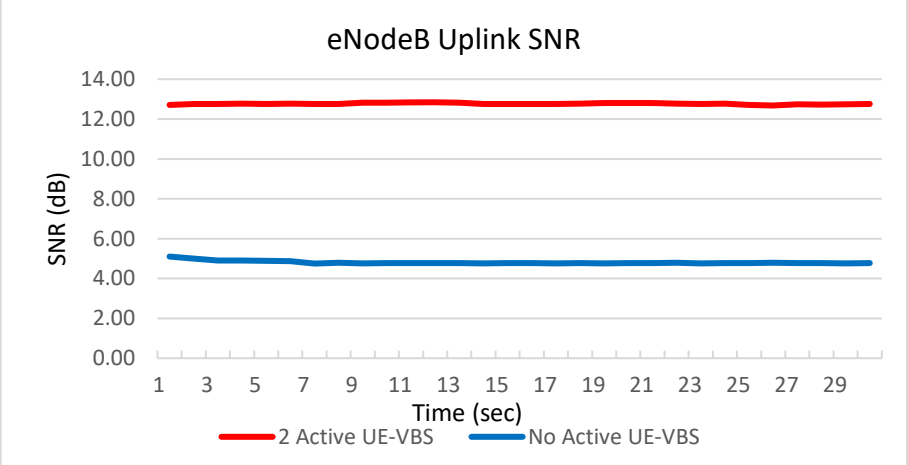


Figure 37. eNodeB Uplink SNR

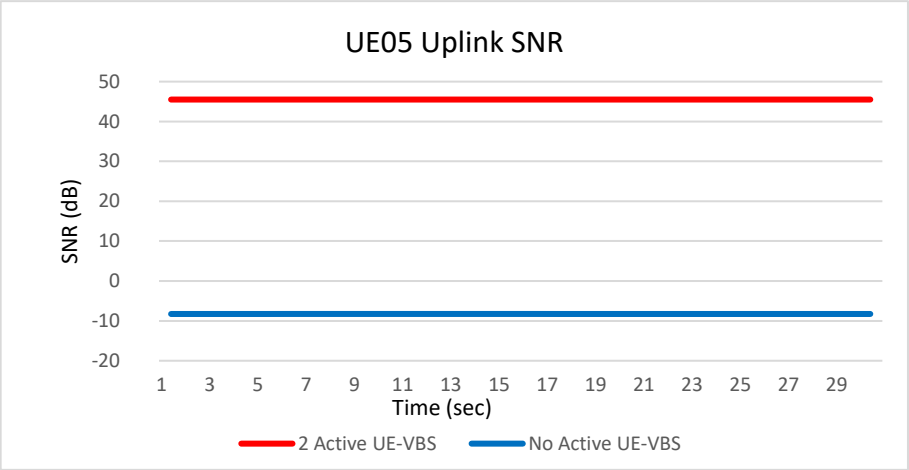


Figure 38. UE05 Uplink SNR

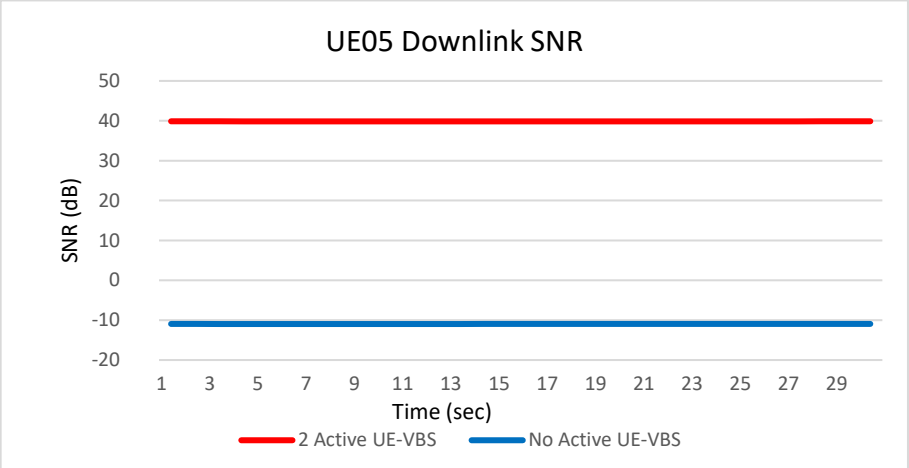


Figure 39. UE05 Downlink SNR

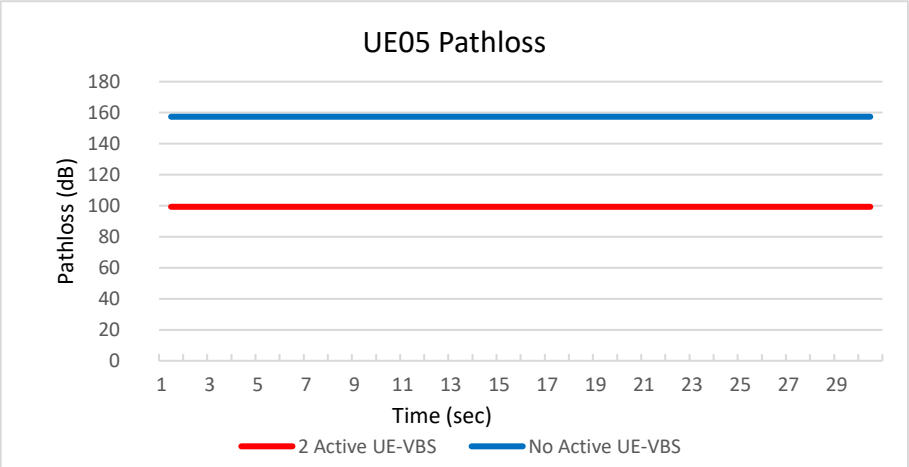


Figure 40. UE05 Pathloss

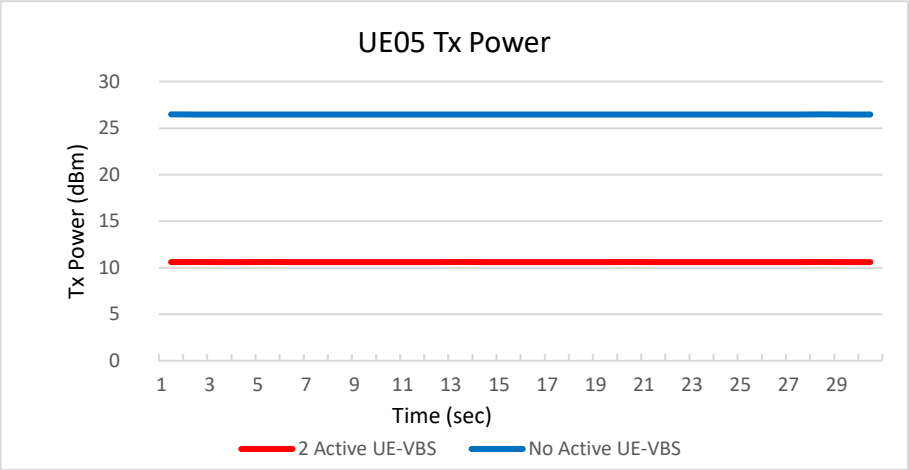


Figure 41. UE05 Tx Power

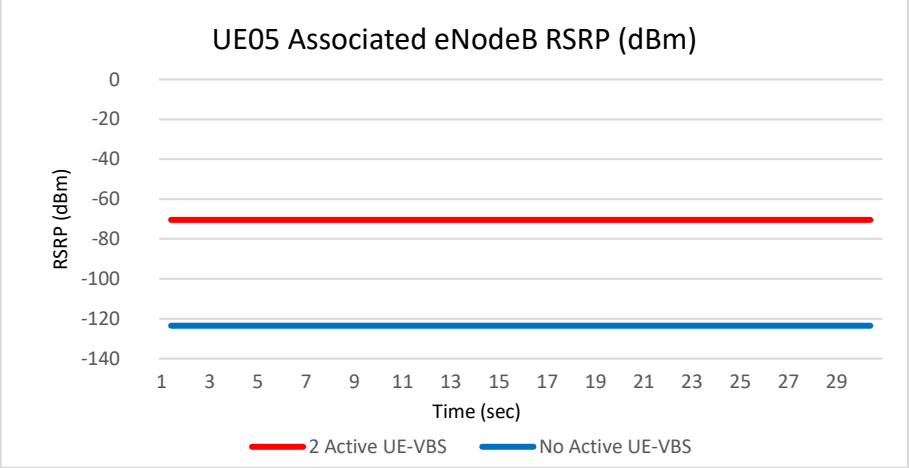


Figure 42. UE05 Associated eNodeB RSRP (dBm)

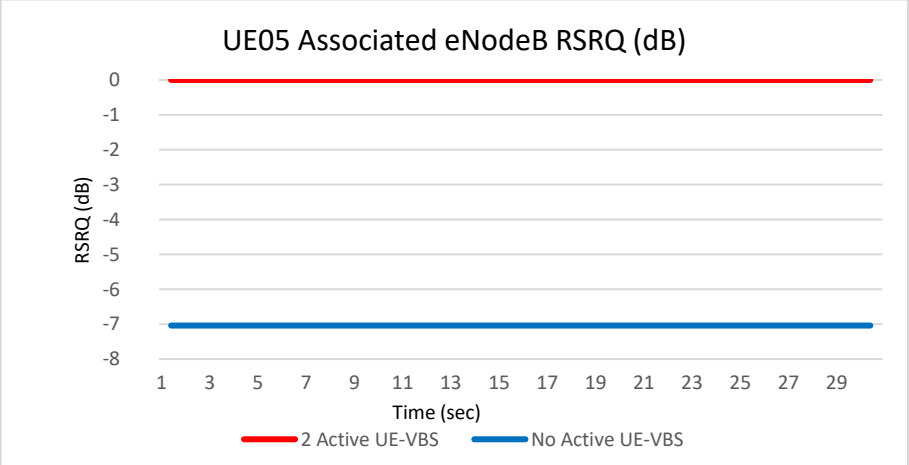


Figure 43. UE05 Associated eNodeB RSRQ (dB)

Result statistics description

RSRP

Reference Signal Received Power (RSRP) is the average received power of a single received signal. UE evaluates the power of the recourse elements that are used to transfer the signal and takes the average value.

RSRQ

Reference Signal Received Quality (RSRQ) shows the quality of the received reference signal. It gives additional information when RSRP is not enough to make a reliable cell selection decision.

Uplink SNR

This statistic records the uplink SNR values (in dB) for packet transmissions through the physical layer. For UE nodes, this statistic represents the SNR measured at the eNodeB for all packets arriving from a particular UE. For eNodeB nodes, this statistic represents the SNR measured at the eNodeB for packets arriving from all UE nodes in the cell.

Downlink SNR

This statistic records the downlink SNR values (in dB) for packet transmissions through the physical layer. For UE nodes, this statistic represents the SNR measured at a particular UE for all packets arriving from the eNodeB. For eNodeB nodes, this statistic represents the SNR measured at all UE nodes in the cell/sector for all packets arriving from the eNodeB.

Uplink packets dropped

This statistic records the uplink **packets** dropped (in packets/second) due to physical layer impairments. For UE nodes, this statistic represents the packet drops measured at eNodeB for all packets arriving from a particular UE. For eNodeB nodes, this statistic represents the packet drops measured at the eNodeB for all packets arriving from all UE nodes in the cell.

Downlink packets dropped

This statistic records the packets dropped in the downlink (in packets/second) due to physical layer impairments. For UE nodes, this statistic represents the packet drops measured at a particular UE for all packets arriving from the eNodeB. For eNodeB nodes, this statistic represents the packet drops measured at all UE nodes in the cell/sector for all packets arriving from the eNodeB.

