Efficient Backhaul Design for 5G Ultra-Dense Cells

by

Mital Kishorbhai Raithatha

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs

in partial fulfillment of the requirements for the degree of

Master of Applied Science

in

Ottawa-Carleton Institute for Electrical and Computer Engineering

Carleton University

Ottawa, Ontario

Abstract

In 5G Ultra-Dense Cells, a distributed wireless backhaul is an attractive solution for forwarding traffic to the core. The macro-cell coverage area is divided into many small cells. A few of these cells are designated as gateways, which are linked to the core by high capacity fiber optic links. Each small cell is associated with one gateway and all small cells forward their traffic to their respective gateway through multi-hop mesh networks. In this thesis, we investigate the Gateway Location Problem and show that finding near optimal gateway locations improves the Backhaul Network Capacity (BNC). Toward this end, the *p*-median problem has been formulated as Integer Linear Program to find optimal gateway locations. Subsequently, we use artificial intelligence based on a Genetic Algorithm (GA) in combination with machine learning based on the *K*-means clustering algorithm and develop a heuristic to find near-optimal gateway locations that maximize the BNC. We evaluate the performance of our new heuristic, K-GA, in comparison with six different approaches in terms of Average Number of Hops (ANH) and BNC at different node densities through extensive Monte Carlo simulations. All approaches including the optimal (or exact) approach are tested under different small cell distribution scenarios, namely, Uniform distribution, bivariate Gaussian distribution, and Cluster distribution. K-GA provides near optimal results achieving ANH and BNC within *3*% of optimal and saves on average *95*% of execution time. The scheme is practical and can be easily adapted to the spatial distribution of traffic. We also analyze the effect of the number of gateways on ANH and BNC. The results show that more gateways are beneficial.

To my beloved parents: Sarla and Kishor Raithatha

and my dearest husband, Mohit A. Vadera

Acknowledgement

First of all, I would like to sincerely thanks my supervisor Professor Roshdy Hafez for his critically important role in coming up with this novel thesis work. He helped to bring a deeper clarity about the problem statement of the research work. He set a high bar of quality and novelty of my research work. Without his kind and perceptive supervision, this research work would have never acquired its current form.

I must thank Dr. Aizaz Chaudhry for his valuable feedback and suggestions. I am grateful for the time and patience that he donated towards the enhancement of my research methodology and work ethics. I am also much obliged for his many iterative reviews of my research work.

I am also indebted to Professor John Chinneck for his invaluable insight and suggestions which improve my concepts about optimization algorithms and software. He also pointed me in the right direction to investigate the run time complexity of my proposed algorithm.

A heartfelt thanks to my wonderful family, my parents Sarla Raithatha, Kishor Raithatha, and my in-laws Jyoti Vadera, Ashok Vadera, for their unconditional love and motivation to achieve this milestone. I also appreciate my husband, Mohit Vadera, for his understanding nature and encouragement during my busy research schedule. My research journey would not have been possible without tireless support from my entire family.

TABLE OF CONTENTS

List of Algorithms

List of Figures

List of Tables

List of Acronyms and Abbreviations

3GPP – Third Generation Partnership Project

5G – Fifth Generation

AI – Artificial Intelligence

ANH – Average Number of Hops

BNC – Backhaul Network Capacity

CD – Cluster Distribution

CI – Confidence Interval

eMBB – Enhanced-Mobile Broadband services

FTTC – Fiber to the Curb

FHD – Full-High Definition

FSO – Free Space Optics

GA – Genetic Algorithm

GD – bivariate Gaussian Distribution

GLP – Gateway Location Problem

GW – Gateway

IAB – Integrated Access and Backhaul

IoT – Internet-of-Things

ILP – Integer Linear Programming

ITU – International Telecommunication Union

K-GA – *K*-means and Genetic Algorithm

KM-GA – *K*-medoids and Genetic Algorithm

LTE – Long-term Evolution

MBS – Macro-cell base station

MIMO – Multiple-Input Multiple-Output

ML – Machine Learning

mMTC – massive Machine Type Communications

Mm-wave – Millimeter Wave

OPL – Optimization Programming Language

PAM – Partitioning Around Medoids

PPP – Poisson Point Process

QoE – Quality of Experience

QoS – Quality of Service

SNR – Signal to Noise Ratio

UD – Uniform Distribution

UDC – Ultra-dense Cell

UDN – Ultra-dense Network

UE – User Equipment

UHD – Ultra-high definition

uRLL – ultra-Reliable and Low Latency services

V2X – Vehicle to everything

List of Symbols

B – Available Bandwidth

S_P – Signal Power

N_P – Noise Power

C – Maximum Capacity given in bits per second

n_{pop} – Population size or number of Chromosomes

G_{max} – Maximum number of Generations

N – Number of Nodes or Small cells

G – Number of Generations

P_m – Mutation Probability

P_c – Crossover Probability

M – Number of Gateways

I – Number of Iterations

R_{max} – Maximum number of Replications

V – Set of nodes in the graph

E – Set of edges in the graph

W_S – Small cell link capacity

W_G – Gateway link capacity

d_{ij} – Distance between small cell i and gateway j

p – Number of medians (gateways) to locate

t – Number of small cells closest to each K-means centroid

t^M – Number of chromosomes generated from K-means output

p_i – Fitness Probability

q_i – Cumulative Probability

f_i – Fitness Value

r – Random number

List of Publications:

Mital Raithatha, Aizaz U. Chaudhry, Roshdy H.M. Hafez, and John W. Chinneck, "Locating Gateways for Maximizing Backhaul Network Capacity of 5G Ultra-Dense Networks," in *proc. 2020 Wireless Telecommunications Symposium (WTS)*, Washington, DC, USA, April 2020.

Aizaz U. Chaudhry, Mital Raithatha, Roshdy H.M. Hafez, and John W. Chinneck, "Using Machine Learning to locate Gateways in the wireless Backhaul of 5G Ultra-Dense Networks," accepted for publication in *International Symposium on Networks, Computers and Communications (ISNCC) Workshop Machine Learning for Next generation Systems and Networks (MLNGSN)*, Montreal, QC, October 2020.

Mital Raithatha, Aizaz U. Chaudhry, Roshdy H.M. Hafez, and John W. Chinneck, "AI/ML-Enabled Gateways Location for 5G Ultra-Dense Networks," *to be submitted to a Journal.*

Chapter 1: INTRODUCTION

1.1. Overview

There is rapid growth in the volume of mobile data traffic due to massive deployment of broadband wireless networks and an explosive increase in the use of wireless devices such as smartphones, laptops, tablets, and wearable accessories. The recent introduction of 5G networks around the world induced the development and deployment of many wireless services such as: ultra-high definition video streaming, augmented reality, sophisticated on-line video gaming, security applications, intelligent farming, and connected vehicles. 5G has three stated objectives: (1) support for Enhanced-Mobile Broadband services (eMBB), (2) support for ultra-Reliable and Low Latency services (uRLL), and (3) support for massive Machine Type Communications (mMTC). This thesis falls in the domain of the first stated objective where 5G networks aim to increase the data rate by as much as two orders of magnitude [1].

Expanding a wireless network's capacity by two orders of magnitude is an ambitious target, but recent extensive research and development efforts put this target within reach. In the quest for much faster data transmission, 5G networks are deploying several technologies to improve network capacity and spectrum efficiency. One of these technologies is *massive MIMO* (multiple-input multiple-output), which helps improve the channel capacity, channel efficiency and signal strength by employing multiple antennas for transmission and reception [2]. Another important technology is the *millimeter wave (mm-wave) band*. 5G is set to exploit the massive spectrum space available at higher frequencies. The mm-wave frequencies are expected to provide hundreds of megahertz of bandwidth to meet the requirements of higher data rates [3]. In addition to massive MIMO and mm-wave, *network densification* is the third and most promising approach to handle higher spectrum demands in crowded venues. Densification of crowded cells increases network capacity in terms of bits/sec/Hz/unit area. The basic approach is to make the network as dense as possible by deploying a large number of access nodes within a macro-cell coverage area. These access nodes are referred to as "small cells". Cell densification improves link quality and significantly increases network capacity [4]. The combination of wider RF bandwidth, Massive MIMO, and deployment of large number of small cells gives rise to what is now

known as an *Ultra-Dense Network* (UDN) or an Ultra-Dense Cell (UDC) [5], in which cells have a very high data rate per unit area. In this thesis, we use the terms ultra-dense cells and ultra-dense networks interchangeably.

1.2. Motivation

While cell densification using a UDN increases the capacity in terms of bits/sec/Hz/unit area, it complicates the backhauling problem. The term "backhaul" refers to the links between access points and the core network. An obvious solution to the backhaul problem is to directly connect each small cell to the core using fiber optic or any broadband cabling, but this is costly and cumbersome. On the other hand, a wireless backhaul solution is flexible and cost-effective. In particular, the use of mm-wave bands provides the spectrum resources needed to interconnect small cells to gateways [6]. The idea of using wireless links to facilitate backhauling received special attention in 3GPP Release 16 [7]. One of the key novelties of 3GPP Release 16 is to integrate access and backhaul using small cells called Integrated Access and Backhaul (IAB) nodes. An IAB node is a miniature base station that communicates with UEs on a given frequency and delivers user's traffic to the core network on a different frequency as part of the backhaul network. This technology has gained lots of attention in the industry because IAB is considered to be a cost efficient and convenient backhaul solution [8]. This thesis explores the use of IAB nodes (or small cells) in multi-hop wireless networks that carry the access traffic to the core. We use the term "distributed wireless backhaul" to imply that N small cells are clustered into groups, each group is associated with a gateway (GW) by means of a multi-hop mesh network, and all connections among small cells (including gateways) use mm-wave links. Our network model represents the standard 3GPP Release 16 IAB architecture, where an IAB node is referred to as small cell and IAB donor is called gateway. IAB nodes (small cells) and donors (gateways) form a multi-hop backhaul architecture. A proper UDN design must use the radio spectrum efficiently and reduce interference among the small cells while managing user mobility and changes in the spatial distribution of the traffic density. Finding the optimum (or near optimum) locations of the fixed gateways is an important design step, which directly affects the *backhaul network capacity* (BNC). However, we must guard against the possibility of creating bottle necks in the multi-hop networks leading up to the gateways. A bottleneck

happens when there are too many hops that cause the aggregated traffic to exceed the small cell-to- small cell link capacity. This could happen if the number of small cells is too large and the number of gateways is not large enough. Therefore, we should also find the minimum number of gateways that avoids congestion. Since the number of gateways should be small anyway, we simply run the simulation for different number of gateways and compare the results. In other words, in this work the number of gateways is one of the simulation parameters.

1.3. Objective of Thesis

In a UDN (or a UDC), the total required access capacity is large and has spatial distribution that matches the demographic distribution of users within the coverage area. Network designers would select the number and locations of small cells to meet the access demand. The main aim of this thesis is to select the best gateway locations that simplify and optimize the distributed wireless backhaul of a 5G ultra-dense network. It allows us to solve the backhaul problem more efficiently and help us to improve the backhaul network capacity. The key idea is that the access traffic will dictate the number and locations of small cells (call it N). The work of this thesis starts by selecting M out of the N small cells as gateways. The GWs will be connected to the core using fiber optics. The remaining (N-M) small cells are grouped into clusters, one cluster for each gateway. Each cluster of small cells forms one multi-hop tree connected by mm-wave wireless links. Therefore, the totality of the backhaul network is made up of M trees feeding the access traffic to the core network. The backhaul network capacity of this wireless backhaul depends on the *average number of hops* (ANH) between small cells and GWs [9]. To maximize BNC, ANH must be minimized. The *gateway location problem* (GLP) studied in this thesis involves finding gateway locations (M out of N small cells) such that BNC is maximized. This involves finding the smallest ANH over every possible combination of M out of N selections; this is a combinatorically explosive problem. For example, if $N = 400$ and $M = 4$, then there are more than a billion possible solutions.

The GLP is similar to the p-median problem, which is known to be NP-Hard [10]. In this context, the p-median problem has been formulated as an Integer Linear Programming (ILP) to find optimal gateway locations. To find the optimal results,

mathematical implementation searches through the entire solution space over every possible solution. This makes the computational complexity and time complexity of optimal (or exact) approach excessively large, and as the size of the problem instance increases, it rapidly becomes too large and complex to solve. For this reason, we explore heuristic methods based on Artificial Intelligence (AI) and Machine Learning (ML) to find GW locations and to associate small cells to them. Genetic Algorithms (GAs) are a type of Evolutionary Algorithm inspired by biological evolution [11]. The well-known *K*-means and *K*-medoids clustering are unsupervised machine learning algorithms, which are simple partition-based algorithms where *k* clusters create *k* centroids or *k* medoids [12][13]. We developed a new algorithm by combining ML's *K*-means clustering Algorithm with AI's stochastic optimization genetic algorithm. The resulting algorithm is called *K-GA*. Dijkstra's shortest path algorithm [14] is used to calculate the average number hops by associating small cells to GWs. Different network topologies for different scenarios such as Uniform Distribution (UD), bivariate Gaussian Distribution (GD) and Cluster Distribution (CD) are investigated to assess the performance of proposed K-GA heuristic under different distribution scenarios. Extensive Monte Carlo based simulations have been conducted to evaluate the performance of the proposed K-GA algorithm. We compare its performance with *K*-means, GA, *K*-medoids, combination of *K*-medoids and GA, and a baseline approach in terms of the ANH and the BNC.

After exploring the GLP in an ultra-dense network, we also examine the effect of the number of gateways on the wireless backhaul network capacity.

1.4. Research Contribution

The main contribution of this thesis is summarized as follows:

We propose the new effective K-GA heuristic algorithm which quickly provides near optimal solution for finding the best locations of GWs. K-GA delivers results of ANH and BNC within *3*% of optimal in all scenarios and performs better compared to other heuristic approaches. The simulation results indicate that K-GA provides a near optimal solution for different spatial distributions of traffic density and works well in all network topologies. Using ML and AI together, K-GA offers a powerful, efficient, and attractive solution to GLP and achieves better backhaul network capacity.

1.5. Outline of Thesis

The rest of the thesis is organized as follows:

Chapter 2: This chapter presents the state of the art and background information for understanding the approach and content of the thesis. In its first section, the requirements for 5G wireless networks, and key technologies to satisfy 5G requirements is explained. Then we explain motivation for network densification, backhaul solutions and network architectures used for 5G UDNs. We further discuss gateway location problem as well as the related work done to solve the GLP in different networks. At the end, we provide several related backgrounds for techniques and heuristics on the *p*-median problem, Genetic Algorithms, the *K*-means clustering algorithm, *K*-medoids clustering algorithm and Dijkstra's shortest path algorithm.

Chapter 3: Explains the problem statement in the 5G ultra-dense network. The problem formulation in terms of BNC and ANH is described.

Chapter 4: Describes the network model of the distributed multi-hop wireless backhaul. The mathematical formulation of the GLP is given and our proposed K-GA heuristic is explained.

Chapter 5: Illustrates the detailed implementation of our proposed K-GA algorithm. It also explained procedure for other comparison approaches and different small cells deployment scenarios.

Chapter 6: Presents the performance evaluation and results for the gateway location problem. It also presents the results for the effect of the number of gateways on the performance of the backhaul.

Chapter 7: Concludes the paper with an outlook on future work.

Appendix A: Explains the effects of different parameters of the genetic algorithm on K-GA.

Chapter 2: BACKGROUND

This chapter provides a review of the essential concepts, background information and studies in 5G ultra-dense wireless networks and backhaul solutions. It briefly illustrates the *p*-median problem and explains an artificial intelligence heuristic as well as machine learning algorithms that relate to the scope of intelligent gateway locations in ultra-dense small cell distributed environments as per the hypothesis indicated in Chapter 1 of this thesis.

2.1. 5G Requirements

The evolution from 4G to 5G marks a significant expansion in the range and volume of wireless communications. Up until recently, wireless systems were mostly human-centric. 4G is known to offer broadband text-audio-visual communications. Crossing over to 5G, the range of applications is expanding significantly, from very low data rate systems designed to monitor and collect information over vast coverage areas to extremely high data rate applications such as tele-presence, ultra-high definition robotic-based systems and similar applications. Moreover, the huge popularity of smart phones is causing scenarios where large number of users in crowded venues can generate huge demands for wireless data far beyond what 4G systems can support. Many new use cases are emerging that require much higher data rate and other critical demands such as ultra-reliability and ultra-low latency.

It is anticipated [15] that the high demand of data rates will lead to improved technologies with better capabilities to handle diverse service needs. To meet requirements of high data rates, extensive research was carried out by major research groups around the world to push wireless technologies to new heights in terms of speed and reliability. Ultra-high definition (UHD) video streaming, video conferencing, augmented reality, virtual gaming, intelligent farming, and connected vehicles are examples of services that the 5G system should support [16]. The 5G services classified by International Telecommunication Union (ITU) can be summarized as follows:

- **Enhanced Mobile Broadband (eMBB):** eMBB aims to meet the high data rates demand of smart devices and concentrates on services that have high data consumption

and data transmission rate such as UHD, Full-high definition (FHD), and innovative user interfaces such as 3D cameras, pattern, and gesture recognition.

- **Ultra-reliable and Low-latency Communications (uRLL):** uRLL services support low-latency communication with very high reliability. It aims to meet the expectations of low latency and high reliable services such as mission critical systems, and automated driving.

- **Massive Machine-type Communications (mMTC):** mMTC supports huge number of Internet-of-Things (IoT) devices and services such as smart city and smart agriculture, which needs low power consumption and low data rates for very large numbers of connected devices.

The next era is predicted to be a fully connected society where data will be used for sharing and accessing the information anywhere at any time. To achieve the 5G vision, new technologies need to be explored, evaluated, and deployed. New wireless technologies combined with current technologies will bring revolution in wireless communication [17]. In order to define the next generation of wireless networks, 3GPP finalized first version of 5G standardization in Release 15. Recently, in Release 16 3GPP includes several enhancements and extensions of the first version of 5G technology which covers variety of topics in 5G systems. In Release 16 various subjects include: Multimedia Priority Service, 5G satellite access, Vehicle-to-everything (V2X) application layer services, terminal positioning and location, Local Area Network support in 5G, wireless and wireline convergence for 5G, communications in vertical domains, network automation and novel radio techniques. Further items being studied include security, codecs and streaming services, Local Area Network interworking, network slicing and the IoT [19]. In the next section, we will introduce design principles for 5G systems in order to meet 5G requirements.

2.2. 5G Key Technology Enablers

5G envisions data rates that are at least two order of magnitude larger than existing rates. The most fundamental concept in link capacity is Shannon's formula [20]:

$$C = B \log_2(1 + S_P / N_P), \qquad (2.1)$$

where C is the maximum capacity given in bits per second, B represents the available bandwidth, and S_P/N_p (SNR) is the signal to noise ratio of power levels. It can be seen from the equation that capacity increases by increasing the spectrum (B) or by increasing the signal power level (S_P) or by decreasing the noise power level (N_p). The capacity increases linearly with B but its increase with SNR is limited by the nature of the log(.) formula. 5G aims at increasing B significantly by developing and using part of the mm-wave band and to overcome the limitation of the Shannon formula, we need to exploit the "space dimension either using the MIMO/massive MIMO technology or by cell splitting or both". The method proposed for cell splitting is the densification of 4G macro-cells by deploying many small cells.

2.2.1. Massive MIMO:

Massive MIMO (Multiple-Input Multiple-Output) technology helps to improve the channel capacity, channel efficiency and signal strength by installing multiple antennas for transmission and reception [2]. Other technique using multiple antennas is beamforming. Various beamforming schemes help massive MIMO to achieve high mobility and spectral efficiency. Received power can be increased by increasing the directivity of antennas [18].

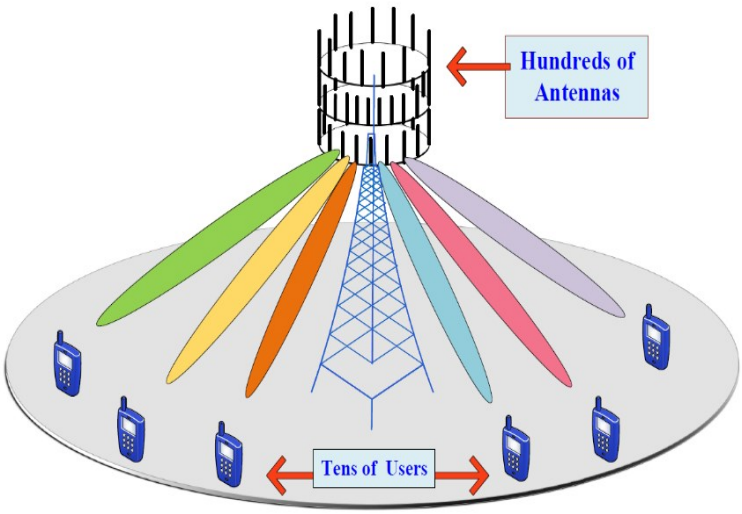

Fig 2.1. An example of massive MIMO concept [21]

Fig 2.1 describes the concept of massive MIMO. Few advantages of multiple antenna techniques are as follows [21]:

- High network capacity due to directional function of beam.
- Beamforming allows antenna to transmit at lower power resulting in lower usage of large power amplifiers, which helps to reduce hardware cost.

2.2.2. Millimeter Wave Communication:

Due to the rapid growth of connected wireless devices and new data hungry applications as discussed earlier, mobile data traffic is experiencing exponential increase, generating challenges for wireless service providers. To achieve higher capacity, more spectrum is required. Most current wireless networks, however, work with frequencies below 6 GHz. This part of the spectrum is crowded and is not enough for 5G.

An important technology to achieve higher capacity for 5G is the Millimeter wave (mm-wave). In order to achieve higher data rates, mm-wave carrier frequencies are expected to provide hundreds of megahertz of bandwidth [3]. Mm-wave uses the frequencies in the range of 3 to 300 GHz, as shown in Fig 2.2 [22]. Due to high propagation losses and sensitivity to physical blockage, the mm-wave is preferred in line-of-sight environments or backhaul links. To counter the high propagation loss at mm-wave frequencies, beamforming can be used in order to satisfy very large bandwidth requirements of 5G [18].

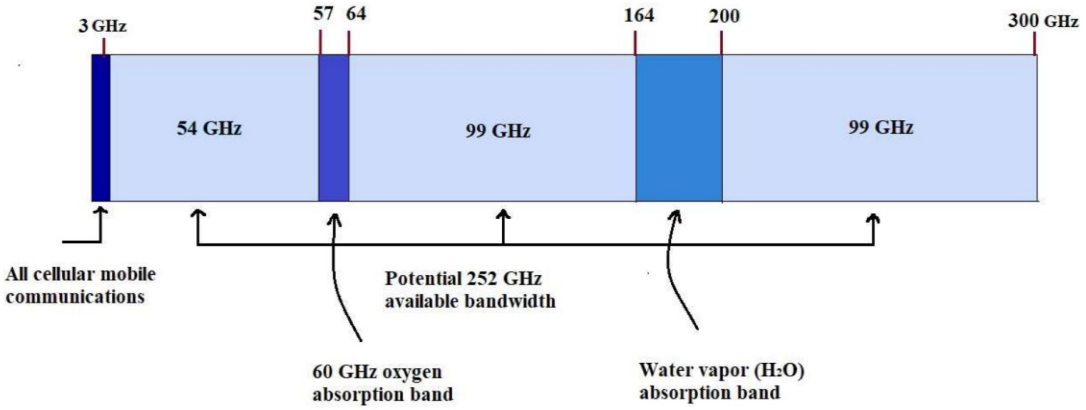

Fig 2.2. Millimeter-wave Spectrum [22]

2.2.3. Ultra-dense Network:

In addition to the requirements of higher capacity and spectral efficiency, an ultra-dense network (or an ultra-dense cell) is the most promising idea to handle the large amount of data traffic. In [5], a UDN is defined as a 5G small cell network where the number of small cells is greater than the number of UEs. The basic idea is to make the network (or macro-cell) as dense as possible by deploying a high number of access nodes near to each other. These access node, referred to as small cells, have low transmission power and small coverage. A UDN benefits by improving link quality and increasing network capacity [5]. An illustration of small cells deployment is presented in Fig 2.3.

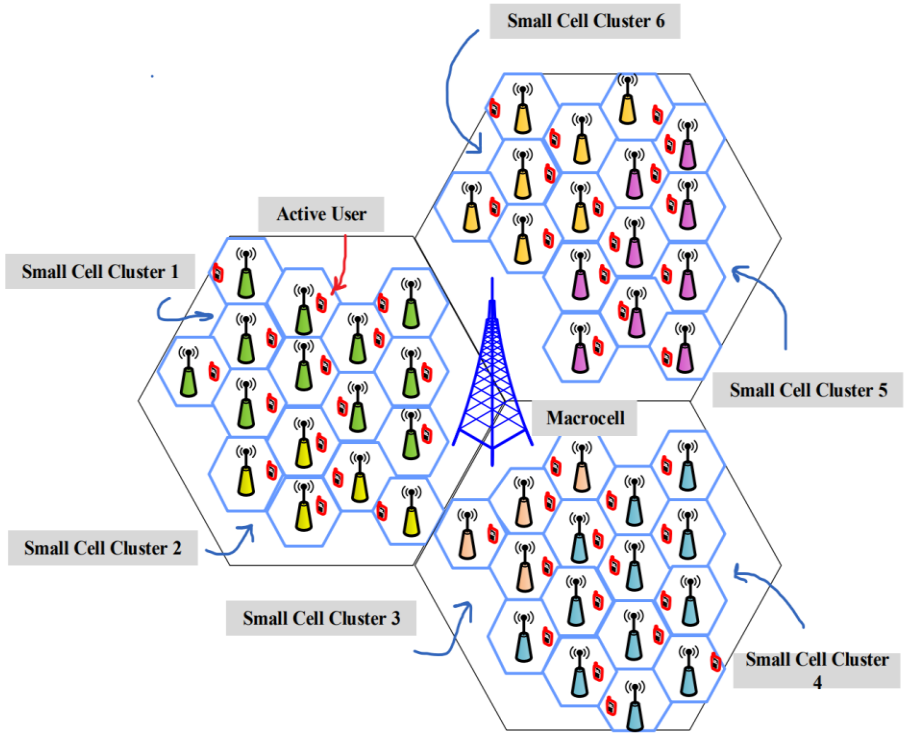

Fig 2.3. An illustration of Small Cells deployment

2.3. Motivation for Network Densification

As a result of huge data increment, operators and engineers need to find new methodologies to increase the network capacity. As discussed earlier, capacity can be efficiently improved by three ways: spectral efficiency improvement, wider bandwidth, and network or cell densification. Among all the approaches of 5G technologies for

increased data rates, network densification can be considered to be a promising solution for achieving realistic higher capacity goals [23]. The term network densification refers to the dense deployment of small cells over the cellular coverage of a macro-cell [4].

In the network densification technique, service locations are densified with more base stations or small cells so that the distance between the active users and small cells is reduced. This improves the system throughput, spectral efficiency and network performance especially in the high traffic demand areas [24]. Network densification also benefits the in-building coverage and cell-edge performances. Due to building penetration loss, signal strength deteriorates and causes problems for outdoor to indoor communications. Network densification using more cells also solves the coverage problems.

2.3.1. Small Cells:

A UDN will consist of a large number of small base stations, which will be densely distributed in a small area. Small cells bring different opportunities for the network operators to meet dense deployment scenarios. Small cells are low-powered cellular radio access points with small coverage distance. These small cells may have transmission range from tens of meters to hundreds of meters depending on cell size and transmission requirements. They can be deployed in indoor environment as well as outdoor areas. These are also known as micro-cells, femto-cells, and pico-cells in small cell terminology depending on transmission range. Femto-cells [25] perform indoor more efficiently as compared to outdoor since they have lowest transmission range and lowest resource allocation while microcells are available for highest resources and more suitable for outdoor access [26].

Deploying small cells in existing network gives more Quality of Service (QoS) and Quality of Experience (QoE) to end user but it requires high number of small cells in small area. This may also create new challenges in wireless communication. Due to highly dense networks many problems occur such as interference management, mobility, and backhaul energy consumption [23]. More small cells create interference in existing networks. Interference can be caused when two cellular networks shares same frequency band or adjacent frequency band for communication. To provide coverage to larger area, more

small cells may need to be deployed. Thus, this may need the deployment planning of small cells. Usage of spectrum in small cells and macro-cell creates new challenges for network designer. There are two options to use spectrum: in-band (spectrum sharing) and out-band (spectrum splitting) [27]. In in-band, small cells and macro-cells use the same spectrum band whereas in out-band small cells and macro-cells use different spectrum bands. In-band spectrum is used for low density network and is more suitable when more mobility and continuous traffic is needed but it also reduces network capacity due to interference among small cells and macro-cell. Out-band spectrum solution is more suitable for high density network design [28].

2.4. Backhaul Solutions

As explained in Section 2.3, despite the benefits brought by small cells in a UDN, there are many challenges that need to be resolved. The major concern among all is backhauling in an ultra-dense network. In UDNs, the essential part that works as the backbone of the entire network is the backhaul architecture. Backhaul network should be capable to handle ultra-dense small cell network traffic by providing enough capacity and stability to the network. It also needs to be cost effective and scalable, otherwise backhaul can become bottleneck of the network. Small cells in a UDN connect with the network via wired or wireless backhaul connection.

2.4.1. Wired Solution:

Fiber optic has been considered as a standard wired backhaul connection that is able to support high capacity and low delay [29]. It is the most stable and straight forward backhaul with no interference and better performance. However, wired backhauls are not a feasible solution for an ultra-dense small cell network as installing fiber at every small cell for future networks is expensive and impractical. It takes longer deployment times and increases backhaul cost [30].

