# Edge Server Deployment in 5G Network using K-Means Clustering Algorithm

Pravallika Katragunta
Vishnu Priya Guddeti
Maryam Havakeshian
5G Networks
Carleton University, Canada

March 28, 2021

## Contents

**1 Introduction** **2**
1.1 Advantages of edge computing in 5g: . . . . . .3
1.2 Performance measures: . . . . . .4
1.3 Machine Learning . . . . . .4

**2 Related Work** **5**

**3 Problem Statement** **5**
3.1 Problem formulation . . . . . .6
3.2 Evaluation metric . . . . . .7
3.2.1 Processing Delay . . . . . .7
3.2.2 Transmission Delay . . . . . .7
3.2.3 Backhaul Delay . . . . . .7
3.2.4 Service Delay . . . . . .8

**4 Proposed Algorithm** **8**
4.1 K-Means . . . . . .9
4.2 PSO (Particle Swarm Optimization) . . . . . .11

**5 Empirical Comparison** **13**
5.1 Results . . . . . .15

**6 Conclusion** **17**

**7 Future Work** **18**

**Abstract**

**Cloud computing is a significant innovation for bringing a major pool of flexible assets to customer gadgets. Their principle disadvantage has for some time been the significant distance among users and servers, however, this has been cured by Edge Cloud Computing, where the cloud servers are situated in the network edge. Edge Cloud Computing is viewed as fundamental for future networks and therefore, there is a lot of exploration on the best way to upgrade its activity. Notwithstanding, by far most of them disregard the choice of where the edge servers ought to be sent, regardless of what seriously this can mean for the presentation of the framework.[1] Besides, future networks should likewise manage massive measures of customers and servers, for example, the ones normal for the Internet of Things and 6G Networks. This demands arrangements that are versatile. Given these two focuses, we propose a Machine Learning-based server sending strategy in 6G Internet of Things conditions. Our answer is demonstrated to move toward optimality while being achievable. Moreover, we likewise demonstrate that our proposition prompts lower latency and higher asset proficiency than regular Edge Cloud Computing server sending arrangements.[2]**

**Keywords:** Edge Cloud Computing, machine learning, artificial intelligence, cloudlet, clustering, 6G, Internet of Things, Transmission Power Control

# 1 Introduction

5G is anticipated as the cutting edge wireless cell network to cater to the necessities of cut- ting edge networks. 5G has three principle attributes concealed in previous generation networks.[1] Firstly, a massive measure of information is produced. As per the Inter- national Telecommunication Union (ITU), Secondly, severe QoS necessities are forced to help highly intuitive applications, requiring super-low latency and high throughput. Thirdly, the heterogeneous climate should be upheld to permit between operability of an assorted scope of UEs (e.g., cell phones and tablets), QoS prerequisites (e.g., various degrees of latency and throughput for sight and sound applications), network types (e.g., IEEE 802.11 and Internet of things, etc. 5G is contained three principle innovations to give higher network limit to help a higher number of UEs.[3]

Firstly, mmWave correspondence, which utilizes high-recurrence groups (i.e., 30 GHz to 300 GHz [19]), gives high transmission capacity (i.e., at any rate, 11 Gbps). Also, small cell arrangement permits UEs to impart utilizing mmWave to lessen transmission reach and obstruction. Thirdly, massive MIMO (multiple-input multiple-output) permits base stations (BSs) to utilize an enormous number of reception apparatuses (e.g., up to 16 receiving wires for each area) to give directional transmission (or beamforming) to lessen obstruction, permitting adjoining nodes to communicate at the same time.

URLLC (Ultra-reliable low latency) is proposed to cover both human and machine-centric communication and later on referred to as critical machine type communication (C-MTC). It is portrayed by use cases with severe necessities for latency, reliability, and high acces- sibility. Some of the examples include vehicle-to-vehicle communication (V2V) including security, remote control of industrial hardware, far off clinical medical procedure, and

appropriation mechanization in a shrewd lattice. An illustration of a human-driven use case is 3D gaming where the low-latency necessity is likewise joined with exceptionally high information rates.[3]

Data can be arranged into three primary classifications as per its time qualities as follows:

- Hard continuous data has a severe predefined latency. Applications, for example, video web-based, gaming, and medical care administrations produce this sort of data.
- Soft constant data has a predefined latency, yet it can endure some pre-characterized and limited latency. Applications, for example, insightful traffic light control frame- works, produce this sort of data.
- Non-ongoing data isn't time-touchy and can endure latency.

**Cloud computing** Cloud computing is the on-demand accessibility of PC framework assets, particularly data storage (cloud storage) and computing power, without direct dynamic administration by the client. The term is by and large used to portray data fo- cuses accessible to numerous clients over the Internet. Enormous clouds frequently have capacities disseminated over numerous areas from focal servers. On the off chance that the association with the client is generally close, it could be assigned an edge server.[3]

**Edge Computing** Edge computing is imagined to deal with applications and adminis- trations with hard continuous necessities utilizing edge workers because of their closeness to UEs prompting a critical decrease in latency. For applications and administrations with delicate genuine-time necessity, or limited start to finish delay, assignments are dealt with by edge workers if the reaction delay among UEs and the cloud is higher than the prerequisite; in any case, the undertakings can be offloaded to the cloud. For appli- cations and administrations with non-continuous prerequisites, errands can be offloaded to the cloud for load adjusting.[1]

## 1.1 Advantages of edge computing in 5g:

- Edge computing offloads a massive measure of data from UEs to edge clouds. While edge servers offer dispersed nearby storage for a lot of data, yet their storage is a lot lower than that in the cloud, which has essentially a limitless storage limit. Instances of data being put away are computing procedures (e.g., calculation offloading tech- nique), metadata (e.g., timestamps and geological areas), checking data. The edge server gives various sorts of storage methodologies to help various types of data. For example, vaporous storage gives brief data storage to a bunch of interconnected cell phones.[1]
- Edge computing measures and performs basic and ongoing data investigation on a massive measure of crude data assembled from various applications in nearness to create important data. The ability to make data examination locally decreases the latency needed to send data to, just as to sit tight for reactions from, the cloud. Thus, the results of the nearby data examination are utilized for dynamic.[3]

- Edge computing fills in as an extra layer between the cloud and associated gadgets to improve network security, incorporating UEs with restricted assets. The edge clouds can fill in as gotten disseminated stages that give security accreditations to the executives, malware discovery, programming patches appropriation, and dependable interchanges, to recognize, approve, and countermeasure assaults.[1]

- The benefit is that, because of the closeness of edge computing, vindictive sub- stances can be immediately identified and disengaged, and continuous reactions can be started to improve the impacts of the assaults. This assists with limiting help interruptions. What's more, the versatility and measured quality nature, just as the abilities, of edge computing can encourage the arrangement of blockchain among UEs with restricted capacities.[3]

## 1.2 Performance measures:

- Edge computing diminishes the operational expense by giving neighborhood capacities, rather than offloading (or sending) errands and data to the cloud. This diminishes offloading overhead (or decreases network asset utilization), like transmission capacity.

