Scientific Research Publishing

Journal of Computer and Communications, 2021, 9, 1-23
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Precoding Real-Time Buffer Based Self-Healing Solution for 5G Networks

Tamer Omar, Thomas Ketseoglou, Omar Naffaa, Asatur Marzvanyan, Connor Carr

Department of Electrical and Computer Engineering, California State Polytechnic University, Pomona, USA
Email: tromar@cpp.edu, tketseoglou@cpp.edu, omnaffaa@cpp.edu, amarzvanyan@cpp.edu, ccarr@cpp.edu

How to cite this paper: Omar, T., Ketseoglou, T., Naffaa, O., Marzvanyan, A. and Carr, C. (2021) A Precoding Real-Time Buffer Based Self-Healing Solution for 5G Networks. *Journal of Computer and Communications*, **9**, 1-23.
https://doi.org/10.4236/jcc.2021.96001

Received: March 31, 2021
Accepted: June 5, 2021
Published: June 8, 2021

Abstract

In order to meet key requirements imposed by transitioning to a 5G network, new network management techniques must be employed in order to increase network reliability and efficiency. Most notably, it is important that a Self-Organizing Network (SON) is able to recover autonomously from network failure or congestion through Self-Healing procedures (*i.e.* autonomous detection, diagnosis, and correction). This paper aims to develop a self-healing algorithm that can effectively "heal" a 5G network by testing a proposed self-healing algorithm within a network simulator that adheres to current 5G standards. The simulator developed in this paper aims to model a network of small cells that can inherit one of multiple states (healthy, congested, and failing) to validate the effectiveness of a programmed self-healing algorithm in recovering a simulated network. Results show that the application of a self-healing in a network is able to resolve issues related to Quality of Service (QoS) and reduced network data rates in portions of a network that are in a partially congested or failing state.

Keywords

Self-Healing, MIMO-Beamforming, Millimeter Wavelength, Base Station, Latency

1. Introduction

The transition to a 5G network requires an upgrade of our current physical infrastructure in order to reliably provide between 1 - 10 Gigabits per second (Gb/s) with less than 1 millisecond of latency through the use of Multiple-Input and Multiple-Output (MIMO) beamforming antennae that will significantly reduce wasted power and allow for more reliable communication, Millimeter WaveLength (mmWave) to achieve higher data rates, and new network manage-

ment techniques to ensure network reliability. Notably, networks must be able to recover autonomously from network failure through a process called Self-Healing. Self-Healing processes are critical for these future networks because they will likely often be comprised of a dense amount of smaller cells, whereas older networks had base stations more spread out and spaced apart. With the greatly increased number of cells, dealing with network failure in person or even remotely becomes impractical, and thus it will be ideal for the network to recover from failure on its own while maintaining the Quality of Service (QoS) for users on the network.

MIMO antenna technology was introduced into many wireless communications systems including 4G LTE to improve signal performance [1]. Multiple antennas are both used as a transmitter and receiver to provide improvements in data rate and the Signal-to-Noise-Ratio (SNR) without causing interference. Through the use of multiple antennas, MIMO is able to utilize the multiple path propagation that exists to provide improvements on signal performance. Massive MIMO is an extension of MIMO that improves performance by relying on signal processing to find the best and fastest way to route data to their destinations as opposed to a fixed number of physical antennas. With more antennas to utilize, massive MIMO provides faster and better spectral efficiency. In the experimental results provided in [2], the group achieved an increase in rate of efficiency that was 22 times better than existing 4G networks. Despite the increase in cost associated with Massive-MIMOs compared to standard MIMO configurations its benefits outweigh its costs.

5G networks are being designed to provide users throughput in the tens of Gb/s; in order to achieve this, higher frequency/shorter wavelength solutions are required. The mmWave band extends from 30 GHz to 300 GHz and offers numerous advantages over existing technologies [3]. Additionally, mmWave lends itself well to use with beamforming MIMO antennae, resulting in narrow beams at high frequencies that have very low attenuation losses and thus require less power [4].

To properly model a network using these parameters, several channel conditions need to be resolved for a realistic result. These include the SNR for a given channel, the data rates that can be provided to the user, the power consumption of the transmitter, and the received power. It is also important to have a model for determining the path loss exponent for a given environment because it will impact the channel's data rates and power consumption, which is important in calculating the SNR [5]. It can be useful to examine the Consumption Factor (CF) describing the maximum data rates that can be achieved for a given power consumption as a function of the channel conditions. It is desirable to maximize the ratio between the data rates achievable by the system and the power consumed by the system [6]. There are a great number of variables to consider in accurately modeling the physical nature of modern wireless networks. Modeling difficulties due to the inherently high scattering of mmWave, and the depen-

dence on the specific environment on the performance of the system [7], requires the simulator to maintain a certain level of flexibility so that multiple methods of channel generation can be used for different simulation profiles.

A self-healing, self-optimizing network is organized throughout a layered management system. Self-healing will focus on the Radio Access Network (RAN) layer, which includes base stations with the capability of transmitting conditional information for any self-healing functions required in the process. Base stations communicate with a controller-level Element Manager (EM), which collects necessary information to provide self-healing. Centralized control happens at the Core Network (CN) layer that manages the system at a high level by overseeing processes for centralized tasks, including monitoring base station conditions. The CN layer includes Network Manager (NM) controllers which communicate with lower-level Domain Manager (DM) controllers from the RAN layer. These two controllers have similar functions, but they differ in the management level that they operate in.

The general 5G network will be based on small cells. A relevant approach entails using a phased array of antennas at the Macro-cell Base Station (MBS), where it will be able to achieve massive MIMO beamforming [8]. The network's implementation method focuses on modular programming to allow for ease of management by having an adaptable system where customization and changes could be made dynamically to the inner functionality of the system. For example, different SNR calculations or scheduling and beamforming algorithms could be tested and implemented for optimization [9]. Further, the higher-level controllers would maintain global control of the network, which provides management optimization regarding resource allocation and base station scheduling [10].

The transition from 4G to 5G will include a migration from base stations to small cells, which would be placed much more densely in a given area and result in a significantly higher amount of total towers to manage. As mentioned in [11] approximately 23% to 26% of revenue acquired from mobile cellular network revenue is spent on operation costs associated with the network, which is a cost that will increase due to the additional infrastructure that would have to be implemented in a 5G system. The solution to this issue is to implement a Self-Organized Network (SON) that would automate the network management process in 3 high level stages as described by [12]: self-configuration, self-optimization, and self-healing.

Based on the research conducted in [13] self-healing can be classified into 3 main phases: detection, diagnosis, and compensation. From these parameters, a proposed self-healing model can be implemented as 5 modules in a way that would meet the 3 defined objectives. Our model for the self-healing process would include fault monitoring, fault detection, fault diagnosis, system compensation, and system recovery. Fault monitoring functions include performing self-healing monitoring and diagnosis functions as suggested by [14] and [15].

The self-healing functions would also perform comparisons with our operational measurements in fixed time intervals, and if a fault is detected, it is the responsibility of this portion of the model to generate a fault detection alarm. The alarm signal will notify system operations and deliver critical information needed to declare a fault occurrence, determine the nature of the fault, and specify if it should be Automatically Detected and Automatically Cleared (ADAC) or Automatically Detected and Manually Cleared (ADMC). Next, it will trigger diagnosis procedures if it is determined that a fault has occurred. Finally, prior to performing system compensation, information is gathered to identify the most probable cause of the fault, a severity level is assigned to the fault, and data is gathered for analysis to improve the system model. Once this is complete, a signal is sent that would initialize a compensation procedure and possibly a recovery procedure if the fault is severe.

At this stage in the model, the system will attempt to recover itself and provide compensation if necessary. System compensation includes surveying the ability of neighboring cells to share unused available resources, and then use the unallocated resources to compensate for the faulty cell until a full recovery is made. Once compensation is complete the digital system manager will return the resources back to their respective small cell. System recovery entails performing available tasks that can be used to repair the system, and if these automated procedures do not resolve the issue system recovery is responsible for initiating an ADMC to get the resources needed to fix the problem manually. This should resolve the issue, and as a result, the alarms will be cleared, normal operation will resume, and a fault report will be generated with information pertinent to future system diagnosis to improve cell outage management using data analytics as described by [16].

This work aims at developing a self-healing algorithm that can be used to efficiently "self heal" the network along with a 5G Self-Healing network simulator that will accurately model channel conditions and downlink communications between small cells and User Equipment (UE). This paper will present a method for self-healing using a simulator with constraints that would be imposed within a standard 5G network. This will be done by configuring the simulator with various scenarios and observing how the network management system will assess and resolve issues during run-time.

