POLITEHNICA UNIVERSITY OF BUCHAREST

Doctoral School of Electronics, Telecommunications and Information Technology

Decision No. ____ from DD-MM-YYYY

Ph.D. THESIS SUMMARY

Mustafa Khaleel Hamadani

TEHNOLOGII CLOUD IN RETEA 5G INTRE DISPOZITIVE DE COMUNICATII

CLOUD COMPUTING TECHNOLOGIES IN 5G NETWORKS FOR D2D COMMUNICATION

THESIS COMMITTEE

Prof. Dr. Ing. Eugen Borcoci Politehnica Univ. of Bucharest	Ph.D. Supervisor

BUCHAREST 2022

Contents

Chapter 1

1 Introduction

The accelerated development of 5G network and deployment of resource-intensive applications (e.g., 3D gaming) need sufficient resources for processing these applications with lower latency is required. On the other hand, the end devices have limited computational resources that insufficient for the resource-intensive applications. The answer to where we can find an available computational resources, accompanied by low latency communication, lies around device-to-device communication and cloud technologies.

The cloud technologies are centralized computing paradigm has almost infinite computational resources and on-demand resource provisioning. Cloud technologies (including Fog and Edge) can overcome the computing resource problem mentioned above.

The Device-to-Device (D2D) communication is considered as one of the key enablers for various 5G services, including public safety services, content sharing, coverage extension, vehicular communication [1]. D2D is regarded as a promising approach to transfer the data traffic in proximity with low latency communication.

In this thesis, we try to integrate the cloud technologies with D2D communication where D2D users' set is seen as a network of D2D clusters. These clusters are considered data sources that generate a resource-intensive services task. The cloud technologies provide a computational resource for these resource-intensive services.

The integration of the cloud technologies to D2D communication could be illustrated in Figure 1.2. the D2D users formed the clusters based on similar content (e.g., video streaming). The D2D users generated a service task that will be offloaded to the fog nodes via the base stations. The fog nodes received the service tasks and scheduled it to appropriate virtual machines based the service tasks category (e.g., online gaming).

Hence, three questions will raise (among other problems):

1. *How can we optimally determine the number of fog nodes to improve the specific criteria such as delay limits?*
2. *How to place the fog nodes at specific wireless infrastructures (Base Station) to minimize the distance between the Base Stations and fog nodes. Thus, the access delay between D2D clusters and Fog nodes is reduced.*

3. *How can we optimally distribute the workload between the computing nodes to achieve an almost equal amount of the workload.*

1.1 The Structure of The Ph.D. Thesis

The structure of this thesis is illustrated as follow:

chapter 1 title as (introduction) motivates thesis and highlights the main issues that this thesis handles and the research questions that arise.

Chapter 2 gives an overview of the Device-to-Device communication and cloud technologies covered in the thesis.

Chapter 3 study the Fog nodes Placement problem including the deremined the numeber of fog nodes and their at potential locations (Base Station) in the networks.

Chapter 4 investigated the distribution of workload amongst the fog nodes (i.e., virtual machines); the genetic algorithm adopted as the main solution for the load balancing issue.

Chapter 5 provided the simulation experiments for both chapters 3 and 4.

Chapter 6 discusses the results collected from previous chapters, highlights the limitations of the adopted solution, and suggests future works.

Chapter 2

2 Background and Related Concepts

This chapter given an overview of related concepts of technologies that included in this thesis, these technologies including D2D Communication, Cloud technologies, and SDN. This chapter introduced an architecture that integrated the D2D communication and Fog computing under SDN supervision. The after mentioned architecture introduce number of issues that will be discussed in the next chapters

2.1 Overview of D2D communications

D2D communication is characterised as the direct communication between UEs in the proximity of each other with complete/partial involvement of the cellular infrastructures (Base Station). Moreover, D2D communication exists in the licensed cellular spectrum, making the D2D provide QoS guarantees and seamless network detection [2].

The 3rd Generation Partnership Project (3GPP) defined two procedures for supporting D2D in cellular network (LTE), namely, D2D Discovery and D2D Communication. in the D2D discovery, the UEs detected and monitored other UEs in proximity. The *device discovery* is further classified into *Centralised Discovery* and *Distributed Discovery* [2].

For the centralised Discovery, a centralised entity such as Base station will assist the mobile devices in discovering phase. Moreover, the *centralised discovery,* also known as *network-assisted discovery,* can be further classified based on the degree of involvement of the Base Station into:

- *The Full BS Involvement:*

In this mode, the Base Station controlled all the devices discovery procedures starting from restrained devices to initial discovery of the others in proximity [2].

- *The Partial BS Involvement:*

In this approach, the devices initial a device discovery without getting any involvement from the base station [2]. The role of the Base station on mode starts in gathering information about the Channel quality [2].

In other hand, the *Distributed Discovery* as known as Direct Discovery, the devices completely discover others in proximity without the involvement of the Base Station [2].

The main advantages of D2D communication could be summarized as follow:

- The D2D communication provides a high date and low latency (i.e., proximity gain) due to the direct link between the UEs.
- The spectrum efficiency improved as increasing the number of bits transmitted in the unit of bandwidth. Furthermore, the energy efficiency is enhanced by compressing more data with low energy per bit.

2.2 Collaborative Technologies: Cloud, Edge, and Fog

The Cloud Computing (CC) considered the robust infrastructure that provides remote data computation and storage [3]. However, the long-distance between the remote cloud servers and end-users results in a high transmission latency that does not satisfy the strict requirements of modern applications. Cloud Computing is a centralized solution involving powerful central nodes that provide Cloud Services to end-users. These services could be classified into Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) [4].

Another two flavor approaches are proposed to overcome the latency issue: Fog and Edge computing. Fog and Edge paradigms shared similar concept, and they could be used them interchangeably [5]. However, there are slight differences between the two; according to the Open Fog Consortium, Fog computing is a hierarchical paradigm and provides

computing, networking, storage, where the Edge Computing is limited to only computing services[6]. Another difference related to node placement, the placement of edge nodes cloud be at the boundaries of service providers (e.g., cellular networks and intelligent transportation), where the fog nodes could be located anywhere nearby of end nodes[7].

2.3 The Software-Defined Networking (SDN)

The typical SDN architecture is structured into three different layers/planes [8]: Data plane, control plane, and application plane, as illustrated in Figure 2.10. the data plane, also known as the forwarding layer, is composed of typical networking devices without any decision-making capability, and the control plane consists of a centralised node that responsible for orchestrating the underlying infrastructure layer.

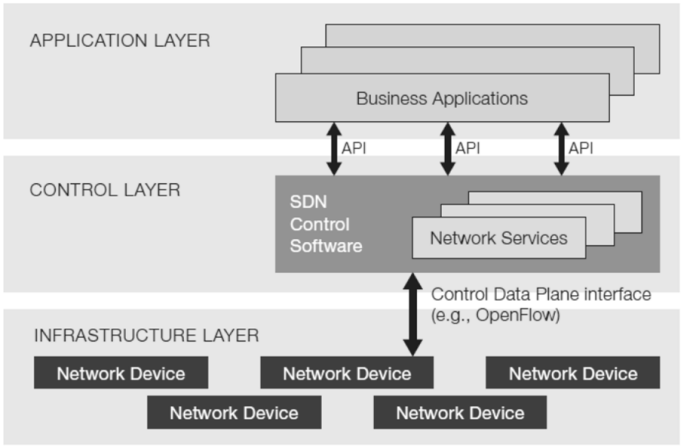


Figure 2-1 The SDN Architecture [8]

The application plane is deployed on the top of the previous planes; the application plane realizes the behavior of the network and specifies the customized policies for the effective management of the networks. The core idea of SDN is the separation between the control plane and the data plane. In SDN, the network devices such as switches and routers forwarded packets according to policies (rules) installed in each device [8].

An essential requirement of our proposed system is managing a high number of mobile users (i.e., D2D Clusters) and their services to satisfy the QoS requirements. Hence, the SDN controller in our work responsible for managing and controlling the fog nodes as follow:

- The SDN controller collected network information, including the number of waiting jobs, the average service time of fog servers, load of network links.
- The SDN controller constructed the optimal service task offloading decisions and created the flow rules at the SDN-enabled Base Stations.
- Updating the services hosting, migration, and replication of services instances; and reconfiguring Base Station in terms of flow rules [9].

As mentioned earlier, the application plane defined the network's behavior and customized policies for specific use cases. In our work, the load balancing policies implemented in the application plane to leverage autonomous management of fog nodes resources in the network.

2.4 Architectural Aspect

In this thesis, we introduce a system architecture that builds based on the work [10], as shown in Figure 2.12. The architecture consists of three layers; the infrastructure layer represented the various wireless infrastructures (e.g., Base Station) and D2D clusters. The second layer is the fog nodes network (FNs) deployed at the base stations and connected to the SDN controllers (third layer) also connected to the cloud for further processing.

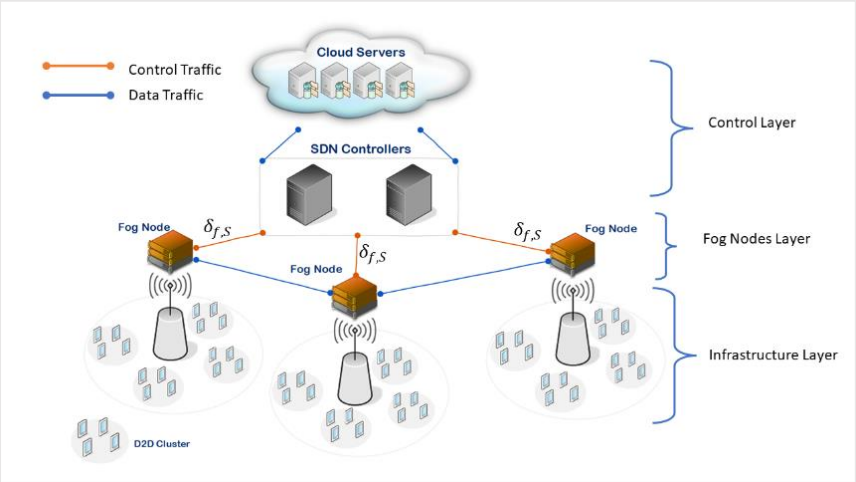


Figure 2-2 The System Architectural Aspect

The third layer comprises the SDN controllers and clouds remote servers and this layer considered as the control layer. The D2D clusters are considered the data source in the system; they generate a service task (e.g., online gaming), and the cluster head is connected to fog nodes through the Base stations. These devices are assumed as resource-limited devices that generate a resource-intensive task, and these service tasks offload to fog nodes for pre-processing. The Fog nodes represented the physical servers running various virtual machines (VMs) that represented different user-case applications. Moreover, these VMs are heterogeneous in terms of computation power and storage capability.