LTE delay

LTE delay in seconds for all traffic arriving at this node that is delivered to higher layers. Delay is measure from the time the traffic arrives to the LTE layer of the eNodeB (downlink) or UE (uplink) until it is delivered to the higher layer of the corresponding UE (downlink) or eNodeB (uplink).

PDCCH utilization

This statistic records the channel utilization percentage of Physical Downlink Control Channel (PDCCH). Even though the size of PDCCH is reduced due to PDCCH resizing in some subframes when possible, the utilization is computed considering the full size of PDCCH.

PDSCH utilization

This statistic records the total percentage of Physical Downlink Shared Channel (PDSCH) channel utilization.

PUSCH utilization

This statistic records the total percentage of Physical Uplink Shared Channel (PUSCH) channel utilization.

Tx power

Current transmission power per each sub-channel in dBm.

.

	No Active UE-VBS		2 Active UE-VBS	
UE	**Total Power Consumption (mWh)**	**Estimated Life Time (hours)**	**Total Power Consumption (mWh)**	**Estimated Life Time (hours)**
UE_01	93.82	5.86	2.53	217.24
UE_02	7.77	70.74	2.38	231.24
UE_03	3.58	153.52	2.43	226.19
UE_04	19.37	28.39	2.26	243.6
UE_05	59.66	9.22	2.33	235.57
UE_06	20.91	26.3	2.24	246.03
UE_08	2.72	202.27	2.38	231.09
UE_09	116.34	4.73	3.13	175.9
UE_10	47.59	11.56	2.36	232.68
UE_12	2.32	237.25	2.24	245
UE_13	6.28	87.52	2.24	245.9
UE_14	2.47	222.82	2.21	248.89
UE_15	2.47	223.06	2.15	256.2
UE_16	3.99	137.83	5.07	108.49
UE_17	55.7	9.87	2.39	230.26
UE_18	7.99	68.84	2.22	248.27
UE_19	2.42	226.95	2.32	236.89
UE_20	2.32	236.95	2.38	231.48
UE_21	2.45	224.47	2.27	242.41
UE_22	2.48	222.07	2.2	249.79
UE_23	2.39	230.23	2.34	234.82

Table 2. UE Power Consumption

	No Active UE-VBS		2 Active UE-VBS	
UE	**Total Power Consumption (mWh)**	**Estimated Life Time (hours)**	**Total Power Consumption (mWh)**	**Estimated Life Time (hours)**
UE_07	2.87	191.64	105.09	5.22
UE_11	5.57	98.81	108.26	5.08

Table 3. UE-VBS Power Consumption

	No Active UE-VBS	2 Active UE-VBS
	Tx Power Consumption (mWh)	**Tx Power Consumption (mWh)**
eNodeB	38.5	16.95

Table 4. eNodeB Tx Power Consumption

3.2.5 Conclusions

In this performance evaluation, I used OPNET modeler to analyze and compare the performance of a network that utilizes the proposed virtual small cell approach against a conventional network where all connections are directly between the UEs and the BS.
My simulations confirmed that by enabling the use of a subset of the UEs as VBSs the throughput increases both in the UL (uplink) and the DL (downlink) directions. Moreover, the delay decreases significantly both in UL and DL. In addition, by enabling the VBSs the amount of the dropped packets at the eNodeB decreases dramatically. The signal to noise ratio (SNR) increases at the eNodeB as well as at the UEs. What is more, with the utilization of UE-VBSs, the received signal quality increases importantly at UEs. The pathloss is reduced and the transmit power of UEs is less. Due to the lower transmission powers, the power consumptions at UEs and at the eNodeB are minimized. Thus, the battery life time of UEs are prolonged as seen in the table 2. There are also power savings at the eNodeB and consequently money saving for the network operators. In addition, the utilization of channels at the eNodeB is reduced, thus more resources are freed up and the capacity can be increased.

The number of required communication links to the eNodeB decreases. Hence, the radio resources are shared between fewer communication links at the eNodeB, so bandwidth allocation and utilization are done more efficiently. Furthermore, the signaling overheads are offloaded from the eNodeB to the VBSs. By using a separate spectrum such as WiFi, Bluetooth, or mmWave for the short-distance intra-small cell communications the cellular network spectrum could be released only for the direct eNodeB communications. Therefore, the overall network capacity could further increase. On the other hand, a drawback of the virtual small cell concept is that the battery consumption of the UEs that act as VBSs increases. Another important challenge is the security and privacy of data, as the user’s data will not be transmitted directly to a BS but through another User’s device.

3.3 Clustering of virtual small cells

In the following scenario, there is one macro BS and 1000 UEs distributed in the coverage are of the BS. For the spatial distribution of UEs Poisson Cluster Process (PCP) was used. PCP is accounted by 3GPP to be suitable for user and BS distributions [53] and is also favoured for modelling small cell base stations in user hotspots. Hence PCP is appropriate for our scenario as we consider an ultra-dense cellular network. I use various clustering algorithms to organize the UEs into groups, where in every group one UE will be an active UE-VBS. Then, I compare the different algorithms according to their results, their required inputs, complexity, time, scalability etc. Afterwards, I benchmark the performance and scaling of the various clustering techniques using big datasets.

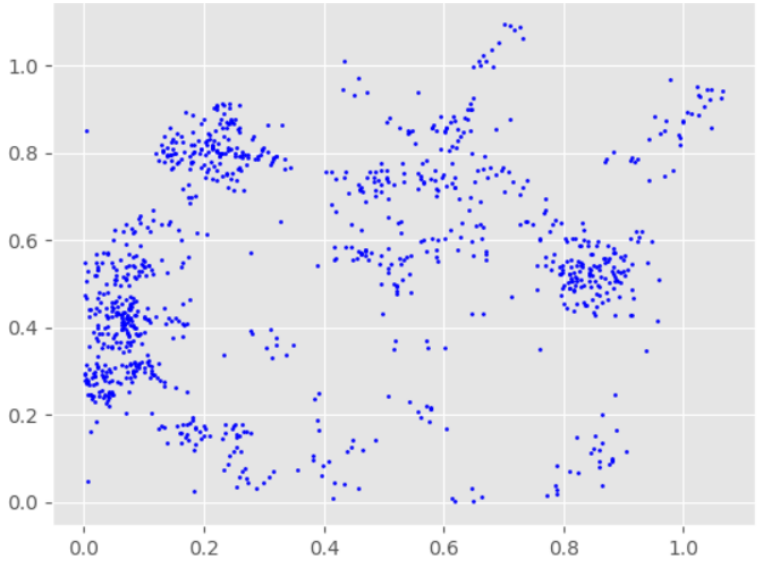


Figure 44. PCP distribution of UEs

3.3.1 Clustering in Python

For the comparison of different clustering algorithms, I used Python and especially the scikit-learn (version v0.19.1) module. Python is a general-purpose interpreted, high-level programming language, released in 1991 and created by Guido van Rossum. Scikit-learn [46] is a well-known module of Python that emphasizes on machine learning. It was started in 2007 by David Cournapeau and since then many volunteers contribute on its

development and maintenance. The Scikit-learn module is open-source and it is distributed under the simplified BSD, encouraging its use by academia. It provides easy to use and effective tools for data analysis and data mining. I particularly used the clustering tools of Scikit-learn for my experiments.

3.3.2 Contrast and comparison of different clustering approaches

Clustering has been a field of research for many decades and numerous algorithms are still being developed. Consequently, there are too many algorithms to be investigated even briefly [54]. I explored the most well-known and generally used clustering algorithms and I tried them on my dataset for the clustering of an ultra-dense network that utilizes the proposed UE-VBS technology. For each algorithm I present its strengths and weaknesses, its input parameters and most importantly its result and whether it is suitable for our proposed technology. Due to the low insight provided by numerical metrics, a visual representation of the clusters is very useful. There are no optimal standards for comparing cluster results, thus I used the visual representation of the cluster results as the main method for evaluating the results and finding the best choice. Moreover, for every algorithm I experimented and adjusted its input parameters in order to achieve a better outcome. As a result, I realized that its crucial for a clustering algorithm to have intuitive input parameters, so that the user can set the right values and achieve the optimal outcome. For the algorithms that require the number of clusters (n) to be given as input, I set n = 45 because this was the result of a related work on the same dataset.

K-Means

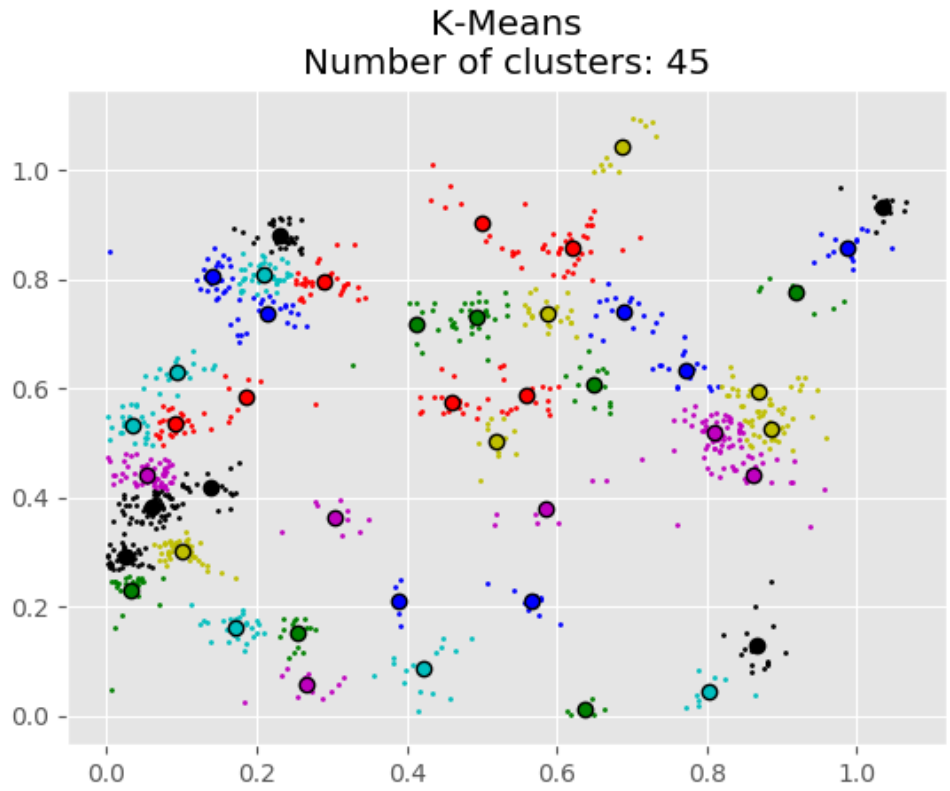


Figure 45. K-Means clustering result

The only required input parameter of k-means is the value of k (n_clusters in python scikit learn) which is the number of clusters (and centroids). I ran the algorithm with k = 45. The complexity of k-means is $O(k \times n \times T)$ in average case and $O(n^{(k+\frac{2}{p})})$ in worst case [55], where k is the number of clusters, n is the number of data points, T is the iteration number and p is the dataset dimension. Hence, k-means has the advantage of being fast, as it is a simple algorithm that is making relatively few computations (computes the distances between data points and centroids). In addition, there are optimizations which make k-means particularly efficient that only a few clustering algorithms can compete. However, many times it is useful to restart k-means numerous times [46] as its performance depends on the initial starting conditions. Thus, k-means lacks consistency and stability. This is because the algorithm begins with a random choice of centroids, so it may give different clustering results on different runs. Another weakness of k-means is that the number of clusters must be specified in advance, making it useless if the goal is

to let the algorithm figure the number by itself. Furthermore, k-means doesn't recognize data points as noise and adds them in a cluster no matter if they don't belong in it. This can result in cases where a point far away from the cluster ends up being a part of the cluster.