The remainder of the paper is organized as follows. Section II provides a general overview of the 5G simulator architecture, including initialization of the system, simulation stage, and outputs produced by the simulator. Section III describes the initialization phase for the simulator. In Section IV the back end, and how the system adheres to a 5G network architecture, is discussed. Section V describes how the throughput (*i.e.* the maximum data rate) and power are calculated for each user object in the network. Section VI goes over the design of the self-healing algorithm and how the instantiated network manager performs self-healing. Finally, section VII discusses the results of the simulation and pro-

vides a visualization of the self-healing performance.

2. System Architecture of the 5G Network Simulator

2.1. Simulator Architecture

The system is composed of 3 stages, which are 1) Initialization, 2) Simulation, and 3) Output. The initialization stage was implemented with a Graphical User Interface (GUI), which allows the user to create a network environment of small cells that are represented as interconnected hexagons by clicking to build a structure of the cell map for testing. Once this portion of the initialization is complete, the user is directed to the next window and can begin specifying parameters to be used in the simulation. Some examples of this include the number of antennae within the small cell and the number of transceivers per antenna.

After setting the parameters the simulation stage can begin. In the simulation stage the small cell structure is formed with appropriate network conditions based on parameters passed from the GUI as well as those taken from a CSV sheet containing accurate data rates and SNR values generated using MATLAB to simulate 16 QAM precoding. The environment is run for a specified amount of time, allowing the ability to analyze network conditions in the program in real time while also producing output log files that summarize how the system operated during run-time for data analytics once the simulation terminates. Specifically, the output of the simulation will be analyzed in real-time using data captured for a specified amount of time within the system (e.g. the most recent 10 seconds) to determine the condition of the network simulation at any given time without having to halt the simulation. This functionality is detailed further in Section VI.

Figure 1 summarizes the process above at a high level.

Figure 2 shows a process diagram for the initialization stages using the Graphical User Interface.

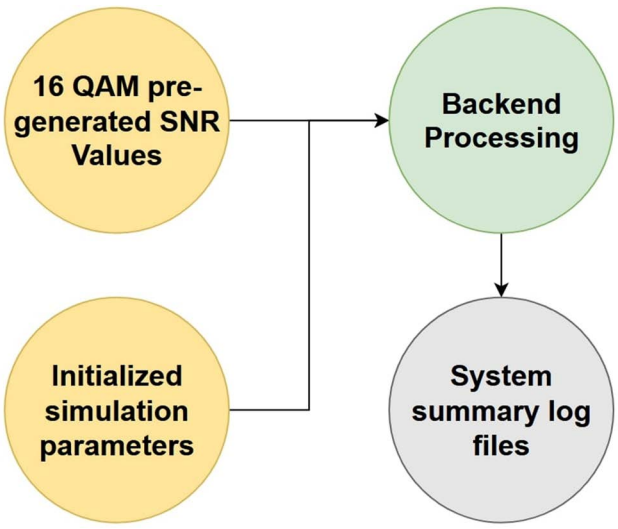

Figure 1. System architecture.

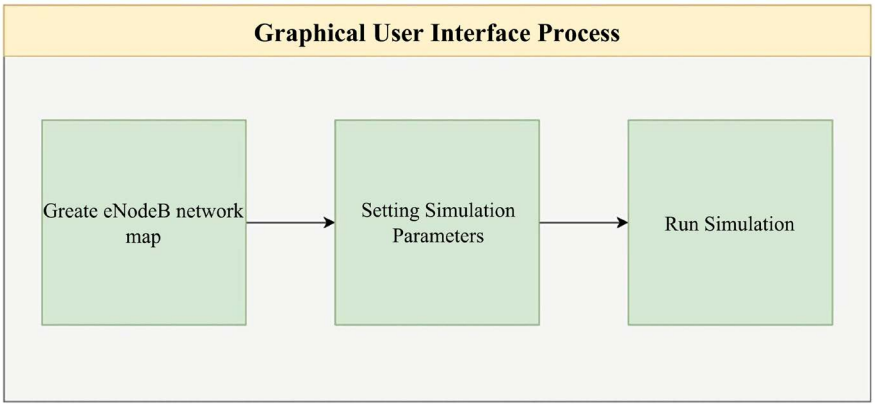

Figure 2. Graphical user interface block diagram.

2.2. Integration Reference Point Manager

The Integration Reference Point (IRP) manager is an entity modeled within the simulator that is used to represent the network manager of the simulator that will handle self-healing functions within the system, as described in [17]. The IRP manager functions as a high-level observer within the simulator that collects information from the system, analyzes available data in real-time, and applies an implemented self-healing algorithm through instructions sent to IRP agents within the simulator. In order to adhere to the modular scheme that is employed for the simulator, the IRP manager is implemented as a C++ class titled "IRP-Manager", which contains different functions to perform required self-healing operations. Key components of the IRP manager include functions that allow the simulator to detect, diagnose, and recover a failing or congested base station, but the flexible nature of this class allows it to accommodate new processes that may be required by the self-healing model at a future point in time.

3. Creating a Network Map

In the first stage of the simulator, the user is presented with a window containing a single hexagon, representing the first small cell with a blue number denoting the station number. Additional small cells can be placed by left clicking an edge of an adjacent hexagon. To change which small cell is selected the user can left click on any existing small cell and the clicked-on small cell will be selected. **Figure 3** shows one arbitrary configuration.

The state of each hexagon is represented by a different color, which is set by left clicking until the appropriate color appears. The states and their corresponding colors are listed below.

- Green = Healthy
- Yellow = Congested from Users
- Orange = Congested from Demand
- Red = Failing

The sample configuration in **Figure 3** shows a scenario with small cells that have different conditions (denoted by the color of the base station). We can see

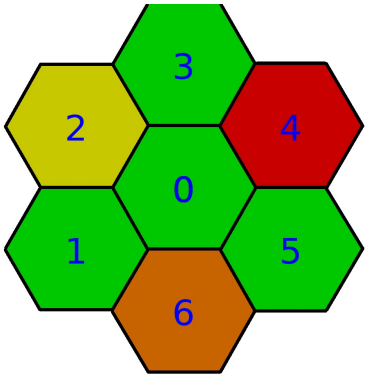

Figure 3. First stage initial screen with arbitrary configuration.

in this example small cells 0, 2, 4, and 6 are considered healthy. Additionally, small cell 1 is congested from the amount of users on the network, small cell 3 is congested by the amount of data being demanded from each user, and small cell 5 is failing.

Additional parameters that can be set for each small cell individually through the GUI include the ability to set an alert level that can be used to determine when the small cell is nearing congestion, and a congestion level that signifies when the small cell is actually congested. As an example, if small cell 1 is modified to be in the congestion state the cell condition and the time needed to reach that condition will be set by the user, which allows the system to "ramp up" to the selected condition as opposed to jumping up to that level of congestion immediately. After the user has created a setup, they can proceed to the next stage. The locations, states, and numbers of the stations are passed to the second stage in order to create a final "bundle" of information that will be used to create and run the simulation environment.

After instantiating the small cell network structure in the first stage of the simulation setup process the operator would then need to define initial parameters for the simulation to run. These parameters are the grouped into 3 categories:

- Network Parameters
- Simulation Parameters
- Self-healing Parameters

3.1. Network Parameters

The first set of parameters needed are used to initialize the simulation. This section is used to define the side length of each small, the number of antennas per small cell, the number of transceivers on an antenna, and the number of UEs per antenna. It is important to note that the small cell side length parameter (denoted as "BS Side Length" in **Figure 4**) only determines the amount of space between cells where UEs can be placed. Instantiated objects are placed on a Cartesian coordinate system, and each entity can only occupy one specific location. The small cell side length helps scale the system accordingly, both in terms of distance and resolution. However, within our model the side length of a small

