**Technical Report**

**Cloudlet Simulator**

|  |  |
| --- | --- |
| Prepared by: | Eng.Mohammad Sakka |
|  | mhdsqq2016noor@gmail.com  to: Dr.Yazan Aljeiroudy |
| Date submitted: | 3/1/2022 |
|  |  |

**Project objectives**

* This project is to develop an optimization algorithm that optimizes the cloudlets deployment in the wireless metropolean networks (later)
* The first step we have done is only building a WMAN simulator

**Methodology**

* First we draw the class diagram of the simulator and it is available

In the “doc” folder with name “WMAN simulator.pdf”

Note: throw the developing, the class diagram was modified, but the diagram sketch has not modified yet. And it is to modify later.

**Parameters**

1. Environment Dimensions: (100 km x 100 km)
2. Access points number: 50
3. Initial range of users number: 1 🡪 100
4. Cloudlets number: 6
5. Number of servers per cloudlet: 5
6. Access point communication data rate: 3 \* 10 ^8 [bit/sec]
7. Wireless Data rate: 1 [Kb/sec]
8. Internet speed: 2 [MB/sec]
9. Wireless delay per meter: 0.1 [sec/meter]
10. Size of task’s data pucket :

The average is random number in range [200 K, 500 K] [bit]

And the size is generated for each task using exponential distrobution // from paper 2

1. Number of code lines in the task:

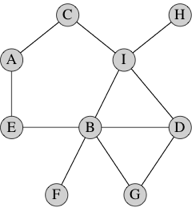
The average is random number in range [1000,2000] [line]

And the number is generated for each task using exponential distribution // similar to paper2 but in paper2 they names it “computation demand” and the unit in that paper is [cycles/bit]

1. Maximum workload of cloudlet: 45
2. Time unit: 1 [sec]
3. Tasks sending rate: normal random with avg=2.5, sigma=0.5
4. User transmission power = 0.1 [W] // from paper 2
5. Server effective switched capacity = 5 \* 10^-27 // from paper 2
6. Server service rate (computing capacity) = 2^30 [line/sec]

**Routing method:**

* We have a graph with access points as nodes and connections between the access points as edges, like the following.



* For example, if we want to move from A node to G.
* We need to find the shortest path
* We use dijkstra algorithm for that purpose, so that we used it to send the puckets from access point to the closest cloudlet, remember that the cloudlet is located in one of the access points.
* The distance between each two access points were set to ‘1’ because we ignore the distance delay considering the links are fiber optics , so the shortest path should only containe the minimum number of hops regardeless the distance between the start and the target access point.

**Cloudlets Deployment:**

Random

**Users Movement:**

Random Walk

**Users Adding:**

Each time unit , one user will enter the area from one of the four sides

**Users Deleting:**

Each time unit, the system will check for the users that exceeds the borders. Then delete them if it detect them

**Assigne the users to access points:**

each user is assigned to the closest access point, and the assigned access point is updated every time unit

**tasks sending:**

the users send the tasks according to poisson process with lamda = sending rate

**tasks servicing:**

the tasks sent to the closest cloudlet then set in the queue of the cloudlet, if the queue is full, then the task will sent to cloud. the queue length is “lamdMax”.

**Task Assigning to the server**:

The arrived task will be assigned to the idle server, if there more than one idle server, the system will choose one server randomly

**Tasks departure:**

after staying the task for the executin period on the server, we consider that the task is executed and releas it, then mark the server as idle to allow a new task to execute on it.

**Performance Measures:**

* We have calculated several measures , as following:

1. Locally served tasks ratio
2. Remotely served taks ratio
3. System respone time = average wireless delay + average routing delay + average queuing time + average locally execution time + average remotely execution time
4. Energy Consumption = Execution Energy + Wireless communication Energy //from paper2

* **Delay Measures Formulas:**
* Wireless\_delay = Wireless\_delay\_per\_meter \* user\_to\_AP\_distance \* (1/wireless\_datarate)
* Routing\_delay = (1/ Access point communication data rate) \* datapucketSize\* numberOfHops
* queingTime = moment of task assigning to server – moment of generationg the task
* locally\_execution\_time = task number of lines / server service rate
* remotely execution time = task data size / internet speed + task number of lines / cloud service rate
* **Energy Consumption Formulas:**

- wirelss Energy = wirelessDelay \* user\_transmission\_power \* task\_data\_pucket\_size