Regarding the result on my case, k-means achieved a decent clustering. Nonetheless, I already knew the desired number of clusters for my dataset but in a real-world scenario this will not be possible. Also, it can be seen from the figure above that a lot of the data points should be recognized as outliers as they are too far from the cluster center. Additionally, I observe that k-means is unable to represent density-based clusters, but clusters with high density are of primary importance for our scenario. Moreover, cluster centers (centroids) in k-means may not be necessarily a point in the dataset, but in our scenario, it would be useful to have a data point as a cluster center, so it can be the enabled UE-VBS. Nevertheless, with truly big amount of data, K-Means might be the only option for clustering the data.

Mean Shift

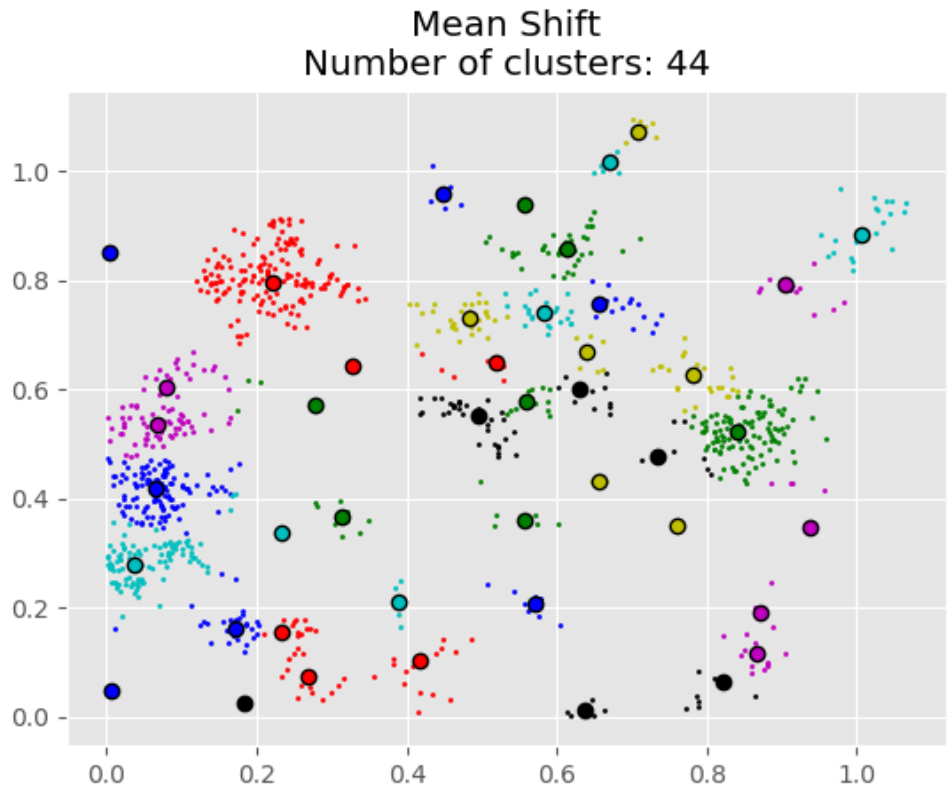


Figure 46. Mean Shift clustering result

The most important required input parameter of Mean Shift is the bandwidth of the kernel. Another parameter in the implementation of scikit-learn is the cluster_all parameter. When cluster_all is set to True, then orphan data points that are not within any kernel are assigned to the closest kernel [46]. Otherwise orhpans are not a member of any cluster (i.e. they are outliers). I ran the algorithm with parameters bandwidth=0.06 and cluster_all=False. The complexity of Mean Shift in the scikit-learn implementation is $O(T \times N \times \log n)$ in lower dimensions and $O(T \times n^2)$ in higher dimensions [46] where n is the number of data points and T is the number of iterations. The bandwidth of the kernel is easier to guess compared to guessing the number of clusters because it has a physical meaning. However, it needs to be chosen carefully for a successful clustering as with a different bandwidth the results can vary a lot. A strength of Mean Shift is that cluster centers converge towards points with maximum density, which is high desirable for our dataset. On the other hand, mean shift is fairly slow and is not highly scalable.

From the figure above, it can be seen that mean shift failed to achieve a good clustering on my data set. There are a lot of cluster centers that should be outliers. Also, there are some big clusters that would be better to split to more clusters. Moreover, maybe mean shift placed some cluster centers in high density areas, but the overall cluster is not highly dense.

Affinity Propagation

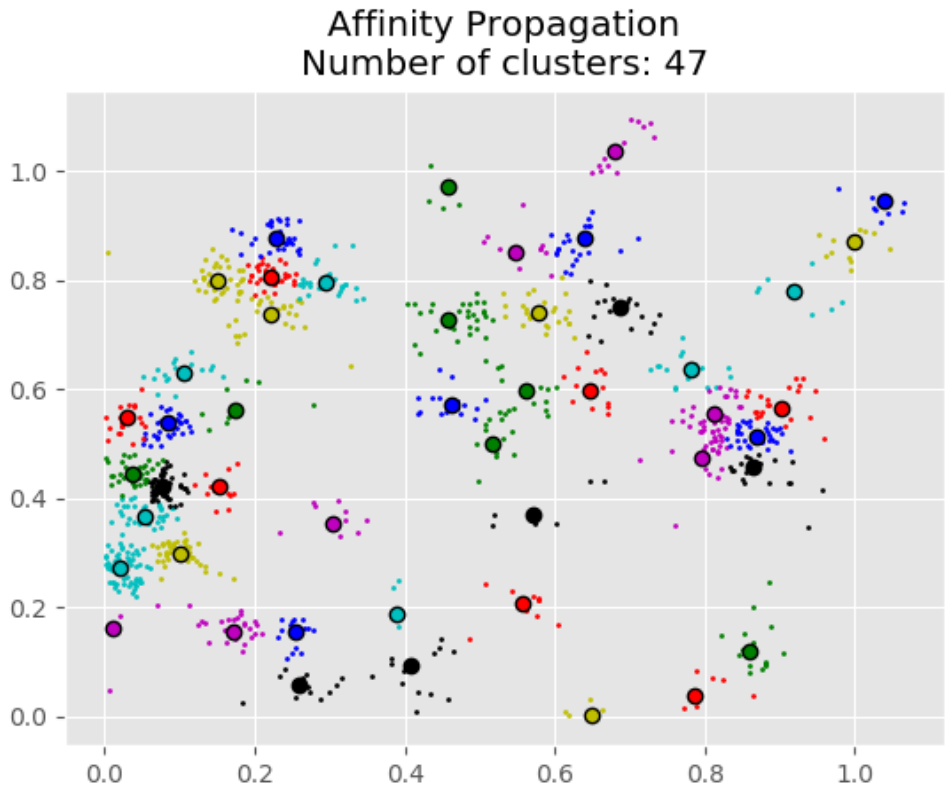


Figure 47. Affinity Propagation clustering result

The two important parameters of affinity propagation are the "preference" that controls the number of cluster heads (exemplars) and the "damping" factor which avoids numerical oscillations when updating the responsibility and availability messages [46]. I ran the algorithm with parameters preference= -0.04, damping=0.58. In the Scikit-learn implementation the default preference is the median dissimilarity, but it yields a very large number of clusters, so I set a different value that results in a better clustering.

The complexity of affinity propagation, which it is its main weakness, is quadratic in the number of data points. More precisely, $O(T \times n^2)$, where n is the number of data points and T is the number of iterations. Hence affinity propagation tends to be very slow, especially on big data sets, because its basic operations are computationally expensive.

While Affinity Propagation eradicates the number of clusters as an input parameter, it requires two other parameters. Another disadvantage of affinity propagation is that it adds all the points into cluster and does not create any noise or outlier points because it is a portioning algorithm. Also, it is an algorithm that assume that clusters are globular.

Agglomerative Clustering

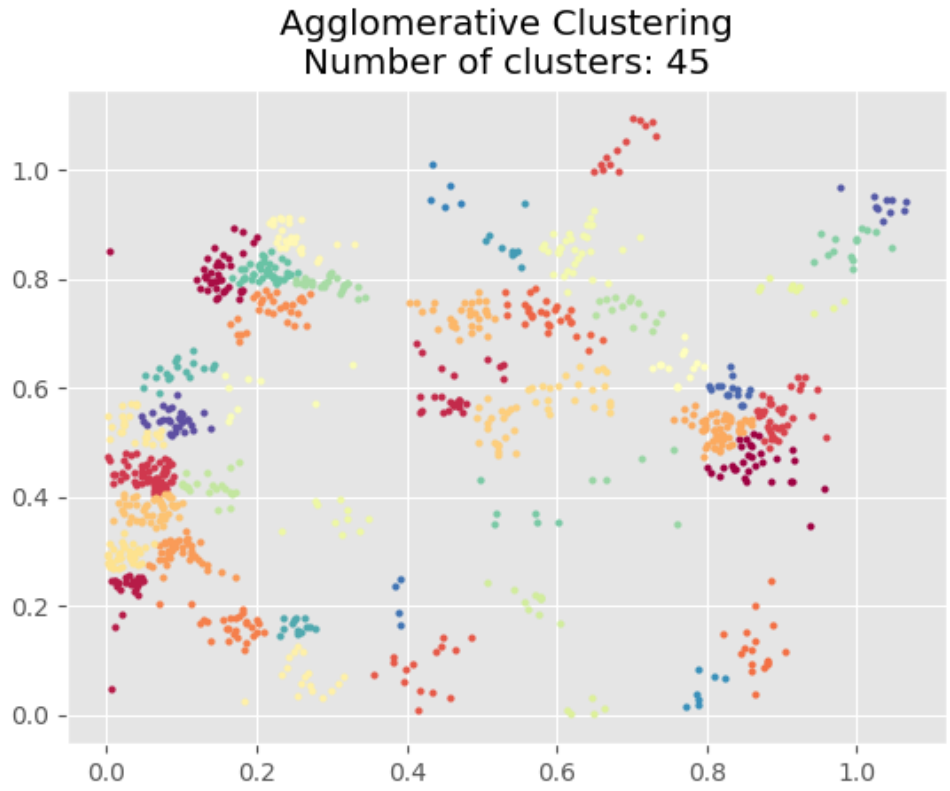


Figure 48. Agglomerative clustering result

The most important parameters of agglomerative clustering are the number of clusters (n_clusters), the metric used to compute the linkage (affinity) and the linkage criterion which defines the proximity method to be used. I ran the algorithm with n_clusters = 45, linkage = "ward" and affinity = "euclidean". These parameters are more detailed described in agglomerative clustering in the Cluster analysis section of this thesis.

Agglomerative clustering can scale to large datasets only with a use of a connectivity matrix because otherwise it considers all possible merges at every step [46]. An advantage of agglomerative clustering is that you can inspect its dendrogram and try to find a cut in the hierarchical structure that yields to a good clustering. Although, this advantage comes at the cost of higher complexity of $O(n^3)$. The scikit-learn implementation takes a different approach and requires the number of clusters as input which is a big drawback. Some other benefits of this algorithm are that it does not assume clusters as globular and its performance can be good especially with the "fastcluster" implementation of sklearn.

Like the three previous algorithms, this one too is adding noise points in our clusters and does not support outliers. In addition, agglomerative clustering does not provide a cluster head.