- Edge computing improves QoS by giving nearby capacities. This decreases the number of undertakings and data offloaded to the cloud, thus it builds network execution (e.g., higher throughput and lower latency), which are essential to delay- delicate applications (e.g, distant medical procedure and internet gaming).

- Edge computing lessens energy utilization by giving neighborhood capacities. This decreases the measure of energy caused to offload assignments and data to the cloud (i.e., the energy brought about in the correspondence), thus it builds network lifetime.

## 1.3 Machine Learning

Machine Learning is additionally a decent option in highly unique situations, as it can rapidly deal with little changes in the issue while as yet offering low execution overhead. This is all conceivable because the Machine Learning programs investigate the examples in the issue and learn through them what is the most effective method of settling or searching for an answer for the issue.[2]

Hence, Machine Learning has been effectively used to take care of issues in a wide range of sorts of networks, for example, digital actual frameworks, portable impromptu net- works, and sensor networks. This paper proposed a cloudlet arrangement strategy for ECC dependent on the Machine Learning calculations k-Means Clustering (kMC) and Particle Swarm Optimization (PSO). The target of our proposition is to discover a send- ing arrangement where cloudlet assets are utilized as proficiently as could be expected while administration delay is limited.[2]

# 2 Related Work

Instinct reveals to us that choosing the area of the cloudlets dependent on a pre-decided target capacity would bring about better assistance. Regardless of this, by far most of ECC research works with the assumption that servers are ineffectively characterized and changeless positions. Since no organization strategy is expressly referenced, it is protected to say that the cloudlets are arbitrarily positioned in the help territory. Some exploration works straightforwardly state so. In any case, overlooking cloudlet arrangement choices implies the last assistance isn't advanced, regardless of whether the cloudlets themselves are designed such that asset designation is improved. Subsequently, to offer the most ideal assistance (for our situation, the one with the least deferral), we will cautiously choose were to convey the cloudlets.[1]

A couple of exploration works proposed cloudlet arrangement strategies of their own and uses a Markov bind design to model the entire ECC administration, including both cor- respondence and calculation stages. This model is then used to foresee the responsibility for each cloudlet in every conceivable position and at that point choose arrangement so that there is no over-burden. Be that as it may, their answer isn't actually adaptable and would bring about very long execution times in situations with numerous servers and, all the more worryingly, base stations. Accordingly delivering their proposition unworkable for IoT and 6G. A few creators offer a more adaptable proposition for the cloudlet orga- nization issue and use Machine Learning looking like PSO to choose where to send the cloudlets. They model the energy consumption of the entire framework and afterward use PSO to discover the areas that lead to higher energy productivity. [1]

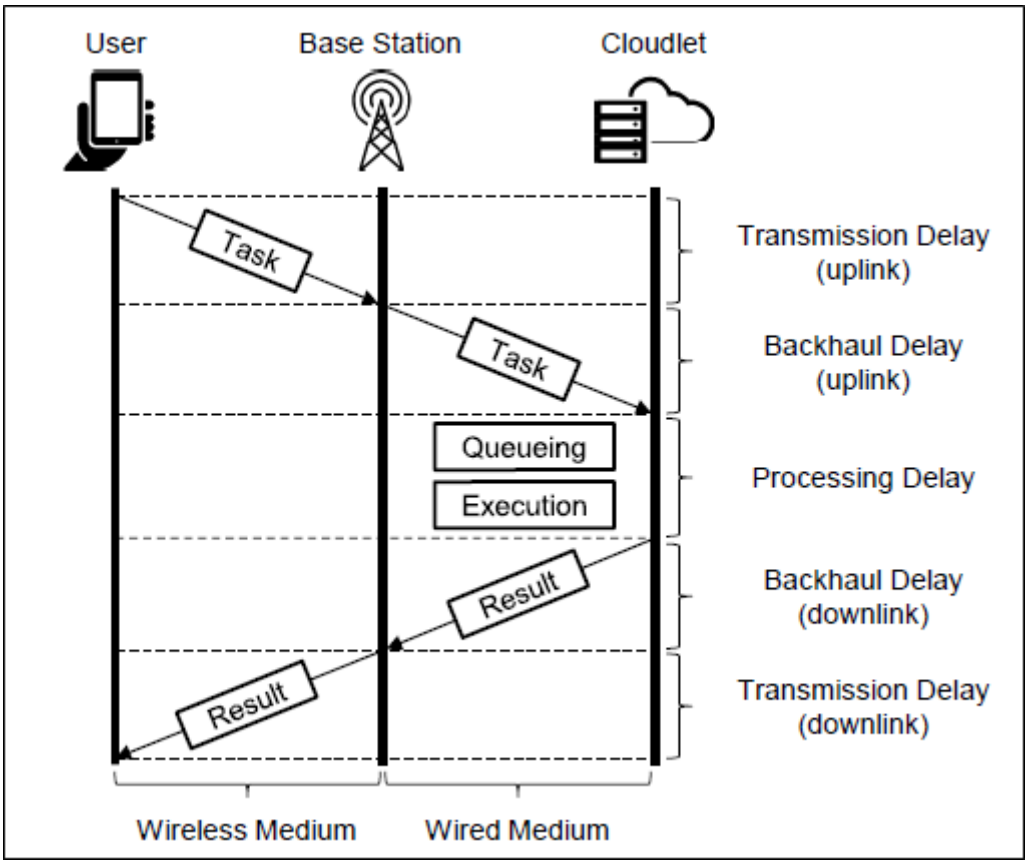
For this exploration work, the objective was to limit administration postponement and kMC was used to pick cloudlet areas that would prompt the most minimal fulfillment times for the clients. The two works use Machine Learning and hence give arrangements that can deal with high quantities of customers, servers, and solicitations. In any case, the help model considered by them is inadequate. They appropriately model thecalculation part of ECC, however, on the correspondence side, they overlook crash and dispute of assets, just thinking about a steady latency. Therefore, whenever applied to genuinesit- uations, their answers would not prompt ideal outcomes as significant angles which they disregarded would unavoidably emerge. Our service model will consider both computa- tion and communication resources and their allocation, which is essential to realistically represent ECC. [3]