- execution Energy = Server effective switched capacity \* (server service rate^2 ) \* task number of lines //from paper2

- totalEnergy = wirelss Energy + execution Energy

**Cost Function:**

* f1 = number of cloudlets
* f2:

the energy consumption for executing a task of a user ui locally is given as:

Where:

The energy consumption required for transmitting a task of a user ui to its associated AP is:

Where:

The total energy consumption of a user ui is given as



Where

Zi,j: the probability of offloading a task of user ui to cloudlet cj



f2

- f3:

the average waiting time composed of the

queue and execution time for executing a task of ui locally is



The average waiting time consisting of the queue and transmission

time for transmitting an offloaded task of ui to its

associated AP is

Where: L: the number of cloudlets

NumberOfSentToCloudTasksForUser\_i =

if NumberOfSentToCloudTasksForUser<0

NumberOfSentToCloudTasksForUser = 0

NumberOfNotSentToCloudTasksForUser\_i =

the average waiting

time comprising of the queue and execution time for executing

a task in a cloudlet ci is

If NumberOfSentToCloudTasksForUser>0

Where: numberOfServers: the number of servers per cloudlet

Else:

Tasks Sending to cloud delay:

timeOfTransmissionToCloud = (numberOfSentToCloudTasks \* mean()) / B

where B: internet speed

Tasks execution on cloud delay:

timeOfExecutionOnCloud = (numberOfSentToCloudTasks \* mean()) /

where

Tcloud = timeOfExecutionOnCloud + timeOfTransmissionToCloud

Total delay:

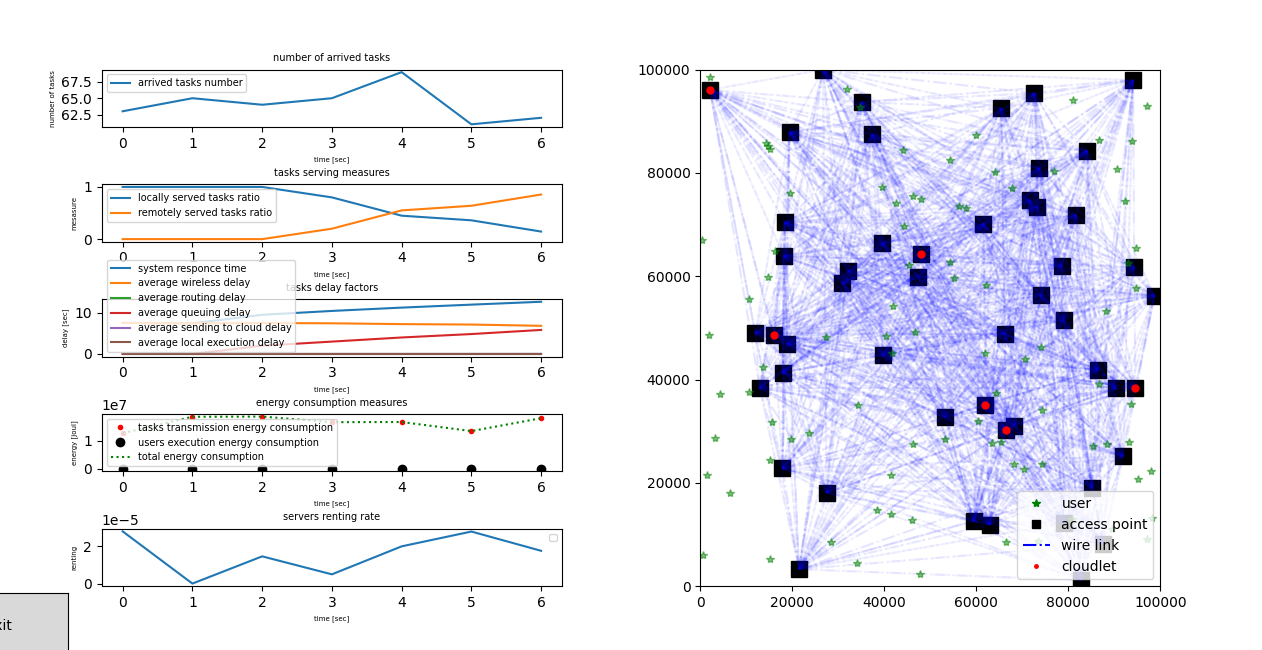
)

T = mean(

* f4 = servers renting

renting = rentingRate \*

**GUI**



Eng. Mohammad Sakka