2.4.2. Wireless Solution:

Wireless backhaul technologies have become popular because of their high deployment flexibility and cost efficiency. Many wireless backhaul solutions exist such as sub-1 GHz band, 1-6 GHz band, above 6 GHz band, free space optics (FSO) and, mm-

wave band for 5G networks. Recently, numerous studies [31][32][33] on wireless backhaul technologies highlight mm-wave wireless backhaul as the most promising backhaul solution for 5G communications. [31] investigated the advantages and disadvantages of mm-wave and FSO to fronthaul/backhaul links and provided analysis showing mm-wave as more beneficial compared to FSO in terms of energy efficiency and availability. Mm-wave backhaul based massive MIMO scheme has been proposed in [32] for 5G ultra dense network. It is shown that mm-wave can be easily merged with massive MIMO by deploying large number of antennas in the wireless backhaul network. [33] Evaluated the advantages such as low latency and high quality of service of mm-wave wireless backhaul. In recent 3GPP Release 16, mm-wave is considered as an acceptable backhaul solution for small cell networks [7].

Based on the above studies, mm-wave is a more attractive and viable solution for wireless small cell backhaul links. Millimeter wave backhaul does not allow to have large transmission range because of high path loss. Generally, 200 m radius is considered as ideal range for small cells in the literature [5][34][35].

2.5. Network Architecture

Different backhaul architectures are used in the literature to achieve better performance, high backhaul throughput, reliability and cost effective solution. Two popular architectures are centralized architecture and distributed architecture. The architecture scenarios of centralized and distributed networks are presented in Fig 2.4 and Fig 2.5 [30].

2.5.1. Centralized Architecture:

In the centralized architecture, the small cells are connected to a single point (usually the site of a macro-cell) through which the entire traffic is backhauled to the core by fiber optic cables. As shown in the Fig 2.4, macro-cell base station is located at center and small cells are scattered within the macro-cell. Small cells transmit their backhaul traffic through wireless links to macro-cell base station (MBS) and MBS forwards all accumulated wireless backhaul traffic of small cells to the core through Fiber to the curb (FTTC) links.

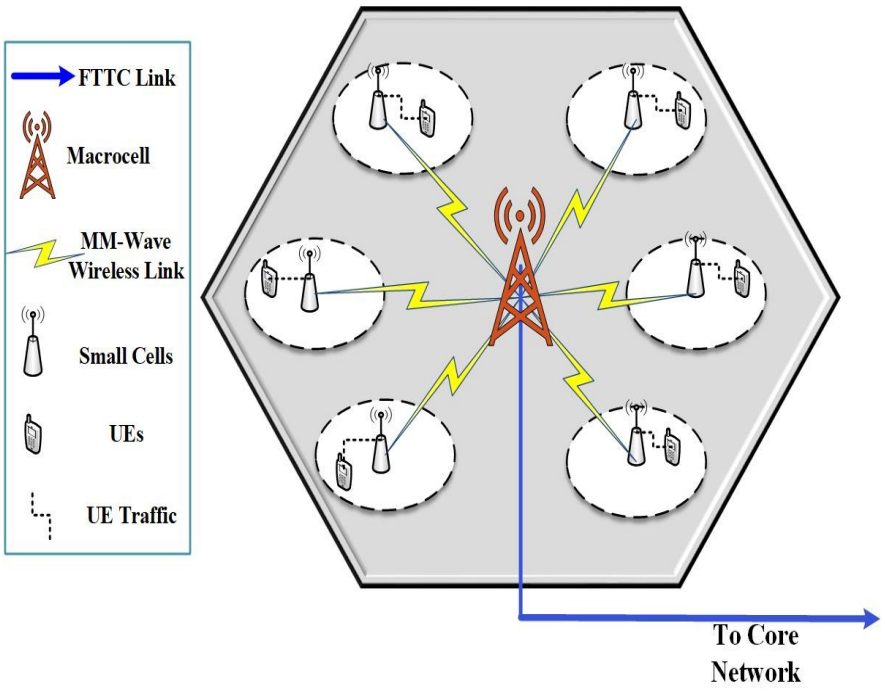

Fig 2.4. Centralized Architecture [30]

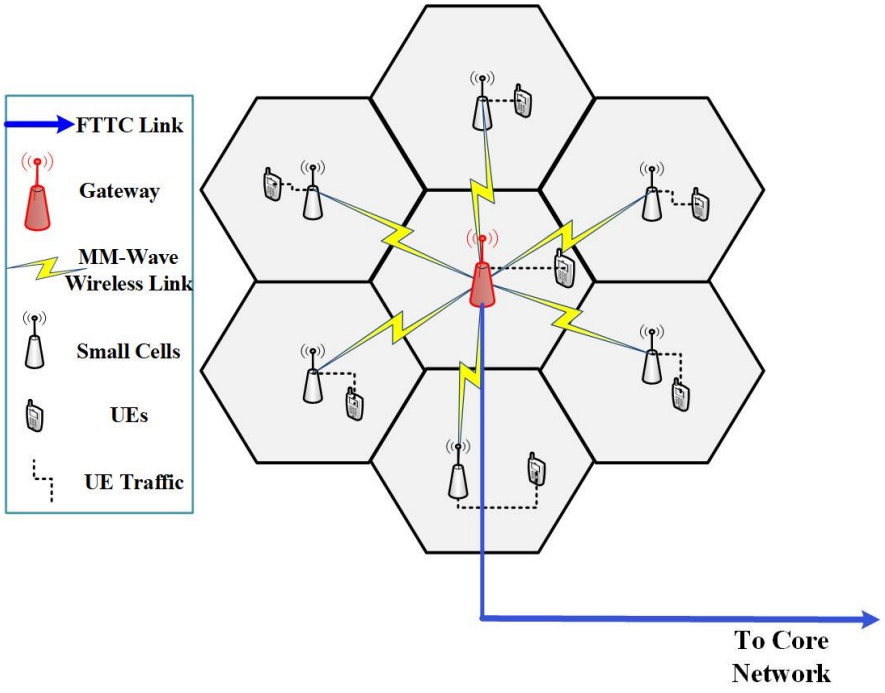

Fig 2.5. Distributed Architecture [30]

2.5.2. Distributed Architecture:

In the distributed architecture, the small cells are clustered into groups. Each small cell connects to a gateway and gateways connect to the core via fiber. As shown in Fig 2.5 in distributed architecture, no macro-cell is present at the center to collect all backhaul traffic of small cells. The backhaul traffic of small cell is forwarded to the nearest neighboring cells and further relayed to gateways using wireless links. Gateways transmit small cell backhaul traffic to the core through FTTC links.

Most reported work compares the centralized scheme to the distributed scheme. In [9], the distributed architecture is shown to have better results in terms of capacity and energy efficiency. It is stated in [30] that the distributed architecture achieved higher throughput performance compared to centralized architecture. [36] suggested the use of distributed architecture for its better scalability compared to centralized architecture and distributed architecture gained better results in comparison with centralized architecture.

2.6. Related Work on the Gateway Location Problem

Gateway Location Problem is the combinatorial optimization problem where a fixed number of gateways are selected out of set of *N* nodes and each node is assigned to a suitable gateway in a way that sum of total distance is minimized.

Several algorithms have been proposed for the GLP in scenarios such as Wireless Mesh Networks, Wireless Sensor Networks, and Satellite Networks. The work in [37] focused on gateway placement in 5G satellite hybrid networks for improving network reliability. Two algorithms, an optimal enumeration algorithm and a cluster-based approximation placement algorithm, are applied for the GLP. [38] proposed the method of Multiple Surface Gateways Positioning in underwater sensor networks. [39][40][41] focused on the GLP for wireless mesh networks, showing that a GA works better compared to other algorithms for optimizing the locations of gateways. [39] used a GA for solving the optimization problem by minimizing the variance in hops count between each internet gateway and its associated meshed router in the network. [40] implemented a GA and a Simulated Annealing algorithm for optimization in WMN for improving performance in

terms of cost and quality of service. In [41], the comparative analysis shows that a genetic algorithm performs best among all combinatorial algorithms.

There is little work on the GLP in UDNs (or UDCs). In [42] the authors explained dense base station deployment for assigning larger bandwidth resources to users. The main objective of the algorithm was to maximize the overall coverage while minimizing the number of gateways. [43] focused on the joint selection of cluster heads and number of base station antennas to maximize the overall system throughput. [44] concentrates on maximizing the wireless BNC, but optimally selecting gateways was not investigated. [45] proposed a solution for the gateway placement to increase energy efficiency.

2.7. *p*-median Problem

The *p*-median problem is in the larger class of problems known as *minisum location–allocation problems* [46]. It is concerned with the location of p facilities (medians) to minimize the total weighted distance between the median points and the demand points. Hakimi [47] introduced the median location problem on graphs in 1964. The aim was to select medians and assign all other points to their nearest median with the goal of minimizing the sums of the distances between the centers and points in their partition. The number of feasible solutions for the *p*-median problem is $N! / (p! (N-p)!)$, which is known to be NP-hard problem on general graphs. This highlights the characteristics of the *p*-median problem, i.e., as the size of the problem increases, it quickly becomes too large and complex to solve. A number of methods for the *p*-median problem have been proposed and developed in general networks. If the basic graph of the network is a tree, then the *p*-median problem can be solved with known algorithms in polynomial-time.

2.7.1. Complexity of *p*-median Problem:

Matula and Kolde [48] in 1976 provided an algorithm with complexity $O(N^3p^2)$ for locating the *p*-medians of a tree where p is the number of facilities to select out of total N facilities and $p > 1$. In 1979, Kariv and Hakimi [49] proved that the *p*-median location problem on a general network is NP-hard. In addition to this, they also investigated the *p*-median problem on tree graph networks and designed an algorithm with complexity

$O(N^2p^2)$. A new algorithm was designed in 1982 for solving the *p*-median problem on trees with $O(N^3p)$ complexity by Hsu [50]. Around after 15 years Tamir [51] improved the time complexity of the *p*-median problem on tree networks to $O(N^2p)$. In 2005, Benkoczi and Bhattacharya [52] designed an algorithm for the *p*-median location problem on trees with $O(N^2log^{p+2})$ runtime. A complete bibliography of median location problems is given in [46].

Besides exact algorithms, several heuristics exist for the *p*-median problem [53]. In this thesis, we focus on Genetic Algorithms because of their stochastic search characteristics and machine learning based *K*-means and *K*-medoids clustering algorithms for their unsupervised learning nature.

2.8. Genetic Algorithms

Optimal algorithms can solve only small instances of the *p*-median problem in reasonable time, so heuristic solutions are needed. A genetic algorithm is parallel in nature, which improves speed when applied to the *p*-median problem [54]. Genetic algorithms are stochastic, efficient, and easily manageable for complex problems. They are a powerful yet simple technique to solve complex problems and have been widely used in analyzing data, integrating information, and using the resulting insights to improve decision making [55][56]. In the fields of neural network, computer science, machine learning, artificial intelligence and others, GAs are used as a stochastic search and optimization heuristic [57][58]. GAs search for a suitable solution through evolution and randomness. A solution is represented as a "chromosome" string. A population of chromosomes is initially generated randomly. These have an associated fitness score which affects their probability of moving to the next operation. Pairs of chromosomes are selected for the crossover operation to create new chromosomes by swapping their ends at a random crossover point. These chromosomes are then subjected to the mutation operation. These operations produce a new population of chromosomes (a new generation), and over a series of generations, better solutions are evolved [59][60]. The basic steps in a GA are outlined in Algorithm 2.1 and explained below [61].

Algorithm 2.1. Genetic Algorithm

Inputs:

- Population size n_{pop}
- Max number of generations G_{max}
- Mutation probability
- Fitness function

Output:

- The best individual in all any generation

BEGIN

1. Generate initial population.

2. Calculate Fitness of each individual chromosome in the initial population.

3. ***Do*** G_{max} times:

4. *Selection*: select *p* chromosomes using probabilities weighted by their relative fitness score.

5. *Crossover*: apply the crossover operator to $n_{pop}/2$ randomly chosen pairs of chromosomes.

6. *Mutation*: randomly mutate some chromosome elements using the mutation probability.

7. Calculate fitness value of new chromosomes.

8. Update the population for next generation.

9. Return the chromosome having the best fitness function in any generation.

END

- *Initial population:* Two important aspects of *population* are the size of the population and the generation of initial population. The initial population is considered from a large set of design space in order to reduce computational complexity. Ideally, the

larger the initial population, the better the results as it is easier to explore more possibilities. The common encoding method for GA is binary encoding, which is suitable for discrete solution space. Modern computers also allow chromosomes to include real numbers, permutations and many other objects.

- *Fitness Function*: Based on the objective function of the problem, *fitness value* is calculated for all chromosomes. Fitness function is used every time for calculating fitness values of new generated population.

- *Selection Process*: After calculating the fitness values of all initial chromosomes, next step is to follow the *selection process*. Basically, selection is the process to select chromosomes from population to perform crossover and mutation operation. Higher the fitness value, higher the chances of chromosomes to get selected for next operations. Selection process should be balanced with variations. Strong selection will manipulate the process to highly fit individuals taking over the population and very weak selection may result in too slow evolution. So, balancing the diversity is necessary for progress and change. Roulette wheel selection, Tournament selection, and Rank selection are some common methods for selecting the chromosomes for the mating pool. In a *Roulette wheel method*, the circular wheel is divided into n_{pop} slots, where n_{pop} is the number of chromosomes. So, each chromosome has some share of the wheel, which is proportional to the fitness value. *Tournament Selection method* generates selective pressure by holding tournament competition among individuals and selects the best individuals for further operation. *Rank selection method* uses the strategy of ranking based on the fitness value. Worst fitness has ranking number 1 and best fitness has n_{pop} number ranking. In result, potentials parents are selected for next processes. After selection process, population is enriched with better individuals. Selection will help to make combinations of good individuals, but it will not create the new offspring.

- *Crossover Operation*: *Crossover* matches two individuals and exchanges their genes to create new off-springs. It is the process to produce new off-springs (children) from fitted individuals (parents). Different crossover methods are *Single point crossover*, *Two-point crossover*, *Multi-point crossover* and *Uniform crossover*.

- *Mutation Operation*: *Mutation* is the next step after the crossover operation for adding diversity into the algorithm. It is basically a process where genes of individuals are randomly changed. Bit Flop Mutation, Swap Mutation, and Scramble Mutation are few techniques for mutation function. Easiest way to perform mutation operator is to convert the bit from '0' to '1' or vice versa. Mutation probability is an important parameter in mutation operation. Based on the probability, mutation function adds randomness to the chromosomes. Ideally, value of mutation probability should be kept low otherwise GA will change to random search.

After crossover and mutation procedure, GAs create the new generation that will be used in next iterations to find the best individual or chromosome. The selection, crossover and mutation processes continue for number of iterations. In GA, chromosomes with higher fitness values have higher chances to get selected for next iterations. This characteristic of GA will help to solve large complex optimization problems. Moreover, GA constantly discards poor solutions and evaluates more and more better solutions.

2.8.1. Complexity of Genetic Algorithm:

The computational complexity of a genetic algorithm is well studied [62][63][64][65]. It depends mainly on the problem size, the fitness function, and parameters such as the selection probability, mutation probability, number of chromosomes, etc. The GA complexity is $O(NGn_{pop})$ where N is the number of nodes or data points, G is the number of generations, and n_{pop} is the number of chromosomes [66]. The complexity is analyzed as follows. Suppose that the mutation probability (P_m) and crossover probability (P_c) for single point crossover are less than 1. The selection process completes its operation in n_{pop} operations at each iteration. Single point crossover exchanges the values in $O(NP_c\, n_{pop}/2)$ time and mutation process mutates its bits in $O(NP_m\, n_{pop})$ time where $P_c << 1$ and $P_m << 1$; this reduces the complexity to $O(N\, n_{pop})$ in a single generation. So, the overall complexity of a genetic algorithm for G generations is $O(NGn_{pop})$.

2.9. *K*-means Clustering Algorithm

K-means is a popular and widely used machine learning algorithm for data mining across different disciplines. It is used to process large amounts of unstructured data [67][68]. The main goal of the *K*-means clustering algorithm is to divide *N* data points into *M* clusters so that the within-cluster squared Euclidean distance is minimized [69]. The *K*-means clustering can be applied to unlabeled data i.e., the data that does not belong to any group or cluster. The steps are given in Algorithm 2.2 and explained as follows:

Algorithm 2.2. *K*-means Clustering Algorithm

Inputs:

- Set of *N* data items $D = \{\boldsymbol{d1}, \boldsymbol{d2}, \ldots\ldots, \boldsymbol{dn}\}$
- Number of desired clusters *M*
- Maximum number of replications

Output:

- A set of *M* clusters having *M* centroids

BEGIN:

1. ***Do*** for maximum number of replications:
2. Arbitrarily choose ***M*** initial centroids.
3. Repeat until convergence is achieved.
4. Assign each data point to the nearest cluster.
5. Calculate the new centroid of each cluster.
6. Return the result having the smallest sum of within-cluster Euclidean distances.
END

- *Initialization:* First step is to initialize the cluster centers randomly among all data points. It is the crucial stage in *K*-means clustering algorithm as the final clustering results may vary depending on the choice of the starting point for big datasets [70].

- *Squared Euclidean Distance*: Next step is to assign each data point to its nearest cluster center depending on the squared Euclidean distance. It is a straight-line distance between the two points and denoted as follows:

$$Squared\ Euclidean\ Distance = \sum_{i=1}^{N} \sum_{j=1}^{M} (||N_i - M_j||)^2 \qquad (2.2)$$

where, $||N_i - M_j||$ is the distance between a point N_i and a centroid M_j iterated over all N points in the i^{th} cluster for all j clusters.

- *Updated centroids:* The next step of the algorithm is to re-calculate the new centers for the clusters. The new cluster centroids are determined by calculating the mean of all cluster data points. Below given equation is used for calculating the new centroids.

$$Centroids = \sum_{j=1}^{M} (||N_i - M_j||) \ / \ N \qquad (2.3)$$

After these M new centroids, each data point is re-assigned to its nearest new centroid. Process is repeated until no changes occur within the replication.

2.9.1. Complexity of *K*-means Algorithm:

The computational complexity of *K*-means is $O(NMI)$ [71], where N is the number of data points (or number of small cells), M is the number of clusters, and I is the number of iterations. Generally, $M << N$, and the computational complexity of *K*-means reduces to $O(NI)$.

2.10. *K*-medoids Clustering Algorithm

The *K*-medoids clustering algorithm is similar to *K*-means, but the cluster center, called a medoid, must be a member of that cluster. Unlike *K*-means, this algorithm returns medoids that are actual nodes. It uses the approach of *partitioning around medoids* (PAM) [59] and proceeds in two steps:

- *Build Step*: For cluster center initialization, M nodes out of N are selected randomly as medoids. Then, M clusters are constructed by assigning each node to the nearest medoid based on squared Euclidean distance.

- *Swap Step*: Within each cluster, each node (or small cell in our case) is tested as a potential medoid by checking whether the sum of within cluster distances gets smaller using that node as the medoid. If so, the node is defined as a new medoid.

The *K*-medoids algorithm iterates through the build and swap steps until the medoids do not change. The steps are presented in Algorithm 2.3.

Algorithm 2.3. *K*-medoids Clustering Algorithm

Inputs:

- Set of *N* data items $D = \{\boldsymbol{d1}, \boldsymbol{d2}, \ldots\ldots, \boldsymbol{dn}\}$
- Number of desired clusters *M*
- Maximum number of replications

Output:

- A set of *M* clusters having *M* medoids

BEGIN:

1. ***Do*** for maximum number of replications:
2. Arbitrarily choose ***M*** initial medoids.
3. Repeat until convergence is achieved.
4. Assign each data point to the nearest medoid.
5. Calculate the new medoid of each cluster.
6. Return the result as medoid having the smallest sum of within-cluster Euclidean distances.

END

2.10.1. Complexity of *K*-medoids Algorithm:

The computational complexity of *K*-medoids is $O(M\ (N - M)^2 I)$ [62], where *M* is the number of clusters or gateways, *N* is the number of nodes (or number of small cells), and *I* is the number of iterations. Generally, $M << N$, and the computational complexity of

K-medoids reduces to $O(N^2I)$. The *K*-means is more efficient compared to *K*-medoids in cases where the number of data points (or small cells) is large.

2.11. Dijkstra's Shortest Path Algorithm

Dijkstra's algorithm is a well-known algorithm to solve shortest path for a given network [14]. It builds shortest path from a source node to the all other nodes for a given graph *G (V, E)*, where *V* is the set of nodes and *E* is the set of edges. Each of the edges in *E* has a weight, which represents the length of the edge in terms of hops or distance. The detail description and pseudocode of Dijkstra's algorithm can be found in many places in the literature [72][73][74].

2.11.1. Complexity of Dijkstra's Shortest Path Algorithm:

The computational complexity of Dijkstra's algorithm for single source shortest path is $O(N^2)$ [75][76] where *N* is the number of nodes. So, for *N* sources of shortest path trees, the computational complexity becomes $O(N^3)$.

Chapter 3: PROBLEM STATEMENT

Ultra-dense networks can meet the very high demand for spectral resources in 5G. The most straightforward way to densify a macro-cell is to implement within it many small cells, each of which with a miniature base station that operates with low power. This is a form of cell splitting that would greatly expand the spectral resources via frequency reuse. This massive number of small cells deployment requires efficient backhaul solution to carry the entire access traffic received by all small cells to the network core. The backhaul network should be accessible, cost effective and capable of handling the large amount of small cell traffic. The capacity of the backhaul network must be at minimum equal to the sum of capacities of all small cells. To manage the dense deployed network, planning and optimization of the backhaul network is essential because it is expensive and complicated to connect large number of small cells to the core through wired backhaul connections. A mm-wave wireless backhaul can be an attractive solution for high capacity links in 5G networks [77]. Mm-wave based backhaul network provides large bandwidth but it faces some challenges such as high path loss, directivity challenges, low penetration and more [78]. These problems lead to shorter transmission range. So, it is important to study how to use and share wired, and wireless backhaul links in a 5G ultra dense network for satisfying backhauling requirements at reasonable cost.

A possible solution would consist of a distributed wireless multi-hop architecture where few small cells act as gateways that are connected to the core through fiber while the remaining small cells are connected to these gateways through mm-wave wireless links. Each small cell is associated with one gateway and all small cells forward their traffic through intermediate small cells to their respective gateway through multi-hop mm-wave wireless links. By using multi-hop architecture, the benefits are to have less interference via multiple shorter links. However, as discussed earlier, small cells in a UDN are expected to be deployed in large numbers without any detailed network planning; so, it is necessary to know which small cells are used as gateways. In the above context we explore and investigate the gateway location problem, in which we assess the performance of our proposed solutions in terms of backhaul network capacity and average number of hops in distributed multi-hop wireless backhaul architecture. Proper selection of gateway locations

minimizes the average number of hops from small cells to gateways which leads to higher backhaul network capacity of 5G ultra-dense network.

3.1. Gateway Location Problem in Ultra-Dense Networks

Our main goal is to maximize the capacity of the wireless backhaul network by choosing M gateways out of the N small cells such that the average number of hops in the entire coverage area is minimized. Each small cell has specific wireless links to its neighbor with capacity equals to W_S. The capacity of gateway is denoted by W_G. A formula for the backhaul network capacity of the UDN is derived in [44] and is given below:

$$C(M, N) = \frac{\min\left(N \cdot W_S,\ M \cdot (W_G - W_S)\right)}{\min\left(\overline{Y(M,N)}\right)} + M \cdot W_S \qquad (3.1)$$

In this formula, $\min\left(\overline{Y(M,N)}\right)$ represents the "minimum average' number of hops from a small cell to its associated gateway. The "average" is taken over the entire backhaul network.

$$\overline{Y(M,N)} = \frac{1}{N}\sum_{i=1}^{N} Y_i\,(M,N)\,, \qquad (3.2)$$

where $Y_i(M,N)$ is the number of hops between i^{th} small cell and its associated gateway, i = 1,…, N, and $M \geq 1$.

In developing this formula, the authors maximized the total number of bits generated at all small cells and successfully delivered to all gateways within a period of T, and took the limit when T tended to infinity. Each information bit was counted once. For a given spectral bandwidth, the information throughput is achieved by maximizing the number of simultaneous transmissions (represented by the numerator of the first term) divided by the average number of hops. The second term (i.e., $M.Ws$) does not utilize any spectral resources because it represents the gateway's own access traffic transmitted directly over cables to the core.

It should be noted that in a UDN, the number of small cells is usually too large. As the traffic is forwarded to a gateway, each hop utilizes a wireless channel which carries larger and larger traffic as it approaches the gateway. The total required spectrum resources

increase as the number of hops increases. Note also that each bit is transmitted many times and this redundancy wastes spectral resources. Therefore, minimizing the number of hops is essential to increasing the capacity of the backhaul network to carry more information bits per unit time. Note that the average number of hops is referred to in this thesis as ANH.

To solve the GLP, one needs to consider all combinations of M gateways chosen from the N small cell locations, i.e., N choose M, which has billions of possible solutions for even moderate N and M. Enumeration of solutions is prohibitive for large systems.

This research problem involves:

- Ensuring the connectivity of the ultra-dense network based on the transmission range of the small cells.
- Generating the neighbor table of small cells which determines the connectivity within the wireless backhaul.
- Selecting the locations of gateways such that average number of hops from small cells to gateways is minimized.
- Calculating ANH based on the routing of small cells to the gateway locations. The goal is to assign small cells to gateways based on minimum number of hops and calculate the average of hop distance from all small cells to their respective gateways.
- Calculating the wireless backhaul network capacity according to (3.1) where the smaller the ANH, the higher the capacity.

As explained earlier in section 2.6., most of the work on gateway location problem is performed on the wireless mesh networks ,wireless sensor networks, mobile ad-hoc networks, and satellite networks [37][38][39][40][41]. However, there have not been much work done in the ultra-dense network (UDN) for gateway placement problem. Our work endeavors to maximize the backhaul network capacity by optimizing gateway locations through a combination of K-means and GA. Our approach directly tackles the GLP in a 5G multi-hop ultra-dense scenario.

In order to realize the effects of small cells distribution on backhaul network capacity, three different distribution scenarios of small cells are considered in this thesis.

Our work first tested on Uniform distribution scenario which is standard and simplest distribution method. In Uniform distribution, nodes are spread uniformly and makes distribution mathematically perfect, but this is rarely happening in real world. So, we also simulated multiple more realistic distribution topologies which are bivariate Gaussian distribution and Cluster distribution to analyze the performance. GD and CD represents the real-life high traffic demand areas.

The gateway location problem studied in this chapter is similar to the *p*-median problem of finding a set of *p* points out of *N* so that the sum of the weighted distances to the vertices is minimized, which is known to be NP-Hard [10]. Since GLP is NP-Hard, and since the computational complexity and time complexity of this optimization problem is too large and complex, we propose a heuristic solution to obtain a near-optimal solution to the GLP. Our aim is to develop a new solution which simplifies the GLP and quickly finds the gateway locations that minimize the ANH and maximize the BNC.

3.2. Effect of the Number of Gateways in Ultra-Dense Networks

As stated earlier, in multi-hop architecture, small cells route the traffic to the core through intermediate small cells. In the case of fully loaded small cells, small cells forward all collected access traffic to the gateways where all the traffic is aggregated at gateways to be transported to the core network. This situation may create bottlenecks within the backhaul network when the cell to cell capacity is smaller than the aggregated traffic. The probability of that happening is higher when the number of gateways is not large enough. Possible solution to avoid backhaul problem is to increase the number of gateways or by creating redundant routes (we did not explore such scenarios in this thesis). The ANH is also a reflection of the congestion in the backhaul. A maximum hop count of one will indicate the ideal case where each small cell is at a single hop from its GW. Increasing the number of GWs can decrease congestion by reducing the ANH.

To this end, analysis using different number of gateways is conducted to understand the effect of the number of gateways on the congestion in the backhaul. It is determined that more gateways can provide more stable network with more capacity and less accumulated traffic flow in distributed multi-hop wireless network. However, more gateways can result in higher cost and more complexity to the network. This situation creates a trade-off for

network operators. In order to characterize this trade off by simulations, we examine the effect of different number of gateways on traffic congestion via ANH. We consider various numbers of gateways for the 5G ultra-dense network and study their effect on ANH as well as on BNC.

This thesis work helps in answering the following fundamental question. How location and number of gateways effect the wireless backhaul network capacity of 5G ultra-dense networks under different distribution scenarios.

Chapter 4: OPTIMIZATION FRAMEWORK

In this chapter, we present the network model for the multi-hop wireless backhaul scheme for the ultra-dense network and describe the model assumptions. We introduce an integer linear programming (ILP) formulation to find optimal gateway locations. We propose our novel K-GA algorithm and analyze the essence of the algorithm to find the near optimal gateway locations for minimizing the average number of hops and maximizing the wireless backhaul network capacity. We also present the computational complexity for optimal (or exact) solution and K-GA algorithm.

4.1. Network Model

In the ultra-dense network considered in this research, the small cell density is assumed to be greater than the user density, indicating that many small cells will be in the inactive mode. It is assumed that in idle mode, no traffic is generated from some of small cells as they are completely inactive. The analysis of access part of the network is not in the scope of this thesis but in general access links may either operate on mm-wave links or using other links and each user should be able to access a significant amount of bandwidth. For backhaul part of the network, small cells connect to the nearest gateway and forward their backhaul traffic to the respective gateway via mm-wave wireless links, and gateways forward all the aggregated traffic to the core via fiber. Backhaul connections aggregate all small cells traffic to the gateways, thus the backhaul links are considered to be high capacity links to ensure stable communication. In the multi-hop scenario, the small cell is connected to the gateway via a set of intermediate point-to-point line of sight connections. In our model, we consider the multi-hop backhaul approach where each small cell connects to the neighboring cells or gateways only if adjacent cells are within the fixed backhaul coverage area of small cells. The selection of path from a small cell to the gateway is determined by the least number of hops between them.

The overall model is shown in Fig 4.1 where we consider a distributed multi-hop wireless backhaul UDN architecture in which the macro-cell coverage area is divided into N small cells. M out of N cells are designated as gateways. The gateways are connected to the core network by fiber optic links with very high capacity. The remaining (N-M) cells are

grouped into *M* clusters and each cluster is served by one gateway. Each small cell connects to its serving gateway either directly or through multiple hops within the cluster. We assume that small cells use mm-wave links to connect with each other or with the gateway [44].