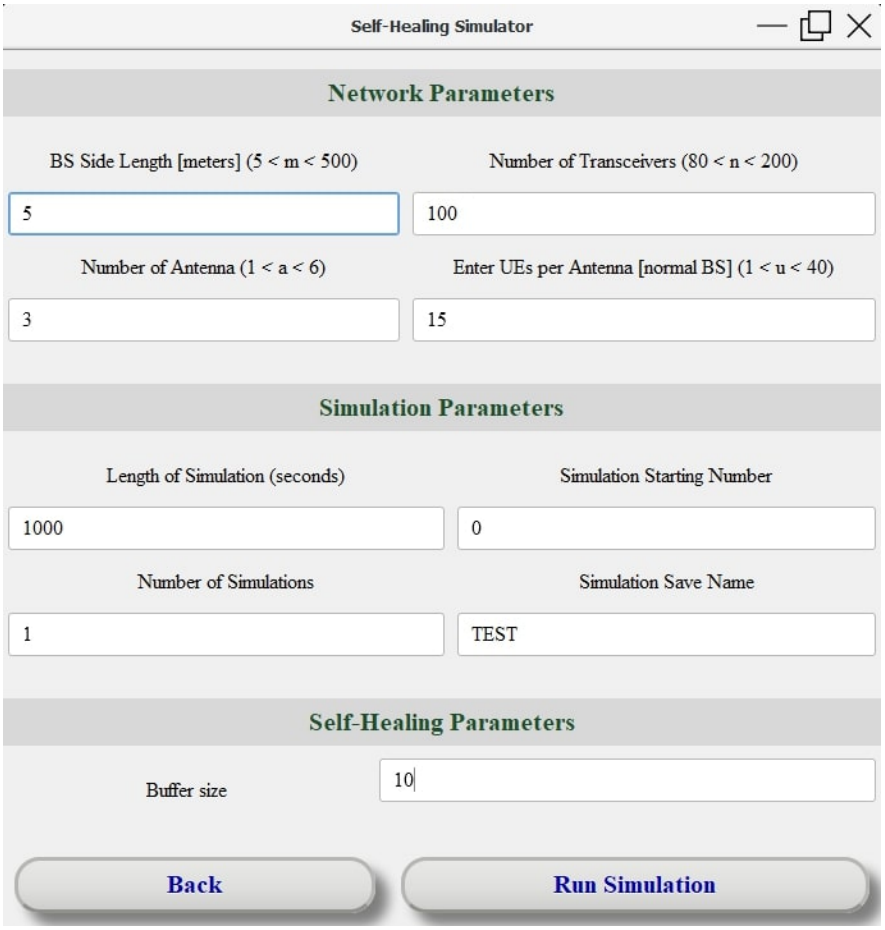

Figure 4. Full parameter window view.

cell is equated to a meter. The parameters, as noted on the GUI, must be specified within a specific range in order to be valid. The parameters in Figure 4 pertain to the network environment setup since all the objects specified are models of physical components of a network. Items specified here are physical entities instantiated as objects (e.g. a small cell is a cellular tower while a UE could be a computer or cell phone).

3.2. Simulation Parameters

The next set of parameters entered are the simulation parameters. This includes the length of the simulation, the simulation starting number (used in naming the output file), the amount of simulations to run in one batch, and the save name of the simulation. The length of simulation is measured in terms of "ticks", which is the fundamental unit of simulation time. A tick represents the time it takes for one set of instructions to be performed, but for the purpose of the simulation each tick is equated to 1 second.

3.3. Self-Healing Parameters

Finally, the self-healing parameter that would need to be specified for the net-

work is the size of a buffer used in the back-end of the system. The buffer structure built into the back-end is used to capture data during the simulator run-time for analysis in order to perform the self-healing procedure. Once these parameters are entered, the simulation can begin running by pushing the button titled "Run Simulation". Figure 4 is provided to show the parameter entry window.

4. Back-End Overview

This project is largely software-based; therefore, the design will mostly go through key portions of the back-end and how key constructs within a cellular network are modeled. By describing the organization of the simulator the hope is to thoroughly document the back-end design. Later sections will discuss how the throughput and power is calculated as well as details within the program regarding self-healing.

The simulator employs object-oriented design when creating new small cells—and the entities within a small cell—during the initialization stage. The Basestation class represents small cell objects and contains a number of antennas that will be used to connect user equipment to a base station as well as a list of user records to document the users connected within a particular small cell. Similarly, an antenna class acts as the program's way of implementing antenna objects within the small cell. The main components of an antenna in this model are its location, angle, and the transceiver list for a specific antenna object. The transceiver list contains a record of all transceivers on an antenna, and can be used to achieve functions such as determining users connected to an antenna. In addition to the antenna, a specific amount of user equipment objects populate the small cell based on the configuration parameters and serve as user devices on the network (e.g. a cellular phone, computer, or other internet connect device). They are attached to a transceiver on an antenna, which resides in the created small cell. A partial relationship diagram is shown below in Figure 5 to describe key components that are modeled in the back end and how they relate to one another.

Sending data from a small cell to the user equipment uses a certain amount of power based on the SNR, which is derived from by the data rate lookup table provided by a supplemental MATLAB script. When the simulation starts running the data rate table that provides the SNR values is loaded into memory while also providing an interface to use it with. The data rate table values are used to give a user connected to the small cells random SNR value between −12 dB and 12 dB in order to provide the network with a realistic model for random SNR value generation. The process the script takes to generate these SNR values is covered in more details in the section "Throughput and Power Calculations".

An environment controller is used to "ramp up" a small cell into a given non-healthy state. Currently, the program operator can specify when the small cell will reach congestion, the time it takes to "ramp up" and the final state of the

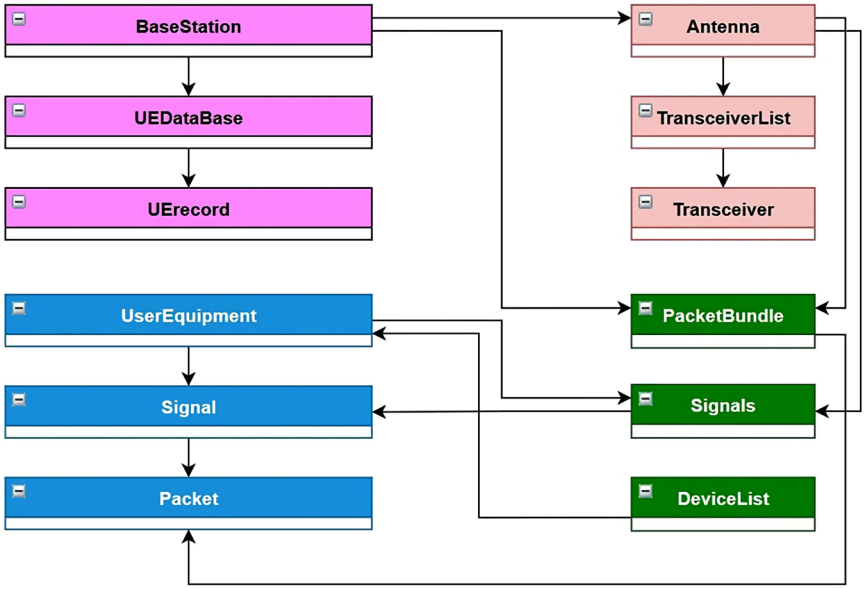

Figure 5. Entity relationship diagram.

small cell once the ramp up is complete. The purpose of this controller is to aid the simulator in modelling a realistic transition of the small cell from a healthy state to an alternate state without making the transition instantaneous.

Finally, the self-healing management functions are implemented as an IRP-Manager class, which contains functions for achieving self-healing and network management. One key function in this class is determining which small cell(s) need full self-healing (*i.e.* it is failing), which small cell(s) need help from self-healing (congestion), and which small cell is the most suitable to offload a user to. This algorithm will be covered in the following sections.

5. Throughput and Power Calculations

5.1. Data Rate Delivered to UEs

The calculations performed to obtain realistic SNR values are based on the work presented in [18], which describes an effective precoding method to be used for the simulator. To calculate the data rate a given UE can receive, the average received SNR $\left(\mathrm{SNR}_r\right)$ of each UE in the cell would be calculated first. The simulator models the SNR based on square propagation law. The breakpoint model [19] is invoked in order to determine the average received SNR at each UE. This model has been applied in realistic deployment scenarios such as the one presented in [20]. Then, for the example presented herein, SNR_r varies between −12 dB to +12 dB. Thus, the SNR(R) for a UE at distance R from the BS is calculated using the following equation

$$\mathrm{SNR}_r\left(R\right)=\frac{\mathrm{SNR}_0}{R^2}, \tag{1}$$

where SNR_0 is the SNR_r at a distance 1 m from the BS. For the example herein, SNR_0 = 12 dB.

The values for the Spectral Efficiency (SE) are generated using 16-QAM modulation in conjunction with optimized precoding [18] for a planar massive MIMO antenna array with 100 elements, and with the assumption of an inter-element (transceiver) spacing of half of a wavelength along with a frequency range between 20 - 40 GHz. In order to take into account multipath fading, there is a set of four different possible values for the SE for each UE, among which the simulator selects one randomly. Columns "Spec-1" - "Spec-4" contain valid SEs for a given SNR value, and differ only due to random fluctuations to the value in the channel that would occur when the SNR is measured in the real world. These four different classes of channels are all created using independent Rayleigh fading.