# 3 Problem Statement

Our network model consists of base stations, users, and cloudlets. For a given number of cloudlets, we need to identify the locations for placing these cloudlets. Since our end goal is to optimize end-to-end service delay, we must decide how each user will be associated with a base station and cloudlet. Therefore, our problem would be to identify locations to place cloudlets and user associations to a base station such that minimum service delay can be achieved.

## 3.1 Problem formulation

We assume that the cloudlet is always co-located with the base station and vice-versa is not true. This assumption is made so that additional infrastructure costs can be re- duced. All users will have two connections associated with it: one connection is to the base station and the other one is to cloudlet. Each user will connect to a base station that offers the highest signal power. It is assumed that tasks are generated with time-triggered devices. Whenever a task is generated at a base station, it is first sent to the associated base station for computation. If that base station has a cloudlet associated with it, then the time to forward the task for computation is negligible. Otherwise, it is forwarded to a base station that has a cloudlet co-located with it through a wired backhaul link. The task will be processed at the cloudlet whenever a processor is idle, and the result is sent back through the same path. Therefore, total service delay is the time between the task generation and result arrival at the user end. This service delay can be modeled using transmission delay, backhaul delay, and processing delay as shown in figure below.[1]




**Figs1: Task request with emphasis on how the service delay is divided into transmission, backhaul, and processing delay[1]**

Given this assumed scenario, we have some sets of essential elements. Firstly, let's define V = $v_0, v_1, \ldots, v_V$ as the set of base stations, where $v_i = (x_{v_i}, y_{v_i}, v_i, k_{v_i})$ for 0 i V, $x_{v_i}$ and $y_{v_i}$ are the cardinal coordinates of base station $v_i$, $v_i$ is the transmission power level of base station $v_i$, and $k_{v_i}$ is a binary flag that indicates whether there is a cloudlet co- located with base station $v_i$. We will assume there are K cloudlets in total (by definition, K V). In the backhaul, we assume that all the base stations are connected using a min- imum spanning tree. There will be only one path between any two base stations. Thus,

all the base stations will be connected using backhaul links. E = $e_0$, $e_1$, ..., $e_E$, where $e_i$ = ($m_{e_i}$, $n_{e_i}$, $R_{e_i}$, $o_{e_i}$) for 0 ≤ i ≤ E, $m_{ei}$ ∈ V and $n_{ei}$ ∈ V are the base stations at the ends of link $e_i$, $R_{ei}$ is the total data rate of link $e_i$ on each direction (all links are duplex with same data rate on each direction), and $o_{ei}$ is the propagation delay associated with link $e_i$.

Additionally, we have the set of users U = $u_0$, $u_1$, ..., $u_U$, where $u_i$ = ($x_{u_i}$, $y_{u_i}$, $h_{u_i}$, $z_{u_i}$) for 0 ≤ i ≤ U, $x_{u_i}$ and $y_{u_i}$ are the cardinal coordinates of user $u_i$, $h_{u_i}$ ∈ V is the physical association of user $u_i$, and $z_{u_i}$ ∈ V is the base station co-located with the virtual associ- ation of user $u_i$. Obviously, if $z_{u_i}$ = $v_j$ for any 0 ≤ i ≤ U then $k_{v_j}$ = 1.

## 3.2 Evaluation metric

### 3.2.1 Processing Delay

This is the time for which the task should wait before it is processed at a cloudlet. It is assumed that c processors are present at any cloudlet and $G_{v_i}$ users are sending the tasks to cloudlet. Suppose, it tasks $\mu$ seconds on an average to execute a task received at the processor, then occupation rate of any processor can be calculated as below:

$$P_{v_i} = \frac{\lambda_{v_i}.\mu}{c} \tag{1}$$

This occupation rate is always maintained as less than 1 in the algorithm. This is because a value greater than 1 cumulatively increases the queuing time. Using this occupation rate, probability that no processors are idle whenever a task is received at a cloudlet can be predicted as below:

$$\psi_{v_i} = \frac{(c.p_{v_i})^c}{c!}.((1-p_{v_i}).\sum_{j=1}^{c}\frac{(c.p_i)^j}{j!} + \frac{(c.p_{v_i})^c}{c!}) \tag{2}$$

Therefore, processing delay for a task can be calculated as below using above equations.

$$P^{delay}_{v_i} = \frac{\psi_{v_i}.\mu}{(1-P_{v_i}).c} + \mu \tag{3}$$

### 3.2.2 Transmission Delay

Transmission delay is the total time needed to transmit the task and result in uplink and downlink, respectively. Therefore, transmission rate and twice of physical propagation between the user and associated base station are considered for modeling the transmission delay. Thus, the expected transmission delay for any user is given as below:[1]

$$T^{delay}_{u_i} = \frac{P^{up}}{D^{up}_{u_i}} + \frac{P^{down}}{D^{down}_{u_i}} + 2.\frac{d^{h_{ui}}_{u_i}}{\xi} \tag{4}$$

### 3.2.3 Backhaul Delay

Backhaul delay comes into the picture if the associated base station of any user does not have a cloudlet co-located with it. It is initially assumed that all base stations are

connected to each other through backhaul links using a minimum spanning tree. Since the backhaul links have a limited data rate, all users that use the same link will have to share the link. Therefore, backhaul delay is the propagation delay of the whole path and the data rate of the bottleneck link with the lowest data rate per user along with the delay in transmission of task and result.[1]

$$B^{u_i}_{delay} = \frac{P^{up} + P^{down}}{\min_{e_i s\zeta h_{u^i}, z_{u_i}} |L_{e_i}|} + 2. \sum_{e_i s\zeta h_{u^i}, z_{u_i}} O_e \quad (5)$$