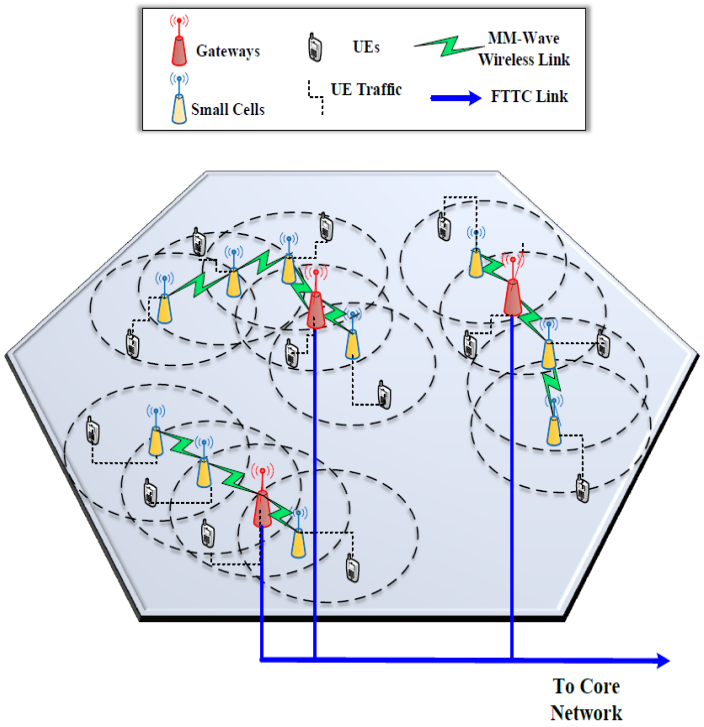

Fig 4.1. Network Model

4.1.1. Model Assumptions:

The main assumptions for this model are listed below:

- Each small cell has a fixed circular backhaul coverage area.
- The backhaul coverage areas of small cells are equal and do not fully overlap.
- Some of the small cells are considered to be in idle mode as all small cells may not have always users connected in the case of UDN.
- There is a line of sight among neighboring cells.
- Two small cells must be in the transmission range of each other to communicate, which means that they must be within the backhaul coverage area of each other.

- The capacity of mm-wave wireless backhaul links is considered to be larger than the incoming access traffic.
- Each small cell has a specific capacity W_S, which is the capacity of the wireless link between a small cell and its neighbor.
- The capacity of a gateway is W_G, which is the capacity of the fiber link between the GW and the core.
- Any small cell can be selected as a gateway for forwarding backhaul traffic to the core network.

4.2. Mathematical Implementation

The **G**ateway **L**ocation **P**roblem in the distributed wireless backhaul for 5G ultra-dense networks is similar to the *p*-median problem. The NP-hard *p*-median problem was originally defined by Hakimi (1964) [47]. The important aspect of the *p*-median problem is that each center is chosen from among the given points.

4.2.1. Problem Formulation for Gateway Location Problem:

Mathematically, the GLP problem for 5G ultra-dense small cell networks can be summarized as follows:

Inputs:

d_{ij} = Distance between small cell i and gateway j

p = Number of gateways to locate

Output: Lowest ANH based on best gateway locations

Decision Variables:

$$X_j = \begin{cases} 1, & if\ we\ locate\ gateway\ at\ small\ cell\ j \\ 0, & if\ not \end{cases} \quad (4.1)$$

$$Y_{ij} = \begin{cases} 1, & if\ small\ cell\ i\ is\ served\ by\ gateway\ j \\ 0, & if\ not \end{cases} \quad (4.2)$$

Minimize:

$$\sum_i \sum_j d_{ij} Y_{ij} \quad (4.3)$$

Subject to:

$$\sum_j Y_{ij} = 1 \quad \forall i \in N \quad (4.4)$$

$$\sum_j X_j = p \quad (4.5)$$

$$Y_{ij} - X_j \leq 0 \quad \forall j \in M ; i \in N \quad (4.6)$$

$$X_j \in \{0,1\} \quad \forall j \in M \quad (4.7)$$

$$Y_{ij} \in \{0,1\} \quad \forall j \in M ; i \in N \quad (4.8)$$

- The objective (4.3) minimizes the total weighted distance of all small cells to the assigned gateways.
- Constraint (4.4) ensures that each small cell is assigned to exactly one gateway.
- Constraint (4.5) ensures exactly p gateways are selected.
- Constraint (4.6) ensures that any small cell i is assigned only to a location that is a gateway ($X_j = 1$).
- Constraints (4.7) and (4.8) specify that the decision variables (location variables X_j and allocation variables Y_{ij}) are binary.

4.2.2. Solving GLP in CPLEX Optimization Studio:

This p-median problem formulation is solved using the CPLEX mixed-integer solver [79]. The input data are:

- A finite number of small cells with locations.
- A finite number of possible gateway locations. In our case any small cell can be selected as a gateway.
- A distance matrix *d[N][N]* with the dimension of $N \times N$ where N is the number of small cells. Each element d_{ij} represents the smallest number of hops between small cell i and

small cell *j*, calculated via the Dijkstra's algorithm where each potential link has nominal length 1. A shortest route tree formulation reduces calculation effort.

The procedure to solve *p* median problem in CPLEX Optimization Studio is as follows:

- The *d* [*N*][*N*] distance matrix is saved to .dat file in IBM ILOG CPLEX Optimization Studio.
- Above Inputs are used in .mod file of IBM ILOG CPLEX Optimization Studio.
- Mathematical problem formulation as explain above is included in .mod file for implementing *p*-median problem.
- Optimization Programming Language (*OPL*) is used to write the mathematical model.
- The default configuration is run to obtain optimal output.

The solution of this problem provides the total minimum distance (in hops) from gateways to small cells along with the optimal gateway locations. The minimum average number of hops is calculated using,

$$ANH = Total\ minimum\ distance\ /\ (N - M) \qquad (4.9)$$

4.2.3. Complexity of Optimal Solution:

Based on the earlier discussion for shortest path trees and *p* median complexities in chapter 2, the computational complexity of our GLP problem is $O(N^3 + (N^2 log^{p+2}))$. GLP includes the complexity of calculating shortest path trees for all nodes for finding the distance matrix plus the complexity of the *p*-median problem. Exact methods search through the solution space to identify the optimal one going over every possible solution. This makes the computational complexity and time complexity of optimal methods excessively large and as the size of the problem instance increases, it rapidly becomes too large to solve.

4.3. K-GA Heuristic

4.3.1. Framework of proposed K-GA Algorithm:

Our proposed K-GA heuristic combines the simplicity of *K*-means with the natural process of evolution of GA. Using GA (an artificial intelligence technique) and *K*-means (a

machine learning technique) together, the proposed K-GA becomes better equipped for handling complex problems. It has three phases which are described next and steps are listed in Algorithm 4.1.

A. *K-means Clustering*:

First we start with unsupervised ML algorithm, the *K*-means clustering algorithm. The *K*-means clustering algorithm uses the squared Euclidean distance parameter, and constitutes the first phase of K-GA. It is executed in two different stages: (i) The first stage finds the centroids depending on the number of clusters and the data points, and then associates each small cell with the nearest centroid. (ii) The second stage updates the centroids. New centroids are calculated by taking the average of the locations of the small cells associated with the cluster. The small cells are then reassigned to the new centroids.

The process is iterated within a replication until no further changes in centroids or small cell associations occur [80]. R_{max} is the maximum number of replications and controls the number of times the clustering process is repeated. For each replication, the initial centroids are selected randomly. The best result from all replications is selected as the final result of this stage. It consists of *M* centroids that have the smallest sum, over all clusters, of the within-cluster sums of small-cells-to-cluster-centroid distances.

B. *Combining K-means and GA*:

In this phase, we generate an initial population for the next phase of K-GA. Using the *K*-means result, we select the $t_1, t_2, t_3, \ldots, t_M$ small cells nearest to the *M* centroids. Note that the number of small cells associated with each centroid could differ. For example, if $M = 4$ then $t_1 = t_2 = t_3 = t_4 = 4$.

C. Genetic Algorithm:

A Genetic Algorithm [61] comprises the last stage of K-GA. An individual GA *chromosome* is a binary string of length *N* with one position for each small cell. A '1' indicates that the associated small cell is a gateway, and a '0' indicates that it is not. GAs require a *population* of numerous different chromosomes, each encoding a different solution specifying the location of the *M* gateways.

Algorithm 4.1 K-GA Heuristic

Inputs:

- Locations of small cells
- M: number of gateways to be chosen
- t: number of small cells closest to each K-means centroid
- R_{max}: maximum number of K-means replications
- P_m: GA mutation probability
- G_{max}: GA maximum number of generations

Output:

- Gateway locations

BEGIN

1. ***Do*** R_{max} times:
2. Arbitrarily choose M initial centroids
3. Repeat until convergence is achieved:
4. Assign each small cell to the cluster having the closest centroid
5. Calculate the new centroid of each cluster
6. Save the result of this replication
7. Find the best result among all replications and extract locations of the M centroids
8. Select t small cells nearest to each of the M centroids
9. Generate t^M small cell combinations as the initial population of chromosomes
10. Calculate *fitness* of each chromosome in the initial population
11. ***Do*** G_{max} times:
12. Perform *selection* process t^M times to generate the *mating pool*
13. ***Do*** $t^M/2$ times:
14. Perform *crossover* operation to generate two new chromosomes
15. Perform *mutation* operation using P_m
16. Perform *repair* operation as necessary
17. Calculate *fitness* of the two new chromosomes
18. Find and save the chromosome with the best fitness in this generation
19. Replace current population with the new population
20. Extract and output the gateway locations from the saved best chromosome

END

The GA population size is determined by the process of generating the initial population, as follows. The *K*-means stage returns *M* clusters with *M* centroids. We then choose *t* small cells nearest to each centroid and generate an initial population of t^M chromosomes by listing all combinations of one gateway taken from each set of *t* small cells near each of the *M* centroids. For example, $M = t = 4$ will result in $4^4 = 256$ combinations for gateway locations, constituting the initial population of 256 chromosomes.

The *fitness* value of a chromosome is the ANH associated with the solution it encodes. The fitness function runs shortest path algorithm to assign small cells to gateways and generate shortest path trees for calculating the ANH as the fitness value. A small cell closest to a GW in terms of number of hops becomes part of the shortest path tree centered at that GW. Lower values of ANH indicate better fitness since the target is to maximize the capacity by minimizing the ANH.

The *selection operator* generates the *mating pool* by selecting chromosomes from the population using probabilities weighted by the fitness value associated with each chromosome (Roulette wheel selection).

The *crossover operator* then randomly selects pairs of chromosomes from the mating pool, randomly determines the *crossover point*, and swaps the ends of the chromosomes beyond the crossover point to generate two new chromosomes. *Mutation* follows crossover to add diversity. The *mutation probability* (P_m) controls the random mutation of bits in a chromosome.

A *repair procedure* may be needed after crossover and mutation if the number of gateways in a new chromosome is not equal to *M*. If there are too many gateways, then a randomly chosen '1' is converted to '0'. If there are insufficient gateways, then a randomly chosen '0' is converted to '1'. After mutation, the fitness values are calculated. The process of crossover and mutation repeats until enough new individuals have been produced to create a new generation.

The selection, crossover, and mutation processes continue for G_{max} generations. The chromosome having the best fitness over all generations is output, providing the final

gateway locations and allowing the calculation of ANH and BNC using (3.2) and (3.1), respectively.

4.3.2. Complexity of K-GA Algorithm:

Based on explanation of the *K*-means algorithm and genetic algorithm complexities, K-GA's complexity is $O(NI + NG\, n_{pop})$ which is analyzed as follows. K-GA algorithm takes $O(NI)$ time to generate best cluster centers for initial population of GA, where N is the number of small cells and I is the maximum number of iterations for *K*-means algorithm. The complexity of final stage of K-GA is denoted by $O(NG\, n_{pop})$ where N is the total number of small cells, G is the number of Generations for GA, and n_{pop} is the number of chromosomes. Hence, the complexity of K-GA algorithm is $O(NI + NG\, n_{pop})$.

Computational complexity of K-GA consists of computational complexity of *K*-means and genetic algorithm, which makes K-GA more complex compared to *K*-means and GA heuristics. Due to its complex nature, K-GA can reach near optimal solutions that cannot be reached by the individual algorithms. So, there is a trade-off between the complexity of this algorithm versus the resulting performance of this algorithm.

Chapter 5: METHODOLOGY

The principle purpose of this chapter is to describe the detailed procedure of implementation of our proposed K-GA algorithm to solve the gateway location problem. This chapter also presents a methodology of the several distribution scenarios. To make the comparison process for K-GA more robust and to validate the effectiveness of our proposed algorithm, some existing heuristics approaches are studied and discussed in this chapter.

5.1. Implementation of K-GA Heuristic

This section explains the details of the proposed K-GA algorithm implementation including the phase procedures. As mentioned in Section 4.3.1, the K-GA algorithm is implemented in three successive phases in order to simplify the combination of both ML and AI approaches. To explain illustrative details of proposed algorithm, we consider a small topology of 22 nodes as an example in this section. The explanation of example is given as follows:

Step 1. Network Topology Generation

The locations of small cells are uniformly generated using homogenous Poisson Point Process scheme on a 100 meters radius circle. We generated 22 nodes. Among those 22 nodes 20 are to be considered as small cells and 2 nodes are taken as gateways. Small cells are considered to be 20 meters apart from each other in this example. The following Fig. 5.1 shows the generated network topology.

Step 2. Connectivity Graph

As mentioned earlier, one of the assumptions of our network model is that small cells can transmit and communicate with other small cells within their transmission range. Therefore, we verify our network by checking the connectivity of the network topology. First, we generate the neighbor table of the small cells. A small cell adds other small cells in its transmission range as its neighbors in its neighbor table once it receives their “Hello messages”. In this example we consider transmission range of 50 meters. Table 5.1 shows the neighbor tables of small cells. Fig. 5.2 represents the connectivity graph of the network. In case of disconnected node(s) in the network, that topology is discarded and replaced.

Table 5.1. Neighbor Table

Node	Neighbor Node	Distance	Node	Neighbor Node	Distance	Node	Neighbor Node	Distance
1	3	23	8	12	49	15	14	48
1	8	38	8	14	20	15	21	32
1	9	29	8	21	42	16	2	37
1	12	39	9	1	29	16	10	27
1	14	35	9	2	34	17	4	24
1	21	39	9	21	40	17	6	25
1	22	23	9	22	35	18	11	27
2	9	34	10	2	37	18	19	50
2	10	37	10	16	27	18	20	28
2	16	37	11	12	40	18	22	40
3	1	23	11	18	27	19	11	26
3	8	25	11	19	26	19	18	50
3	12	25	11	22	45	20	18	28
3	14	35	12	1	39	20	22	46
3	22	34	12	3	25	21	1	39
4	6	41	12	6	36	21	8	42
4	7	28	12	8	49	21	9	40
4	17	24	12	11	40	21	14	23
5	13	39	12	22	33	21	15	32
6	4	41	13	5	39	22	1	23
6	12	36	13	15	44	22	3	34
6	17	25	14	1	35	22	9	35
7	4	28	14	3	35	22	11	45
7	8	41	14	8	20	22	12	33
8	1	38	14	15	48	22	18	40
8	3	25	14	21	23	22	20	46
8	7	41	15	13	44			

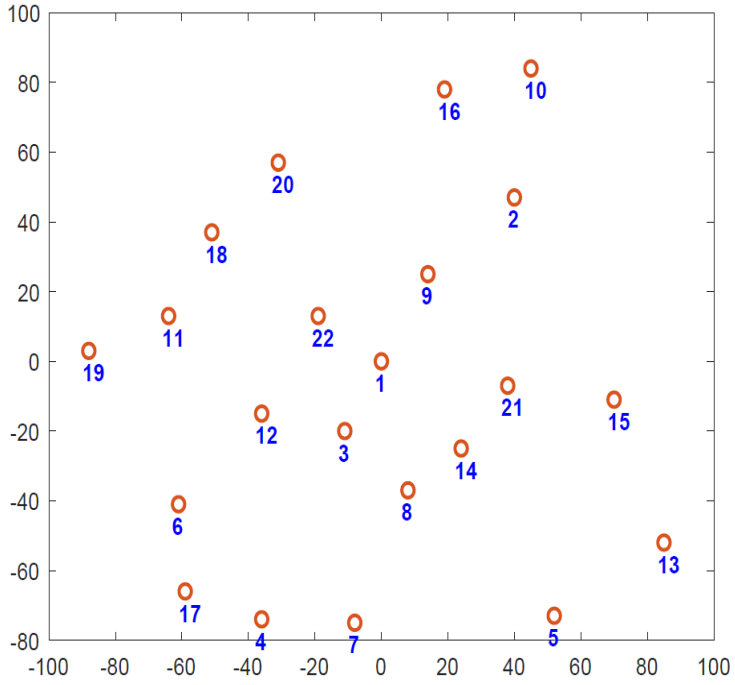

Fig. 5.1. Network Topology with Node ID

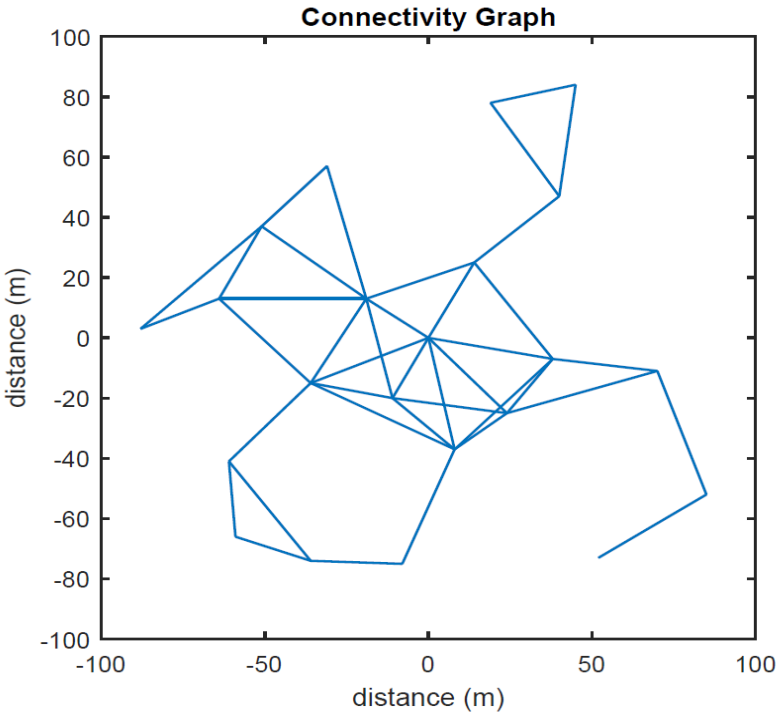

Fig. 5.2. Connectivity Graph

Step 3. *K*-Means Algorithm

Once we have connected network, we run our proposed K-GA algorithm on network topology with some input parameters. Table 5.2 represents input parameters for K-GA in this example.

Table 5.2. Input Parameters

Parameters	Values
Number of Total Small Cells	22
Number of Gateways	2
Population Size	6
Crossover Type	Single Point, Two-Point
Mutation Probability	1%
Maximum No. of Generations	1

As explained earlier in chapter 4, for our 1st phase of K-GA we run *K*-means algorithm and generate *M* centroids. For this example, two gateways are considered so at the end of first phase two centroids will be generated and small cells are assigned to two centroids based on the squared Euclidean distances. Two centroids are generated using *K-means* algorithm at following locations:

	X	*Y*
Centroid 1	-15.54	31.09
Centroid 2	9.27	-43.72

Next, *K*-means assigns node 1, 2, 9, 7, 10, 11, 12, 16, 18, 19, 20, 22 to *centroid 1* and generates first cluster. Similarly, it also assigns nodes 3, 4, 5, 6, 7, 8, 13, 14, 15, 17, 21 to *centroid 2* and generates another cluster. Fig. 5.3 highlights the two clusters in different colors and '+' signs are the centroids of the network.

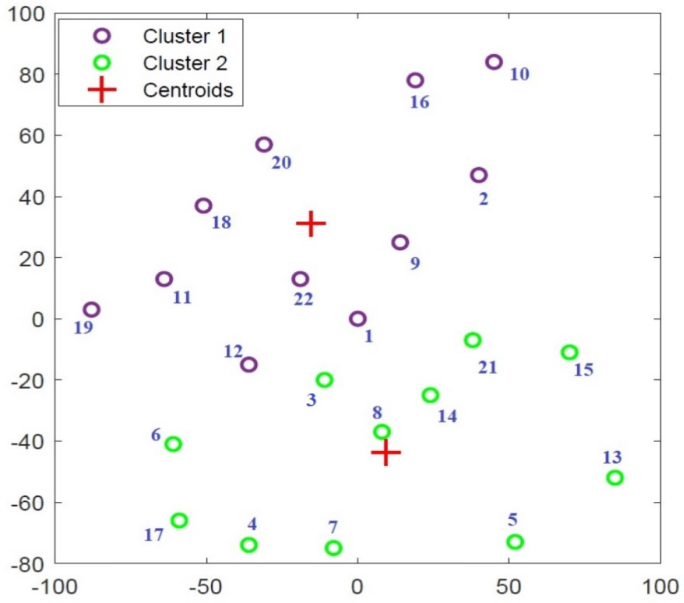

Fig. 5.3. *K*-means output

Step 4. Generate Initial Population for K-GA

As mentioned earlier, to generate chromosomes for K-GA, we select nearest small cells to the centroids and consider them as potential gateways. In this example to generate six chromosomes, *three* nearest nodes (*9, 20* and *22*) are selected from *centroid 1* and *two* nearest nodes (*8* and *14*) are selected from *centroid 2*. These *five* [*2+3*] total nodes create six different combinations (i.e. 3 × 2 = 6) and using these six different combinations, six chromosomes will be generated. Table 5.3 displays six chromosomes where '0' represents small cells and '1' represents gateways. Note that all chromosomes' length is equal to the total number of nodes.

Table 5.3. Number of Chromosomes

Node ID / Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
5	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0

Step 5. Fitness Function

As explained in Chapter 4, the fitness function calculates the average number of hops by implementing the shortest path algorithm to assign small cells to gateways and generate shortest path trees. In this step of K-GA, Dijkstra's shortest path algorithm is used to calculate the fitness values of all chromosomes. For each chromosome gateway node locations are considered as source nodes for the shortest path algorithm. Our fitness function is divided into two parts.

A. First Part: Generate Shortest Path Trees

In the first part of the function it calculates the shortest path trees in terms of hops from each given gateway location to all other nodes (small cells). To calculate shortest path trees, first it converts the distances into hops to unweighted links of length 1 and after that it calculates shortest hop routes from one source node (i.e. gateway location) to every other node. In Table 5.3, *Chromosome 1* has node 8 and node 22 as gateways. As a result, indexes [8, 22] are used as input parameters for the shortest path algorithm. Using these input parameters our algorithm first generates shortest hop trees with node 8 to all other small cells and similarly with node 22 to all other nodes. Table 5.4 shows the hop counts from node 8 and node 22 to all other nodes. Fig. 5.4 and Fig. 5.5 display the shortest routes from node 8 and node 22 to all other nodes, respectively.

Table 5.4. Hop Counts from Gateway to Other Nodes

Node ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**
Node 8	1	3	1	2	4	2	1	0	2	4	2	1	3	1	2	4	3	3	3	3	1	2
Node 22	1	2	1	3	5	2	3	2	1	3	1	1	4	2	3	3	3	1	2	1	2	0

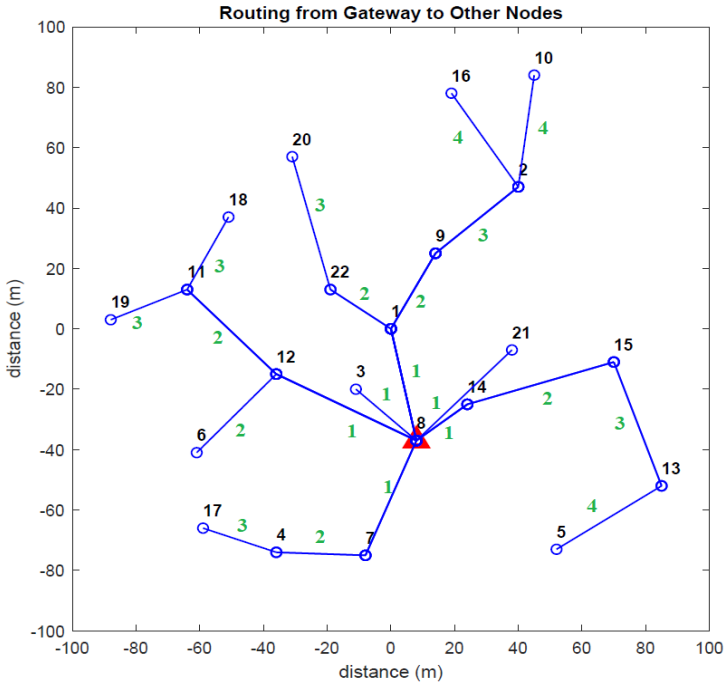

Fig. 5.4. Shortest Route from Node 8

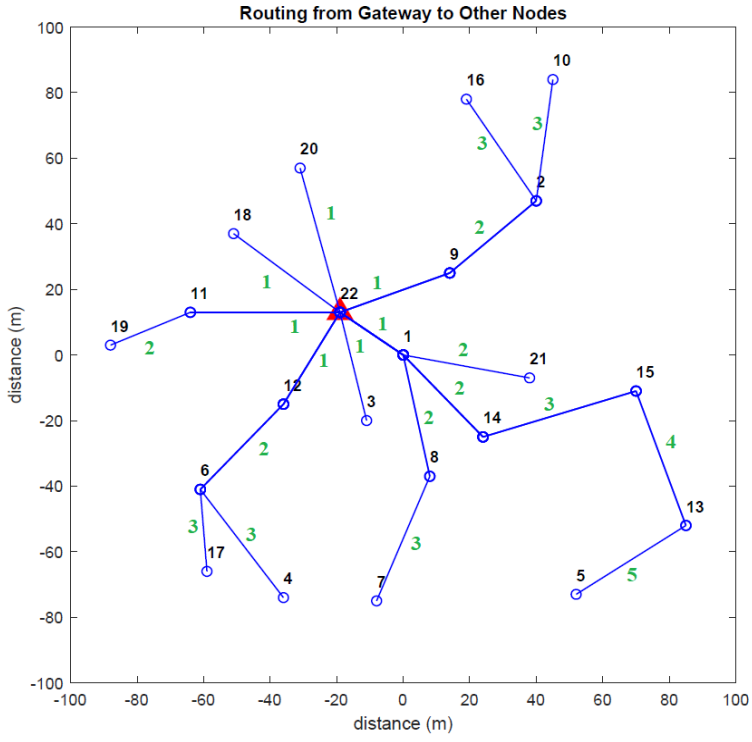

Fig. 5.5. Shortest Route from Node 22

Same procedure will repeat for all six chromosomes and total 12 shortest route trees will be calculated based on gateway locations. To make our algorithm more efficient, we

are storing adjacency list of each gateway locations with shortest routes trees in global variable to avoid regenerating it when it necessary. This list can be easily available to use for second part of our fitness function.

B. Second Part: Calculate ANH

In second step of fitness function, assignment of small cells to all available gateways will take place and based on these assignments average number of hops will be calculated. Small cells allocation to the gateways depends on the minimum hop counts. In *chromosome 1*, 20 small cells are distributed between two gateways based on the shortest hop counts and is shown in Fig. 5.6.

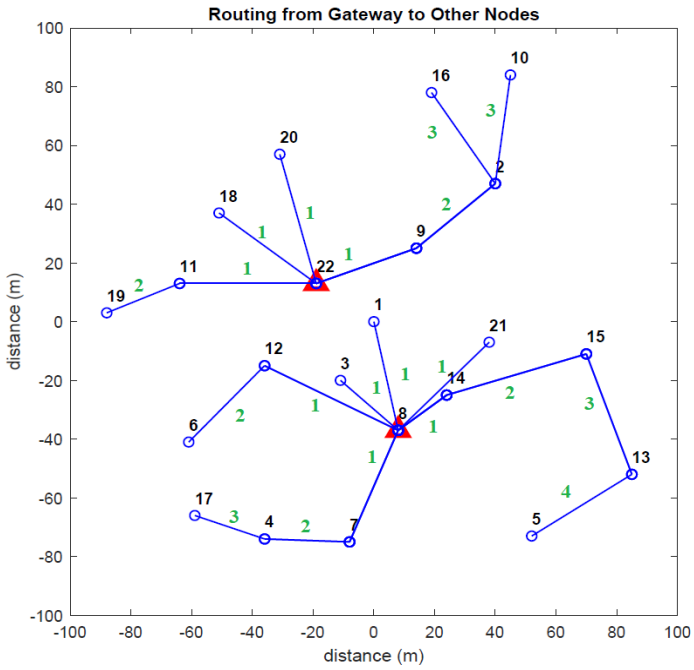

Fig. 5.6. Shortest Route from Node 8 and Node 22

Next is to calculate total minimum hops from gateways to small cells and using formula (4.9) from Chapter 4, average number of hops is obtained. Based on above routing results, total minimum hops for *chromosome 1* is 36 and ANH is 36/(22-2) = 1.8. Similarly, fitness value as ANH is calculated for all six chromosomes. To make our algorithm fast and more efficient, we design *Map object* data structure which adds new key (gateway locations) and value (ANH) in map object for every unique key. If same gateway location combinations repeat from other chromosomes in any generation for same network

topology, *Map object* allows us to retrieve fitness values using corresponding gateway locations. Table 5.5 represents the fitness value for all six chromosomes.

Table 5.5. K-GA Fitness Values

	Gateway 1	Gateway 2	Fitness value
Chromosome 1	22	8	1.80
Chromosome 2	9	8	1.85
Chromosome 3	20	8	2.05
Chromosome 4	22	14	1.75
Chromosome 5	9	14	1.95
Chromosome 6	20	14	2.15

Step 6. Selection Process

In this step, selection of chromosomes using Roulette Wheel Selection Method is explained. The detail explanation for roulette wheel method is as follows:

Roulette wheel selection method selects the fittest individuals for further operations. In order to achieve that in our implementation, first we inverse the fitness value so that a higher value represents better fitness. Table 5.6 shows the inverse fitness for six chromosomes.