DBSCAN

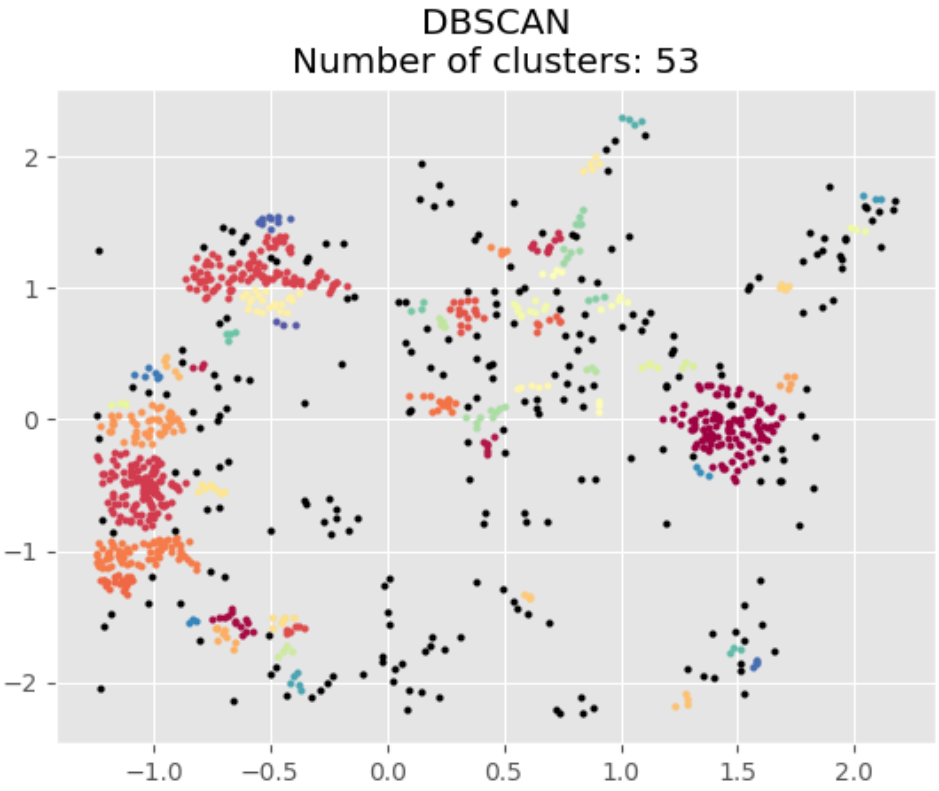


Figure 49. DBSCAN clustering result

DBSCAN has two important parameters, the epsilon value (ε or eps in sklearn) and MinPts (or min_samples in sklearn). The ε value is the minimum distance among two points for them to be in the same neighborhood. MinPts is the minimum number of datapoints in a neighborhood for a data point to be a core point. In other words, these two values define the density threshold (i.e. the minimum number of data points (MinPts) in a sphere of radius ε). I ran the algorithm with eps=0.06 and min_samples=3. DBSCAN, as defined by its name (Density-Based Spatial Clustering of Applications with Noise), creates clusters from data points with high density and recognizes sparse background points (noise) as outliers. It is a good algorithm for data with clusters of similar density. On the other hand, for clusters with varying density it does not perform well, thus HDBSCAN was introduced. Some benefits of DBSCAN is that it does not require a

predefined number of clusters, clusters are not globular, and it can find arbitrarily sized and shaped clusters quite well. Another advantage of DBSCAN is its performance which makes it able to handle large datasets.

The combination of MinPts and ε provides the user the choice of density and the algorithm finds only the clusters equal or above this density. This can either be a strength of DBSCAN if the user knows the right density for the underlying application, or a weakness if the user has to experiment with different combinations of MinPts and ε.

In the basic case, the time complexity of the DBSCAN algorithm is $O(n \times t)$, where n is the number of data points and t is the time required to find the points in the ε neighborhood. In the worst case its complexity is $O(n^2)$. Nevertheless, with the utilization of efficient data structures like kd-trees, the complexity for low dimensional data can be $O(n \times logn)$ [54]. The general storage complexity of DBSCAN is $O(n)$ because it keeps only a few information for every data point.

From the clustering of DBSCAN on my dataset, I observe that it successfully found the areas with high density and ignored the sparse data points. This is a desirable result because the dense UEs will form clusters where only one UE (i.e. the UE-VBS) will connect with the BS and the rest UEs will form intra-cell connections. On the other hand, sparse UEs will be preferred to form direct connections with the BS. The fact that DBSCAN can form arbitrary shaped cluster is not desirable in our scenario, as it would be better for the clusters to have round shapes so the UE-VBS can be in the center and the rest of the UEs around it.

HDBSCAN

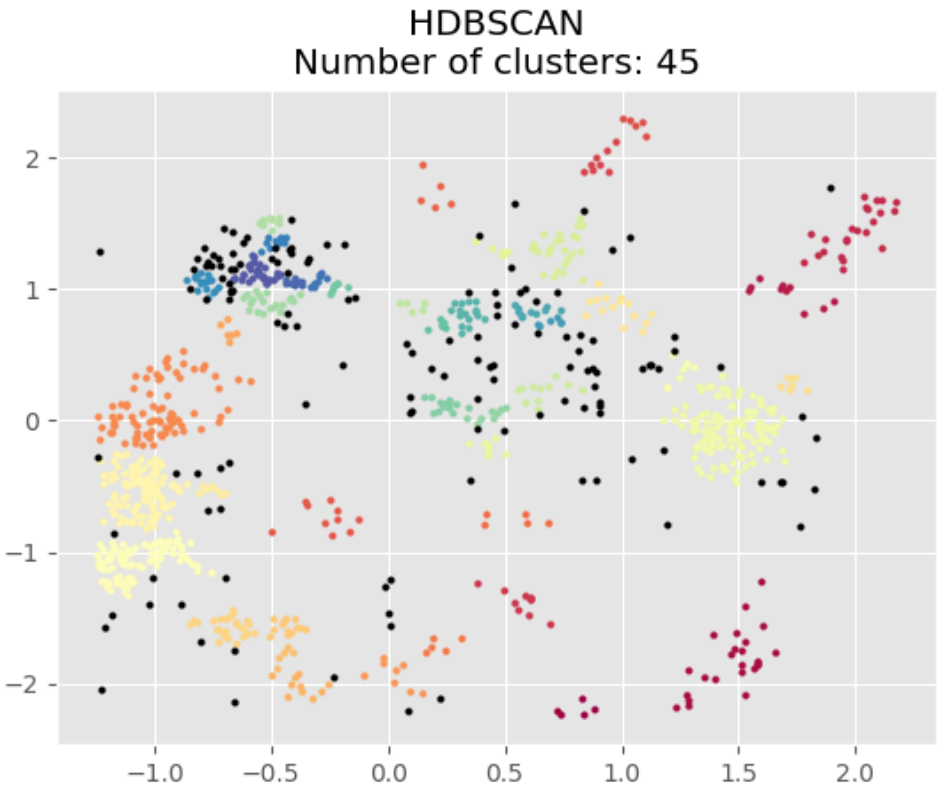


Figure 50. HDBSCAN clustering result

HDBSCAN is a conversion of DBSCAN into a robust hierarchical clustering algorithm. Unlike DBSCAN it does not require ε as an input value. Instead it performs DBSCAN with different ε values and finds the clustering with the best stability over ε [56]. HDBSCAN is not implemented in sklearn module, but there is an hdbscan standalone module in python. Its input parameters are MinPts (min_samples) which is inherited from DBSCAN and has the same role in the algorithm and min_cluster_size which is the minimum cluster size. I ran the algorithm with the parameters min_cluster_size=5, min_samples = 3.The improvements of HDBSCAN over DBSCAN is the improved performance on low dimensional data, the allowance of clusters with varying densities and the robustness to parameter selection [56].

In the figure above, it can be confirmed indeed that HDBSCAN allowed clusters of varying density. Thus, it found some bigger clusters and less outliers in comparison with DBSCAN. This was intended by the developers of this algorithm, but in my scenario, this

is not advantageous. It is much preferred to find only the clusters with a specified high density, so as the UEs to be really close to their UE-VBS and maximize the benefits of the virtual small cell technology. Moreover, just like in the previous algorithm, as it would be better for the clusters to have round shapes.

Spectral Clustering

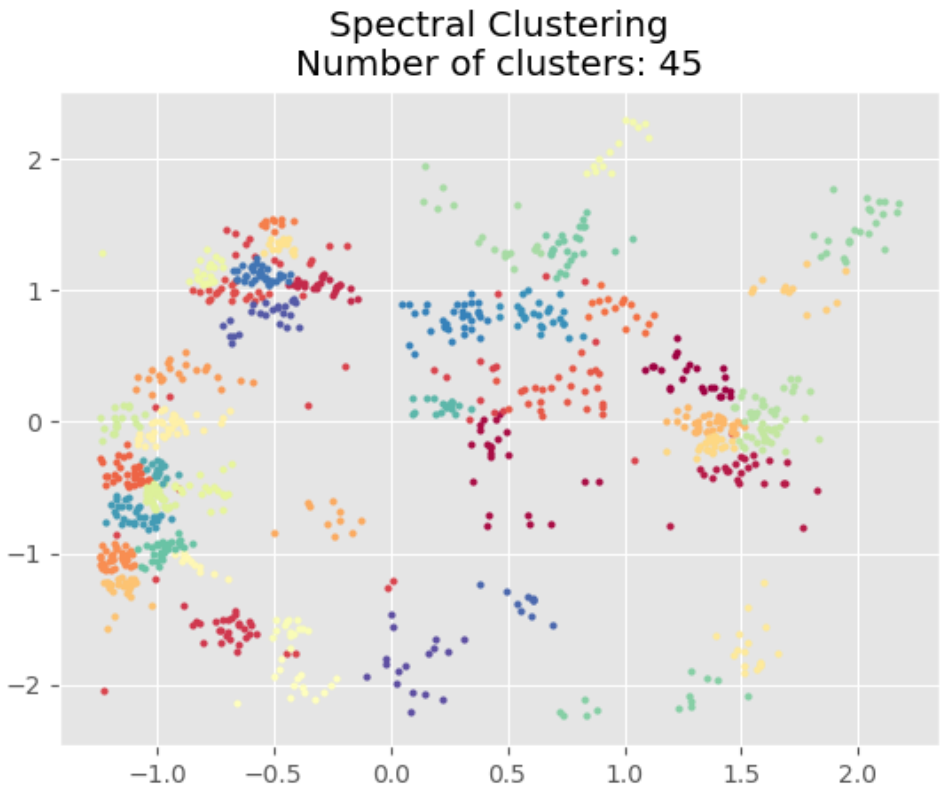


Figure 51. Spectral clustering result

The most important parameters of Spectral clustering are the number of clusters (n_clusters), the eigenvalue decomposition strategy to use (eigen_solver) and the affinity metric (affinity) [46]. The sklearn module also provides the n_neighbors parameter which is the number of neighbors that will be used when building the affinity matrix with the closest neighbors method. I ran the algorithm with the parameters n_clusters=45, eigen_solver='arpack' and affinity="nearest_neighbors". Spectral Clustering is efficient if the affinity matrix A is sparse and when the number of clusters is relatively small.

From the results of Spectral Clustering on my data set, it can be seen that it is another algorithm that lets noise points pollute the clusters. Moreover, Spectral Clustering requires the number of clusters as an input parameter which is a major drawback. The advantage of this algorithm is the dimensionality reduction, which is not necessarily needed in my scenario. In addition, Spectral Clustering in sklearn uses the k-means algorithm to cluster the data after the dimensionality reduction. Thus, it inherits all the limitations of k-means.