The SDN controller is responsible for managing and controlling the Fog nodes by setting the flow rules that defined the load balancing policies. The SDN controller monitors the fog nodes status by gathering information about the Fog nodes, including the available computation resource, amount of service tasks waiting, and others. Based on this information, the controller built a global network view and defined the load balancing

algorithm. In other words, the load balancing algorithm will be run centrically as an application model in the SDN architecture.

Finally, the remote cloud server (CRS) represented a source of almost unlimited computation resources and storage. CRS processed the service tasks offloaded from the fog nodes and provided additional processing for the SDN controllers.

Chapter 3

3 Fog Computing and D2D Networks Cooperation

This chapter compose of two parts to study the fog nodes deployment issue (i.e., count of fog nodes, and their deployment location in the network). in the first part of this chapter focused on computing the number of fog nodes in the network with aim to minimize the response time of fog nodes network. Further, the second part determined the candidate locations (Base Station) of the fog nodes in the network with objective to minimum the distance between the base stations and fog nodes. After determining the number of fog nodes, and their location in the network. the characteristic of fog nodes (number of virtual machines and their attributes), also and the load balancing issue will be discussed in chapter 4.

3.1 Mathematical Model for Fog-D2D cooperation

In this section, we shortly present a mathematical model that describes the architecture introduced in that section. This mathematical model has been build based on works proposed in [11].

The mathematical model has been build based on Queue theory. As depicted in Figure 3.2, the system model consists of three layers: the D2D cluster set, fog nodes layer set, controllers, and the cloud remote servers.

We consider a cellular network with several small cells, and each cell is covered by one Base Station (BS) located at the center of the small cell. We consider a set $D = \{d_1, d_2, \dots, d_i\}$ of the group of D2D clusters; they generate service requests $\{\lambda_1, \lambda_2, \dots, \lambda_i\}$ that will be sent to Fog nodes set $F = \{f_1, f_2, \dots, f_m\}$ through base stations set $\beta = \{\beta_1, \beta_2, \dots, \beta_n\}$.

Furthermore, in case the fog nodes are not able to handle the service request, it is forwarded to the cloud remote servers. Moreover, each base station managed and controlled several D2D clusters denoted as α.

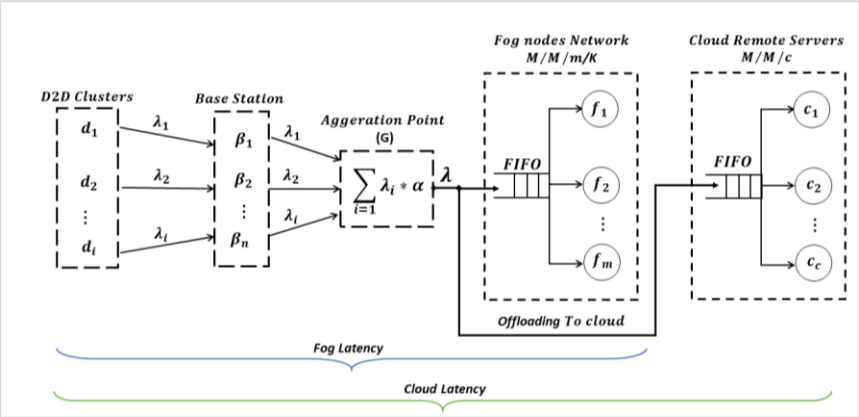


Figure 3-1 The System Model (Fog Placement)

The remote cloud servers denoted as $C = \{c_1, c_2, \dots, c_c\}$, the remote cloud servers have unlimited computation power and storage capacity.

3.2 D2D Clusters Service Request Model

The D2D cluster d_i generate service-request λ_i. Moreover, these requests are assumed to be Poisson Distribution (i.e., the requests independent of each other) [12]. In general, the total service-requests (λ) aggregated at $\boldsymbol{G}$ could defined as:

$$Total\ Service\ Request\ (\lambda) = \sum_{i=1}^{\beta_n} \lambda_i . \alpha \tag{3.1}$$

α : The number of D2D clusters managed by each Base Station.

λ_i : The service request from a single D2D cluster.

3.3 Fog Nodes Network and Cloud Servers models

In this section, the Fog nodes and Cloud server are modeled as a multi-server queueing model where the Fog nodes are described as $M/M/m/K$, where K represented the maximum capacity of fog nodes. The cloud servers are described as $M/M/c$ with unlimited capacity of receiving services tasks from D2D users. The main goal of this section is to determine the response time (i.e., computation time) for both queueing models. Hence, the total response of the system is computed.

3.3.1 Fog nodes Model (*M/M/m/K* queueing model)

The Fog nodes network modeled as a multi-server queueing model $M/M/m/K$ queuing model. Where m represented the number of identical parallel physical servers that provide a fog service to D2D clusters where $F = \{1,2, \dots m\}$ where K represented the maximum capacity of fog nodes to receive service tasks. Assume that λ and μ_F are the arrival service requests and service rate, respectively. Therefore, if a service task arrives to find all m servers busy, then the task is dropped. Hence, the service rate at that time is μ_F.

In order to compute the response time (i.e., computation time) of fog nodes, some entities need to determine, namely P_b (the probability of blocking), and number of customers (service tasks) in the system.

The probability of blocking (P_b) also refers as *Erlang B formula* or *Erlang's loss formula* and is often written as $B(\lambda/\mu_F, m)$[13].

$$P_b = \frac{(m\rho)^m/m!}{\sum_{k=0}^{m} \frac{(m\rho)^m}{k!}} \tag{3.2}$$

. The number of services tasks is denoted as (N_q) and equal to:

$$N_q = \sum_{k=0}^{m} kP_k$$

$$N_q = (m\rho) \sum_{k=0}^{m} p_0 \frac{(m\rho)^k}{k!} = m\rho(1 - P_b) \tag{3.3}$$

Finally,
The response time (computation tine) is denoted as (T_F) can be computed as:

$$T_F = \frac{m\rho(1 - P_b)}{\lambda} + \frac{1}{\mu} \tag{3.4}$$

As seen from (3.10), as the number of fog nodes (m) increases, the response time (T_F) in network decrease; in other words, deploying more fog nodes in the network will be shorten the processing time for the services that provided to the D2D users. to satisfied the QoS requirements for D2D users, we introduced a QoS time threshold (T_{QoS}) from work [14]. Moreover, we restricted the response time (T_F) to be equal to or lower than (T_{QoS}).

So, our objective is to determine the optimal number of fog nodes in the network to satisfy the QoS requirement, and it could define as:

$$minimize\ (T_F) \tag{3.5}$$

Subject to:

$$T_F \leq T_{QoS} \quad (a) \qquad m \in \mathbb{Z}^+ \quad (b) \tag{3.6}$$

3.12 (a) defined as a constraint that the response time should be equal to or lower than QoS time (T_{QoS}) introduced from work [15]. 3.12 (b) defined as a constraint that the number of fog nodes should be a positive integer number ($\mathbb{Z}^+$ represented the positive integer number). to solve equation (3.11), we proposed an algorithm (see Algorithm 3.1) to obtain the required number of fog nodes.

Algorithm 3.1 The calculation of the number of Fog Nodes

Input: service-request λ_i, number of BS β, number of D2D cluster α, service-rate μ_{FC}, initial fog node number Fn, max fog nodes $FnMax$, and T_{QoS}

Output: $BestTps$, optimal number of fog nodes $FnBest$, ρ.

$Fn = inital\ value$

$T_R^F = inf$

1. **While $Fn < FnMax$ do**
2. **compute total arrival rate lambda λ equation (1)**
3. **compute the utilization factor ρ**

if $\rho < 1$ than continue

else

$Fn = Fn + 1$, go to step 3

4. **compute erlang C formula**
5. **compute Tps, equation (9)**

if $Tps < T_{QoS}$ than

$BestTps = Tps$

$FnBest = Fn$

else

$\boldsymbol{Fn = Fn + 1}$

go to step 1

6. **end while**
7. **Return $\boldsymbol{BestTps}$ and $\boldsymbol{FnBest}$.**

3.4 The K-Medoid algorithm for Fog Nodes Placement

After the number of fog nodes has been determined, the fog nodes need to place/deploy at specific locations so as the distance between the base station and fog nodes will be shorted. Hence, the access delay between the D2D cluster and Fog nodes will be minimized.

The system architecture (see Figure 3-3) consists of a set of $D = \{d_1, d_2, \dots, d_i\}$ of the group of D2D clusters; they generate service requests $\lambda = \{\lambda_1, \lambda_2, \dots, \lambda_i\}$ that will be offloaded to Fog nodes set $F = \{f_1, f_2, \dots, f_m\}$ through base stations set $\beta = \{\beta_1, \beta_2, \dots, \beta_n\}$ that deployed at a specific location in the network. The fog nodes set F installed at the base station set β to provide fog service to the edge users (D2D clusters).
The system architecture could be defined as an undirected graph $G = (V, E)$, where $V = \beta \cup K$ and β is the set of the base stations and $K = \{K_1, K_2, \dots, K_m\}$ is the set of the potential location of fog nodes set F as shown in Figure 3.2.

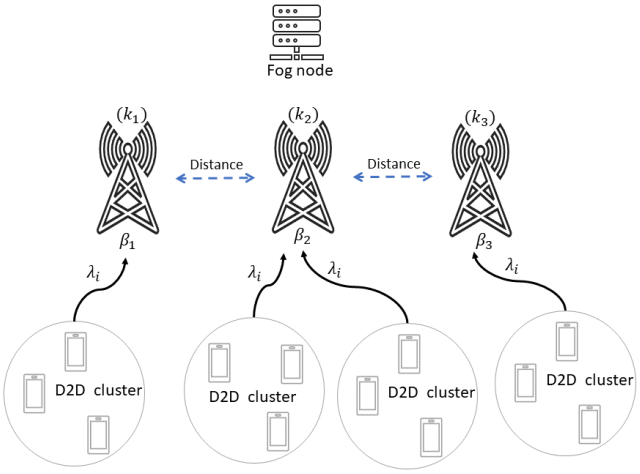


Figure 3-2 The Fog Nodes Placement model

Where E represented the set of connections between a base station and fog node, the connections between the base stations and links between the mobile devices and base stations are out of our scope.

The squared Euclidean distance will be considered as a measurement between the base station and fog nodes network. Thus, our objective is to obtain a minimum access delay between the Base Station and Fog nodes:

$$Min\ D(C_\beta, C_f) \tag{3.7}$$

Subject to following constraints:

1. $x_{\beta f}$: binary decision indicates whether a base station β_n is assigned to the fog node f_m.

$$x_{\beta f} = \begin{cases} 1, & \beta_n \text{ is assign to } f_m \\ 0, & otherwise \end{cases}$$

2. Two Fog nodes are not allowed to be installed at the same Base Station.
3. Only One Base Station connects to One Fog node.