Table 5.6. Inverse Fitness for Roulette Wheel Method

	Inverse fitness value (f_i) = 1/Fitness value
Chromosome 1	0.5556
Chromosome 2	0.5405
Chromosome 3	0.4878
Chromosome 4	0.5714
Chromosome 5	0.5128
Chromosome 6	0.4651
Total Fitness	**3.133**

Next is to calculate the fitness probability (p_i) and cumulative probability (q_i) of each chromosome using (5.1) and (5.2):

$$p_i = \frac{f_i}{\sum_{k=1}^{n_{pop}} f_k}, \quad (5.1)$$

$$q_i = \sum_{k=1}^{i} p_k, \quad (5.2)$$

where n_{pop} is the number of chromosomes, f_i is the fitness value of i^{th} chromosome and q_i is the cumulative probability for i^{th} chromosome. Table 5.7 shows the fitness probability and cumulative probability of chromosomes.

Table 5.7.Fitness Probability and Cumulative Probability

	Fitness Probability(f_i)	**Cumulative Probability(q_i)**
Chromosome 1	0.1773	0.1773
Chromosome 2	0.1725	0.3498
Chromosome 3	0.1557	0.5055
Chromosome 4	0.1824	0.6879
Chromosome 5	0.1637	0.8516
Chromosome 6	0.1484	1.0000

Thereafter, is to generate random number r between 0 and 1. Now to select the chromosomes, if $r \leq q_1$ (cumulative probability of 1st chromosome) then select *chromosome 1* else select individuals such that $q_{i-1} < r \leq q_i$. Repeat this step n_{pop} times to select n_{pop} chromosomes. Table 5.8 shows the random numbers and selected chromosomes based on above explained selection criteria. After selection process the population is enriched with better individuals.

Table 5.8. Selected Chromosomes

Random Number	**Cumulative Probability (q_i)**	**Selected Chromosome**
0.5233	0.6879	Chromosome 4
0.4279	0.5055	Chromosome 3
0.0500	0.1773	Chromosome 1
0.0054	0.1773	Chromosome 1
0.6681	0.6879	Chromosome 4
0.2429	0.3498	Chromosome 2

Step 7. Crossover Operation

In our K-GA algorithm, we implemented both single point crossover and two-point crossover method. In Section A.1.3 of the Appendix A, we experiment using single point crossover and two-point crossover with few node densities and select one method which performs fast and efficiently.

A. Single Point Crossover

It is a simple method to do crossover operation where two random chromosomes are selected from mating pool and random cross-site or crossover point is generated between 1 to N (total number of nodes) and the bits are swapped between *chromosome 1* & *chromosome 2* from crossover point. It randomly selects chromosomes, generates random crossover point, and repeats $n_{pop}/2$ times.

Table 5.9. Chromosomes Pairs for Crossover

Node ID / Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
(4,1)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
(3,4)	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
(1,2)	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.10. Single Point Crossover

Node ID / Crossover point	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
21	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
13	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

In this case total 6 chromosomes are there, so 3 times above process is repeated. Table 5.9 shows the randomly generated pairs for crossover operation. Table 5.10 shows the random point for crossover and highlights the swapping parts from crossover point.

B. Two-Point Crossover

In a two-point crossover, two random points will be generated and bits between these crossover points will be exchanged. Similarly, as presented in Table 5.9 random chromosomes will be selected for two-point crossover method as well. Table 5.11 represents the two random crossover points and highlights the bit exchanges between them.

Table 5.11. Two Point Crossover

Node ID / Crossover Point	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
5,18	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
9,13	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
2,8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Step 8. Repair Procedure for Crossover Function

As explained earlier, repair procedure is necessary to maintain number of gateway locations per topology. As we can see in Table 5.10 after single point crossover operation *chromosomes 1 to 4* have two number of gateways but *chromosome 5* is left with only 1 gateway at node 8. Therefore, we convert randomly any bit '0' to '1' and add 1 gateway to the *chromosome 5*. In this case, at node 15 gateway is added. Similarly, *chromosome 6* have 3 gateways in total at node 8, node 9, and node 22. So, we reduce one gateway by randomly converting a '1' to '0'. Here, at node 22 gateway is removed. Table 5.12 shows final chromosomes after crossover and repair process. Repair procedure for both crossover methods is similar. Only repair procedure for single point crossover with example is explained in this section.

Table 5.12. Chromosomes after Crossover Repair Process

Node ID / New Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Step 9. Mutation Process

The mutation process works on the probability of mutation. The higher the probability, the higher the chances of bit to mutate.

- Mutation function generates *N* times random number (*R*) between 0 to 1.
- After that it checks that $R_N <= P_m$, where *N* is the number of nodes in one chromosome.
- Let's say if we have mutation probability (P_m) 1% (0.01) and we get random number $R_1 = 0.22$ for 1st bit of *1st chromosome* then mutation function will not mutate 1st bit of *1st chromosome*. Next, we generate another random number for second bit, and we get $R_2 = 0.01$, then mutation function mutates the 2nd bit of *1st chromosome*. Same procedure repeats for all bits (*N* nodes) in a chromosome. Mutation function checks for all six chromosomes with the same steps as explained above.

Table 5.13. Chromosomes after Mutation Process

Node ID / New Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.13 represents the chromosomes after mutation process. In *2nd chromosome* bit 11 changed from 0 to 1 and in *6th chromosome* bit 9th changed from 1 to 0.

Step 10. Repair Procedure after Mutation Function

In Table 5.13 *chromosome 2* has more gateways and *chromosome 6* has less gateways so repair procedure is necessary after mutation function. Repair procedure for mutation function is similar to crossover function as explained in step 8. Table 5.14 shows chromosomes after mutation repair process.

Table 5.14. Chromosomes after Mutation Repair Process

Node ID / New Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Step 11. Calculate Fitness

This step calculates the fitness value after repair procedure of mutation process of all chromosomes as explained in step 5. Table 5.15 shows the fitness value of all chromosomes. It finds and saves the chromosome with the best fitness of least average number of hops in this generation.

Table 5.15. K-GA Fitness Value

	Gateway 1	Gateway 2	Fitness value
Chromosome 1	14	22	1.75
Chromosome 2	8	11	2.0
Chromosome 3	14	22	1.75
Chromosome 4	8	20	2.05
Chromosome 5	8	15	2.10
Chromosome 6	20	16	2.75

We replace the current population with new population and repeat the process from step 5 to step 11 for given number of generations. Finally, we extract and output the gateway locations and average number of hops from the saved best chromosome. Here in this example, we have given 1 generation so our final best output in terms of average number of hops is *1.75*. We also see the effects of different genetic parameters on our K-GA algorithm and details on tuning of GA parameters are found in Appendix A.

5.2. Implementation of Different Distribution Scenarios

We consider a UDN in a circular area with a radius of 1000 m for all three distribution scenarios. Small cells are placed using a homogenous Poisson Point Process (PPP) scheme [82]. The number of small cells is a Poisson random variable with mean λ , where λ represents the average number of small cells in the circular area. In the circular area, our network topology is simulated in three different distribution scenarios, namely, Uniform distribution, bivariate Gaussian distribution, and Cluster distribution. The transmission range for all small cells is considered as 200 m in all distributions. Small cells are placed apart from each other to avoid complete overlap of their backhaul coverage areas. While generating a network distribution, we enforced restrictions on network topologies to maintain a certain minimum inter-small cell distance.

5.2.1. Uniform Distribution:

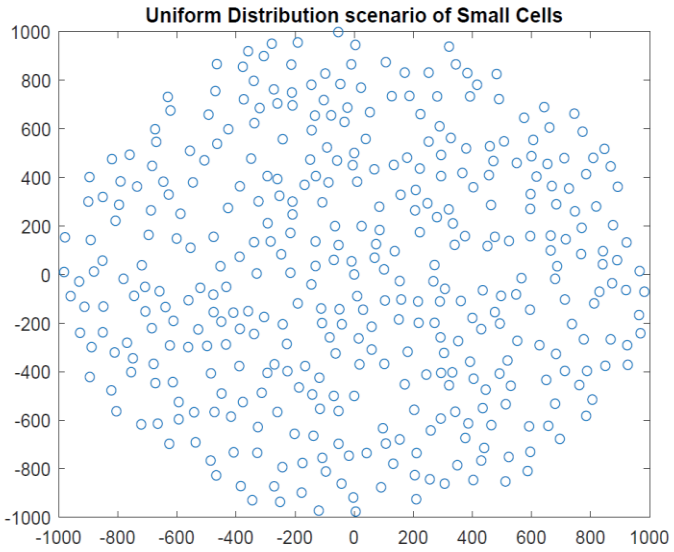

Fig. 5.7. UD Network Topology

In the circular area, the locations of small cells are generated using *Uniform random distribution*, i.e., nodes are distributed uniformly and randomly in all directions. Uniform distribution is a widely used model because of its simplicity and systematic flexibility. Small cells are separated by a minimum distance of 50 meters to avoid total overlap of backhaul coverage area between adjacent cells. An example of a topology created using Uniform random distribution is shown in Fig. 5.7.

5.2.2. Bivariate Gaussian Distribution:

The locations of small cells are distributed in the circular region according to a symmetric bivariate Gaussian (2D Gaussian) with a peak at the area center. The Gaussian distribution is another common type of distribution which extends the scope of simulation environment to be more practical and realistic. Example of this is to place small cells in the city where the downtown (center of the city) has more users compared to the outer regions of the city. This scenario follows a Gaussian distribution where high probability of users' presence occurs around the city center and less probability occurs at city outskirts.

The bivariate Gaussian distribution (or Normal Distribution) is determined by its parameters μ and σ^2, which are the expected value and variance for a normal random variable [83]. In a Normal distributed network, at center more small cells are deployed with mean value 0, which means maximum value occurs at $\mu = 0$ and the standard deviation of Gaussian distribution is normalized to a value of 0.40 to provide more narrower and denser distribution at center. 0.40 value gives more realistic scenario where small cells are denser at center and distributed properly within 1000 m radius. The theoretical distribution of Gaussian extends to infinity, but we are limiting the area to a radius of 1000. So basically, we are generating a truncated 2D Gaussian distribution. If the radius is large enough compared to the standard deviation, the edge effect is negligible (i.e. the probability of generating a point outside the service area is very small). Small cells are separated by a minimum distance of 40 meters to avoid a complete overlap among adjacent cells. A topology generated using bivariate Gaussian distribution is shown in Fig. 5.8.

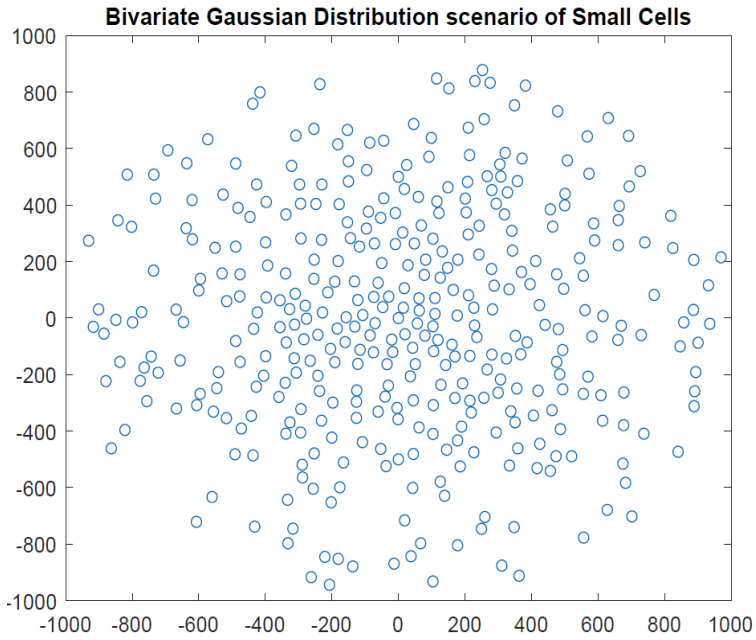

Fig. 5.8. GD Network Topology

5.2.3. Cluster Distribution:

We generated a UDN in the circular area using clustered network distribution to generate a more realistic environment for our experiments. Nodes are clustered at specific regions which represent users gathering at hotspots, and these places may be offices, university, or shopping malls. To this end, we have proposed the clustered based distribution topology, which is the combination of group of small cells (clusters) and uniformly distributed small cells to provide more closer scenario to real world since more users are clustered at particular places (i.e. offices/shopping malls etc.) but few other users are also present outside of such regions. Small cells are usually deployed based on traffic demands per unit area.

Clustered based distribution topology is constructed in mainly two phases: In first phase we generate clusters in a 500 m radius circle area within the original circular area. To get first cluster's center coordinate, random angle θ on the 500m radius circle is generated and first center's coordinates are derived using:

$$x = radius * cos(\theta), \tag{5.3}$$

$$y = radius * sin(\theta) \tag{5.4}$$

Around first cluster center, we generate locations of small cells within 100 m radius using bivariate random Gaussian (2D Gaussian) with a peak at the cluster center and standard normal parameters $\mu = 0$ and $\sigma^2 = 1$. Small cells in a cluster are separated by a minimum distance of 25 m. For next cluster design we change θ using:

$$\text{New } \theta = \theta + \frac{360^\circ}{Number\ of\ clusters\ needed\ in\ a\ topology} \qquad (5.5)$$

Using equation 5.3 and 5.4, new cluster center coordinates are generated. As explained earlier random number of small cells around the new cluster center are generated using a "normal distribution". In first phase, above process is repeated for the maximum number of clusters required in topology. Number of nodes in each cluster is different. A topology with Cluster distribution and 3 clusters is shown in Fig. 5.9.

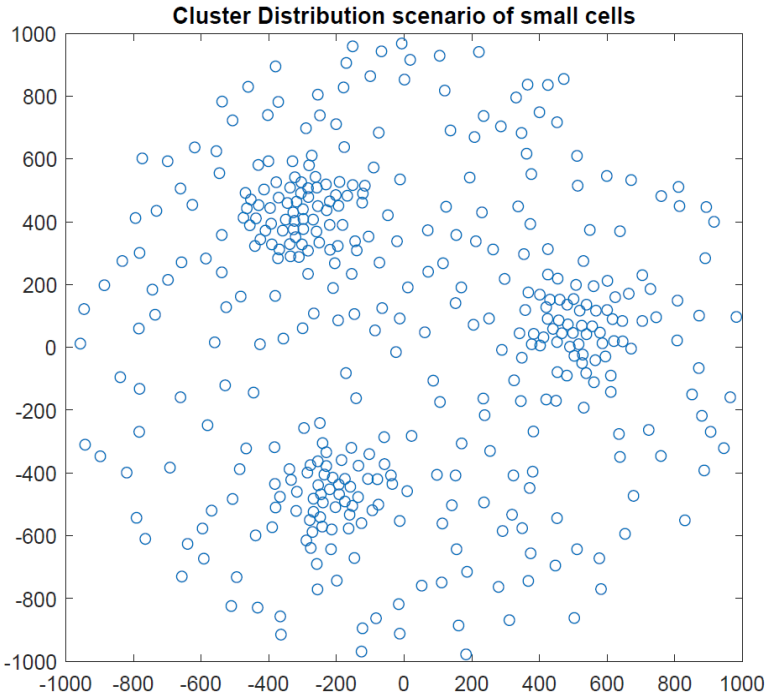

Fig. 5.9. CD Network Topology with 3 clusters

In second phase of CD topology, remaining nodes are generated using Uniform random distribution. For uniformly generated nodes, minimum distance of 50 meters is considered to separate the small cells and to prevent overlap between neighboring cells.

This is an example of a clustering model where several small cells are normally distributed to create clusters while the remaining small cells are uniformly distributed within the simulation region to illustrate more realistic scenarios.

5.3. Implementation of Comparison Methods

In this section we describe other heuristics and approaches employed to find gateway locations to calculate average number of hops and backhaul network capacity. We implemented these different methods to compare with our proposed K-GA heuristic. We compare K-GA with genetic algorithm, *K*-means, *K*-medoids, and baseline approach. We have also considered the combination of *K*-medoids and genetic algorithm (*KM-GA*) to validate the performance of K-GA.

5.3.1. Genetic Algorithm:

We compare our proposed K-GA with pure genetic algorithm where initial population is generated randomly. Implementation is identical to the GA stage in the K-GA algorithm except for the generation of the initial population, which is done by randomly generating *M* 1s in each chromosome. Using sample topology generated in section 5.1 and using same parameters, a generation of initial population is given in Table 5.16.

Table 5.16. Initial Population

Node ID / Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
4	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Similar to section 5.1, genetic algorithm works for fitness function, selection process, crossover function, and mutation function for the required number of generations. After mutation function, final fitness value is evaluated in Table 5.17.

Table 5.17. GA Fitness Value

	Gateway 1	Gateway 2	Fitness value
Chromosome 1	9	20	2.2
Chromosome 2	9	13	2.05
Chromosome 3	5	20	2.6
Chromosome 4	10	21	2.15
Chromosome 5	10	22	2.0
Chromosome 6	10	22	2.0

In the Final step of genetic algorithm, we extract and output the gateway locations and average number of hops from the saved best chromosome. Here in this example, we have considered 1 generation so our final best output in terms of average number of hops is *2.0*.

5.3.2. *K*-means Clustering Algorithm:

Based on the final output of *M* centroids of *M* clusters from the *K*-means algorithm, we choose the small cell nearest to each centroid as a Gateway. This set of *M* GW locations is used as input to the shortest path algorithm for calculating the ANH. For explanation of *K*-means algorithm, same small network topology and parameters are considered as explained in section 5.1. For this example, two gateways are considered, so at the end of *K*-means output two centroids will be generated and remaining small cells assigned to these two centroids based on the squared Euclidean distance. Two centroids at the output of *K-means* algorithm are obtained at the following locations:

	X	*Y*
Centroid 1	-6.56	-29.18
Centroid 2	6.00	54.66

Next, *K*-means assigns node 1, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 15, 17, 19, 21, and 22 to first cluster. Similarly, it also assigns nodes 2, 9, 10, 16, 18, and 20 to another cluster. Fig. 5.10 highlights the two clusters in different colors and '+' signs indicates the centroids. Next, we calculate the average number of hops using nearest small cell to the centroid as

GW. In this case, using *node 3* and *node 16,* we get ANH value of *1.95* as final output of *K*-means algorithm.

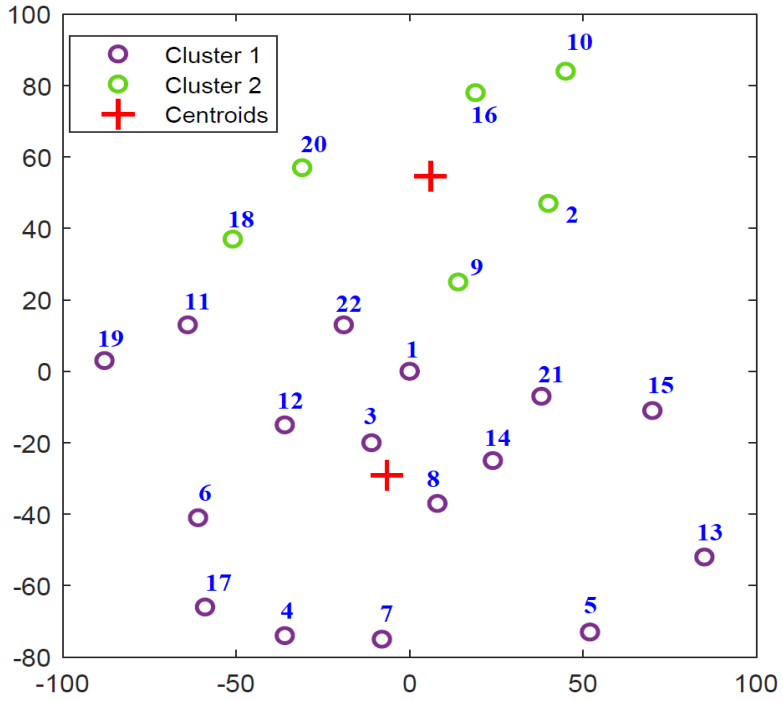

Fig. 5.10. *K*-means Algorithm

5.3.3. *K*-medoids Clustering Algorithm:

As explained in section 2.10, *K*-medoids selects medoids as potential gateways and best result from all replications is selected as the final result of *K*-medoid clustering algorithm. The final *M* small cells are selected as gateway locations for calculating ANH. Identical to previous sections, same small network topology is considered to understand the implementation of *K*-medoids Algorithm. At the end of *K*-medoids algorithm, two small cells will be output, and other remaining small cells will be assigned between these two small cells (or medoids) based on the squared Euclidean distances. This gives two nodes, *node 2* and *node 3* using *K*-medoids algorithm at following locations:

	X	*Y*
Node 3	-11	-20
Node 2	40	47

Next, the algorithm assigns node 1, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 17, 18, 19, 21, and 22 to first cluster. Similarly, it also assigns nodes 2, 9, 10, 15, 16, and 20 to another cluster. Fig. 5.11 highlights the two clusters in different colors and '+' signs indicate the two medoids. Next the algorithm calculates the average number of hops using nearest small cells to the medoid. In this case, using *node 2* and *node 3,* we get ANH value of *1.90* using *K*-medoids algorithm. Fig. 5.11 represents the output of *K*-medoids algorithm.

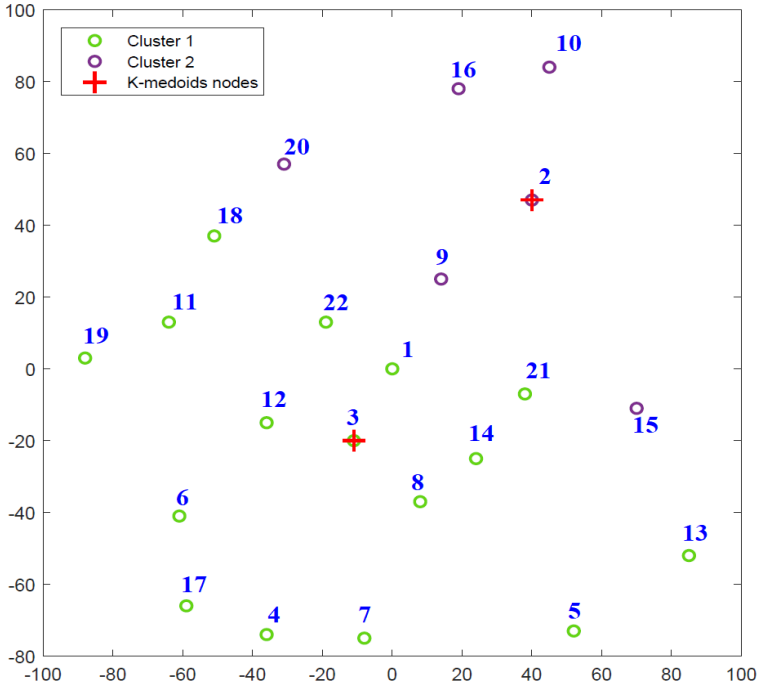

Fig. 5.11. *K*-medoids Algorithm

5.3.4. *K*-medoids and Genetic Algorithm (KM-GA):

To compare the effectiveness of our K-GA algorithm, we have also implemented the combination of *K*-medoids and the GA. Implementation of KM-GA is identical to the K-GA algorithm explained in Chapter 4 except for the 1st phase, where an actual small cell is used as the cluster center. Using the *K*-medoids result, we select t_1-1, t_2-1, t_3-1, …, t_M-1 small cells nearest to the *M* medoids and the *M* medoids to generate the initial population. The complexity of KM-GA is $O(N^2R + NG\,n_{pop})$, where *N* is the total number of small cells, *G* is the number of generations for GA, and n_{pop} is the number of chromosomes. *R* is the number of replications for *K*-medoids.

To illustrate the implementation for KM-GA same small network topology and parameters are considered as explained in section 5.1. In 1st phase of KM-GA, we run *K*-medoids algorithm and obtain *M* nodes. For this example, two gateways are considered so at the end of first phase, two points will be obtained and remaining small cells are assigned among these two small cells (or medoids) based on the squared Euclidean distance. The algorithm gives two nodes, *node 22* and *node 8* at following locations:

	X	*Y*
Node 22	-19	13
Node 8	8	-37

Next, it assigns node 1, 2, 6 , 9, 10, 11, 12, 16, 18, 19, 20,and 22 to first cluster. Similarly, it also assigns nodes 3, 4, 5, 7, 8, 13, 14, 15, 17, and 21 to another cluster. Fig. 5.12 highlight the two clusters in different colors and '+' signs indicate the two medoids.

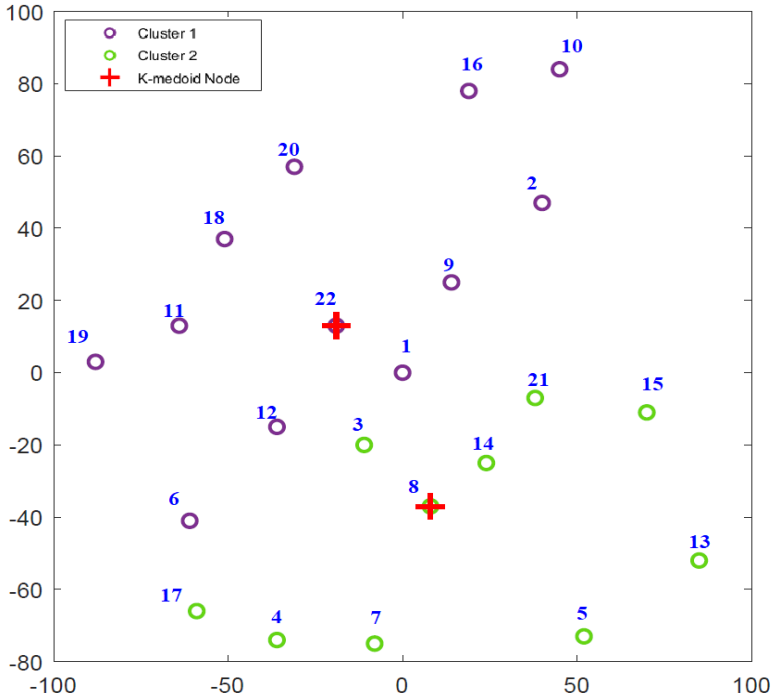

Fig. 5.12. KM-GA Algorithm

Next step is to generate initial population from *K*-medoids output and this process is similar as explained in step 4 in section 5.1. To generate six chromosomes, KM-GA chooses *three* nodes (*3, 8* and *14*) which includes the medoid and nearest nodes to the

medoid of *cluster 1* and *two* nodes (*1* and *22*) including the medoid and nearest node of *cluster 2.* Table 5.18 represents the initial population of the KM-GA algorithm.

Table 5.18. Number of Chromosomes

Node ID / Chromosomes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
3	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
6	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

After generating initial population of KM-GA, implementation is similar to section 5.1 for fitness function, selection process, crossover function, and mutation function for the required number of generations. After mutation function, final fitness value is given in Table 5.19.

Table 5.19. KM-GA Fitness Value

	Gateway 1	Gateway 2	Fitness value
Chromosome 1	3	22	1.95
Chromosome 2	8	22	1.80
Chromosome 3	1	3	2.10
Chromosome 4	1	8	2
Chromosome 5	3	14	2.15
Chromosome 6	1	22	1.90

Finally, we extract and output the gateway locations and average number of hops from the saved best chromosome. Here in this example, for 1 generation our final best output in terms of average number of hops is *1.80.*

5.3.5. Baseline Method:

In the baseline method, gateways are placed at fixed locations at equal distances around a 500 m radius circle within the original circular area for Uniform random

distribution and bivariate Gaussian distribution. For Cluster distribution, gateways are considered as small cells nearest to the cluster centers. From C small cells nearest to cluster centers, M gateways are taken. Here C is the total number of clusters present in the topology and $C \geq M$ .These gateway locations are provided as input to Dijkstra's shortest path algorithm to obtain the ANH.

Chapter 6: EXPERIMENTAL RESULTS AND EVALUATION

In this chapter, several scenarios are considered to make the comparison process more robust and to illustrate that our proposed algorithm performs better than the other approaches in all distribution environments. Section 6.1 represents the simulation environment for execution and states the essential performance parameters. In Section 6.2, we compare the performance of K-GA with the optimal solution, *baseline* scenario, *GA, K-means, K*-medoids, and *KM-GA* in terms of average number of hops and backhaul network capacity. In section 6.3, we analyze the effect of different number of gateways ranging from 2 to 8 on the performance of the ultra-dense network and study its effect on the backhaul network capacity.

6.1. Simulation Environment

A. Software Used

MATLAB version R2019b [81] has been used to implement the heuristic algorithms K-GA, KM-GA, *K*-means, *K*-medoids, and baseline method. Mathematical formulation for GLP is implemented in IBM ILOG CPLEX Optimization Studio using OPL language. IBM's CPLEX solver version: 12.10.0.0 has been used to solve the problem.

B. Hardware Used

The software programs including MATLAB, OPL, and CPLEX were run on a personal laptop having Intel Core i7-8550U CPU processor running at 1.80 GHz, and 16 GB of memory. The operating system was Microsoft Windows 10 Home.

6.1.1. Key Parameters:

We tested different node densities of small cells in a circular area having a radius of 1000 meters for Uniform distribution, bivariate Gaussian distribution and Cluster distribution. They range from 110, 130 and so on until 470. For each node density in all distribution scenarios, we generate 100 different topologies as a part of our Monte Carlo simulation setup. In order to evaluate all scenarios, we generate 2700 different network topologies and run simulations for all different approaches separately on all 2700 topologies. We calculate the mean value of the ANH and the BNC for the 100 topologies for each node

density in different distribution scenarios and compute the 95% confidence interval (CI) as well. We calculate the backhaul network capacity using (3.1) where W_S is 1 Gbps and W_G is 100 Gbps [44][84]. In Section 6.2 we evaluate all methods and all distribution scenarios using 4 gateways for all node densities and for section 6.3 we consider different number of gateways ranging from 2 to 8 to evaluate the performance of the ultra-dense network.