Knowing the SE for the channel allows the calculation of the data rate, DR, for the channel using the bandwidth (B) which is given by

$$DR = SE \cdot B, \tag{2}$$

where B is the RF bandwidth of the channel. Note that SE has a maximum value of 8 bits/s/Hz. This is due to the 16-QAM modulation scheme used in conjunction with Zero_Forcing Per-Group Precoding (ZF-PGP) [18]. This selection of modulation/precoding scheme was made because it allows for **Table 1** to be generated efficiently using MATLAB [21], while also being very spectrally efficient. Thus, since the SE value can reach up to 8 bits/s/Hz, the simulator example only allows small cells sizes, e.g., ranging from 5 to 100 m, in order to maintain the accuracy of the simulation model. Finally, the bandwidth used in the simulator is $B = 20$ MHz.

Table 1. MATLAB generated SNR values (−12 dB to 12 dB).

SNR	NxtSNR	Spec-1	Spec-2	Spec-3	Spec-4
−12.0	−11.9	3.24	3.31	3.04	3.19
−11.0	−10.9	3.51	3.59	3.31	3.46
−10.0	−9.9	3.79	3.86	3.59	3.74
−9.0	−8.9	4.07	4.15	3.87	4.02
−8.0	−7.9	4.35	4.43	4.15	4.3
−7.0	−6.9	4.76	4.84	4.55	4.7
−6.0	−5.9	5.19	5.28	4.97	5.14
−5.0	−4.9	5.51	5.59	5.28	5.45
−4.0	−3.9	5.81	5.89	5.6	5.76
−3.0	−2.9	6.1	6.18	5.9	6.05
−2.0	−1.9	6.37	6.45	6.18	6.32
−1.0	0.9	6.65	6.72	6.45	6.6
0.0	1.0	6.88	6.94	6.72	6.84
1.0	1.9	7.1	7.16	6.94	7.06
2.0	2.9	7.3	7.36	7.16	7.27

Continued

3.0	3.9	7.53	7.59	7.36	7.41
4.0	4.9	7.72	7.76	7.59	7.69
5.0	5.9	7.85	7.91	7.77	7.83
6.0	6.9	7.97	7.98	7.91	7.95
7.0	7.9	8	8	7.98	7.99
8.0	8.9	8	8	8	8
9.0	9.9	8	8	8	8
10.0	10.9	8	8	8	8
11.0	11.9	8	8	8	8
11.9	12.0	8	8	8	8

Table 1 is incorporated into the simulator as a lookup table that is created and inserted before running the simulation. It is important to note that the actual table increments the SNR value by steps of 0.1 rather than whole-number steps to more accurately model the variance present in real-life SNR_r values, which was sampled once every 10 rows to create Table 1.

5.2. Transmitted and Received Power Calculation

The power of the received signal P_r is proportional to the SNR at the generic UE positioned at distance R from the BS and is given

$$P_r(R) = \mathrm{SNR}_r(R) \cdot P_n, \tag{3}$$

where the total noise power $P_n = N_0 \cdot B$, with N_0 being the one-sided power spectral density (PSD) of the noise.

Based on our breakpoint propagation model [19], $P_r(R)$ can be written as

$$P_r(R) = \frac{P_0}{R^2}, \tag{4}$$

where P_0 represents the power at distance 1 m from the BS.

6. Self-Healing Process—Output and Analytics

6.1. Storing Output for Analytics

The self-healing implementation of the 5G Network Simulator starts by detecting the condition of a small cell. These conditions can vary from healthy, congested from user volume, congested from user demand, and failing entirely.

In a healthy cell, the cell is operating at normal conditions, where users who are connected to the base station receive the amount of data they are requesting (not experiencing lag or buffer issues). The next condition is congestion from a significant amount of data being requested by users connected to the small cell requesting large data packets (e.g. streaming in 4k, downloading games/videos, etc.). Congestion can also be caused by a large number of users connected to a single base station, even if the users are requesting lower amounts of data. In a

congested state, some users will receive all of the data that they are requesting, and some users will not receive any packets at all. Lastly, the final condition is "Failure" in which the base station is not sending any data/packets, so all users in the service area attempting to request data receive 0 packets.

Detecting the various conditions of a base station is an important aspect of the self-healing process. In order to treat either a congested or failing cell, a problem must be detected first. It is also important that the diagnosis of the cell is accurate as a misdiagnosis could prevent a congested or failing cell from recovering. In order to begin the process of detecting the condition of a small cell, a storage container is created first—referred to as a buffer from this point—to store the data of all the small cells for a specified amount of time. This time is specified as one of the parameters during initialization (see section "Setting Simulation Parameters"). The time the user inputs is the amount of data that this buffer will store at any given time in the simulation. Figure 6 depicts a visual representation of how the buffer works.

For demonstration purposes there is only one UE associated with each small cell, but in a typical simulation there can be hundreds of UEs associated with each small cell. As a result, there will be many more data points attached to each user that is used in the decision-making process of the IRP manager. The buffer begins by holding the data for time interval 0 - 6 (s). This indicates that the buffer will always hold 7 seconds worth of data, and that this buffer will then slide as time passes and hold data from 1 - 7 (s) as seen above. Although the buffer stores data continuously, the network manager will only analyze the buffer for the time specified during initialization in an effort to save time and power. For example, in Figure 6 although the buffer slides to hold data from 1 - 7 (s), the manager will only analyze the data once the 7 initial seconds from 0 - 6 (s) have passed and will analyze the contents of the buffer again at 7 - 13 (s). In order to detect the condition of the cell, the network manager will first analyze the requested data rates with the actual data rates received by the user and determine whether or not the cell is failing (all users in a base station are not receiving packets).

In order to detect congestion, Equation (1) is utilized to determine 2 parameters based on the calculated SNR range: the maximum data rate a small cell can provide DR_P and the data rate the user requested DR_E. DR_P is the maximum

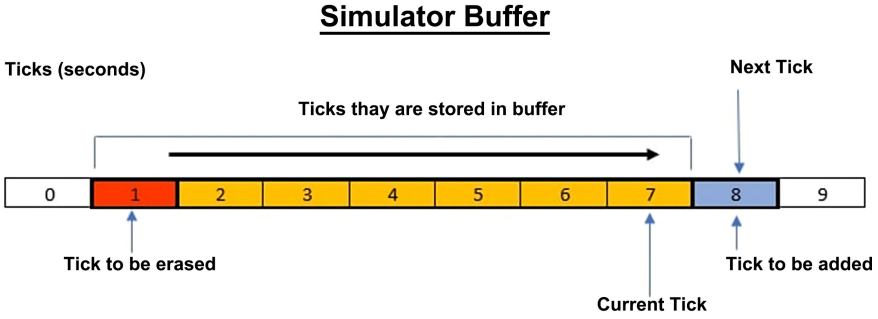

Figure 6. Buffer visualization.

number of packets that a single small cell is able to send out to UEs in its service area in a second. DR_E is the amount of data that an individual UE is requesting in a second. The expected data rate of all the users in an individual small cell is summed once every second in order to determine the total amount of data being requested. This yields the following equation to determine a given congestion level, where DR_M denotes a constant maximum data rate:

$$C = \frac{DR_P - \sum DR_E}{DR_M} \times 100\% \tag{5}$$

Once the Congestion Level is calculated, the percentage of data a small cell is still able to send out is recorded to allow the IRP manager to determine the status of each base station during the simulation. The network manager will first look at the expected and real data rate and compare them. If these values are 0 for the entire buffer, then the manager determines the cell to be failing and will initiate self-healing. Next, the network manager will look at the congestion level, and if the congestion level is at the "congestion state" (as specified during initialization) or above, then the manager will determine the cell to be in "congestion" and will begin self-healing. If the congestion level is at the "alarm state" (as specified during initialization), then an "alarm" will be triggered. The alarm signifies that the small cell raising the alarm is approaching congestion, and that this small cell will not be able to assist in the self-healing process. If the network manager does not detect the failing or congested cell, then the manager determines the cell to be "healthy" and will pursue no further action.