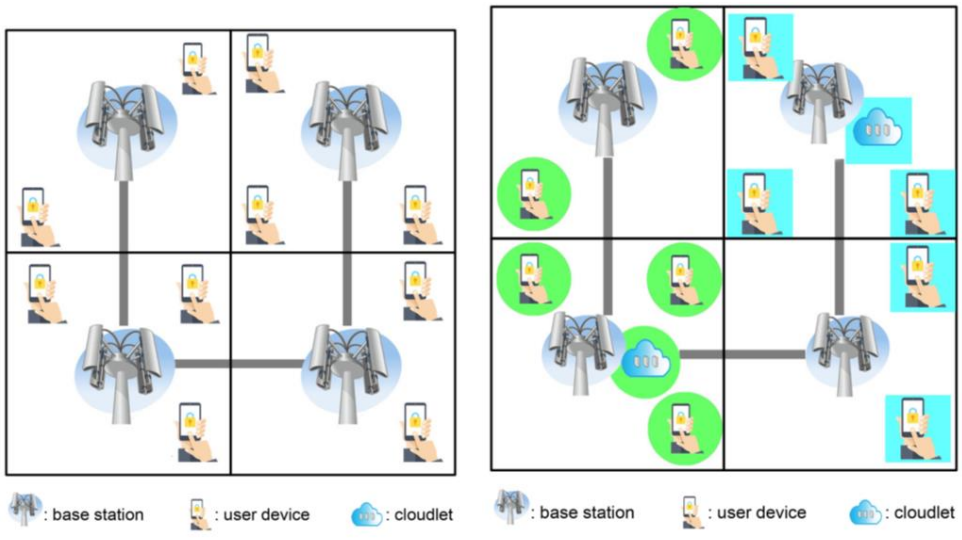
### 3.2.4 Service Delay

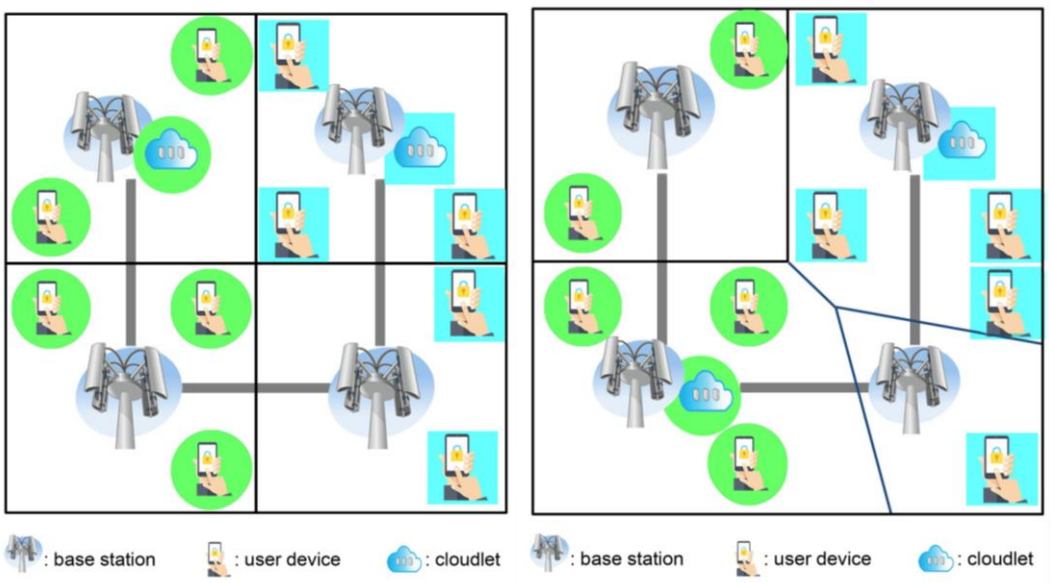
Service delay is the end-to-end delay experienced by the user from transmitting the task to receiving the result. Therefore, service delay can be modeled as the addition of processing, transmission, and backhaul delay and our objective function is to minimize the service delay as shown below:[1]

$$minS^{delay} = \sum_{i=0}^{i} \frac{P_{h_u}^{delay} + T_{u_i}^{delay} + B_{u_i}^{delay}}{V} \quad (6)$$

# 4 Proposed Algorithm

A cloudlet deployment policy for ECC based on the Machine Learning algorithms k- Means Clustering (kMC) and Particle Swarm Optimization (PSO) is proposed in this paper.






**Figs2:How the proposed algorithm works by assigning users to cloudlets with kMC and then using PSO to adjust the transmission power levels.[1]**

An outline of how the calculation functions can be found in figure. The figure first shows the area of the base stations and the users. At that point, kMC is used by first sending the cloudlets in arbitrary base stations. Users are assigned to the cloudlets based on the backhaul proliferation expected to contact them while simultaneously keeping the re- sponsibility adjusted between the servers. Users have a similar symbol (circle orsquare) as their comparing cloudlets. At that point, kMC moves the cloudlets to the base station nearest to the users in their clusters. This is appeared by moving the cloudlet with the circle symbol to an alternate base station. With virtual associations concluded, PSO is utilized to at last decide the transmission power levels and change some actual associa- tions. It tends to be found in the figure how this takes into account holding one client with the square symbol back from utilizing the backhaul connections to get to its cloudlet (which means zero backhaul delay for this client and more transmission capacity for the users that much utilize those connections). This likewise means we can decrease the transmission power level of the base right base station, which thus means less obstruction by and large in the framework. The two measures improve the general transmission and backhaul postponement of the users.[3]

## 4.1 K-Means

Cluster analysis was the main topic at the beginning of Edwin Diday's scientific career. With the consequence that the basic problems and methods of clustering became well- known in a broad scientific community, in statistics, data analysis, and in particular in applications. One of the major clustering approaches is based on the sum-of-squares cri- terion and on the algorithm that is today well-known under the name 'k-means'. When tracing back this algorithm to its origins, it has been proposed by several scientists in different forms and under different assumptions. Later on, many researchers investigated theoretical and algorithmic aspects and modifications of the method and by extending its domain to new data types and probabilistic models. Certainly, Diday's monograph