Parameter settings are as follows:

- *K-means*: 100 replications for each node density.
- *K-medoids*: 100 replications for each node density.
- *GA*: 100 generations for each node density. The population size is 300 for all node densities. 1% mutation probability and single-point crossover is applied.
- *K-GA*: Based on the analysis explained in Appendix A, 50 replications of *K*-means and 50 GA generations for each node density are considered. A population size of 256 is used for all node densities because a constant size initial population is generated from the *K*-means stage. 1% mutation probability and single-point crossover type is considered.
- *KM-GA:* 50 replications of *K*-medoids and 50 GA generations for each node density are considered. Population size of 256, 1% mutation probability and single-point crossover is used.
- *Baseline*: On the 500 m circle for UD and GD, the 4 gateway locations are generated at the following coordinates: (294, 405), (-294, 405), (-294, -405), (294, -405). These gateway locations remain the same for all topologies for all node densities. For CD, gateways are taken as the small cells nearest to the cluster centers.

6.2. GLP in Ultra-Dense Networks

6.2.1. Results of Uniform Distribution:

Our proposed heuristic K-GA uses the *K*-means algorithm along with the genetic algorithm. However, we also implemented the combination of *K*-medoid algorithm with genetic algorithm to confirm the effectiveness of our K-GA algorithm. In this section we present two different analysis in Uniform distribution scenarios. First, we compare the K-GA algorithm with the KM-GA algorithm and subsequently we compare our K-GA algorithm with five other methods which were explained in section 5.3.

A. Comparison of K-GA and KM-GA

To compare K-GA with KM-GA, we consider 6 different node densities which are 150, 190, 230, 270, 310, and 350. We evaluate and compare the results for both algorithms based on their mean value of ANH and their runtimes (or execution times). Fig. 6.1 represents the graph of ANH for K-GA and KM-GA. It can clearly be seen from this graph that K-GA performs better compared to KM-GA for all six node densities. In K-GA, the nearest small cells based on *M* centroids are chosen to generate the initial population for GA. Centroids provide more accurate locations for initial population of GA, which helps K-GA to perform better. In KM-GA actual small cell (medoid) is used as a reference and the nearest small cells around the medoid as well as the medoid itself are selected to generate initial population. The initial population generated by medoid itself and small cells near to medoids are less diverse and may restrict the search space compared to small cells near to centroids. This behavior increases the chance to result in poor performance of KM-GA compared to K-GA. Table 6.1 represents the fitness value and time analysis for K-GA versus KM-GA. It can be clearly seen from the time analysis graph that KM-GA takes more time to be completed compared to the K-GA algorithm. There is a consistent difference in the value of average number of hops between both the schemes for all node densities. Based on the results of ANH, computational complexity and runtime for both K-GA and KM-GA, K-GA proves to be better compared to KM-GA.

Table 6.1. ANH and Time analysis of K-GA versus KM-GA

Small Cell Density	K-GA				KM-GA			
	ANH Mean	ANH 95% CI	Time Mean	Time 95% CI	ANH Mean	ANH 95% CI	Time Mean	Time 95% CI
150	2.70	2.67—2.73	7.83	7.78—7.89	2.71	2.68—2.73	10.07	9.95—10.19
190	2.61	2.59—2.63	8.62	8.55—8.68	2.62	2.60—2.64	11.03	10.86—11.21
230	2.51	2.50—2.53	9.85	9.77—9.93	2.52	2.50—2.53	12.23	12.08—12.37
270	2.46	2.45—2.47	10.69	10.61—10.77	2.47	2.46—2.48	13.12	12.98—13.26
310	2.44	2.43—2.45	11.91	11.73—12.10	2.45	2.44—2.46	13.46	13.30—13.61
350	2.41	2.40—2.42	12.76	12.65—12.87	2.42	2.41—2.43	15.56	14.81—16.31

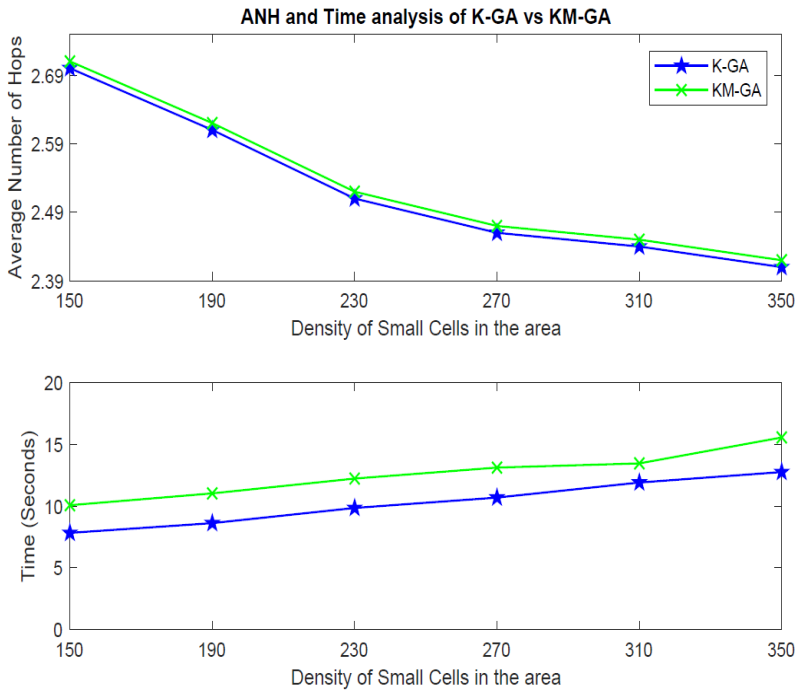

Fig. 6.1. ANH and Time analysis for K-GA and KM-GA

B. Comparison of K-GA with 5 Different Methods

The simulation results obtained in this section are to demonstrate the performance of our proposed scheme in terms of ANH and BNC. The simulation environment is composed of 5G ultra-dense network according to our network model, with 4 gateways and the number of small cells ranges from 110 to 470. As an example, deployment of gateway locations for a single topology with all approaches is plotted in Fig. 6.2 and the index number of gateway locations is given in Table 6.2. We can see from the table that *K*-means and *K*-medoids give the same locations for 3 gateways, only 1 gateway location changes. GA only gives 1 same location as *K*-means and *K*-medoids while other 3 gateways have different locations. KM-GA provides 2 similar locations to *K*-means, *K*-medoids and 1 similar location to K-GA and optimal. K-GA gives 2 same locations and 2 different locations compared to optimal solution. As seen from Fig. 6.2, 2 GW locations for K-GA and optimal are mapped near to each other.

Table 6.2. Gateway Locations Indexes in UD

Method	Gateway Location Indexes			
	GW-1	GW-2	GW-3	GW-4
Baseline	3	5	8	10
***K*-means**	167	289	302	322
***K*-medoids**	167	267	302	322
GA	72	202	322	353
KM-GA	128	302	322	431
K-GA	94	289	376	431
Optimal	159	225	376	431

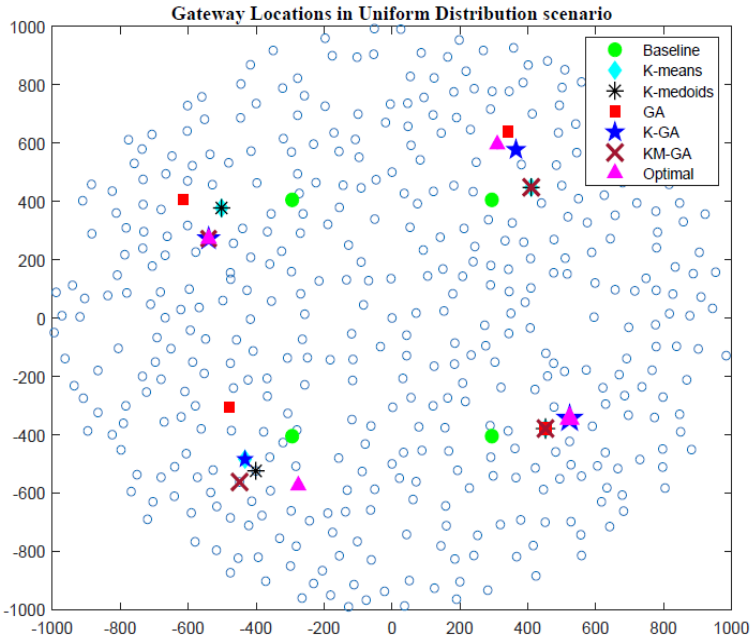

Fig. 6.2. UD simulation scenario with GW Locations

Fig. 6.3 shows the ANH vs. small cell density. As the node density increases, connectivity increases, enabling the shortest path algorithm to find better paths resulting in smaller ANH at higher node densities. As shown in Fig. 6.3, ANH for K-GA is significantly smaller than the baseline method. At a lower node density of 110, ANH is reduced from 4.49 (Baseline) to 3.61 (*K*-means) to 3.56 (*K*-medoids) to 2.89 (GA) to 2.88 (K-GA). This shows that *K*-medoids slightly works better compare to *K*-means because of its more

effective clustering and gateway selection mechanism. GA performs well compared to the *K*-means and *K*-medoids algorithm at lower node densities. In *K*-means, gateway locations are based on *M* centroids, so at lower node densities, hops are longer due to larger distances from centroids to small cells. This reduces the options for shorter hop paths, resulting in more hops compared to *K*-medoids and GA. In contrast, *K*-medoids returns actual small cells as GWs and GA works directly with the number of hops as the fitness measure. K-GA improves over GA due to a better initial population provided by its *K*-means stage. K-GA shows an increase in performance of *35.85%*, *20.22%, 19.10%,* and *0.34%* compared to baseline, *K*-means, *K*-medoids, and GA, respectively. At a higher node density of 470, ANH is reduced from 2.53 (Baseline) to 2.42 (GA) to 2.41(*K*-means and *K*-medoids) to 2.37 (K-GA). This shows an increase in performance with K-GA of *6.32%, 1.66%*, *1.66%,* and *2.07%* compared to baseline, *K*-means, *K*-medoids, and GA respectively. At higher node densities, *K*-means, *K*-medoids perform almost similar as GA and additionally baseline also shows huge improvement. This is because of smaller distances from gateways to small cells and hence more connectivity options being available. K-GA outperforms the other methods in terms of ANH at all node densities.

We further evaluate the quality of proposed K-GA algorithm with the exact solution performed in CPLEX optimization studio. As expressed in results Table 6.4, K-GA performs within *2.49%* of optimal value at low node density of 110 and for high node density of 470 shows performance gap of *0.85%* compared to optimal solution. It illustrates that K-GA has near-optimal performance at high node densities. To evaluate the computational time, we calculate the average runtime for each node density for K-GA and Optimal solution in Uniform distribution scenario. The runtime analysis for K-GA and Optimal in UD is given in Table 6.5 and Fig. 6.4 which show that optimal solution requires very large amount of time to achieve exact solution of GLP. In contrast, our proposed K-GA algorithm finds near optimal solution within *3%* of optimum very quickly in almost *95%* less time compared to optimal solution for all node densities. Mean values and CI for average number of hops for baseline, *K*-means, *K*-medoids and GA are given in Table 6.3. Additionally, results of ANH for optimal and K-GA are given in Table 6.4 in Uniform distribution scenario.

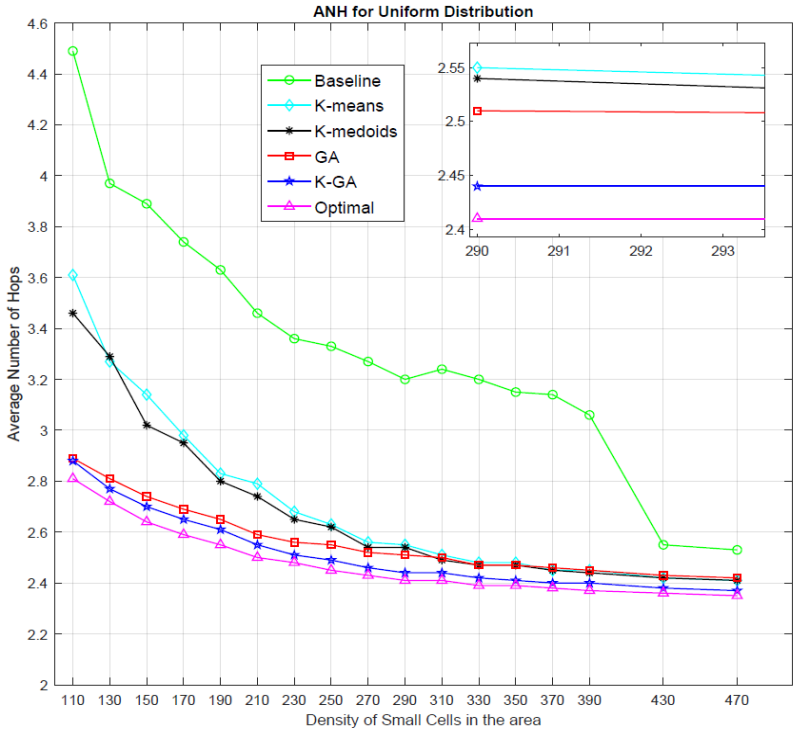

Fig. 6.3. Average Number of Hops in UD

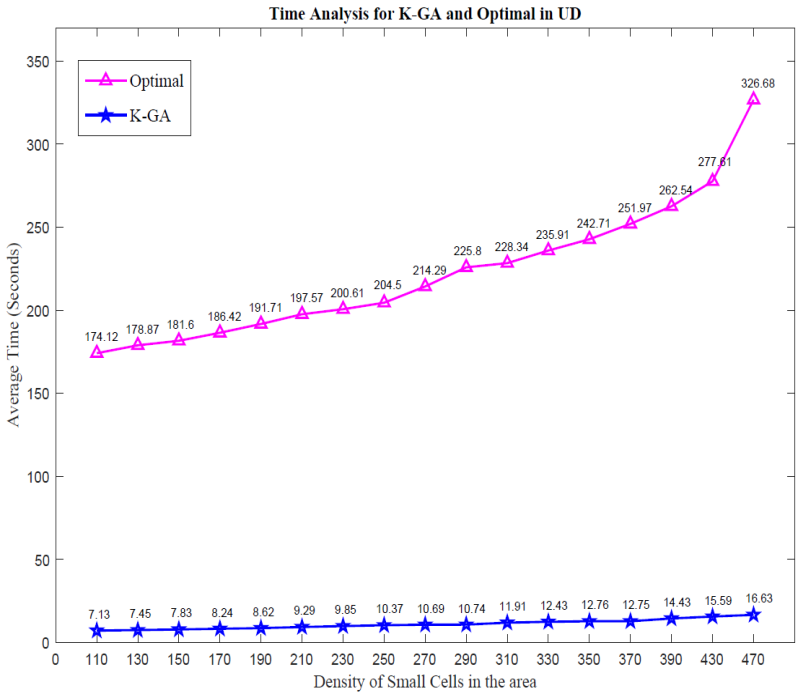

Fig. 6.4. Time Analysis of K-GA and Optimal Approach in UD

Table 6.3. ANH for Baseline, *K*-medoids, *K*-means and GA in UD

Average Number of Hops								
Small Cell Density	**GA**		***K*-means**		***K*-medoids**		**Baseline**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
110	2.89	2.86—2.93	3.61	3.48—3.74	3.56	3.36—3.66	4.49	4.31—4.67
130	2.81	2.78—2.84	3.27	3.17—3.37	3.29	3.18—3.40	3.97	3.84—4.10
150	2.74	2.71—2.76	3.14	3.05—3.24	3.02	2.94—3.10	3.89	3.78—4.00
170	2.69	2.67—2.71	2.98	2.91—3.05	2.95	2.88—3.02	3.74	3.63—3.86
190	2.65	2.63—2.66	2.83	2.78—2.88	2.80	2.75—2.84	3.63	3.52—3.73
210	2.59	2.58—2.61	2.79	2.73—2.84	2.74	2.69—2.79	3.46	3.38—3.54
230	2.56	2.55—2.58	2.68	2.63—2.73	2.65	2.60—2.69	3.36	3.28—3.44
250	2.55	2.54—2.56	2.63	2.59—2.67	2.62	2.58—2.67	3.33	3.23—3.43
270	2.52	2.51—2.53	2.56	2.53—2.59	2.54	2.52—2.56	3.27	3.19—3.35
290	2.51	2.50—2.52	2.55	2.52—2.58	2.54	2.50—2.57	3.20	3.12—3.27
310	2.50	2.49—2.50	2.51	2.49—2.54	2.49	2.48—2.51	3.24	3.17—3.32
330	2.47	2.47—2.48	2.48	2.46—2.50	2.47	2.46—2.49	3.20	3.12—3.27
350	2.47	2.46—2.48	2.48	2.46—2.49	2.47	2.45—2.48	3.15	3.09—3.22
370	2.46	2.46—2.47	2.45	2.44—2.46	2.45	2.44—2.46	3.14	3.07—3.21
390	2.45	2.45—2.46	2.45	2.44—2.47	2.44	2.43—2.45	3.06	2.99—3.12
430	2.43	2.43—2.44	2.42	2.42—2.43	2.42	2.42—2.43	2.55	2.54—2.56
470	2.42	2.42—2.43	2.41	2.41—2.42	2.41	2.41—2.42	2.53	2.53—2.54

Table 6.4. ANH for K-GA and Optimal Approach in UD

Average Number of Hops					
Small Cell Density	**Optimal**		**K-GA**		**Gap between Optimal and K-GA (%)**
	Mean	**95% CI**	**Mean**	**95% CI**	**ANH Gap**
110	2.81	2.77—2.83	2.88	2.84—2.91	2.49%
130	2.72	2.69—2.75	2.77	2.74—2.81	1.84%
150	2.64	2.61—2.66	2.70	2.67—2.73	2.27%
170	2.59	2.57—2.61	2.65	2.63—2.67	2.32%
190	2.55	2.53—2.56	2.61	2.59—2.63	2.35%
210	2.50	2.49—2.51	2.55	2.53—2.56	2.00%
230	2.48	2.46—2.50	2.51	2.50—2.53	1.21%
250	2.45	2.44—2.47	2.49	2.47—2.51	1.63%
270	2.43	2.42—2.44	2.46	2.45—2.47	1.23%
290	2.41	2.41—2.42	2.44	2.43—2.45	1.24%
310	2.41	2.40—2.42	2.44	2.43—2.45	1.24%
330	2.39	2.38—2.39	2.42	2.41—2.42	1.26%
350	2.39	2.38—2.40	2.41	2.40—2.42	0.84%
370	2.38	2.38—2.39	2.40	2.40—2.41	0.84%
390	2.37	2.35—2.38	2.40	2.39—2.40	1.27%
430	2.36	2.36—2.37	2.38	2.38—2.39	0.85%
470	2.35	2.35—2.36	2.37	2.37—2.38	0.85%

Table 6.5. Time Analysis for K-GA and Optimal Approach in UD

Time Analysis (Seconds)					
Small Cell Density	**Optimal**		**K-GA**		**Time Statistics between Optimal and K-GA**
	Mean	**95% CI**	**Mean**	**95% CI**	**Saved Time (%)**
110	174.12	174.06—174.18	7.13	7.06—7.19	95.91%
130	178.87	178.80—178.94	7.45	7.40—7.50	95.83%
150	181.60	181.52—181.68	7.83	7.78—7.89	95.69%
170	186.42	186.33—186.50	8.24	8.17—8.31	95.58%
190	191.71	191.60—191.83	8.62	8.55—8.68	95.50%
210	197.57	197.41—197.73	9.29	9.21—9.37	95.30%
230	200.61	200.46—200.78	9.85	9.77—9.93	95.09%
250	204.50	204.31—204.68	10.37	10.30—10.45	94.93%
270	214.29	213.98—214.59	10.69	10.61—10.77	95.01%
290	225.80	225.48—226.12	10. 74	10.65—10.83	95.24%
310	228.34	227.99—228.69	11.91	11.73—12.10	94.78%
330	235.91	235.69—236.14	12.43	12.33—12.53	94.73%
350	242.71	242.41—243.01	12.76	12.65—12.87	94.74%
370	251.97	251.51—252.43	12.75	12.66—13.85	94.94%
390	262.54	261.89—263.19	14.43	14.28—14.59	94.50%
430	277.61	275.62—-279.61	15.59	15.44—15.75	94.38%
470	326.68	322.32—331.04	16.63	16.50—16.75	94.91%

Fig. 6.5 illustrates the BNC for all methods. As node density increases, backhaul network capacity also increases for all methods. With increasing node density, ANH decreases resulting in increased BNC. At a lower node density of 110, BNC is 31.64 (Baseline), 38.22 (*K*-means), 39.43 (*K*-medoids), 45.55 (GA), and 45.82 (K-GA). This indicates a capacity improvement of *44.81%, 19.8%*, *16.20%,* and *0.59%* for K-GA compared to baseline, *K*-means, *K*-medoids, and GA respectively. At a higher node density of 470, K-GA shows an improvement in capacity of *6.63%, 1.63%*, *1.56%,* and *2.26%* compared to baseline, *K*-means, *K*-medoids, and GA respectively. K-GA achieves better capacity at all node densities compared to the other methods. In addition to this, K-GA obtained backhaul network capacity within maximum of *3*% of the optimal solution for all

node densities in Uniform distribution scenario. As the node density increases, K-GA improves backhaul network capacity. At high node density of 470, K-GA achieves result within *0.55*% of optimal solution.

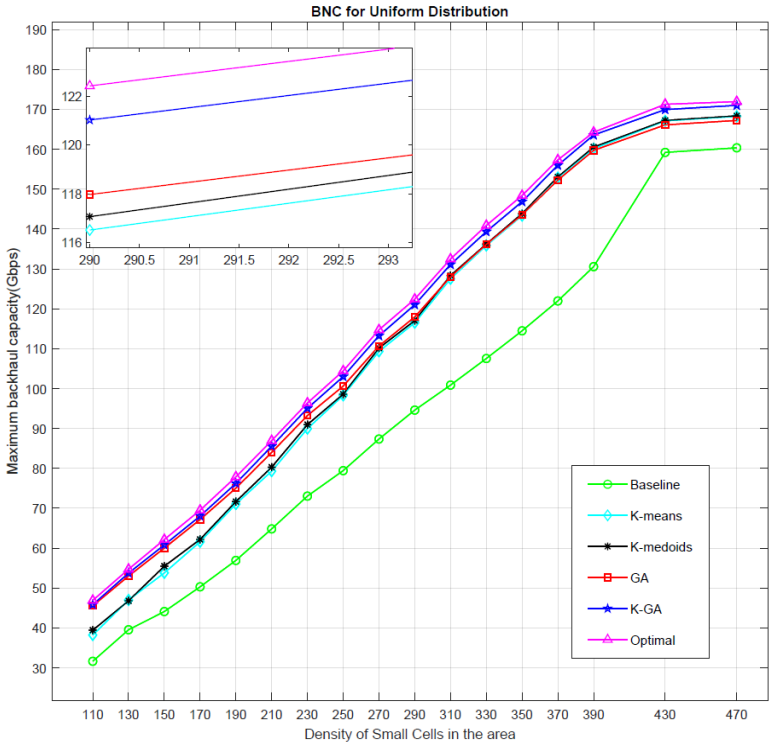

Fig. 6.5. Backhaul Network Capacity in UD

Table 6.6. BNC for Baseline, *K*-medoids, *K*-means and GA in UD

Backhaul Network Capacity (Gbps)								
Small Cell Density	**GA**		***K*-means**		***K*-medoids**		**Baseline**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
110	45.55	44.67—46.43	38.22	36.90—39.55	39.43	38.19—40.66	31.64	30.56—32.73
130	53.00	52.06—53.95	47.00	45.55—48.46	46.80	45.28—48.33	39.54	38.22—40.86
150	60.02	59.02—61.01	53.80	52.17—55.44	55.48	54.02—56.94	44.11	42.82—45.39
170	67.18	66.07—68.28	61.60	60.03—63.17	62.19	60.59—63.79	50.31	48.78—51.84
190	75.05	73.77—76.32	71.06	69.37—72.74	71.67	70.11—73.24	56.93	55.20—58.66
210	84.01	82.62—85.40	79.30	77.27—81.32	80.31	78.42—82.19	64.83	63.13—66.53
230	93.24	91.75—94.73	89.96	87.85—92.06	90.97	89.00—92.94	73.05	71.12—74.98
250	100.63	99.07—102.2	98.30	96.23—100.37	98.61	96.52—100.70	79.44	77.15—81.74
270	110.69	109.30—112.09	109.34	107.65—111.03	110.16	108.54—111.79	87.36	85.18—89.55
290	117.97	116.57—119.38	116.51	114.68—118.34	117.06	115.23—118.90	94.61	92.26—96.96
310	128.01	126.50—129.53	127.53	125.63—129.43	128.34	126.67—130.02	100.88	98.23—103.53
330	136.13	134.68—137.58	135.87	134.08—137.67	136.24	134.49—137.98	107.54	105.04—110.05
350	143.54	141.80—145.28	143.36	141.37—145.36	143.87	141.86—145.88	114.49	111.88—117.11
370	152.29	150.79—153.79	152.90	151.27—154.53	153.07	151.45—154.68	121.98	119.01—124.96
390	159.73	158.42—161.04	160.14	158.46—161.81	160.55	159.12—161.98	130.58	127.70—133.45
430	166.12	165.64—166.60	167.12	166.56—167.68	167.23	166.66—167.80	159.22	158.57—159.87
470	167.23	166.87—167.58	168.26	167.89—168.64	168.38	167.99—168.77	160.37	159.93—160.80

Table 6.7. BNC for K-GA and Optimal Approach in UD

Backhaul Network Capacity (Gbps)					
Small Cell Density	Optimal		K-GA		Gap between Optimal and K-GA (%)
	Mean	95% CI	Mean	95% CI	ANH Gap
110	46.91	45.99—47.83	45.82	44.9—46.72	2.32%
130	54.72	53.73—55.71	53.74	52.73—54.75	1.79%
150	62.16	61.11—63.22	60.81	59.74—61.89	2.17%
170	69.52	68.37—70.68	68.04	66.85—69.24	2.13%
190	77.87	76.56—79.19	76.23	74.89—77.56	2.11%
210	86.96	85.54—88.39	85.54	84.06—87.02	1.63%
230	96.39	94.76—98.02	95.06	93.51—96.60	1.38%
250	104.45	102.70—106.18	103.01	101.23—104.78	1.38%
270	114.74	113.31—116.17	113.25	111.79—114.71	1.30%
290	122.43	120.96—123.89	121.03	119.56—122.51	1.14%
310	132.5	130.99—134.07	131.11	129.56—132.66	1.05%
330	140.90	139.36—142.43	139.35	137.77—140.93	1.10%
350	148.42	146.60—150.23	146.87	145.02—148.71	1.04%
370	157.35	155.82—158.89	155.99	154.43—157.57	0.86%
390	164.28	162.23—166.32	163.58	162.23—164.93	0.43%
430	171.28	170.81—171.76	169.95	169.42—170.48	0.78%
470	171.95	171.67—172.23	171.01	170.71—171.31	0.55%

6.2.2. Results of Bivariate Gaussian Distribution:

In this section, simulations for all approaches are carried out in the bivariate Gaussian Distribution scenario where only high node densities of small cells are considered from 310 to 470 with an interval of 40 to verify the proposed algorithm. To illustrate the gateway locations in GD scenario, results with single topology for all methods are plotted in Fig. 6.6. As seen from Table 6.8 *K*-means and *K*-medoids have 2 same GW locations and 2 different GW locations but GA has totally different locations compare to *K*-means and *K*-medoids clustering methods. Furthermore, our proposed algorithm K-GA has 3 same

gateway locations as optimal solution. From this analysis, it can be said that our proposed scheme obtains similar results as the optimum approach under Gaussian distribution.

Table 6.8. Gateway Locations Indexes in GD

Gateway Location Indexes				
Approach	**GW-1**	**GW-2**	**GW-3**	**GW-4**
Baseline	3	5	8	10
***K*-means**	40	63	113	124
***K*-medoids**	40	124	164	376
GA	56	282	298	352
K-GA	75	80	280	298
Optimal	75	199	280	298

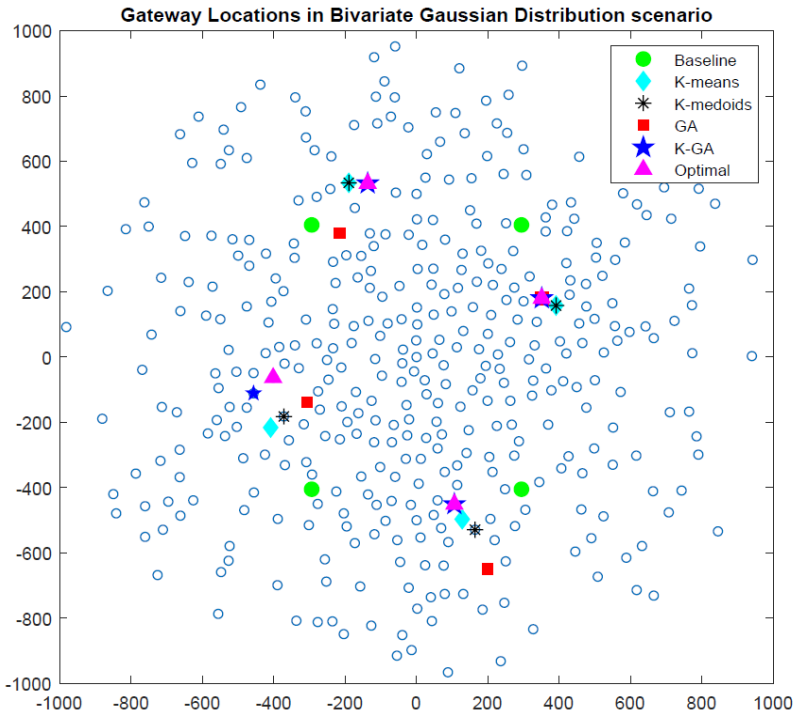

Fig. 6.6. GD simulation scenario with GW Locations

In Fig. 6.7 average number of hops is plotted for all heuristic approaches, baseline method and optimal solution in bivariate Gaussian distribution scenario. It can be observed from the graph that when the network size is increased from the 310 nodes to 470 nodes, the difference between the ANH for baseline, *K*-means, *K*-medoids, GA, and K-GA is

about 0.01–0.05, which is less compared to Uniform distribution scenario. In Gaussian distribution scenario, probability of small cells placed in center is high and at border a smaller number of small cells are present. In this distribution, more and more small cells are placed near the center and are at closer distance to the gateways. As a result, most of the small cells need fewer number of hops to reach to the gateways. As can be seen, the Baseline has poor performance compared to all other approaches. *K*-means, *K*-medoids and GA have very comparable performances. GA performs better at low node densities of 310 and 350 while *K*-means and *K*-medoids have slightly better performances at high node densities of 390, 430, and 470. It is evident from the graph that K-GA provides better performance in terms of ANH for all node densities compared to all heuristic approaches and baseline method. It shows improvement of *7.79*% compared to baseline at lower node density of 310 and *4.87*% at higher node density. The combination of *K*-means and GA, i.e., K-GA beats the standalone results of *K*-means and GA for average number of hops in this scenario.