Run	K-Means	Mean Shift	Affinity Propagation	Agglomerative Clustering	DBSCAN	HDBSCAN	Spectral Clustering
1	0.150552	0.875353	0.902957	0.044311	0.006121	0.020036	0.362166
2	0.133137	0.876363	0.837579	0.028037	0.004011	0.020206	0.318985
3	0.148614	0.913842	0.835922	0.029230	0.005089	0.024038	0.384284
4	0.155409	0.893479	0.864550	0.027658	0.004005	0.020026	0.321140
5	0.136215	0.921729	0.834449	0.027771	0.004004	0.020027	0.313796
6	0.141427	0.884071	1.140028	0.024770	0.004007	0.020305	0.318687
7	0.140808	0.973454	0.874665	0.028003	0.008014	0.021536	0.324338
8	0.145943	0.942506	0.895740	0.026045	0.004140	0.020707	0.336871
9	0.143549	0.881305	0.844976	0.027998	0.002250	0.021122	0.323387
10	0.149394	0.887020	0.865153	0.024032	0.004003	0.020026	0.319829
Average	0.144505	0.904912	0.889602	0.028786	0.004564	0.020803	0.332348

Table 5. Run Time for each algorithm in Seconds

Algorithm	Input Parameters	Scalability	Usecase	Metrics
K-Means	Number of clusters	Very large number of data points. Medium number of clusters.	General-purpose, Even cluster size, Flat geometry, Not too many clusters.	Distances between points
Mean-shift	Kernel Bandwidth	Medium number of data points.	Many clusters, Uneven cluster size, Non-flat geometry.	Distances between points
Affinity propagation	Damping, Sample Preference	Not scalable on the number of data points.	Many clusters, Uneven cluster size, Non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Agglomerative clustering	Number of clusters, Linkage metric, Linkage criterion	Medium number of data points and number of clusters.	Many clusters, Connectivity constraints, Non-Euclidean distances	Any pairwise distance
DBSCAN	Maximum radius of the neighborhood. Minimum number of datapoints in a neighborhood.	Very large number of data points, medium number of clusters.	Non-flat geometry, Uneven cluster sizes	Distances between nearest points
HDBSCAN	Minimum cluster size, Minimum number of datapoints in a neighborhood	Very large number of data points, medium number of clusters.	Non-flat geometry, Uneven cluster sizes	Distances between nearest points
Spectral clustering	Number of clusters	Medium number of data points, small number of clusters.	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbour graph)

Table 6. Summarized comparison of the clustering algorithms [46]

3.3.3 Benchmarking performance and scaling of different clustering approaches

Performance and scaling depend on the clustering algorithm as well as the implementation. The programming language and the data structures can have a high influence on the performance of the clustering technique. Instead of just analyzing the algorithms and their implementations to derive the asymptotic time and space complexity, it is better to run them and get empirical results. Specifically, I ran together all the seven clustering implementations in python (sklearn) that I compared in the previous section. Then I collected all the requisite data and used the seaborn library, which provides an interface for drawing statistical graphics, to visualize the results.

I ran each clustering technique many times with different dataset sizes. In addition, for each data size I created several different random datasets and extracted the average performance to get more reliable statistics. Because some algorithms are not scalable, I divided the performance benchmarking in three parts. Firstly, I ran all the algorithms with medium sized datasets. Then I aborted the algorithms that performed poorly and ran the remaining algorithms with bigger sized datasets. Finally, I ran the fastest algorithms with even bigger datasets.

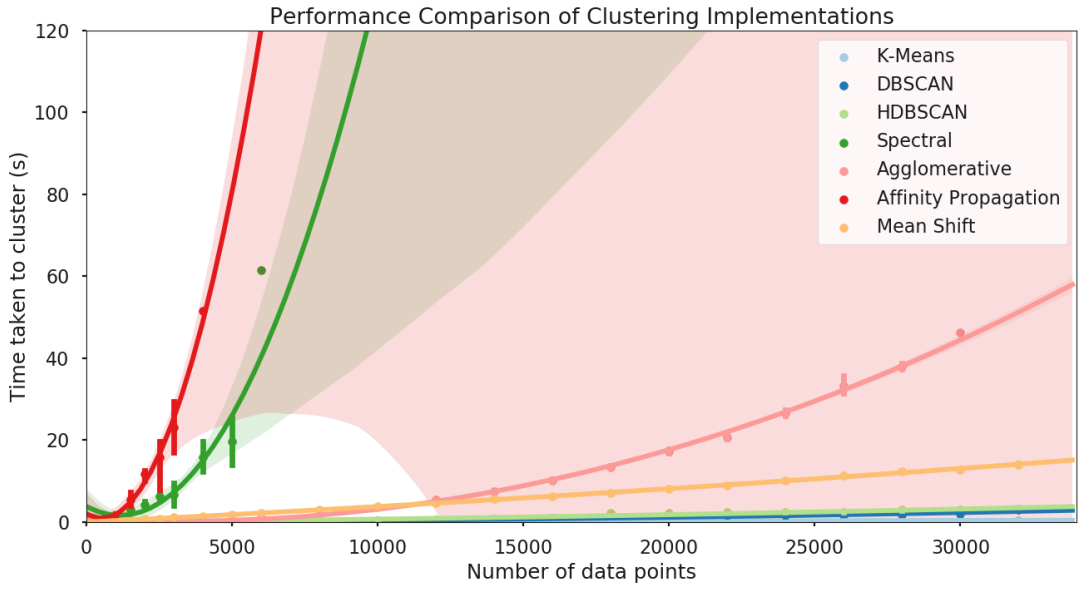


Figure 52. Performance comparison of clustering techniques

Firstly, I ran all the seven clustering techniques (K-Means, Mean-shift, Affinity propagation, Agglomerative clustering, DBSCAN, HDBSCAN and Spectral Clustering) with datasets of sizes from 0 to 32,000 data points. The results are shown in the figure above. It is confirmed that Affinity Propagation is not scalable on the number of data points as it took over two minutes to cluster 5,000 points. The second slowest implementation is Spectral Clustering which took more than two minutes to cluster the 10,000-point dataset. The rest of the algorithms achieved much higher performance. However, Agglomerative Clustering started to get slower after the 10,000-point dataset and is ranked as the third slowest implementation. The fourth slowest is Mean Shift whereas the rest of the implementations (K-Means, DBSCAN and HDBSCAN) were really fast on these datasets. Notably, for practical purposes Affinity Propagation and Spectral Clustering cannot be used to cluster datasets with more than 10,000 points (if time is a constrain).

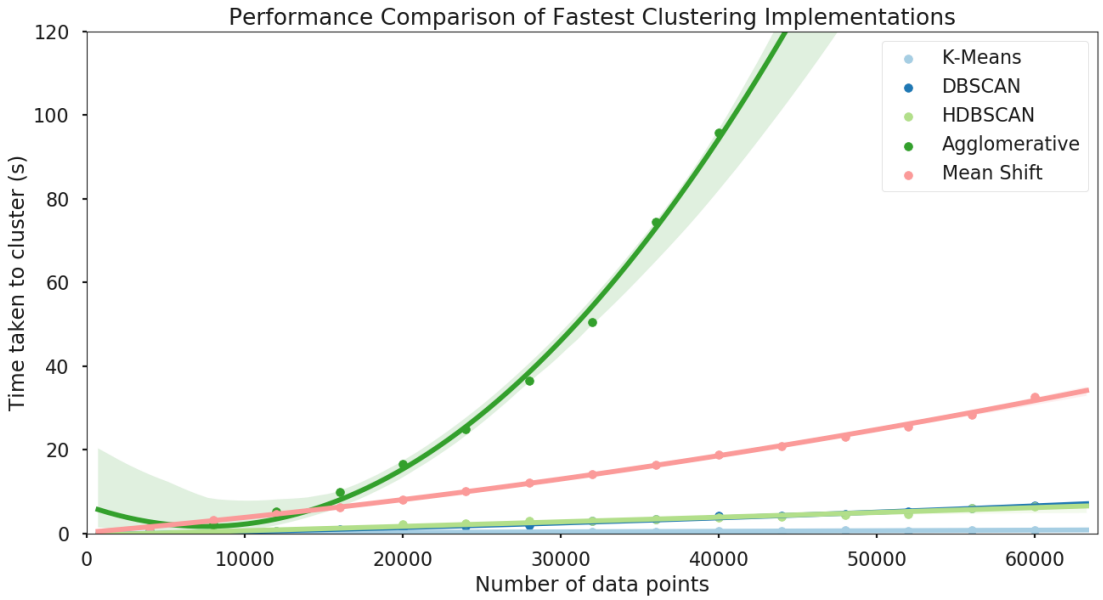


Figure 53. Performance comparison of the fastest clustering techniques

Consequently, I ran again all the implementations except Affinity Propagation and Spectral Clustering to see how they will perform on bigger datasets. The difference in the performance of DBSCAN and HDBSCAN in contrast with the performance of K-Means has slightly started to show up. As the number of points in the datasets increases,

DBSCAN and HDBSCAN take longer time to cluster them. On the other hand, K-Means is still managing to cluster the 60,000-point dataset in under one second. Agglomerative Clustering took more than two minutes to cluster the 45,000-point dataset, thus it is another outsider in terms of performance and scaling. Mean Shift is still managing to be in the four faster implementations. Nevertheless, Mean Shift is slower than the remaining three competitors (K-Means, DBSCAN and HDBSCAN).

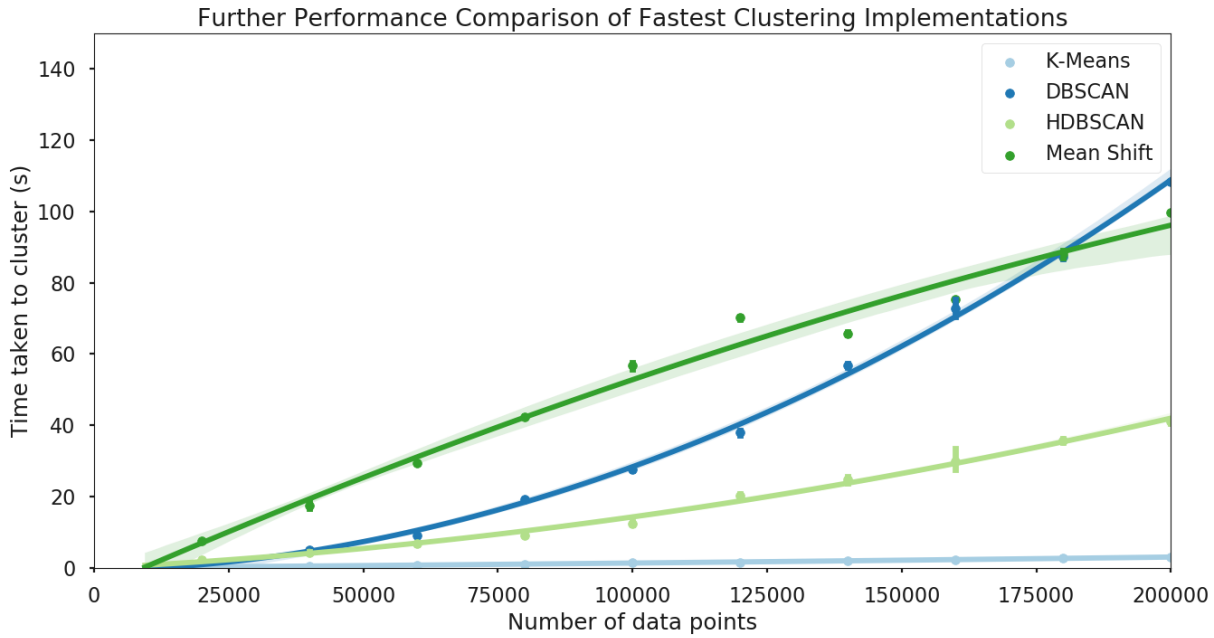


Figure 54. Further Performance comparison of the fastest clustering techniques

Finally, I ran the four fastest algorithms (K-Means, DBSCAN, HDBSCAN and Mean Shift) with even larger datasets (up to 200,000 points). Here we can see that HDBSCAN scales better than its predecessor DBSCAN and what has been discussed in the theory is confirmed in practice. Moreover, at the 175,000-point dataset DBSCAN and Mean Shift took the same amount of time to cluster the points. In summation, K-Means is the fastest and most scalable algorithm and HDBSCAN the second one.