6.2. Analysis Methods

Along with producing a novel self-healing algorithm, this research will focus on analyzing the accuracy of the simulator and the results it produces. The implemented self-healing process should allow the network to recover from failure, so the extracted data can be examined to improve on the methods implemented. The key information is the downlink data rates within a small cell as they are the biggest indicator that failure has occurred. In a small cell that is in a "total failure" state, for example, the User Equipment will not receive data, so for a given self-healing technique the indicator of its efficacy will be how quickly and to what extent it was able to compensate for the loss of that small cell and restore the QoS to the user. This compensation will be evident in the outputs as users in a failing small cell would receive an acceptable data rate after self-healing as opposed to having a data rate of 0 that would occur without intervention. The same logic could also be applied to a small cell that is in a state of congestion.

6.3. Self-Healing Algorithm

Once the condition of the cell is determined to be either failing or congested, the IRP manager will then initiate self-healing. In order to perform self-healing, the system begins with a function titled "checkStatus()" that calculates the availability of all cells in that particular system. A cell is determined to be available if it is

not congested or failing. Additionally, a small cell will not be marked as available if it is in an alarm state, which signifies when healthy small cell can no longer provide aid since it is approaching congestion. Small cells that are eligible to be helpers will then be added to a small cell help list that marks them as usable in the self-healing process to alleviate load if needed. Conversely, if a cell is in either a congested or failing state, this function will then add them to a list of disabled small cells. The "checkStatus()" function is summarized in Figure 7 and Algorithm 1.

The cells in the help list are marked so that the "offloadUser()" function shown in Algorithm 2 can identify which small cell to aid based on "checkStatus()". If a small cell is failing they are prioritized to receive help, which notifies the system to prioritize these UEs first. Algorithm 2 takes the disabled list and begins analysis on which small cell can be aided by first determining the number of users that need to be offloaded to bring an individual small cell relief. It will then examine neighboring small cells within the help list and calculate the distance between the location of the UE to be offloaded and the closest small cell to the UE, storing the coordinates of the closest identified small cell to offload to.

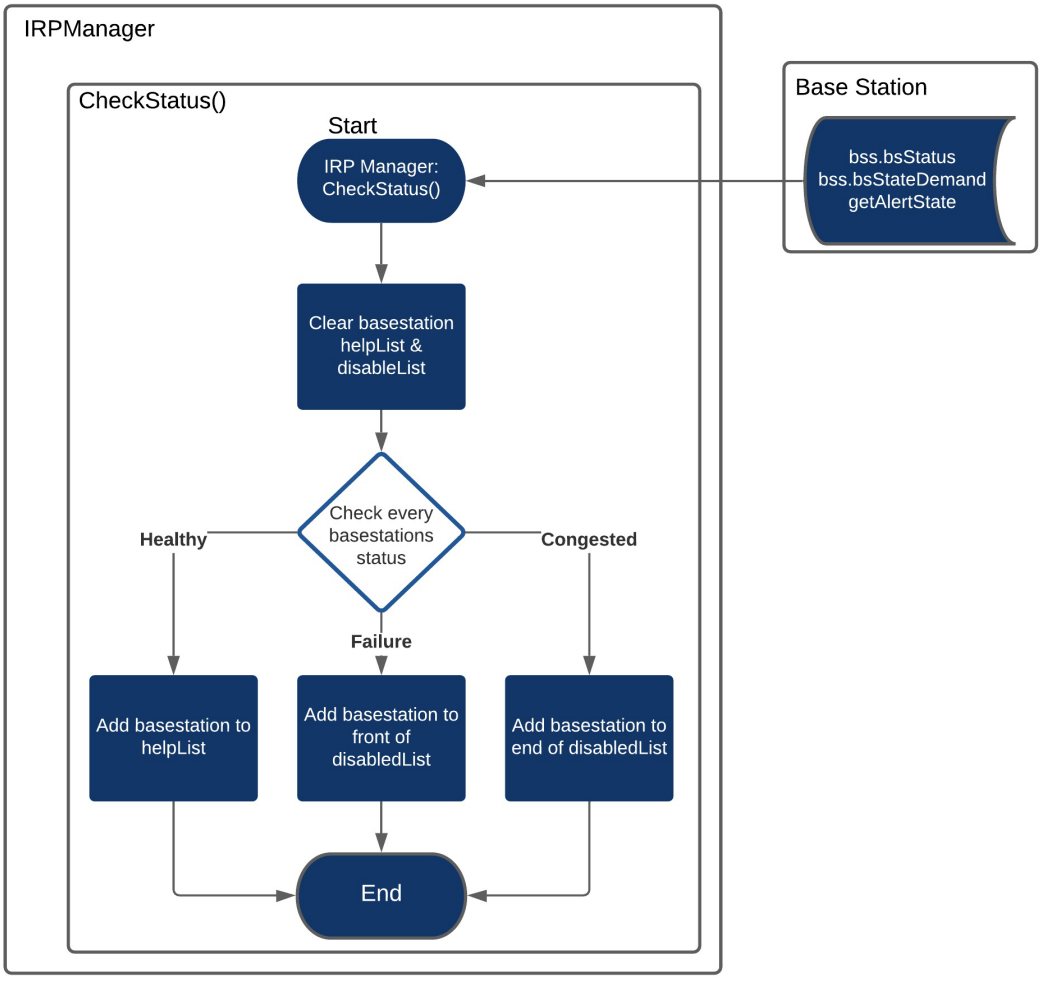

Figure 7. Algorithm 1 flowchart.

Algorithm 1. Self-healing detection.

```
1:  procedure IRP MANAGER:CHECKSTATUS()
2:      basestation helpList refreshes
3:      if basestation == Failure then
4:          basestation -> disabledList.begin()
5:      else if basestation == Congested then
6:          basestation -> disabledList.end()
7:      else if basestation == Healthy then
8:          basestation -> helpList.add()
9:      end if
10: end procedure
```

Algorithm 2. Self-healing recovery procedure.

```
    procedure IRP MANAGER:OFFLOADUSER()
2:      Initialize Failure
        Initialize Congested
4:      if BS == Failure || BS == Congested then
            amntToRemove = BS.userList.size()
6:      end if
        while amntToRemove > 0 do
8:          if failedRemoval < maxRemovalFails then
                exit loop
10:         end if
            ueToOffload = maxVal(BS.ueList.getDataRates())
12:         nearestBA = minDist(helpList)
            Offload User
14:         if offloaded == false then
                failedRemovalCnt = failedRemovalCnt + 1
16:             continue
            end if
18:         amntToRemove = amntToRemove - 1
        end while
20: end procedure
```

Once the nearest small cell is identified, the system will attempt to offload the UE to the nearest antenna sector that is facing the UE's location in order to optimize the offloading process by keeping the SNR as high as possible. Once completing the identification process, the IRP manager will then assign the UE with the highest data rate requested to this small cell in order to offload the UE with the highest demand and reduce the load on the congested or failing small cells. This process is summarized in Figure 8 and Algorithm 2, where "BS" represents a base station within the system.

The users will continuously be offloaded until either the unhealthy small cell is brought to a healthy state, or until an alarm is triggered on the aiding small cell. If the alarm is triggered due to approaching congestion the aiding small cell is removed from the help list and will provide no further relief. At this point, the