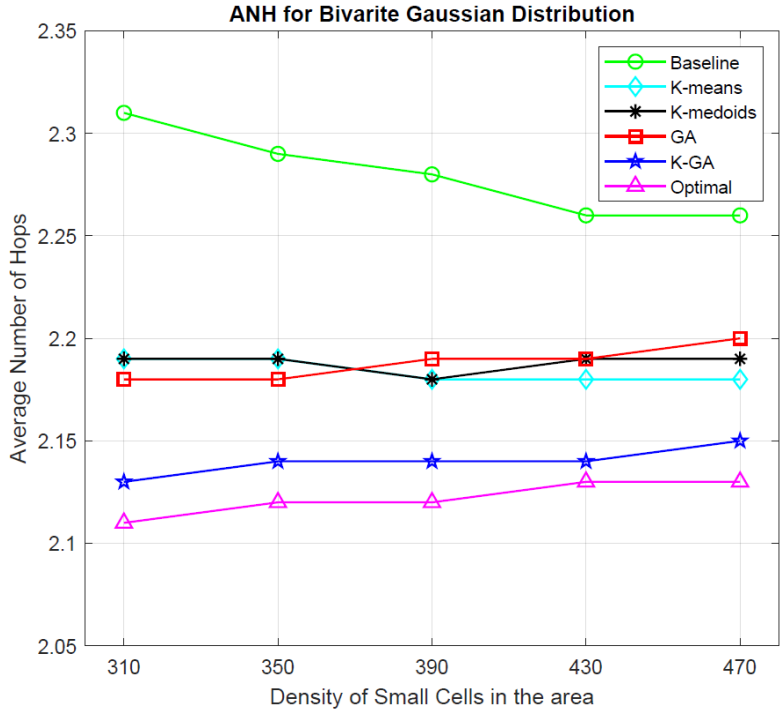

Fig. 6.7. Average Numbers of Hops in GD

Mean values and CI for the ANH for baseline, *K*-means, *K*-medoids, and GA are given in Table 6.9 and ANH results for optimal and K-GA are given in Table 6.10.

Table 6.9. ANH for Baseline, *K*-medoids, *K*-means and GA in GD

Average Number of Hops								
Small Cell Density	**GA**		***K*-means**		***K*-medoids**		**Baseline**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
310	2.18	2.17—2.19	2.19	2.18— 2.20	2.19	2.18—2.20	2.31	2.29—2.32
350	2.18	2.18—2.19	2.19	2.18 —2.20	2.19	2.18—2.20	2.29	2.28—2.29
390	2.19	2.18—2.19	2.18	2.18 —2.19	2.18	2.18—2.19	2.28	2.27—2.28
430	2.19	2.19—2.20	2.18	2.18—2.19	2.19	2.18—2.19	2.26	2.26—2.27
470	2.20	2.20— 2.21	2.18	2.18—2.19	2.19	2.18—2.21	2.26	2.25—2.27

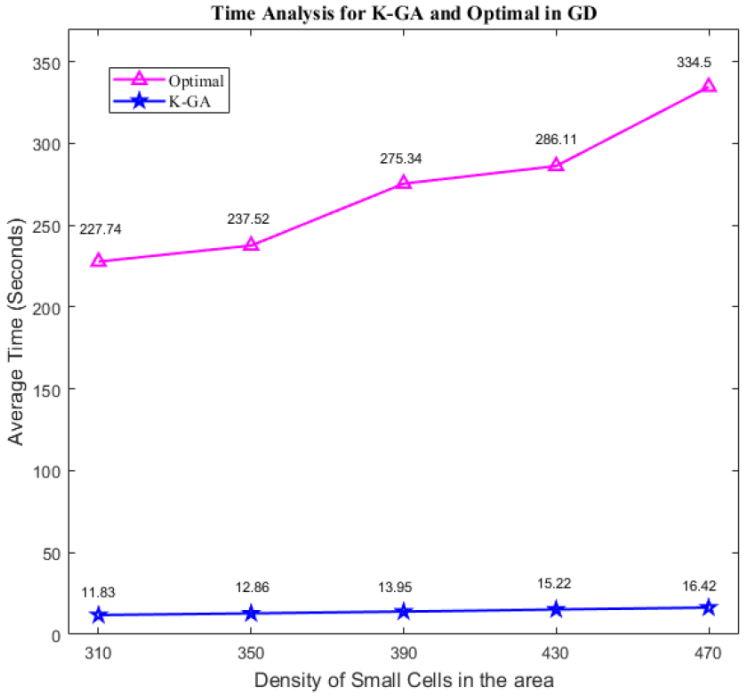

Fig. 6.8. Time Analysis for K-GA and Optimal Approach in GD

As observed from the table, there is an average gap of *1%* between K-GA and optimal approach for all node densities which implies that K-GA achieved near optimal results. Average runtime information of all runs for K-GA and Optimal in GD is given in Table 6.11 to evaluate the computational time details. It can be observed from the Fig. 6.8 of time statistics that optimal consumes much more time compared to K-GA heuristic. K-

GA provides almost similar results as mathematical solution in a very small amount of time. Overall, K-GA saves almost *95%* of time compared to mathematical solution and gives results within *1%* of optimal value for each node density.

Table 6.10. ANH for K-GA and Optimal Approach in GD

Average Number of Hops					
Small Cell Density	**Optimal**		**K-GA**		**Gap between Optimal and K-GA (%)**
	Mean	**95% CI**	**Mean**	**95% CI**	
310	2.11	2.09—2.11	2.13	2.12— 2.14	0.95%
350	2.12	2.10—2.12	2.14	2.14—2.15	0.94%
390	2.12	2.11—2.13	2.14	2.13—2.15	0.94%
430	2.13	2.12—2.13	2.14	2.14—2.15	0.47%
470	2.13	2.13—2.14	2.15	2.14—2.16	0.94%

Table 6.11. Time Analysis for K-GA and Optimal Approach in GD

Time Analysis					
Small Cell Density	**Optimal**		**K-GA**		**Time Statistics between Optimal and K-GA**
	Mean	**95% CI**	**Mean**	**95% CI**	**Saved Time (%)**
310	227.74	227.38—228.09	11.83	11.74—11.93	94.81%
350	237.52	233.54—241.50	12.86	12.77—12.96	94.59%
390	275.34	274.74—275.94	13.95	13.84—14.06	94.93%
430	286.11	284.97—287.25	15.22	15.1—15.33	94.68%
470	334.50	327.25—341.74	16.42	16.22—16.62	95.09%

Fig. 6.9 illustrates the backhaul network capacity for all approaches in bivariate Gaussian distribution scenario. Compared to baseline method, capacity improved from 138.02 to 148.68 at node density of 310 with K-GA. At node density of 470, capacity also improved from 179.61 to 187.76 by optimizing gateway locations using proposed K-GA method. Similar to ANH results, *K*-means, *K*-medoids, and GA almost perform equivalent to each other in GD. K-GA shows improvement compare to baseline, *K*-means, *K*-medoids,

GA and provides near optimal results in all node densities. Table 6.12 presents the backhaul network capacity results for comparison methods.

Table 6.12. BNC for Baseline, *K*-medoids, *K*-means and GA in GD

Backhaul Network Capacity (Gbps)								
Small Cell Density	**GA**		***K*-means**		***K*-medoids**		**Baseline**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
310	145.42	143.71—147.13	145.10	143.24—146.95	145.25	143.37—147.13	138.02	136.19—139.85
350	162.48	160.93—164.02	162.27	160.62—163.92	162.37	160.74—164.01	155.51	153.88—157.15
390	178.74	177.43—180.06	178.89	177.55—180.23	178.88	177.48—180.28	171.91	170.49—173.32
430	184.07	183.60—184.54	185.18	184.61—185.76	185.05	184.47—185.63	178.88	178.35—179.41
470	183.90	183.36—184.43	184.86	184.31—185.41	184.47	183.54—185.40	179.61	179.03—180.19

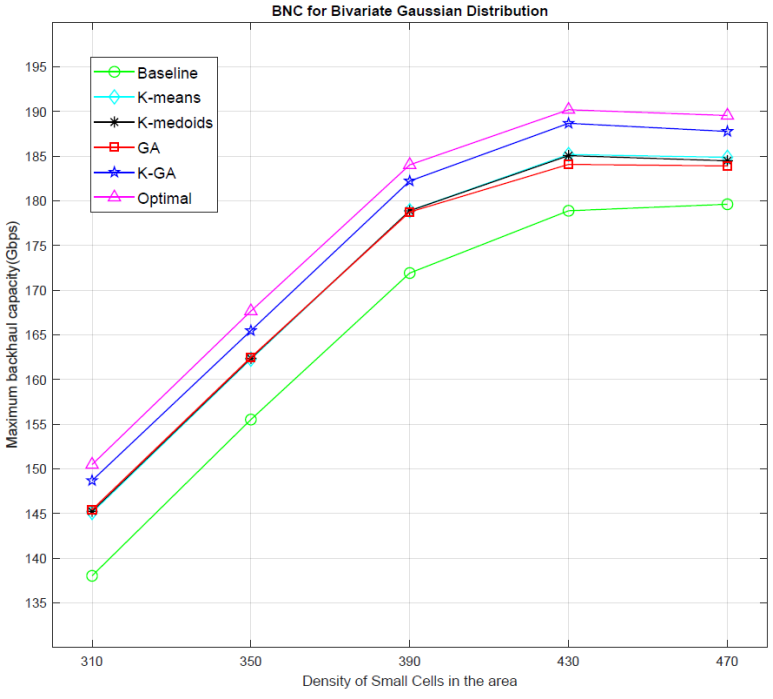

Fig. 6.9. Backhaul Network Capacity in GD

As seen from the Table 6.13, gap between K-GA and optimal reduces from *1.19%* to *0.94%* as the node density increases. This indicates that at higher node densities K-GA performs better and provides within *1%* results of optimal value of backhaul network capacity under Gaussian distribution scenario.

Table 6.13. BNC for K-GA and Optimal Approach in GD

Backhaul Network Capacity (Gbps)					
Small Cell Density	**Optimal**		**K-GA**		**Gap between Optimal and K-GA (%)**
	Mean	**95% CI**	**Mean**	**95% CI**	
310	150.47	148.74—152.20	148.68	146.88 —150.47	1.19%
350	167.65	166.0 —169.21	165.46	163.88 —167.03	1.31%
390	184.02	182.65—185.39	182.21	180.84 —183.59	0.98%
430	190.19	189.74—190.64	188.67	188.18—189.17	0.80%
470	189.54	189.06—190.02	187.76	187.22—188.30	0.94%

6.2.3. Results of Cluster Distribution:

This section presents the implementation results of proposed K-GA algorithm with other approaches and optimal solution under Cluster-based distribution environment. Identical to previous section, small cell densities varying from 310 to 470 with a gap of 40 are considered. Fig. 6.10 shows the optimized gateway locations of single topology which is obtained from the implementation of all approaches including K-GA and optimal solution in Cluster distribution scenario, which has 6 different clusters and also few spread out other small cells. As viewed from the Table 6.14, baseline, *K*-medoids, *K*-means, and GA have most of different gateway locations from each other. In contrast, K-GA has 3 same gateway locations and 1 different location as compare to optimal solution. It shows a better performance of our proposed K-GA algorithm as it gives GW locations similar to the exact solution.

Table 6.14. Gateway Locations Indexes in CD

Gateway Location Indexes				
Approach	**GW-1**	**GW-2**	**GW-3**	**GW-4**
Baseline	45	76	169	199
***K*-means**	102	225	267	393
***K*-medoids**	107	217	267	396
GA	36	112	167	216
K-GA	81	186	212	337
Optimal	94	186	212	337

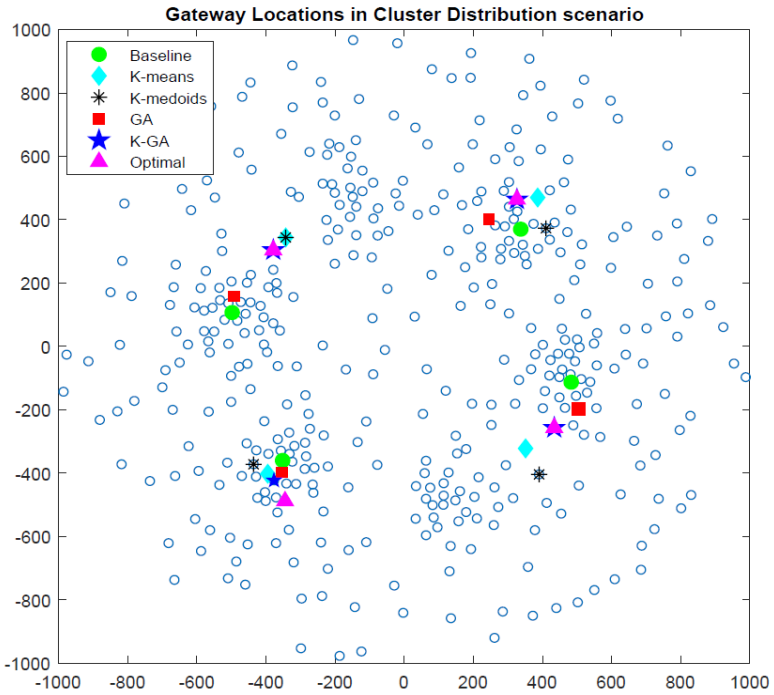

Fig. 6.10. CD simulation scenario with GW Locations

Fig. 6.11, Table 6.15, and Table 6.16 represents the average number of hops for baseline, *K*-means, *K*-medoids, genetic algorithm, K-GA, and optimal solution in Cluster distribution. As viewed from the results as the node density increases, ANH is decreasing for all approaches. From the Fig. 6.10 and Fig. 6.11, we can see that baseline gateway locations are considered as nearest small cells to the cluster center and placed at very close distance to other methods although it performs poorly in terms of the average number of hops compared to all other approaches. This behavior shows that planning and optimization of gateway locations in multi-hop distributed ultra-dense network is really important to

achieve good results. *K*-means and *K*-medoids perform very similar for all node densities in clustered based topology because of their clustering nature. Genetic algorithm behaves better at 310 and 330 node density while its performance degrades at 430 and 470 node density compared to *K*-means and *K*-medoids. At 390 node density GA, *K*-means, and *K*-medoids gives same results. This behavior of heuristics indicates that density of small cells affects the performance of the optimization algorithms. K-GA works on better initial population provided by *K*-means clustering algorithm and these enriched initial population helps K-GA to provide superior results compared to other heuristics and baseline method. Moreover, K-GA achieves near optimal results. A comparison of ANH between K-GA and optimal solution is given is Table 6.16 and Table 6.17. Results shows that K-GA obtained average number of hops within *2*% of optimal values in Cluster distribution scenario.

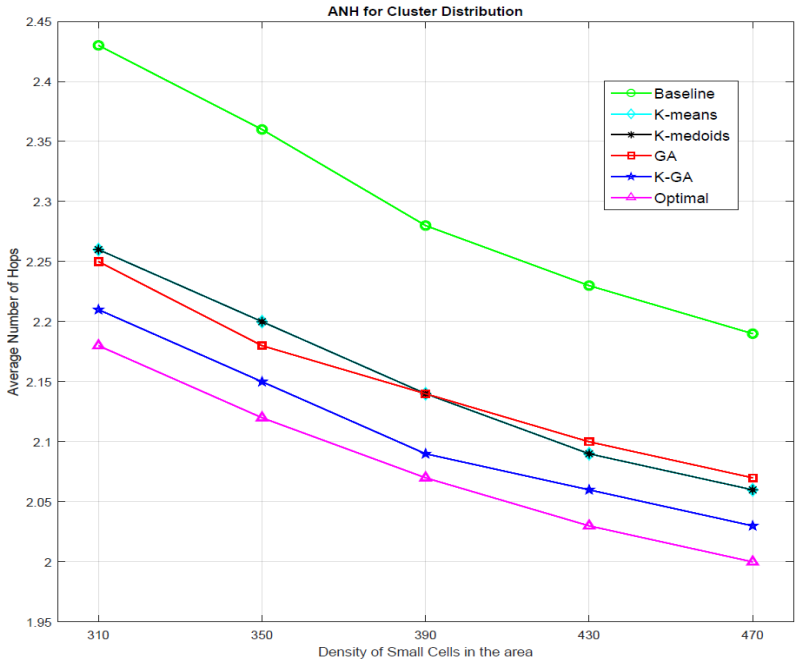

Fig. 6.11. Average Numbers of Hops in CD

It is noted that, although the optimal solution provides slightly better results for ANH, it takes much more time to find the exact solution than the proposed K-GA algorithm. Therefore, it could be concluded that, to the same GLP, the K-GA approach can find a solution much faster than the exact solution; the larger the size of the GLP, the much more computational time our proposed K-GA algorithm can save. It takes about 16 seconds

to find the solution of gateway location problem while optimal method takes 328 seconds for high node density of 470. Table 6.17 gives detailed comparison between the two solutions in terms of computational time. As shown in Table 6.16 and Table 6.17, the two approaches achieve very similar results with a maximum percentage gap of *1.5%,* while the difference between their computational time is very significant. More than *95%* computational time used by optimal approach is saved by our proposed K-GA algorithm.

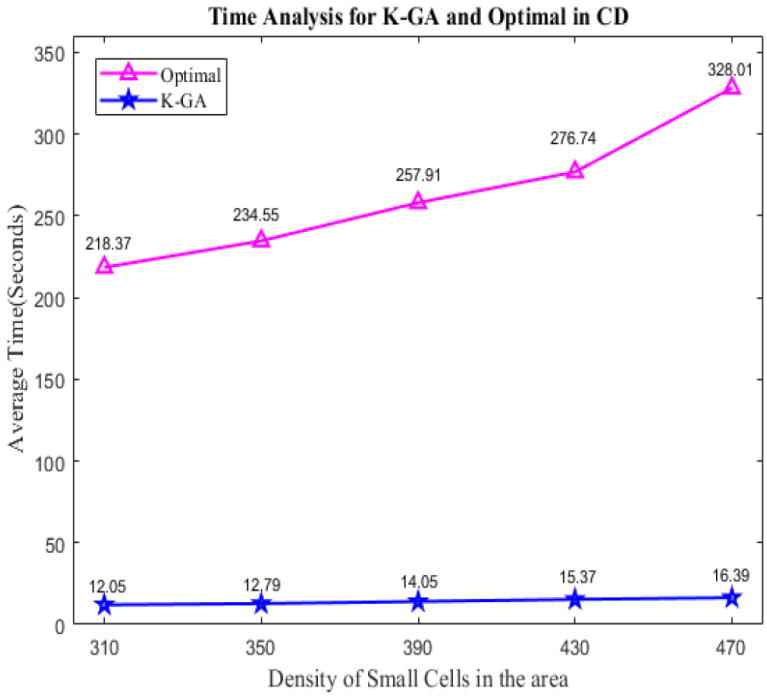

Fig. 6.12. Time Analysis for K-GA and Optimal Approach in CD

Table 6.15. ANH for Baseline, *K*-medoids, *K*-means and GA in CD

Average Number of Hops								
Small Cell Density	**GA**		***K*-means**		***K*-medoids**		**Baseline**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
310	2.25	2.24—2.26	2.26	2.24—2.27	2.26	2.25—2.27	2.43	2.41 — 2.44
350	2.18	2.18—2.19	2.20	2.19—2.21	2.20	2.19—2.21	2.36	2.34 — 2.37
390	2.14	2.13—2.14	2.14	2.13—2.15	2.14	2.13—2.15	2.28	2.26 — 2.29
430	2.10	2.09—2.10	2.09	2.08—2.10	2.09	2.08—2.10	2.23	2.21 — 2.24
470	2.07	2.06—2.08	2.06	2.05—2.07	2.06	2.05—2.07	2.19	2.18 —2.20

Table 6.16. ANH for K-GA and Optimal Approach in CD

Average Number of Hops					
Small Cell Density	**Optimal**		**K-GA**		**Gap between Optimal and K-GA (%)**
	Mean	**95% CI**	**Mean**	**95% CI**	
310	2.18	2.17—2.19	2.21	2.19—2.22	1.38%
350	2.12	2.11—2.13	2.15	2.14—2.16	1.42%
390	2.07	2.06—2.07	2.09	2.09—2.10	0.97%
430	2.03	2.02—2.04	2.06	2.05—2.07	1.48%
470	2.00	1.99—2.01	2.03	2.02—2.04	1.50%

Table 6.17. Time Analysis for K-GA and Optimal Approach in CD

Time Analysis					
Small Cell Density	**Optimal**		**K-GA**		**Time Statistics between Optimal and K-GA**
	Mean	**95% CI**	**Mean**	**95% CI**	**Saved Time (%)**
310	218.37	217.76—218.98	12.05	11.84—12.26	94.48%
350	234.55	233.69—235.42	12.79	12.67—12.92	94.55%
390	257.91	256.82—258.99	14.05	13.89—14.21	94.55%
430	276.74	275.48—278.0	15.37	15.22—15.51	94.45%
470	328.01	323.69—332.33	16.39	16.21—16.58	95.00%

Fig. 6.13, Table 6.18, and Table 6.19 represent the results of backhaul network capacity in Cluster distribution scenario. As the node density increases, backhaul network capacity is increased. K-GA shows an improvement of *9.65%* compared to baseline at low node density of 310 and *7.77%* at high node density of 470. *K*-means, *K*-medoids, and GA perform almost similar for all node densities. K-GA provides results within *2*% of the optimal value for all node densities under Cluster-based distribution situations.

Table 6.18. BNC for Baseline, *K*-medoids, *K*-means and GA in CD

Backhaul Network Capacity (Gbps)								
Small Cell Density	**GA**		***K*-means**		***K*-medoids**		**Baseline**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
310	144.18	142.18—146.18	143.78	141.62—145.94	143.55	141.39—145.70	134.05	132.10—136.01
350	162.66	160.61—164.70	162.29	160.23—164.34	162.09	159.94—164.25	151.62	149.51—153.72
390	183.35	181.75—184.95	183.32	181.73—184.92	183.06	181.46—184.65	172.36	170.61—174.11
430	193.03	192.29—193.77	193.34	192.52—194.16	193.22	192.46—193.97	181.93	181.01—182.85
470	195.59	194.85—196.33	196.14	195.28—197.00	196.03	195.15 —196.91	184.76	183.80—185.72

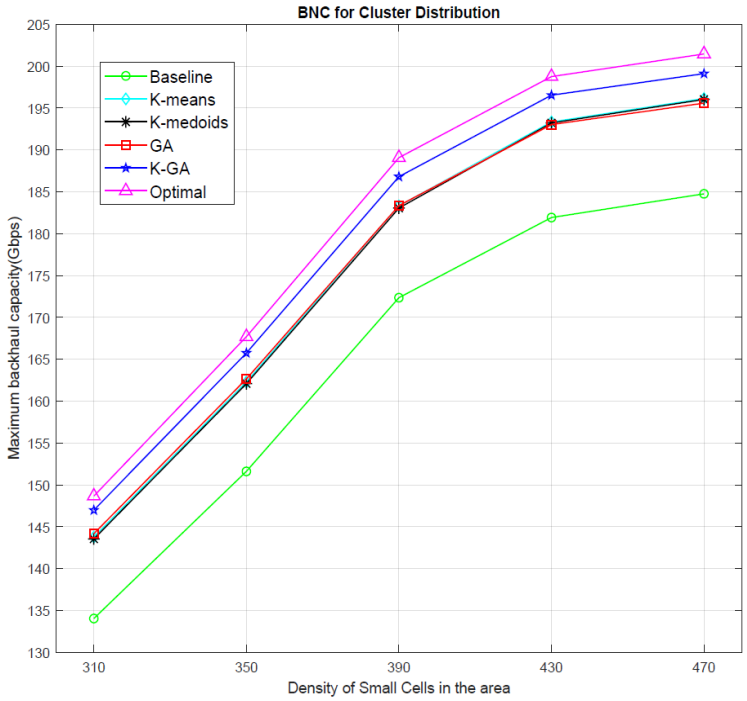

Fig. 6.13. Backhaul Network Capacity in CD

Table 6.19. BNC for K-GA and Optimal Approach in CD

Backhaul Network Capacity (Gbps)					
Small Cell Density	**Optimal**		**K-GA**		**Gap between Optimal and K-GA (%)**
	Mean	**95% CI**	**Mean**	**95% CI**	
310	148.67	146.59—150.75	146.99	144.88—149.10	1.13%
350	167.70	165.57—169.83	165.75	163.60—167.91	1.16%
390	189.08	187.57—190.59	186.81	185.21—188.42	1.20%
430	198.76	198.05—199.47	196.54	195.74—197.34	1.12%
470	201.46	200.72—202.20	199.11	198.31—199.91	1.17%

6.3. Effect of the Number of Gateways in Ultra-Dense Networks.

The performance with different number of gateways under UD, GD, and CD is evaluated in this section. We deploy 2 to 8 gateways in the connected multi-hop distributed network to investigate their impact on ANH and BNC with different small cell node densities. The locations of the gateways are obtained using our proposed K-GA algorithm.

6.3.1. Results of Uniform Distribution:

All small cells are distributed uniformly and node density ranges from 310 to 470 with an interval of 40. The results for average number of hops are given in Table 6.20 and Fig. 6.14. As seen, when small cell density increases from 310 to 470, average number of hops decreases for all gateways. Additionally, when number of gateways are increases, ANH is decreases for all node densities. This is because more gateways generate more shortest path trees and all small cells can be shared among these gateways. More gateways help to distribute the load of small cells traffic and reduce the congestion issues. The ANH difference between 310 and 470 nodes for 2 GWs is 0.10 which is more than the difference of 0.02 for 8 GWs. This difference indicates that after certain point, an increment in number of gateways does not help much in minimizing the average number of hops in Uniform distribution.

Table 6.21 and Fig. 6.15 represents the BNC in UD scenario for different number of gateways. It can be observed from the Fig. 6.15 that with increment of small cell node density, capacity is also increasing for all number of gateways. GWs-2 and GWs-3 have

lower increment rate for capacity improvement compare to other gateways. Although as seen earlier in Fig. 6.14, difference of ANH gap from 470 to 310 for GWs-2 and GWs-3 is 0.10 and 0.8 respectively, this large gap does not help much to increase the capacity. This shows that more gateways impact more on backhaul networks to increase the capacity.