(Diday et al. 1979), written with 22 co-authors, marks a considerable level of generaliza- tion of the basic idea and established its usage for model-based clustering.[2]

kMC is a calculation for separating components into clusters dependent on distance. The absolute number of clusters is concluded heretofore to be K. The calculation begins by choosing an arbitrary area for the centroid of every single one of its K clusters. At that point, every component will be related to the centroid that is truly nearest to them. After this stage, for each cluster, the centroid is moved to the normal focus of all components of that cluster. At that point, all components are appointed new clusters dependent on these new centroids. This pattern of allotting components to clusters/moving the cen- troids in light of the components is rehashed until no centroid changes position.[1]

The general exhibition of the calculation is subject to the underlying irregular centroids. To wipe out this predisposition, this whole interaction is rehashed for multiple emphases, with distinctive introductory irregular areas for the centroids. For every cycle, the exhi- bition of the last clusters, as decided by a goal work, is recorded. The yield of kMCwill be the clusters from the cycle that brought about the best execution. For our concern, we will make a few variations to kMC. We will utilize kMC for choosing the virtual rela- tionship of the clients. In this way, we will utilize the calculation for choosing K clusters, one for each cloudlet. In that capacity, the area of the cluster centroids (for example cloudlets) will be restricted to the area of the base stations.[2]

We will assume that actual associations are as of now chose dependent on the best-advertised signal force at every client before the calculation begins and that kMC will just change the virtual associations. Thus, utilizing actual distance to figure out which cluster the clients will join isn't the most ideal decision. All things being equal, clients will cluster with cloudlets whose co-found base station has the most limited backhaul way as far as to spread to the actual relationship of the comparing client. Just if of ties, we will use the actual distance between the client and the cloudlet. Additionally, with no guarantees, the calculation is defenseless to over-burdening cloudlets by making clusters that are too enormous in situations where numerous clients are concentrated around a base station. This is awful because it prompts long handling delays. Subsequently, we will restrict the cluster measures so that all clusters have a similar number of clients. At long last, the emphasis on the best design of clusters will be resolved through Equation.[1]

**K-Means Pseudo Code:**

1: for all $v_i \in V$ $do_{v,}$ ← base transmission power level 2:

for $R^{kMC}$ iterations do

3: V ← V

4: for all cloudlets k do 5:

Choose random $v_i \in V$ 6:

Deploy k in $v_i$

7: Remove $v_i$ from V 8:

repeat

9: K ← all cloudlets 10:

for all $u_i \in Udo$

11: $z_{ui}$ ← k ∈K with min. propagation to $h_{ui}$

12: In case of ties, choose based on $d^k{}_{ui}$

13: if $z_{ui}$ has U/K users then remove $z_{ui}$ from K 14: V

← V

15: for all cloudlets k do

16: Move k to $v_i \in$V closest to its users 17:

Remove $v_i$ from V

18: until no cloudlet changes location

19: Execute PSO for setting transmission power levels 20:

return configuration with best performance

## 4.2 PSO (Particle Swarm Optimization)

Particle swarm optimization was introduced by Kennedy and Eberhart (1995). It has roots in the simulation of social behaviors using tools and ideas taken from computer graphics and social psychology research. Within the field of computer graphics, the first antecedents of particle swarm optimization can be traced back to the work of Reeves (1983), who proposed particle systems to model objects that are dynamic and cannot be easily represented by polygons or surfaces. Examples of such objects are fire, smoke, water, and clouds. In these systems, particles are independent of each other and their movements are governed by a set of rules. Some years later, Reynolds (1987) used a particle system to simulate the collective behavior of a flock of birds. In a similar kind of simulation, Heppner and Grenander (1990) included a roost that was attractive to the simulated birds. Both models inspired the set of rules that were later used in the original particle swarm optimization algorithm.[1]

PSO works by having multiple agents taking a gander at the space of all potential an- swers for the ideal setup. Each position in this space addresses a potential answer for the issue being handled by PSO. The agents, called particles, move around this space,

assessing each the arrangements relating to the positions they stop on before moving to another position. Their development is one-sided towards the best arrangement found so far universally, for example, they swarm around the best arrangement found by the arrangement, everything being equal. To try not to stall out in neighborhood ideal focuses, the development is likewise one-sided towards what is called nearby best, which is the best arrangement found by that specific element and one-sided towards the course and speed of their previous development, which is known as the inactivitypart.[1]

These three components (global best, local best, and dormancy) have loads related to them to direct how a lot they impact the arrangement search. Besides, the swarming conduct around the worldwide/nearby best arrangements is normally likewise molded by irregular factors that can take any esteem somewhere in the range of 0 and 1 that ad- justment in every emphasis. The objective of these components is to adjust the inquiry between misuse (the swarm around the best arrangements found) and investigation (the irregular quest for new components to try not to stall out in nearby optima). In our concern, we need to track down the ideal arrangement of transmission power levels for every base station. Accordingly, our answers are tuples of V components, one transmis- sion power level for each base station. We will assume that virtual associations are now characterized before the calculation begins, so PSO can just change actual associations. The target work that will direct what the worldwide/nearby best are will be and this proposed calculation can discover the arrangement that prompts the least postponement for our clients.[1]

**PSO Pseudo Code:**

1: for all particles P do set $q_P$ with random V values 2: for

all particles P do set $V_P$ with random V values 3: for all

particles P do $B^l \leftarrow q_P$

4: $B^g \leftarrow \arg min_{particles P} f(B^l)_P$

5: for $R^{PSO}$ iterations do 6:

for all particles P do

7: if $f(q_p) < B^l_P$ then $B^l_P \leftarrow q_p$

8: if $f(q_p) < B^g$ then $B^g \leftarrow q_p$

9: $F^l \leftarrow$ random number between 0 and 1 10:

$F^g \leftarrow$ random number between 0 and 1

11: $V_P \leftarrow W^i \cdot V_P + F^l \cdot W^l \cdot (q_P B^l) + F^g \cdot W^g \cdot (q_P - B^g)$

12: $q_P \leftarrow q_P + V_P$

13: return $B^g$

# 5 Empirical Comparison

Using K-means clustering and particle swarm optimization, end-to-end service delay is minimized when compared to random placement of servers (conventional method consid- ered as a benchmark for most of the literature). To check the reliability, simulation is performed for 50 iterations in each scenario. In all scenarios, locations of users and base stations are randomly chosen and backhaul links are considered as edges of a minimum spanning tree. K-means clustering is used in deciding backhaul propagation and PSO is used in deciding the transmission power levels of each base station and physical associa- tions with base stations.