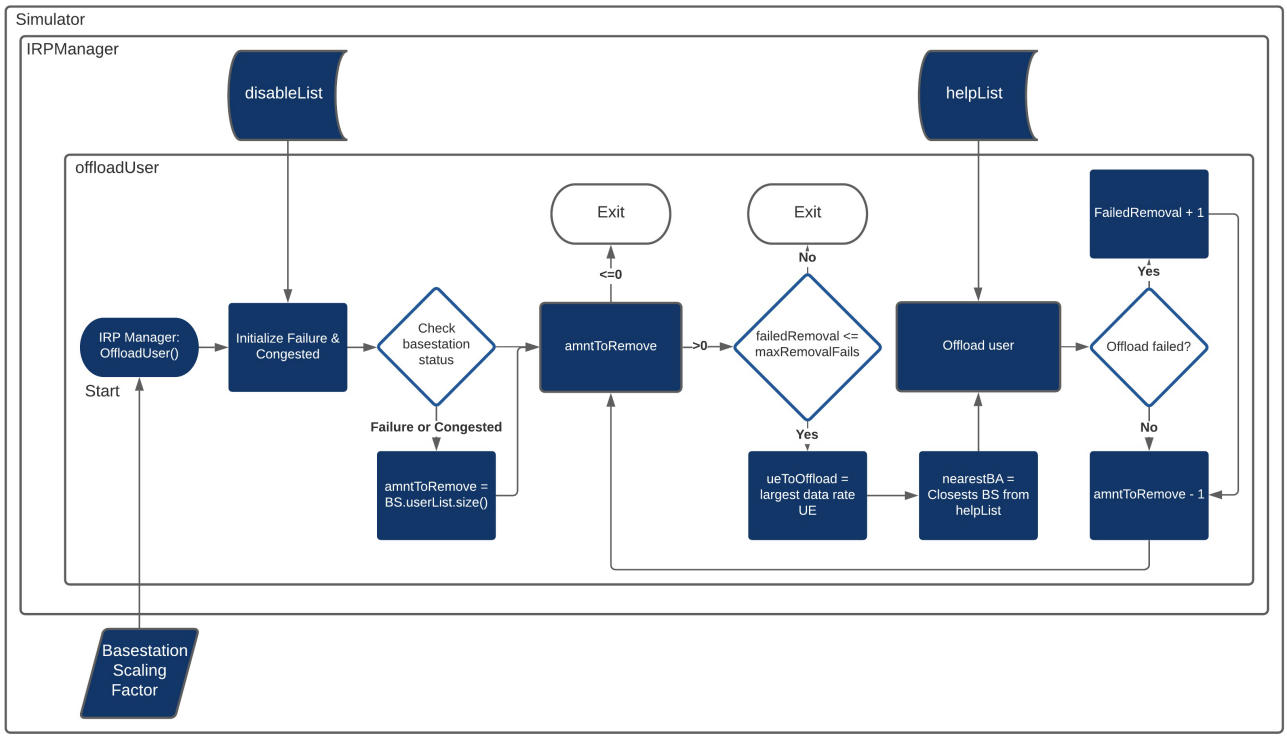

Figure 8. Algorithm 2 flowchart.

function will then assign users of the unhealthy small cell to the next closest available small cell to minimize the decrease in SNR and increase the efficiency of the system. The IRP manager will continue to implement these self-healing procedures until either all of the small cells in the network reach a "healthy" state, or until there are no remaining healthy small cells available to provide aid. In either case, this procedure will provide self-healing (if permitted by the general condition of the network), which will allow a user to determine whether the health of the network has improved compared its state prior to self-healing.

7. Simulation and Results

Using the 5G Network simulator as our ecosystem, the implemented IRP manager is successfully verified and is performing self-healing as intended. To verify the simulator can detect and compensate for both congestion and failing small cells, a honeycomb structure composed of 7 small cells was constructed during the initialization phase where one small cell of the 7 is set to either a congested or failing state. In this section, the 2 scenarios are examined to determine the efficacy of the self-healing algorithm.

7.1. Scenario 1—Congestion

In **Figure 9**, small cell 1 is congested by the number of users requesting data, while all the remaining small cells are healthy. Shown in **Figure 10** is a graph generated from the post-simulation log files. In **Figure 10**, the IRP manager is called to perform self-healing after 50 seconds within the simulator. Small cell 1,

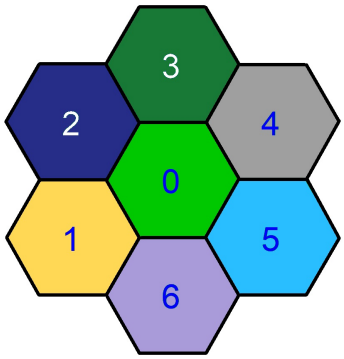

Figure 9. Congestion configuration.

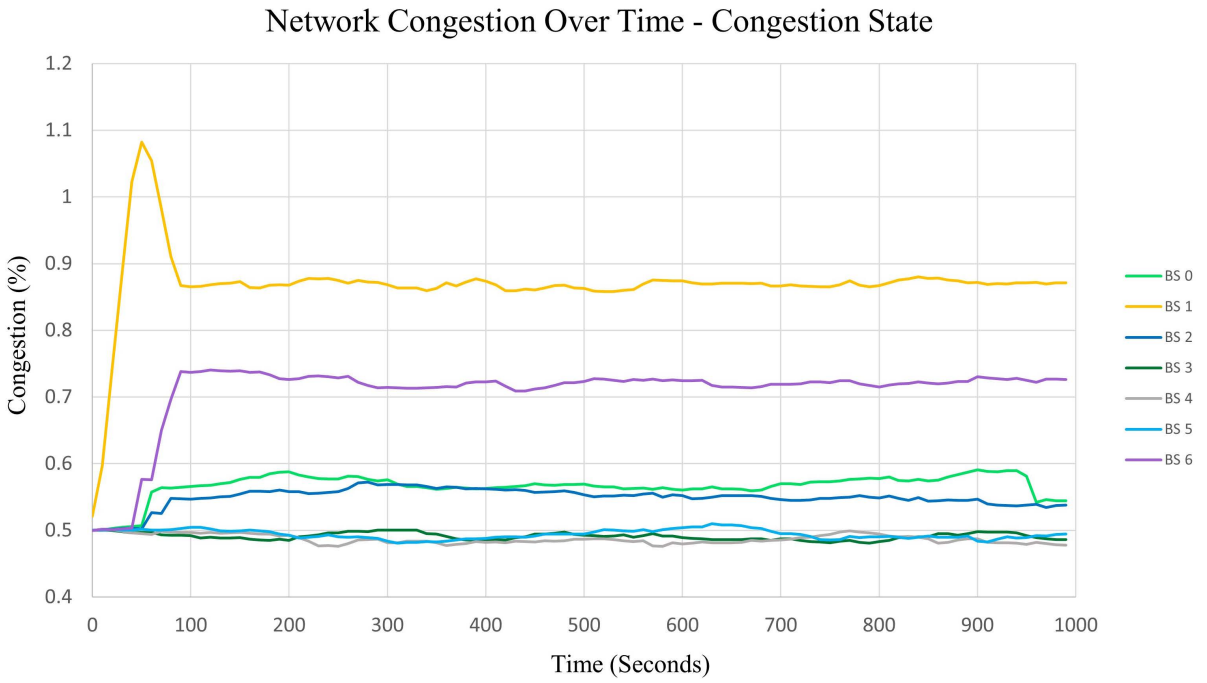

Figure 10. Congestion level after self-healing.

denoted by the yellow line, approaches 120% congestion, but is reduced to a healthy level using the self-healing techniques described in this paper. It can also be observed that small cells 0, 2, and 6 (shown by the light green, navy blue, and light purple lines respectively) have increased their congestion level since they inherited the users that need to be offloaded. Nonetheless, the network is now in an overall healthy state since all of the small cells are below the threshold that defines congestion.

The scatter plots in **Figure 11** & **Figure 12** display the UE distribution of the network before and after self-healing is performed. Each point on the scatter plot is equivalent to the location of a UE, and the color of the point corresponds to the small cell that the UE is associated with. In this case, each of the small cells 0, 1, 2, and 6 are uniquely colored clusters to aid in visualizing how self-healing takes place. The yellow cluster of points in the bottom left of the graph above

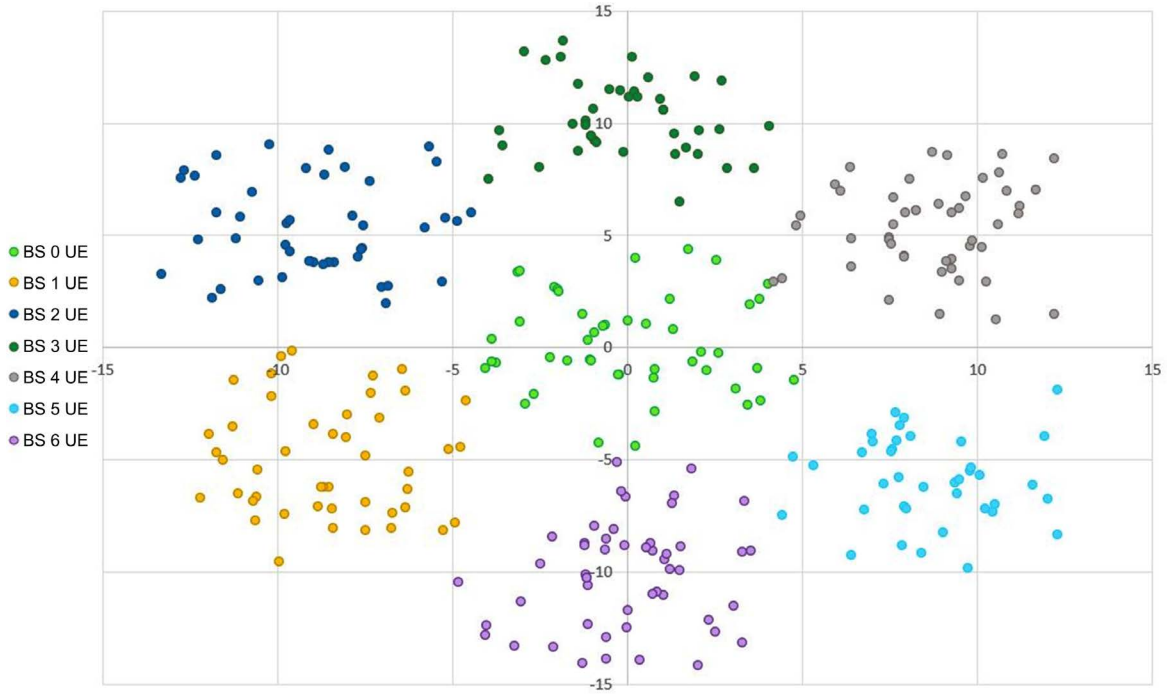

Figure 11. Scatter plot before self-healing.