3.3.4 Comparison conclusions

I evaluated and compared the performance of seven well-known and generally used clustering algorithms: K-Means, Mean-shift, Affinity propagation, Agglomerative clustering, DBSCAN, HDBSCAN and Spectral Clustering. For the comparison I used a dataset with 1000 points distributed with Poisson Cluster Process (PCP). Also, I benchmarked the performance and scaling of the algorithms.

The evaluation of clustering algorithms is not a trivial task because each algorithm has its strengths and weaknesses and needs to be run with varied parameter settings. The choice of the best clustering algorithm depends on the particular purpose and application. In my case the purpose is the clustering of an ultra-dense network that utilizes the proposed UE-VBS technology. In each cluster, only one UE (i.e. the UE-VBS) will be directly connected with the main BS while the rest UEs in the cluster will communicate through their UE-VBS. Hence, for my application a good algorithm will produce clusters based on the density of points (UEs). Moreover, it is desired for the clusters to have a round shape so the active UE-VBS can be in the middle and the rest of the UEs around it. Another requirement is for the algorithm to give us the choice to specify the distance between the UEs of each cluster. Moreover, it would be useful if we could specify the minimum and maximum number of UEs in a cluster. In addition, the algorithm must be efficient enough, so it can be scaled to big datasets and used in real-time applications.

With the aforementioned requirements in mind and the score of each algorithm in terms of their results, required inputs, complexity, time and scalability the two algorithms that stood out are DBSCAN and HDBSCAN. DBSCAN was the most suitable clustering method for my scenario. My comparative study showed that DBSCAN took the least time to form clusters. Moreover, in the benchmarking analysis, DBSCAN was in the top three most scalable and efficient algorithms. In addition, it successfully found the noise points and marked them as outliers. Also, it allows us to define the maximum radius of the neighborhood of points as well as the minimum number of datapoints in a neighborhood. However, as the number of points in the dataset increased, DBSCAN started to slow down. On the other hand, HDBSCAN performed much better. In addition, HDBSCAN allows clusters of varying density and has better input parameters than DBSCAN.

A notable conclusion derived from my experimental results is that a density-based clustering technique is most fitting for the clustering of Virtual Small Cells.

3.3.5 Density-based Clustering algorithms

As my comparison of various clustering algorithms in the previous section showed that a density-based clustering technique can produce good clusters for a UE-VBS network, I decided to explore more density-based clustering algorithms. Unfortunately, there are no libraries in python for density-based clustering algorithms other than DBSCAN and HDBSCAN. Thus, I used the ELKI (Environment for Developing KDD-Applications Supported by Index-Structures) tool to further explore density-based clustering algorithms.

ELKI is an open source data mining software written in Java which emphasizes on the research of unsupervised methods in cluster analysis and outlier detection. ELKI achieves high performance and scalability using data index structures that provides major performance gains. It provides a large set of highly parameterizable algorithms, to allow the evaluation and benchmarking of algorithms by researchers and students [57].

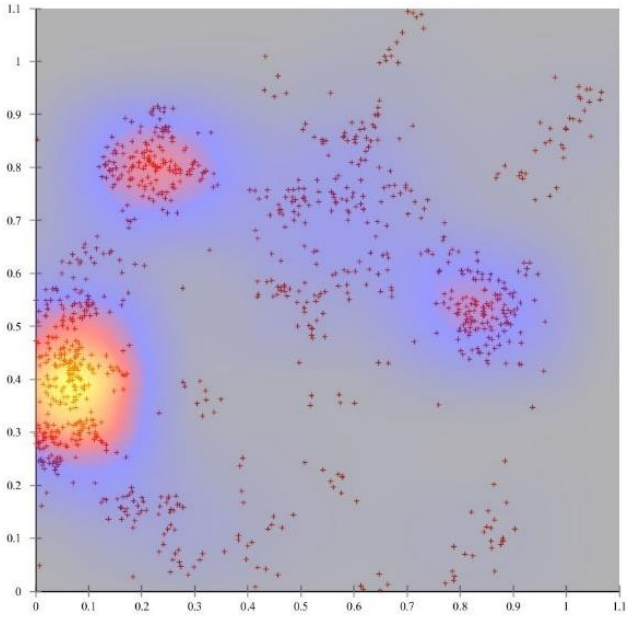


Figure 55. Density estimation overlay of my dataset in ELKI

In density-based clustering, a cluster is a group of data points spread over a contiguous region of high-density points. Density-based clusters are dense areas separated from each other by sparser areas (low density of points). The data points positioned in the sparse areas are normally marked as outlier points (noise) [58] and are not assigned to any cluster. The density of the areas of outlier points is lower than the density in any of the clusters. Moreover, in density-based clustering, for each cluster, the neighborhood of a given radius has to contain at least a minimum number of data points [37]. The key idea, is to keep expanding a cluster as long as the density and the number of points in the neighborhood exceeds some threshold [50]. Provided an index structure that supports region queries (which ELKI utilizes), density-based clusters can be efficiently formed by executing at most one region query per data point [48]. Furthermore, because of their local nature, dense connected areas in can have arbitrary shaped clusters. Additionally, in density-based clustering the number of clusters does not need to be specified beforehand. However, finding the correct parameters for standard density-based clustering can be challenging.

LSDBC

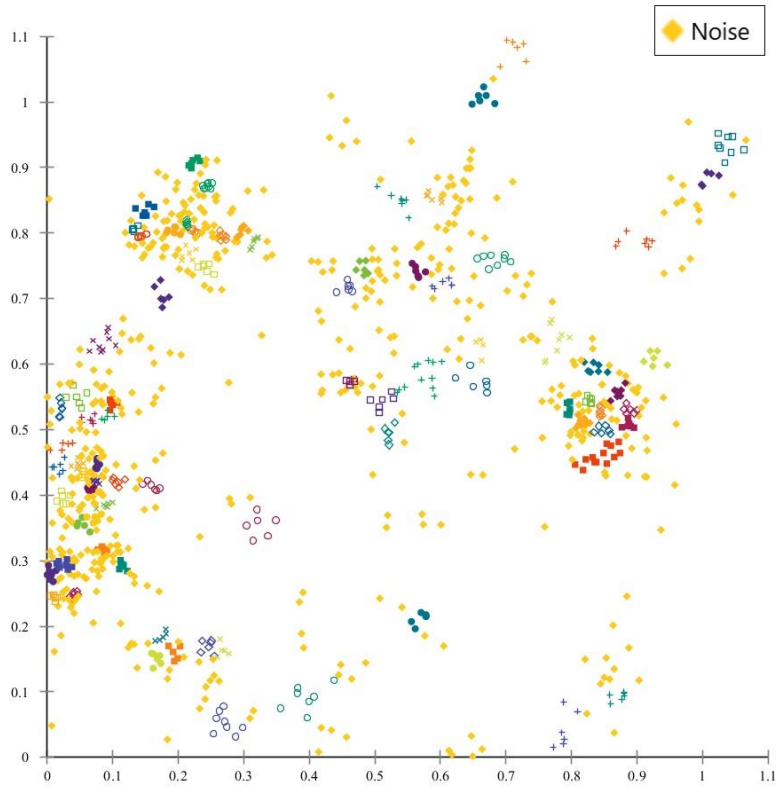


Figure 56. LSDBC clustering result

In LSDBC A k-nearest-neighbor density estimation is used to discover the local maxima of density which are used as cluster centers. Each cluster is expanded until the density falls below a predefined ratio of the center point's density. LSDBC has two input parameters, *k*, which is the order of nearest neighbor to consider for each data point for density calculation and *α*, that controls the boundary of the current cluster expansion based on its density. I run the algorithm on my dataset using the ELKI data mining software with parameters $k = 5$ and $a = 0{,}1$ and it took 37ms time to cluster the dataset. LSDBC can produce clusters of arbitrary shape with noisy backgrounds that include density gradients. Also, it does not require fine tuning of its parameters and is more robust than DBSCAN. This clustering technique succeeded in finding dense clusters and recognizing noise points. However, there are some dense areas that should be a cluster but are marked as noise.

OPTICSxi

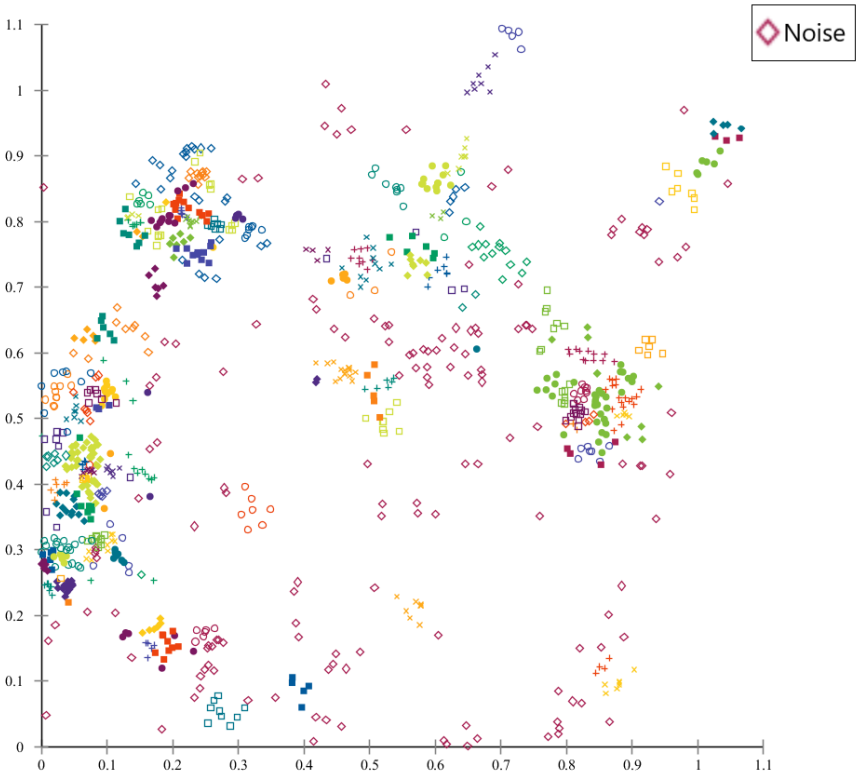


Figure 57. OPTICSXi clustering result

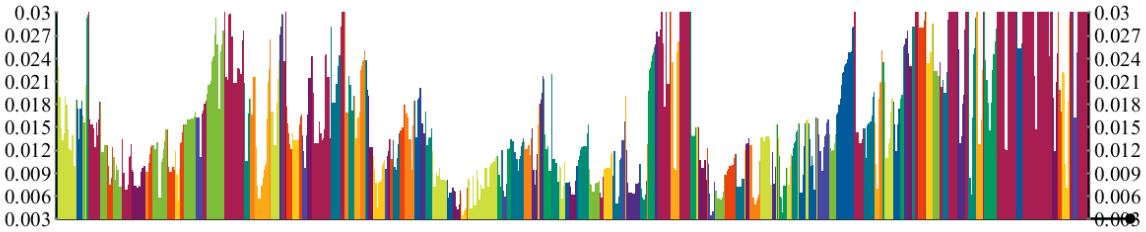


Figure 58. OPTICS reachability plot

OPTICS is a density-based algorithm that tries to overcome the problems of DBSCAN with varying density clusters and with the high sensitivity to the choice of parameter values. The OPTICS algorithm does not produce a strict cluster partition, but an augmented ordering of the dataset. To produce the cluster partition, I used OPTICSxi, which is an algorithm that produces a classification based on the output of OPTICS. OPTICSxi has three input parameters, *xi*, *ε* and *MinPts*. Unlike DBSCAN, the *ε* parameter is meant as a "maximum" distance to consider instead of a specific distance (a range of distances is considered in the OPTICS algorithm, up to *ε*). A large *ε* can be chosen, however this will increase the time until the convergence of the algorithm. The *MinPts* parameter has the same meaning with DBSCAN (i.e. the minimum number of data points required to form a cluster). The *xi* parameter is the contrast parameter that established the relative decrease in density. This parameter directly controls the number of clusters that will be obtained. In OPTICS the ε parameter becomes less important but still needs to be given a value. The idea of varying densities is implemented by having a range of *ε* parameters. By setting the *xi* parameter we set the density variation we accept to consider a group as a cluster.