The K- medoids clustering technique will be adapted for fog node placement. In K-medoids, the network of base stations is divided into K area (or region), where K represents the number of fog nodes that calculated earlier (see section 3.3.2 and algorithm 3.1).

We consider a Euclidean distance as a measurement to define the cluster boundary and defined as follow:

$$D_{Euclidean}(C_\beta, C_f) = \sqrt{(C_{\beta 1} - C_{f1})^2 + (C_{\beta 2} - C_{f2})^2} \tag{3.8}$$

Where C_β and C_f are location coordinates of the base station and fog nodes, respectively.

In K-medoid, a medoid can be defined as the point in the cluster, with a minimum sum of distances to all other points (see Figure 3.3)[16]. The minimum distance between medoid (in our case, will be Fog nodes) and clusters point (Base Station) is calculated as follow:

$$D_{medoid}(C_\beta, C_f) = \sum_{C_{\beta_n}} \sum_{C_{fm}} |C_\beta - C_f| \tag{3.9}$$

The K-medoid clustering algorithm could be described as following steps:

1. *Step 1 (Initialization)*
- Choose K points as initial medoids.
- Calculate distance measurement (in our case Euclidean Distance Squared).

- Defined initial clusters by assigning each to the nearest medoids.

2. *Step 2 (Medoids selection)*
   Select initial medoids of each cluster, which is the point that has minimum distance to all points in the same cluster.
3. *Step 3 (Medoids Update)*

Swap the previous medoids (from step 3) with all other points from the current cluster. Calculate the sum of the distance between points; the new Medoids will be the points that have a minimum distance to all other points in the cluster the current cluster. Figure 3.4 gives a flow chart of the K-Medoids step mentioned above.

Chapter 4

The Load Balancing Techniques for Fog Computing Networks

This chapter study the load balancing issue that arises due to unequal workload distribution among the fog nodes. the study of characteristic of the of the fog nodes (virtual machines and their attributes) will be provided. the load balancing issue can be defined as optimization problem with objective is to minimize the execution of the service tasks received from the D2D clusters. the main objective of this chapter is distributing the workload almost equally among the fog nodes. Fog computing deploys as fog nodes (i.e., Fog servers) at specific regions in the network; each fog node provides computing resources and storage capabilities to edge users.

The operation of distribution of users' service requests among all the fog nodes so that no fog node will be overloaded or underloaded (i.e., idle) is called Load Balancing (LB). Load Balancing is one of the criteria issues in cloud computing technologies; the goal of load balancing ensures that all processing nodes (e.g., Fog nodes) approximately received an equal amount of workload [17].

The system model could be considered as undirected graph (see Figure 4-2), where $\boldsymbol{V} = \{F, \mathrm{S}, \mathrm{C}\}$ is node-set. Where F represented the fog nodes set $F = \{f_1, f_2, \dots, f_m\}$, $S = \{s_1, s_2, \dots, s_j\}$ presented the SDN controllers set, and C is remote cloud servers set and $C = \{c_1, c_2, \dots, c_c\}$. $\boldsymbol{E} = \{e_{a,b}\}$ presented the edge set and $\boldsymbol{e_{a,b}}$ denoted as the link between two a nodes and b nodes.

Moreover, $\delta_{a,b}$ denoted as the communication latency between a and b nodes, respectively. Moreover, the fog node (e.g., f_1) contain a number of virtual machines, hence, the total number of VMs among all the fog nodes denoted as $N = \{VM_1, VM_2, \dots, VM_w\}$, each VM_v ($VM_v \in N$) has limited capability of computing resources such as CPU, memory, and storage capacity, and each VM_v is described with following attributes [18]:

$$VM_v = \{VM_v{}^{Id}, VM_v{}^{MIPS}\}$$

Where $VM_v{}^{Id}$ denoted the identifier of VM_v and $VM_v{}^{MIPS}$ presented the processing capability of VM_v. Further, the processing capability of VM_v denoted as P_v.

The D2D clusters $\{D_1, D_2, \dots, D_i\}$ generate a computational service task denoted as $\{\lambda_1, \lambda_2, \dots, \lambda_i\}$, each services task λ_k ($\lambda_k \in$ Service Tasks) can be described as following [18]:

$$\lambda_k = \{\lambda_k{}^{Id}, \lambda_k{}^{Length}\}$$

Where $\lambda_k{}^{Id}$ denoted the service task identifier, $\lambda_k{}^{Length}$ presented the length of the task λ_k stated as instructions per second. Moreover, the expected time to complete a tasks λ_k on VM_v can be calculated as follow [19]:

$$ETC_{k,v} = \frac{\lambda_k{}^{Length}}{P_v} \tag{0.1}$$

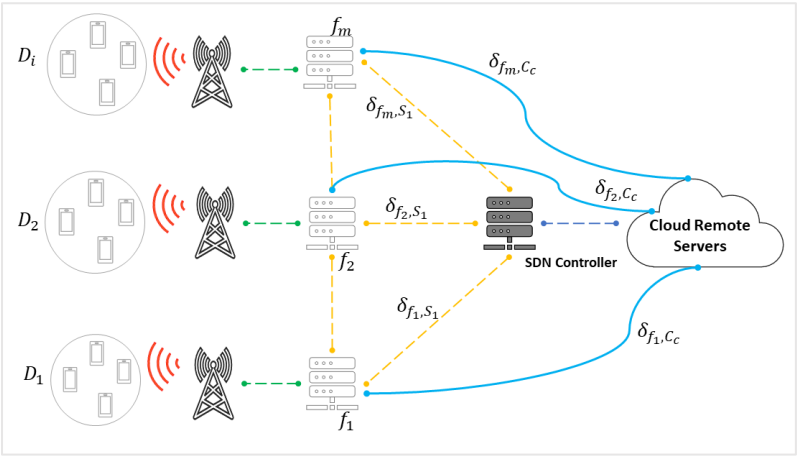


Figure 0-1 System Model (with three fog nodes and one SDN controller)

We introduced an ETC (Expected Time to Compute) matrix for describing the task model for cloud technologies in the heterogeneous environment [20]. The ETC matric illustrated the expected time to completed/compute a specific service task on specific VM as shown in (4.2).

$$ETC = \begin{bmatrix} ETC_{1,1} & ETC_{1,2} & \cdots & ETC_{1,w} \\ ETC_{2,1} & ETC_{2,2} & \cdots & ETC_{2,w} \\ ETC_{3,1} & ETC_{3,2} & \cdots & ETC_{3,w} \\ \vdots & \vdots & \cdots & \vdots \\ ETC_{k,1} & ETC_{k,2} & \cdots & ETC_{i,w} \end{bmatrix} \quad (0.2)$$

In ETC matrix, the row of the matrix indicates the service task (λ_k) and the column present the VM_v. For example $ETC_{1,1}$ expressed the required time to complete a task λ_1 on VM_1, $ETC_{3,2}$ expressed the required time to complete a task λ_3 on VM_2 and so on.

Hence, the purpose of this work is to find an optimal mapping of service task to VM such that the load is approximately placed on all the nodes, and one or more objective satisfied. Thus, load balancing is achieved.

3.5 Load Balancing Problem Formulation

The load balancing problem can be described as assigning k service task to v virtual machine, and the problem could be formulated as an optimization problem that satisfied one or more objectives. The most common objective studied in the literature is the makespan (MS), which is defined as "*the maximum execution time among all the VMs*" [21]. To calculate the makespan, firstly; the execution time (ET) of all VM has be compute:

$$ET_{kv} = \sum_{k=1}^{i} \sum_{v=1}^{w} x_{k,v} \ . ETC_{k,v} \quad (0.3)$$

Where ($k \in$ Service Tasks, and $1 < k < i$) and ($v \in w$, and and $1 < v < w$).

$x_{k,v}$ is a decision variable indicted if a service task is allocated to a specific VM or not. $x_{k,v} = 1$, if service task λ_k is allocated to VM_v and otherwise $x_{k,v} = 0$.

Then, the MS can be calculated as:

$$MS = maximum\ (ET_{kv}) \quad (0.4)$$

Finally, the load balancing problem for fog computing can be formulated as:

$$minimize\ (MS) \tag{0.5}$$

The purpose of the load balancing problem is to find an optimal mapping that minimizes the maximum load. The complexity of mapping of i task to $w\ VM$ equal to (w^i), hence; load balancing load in fog computing is considered a bin-packing problem known as NP-hard and NP-complete groups [22].

3.6 The Genetic Algorithm for Load Balancing

The Approximation Algorithms (e.g., Genetic Algorithms) are considered an alternative approach for exhaustive search techniques in exponentially large space solutions. The Genetic Algorithms (GAs) are metaheuristics algorithms inspired by Charles Darwin's theory of natural evolution.

In general, The Genetic Algorithm could be summarised as the following:

- *The Population:*

a set of individuals represent a population, every individual in the population is considered a chromosome.

- *The chromosome selection:*

select the individuals (i.e., parents) from the current population for an intermediate solution.

- *The Crossover:*

The crossover operation can be summarised by selecting two individuals (considered as the parents) to produce a new individual called the children that presented for the next generation.

- *The Mutation:*

The mutation operation modifies the genes of chromosomes to form a new generation. Thus, the algorithm able to produce a better solution than previously computed.

- *The Fitness Function:*

The fitness function measured the suitably of the individuals in the population. Therefore, the individuals survive or terminated according to the fitness or function value.

Chapter 5

4 Simulation Results

This chapter provide the implementation and simulation results for issues mentioned in previous chapters. hence, this chapter divided into two sub-chapters.

the First part discusses the simulation results for chapter 3 (determine the number of Fog Nodes and the deployment locations), the MATLAB considered as a simulation tool in this part.

The Second Part discussed the simulation results for chapter 4 (the load balancing for Fog Nodes network), the CloudSim considered as a simulation tool in this part.

4.1 The Fog Nodes Placement Simulation Scenarios

In this section, two simulation scenarios are performed to study and analyze the integration of fog computing to D2D networks. The simulation experiments have been implemented with MATLAB.

4.1.1 The Small size and large size service tasks Scenario

In this scenario, we consider two different types of service tasks (small and large size of the tasks) that are generated by D2D clusters. The small service tasks ($\boldsymbol{\lambda_i}$) have a size of about 2500 instructions per second and the size of large service tasks about 5000 instructions per second. We considered that a total number of D2D cluster in the network about 1000 cluster and each Base Station manage and control a group of clusters variated from 10 to 40 clusters (i.e., 10, 20, 30, and 40). Hence, the number of D2D clusters varied from 250, 500, 750, 1000 clusters, as shown in Table 5.1.