In this segment, we will introduce results to help our case that our proposition is a legit- imate answer for the ECC sending issue. Our goal is to demonstrate that our calculation brings considerable execution enhancements when contrasted with the irregular situation of servers. The outcomes that appeared here are gotten through recreation. To accom- plish measurable unwavering quality, the outcomes shown are normal of 1000 diverse arbitrary simulations, with every recreation having particular areas randomly decided for base stations and users. The backhaul joins are chosen based on distance and, as referenced previously, structure a passive optical network. The propagation distance is dictated by the distance between the base stations, while the speed is taken as the speed of light. Also, the simulations use as parameters the qualities that appeared in the Table 1 except if unequivocally expressed something else. These qualities follow assessments of what ought normal in 5G IoT networks. We accept values are something similar in the uplink and the downlink.

The parameters for the AI algorithms were gotten from experimentation focused on what turns out best for this particular issue. Various situations will be assessed to show how our proposition has the best exhibition. For correlation, we will use another arrangement strategy ordinarily found in the writing, where cloudlets are randomly sent, autonomously of the area of users, in any accessible base station. Virtual affiliations are chosen based on backhaul propagation. PSO is used to choose transmission power levels and actual affiliations. Furthermore, cloudlets too will serve U/K users each (for example adjusted responsibility among cloudlets) in the irregular approach. These last two focuses imply that the distinctions in execution come carefully from the arrangement choices. For the main chart, we will shift the number of base stations. The outcomes can be seen in Figs:3 As expected, more base stations lead to bring down service delay for all arrangements. This is sensible as adding base stations implies a lower communication responsibility for each base station which thus brings down transmission delay. Besides, situations with few base stations regularly mean those base stations will require a high transmission power level to arrive at separated users, which prompts high transmission delay because of longer propagation and helpless sign for those users and higher impedance across the entire framework. Then again, more base stations mean less confined users. The arbi- trary arrangement strategy has the most noticeably awful exhibition, as it has no real

Table 1: Parameters used in the simulation

| | |
|---|---|
| User average task rate | 1 task/s |
| Average task execution time | 50ms |
| Number of processors per cloudlet | 16 |
| Path loss floating intercept | 75.85 dB |
| Average path loss exponent | 3.73 |
| Total area size | 10000 m2 |
| Number of base stations | 10 |
| Number of cloudlets | 3 |
| Number of users | 10000 |
| Base station antenna gain | 24.5 dBi |
| Rayleigh fading coefficient | -1.59175 |
| User transmission power | 27 dBm |
| User antenna gain | 2.15 dBi |
| Noise density | 4 · 10-19W/Hz |
| Wireless channel bandwidth | 1 THz |
| Packet size | 128 KB |
| Data rate per backhaul link | 10 Gb/s |
| Number of kMC iterations | 6 |
| Number of PSO iterations | 6 |
| Number of PSO particles | 7 |
| PSO inertia bias | 2 |
| PSO local best bias | 18.5 |
| PSO global best bias | 2.5 |

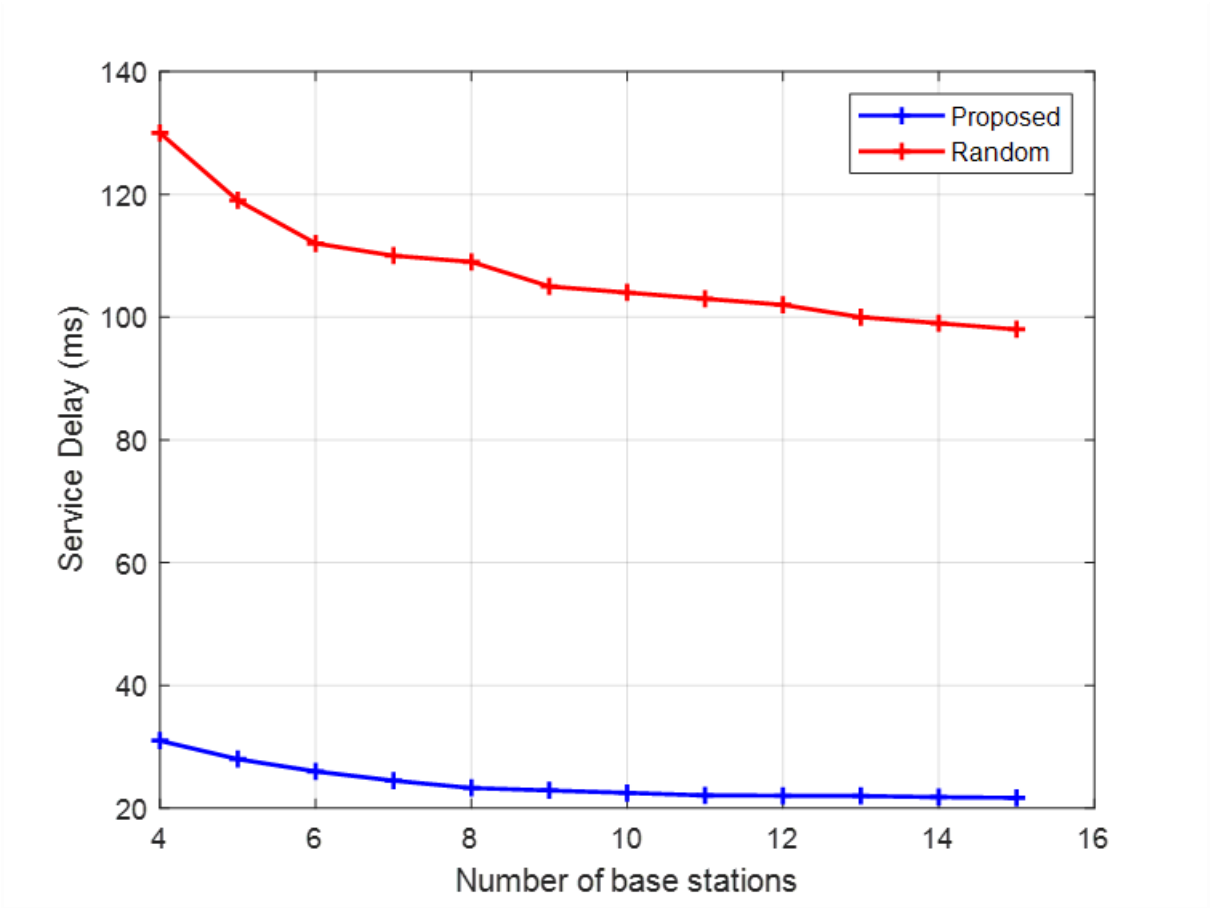
way to position cloudlets in the best base stations.