I run the algorithm on my dataset using the ELKI data mining software with parameters *ε* = 0,03, *MinPts* = 4 and *xi* = 0,01 and it took 52ms time to cluster the dataset. Because the algorithm focuses on density variation, instead of a global value of density (like DBSCAN), it is possible for areas that have a very low density to become a cluster, just because they have a density variation from their surrounds that is higher than the given threshold. Correspondingly, it is possible to have dense areas that are not detected as clusters because there is a smooth density variation from their surroundings. From the results of OPTICSxi on my dataset it can be seen that it successfully found some dense

clusrers. However, the aforementioned problems occurred as there are areas with low density that should not be a cluster and also there are some dense areas that were not detected as clusters.

3.3.6 Similarity parameter

In my comparison of different clustering techniques, I used the distance between the data points as the similarity parameter. However, in a real-world scenario it would be more appropriate to add more attributes in the similarity parameter. In the paper "*Selection of UE-based Virtual Small Cell Base Stations using Affinity Propagation Clustering*" [59] the strength of the received signal between the UE-VBSs and the UEs is taken as the similarity parameter. Each UE is associated with the UE-VBS from which it receives the signal with the maximum power. The detailed formula of the received signal strength can be seen in the figure below.

$$P_{ij}^{r} = P_{j}^{t} - PL(d_{ij})$$
$$PL(d_{ij}) = \alpha + \beta 10log_{10}(d_{ij}) + \sigma$$

Figure 59. Equation of the received power (dBm) as a similarity parameter [59]

Where P_j^t is the transmit power of the j_{th} UE-VBS, P_{ij}^r is the received power at the i_{th}UE from the j_{th} UE-VBS, d_{ij} is the distance from the j_{th} UE-VBS, σ is the noise and α, β have channel dependent values.

3.3.7 Intra-cluster communication

For the intra-cluster communications (i.e. the communications between the active UE-VBS and its associated UEs) a separate spectrum can be utilized. Direct D2D communication is the communication among UEs that does not involve a BS in their communication path. By using a different communication technology for the intra-cluster

communications, the cellular network spectrum can be released for the UE-VBS to BS and for the outlier UEs to BS connections.

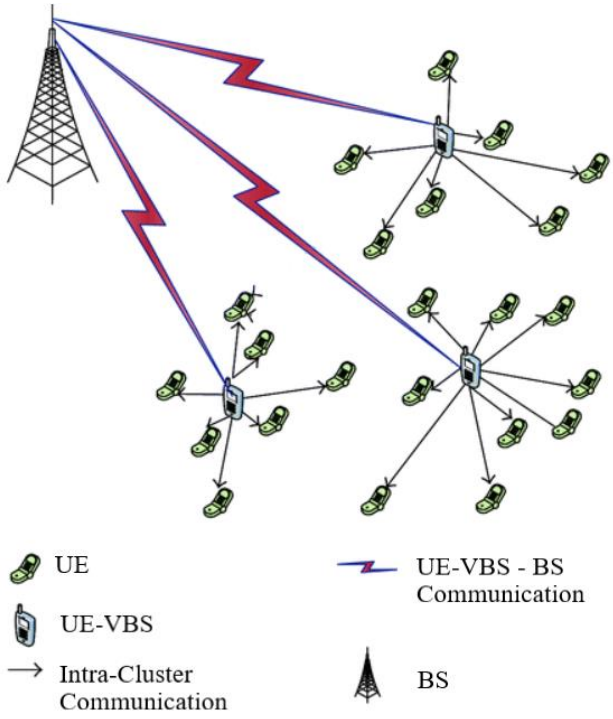


Figure 60. Network comprised of Clusters of UE-VBS and UEs

The most generally accepted contestants for D2D communications are Bluetooth and Wi-Fi that operate in the 2.4 GHz unlicensed spectrum [60]. However, the deployment of Wi-Fi and Bluetooth D2D communications can be unmanageable and cannot guarantee acceptable QoS. Thus WiFi-Direct and LTE-Direct technologies were introduced.

Wi-Fi-Direct

Wi-Fi-Direct enables D2D communication without requiring Wi-Fi infrastructure and with negligible client interaction. It allows a device to act as a network controller and allows multiple devices to exchange data between them [60]. Wi-Fi Direct can reach about ~20Mbps at less than 1m distance between the UEs, and decreases to 0 at ~41m distance [61]. WiFi-Direct allows efficient D2D connections in unlicensed bands and unlike LTE, it is an uncomplicated protocol, thus it has less energy consumption. Moreover, it enables higher spatial reuse than LTE, because it works in shorter distances. Nowadays most mobile devices have multi-radio capabilities, so they can simultaneously

use both WiFi and LTE [49]. Finally, WiFi-Direct permits a UE to have the role of either a client or an access point (AP).

LTE Direct

LTE Direct is a D2D technology that uses licensed LTE spectrum. The LTE Direct protocol was introduced in 3GPP Release 12 specification. In LTE Direct, the network functionalities like D2D connection setup, resource allocation and power control are implement at the eNodeB [61]. Nevertheless, when the link between two UEs is established, they can communicate without any eNodeB involvement. LTE Direct has the ability to maintain privacy, battery efficiency, better throughput and less spectrum utilization. Moreover, it can enable D2D location-based applications and services.

Scientific simulations showed that both the utilization of WiFi-Direct and LTE-Direct for aggregating the data in a single UE through intra-cluster communications outperform the standard LTE uploading where each UE transmits its own data to the eNodeB [61]. More precisely, the simulations showed that LTE-Direct can achieve higher energy efficiency when the number of UEs is relatively high and WiFi-Direct can achieve higher energy efficiency when the amount of data is small.

Chapter 4

Conclusion

4.1 Conclusions

This thesis initially made an introduction to the anticipated fifth generation of mobile networks and its ambitious goals, emerging technologies and challenges. Then, the concept of small cells, their benefits and challenges were discussed. It was highlighted that the dense deployment of a massive number small cells will be one of the most promising technologies towards realizing the target specifications of 5G. As the conventional deployment of small cells is static, it is unable to cope with the mobile traffic dynamics. Thus, a different approach of small cells was investigated where a subset of User Equipment (UE) is dynamically selected to serve as the base stations of other users. These UEs are referred as UE-based Virtual Small Cell Base Stations (UE-VBSs) and they can be used in a targeted manner to effectively relieve traffic in hot spot areas, increase coverage, and spectral efficiency. UE-VBSs can remove the constraint of the static deployment of existing Small Cell technologies and bring renaissance to wireless communication networks as a major technological breakthrough.

One of the goals of this thesis was to evaluate the performance of the UE-VBSs technology. To achieve this, I used OPNET modeler to analyze and compare the performance of a network that utilizes the UE-VBSs technology against a conventional network where all connections are directly between the UEs and the BS. My simulations confirmed that by enabling the use of a subset of the UEs as VBSs the performance of the network is significantly enhanced.

The second goal of this thesis was to investigate different clustering techniques in order to find the most suitable algorithm for clustering an ultra-dense network which utilizes

UE-VBSs. Towards this goal, I studied and presented the background of clustering theory and overviewed some of the most popular clustering algorithms. Afterwards, I evaluated and compared the performance of seven well-known and generally used clustering algorithms: K-Means, Mean-shift, Affinity propagation, Agglomerative clustering, DBSCAN, HDBSCAN and Spectral Clustering. For the comparison I used a dataset with 1000 points distributed with Poisson Cluster Process (PCP). Additionally, I benchmarked the performance and scaling of the algorithms using large datasets.

With these requirements of my particular application in mind and the score of each algorithm in terms of their results, required inputs, complexity, time and scalability the two algorithms that stood out are DBSCAN and HDBSCAN. Therefore, I decided to explore more density-based clustering algorithms. Density-based clustering algorithms are efficient, so they can scale to big datasets. Moreover, they can find arbitrary shaped clusters and they do not require the number of clusters to be specified beforehand. The two clustering algorithms that I tested are OPTICS and LSDBC. Both algorithms produced satisfactory clustering results. Thus, along with DBSCAN and HDBSCAN, they are the most suitable algorithms for clustering a network that uses UE-VBSs, while each one has some strengths and some weaknesses over the others.

The paper *"Selection of UE-based Virtual Small Cell Base Stations using Affinity Propagation Clustering"* proposes the dynamic clustering of virtual small cells and the activation of UE-VBSs using a modified version of the affinity propagation clustering technique. The UEs measure the RSS (Received Signal Strength) from every eligible UE-VBS in their proximity. Each UE will connect with the UE-VBS from which it receives the maximum RSS. In contrast with the original Affinity Propagation algorithm which passes messages between all points, the modified algorithm passes messages only between the UEs and the eligible UE-VBSs. Nevertheless, Affinity Propagation, as proved in my study, tends to be very slow, especially on big data sets because its basic operations are computationally expensive.

Concluding, the most suitable technique for clustering an ultra-dense network which utilizes UE-VBSs is a two-step method. In the first step a density-based clustering algorithm will be used on the initial big dataset to efficiently extract the dense areas of

UEs and detect outlier UEs. In the second step, the modified affinity propagation algorithm will be used in each dense area provided from the previous step to choose the active UE-VBSs and connect every UE with its associated UE-VBS.

4.2 Future Work

As a future work I propose the further study of the four density-based (DBSCAN, HDBSCAN, LSDBC and OPTICS) on bigger datasets with PCP distribution. Also, it would be interesting to see how these algorithms behave with the Received Signal Strength as a similarity parameter instead of the distance. In addition, the aforementioned two-step clustering method can be tested in a real-world scenario. Finally, the OPNET simulations can be extended for scenarios with more UEs and UEs with mobility.

References

[1] A. Gupta and R. K. Jha, “A Survey of 5G Network: Architecture and Emerging Technologies,” *IEEE Access*, vol. 3, pp. 1206–1232, 2015.

[2] Ericsson, “More Than 50 Billion Connected Devices,” *White Pap.*, no. February, pp. 1–12, 2011.

[3] Qualcomm, “The 1000x Mobile Data Challenge,” *White Pap.*, no. November, pp. 1–38, 2013.

[4] N. Heuveldop, “Ericsson mobility report,” 2017.