Table 4.1 The Total Service Requests

The Number of D2D Cluster	$\boldsymbol{\lambda_i}$ ***(In Second)***	***The Total Service Request (λ)***
250	0.25 req/s	62 req/s
	0.5 req/s	125 req/s
500	0.25 req/s	125 req/s
	0.5 req/s	250 req/s
750	0.25 req/s	187 req/s
	0.5 req/s	375 req/s

1000	0.25 req/s	250 req/s
	0.5 req/s	500 req/s

The service rate of the fog node, about 50 requests per millisecond, also has a limited queue size (see chapter 3 section 3.3.2). We also set the maximum number of fog nodes by about 16 nodes, Table 5.2 lists of parameters used in this simulation scenario.

In this study, several performance metrics are considered as follow:

1. *The Response Time*.
   This metric measures the time required to process/compute the individual request. It can be determined as the sum of service time and queueing time (see equation 10).
2. *The utilization factor* (ρ) : This metric represented the ratio number of task/massages requests to service rate. Moreover, the utilization factor (ρ) consider as a measurement of system stability, and its value should be less than 1.
3. *The System Workload*: we measure the fog node network workload under the different number of D2D clusters during a particular duration of time. This metric computes by taken the utilization factor multiply 100.

The effect of the number of fog nodes on the system stability is illustrated in Figures 5.2 and 5.3. In both Figures, the system reached stability as the number of fog nodes increase. We mean by the system stability that ρ must be less than 1.

Table 4.2 Simulation Parameters (First Scenario)

Simulation Parameters	**Description**
D2D Cluster (α)	10, 20, 30, and 40 cluster
Base Station (BS)	25 BSs
Service requests λ_i	0.25 and 0.5 req/s
Service rate μ	50 req/s
Maximum Fog node $FnMax$	16 nodes
T_{QoS}	0,4 s
Maximum Fog node capacity (k)	254

However, the total service request λ affects determined the number of fog nodes to reach stability.

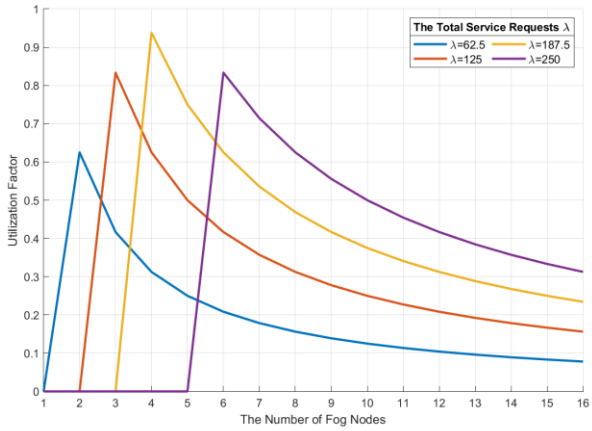


Figure 4-1 The System Stability (Small Service Tasks)

For example, when the total service request equals 62 (small size service tasks); the system needs about 2 fog nodes for reaching stability (see Figure 5.2), and on the other hand, when the total service request equals 125, we need about 3 to reach the system stability as shown in Figure 5.2

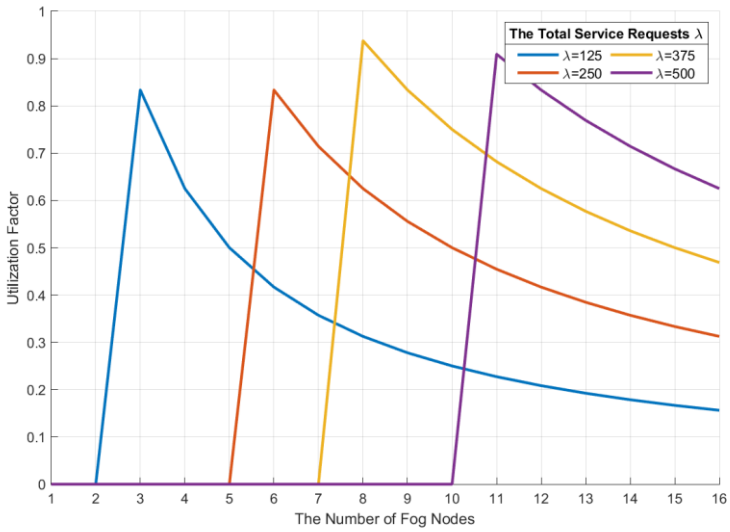


Figure 4-2 The System Stability (Large service tasks)

Similarly, when the total service request (large size service tasks) equals 250 (figure 5.3), the required number of fog nodes is about 6. Moreover, the required number of fog nodes to reach a stable system is equal to 11 in case the total service request equals 500.

As illustrated in Figure 5.4, the number of fog nodes increased, the computation time decrease; for example, when $\lambda = 250$ request we need about 9 fog nodes to meet the

threshold (T_{QoS}) and the reason behind that specified number that the system between 1 to 8 is unstable ($\rho = 0$) as illustrated in figure 26.

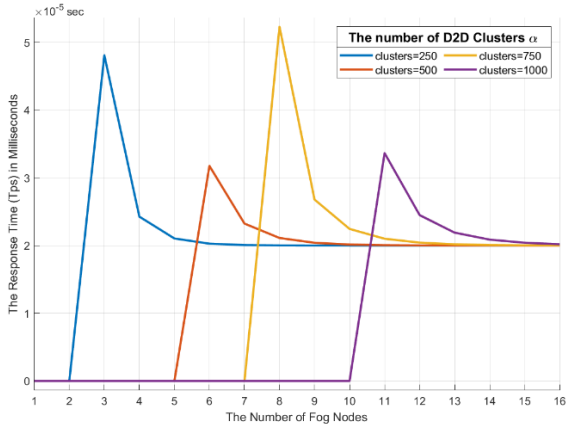


Figure 4-3 The Response Time of Fog nodes networks (Large size Service Tasks)

Figure 5.6 and 5.7 shown the network workload under different D2D clusters density with different service tasks size.

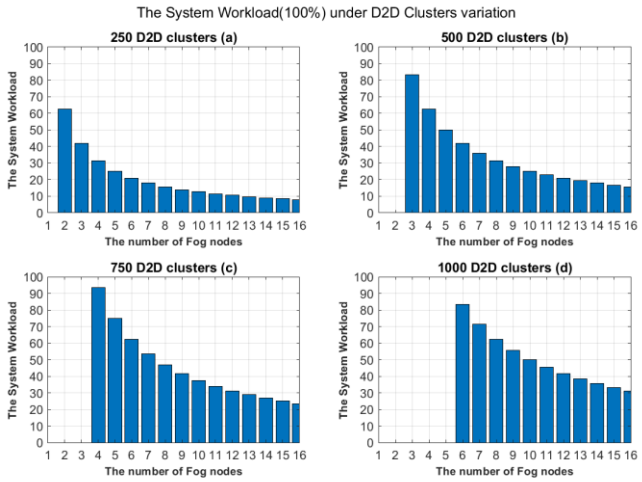


Figure 4-4 The System Workload with different D2D clusters Density (small size service tasks)

As we notice in Figure 5.2 (a) the workload decrease as the number of fog nodes decrease; reaching 45% and with 3 nodes and continue to reach less than 40% with 4 nodes.

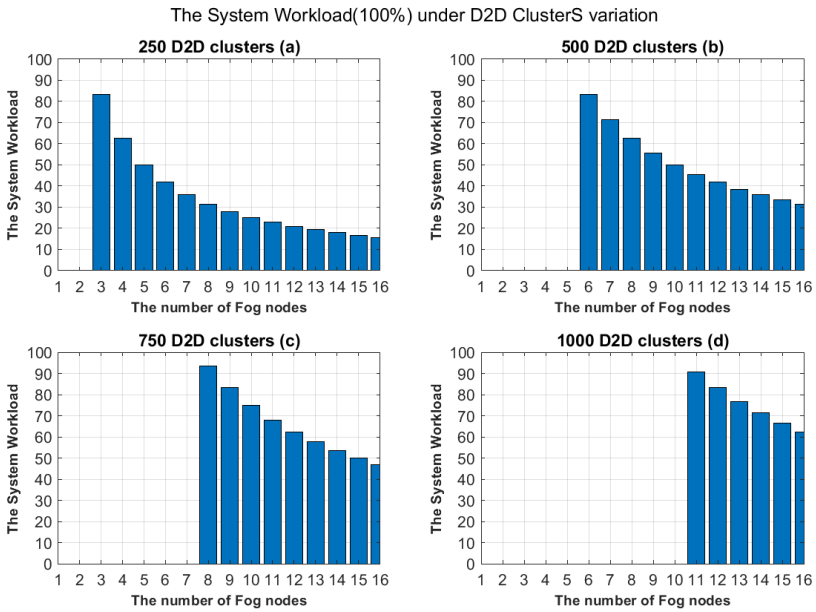


Figure 4-5 The System Workload with different D2D clusters Density (large size service tasks)

Thus, fog nodes 3 to 4 are ready to serve an additional cluster. Moreover, as the number of D2D clusters increased to 500 clusters (b), the fog nodes' workload reached 60% with 4 nodes and decreased to 40% with 7 nodes and continued decreasing below 40% as fog nodes decreased.

4.1.2 The K-Medoids Simulation Scenario

In this section, a simulation experiment is implemented for fog nodes deployment/placement at potential locations (Base Stations). A real-world network dataset (EUA data set) is considered in this work. The dataset used in this simulation provided by Swinburne University of Technology [23] contains the geographical coordinates of 125 base stations and 816 mobile users and those base stations deployed in the Melbourne central business region.

In this scenario, we only focus on the geographical location of the base station, and the distribution of mobile users is out of the scope of this work. For fog nodes placement, we adopt the random deployment approach of fog nodes as a comparative approach against the K-medoid clustering technique. the number of fog nodes considered in this scenario has been determined from the previous section under large service tasks, as illustrated in table 5.5.

Table 4.3 The number of Fog Nodes (Large Size Service Tasks)

The Number of D2D Cluster	λ_i	***The Total Service Request (λ)***	***Number of Fog nodes***
250	$0.5\ req/s$	$125\ req/s$	4 nodes
500	$0.5\ req/s$	$250\ req/s$	6 nodes
750	$0.5\ req/s$	$375\ req/s$	9 nodes
1000	$0.5\ req/s$	$500\ req/s$	12 nodes

Figure 5.8 shown the base station deployment by their longitude and latitude.