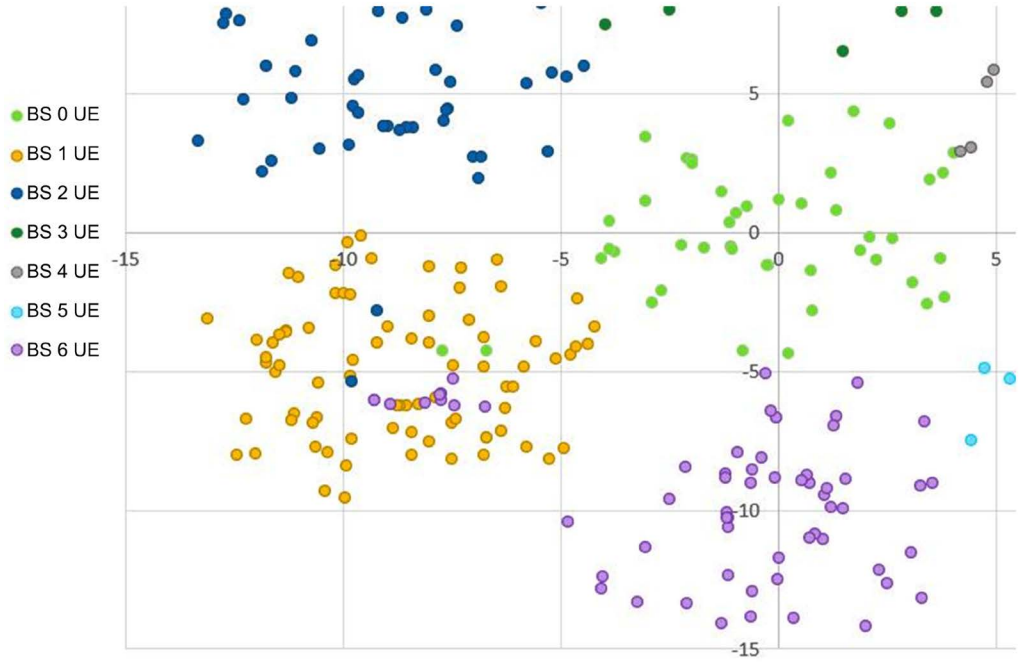

Figure 12. Congested small cell after self-healing.

represents small cell 1, while the light green, navy blue, and light purple clusters represents small cells 0, 2, and 6. When comparing Figure 11 and Figure 12 it can be observed that the users within small cell 1 are offloaded to its neighboring small cells, which is shown by how users within small cell 1 inherit the color of the small cells it is offloaded to.

7.2. Scenario 2—Failing

In the failing scenario shown in Figure 13 small cell 0 is in a failing state, while the remaining small cells are all healthy. Figure 14 and Figure 15 represent the UE distribution of the network before and after offloading the users from the failing small cell. It can be seen that all users have been offloaded successfully since the neighboring small cells have the capacity to support all the users of small cell 0, ensuring all user demand is met.

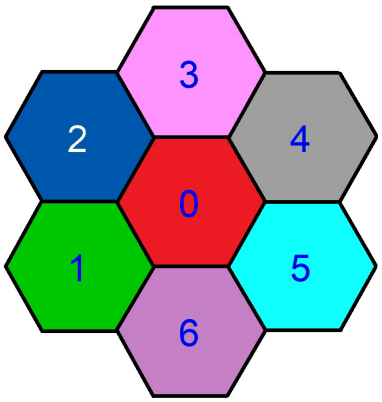

Figure 13. Failing configuration.

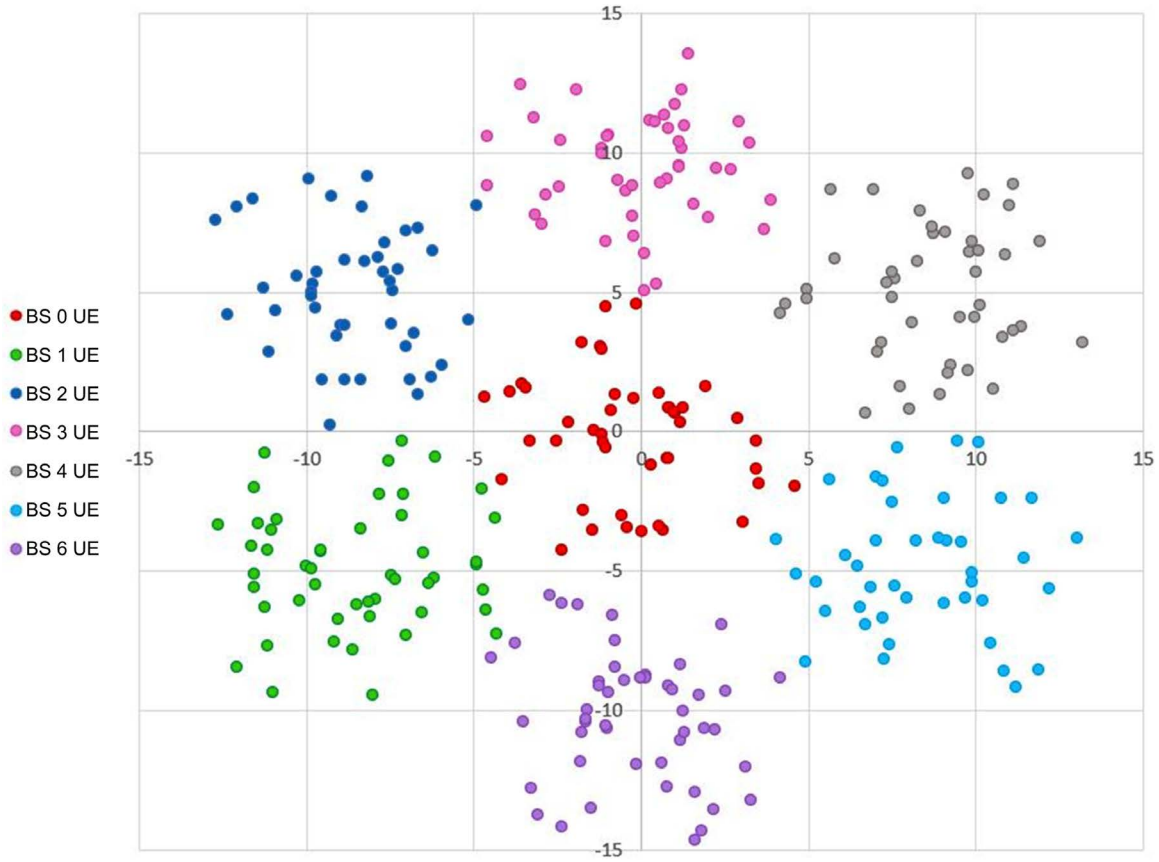

Figure 14. UE distribution prior to offloading process.