Table 6.20. Effect of the Number of Gateways on ANH for UD

Average Number of Hops										
Number of Gateways	**Small Cell Density**									
	310		**350**		**390**		**430**		**470**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
2	3.45	3.43—3.46	3.42	3.41—3.43	3.39	3.38—3.40	3.36	3.36—3.37	3.35	3.34—3.36
3	2.80	2.80—2.81	2.77	2.77—2.78	2.76	2.75—2.76	2.73	2.72—2.74	2.72	2.72—2.73
4	2.44	2.43—2.45	2.41	2.40—2.42	2.40	2.39—2.40	2.38	2.38—2.39	2.37	2.37—2.38
5	2.20	2.19—2.21	2.19	2.18—2.19	2.17	2.17—2.18	2.16	2.16—2.17	2.15	2.14—2.15
6	2.03	2.03—2.04	2.02	2.01—2.03	2.01	2.01—2.02	2.01	2.0—2.01	2.0	1.99—2.0
7	1.88	1.88—1.89	1.87	1.87—1.88	1.86	1.85—1.86	1.86	1.85—1.86	1.85	1.84—1.85
8	1.77	1.77—1.78	1.77	1.76—1.77	1.76	1.75—1.76	1.75	1.75—1.76	1.75	1.75—1.75

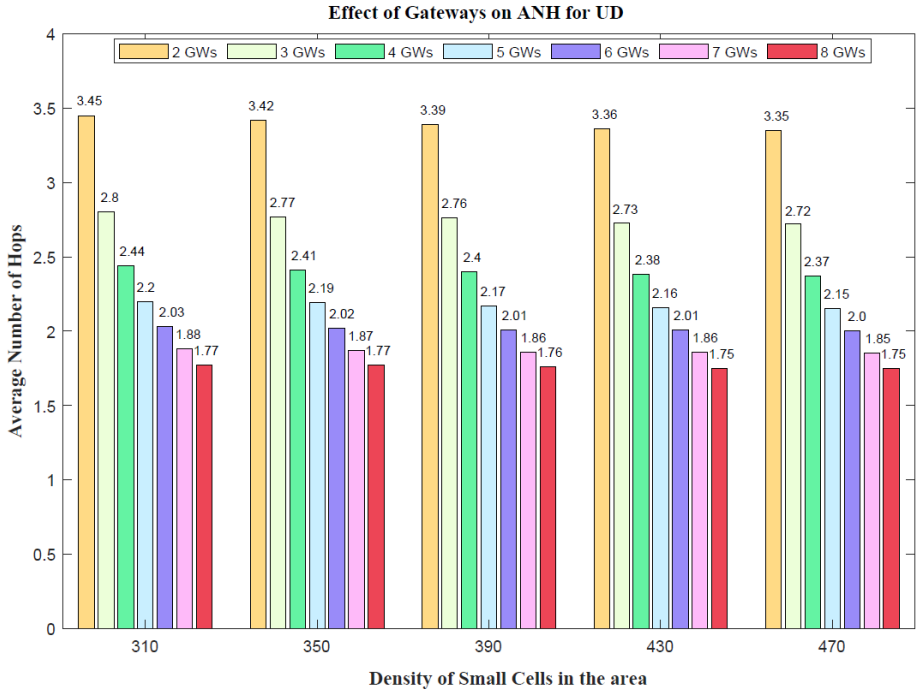

Fig. 6.14. Effect of the Number of Gateways on ANH for UD

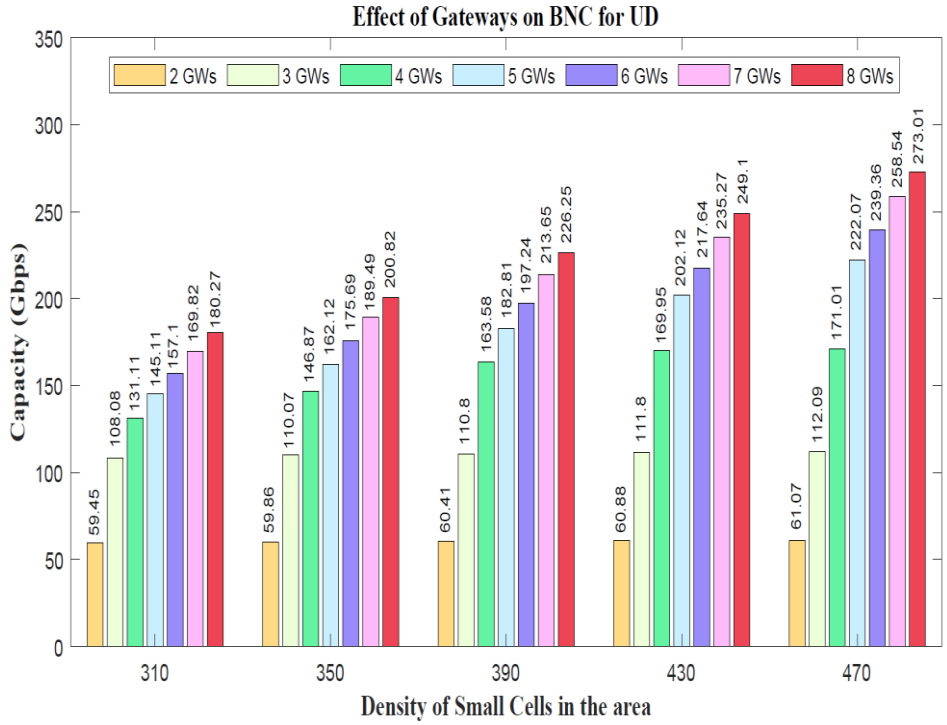

Fig. 6.15. Effect of the Number of Gateways on BNC for UD

Table 6.21. Effect of the Number of Gateways on BNC for UD

Backhaul Network Capacity (Gbps)										
No. of GWs	Small Cell Density									
	310		350		390		430		470	
	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI
2	59.45	59.25 — 59.66	59.86	59.68 — 60.04	60.41	60.25 — 60.57	60.88	60.76 — 61.0	61.07	60.94 — 61.21
3	108.08	107.49 — 108.67	110.07	109.72 — 110.43	110.80	110.53 — 111.06	111.80	111.57 — 112.03	112.09	111.85 — 112.33
4	131.11	129.56 — 132.66	146.87	145.02 — 148.71	163.58	162.23 — 164.93	169.95	169.42 — 170.48	171.01	170.71 — 171.31
5	145.11	143.43 — 146.79	162.12	160.18 — 164.07	182.81	180.74 — 184.88	202.12	200.13 — 204.10	222.07	220.32 — 223.82
6	157.1	155.29 — 158.91	175.69	173.57 — 177.80	197.24	195.03 — 199.45	217.64	215.51 — 219.76	239.36	237.38 — 241.34
7	169.82	167.78 — 171.87	189.49	187.16 — 191.81	213.65	211.21 — 216.09	235.27	232.99 — 237.54	258.54	256.30 — 260.78
8	180.27	178.12 — 182.43	200.82	198.37 — 203.27	226.25	223.68 — 228.83	249.10	246.67 — 251.52	273.01	270.68 — 275.34

6.3.2. Results of Bivariate Gaussian Distribution:

The small cell density varies from 310 to 470 with a gap of 40 under bivariate Gaussian distribution. Generally, as the node density increases, average number of hops decreases. It is because at higher node density in the same circular area, a small cell can easily find the nearest neighbors and have more connectivity options which results in smaller average number of hops. However, in the case of Gaussian distribution, network model acts differently compared to UD and CD. As seen from the Fig. 6.16 and Table 6.22, with increase in node density, average number of hops is also increasing. Reason for this behavior in GD is as follows: at low node density around the center more small cells are deployed and need shorter routing paths towards the gateways which results in fewer average number of hops. While at high node density as the number of small cells increases, towards the end of same circular area more small cells exist which need more hops to reach to the gateways and results in more average number of hops.

Fig. 6.16 and Table 6.22 represent the backhaul network capacity with different number of gateways. It can be observed from the BNC results that with 2 GWs after 390 node density, capacity is decreasing. Similarly, with 3 GWs as node density increases, capacity is decreasing and with 4 GWs as well capacity performance degrades at 470 nodes. On the other hand, with 5 and more gateways capacity keeps improving with increasing the node density. This indicates that 2, 3, and 4 GWs are not enough for very high dense networks in GD scenario.

Table 6.22. Effect of the Number of Gateways on ANH for GD

Average Number of Hops										
Number of Gateways	Small Cell Density									
	310		350		390		430		470	
	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI
2	2.79	2.78—2.80	2.80	2.79—2.81	2.82	2.81—2.83	2.84	2.83—2.84	2.86	2.85—2.87
3	2.37	2.36—2.38	2.38	2.38—2.39	2.39	2.38—2.39	2.40	2.39—2.40	2.41	2.40—2.41
4	2.13	2.12—2.14	2.14	2.14—2.15	2.14	2.13—2.15	2.14	2.14—2.15	2.15	2.14—2.16
5	1.97	1.96—1.98	1.98	1.98—1.99	1.98	1.98—1.99	1.99	1.98—1.99	2.0	1.99—2.0
6	1.82	1.82—1.83	1.83	1.83—1.84	1.83	1.83—1.84	1.84	1.83—1.84	1.84	1.84—1.85
7	1.72	1.72—1.73	1.73	1.73—1.74	1.73	1.73—1.74	1.73	1.73—1.74	1.74	1.74—1.74
8	1.64	1.64—1.65	1.65	1.65—1.66	1.66	1.65—1.66	1.66	1.65—1.66	1.67	1.66—1.67

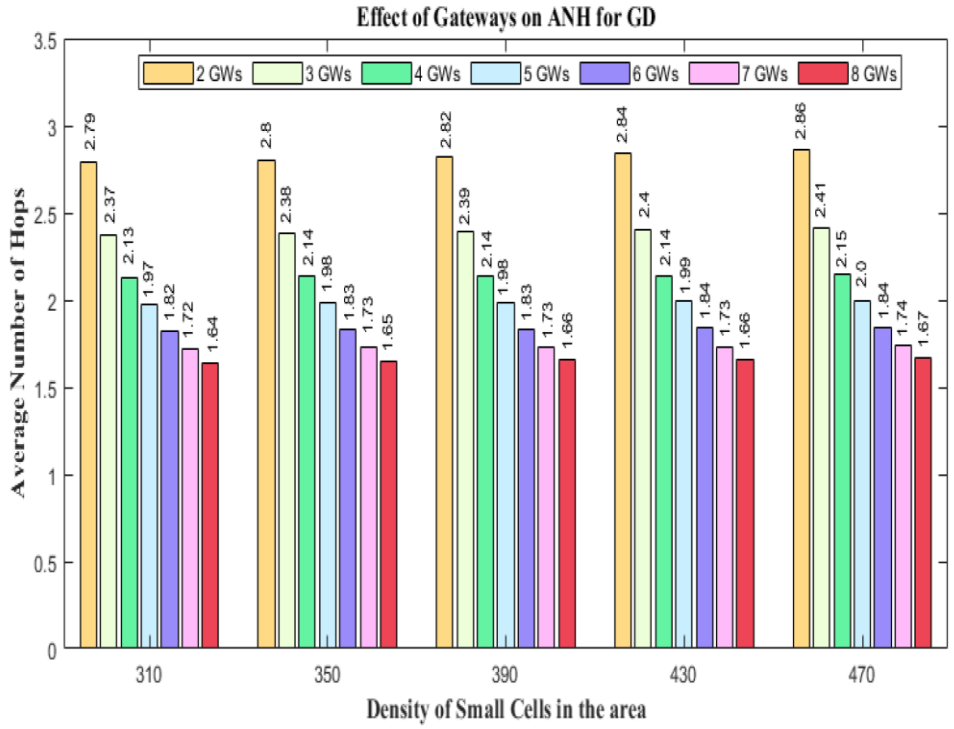

Fig. 6.16. Effect of the Number of Gateways on ANH for GD

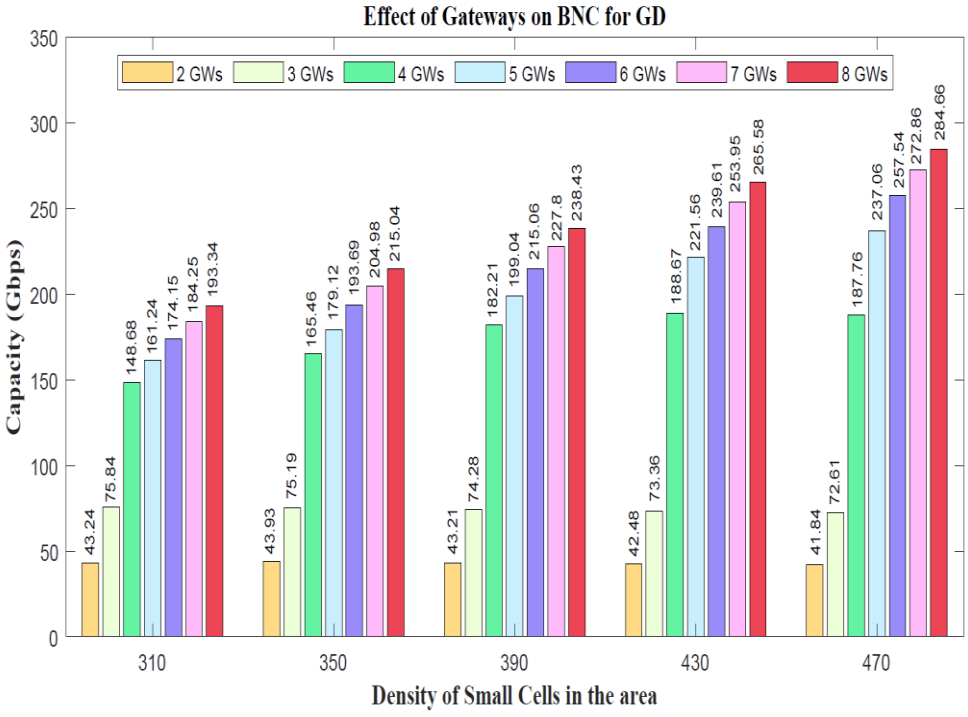

Fig. 6.17. Effect of the Number of Gateways on BNC for GD

Table 6.23. Effect of the Number of Gateways on BNC for GD

Backhaul Network Capacity (Gbps)										
No. of GWs	**Small Cell Density**									
	310		**350**		**390**		**430**		**470**	
	Mean	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**	**Mean**	**95% CI**
2	43.24	37.33 — 49.15	43.93	38.25 — 49.61	43.21	37.47 — 48.94	42.48	36.69 — 48.28	41.84	36.03 — 47.64
3	75.84	65.60 — 86.09	75.19	64.81 — 85.57	74.28	63.81 — 84.74	73.36	62.79 — 83.94	72.61	61.98 — 83.24
4	148.68	146.88 — 150.47	165.46	163.88 — 167.03	182.21	180.84 — 183.59	188.67	188.18 — 189.17	187.76	187.22 — 188.30
5	161.24	159.35 — 163.12	179.12	177.47 — 180.76	199.04	197.11 — 200.98	221.56	219.65 — 223.47	237.06	234.95 — 239.17
6	174.15	172.16 — 176.14	193.69	191.84 — 195.53	215.06	212.95 — 217.17	239.61	237.59 — 241.63	257.54	255.15 — 259.93
7	184.25	182.12 — 186.38	204.98	203.06 — 206.9	227.8	225.48 — 230.11	253.95	251.73 — 256.18	272.86	270.26 — 275.46
8	193.34	191.19 — 195.49	215.04	213.03 — 217.06	238.43	236.02 — 240.84	265.58	263.27 — 267.90	284.66	281.99 — 287.32

6.3.3. Results of Cluster Distribution:

Small cells are deployed according to Cluster-based distribution scenario and their density ranges from 310 to 470 with an interval of 40. Similar to Uniform distribution, in Cluster distribution as the node density increases, average number of hops reduces for all gateways. At higher node density, small cells have more connectivity options which results in fewer average number of hops. Moreover, as observed from the Fig. 6.18 and Table 6.24, as the number of gateways increases, ANH is decreasing further. Reason for this behavior is more gateways generate more shortest path trees and share small cell traffic among gateways. This helps to avoid congestion issue.

Fig. 6.19 and Table 6.25 represent backhaul network capacity for Cluster distribution. As noticed from results, 2 GWs and 3 GWs are not enough for higher node densities in CD as well because capacity is declining with both at 430 nodes and further. Although, 4 and more gateways performs better in Cluster distribution. The higher the number of gateways, the better the performance.

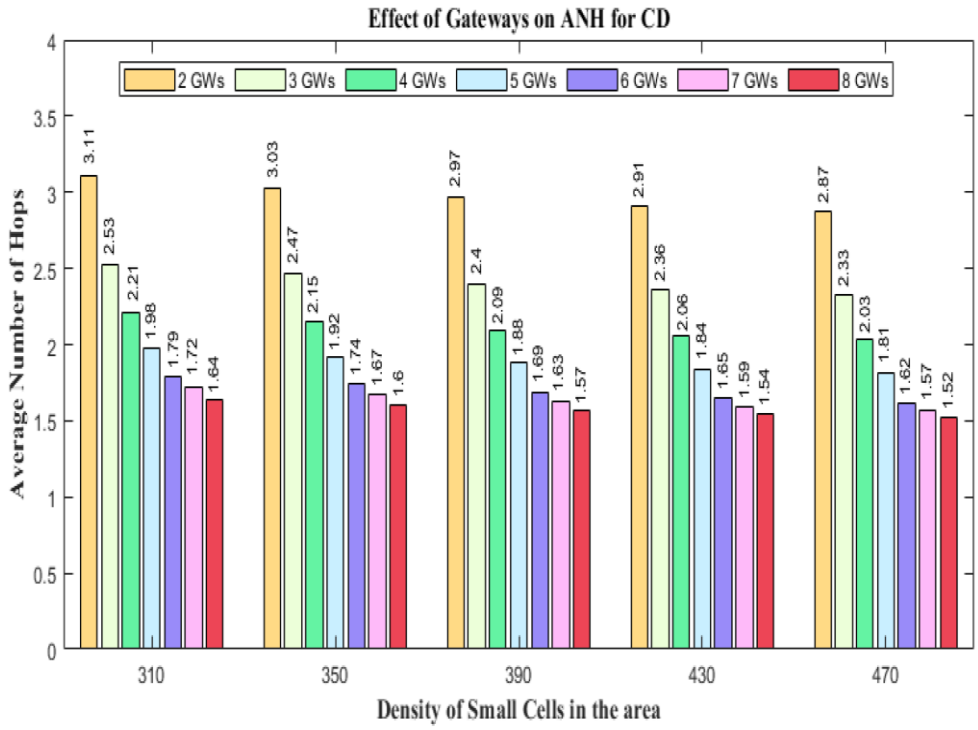

Fig. 6.18. Effect of the Number of Gateways on ANH for CD

Table 6.24. Effect of the Number of Gateways on ANH for CD

Average Number of Hops										
Number of Gateways	Small Cell Density									
	310		350		390		430		470	
	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI
2	3.11	3.10—3.12	3.03	3.02—3.04	2.97	2.96—2.98	2.91	2.90—2.92	2.87	2.86—2.88
3	2.53	2.52—2.54	2.47	2.46—2.48	2.40	2.39—2.41	2.36	2.35—2.37	2.33	2.32—2.33
4	2.21	2.19—2.22	2.15	2.14—2.16	2.09	2.09—2.10	2.06	2.05—2.07	2.03	2.02—2.04
5	1.98	1.97—1.99	1.92	1.91—1.93	1.88	1.87—1.88	1.84	1.83—1.85	1.81	1.81—1.82
6	1.79	1.79—1.80	1.74	1.73—1.75	1.69	1.68—1.70	1.65	1.64—1.66	1.62	1.61—1.63
7	1.72	1.71—1.73	1.67	1.66—1.68	1.63	1.62—1.64	1.59	1.59—1.60	1.57	1.56—1.58
8	1.64	1.63—1.65	1.60	1.59—1.61	1.57	1.56—1.57	1.54	1.53—1.54	1.52	1.51—1.52

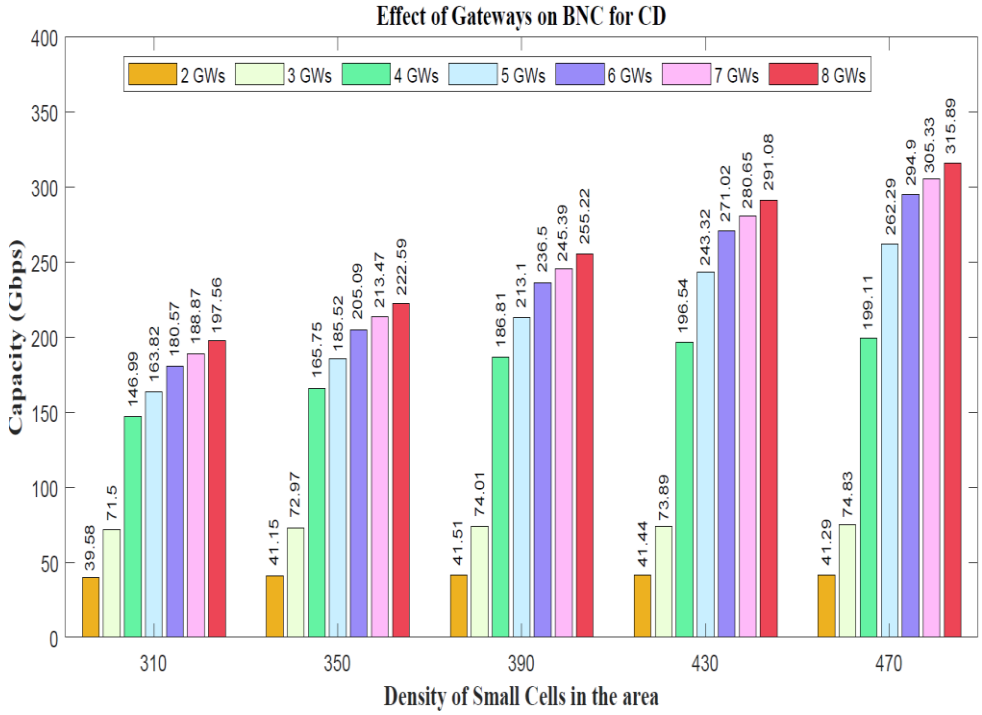

Fig. 6.19. Effect of the Number of Gateways on BNC for CD

Table 6.25. Effect of the Number of Gateways on BNC for CD

Backhaul Network Capacity (Gbps)										
No. of GWs	Small Cell Density									
	310		350		390		430		470	
	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI	Mean	95% CI
2	39.58	34.48 — 44.68	41.15	36.04 — 46.27	41.51	36.12 — 46.91	41.44	35.83 — 47.05	41.29	35.45 — 47.13
3	71.50	62.04 — 80.96	72.97	63.08 — 82.87	74.01	63.59 — 84.44	73.89	63.03 — 84.74	74.83	63.76 — 85.90
4	146.99	144.88 — 149.10	165.75	163.60 — 167.91	186.81	185.21 — 188.42	196.54	195.74 — 197.34	199.11	198.31 — 199.91
5	163.82	161.50 — 166.13	185.52	183.14 — 187.90	213.10	210.47 — 215.73	243.32	240.99 — 245.65	262.29	259.90 — 264.69
6	180.57	178.13 — 183.01	205.09	202.53 — 207.64	236.50	233.58 — 239.42	271.02	268.56 — 273.49	294.90	291.72 — 298.09
7	188.87	186.31 — 191.44	213.47	210.91 — 216.03	245.39	242.33 — 248.45	280.65	278.10 — 283.20	305.33	302.06 — 308.59
8	197.56	194.96 — 200.15	222.59	219.96 — 225.21	255.22	252.03 — 258.41	291.08	288.48 — 293.69	315.89	312.64 — 319.15

Chapter 7: CONCLUSIONS AND FUTURE WORK

7.1. Conclusions

In this thesis, we considered the small cell ultra-dense networks that are necessary to achieve high data rates for 5G. Backhaul is the major challenge for accomplishing the performance objectives for such networks. Our network model uses mm-wave wireless backhaul in multi-hop distributed architecture, which is similar to the 3GPP Release 16 IAB architecture. We examined the use of IAB as multi-hop networks delivering access traffic to the core.

In this research, we address the gateway location problem by selecting best locations for gateways. Our work simplifies and optimizes the distributed wireless backhaul and shows that efficiently locating gateways improves the backhaul network capacity. Toward this end, the research presented in this thesis introduces a novel approach to the gateway location problem with an effort of finding near optimal gateway locations effectively by minimizing the average number of hops and maximizing the backhaul network capacity. The developed approach, referred to as K-GA, combines the simplicity of *K*-means with the evolutionary improvement in the genetic algorithm. A distributed routing scheme was formed to calculate the average number of hops in the network and based on these average number of hops, backhaul network capacity is calculated.

To make our algorithm more robust and efficient we use an improved initial population using *K-means* clustering algorithm and design our own efficient fitness function. We also consider the effect of different parameters such as maximum generations, population size, crossover types, and mutation probability in our work and tested various control parameters by running simulations on several different network topologies.

We further formulate the *p*-median problem as integer linear program to obtain the optimal gateway locations and analyze the quality of outcomes by comparing K-GA method with the exact solution achieved in CPLEX optimization studio. Simulation results determine that the K-GA heuristic can solve the GLP very quickly with a small percentage deviation from the optimal solution. To demonstrate more, we compared the performance of proposed algorithm with well-known heuristic techniques such as *K*-means algorithm,

K-medoids algorithm, genetic algorithm, combination of *K*-medoids with genetic algorithm, and a baseline approach.

To prove the effectiveness of our proposed K-GA approach, we tested on three different distribution scenarios: Uniform distribution, bivariate Gaussian distribution and Cluster distribution. We compared all other approaches and optimal solution with K-GA and proved that K-GA provides better solution in all three distribution environments. Additionally, K-GA saved on average *95*% of execution time compared to optimal approach and provided results of ANH and BNC within *3*% of optimal solution.

We also investigate the possibility of bottlenecks in the backhaul network when number of small cells are too high, and the number of gateways are very few. To analyze the effect of various number of gateways, we consider a number of gateways ranging from 2 to 8 in 5G ultra-dense network and study their effect on average number of hops and backhaul network capacity in all distribution scenarios. The results show that small number of gateways and large number of small cells reduce the backhaul network capacity. The results also indicate that more gateways are required to achieve higher backhaul network capacity and to avoid congestion in ultra-dense backhaul networks.

7.2. Future Work

The following are possible future research directions which would extend and enhance the work presented in this thesis:

- As discussed in section 3.2, the large number of fully loaded small cells and few gateways may cause blockage at gateways by aggregating small cell access traffic through mm-wave wireless backhaul links in the distributed multi-hop wireless backhaul network. This possible presence of bottlenecks can be avoided by introducing a dynamic gateway allocation scheme. If too many small cells assigned to the available gateways and all gateways are overloaded by access traffic then such a scheme can add one or more gateways to the network and re-assign small cells among all gateways based on the minimum average number of hops. In summary, mm-wave backhaul with dynamic gateway allocation scheme helps to avoid the congestion issue in the backhaul network and add redundancy to the 5G ultra dense networks.

- As stated throughout the thesis, mm-wave is a promising solution for backhaul connections to support the high data rate traffic demands in ultra-dense networks. However, mm-wave communications have short range due to high path loss. To overcome this problem, we need to conduct comprehensive study on dynamic beamforming and include these results in finding the optimum GWs locations by decreasing the average number of hops.
- Our results indicate that we get approximately 2 to 4 minimum average number of hops in all distribution scenarios. However, to satisfy ultra-low latency requirements (for some applications), one may investigate a system where only one hop is allowed. Then we need to find optimum number of GWs and their locations and then use variable beamforming gains to adjust the small cell to GW range.
- Investigation of Integrated Access and Backhaul can be done where small cells are adjusted in size to match different users' distribution while optimizing the backhaul capacity and latency. Such a study would require a comprehensive probabilistic access traffic distribution.
- An important future direction is to include the possibility of deploying mobile GWs.

REFERENCES

[1] J. Thompson et al., "5G Wireless Communication Systems: Prospects and Challenges," *IEEE Communications Magazine*, vol. 52(2), pp. 62–64, 2014.

[2] N.H.M. Adnan, I. M. Rafiqul, and A. H. M. Z. Alam, "Massive MIMO for Fifth Generation (5G): Opportunities and Challenges," in proc. *International Conference on Computer and Communication Engineering (ICCCE)*, Kuala Lumpur, pp. 47-52, 2016.

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

[4] N. Bhushan et al., "Network Densification: The Dominant Theme for Wireless Evolution into 5G," *IEEE Communications Magazine*, vol. 52(2), pp. 82–89, 2014.

[5] M. Kamel, W. Hamouda, and A. Youssef, "Ultra-Dense Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 18(4), pp. 2522–2545, 2016.

[6] W. Feng, Y. Li, D. Jin, L. Su, "Millimeter-Wave Backhaul for 5G Networks: Challenges and Solutions," *Sensors*, vol. 16(6), 2016.

[7] 3GPP Release 16, [Online]. Available: https://www.3gpp.org/release-16.

[8] 3GPP TR 38.874, *NR; Study on Integrated Access and Backhaul*, [Online]. Available: https://www.3gpp.org/release-16

[9] X. Ge, S. Tu, G. Mao, C. Wang, and T. Han, "5G Ultra-Dense Cellular Networks," *IEEE Wireless Communications*, vol. 23(1), pp. 72–79, 2016.

[10] P. Cappanera, and M. Nonato, "The Gateway Location Problem: Assessing the Impact of Candidate Site Selection Policies," *Discrete Applied Mathematics*, vol. 165, pp. 96–111, 2014.

[11] Ansari, Abdollah, and Azuraliza Abu Bakar. "A Comparative Study of Three Artificial Intelligence Techniques: Genetic Algorithm, Neural Network, and Fuzzy Logic, on Scheduling Problem," in proc. *4th IEEE International Conference on Artificial Intelligence with Applications in Engineering and Technology*, pp. 31–36, 2014

[12] C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, NY, 2006.

[13] C. Sammut, and G.I. Webb, *Encyclopedia of Machine Learning*, Springer, New York, NY, 2011.

[14] E.W. Dijkstra, "A Note on Two Problems in Connection with Graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[15] J. Rodriguez, *Fundamentals of 5G Mobile Networks*, Wiley, 2015.

[16] J. Lun, "Wireless Backhaul Architectures for 5g Networks," *Doctoral Dissertation, The University of York,* United Kingdom, ProQuest Dissertations Publishing, 2017.

[17] T. Rappaport, *Wireless Communications: Principles and Practice*, Englewood Cliffs, NJ, USA, 1996.

[18] W. Roh et. Al. "Millimeter-Wave Beamforming as an Enabling Technology for 5G Cellular Communications: Theoretical Feasibility and Prototype Results," *IEEE communications Magazine*, vol. 52(2), pp. 106-113, 2014

[19] 3GPP TR 38.825, *Study on NR industrial Internet of Things (IoT)*, [Online]. Available at: https://www.3gpp.org/release-16.

[20] Y. Ni, et al. "Research on Key Technology in 5G Mobile Communication Network," in proc. *International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, pp. 199–201, 2019.

[21] B. Zhang, "Massive MIMO and Millimeter Wave Communications for 5G and Beyond Wireless Systems," *Doctoral Dissertation, University of Delaware*, USA, ProQuest Dissertations Publishing, 2019.

[22] Z. Pi and F. Khan, "An Introduction to Millimeter-wave Mobile Broadband Systems," *IEEE Communications Magazine*, vol. 49(6), pp. 101-107, 2011.

[23] H. Peng et al. "Ultra-Dense Network: Challenges, Enabling Technologies and New Trends," *China Communications*, vol.13(2), pp. 30-40, 2016.

[24] D. Astely, E. Dahlman, G. Fodor, S. Parkvall, and J. Sachs, "LTE Release 12 and Beyond [accepted from open call]," *IEEE Communications Magazine*, vol. 51(7), pp. 154-160, 2013.

[25] V. Chandrasekhar, J. Andrews, and A. Gatherer, "Femtocell Networks: a Survey," *IEEE Communications Magazine*, vol. 46(9), pp. 59-67, 2008.

[26] Fujitsu Network Communications Inc., *High-Capacity Indoor Wireless Solutions: Picocell or Femtocell* [white paper], 2013, [Online]. Available: https://www.fujitsu. com/us/Images/High-Capacity-Indoor-Wireless.pdf.

[27] S. R. Samal, "Interference Management Techniques in Small Cells Overlaid Heterogeneous Cellular Networks," *Journal of Mobile Multimedia,* vol.14(3), pp. 273-306, 2018.