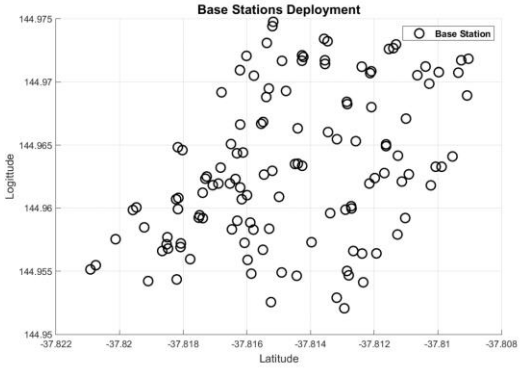
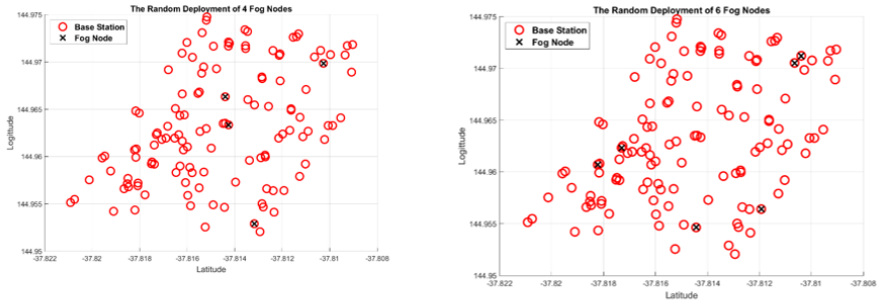


Figure 4-6 The Base Station Deployment

Thus, the fog nodes should be deployed such that a set of base stations are covered by one fog node, and the fog node installs on the base station that is located almost centrally in that set.

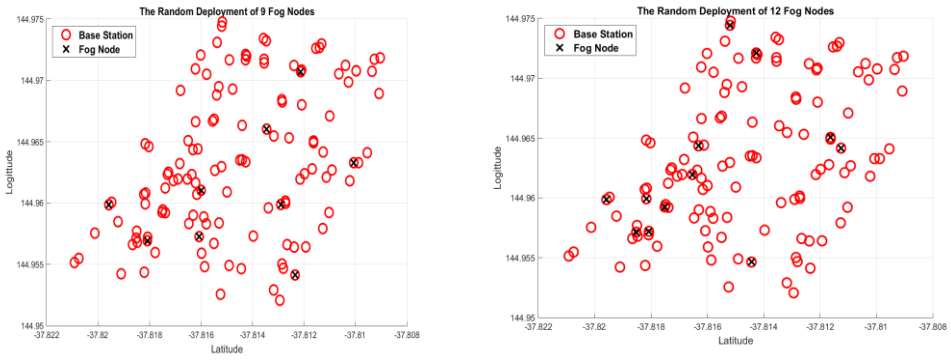


(a) 4 Nodes

(b) 6 Nodes

Figure 4-7 The Fog Node Random Deployment (4 and 6 nodes)

The random deployment approach is illustrated in Figures 5.9 and 5.10; the fog nodes are not equally installed (distributed) in the network.



(a) 9 Nodes

(b) 12 Nodes

Figure 4-8 The Fog Node Random Deployment (9 and 12 nodes)

As we can notice, some fog nodes deployed close to each other, leaving the number of base stations without fog node coverage.

Thus, a high distance between the fog nodes and the base stations. The K-medoids clustering techniques illustrated in Figure 5.11; the base station set is divided into K set (here K =12). The base station located almost centrally in each cluster will be a suitable candidate for fog node installation.

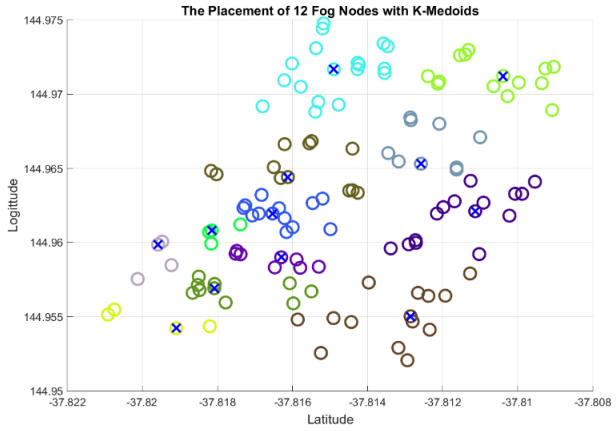


Figure 4-9 The Fog nodes K-Medoids placement (12 Fog nodes)

The distribution of fog nodes covered the base station equally; the average distance between the Fog nodes and base stations on the cluster is minimal compared to the random deployment illustrated in Figure 5.12 (b).

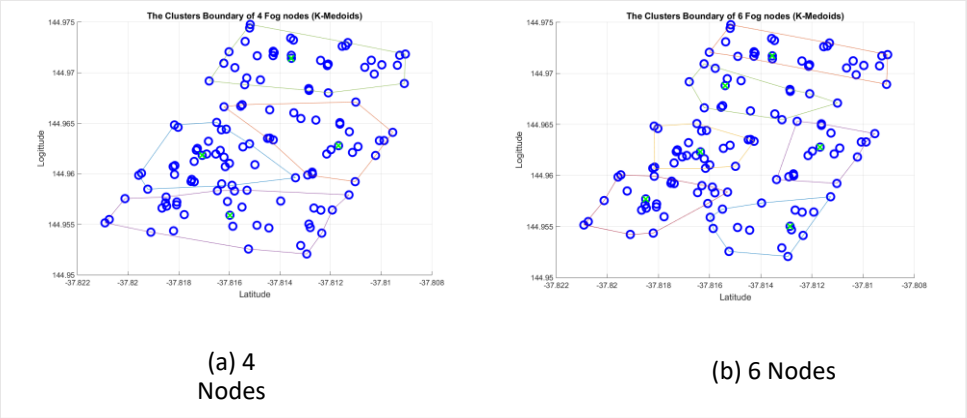


Figure 4-10 The Clusters Boundary for different number of Fog Nodes (4 and 6 nodes)

Figures 5.12 and 5.13 illustrated the definition of the cluster for a different number of fog nodes. We can notice, as the number of clusters increases (i.e., increasing the fog nodes), the distance between the base stations and fog nods decreases.

Hence, the access delay between the mobile users and fog nodes improved. However, the installation of more fog nodes comes with the deployment cost.

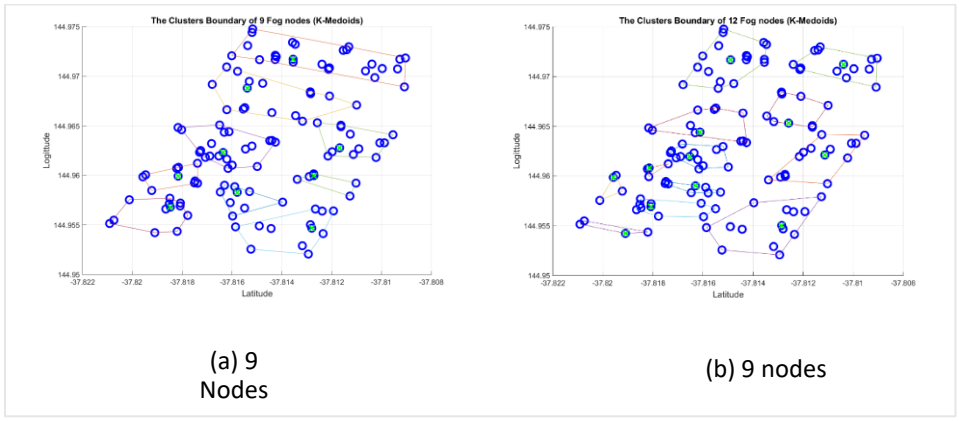


Figure 4-11 The Clusters Boundary for different number of Fog Nodes (9 and 12 nodes)

4.1.3 Conclusion

In this work, we have studied via simulation experiments the impact of the number of fog nodes on network performance. We observed that increasing the number of fog nodes will decrease the response time of service requests, which in turn improves the end-to-end latency and QoS at the user side. As we noticed from Figure 4-1, and Figure 4-2, the system will be stable after a certain number of fog nodes are deployed; 5 nodes in case of λ = 250, 10 nodes for λ = 500 and so on.

After reaching the stability, the number of fog nodes can be increased to achieve the QoS requirements as illustrated in Figures Figure 4-3, and **Error! Reference source not found.**,

when $\lambda = 250$ we need to deploy about 4 fog nodes to satisfy the QoS constraint (T_{QoS}). Same case for $\lambda = 375$, the fog nodes installed in the network should be greater than or equal to 9 nodes.

The simulation results show that, the workload increased as the number of D2D clusters increased, then adding more fog nodes to the network, one gets a decrease of the workload per server. For example, when $\lambda = 250$ (in Figure 4-4), with 5 fog nodes and above, the fog nodes work under 60%. Thus, efficient load balancing is needed to design for optimizing the network performance.

For fog nodes placement, We observed that random deployment techniques do not ensure equal workload distribution among the fog nodes as shown in Figures Figure 4-7, and Figure 4-8. Meaning, group of base stations will connect to the fog nodes located far away from their positions. Thus, high latency resulting (i.e., access delay) affects the network performance. As we noticed from Figure 4-9, the base stations have been divided into K clusters, and the fog nodes installed at the base station are located almost in the center of that cluster. Moreover, as the number of clusters increases, the distance between fog nodes and base stations decreases, as shown in Figures Figure 4-10, and Figure 4-11.

4.2 The SDN-Load Balancing for Fog Nodes Networks

The simulation scenarios assumed there is a number of D2D clusters that generate service tasks, and these are considered heterogeneous in terms of their length (i.e., size) and processing computing requirements. Hence, different execution times are required for the service tasks.

As mentioned, each fog node running a number of VMs (equal to 15, 20, 25, 30 VMs) and each VM has certain specifications including the different computation power (1000 and 2000) $MIPS$, number of processing elements, memory size, and others. Moreover, the number of D2D clusters in the network varied from 100 to 1000 clusters, and each cluster generates two types of services that are different in size (500-2500 and 3000-5000 instruction per second). Table 5.6 illustrated the features of entities used in this simulation.

Table 4.4 The Parameters Description

Parameters		**Description**
D2D clusters	Number of D2D clusters	Between 100-1000
	Length of Task (Instruction Per Millisecond)	500-2500 (Small Size Tasks) 3000-5000 (Large Size Tasks)
Virtual Machines	Processing Power ($MIPS$)	1000 and 2000

Memory Size (RAM)	512MB
Number of cores	1
Number of VMs	15, 20, 25, 30 VMs

In this work, the CloudSim [24] will be used for modeling the above simulation scenario. this work considers three First Come First Serve (FCFS) technique that used in comparison to the GA algorithm respect to the following performance metrics:

1. Makespan (MS): Makespan is defined as the overall completion time required to execute all tasks in the system. The lower value of makespan indicated the efficiency of algorithms or techniques. The MS determined with equation 30

2. Resource Utilization (RU): considered as desire criteria for service providers, and it measures the utilization of resources (the leverage of available resources). Hence, the Highest value is desirable. The RU is computed as shown in equation 33:

$$RU = \frac{Total\ completion\ Time\ of\ all\ Tasks}{makespan} \tag{4.1}$$

3. Degree of Imbalance (DI): measured the amount of load distribution among nodes (VMs in our case). The small value shows that the load of the system is more balanced. It is determined as:

$$DI = \frac{T_{max} - T_{min}}{T_{avg}} \tag{4.2}$$

Where T_{max} and T_{min} presented the maximum and minimum execution time among all VMs and T_{avg} denoted the total average execution time.