Our kMC-based proposition really picks the best areas to convey the cloudlets what's more, that is the place where the pertinent contrast in execution comes from. This distinction can be noticed whether or not numerous or few base stations are in the frame- work. Our proposition is likewise prepared to do proficiently utilizing the resources in situations where there are not many base stations to such an extent that adding new base stations gets huge upgrades. Then, we will investigate the approaches as the quantity of servers increment. Results have appeared in figs:4. Fascinating conduct can be seen where adding servers after 5 for our proposition prompts the presentation getting more terrible, in spite of the framework having more resources. Adding cloudlets improves the service by bringing down the processing responsibility per cloudlet, which prompts lower line times for the processors as there are fewer users per cloudlet. In any case, in the event that you have enough cloudlets, this stand-by time is low to the point that it gets superfluous.

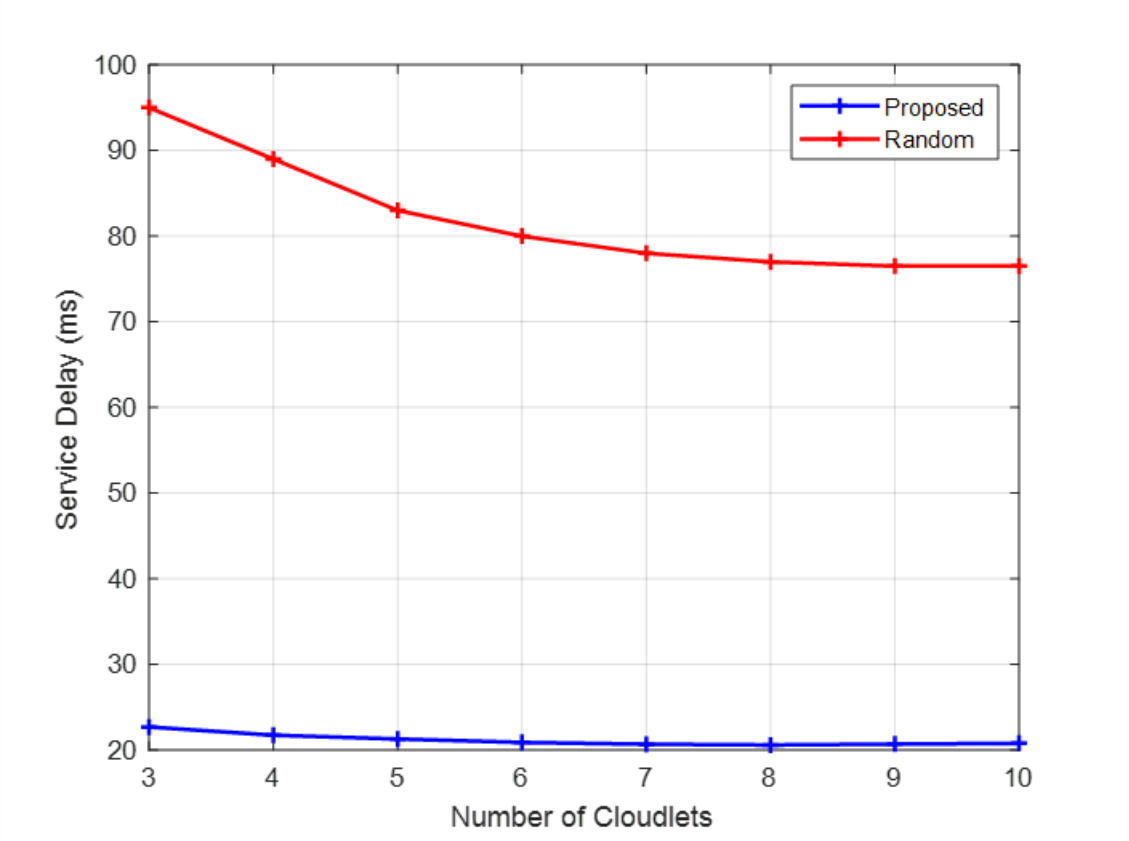
Moreover, adding cloudlets past that demolishes the service since servers are compelled to all have U/K users. Given this, more cloudlets mean a higher possibility that a client will be compelled to interface with a worker that is farther away since the nearest worker is at the pre-decided cutoff as of now, for example, adding more cloudlets after line time is as of

now low doesn't improve processing time and truth be told, exacerbates transmission. In this situation, there are insufficient cloudlets for the irregular arrangement strategy to test this conduct. The diagram additionally shows how, once more, the arbitrary approach is more terrible for not choosing where to send the cloudlets. The proposition is better paying little mind to the number of cloudlets are in the situation. At last, a correlation is made while the quantity of users is shifted. The outcomes are introduced in Figs:5. True to form, more users bring about more terrible service delays in general. This is effortlessly clarified by how users are having a similar asset pool, both in communication and calculation. In this way, more users mean fewer resources per client and therefore longer service delays. This is most exceedingly terrible in the arbitrary approach, as there is no sending decision to offset the higher responsibility. kMC can situate the servers concurring to the client areas, which brings about a lower increment to the service delay, as found in the diagram.