[28] Nokia, *Deployment Strategies for Heterogeneous Networks* [white paper], 2015, [Online]. Available: https://resources.nokia.com/asset/200070.

[29] C. Ranaweera et al., "Design and Optimization of Fiber Optic Small-cell Backhaul Based on an Existing Fiber-to-the-Node Residential Access Network," *IEEE Communication Magazine*, 51(9), 62–69, September 2013.

[30] X. Ge, H. Cheng, M. Guizani, and T. Han, "5G Wireless Backhaul Networks: Challenges and Research Advances," *IEEE Network*, vol. 28(6), pp. 6-11, 2014.

[31] J. Lun, D. Grace, A. Burr, Y. Han, K. Leppanen, and T. Cai, "Millimeter Wave Backhaul/Fronthaul Deployments for Ultra-Dense Outdoor Small Cells," in proc. *IEEE Wireless Communications and Networking Conference workshops(WCNCW)*, pp. 187-192, Doha, 2016.

[32] Z. Gao, L. Dai, D. Mi, Z. Wang, M. A. Imran, and M. Z. Shakir, "Mm-wave Massive-MIMO-Based Wireless Backhaul for the 5G Ultra-Dense Network," *IEEE Wireless Communications*, vol. 22(5), pp. 13-21, 2015.

[33] C. Dehos, J. L. González, A. D. Domenico, D. Kténas, and L. Dussopt, "Millimeter-Wave Access and Backhauling: The Solution to the Exponential Data Traffic Increase in 5G Mobile Communications Systems," *IEEE Communications Magazine*, vol. 52(9), pp. 88-95, 2014.

[34] J. G. Andrews, T. Bai, M. N. Kulkarni, A. Alkhateeb, A. K. Gupta, and R. W. Heath, "Modeling and Analyzing Millimeter Wave Cellular Systems," *IEEE Transactions on Communications*, vol. 65(1), pp. 403–430, 2017.

[35] M. R. Akdeniz, Y. Liu, M. K. Samimi, S. Sun, S. Rangan, T. S. Rappaport, and E. Erkip, "Millimeter Wave Channel Modeling and Cellular Capacity Evaluation," *IEEE Journal on Selected Areas in Communications*, vol. 32(6), pp. 1164–1179, 2014.

[36] A. I. Nasr, and Y. Fahmy, "Millimeter-wave Wireless Backhauling for 5G Small Cells: Star versus Mesh Topologies," in proc. *28th International Conference on Microelectronics (ICM)*, Giza, pp. 85-88, 2016.

[37] Y. Cao, L. Zhao, Y. Shi, and J. Liu, "Gateway Placement for Reliability Optimization in 5G-Satellite Hybrid Networks" in proc. *International Conference on Computing, Networking and Communications (ICNC)*, Maui, 2018.

[38] J. Nie, D. Li, Y. Han, W. Fu, and G. Zhang, "The Method of Multiple Surface Gateways Positioning in UWSNs," in proc. *6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, Chengdu, China, 2010.

[39] M. Awadallah, and H. Aisha, "A Genetic Approach for Gateway Placement in Wireless Mesh Networks," *International Journal of Computer Science and Network Security*, vol. 15(7), pp. 11-19, 2015.

[40] T. Mahmoud, M. Girgis, B. Abdullatif, and A. Rabie, "Solving the Wireless Mesh Network Design Problem using Genetic Algorithm and Simulated Annealing Optimization Methods," *International Journal of Computer Applications*, vol. 96(11), 2014.

[41] A.M. Ahmed, A.H.A. Hashim, and W.H. Hassan, "Investigation of Gateway Placement Optimization Approaches in Wireless Mesh Networks Using Genetic Algorithms," in proc. *International Conference on Computer and Communication Engineering*, Kuala Lumpur, 2014.

[42] N. Palizban, "Millimeter Wave Small Cell Network Planning for Outdoor Line-of-Sight Coverage," *M.A.Sc. thesis, Systems and Computer Engineering, Carleton University*, Ottawa, Canada, 2017.

[43] P. Huang, and K. Psounis, "Optimal Backhauling for Dense Small-Cell Deployments using mmWave Links," *Computer Communications*, vol. 138, pp. 32-44, 2019.

[44] X. Ge, L. Pan, S. Tu, H. Chen, and C. Wang, "Wireless Backhaul Capacity of 5G Ultra-Dense Cellular Networks," in proc. *IEEE 84th Vehicular Technology Conference (VTC-Fall)*, Montreal, 2016.

[45] X. Ge, S. Tu, G. Mao, V. Lau, and L. Pan, "Cost Efficiency Optimization of 5G Wireless Backhaul Networks," *IEEE Transactions on Mobile Computing*, vol. 18(12), pp. 2796-2810, 2019.

[46] J. Reese, "Solution Methods for the p-median Problem: An Annotated Bibliography," *Networks*, vol. 48(3), pp. 125–142, 2006.

[47] S. Hakimi, "Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph," *Operations Research*, vol. 12(3), pp. 450–459, 1964.

[48] D. W. Matula, and R. Kolde, "Efficient Multi-Median Location in Acyclic Networks," presented at *ORSA-TIMS meeting*, Miami, FL, 1976.

[49] O. Kariv, and S. Hakimi, "An Algorithmic Approach to Network Location Problems, Part II: p-medians", *SIAM Journal on Applied Mathematics,* vol. 37, pp. 539-560, 1979.

[50] W.L. Hsu, "The Distance-Domination Numbers of Trees," *Operations Research Letters,* pp. 96-100, 1982.

[51] A. Tamir, "An $O(pn^2)$ Algorithm for the p-median and Related Problems on Tree Graphs," *Operations Research Letters*, Vol 19(2), pp. 59-64, 1996.

[52] R. Benkoczi and B. Bhattacharya, "A New Template for Solving p-median Problems for Trees in Sub-Quadratic Time," *Lecture Notes in Computer Science*, vol. 3669, pp. 271–282, Springer, Berlin, 2005.

[53] N. Mladenović, J. Brimberg, P. Hansen, and A. Moreno-Pérez, "The p-median Problem: A Survey of Metaheuristic Approaches," *European Journal of Operational Research*, vol. 179(3), pp. 927-939, 2007.

[54] S. D. de S. Silva, M. G. F. Costa, and C. F. F. Costa Filho, "Customized Genetic Algorithm for Facility Allocation using p-median," in proc. *Federated Conference on Computer Science and Information Systems (FedCSIS)*, Leipzig, Germany, pp. 165-169, 2019.

[55] L. Haoze, "Research on Artificial Intelligence Optimization Based on Genetic Algorithm", *Light Industry Science and Technology*, pp.77-79, 2012.

[56] E. Wirsansky, "Hands-on Genetic Algorithms with Python : Applying Genetic Algorithms to Solve Real-World Deep Learning and Artificial Intelligence Problems" in *Hands-on Genetic Algorithms with Python*, Packt Publishing Limited, 2020.

[57] C. Guoliang, "Genetic Algorithm and Its Application", *Posts and Telecommunications Press*, 1999.

[58] J. Holland, "Concerning Efficiet Adaptive Systems," in *Yovits,M.C.,Eds.,Self -Organizing Systems*, pp. 215 -230, 1962.

[59] F. Buontempo, "Genetic Algorithms and Machine Learning for Programmers : Create AI Models and Evolve Solutions," *The Pragmatic Bookshelf*, 2019.

[60] A. Kapoor, "Hands-On Artificial Intelligence for IoT: Expert machine learning and Deep Learning Techniques for Developing Smarter IoT Systems," *Packt Publishing Limited*, Birmingham, 2019.

[61] S.N. Sivanandam, and S.N. Deepa, "Terminologies and Operators of GA" in *Introduction to Genetic Algorithms*, Springer Berlin Heidelberg, 2008.

[62] B. K. Ambati, J. Ambati, and M. M. Mokhtar, "Heuristic Combinatorial Optimization by Simulated Darwinian Evolution: A Polynomial Time Algorithm for the Traveling Salesman Problem," *Biological Cybernetics*, vol. 65(1), pp. 31–35, 1991.

[63] F. G. Lobo, D. E. Goldberg, and M. Pelikan, "Time complexity of Genetic Algorithms on Exponentially Scaled Problems," in proc. *Genetic and Evolutionary Computation Conference (GECCO '00)*, pp. 151–158, Las Vegas, Nevada, USA, 2000.

[64] K. Deb,, et al. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II." *IEEE Transactions on Evolutionary Computation*, vol. 6(2), pp. 182–97, 2002.

[65] K. Krishna and M. Narasimha Murty. "Genetic K-Means Algorithm." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29(3), pp. 433–39, 1999.

[66] Tsai, Chun-Wei, et al. "A High-Performance Genetic Algorithm: Using Traveling Salesman Problem as a Case," *The Scientific World Journal*, vol. 2014, pp. 178621–178621, 2014.

[67] S. Yaram, "Machine Learning Algorithms for Document Clustering and Fraud Detection," in proc. *International Conference on Data Science and Engineering (ICDSE)*, Cochin, pp. 1-6, 2016.

[68] A. P. Tindi, R. Gernowo, O. D. Nurhayati, “Machine learning: Fisher fund classification using neural network and particle swarm optimization,” in proc. *Information and Communications Technology (ICOIACT)*, pp. 315-320, 2018.

[69] Xie, and S. Jiang, “A Simple and Fast Algorithm for Global K-means Clustering,” in proc. *Second International Workshop on Education Technology and Computer Science*, Wuhan, pp. 36-40, 2010.

[70] J. Qi, Y. Yu, L. Wang, and J. Liu, “K*-Means: An Effective and Efficient K-Means Clustering Algorithm,” in proc. *IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, Atlanta, GA, pp. 242-249, 2016

[71] K.A. Abdul Nazeer, and M.P. Sebastian, “Improving the Accuracy and Efficiency of the k-means Clustering Algorithm,” in proc. *World Congress on Engineering (WCE)*, London, UK, 2009.

[72] N. Makariye, “Towards Shortest Path Computation Using Dijkstra Algorithm,” in proc. *International Conference on IoT and Application (ICIOT)*, Nagapattinam, 2017.

[73] J. Donald, “A Note on Dijkstra’s Shortest Path Algorithm,” *Journal of the ACM*, vol. 20(3), pp. 385–88, 1973.

[74] H. Ortega-Arranz, et al. “The Shortest-Path Problem : Analysis and Comparison of Methods,” *Morgan & Claypool*, 2014.

[75] A. Madkour, et al. “A Survey of Shortest-Path Algorithms,” *arXiv.org, Cornell University Library*, 2017.

[76] M. Barbehenn, “A Note on the Complexity of Dijkstra's Algorithm for Graphs with Weighted Vertices,” *IEEE Transactions on Computers*, vol. 47(2), pp. 263-, 1998.

[77] U. Siddique, H. Tabassum, E. Hossain, D.I. Kim, “Wireless Backhauling of 5G Small Cells: Challenges and Solution Approaches,” *IEEE Wireless Communication,* vol. 22(5), pp. 22–31, 2015.

[78] S. Sun, T. S. Rappaport, R.W. Heath Jr., et al., “Mimo for Millimeter-wave Wireless Communications: Beamforming, Spatial Multiplexing, or Both?,” *IEEE Communication Magazine,* vol. 52(12), pp. 110–121, 2014.

[79] IBM ILOG CPLEX Studio IDE, Version: 12.10.0.201911271611, Build id: 201911271611, Licensed Materials - Property of IBM Corp. © Copyright IBM Corporation and other(s) 1998, 2019.

[80] M. Dunham et al., “Clustering”, in *Data Mining: Introductory and Advanced Topics*, Upper Saddle River, NJ: Prentice Hall, 2002.

[81] The MathWorks Inc., “MATLAB® Reference Guide,” Natick, MA, USA, 1992.

[82] H.P. Keeler, “Notes on the Poisson Point Process,” *Weierstrass Institute*, Berlin, Germany, Tech. Rep., 2016.

[83] Martinez, L. Wendy, and A. R. Martinez, *Computational Statistics Handbook with MATLAB*, Chapman & Hall/CRC, 2002.

[84] M. Jaber, M.A. Imran, R. Tafazolli, and A. Tukmanov, “5G Backhaul Challenges and Emerging Research Directions: A Survey,” *IEEE Access*, vol.4, pp. 1743–1766, 2016.

[85] Li, Xiang et al. “Impact on Genetic Algorithm of Different Parameters,” in proc. *Advances in Computation and Intelligence: Third International Symposium ISICA,* Vol. 5370, pp. 479-488, Wuhan, China, 2008.

Appendix A

This appendix presents an in-depth analysis of the effect of different configuration parameters on K-GA algorithm. This includes the variations for the number of chromosomes, different generations, different crossover types, various mutation probabilities, and time analysis for each parameter.

A.1. Tuning of K-GA Parameters

The convergence of genetic algorithm relates to many factors: the settings of initial populations, the design of fitness function, genetic manipulation, and the set of control parameters [85]. In order to satisfy above requirements in our proposed algorithm, as explained earlier in Chapter 4, we use better initial population with the help of *K-means* clustering algorithm and design our own efficient fitness function. To study the effect of control parameters in our work, we are considering different parameters for maximum generations, population size, crossover types, and mutation probability in our proposed K-GA algorithm. The efficiency of the algorithm depends on the choice of the genetic parameters. This section highlights the effect of different values and how much they impact on the behavior of the algorithm. We consider Monte-Carlo simulation for 130, 230, 330, and 430 Node densities of small cells in Uniform distribution scenario to analyze the effect of different GA parameters on the performance of K-GA.

A.1.1. Effect of the Different Number of Generations:

As explained in background section of algorithm, the selection, crossover, and mutation processes continue until the termination condition is met. One of the criteria for termination condition is the maximum number of generations. The larger the number of generations, the greater the probability of discovering a more near optimal solution. Higher number of generations, however, would also lead to longer execution times. The analysis shown in this section is performed for different values, i.e., 25, 50, and 100 number of generations for 100 different topologies of 130, 230, 330, and 430 node densities. The average number of hops for 130, 230, 330, and 430 node density and computational times are compared for various numbers of generations in Uniform distribution scenario.

Table A.1. Parameters to Evaluate the Effect of the Number of Generations

Parameters	Values
Number of Small Cells	130, 230, 330, 430
Number of Gateways	4
Population Size	256
Crossover Type	Single Point
Mutation Probability	1%
Maximum No. of Generations	*25,50,100*

Table A.2. Effect of Number of Generations

ANH and Time Analysis								
Number of Generations	Small Cell Density							
	130		230		330		430	
	ANH	Time (Seconds)	ANH	Time (Seconds)	ANH	Time (Seconds)	ANH	Time (Seconds)
25	2.79	4.38	2.52	5.57	2.43	9.86	2.39	13.76
50	2.77	7.45	2.51	9.85	2.42	12.43	2.38	15.59
100	2.76	17.76	2.51	20.77	2.41	28.2	2.38	32.37

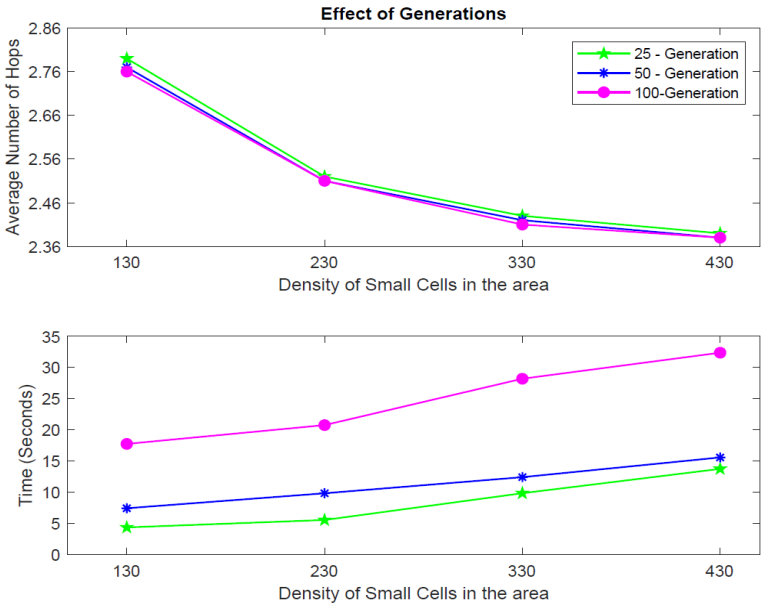

Fig. A.1. Effect of Number of Generations

The algorithm parameters that were used in the experiment are shown in Table A.1. Table A.2 represents the ANH values and time statistics with different generations. Fig. A.1 shows the graph of fitness value of average number of hops for the three different number of generations versus the different small cell node densities. Although, in general, the fitness tends to be better with number of generations, the improvement in fitness is not consistent. There is no guarantee that a higher fitness value will be achieved with a higher generation. Even if there is an increase in fitness, the increase is not very significant. At node density 130 and 330, generation of 100 reaches to maximum fitness while node density 230 and 430 have the same results at 50 generations. At 25 generations , results are poor compared to 50 generations. This performance indicates that most of the time better fitness can be found without requiring a very large number of generations. As such, an increase in the very larger number of generations will not have much effect on the results. On the other hand, the total execution time increases more with higher number of generations as shown in Fig. A.1.

A.1.2. Effect of Different Population Sizes:

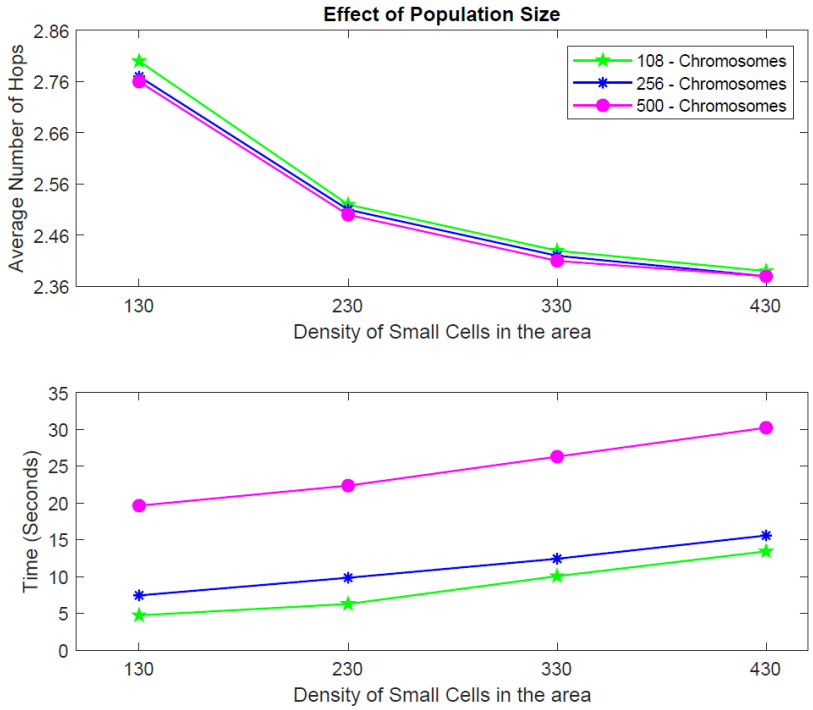

Fig. A.2. Effect of Number of Population Sizes

The initial population is considered from a large set of design space in order to reduce computational complexity. Ideally, the larger the initial population, the better the results as it is easier to explore more possibilities. To examine the impact of the number of chromosomes on the execution of algorithm, the simulation is run with different population sizes. As discussed in phase 2 in Chapter 4, K-GA selects population size based on the combination of the small cells nearest to each gateway. The population sizes considered for analysis are 108 (i.e., 3×3×3×4), 256 (i.e., 4×4×4×4), and 500 (i.e., 5×5×5×4). We consider 108 chromosomes for small population size, 256 chromosomes for moderate population size and 500 chromosomes for large population size. The algorithm parameters that were used in the experiment of different population sizes is shown in Table A.3. Table A.4 represents the ANH values and time statistics with multiple population sizes. Fig. A.2 shows the graph of the average number of hops with 108, 256, and 500 number of chromosomes.

Table A.3. Parameters to Evaluate the Effect of Different Population Size

Parameters	Values
Number of Small Cells	130, 230, 330, 430
Number of Gateways	4
Population Size	*108, 256 , 500*
Crossover Type	Single Point
Mutation Probability	1%
Maximum No. of Iterations	50

Table A.4. Effect of Different Population Size

ANH and Time Analysis								
Number of Chromosomes	Small Cell Density							
	130		230		330		430	
	ANH	Time (Seconds)	ANH	Time (Seconds)	ANH	Time (Seconds)	ANH	Time (Seconds)
108	2.8	4.74	2.52	6.29	2.43	10.07	2.39	13.44
256	2.77	7.45	2.51	9.85	2.42	12.43	2.38	15.59
500	2.76	19.65	2.50	22.37	2.41	26.31	2.38	30.24

Fig. A.2 represents the graph of average number of hops for multiple chromosomes for different node densities ranging from 130 to 430. In all node densities, a higher number of chromosomes results in better fitness value. However, at higher node density there is a limit on the fitness after which the increase in number of chromosomes would no longer result in significant increase of fitness value. For node densities 130, 230, 330 ANH keeps decreasing with the increase of population size while for node density 430 ANH is not reduced after 256 chromosomes. Similar to the number of generations, an increase in number of chromosomes also increases the total execution time as shown in Fig. A.2. However, execution time increase gap is more between 256 and 500 chromosomes compared to the execution time gap between 108 and 256 chromosomes. Therefore, it is important to find an ideal population size in order to have a good balance between better fitness value and execution time.

A.1.3. Effect of the Different Crossover Types:

As mentioned earlier, crossover is the process to produce new off-springs (children) from fitted individuals (parents). In this section two important methods, namely single point crossover, and two-point crossover are implemented, and their performance is compared. The advantage of single point crossover is the simplicity of the method. The analysis for the effects of crossover types are performed for 130, 230, 330, and 430 node densities. The algorithm parameters that were used in the experiment of crossover techniques is shown in Table A.5. Table A.6 represents the ANH values and time statistics with two different crossover techniques. Fig. A.3 shows the results for the effects of crossover types.

Table A.5. Parameters to Evaluate the Effect of Different Crossover Types

Parameters	Values
Number of Small Cells	130, 230, 330, 430
Number of Gateways	4
Population Size	256
Crossover Type	*Single Point , Two-Point*
Mutation Probability	1%
Maximum No. of Iterations	50

Table A.6. Effect of Different Crossover Types

ANH and Time Analysis								
Crossover Types	**Small Cell Density**							
	130		**230**		**330**		**430**	
	ANH	**Time (Seconds)**	**ANH**	**Time (Seconds)**	**ANH**	**Time (Seconds)**	**ANH**	**Time (Seconds)**
Single Point	2.77	7.45	2.51	9.85	2.42	12.43	2.38	15.59
Two - Point	2.77	9.34	2.51	11.92	2.42	14.32	2.38	17.53

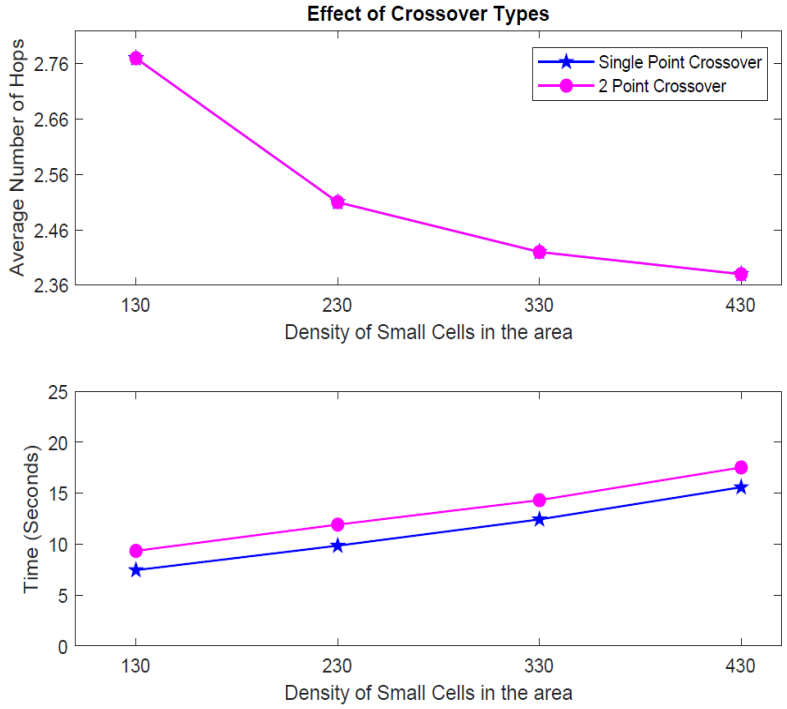

Fig. A.3. Effect of Crossover Types

The results illustrate a very similar performance for average number of hops between the single point crossover and two-point crossover method. In terms of runtime, however, the single point crossover requires less time to perform with respect to the two-point crossover. This is expected, since in the second case, the algorithm spends more time to generate two points and break the chromosome from those points to perform two-point

crossover operation. According to time analysis results for both methods, single-point crossover is considered to be more efficient and simplest method for crossover operation.

A.1.4. Effect of the Different Mutation Probabilities:

As stated earlier, based on the mutation probability, mutation function adds randomness to the chromosomes. Ideally, value of mutation probability should be kept low otherwise GA will change to random search. Mutation probabilities considered for observing the effect on the performance of the algorithm are 0.01 (1%), 0.02 (2%), 0.05 (5%), 0.1 (10%), 0.25 (25%), and 0.50 (50%).

Table A.7. Parameters to Evaluate the Effect of Different Mutation Probabilities

Parameters	Values
Number of small cells	130, 230, 330, 430
Number of Gateways	4
Population Size	256
Crossover Type	Single Point
Mutation Probability	*0.01, 0.02, 0.05, 0.10, 0.25, 0.50*
Maximum No. of Iterations	50

Table A.8. Effect of Different Mutation Probabilities

ANH and Time Analysis								
Mutation Probability	Small Cell Density							
	130		230		330		430	
	ANH	Time (Seconds)	ANH	Time (Seconds)	ANH	Time (Seconds)	ANH	Time (Seconds)
1%	2.77	7.45	2.51	9.85	2.42	12.43	2.38	15.59
2%	2.77	8.99	2.51	11.06	2.42	12.89	2.38	18.35
5%	2.77	9.37	2.52	11.82	2.42	13.18	2.38	19.4
10%	2.79	10.04	2.51	12.56	2.42	14.85	2.38	21.63
25%	2.81	11.35	2.52	14.24	2.42	17.66	2.38	23.16
50%	2.80	13.4	2.51	19.77	2.42	27.87	2.38	31.82

The algorithm parameters that were used in the experiment for probabilities of mutation function are shown in Table A.7. ANH values and time statistics with different mutation probabilities is given in Table A.8.

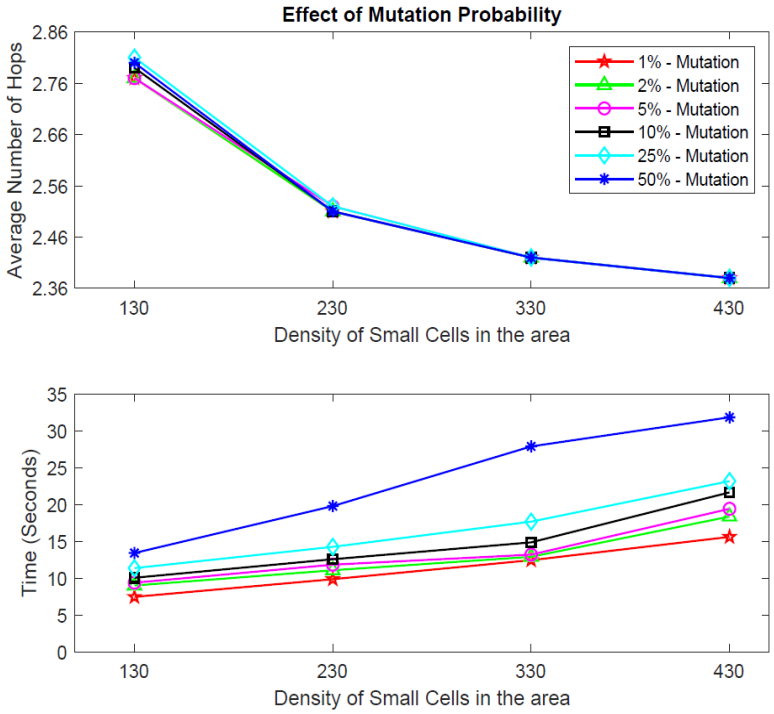

Fig. A.4. Effect of Mutation Probabilities

The result for this experiment is given in Fig. A.4. The results show that there is minor change in fitness value with different mutation probability on low node densities of 130 and 230 while for high node densities 330 and 430 there is no change in the fitness value with different mutation probabilities. However, time analysis indicates that the execution time linearly increases with the increase of mutation probability.

To decide the genetic parameters for our K-GA algorithm, we ran few experiments on different parameters and selected the ones that give the best results in terms of execution time and fitness value. The conclusions of the analysis can be summarized as follows:

- The number of generations plays a role and affects the fitness value to some extent, but it also plays a major role in the runtime. In this experiment, a generation value of 50 seems to give a good balance between better fitness value and execution time.

- The use of higher population size allows the algorithm to reach higher fitness values. However, increase in population size also increase the time exponentially. Based on the fitness and execution time, 256 is a moderate chromosome size that seems to be better fit for K-GA implementation.
- There is no effect on fitness value for different crossover methods, but single point crossover is favored because of its simplicity and less execution time.
- The fitness values vary based on different mutation probability at lower node densities. However, at higher node density mutation probability does not help to change fitness value. Higher value of mutation probability leads to higher execution time. The 1% mutation probability gives a reasonably good result for all given node densities, in terms of execution time and fitness value.

Based on above experiments and analysis for various genetic parameters, we decide to use 50 generations, 256 chromosomes, single point crossover method and 1% mutation probability for the performance evaluation of our proposed K-GA algorithm in Chapter 6.