The First Come First Serve (FCFS) algorithm is a non-preemptive algorithm. The service task that comes first will be completed its execution first and then the other process be required to wait in queue until the execution is completed [25].

Several parameters are needed to set for the genetic algorithm operation, as illustrated in Table 5.7. mainly, the crossover rate (C_R) set to 0.95 and mutation probability (M_p) to 0.6.

Table 4.5 The Genetic Algorithm Parameters

Genetic Parameters	Description
Population size	50
Selection method	Roulette-wheel
Crossover rate	0.95
Crossover type	Two-points
Mutation rate	0.6

In this work, three simulation scenarios are considered as follow:

- The first scenario considered a fixed number of VMs (about 15 VMs) and different numberD2D clusters that generate service tasks between 500-2500 and 5000-3000 Instruction. in this scenario, the performance metric mentioned early will measure and analysis between the above algorithms.
- In the second scenario, the number of service tasks will be fixed to a maximum number of tasks received (about 1000 Tasks), where the number of VMs varied from 15 to 30 VMs. Further, the $akespan$,RU, and DI will be analyzed, respectively.
- The third scenario considers two types of VMs (Low and high VMs) that different in processing power. The low VMs has the processing power of about 1000 $MIPS$ where the high VMs has a processing power of about 2000 $MIPS$. Moreover, the number of VMs varied from 10 to 30.

4.2.1 The First Scenario (Different Service Tasks Size)

As mentioned early, the number of D2D clusters distributed in the network between 100 to 1000 clusters, and these clusters generated service tasks in two types of small size (500-2500) and large size (3000-5000) and a number of VMs in the system fixed to 15.

Figure 5.16 shown the performance of two load balancing techniques under a different number of D2D clusters; the FCFS technique gives an almost similar makespan value to the GA technique at a low number of clusters. This case could be noticed when the number of D2D clusters is between 100 to 300. However, as the number of clusters increased, the FCFS provides a high makespan compared to the GA algorithm.

For example, when we have about 600 D2D clusters, the FCFS gives the value of makespan above 0.05, and in the case of GA, the value of makespan is below 0.05. Moreover, the number of D2D clusters reached 900 to 1000; the GA performed better than the FCFS.

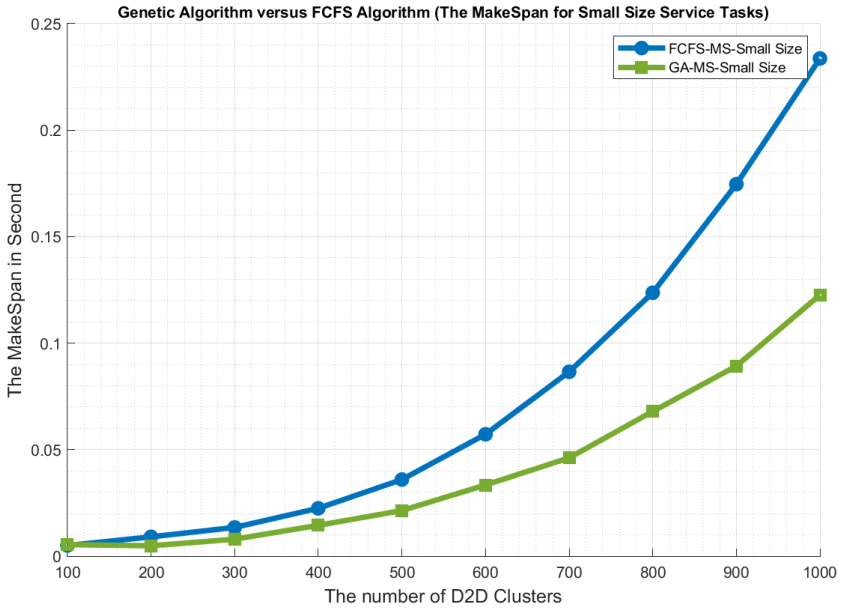


Figure 4-12 The Makespan of GA vs. FCFS (small size service tasks)

The next figure showed the case when D2D clusters generated large size service tasks. As noticed, the GA provides a low makespan value in comparison to the FCFS technique.

where the FCFS suffered from high makespan value as the number of clusters increased in the network.

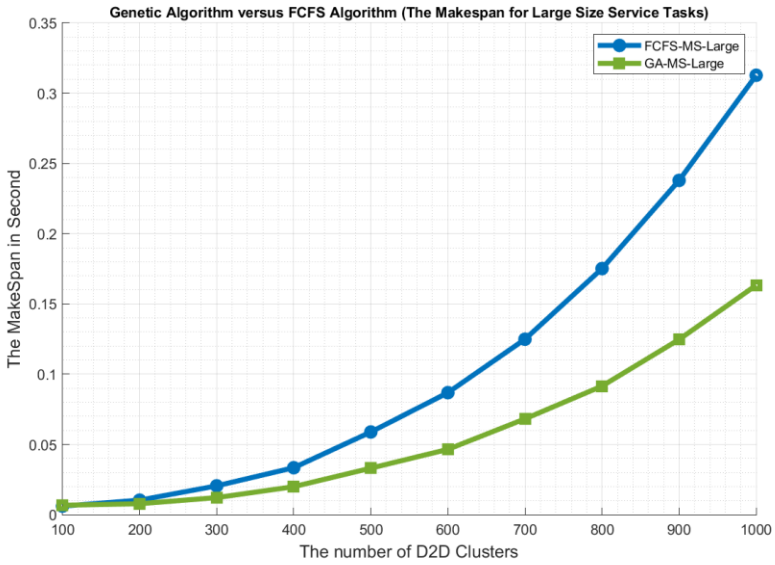


Figure 4-13 The Makespan of GA vs. FCFS (Large size service tasks)

Figure 5.18 illustrated the overall performance of GA and FCSF algorithms under small and large size service tasks.

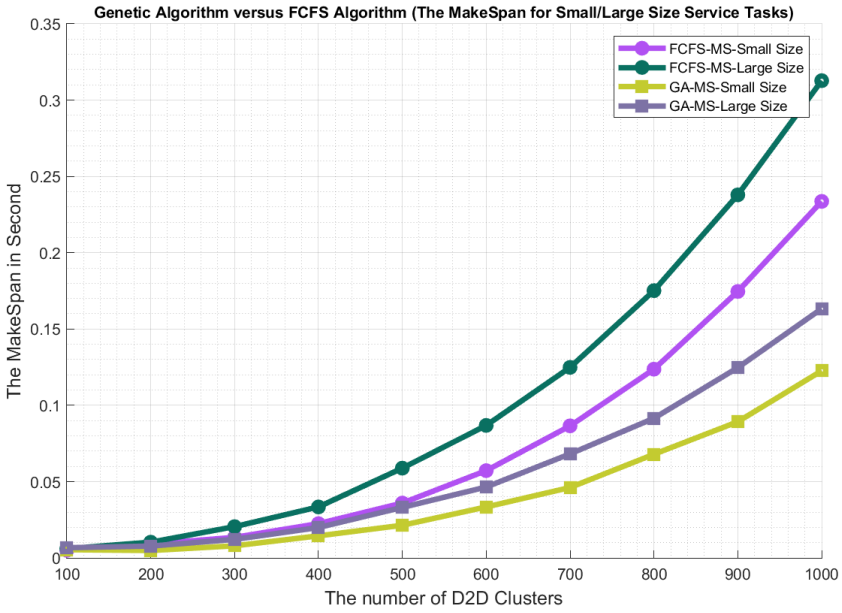


Figure 4-14 The The Makespan for small/large size of service tasks

The Degree of Imbalance (DI) for small-size service tasks is shown in Figure 5.19. as mentioned earlier, The degree of imbalance measured the amount of load distribution VMs in our case. The lowest value showed that the VMs received an almost equal amount of load.

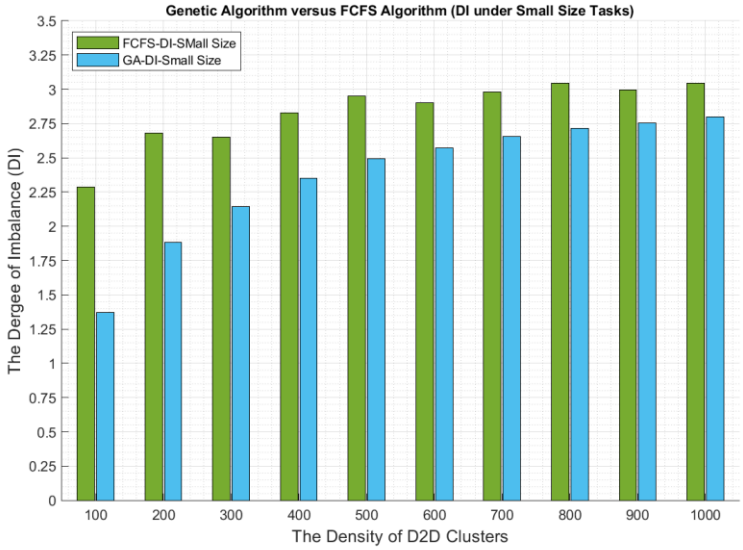


Figure 4-15 The Degree of Imbalance (small size tasks)

The GA algorithm gives a low value of DI compared to FCSF, as shown in the figure above, with a different number of the D2D clusters.

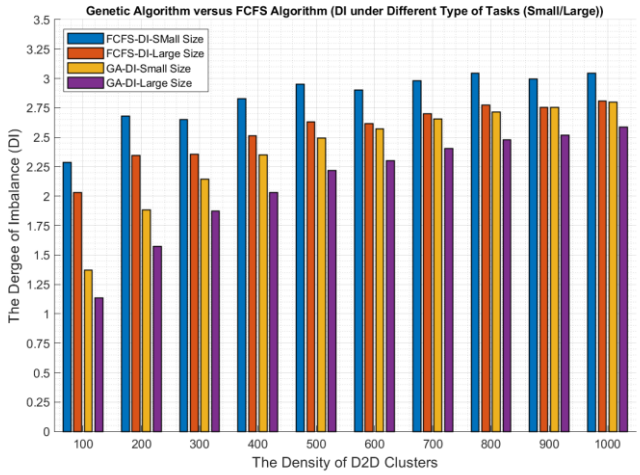


Figure 4-16 The Degree of Imbalance (Different types of service tasks)

Figure 5.20 illustrated the performance of GA and FCFS of small and large size service tasks with a different number of D2D clusters. For example, when the number of clusters in the network is equal to 500, the DI with GA has a value below 2.5, and on the other hand, the value of DI with FCFS about 3.5. the same case when the D2D clusters equal to 1000, the DI value equal about 2.75 (GA) and almost 3.0 for FCFS.

4.2.2 The Second Scenario (Different VMs deployment)

In this scenario, the number of D2D clusters in the network will be fixed to 1000 clusters where the number of VMs deployed varied from 15 to 30. Moreover, two types of service tasks (small and large) are considered in this section.

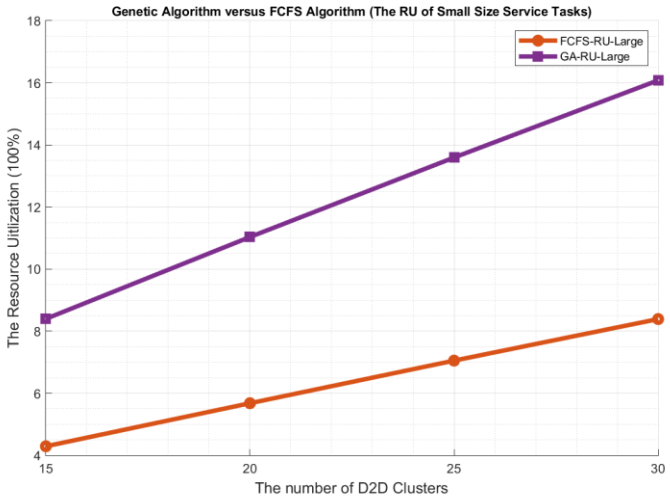


Figure 4-17 The RU of GA vs. FCFS (Small Size Service Tasks)

As mentioned, the highest value of resource utilization (RU) is desirable. However, the FCFS provides the lowest value of RU in comparison to GA.

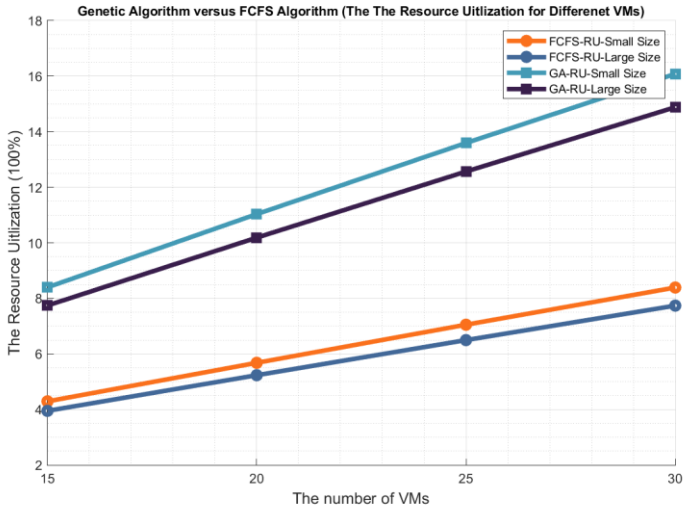


Figure 4-18 The Resource Utilization for Different VMs (small and Large size service tasks)

Hence, as increased the number of VMs; the GA manages the resource efficiently. For example, where the number of VMs equal to 20; RU value is equal to 13%; on the other hand, the FCFS utilized the resource about 5% (see Figure 5.21).

Figure 5.22 shown the resource utilization with different types of service tasks. As we notice, as increased the number of VMs, the GA performs better than the FCFS.

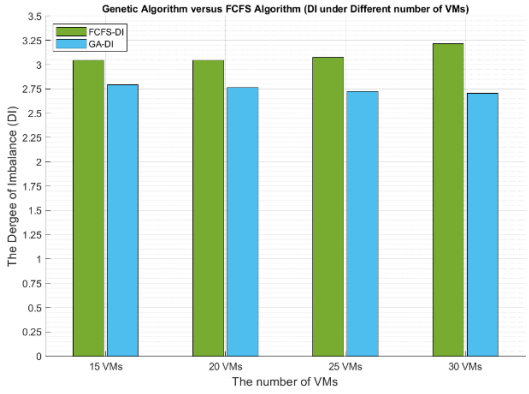


Figure 4-19 The Degree of Imbalance (Different number of VMs)

However, the GA with large size service tasks provides a low value compared to small size service tasks; the reason is that the GA provides a high makespan as shown in figure 5.22. The same case for the FCFS algorithm.

Figure 5.23 illustrated the performance of the GA under different of VMs in the term of The Degree of Imbalance.

The GA provides a low value of DI with a slightly different value as increased the number of VMs. For example, at 15 VMs, the value of DI is about 2.76, and at 30 VMs, about 2.6.

4.2.3 The Third Scenario (Low and High VMs)

In this scenario, two types of VMs considered (low VMs with the computation power of about 1000 $MIPS$ and high VMs with 2000 $MIPS$), and the number of VMs varied from 10 to 30. Also, the density of D2D clusters will be fixed to 1000 clusters that generated small and large service tasks.

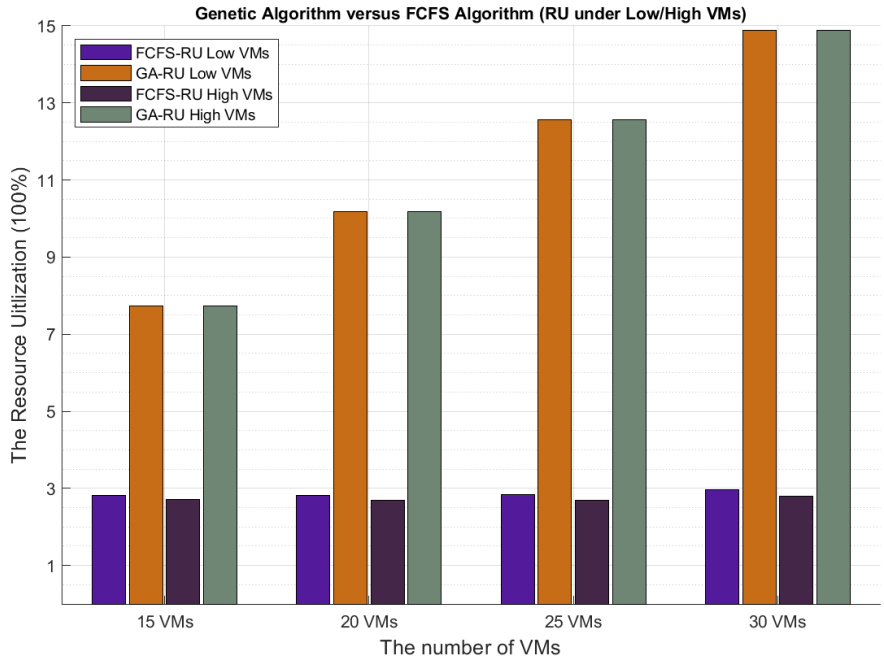


Figure 4-20 The Resource Utilization for Low/High VMs

As illustrated in Figure 5.24, the FCFS suffers from low resource utilization compared to the GA algorithm. For example, when the number of VMs deployed is equal to 15, the RU at FCFS below 3%, and on the other hand, the GA gives high resource utilization equal to about 7%.

The performance of FCFS and GA algorithms in terms of load distribution is illustrated in Figure 5.25. As it is noticed, the increased number of VMs has no effect on FCFS performance to reach a similar DI value that provided from the GA.

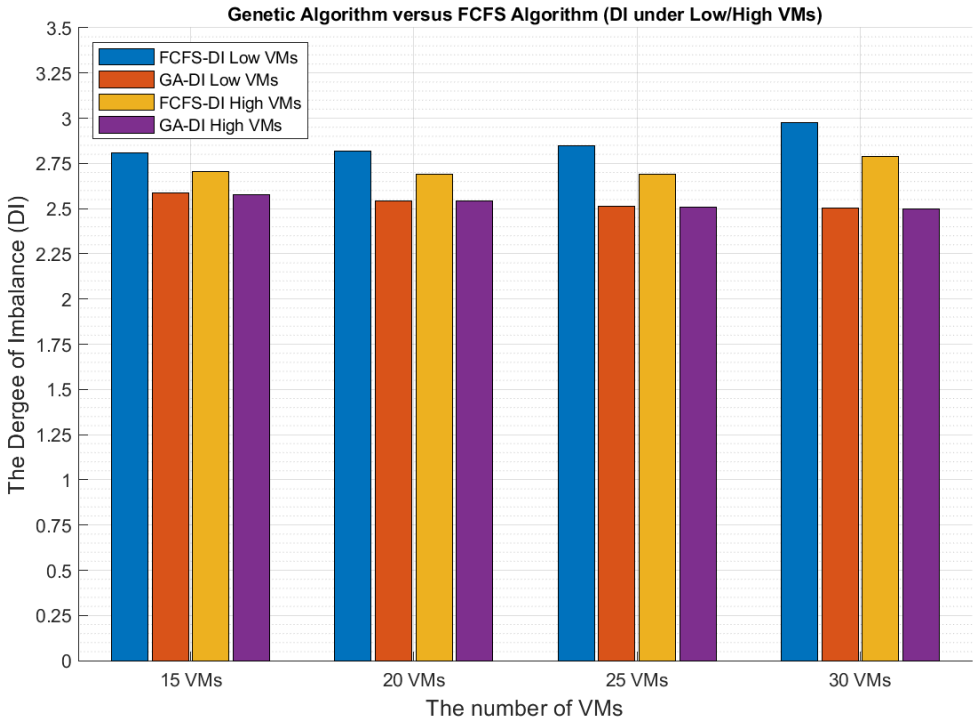


Figure 4-21 The Degree of Imbalance (Low/High VMs)

4.2.4 Conclusion

In this section, the load balancing issues in fog computing have been studied and modeled mathematically. The load balancing issue is considered an NP-Hard problem. Consequently, the genetic algorithm has been considered as the main technique to solve load balancing. The simulation experiments shown, the genetic algorithm performed better than other algorithms, namely, the FCFS in terms of makespan, resource utilization, and imbalance factor.

We observed that the genetic algorithm gives a better value of makespan than FCFS and shown in Figures Figure 4-12, Figure 4-13, and Figure 4-14. As illustrated, for example, in Figure Figure 4-12 when we have about 600 D2D clusters, the FCFS gives the value of makespan above 0.05, and in the case of GA, the value of makespan is below 0.05.

The simulation results show that the genetic algorithm provides low Degree of imbalance (low value is desirable) about 2.4 at 500 clusters where FCFS has 3.0 of DI for small-sized services tasks, as shown in Figure Figure 4-15. The FCFS performs worst compared to the genetic algorithm, as seen in Figure Figure 4-16 in both cases of service task types.

The second scenario illustrated the case of a different number of VMs installed in the network and the maximum amount of service tasks. As the number of VMs increased, the GA algorithm provides a high resource equal to 13%, and when the number of VMs equals 20 and on the other hand, the FCFS suffers from the lowest utilized resource (Figure Figure 4-19). The performance of both techniques under different service tasks characteristics shows that the GA has the highest resource utilization, as seen in Figure Figure 4-18; at 20 VMs, the GA gives a value of RU about 10% (small size) and 12 (large size).

The third scenario considered the different processing power of VMs. The genetic algorithm (GA) has high resource utilization in both cases (low and high VMs), as seen in Figure Figure 4-20, when the number of VMs deployed is equal to 15 RU at FCFS below 3%. On the other hand, the GA gives high resource utilization equal to about 7%. As noticed in Figure Figure 4-21, the increased number of VMs has no effect on FCFS reaching a similar DI value from the GA.

Chapter 6

5 Discussions and Future Work

In this work, it has been investigated the integration of fog computing to D2D communication. Hence, the main contribution of this work is determining the required number of fog nodes to provide a Fog service to D2D users and their candidate locations in the network, also distributing the workload equally among the fog nodes. The fog computing is deployed as computation nodes that provide the fog service to the D2D clusters.

This work has used the multi-server queue model to determine the required number of fog nodes in the network. Additionally, the remote cloud servers also have been modeled as multi-server queue models. Further, the deployment of the fog nodes has been implemented according to the clustering algorithms, namely, K-medoids. Moreover, the problem of load balancing has been formulated with expect time to complete matrix (ETC). With SDN involvement, the Genetic algorithm has been considered the main approach to solving the load balancing issue. A suggestion for future works can be summarized as follow:

1. Considering user mobility while maintaining the services connectivity that offloaded to the Fog Nodes network. Moreover, these services could have different characteristics, such as latency.
2. Considering SDN Orchestration of different slices (e.g., Vehicular Network slice and the mobile network slice). the SDN Orchestration maintaining and managing the provided services of different slice.

List of Scientific Reports

This list comprises only research reports doctoral program.

1. Scientific Report No. 1, *Device to Device communications in 4G and 5G technologies*, Polytechnic University in Bucharest, Doctoral School of Electronics, Telecommunications and Information Technology, SD-ETTI-B, Contract of university studies, Doctoral, March 2017.

2. Scientific Report No. 2, *Resource Management and Control in D2D based on SDN/NFV*, Polytechnic University in Bucharest, Doctoral School of Electronics, Telecommunications and Information Technology, SD-ETTI-B, Contract of university studies, Doctoral, November 2017.

3. Scientific Report No. 3, *Centralised Multi-Hop Routing for D2D Communication*, Polytechnic University in Bucharest, Doctoral School of Electronics, Telecommunications and Information Technology, May-2019.

4. Scientific Report No. 4, *Fog Computing and D2D Networks Integration*, Polytechnic University in Bucharest, Doctoral School of Electronics, Telecommunications and Information Technology, March 2021.

The Summary of the Original Contributions

The Summary the main contributions of this thesis are:

1. Modelling fog computing as a network of fog nodes network (FCN) with the Queueing theory.
2. The optimal number of fog nodes has been determined through a series of equations formulated under QoS constraints.
3. The simulation results shown as increasing the number of fog nodes will decrease the response time of service request and satify the QoS threeshold.
4. The simulation results show that, the workload increased as the number of D2D clusters increased. Moreover, as adding more fog nodes to the network, the workload decreased per fog nodes.

5. K-Medoids clustering technique adopted for fog nodes deployment with ensuring minimum distance between the base stations and Fog nodes.
6. The base station set in the network divided into serval clusters, and in each cluster a Fog nodes will be deployed almost in center of that clusters.
7. The simulation results shown that as as the number of clusters increases, the distance between fog nodes and base stations decreases. In other words, the distance between the Fog Node and base station is reduced as increase the number of clusters.
8. The simulation results shown the integration of SDN to fog computing showed the lowest response time value compared to the cloud remote servers.
9. A different characteristics of the Fog Node considered including variety number of virtual machines, low and high processing power of the virtual machines.
10. The simulation results shown, the genetic algorithm performed better than FCFS algorithm in terms of makespan, resource utilization, and imbalance factor.

Publications

Mustafa Khaleel Hamadani, and Eugen Borcoci. "Fog Nodes Placement for D2D Networks." In *2021 44th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 152-156. IEEE, 2021. **[Accepted, Invited to be published in an extended form for the Sensors Journal]**. Link: https://ieeexplore.ieee.org/document/9522637

Mustafa Khaleel Hamadani, and Eugen Borcoci. "Fog Computing and D2D Networks Integration." In *2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pp. 1-6. IEEE, 2021. **[Accepted]**. Link: https://ieeexplore.ieee.org/document/9527740

Mustafa Khaleel Hamadani, Mahdi AA. Centralised Multi-hop Routing for Device-to-Device communication: simulation and results. In2019 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI) 2019 Jun 27 (pp. 1-6). IEEE. **[Accepted]**. Link: https://ieeexplore.ieee.org/document/9042024/

Mustafa Khaleel Hamadani, Al-Alwash Husam Mahdi, Centralised Multihop Routing Techniques for Device-toDeviceCommunication, ICN 2019: The Eighteenth International Conference on Networks, ARIA, 2019. ISBN: 978-1-61208-695-8. **[Accepted]**. Link: http://www.thinkmind.org/index.php?view=article&articleid=icn_2019_1_30_30040

Al-Alwash Husam Mahdi, **Mustafa Khaleel Hamadani**, Vehicular to Grid echnologies– A Survey on Architectures and Solutions, ICN 2019 : The Eighteenth International Conference on Networks, ARIA, 2019. ISBN: 978-1-61208-695-8. **[Accepted]**. Link: http://www.thinkmind.org/index.php?view=article&articleid=icn_2019_2_20_30045

Silviu-Andrei Lazar, **Mustafa Khaleel Hamadani**, Carmen-Elena Stefan, Optimization Analysis of VANET's Control Plane for Safety Application Traffic, 2018 International Conference on Communications (COMM). **[Accepted]**. Link: https://ieeexplore.ieee.org/document/8484272/

Mustafa Khaleel Hamadani, and Eugen Borcoci. " Load Balancing Techniques for Fog Computing Integrated to D2D Networks.". **[submitted to Scientific Bulletin - University POLITEHNICA of Bucharest]**.

Reference

[1] M. Series, “IMT Vision--Framework and overall objectives of the future development of IMT for 2020 and beyond,” *Recommendation ITU*, vol. 2083, p. 21, 2015.

[2] G. Fodor *et al.*, “An overview of device-to-device communications technology components in METIS,” *Ieee Access*, vol. 4, pp. 3288–3299, 2016.

[3] G. Araniti, A. Raschella, A. Orsino, L. Militano, and M. Condoluci, “Device-to-device communications over 5G systems: Standardization, challenges and open issues,” in *5G mobile communications*, Springer, 2017, pp. 337–360.

[4] R. I. Ansari *et al.*, “5G D2D networks: Techniques, challenges, and future prospects,” *IEEE Systems Journal*, vol. 12, no. 4, pp. 3970–3984, 2017.

[5] X. Lin, J. G. Andrews, A. Ghosh, and R. Ratasuk, “An overview of 3GPP device-to-device proximity services,” *IEEE Communications Magazine*, vol. 52, no. 4, pp. 40–48, 2014.

[6] O. Hayat, R. Ngah, S. Z. M. Hashim, M. H. Dahri, R. F. Malik, and Y. Rahayu, “Device discovery in D2D communication: A survey,” *IEEE Access*, vol. 7, pp. 131114–131134, 2019.

[7] Z. Zhu, X. Luo, K. Wang, and Y. Yang, “Integration of D2D with Edge/Fog Computing,” *Wiley 5G Ref: The Essential 5G Reference Online*, pp. 1–18, 2019.

[8] P. Mell, T. Grance, and others, “The NIST definition of cloud computing,” 2011.

[9] C. M. Mohammed, S. R. M. Zeebaree, and others, “Sufficient comparison among cloud computing services: IaaS, PaaS, and SaaS: A review,” *International Journal of Science and Business*, vol. 5, no. 2, pp. 17–30, 2021.

[10] V. Prokhorenko and M. A. Babar, “Architectural resilience in cloud, fog and edge systems: A survey,” *IEEE Access*, vol. 8, pp. 28078–28095, 2020.

[11] M. Chen, Y. Hao, Y. Li, C.-F. Lai, and D. Wu, “On the computation offloading at ad hoc cloudlet: architecture and service modes,” *IEEE Communications Magazine*, vol. 53, no. 6, pp. 18–24, 2015.

[12] M. Firdhous, O. Ghazali, and S. Hassan, “Fog computing: Will it be the future of cloud computing?,” 2014.

[13] O. C. A. W. Group and others, “OpenFog reference architecture for fog computing,” *OPFRA001*, vol. 20817, p. 162, 2017.

[14] Y. P. Llerena and P. R. L. Gondim, “SDN-controller placement for D2D communications,” *IEEE Access*, vol. 7, pp. 169745–169761, 2019.

[15] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, “Mobile edge computing—A key technology towards 5G,” *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[16] R. Roman, J. Lopez, and M. Mambo, “Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges,” *Future Generation Computer Systems*, vol. 78, pp. 680–698, 2018.

[17] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, “Mobile edge computing—A key technology towards 5G,” *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[18] Z. Pang, L. Sun, Z. Wang, E. Tian, and S. Yang, “A survey of cloudlet based mobile computing,” in *2015 International Conference on Cloud Computing and Big Data (CCBD)*, 2015, pp. 268–275.

[19] A. V. Dastjerdi and R. Buyya, “Fog computing: Helping the Internet of Things realize its potential,” *Computer*, vol. 49, no. 8, pp. 112–116, 2016.

[20] S. Yi, Z. Hao, Z. Qin, and Q. Li, “Fog computing: Platform and applications,” in *2015 Third IEEE workshop on hot topics in web systems and technologies (HotWeb)*, 2015, pp. 73–78.

[21] P. Habibi, M. Farhoudi, S. Kazemian, S. Khorsandi, and A. Leon-Garcia, “Fog computing: a comprehensive architectural survey,” *IEEE Access*, vol. 8, pp. 69105–69133, 2020.

[22] V. Prokhorenko and M. A. Babar, “Architectural resilience in cloud, fog and edge systems: A survey,” *IEEE Access*, vol. 8, pp. 28078–28095, 2020.

[23] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, “Software-defined networking (SDN): Layers and architecture terminology,” *RFC 7426*, 2015.

[24] F. Hensh, M. Gupta, and M. J. Nene, “Mist-Edge-Cloud (MEC) Computing: An Integrated Computing Architecture,” in *2021 Second International Conference on*

Electronics and Sustainable Communication Systems (ICESC), 2021, pp. 1035–1040.

[25] A. Islam, A. Debnath, M. Ghose, and S. Chakraborty, “A survey on task offloading in multi-access edge computing,” *Journal of Systems Architecture*, vol. 118, p. 102225, 2021.