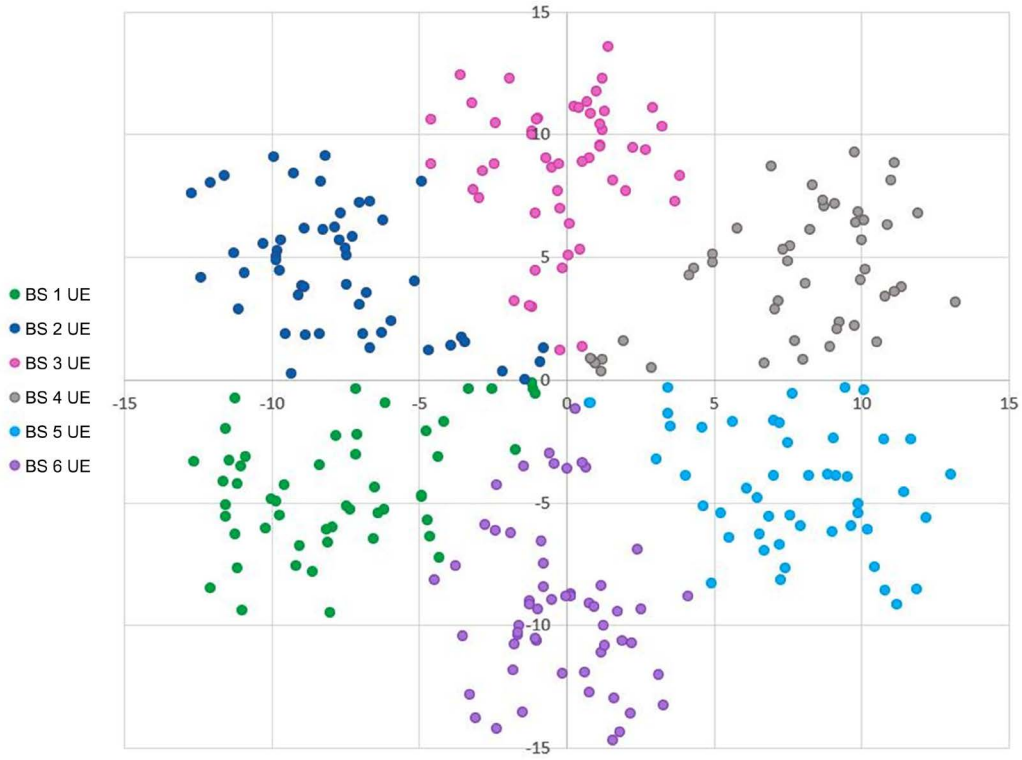

Figure 15. UE distribution following offloading process.

8. Conclusion

With the development of 5G networks that require a significantly larger amount of small cells, the need to implement effective Self-Healing procedures within a network is necessary to allow networks to autonomous recovery from failures that may occur. This paper proposed a heuristic self-healing algorithm that can be used within a network to allow neighboring small cells to provide aid to a small cell that is either in a congested or failing state, which allows the overall Quality of Service (QoS) of a network to be maintained while recovery procedures are performed. Additionally, the paper proposed a model for developing and testing novel self-healing algorithms through the use of a 5G Self-Healing network simulator, which applies a modular approach to allow end-users to customize network test cases and analyze network data from simulation log files that are produced post-simulation. Future work includes integrating network visualization within the simulator itself to allow users to graphically view the performance of self-healing during run-time.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of the paper.

References

[1] Jang, J., Chung, M., Hwang, S.C., Lim, Y., Yoon, H., Oh, T., Min, B., Lee, Y., Kim,

K.S., Chae, C. and Kim, D.K. (2016) Smart Small Cell with Hybrid Beamforming for 5g: Theoretical Feasibility and Prototype Results. *IEEE Wireless Communications*, **23**, 124-131. https://doi.org/10.1109/MWC.2016.1500387WC

[2] Alias, M., Saxena, N. and Roy, A. (2016) Efficient Cell Outage Detection in 5g Hetnets Using Hidden Markov Model. *IEEE Communications Letters*, **20**, 562-565. https://doi.org/10.1109/LCOMM.2016.2517070

[3] Asghar, A., Farooq, H. and Imran, A. (2018) Self-Healing in Emerging Cellular Networks: Review, Challenges, and Research Directions. *IEEE Communications Surveys Tutorials*, **20**, 1682-1709. https://doi.org/10.1109/COMST.2018.2825786

[4] Scheit, O. (2014) Self-Healing in Self-Organizing Networks. 175-181.

[5] Ali-Tolppa, J., Kocsis, S., Schultz, B., Bodrog, L. and Kajo, M. (2018) Self-Healing and Resilience in Future 5g Cognitive Autonomous Networks. 2018 *ITU Kaleidoscope*: *Machine Learning for a* 5*G Future* (*ITU K*), Santa Fe, 26-28 November 2018, 1-8. https://doi.org/10.23919/ITU-WT.2018.8598115

[6] Nisa, F. and Haryadi, S. (2016) Simulation of the Fault Management with Self Healing Mechanism (Case Study: LTE Network in Banda Aceh Area). 2016 10*th International Conference on Telecommunication Systems Services and Applications* (*TSSA*), Bali, 6-7 October 2016, 1-6. https://doi.org/10.1109/TSSA.2016.7871066

[7] Roh, W., Seol, J., Park, J., Lee, B., Lee, J., Kim, Y., Cho, J., Cheun, K. and Aryanfar, F. (2014) Millimeter-Wave Beamforming as an Enabling Technology for 5g Cellular Communications: Theoretical Feasibility and Prototype Results. *IEEE Communications Magazine*, **52**, 106-113. https://doi.org/10.1109/MCOM.2014.6736750

[8] Passoja, M. (2018) 5g nr: Massive Mimo and Beamforming—What Does It Mean and How Can I Measure It in the Field.

[9] Dewar, C. and Warren, D. (2014) Understanding 5g: Perspectives on Future Technological Advancements in Mobile.

[10] Klautau, A., Batista, P., González-Prelcic, N., Wang, Y. and Heath, R.W. (2018) 5g Mimo Data for Machine Learning: Application to Beam-Selection Using Deep Learning. 2018 *Information Theory and Applications Workshop* (*ITA*), San Diego, 11-16 February 2018, 1-9. https://doi.org/10.1109/ITA.2018.8503086

[11] Davaslioglu, K. and Gitlin, R.D. (2016) 5g Green Networking: Enabling Technologies, Potentials, and Challenges. 2016 *IEEE* 17*th Annual Wireless and Microwave Technology Conference* (*WAMICON*), Clearwater, 11-13 April 2016, 1-6. https://doi.org/10.1109/WAMICON.2016.7483860

[12] Fourati, H., Maaloul, R. and Chaari, L. (2019) Self-Organizing Cellular Network Approaches Applied to 5g Networks. 2019 *Global Information Infrastructure and Networking Symposium* (*GIIS*), Paris, 18-20 December 2019, 1-4. https://doi.org/10.1109/GIIS48668.2019.9044964

[13] Trehan, A. (2013) Algorithms for Self-Healing Networks.

[14] Xie, Y., Li, B., Zuo, X., Yan, Z. and Yang, M. (2018) Performance Analysis for 5g Beamforming Heterogeneous Networks. *Wireless Networks*, **26**, 463-477. https://doi.org/10.1007/s11276-018-1846-5

[15] Murudkar, C.V. and Gitlin, R.D. (2019) User-Centric Approaches for Next-Generation Self-Organizing Wireless Communication Networks Using Machine Learning. 2019 *IEEE International Conference on Microwaves*, *Antennas*, *Communications and Electronic Systems* (*COMCAS*), Tel-Aviv, 4-6 November 2019, 1-6. https://doi.org/10.1109/COMCAS44984.2019.8958302

[16] de la Bandera, I., Munoz, P., Serrano, I. and Barco, R. (2017) Improving Cell Outage

Management through Data Analysis. *IEEE Wireless Communications*, **24**, 113-119. https://doi.org/10.1109/MWC.2017.1600076WC

[17] Omar, T., Ketseoglou, T. and Naffaa, I. (2021) A Novel Self-Healing Model Using Precoding & Big-Data Based Approach for 5g Networks. *Pervasive and Mobile Computing*, **73**, Article ID: 101365. https://doi.org/10.1016/j.pmcj.2021.101365

[18] Ketseoglou, T. and Ayanoglu, E. (2019) Zero-Forcing Per-Group Precoding (ZF-PGP) for Robust Optimized Downlink Massive MIMO Performance. *IEEE Transactions on Communications*, **67**, 6816-6828. https://doi.org/10.1109/TCOMM.2019.2927205

[19] Molisch, A. (2011) Wireless Communications. Wiley, New York.

[20] Ketseoglou, T., Valenti, M.C. and Ayanoglu, E. (2019) Millimeter Wave Massive MIMO Downlink Per-Group Communications with Hybrid Linear Precoding. 2019 53*rd Asilomar Conference on Signals*, *Systems*, *and Computers*, Pacific Grove, 3-6 November 2019, 973-977. https://doi.org/10.1109/IEEECONF44664.2019.9048660

[21] Ketseoglou, T. and Ayanoglu, E. (2016) Linear Precoding Gain for Large MIMO Configurations with QAM and Reduced Complexity. *IEEE Transactions on Communications*, **64**, 4196-4208. https://doi.org/10.1109/GLOCOM.2016.7841956