## 5.1 Results



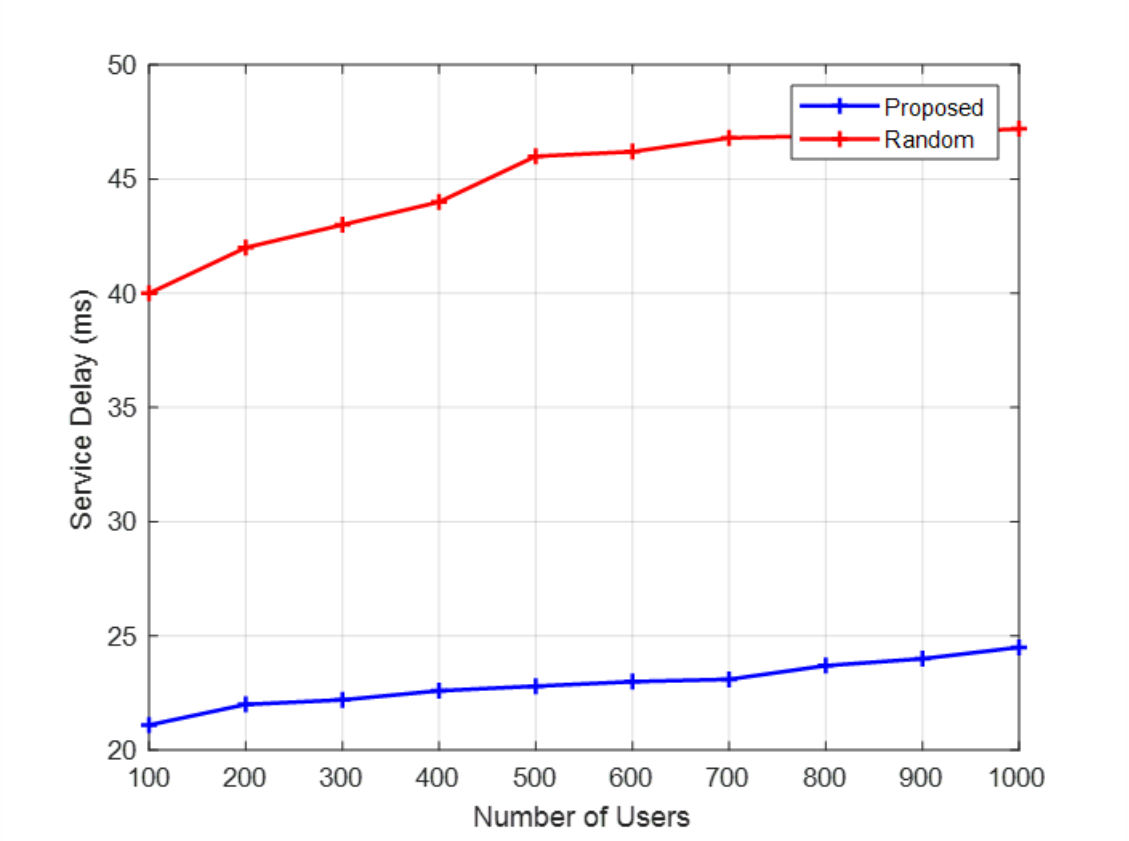

Figs3: ECC service delay variation as the number of base stations is increased.




Figs4: ECC service delay variation as the number of cloudlets is increased.

In scenario 1, the number of base stations is varied, and several cloudlets and users are considered as fixed, and the results are plotted in Figs:3. We can observe that the service delay is reduced as the number of base stations increases. This is expected as increasing base stations will reduce the communication workload on each base station and thus re- ducing the transmission delay. Random deployment scenario has the worst performance in all cases as our proposed model is significantly using the resources for optimal place- ment and correcting the positions in case of undesired results.

In scenario 2, demonstrated in the figs:4, the number of cloudlets is varied, and the num- ber of base stations and users remains fixed. We can observe that delay decreases as the number of cloudlets increase and there is not much change after the number of cloudlets are 5. By adding more cloudlets, the workload per cloudlet gets reduced, and queuing time for tasks at cloudlet is also reduced. But after sufficient cloudlets are present, wait time becomes very low and is not affected by processing time after a threshold. The proposed algorithm is better than random deployment regardless of how many cloudlets.




Figs5: ECC service delay variation as the number of users is increased

In scenario 3, demonstrated in the figs:5, the number of users is varied, and the number of base stations and cloudlets remains fixed. As the number of users increases, fewer resources for processing and computation will be available per user resulting in longer delays. This becomes worse in random deployment as there are no measures considered to improve the workload.

# 6 Conclusion

ECC is a vital innovation for giving resources to users. It takes into account those re- sources to be utilized economically on-request and divided between various customers. In addition, uniquely in contrast to customary cloud computing, the dormancy among users and workers is lower so even constant applications can utilize the cloud resources. Accordingly, ECC is a vital innovation. Since in ECC, workers are sent to the edge rather than a solitary center spot, the area of the organization matters and influences the sub- sequent service quality. Notwithstanding this, most ECC research ignores sending and assumes an irregular arrangement strategy.

In this paper, we worked on a sending strategy for 5G IoT conditions that use AI algo- rithms PSO and KMC. The proposition takes thought of processing, transmission, and backhaul communication to viably bring down service delay. Our proposition has a low intricacy and execution time and can be applied to situations with numerous factors. This is an outcome of the machine learning algorithms and it is fundamental for using ECC with IoT and 6G that accompanies huge measures of users, workers, and solici-

tations. Additionally, the reproductions showed that the proposition is fundamentally better compared to an irregular organization. This ought to demonstrate that sending choice is significant for improving service delay in ECC and that considering transmission, processing, and backhaul communication are generally applicable components to consider when planning such an answer. The service is appropriately more slow if sending isn't done as expected and our proposition is equipped for offering a quicker service by wisely choosing where to put each cloudlet as per client prerequisites.

# 7 Future Work

Our future work includes placement of edge servers can be further studied considering individual delays with distributed and centralized units. We would like to compare the results with other suitable unsupervised machine learning algorithm to find find better deployment settings.

# References

[1]Tiago Koketsu Rodrigues,Katsuya Suto,Nei Kato,2019. Edge Cloud Server Deploy- ment with Transmission Power Control through Machine Learning for 6G Internet of Things. DOI:10.1109/TETC.2019.2963091

[2]Bo Li, Keyue Wang, Duan Xue, Yijian Pei,2018. K-Means Based Edge Server Deployment Algorithm for Edge Computing Environments. DOI: 10.1109/Smart-World.2018.00203

[3]Najmul Hassan, Kok-Lim Alvin Yau, Celimuge Wu, 2019. Edge Computing in 5G: A Review. DOI: 10.1109/ACCESS.2019.2938534