[5] P. K. Agyapong, M. Iwamura, D. Staehle, W. Kiess, and A. Benjebbour, “Design considerations for a 5G network architecture,” *IEEE Commun. Mag.*, vol. 52, no. 11, pp. 65–75, 2014.

[6] “GSMA Mobile Economy 2017,” *Mobile Economy 2017*.

[7] Huawei, “5G : A Technology Vision,” *Huawei, White paer*, pp. 1–16, 2014.

[8] I. F. Akyildiz, S. Nie, S.-C. Lin, and M. Chandrasekaran, “5G roadmap: 10 key enabling technologies,” *Comput. Networks*, vol. 106, pp. 17–48, Aug. 2016.

[9] T. S. Rappaport *et al.*, “Millimeter Wave Mobile Communications for 5G Cellular: It Will Work!,” *IEEE Access*, vol. 1, pp. 335–349, 2013.

[10] N. Panwar, S. Sharma, and A. K. Singh, “A survey on 5G: The next generation of mobile communication,” *Phys. Commun.*, vol. 18, Part 2, pp. 64–84, Aug. 2016.

[11] V. Chandrasekhar, J. Andrews, and A. Gatherer, “Femtocell networks: a survey,” *IEEE Commun. Mag.*, vol. 46, no. 9, pp. 59–67, Aug. 2008.

[12] A. Osseiran *et al.*, “Scenarios for 5G mobile and wireless communications: the vision of the METIS project,” *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 26–35, 2014.

[13] “5G radio network architecture,” *RAS Clust.*, pp. 1–18, 2014.

[14] J. Rodriguez, “Fundamentals of 5G Mobile Networks.” 2015.

[15] J. Eichinger *et al.*, “Building a new multi-facial Architecture of 5G,” *IEEE COMSOC MMTC Lett.*, no. x, pp. 3–6, 2014.

[16] M. R. Palattella *et al.*, “Internet of Things in the 5G Era: Enablers, Architecture, and Business Models,” *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[17] L. Gavrilovska, V. Rakovic, and V. Atanasovski, “Visions Towards 5G: Technical Requirements and Potential Enablers,” *Wirel. Pers. Commun.*, vol. 87, no. 3, pp. 731–757, Apr. 2016.

[18] “5G Massive MIMO Testbed: From Theory to Reality - National Instruments.” [Online]. Available: http://www.ni.com/white-paper/52382/en/. [Accessed: 24-Aug-2017].

[19] M. Tehrani, M. Uysal, and H. Yanikomeroglu, “Device-to-device communication in 5G cellular networks: Challenges, solutions, and future directions,” *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 86–92, May 2014.

[20] G. Fodor *et al.*, “Design aspects of network assisted device-to-device communications,” *IEEE Commun. Mag.*, vol. 50, no. 3, pp. 170–177, Mar. 2012.

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

[22] X. Ge, S. Tu, G. Mao, C. X. Wang, and T. Han, “5G Ultra-Dense Cellular Networks,” *IEEE Wirel. Commun.*, vol. 23, no. 1, pp. 72–79, Feb. 2016.

[23] Y. Zhang and M. Chen, *Cloud based 5G wireless networks*. Springer, 2016.

[24] N. Saquib, E. Hossain, L. B. Le, and D. I. Kim, “Interference management in OFDMA femtocell networks: Issues and approaches,” *IEEE Wirel. Commun.*, vol. 19, no. 3, pp. 86–95, Jun. 2012.

[25] S. C. Forum, “Small cells – what’s the big idea?,” 2012.

[26] O. Galinina, A. Pyattaev, S. Andreev, M. Dohler, and Y. Koucheryavy, “5G multi-RAT LTE-WiFi ultra-dense small cells: Performance dynamics, architecture, and trends,” *IEEE J. Sel. Areas Commun.*, vol. 33, no. 6, pp. 1224–1240, 2015.

[27] D. López-Pérez, M. Ding, H. Claussen, and A. H. Jafari, “Towards 1 Gbps/UE in Cellular Systems: Understanding Ultra-Dense Small Cell Deployments,” *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2078–2101, 2015.

[28] X. Ge, H. Cheng, M. Guizani, and T. Han, “5G wireless backhaul networks: Challenges and research advances,” *IEEE Netw.*, vol. 28, no. 6, pp. 6–11, Nov. 2014.

[29] U. Siddique, H. Tabassum, E. Hossain, and D. I. Kim, “Wireless backhauling of 5G small cells: Challenges and solution approaches,” *IEEE Wirel. Commun.*, vol. 22, no. 5, pp. 22–31, Oct. 2015.

[30] C. Christophorou, A. Pitsillides, and I. Akyildiz, “CelEc framework for reconfigurable small cells as part of 5G ultra-dense networks,” in *IEEE International Conference on Communications*, 2017, pp. 1–7.

[31] A. Behnad and X. Wang, “Virtual Small Cells Formation in 5G Networks,” *IEEE Commun. Lett.*, vol. 21, no. 3, pp. 616–619, Mar. 2017.

[32] A. Asadi and V. Mancuso, “Network-Assisted Outband D2D-Clustering in 5G Cellular Networks: Theory and Practice,” *IEEE Trans. Mob. Comput.*, vol. 16, no. 8, pp. 2246–2259, Aug. 2017.

[33] M. D. Boomija, “COMPARISON OF PARTITION BASED CLUSTERING ALGORITHMS,” *J. Comput. Appl.*, vol. 1, no. 4, pp. 18–21, 2008.

[34] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of massive datasets: Second edition*. 2014.

[35] R. M. Mccutchen and S. Khuller, “Streaming Algorithms for k-Center Clustering with Outliers and with Anonymity.”

[36] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.

[37] G. Sehgal and K. Garg, “Comparison of Various Clustering Algorithms,” *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 3, pp. 3074–3076, 2014.

[38] K. Wang, J. Zhang, D. Li, X. Zhang, and T. Guo, “Adaptive Affinity Propagation Clustering,” *Automatica*, vol. 33, no. 12, pp. 1242–1246, 2007.

[39] B. J. Frey and D. Dueck, “Clustering by Passing Messages Between Data Points.”

[40] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” *Proc. 2nd Int. Conf. Knowl. Discov. Data Min.*, pp. 226–231, 1996.

[41] T. N. Tran, K. Drab, and M. Daszykowski, “Revised DBSCAN algorithm to cluster data with dense adjacent clusters,” *Chemom. Intell. Lab. Syst.*, vol. 120,

pp. 92–96, Jan. 2013.

[42] R. J. G. B. Campello, D. Moulavi, A. Zimek, and J. Sander, “Hierarchical Density Estimates for Data Clustering, Visualization, and Outlier Detection,” *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 1, pp. 1–51, Jul. 2015.

[43] V. Kumar, P. Tan, M. Steinbach, and V. Kumar, *Cluster analysis: basic concepts and algorithms*. 2006.

[44] I. Davidson and S. S. Ravi, “Agglomerative Hierarchical Clustering with Constraints: Theoretical and Empirical Results,” Springer, Berlin, Heidelberg, 2005, pp. 59–70.

[45] K.-C. Wong, “A Short Survey on Data Clustering Algorithms,” *2015 Second Int. Conf. Soft Comput. Mach. Intell.*, pp. 64–68, Nov. 2015.

[46] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J Mach Learn Res*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[47] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, “Optics: Ordering points to identify the clustering structure,” *ACM Sigmod Rec.*, pp. 49–60, 1999.

[48] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*. 2013.

[49] A. Pyattaev, K. Johnsson, S. Andreev, and Y. Koucheryavy, “3GPP LTE traffic offloading onto WiFi Direct,” in *2013 IEEE Wireless Communications and Networking Conference Workshops, WCNCW 2013*, 2013, pp. 135–140.

[50] G. H. Shah, C. K. Bhensdadia, and A. P. Ganatra, “An Empirical Evaluation of Density-Based Clustering Techniques,” *Int. J. Soft Comput. Eng.*, no. 1, pp. 216–223, 2012.

[51] E. Biçici and D. Yuret, “Locally scaled density based clustering,” *Adapt. Nat. Comput. Algorithms*, pp. 739–748, 2007.

[52] M. Afshang, H. S. Dhillon, and P. H. J. Chong, “k-Closest coverage probability and area spectral efficiency in clustered D2D networks,” in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–6.

[53] C. Saha, M. Afshang, and H. S. Dhillon, “3GPP-inspired HetNet Model using Poisson Cluster Process: Sum-product Functionals and Downlink Coverage,” *IEEE Transactions on Communications*, no. 1, 2017.

[54] S. Firdaus and A. Uddin, “A Survey on Clustering Algorithms and Complexity Analysis,” *IJCSI Int. J. Comput. Sci. Issues*, vol. 12, no. 2, pp. 62–85, 2015.

[55] D. Arthur and S. Vassilvitskii, “How slow is the k - means method?,” in *Proceedings of the twenty-second annual symposium on Computational geometry - SCG ’06*, 2006, p. 144.

[56] L. McInnes, J. Healy, and S. Astels, “hdbscan: Hierarchical density based clustering,” *J. Open Source Softw.*, vol. 2, no. 11, 2017.

[57] E. Schubert, A. Koos, T. Emrich, A. Züfle, and A. Z. Klaus Arthur Schmid, “[ELKI 0.7] A Framework for Clustering Uncertain Data,” *Proc. VLDB Endow.*, vol. 8, no. 12, pp. 1976–1989, 2015.

[58] H. P. Kriegel, P. Kröger, J. Sander, and A. Zimek, “Density-based clustering,” *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 1, no. 3, pp. 231–240, May 2011.

[59] P. Swain, C. Christophorou, U. Bhattacharjee, C. Silva, and A. Pitsillides, *Selection of UE-based Virtual Small Cell Base Stations using Affinity Propagation Clustering*. 2018.

[60] A. Pyattaev *et al.*, “3GPP LTE-assisted Wi-Fi-direct: Trial implementation of live D2D technology,” *ETRI J.*, vol. 37, no. 5, pp. 877–887, Oct. 2015.

[61] M. Condoluci, L. Militano, A. Orsino, J. Alonso-Zarate, and G. Araniti, “LTE-direct vs. WiFi-direct for machine-type communications over LTE-A systems,” in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, 2015, vol. 2015–Decem, pp. 2298–2302.

Appendix A - Abbreviations

3GPP	Third-Generation Partnership Project
DeNB	Donor eNB
AMPS	Advanced Mobile Phone System
BBU	Bandwidth-Based Unit
BS	Base Station
CaPex	Capital Expenditure
CGS	Closed Subscriber Group
C-RAN	Cloud Radio Access Network
D2D	Device to Device
DNS	Domain Name System
DoS	Denial of Service
E2E	End-to-End
EDGE	Enhanced Data Rates for GSM Evolution
eNB	evolved NodeB.
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HSPA	High Speed Packet Access
IoT	Internet of Things
LTE	Long Term Evolution.
MBS	Macrocell Base Station
mmWave	millimeter-Wave,
MTC	Machine-Type Communication
NFV	Network Function Virtualization
NMT	Nordic Mobile Telephone
OFDMA	Orthogonal Frequency Division Multiple Access
OpEx	Operation Expenditure
PCP	Poisson Cluster Process
PDCCH	Physical Downlink Control Channel
PDSCH	Physical Downlink Shared Channel
PUSCH	Physical Uplink Shared Channel
QoE	Quality of Experience

QoS	Quality of Service
RAN	Radio Access Network
RRU	Remote Radio Unit
RSRP	Reference Signal Received Power
RSRQ	Reference Signal Received Quality
SDN	Software Defined Network
SNR	Signal to Noise Ratio
TACS	Total Access Communications System
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle