Modelling and Simulation of Edge Computing Environments

Çağatay Sönmez

04.11.2022

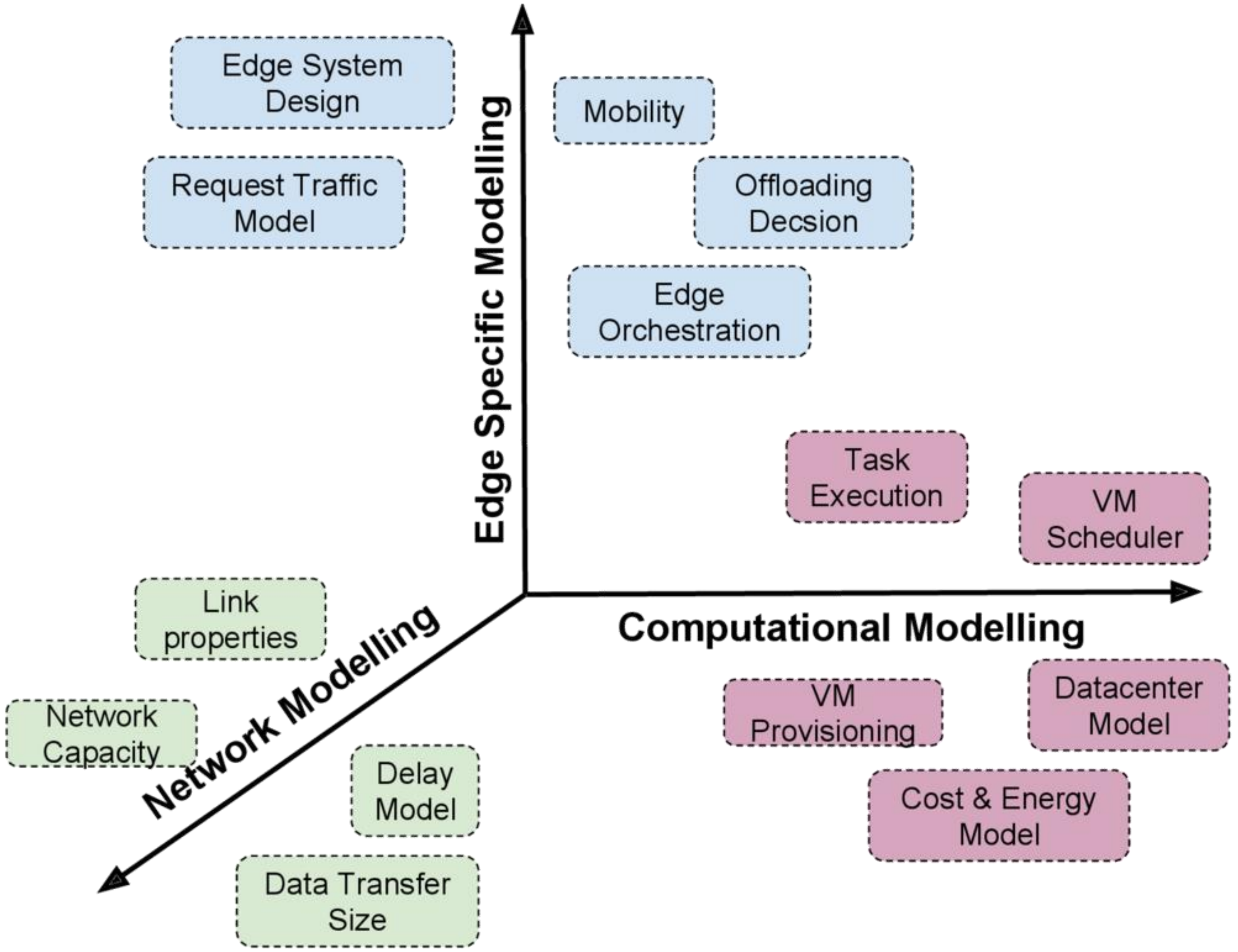
EdgeCloudSim

C. Sonmez, A. Ozgovde and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of edge computing systems," *Transactions on Emerging Telecommunications Technologies*, Vol. 29, No. 11, p. e3493, 2018

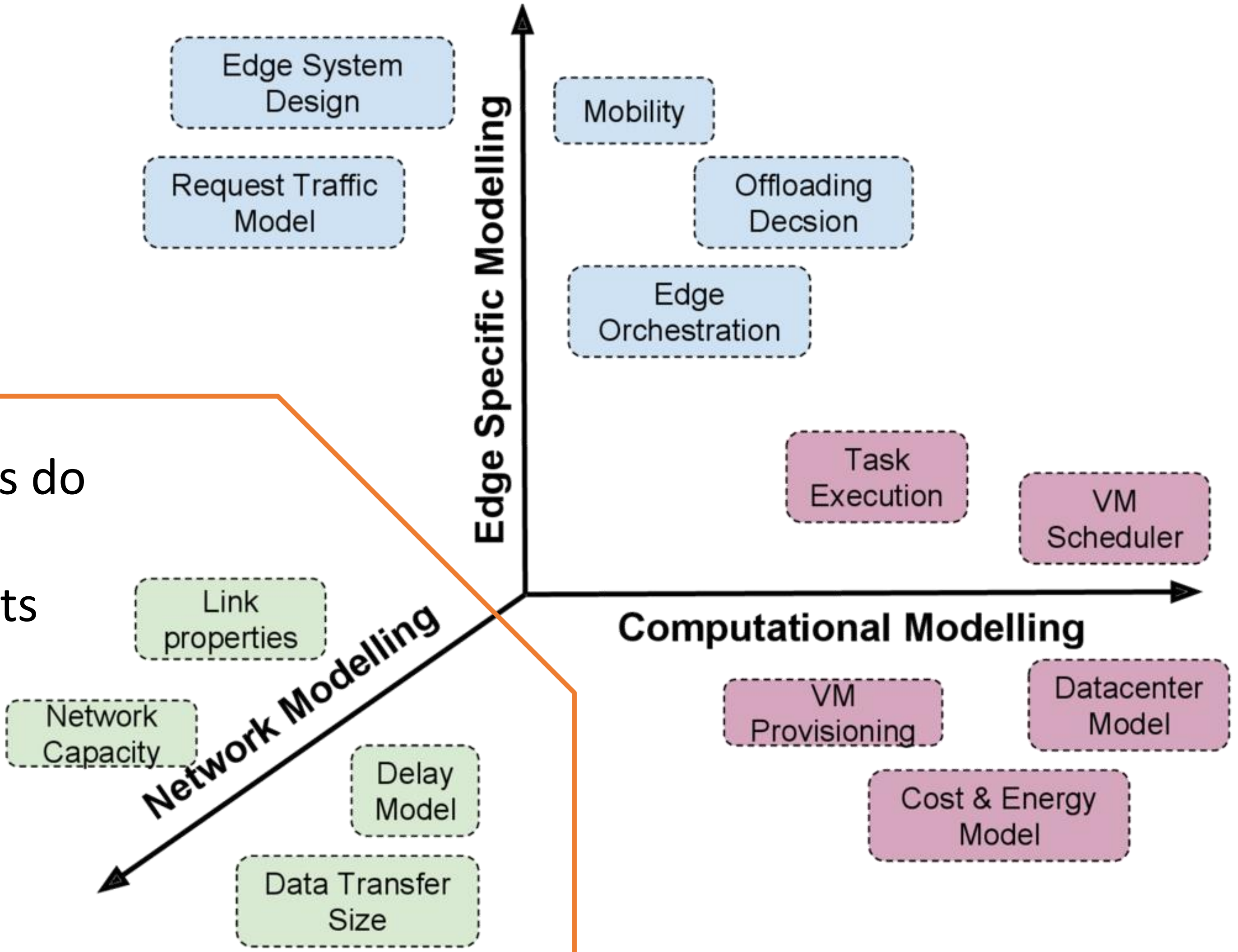
What is EdgeCloudSim

- EdgeCloudSim is a new simulator
- Provides a simulation environment specific to edge computing scenarios
- EdgeCloudSim is based on CloudSim but adds some additional functionalities
- Extensible and easy-to-use
- Publicly available on GitHub
  - https://github.com/CagataySonmez/EdgeCloudSim
- Has high reputation; as of October 2022
  - 396 citations based on Google Scholar
  - A discussion forum with 171 active members
  - More than 15K views on YouTube channel

Motivation of Developing a Simulator

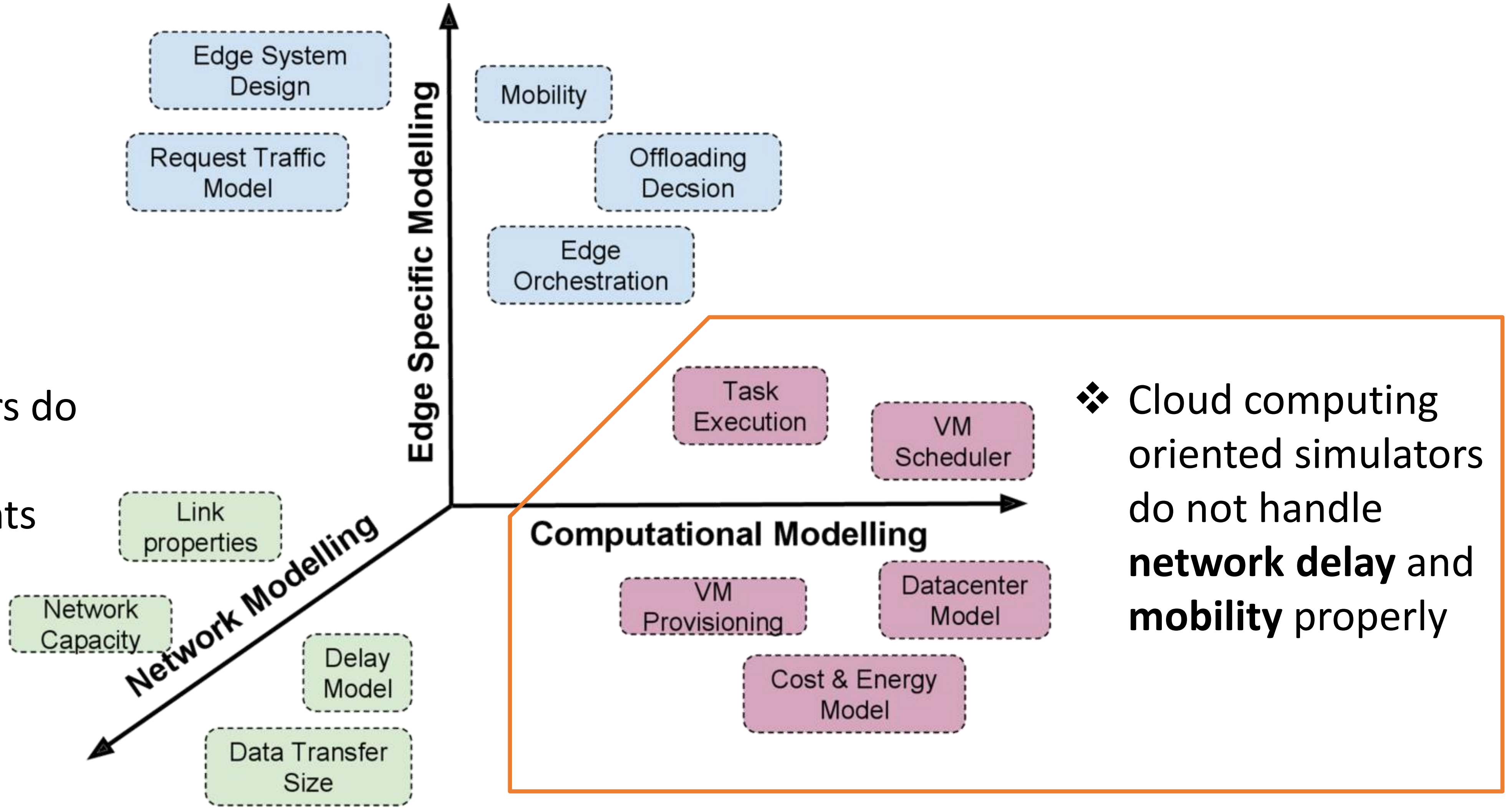


Motivation of Developing a Simulator



❖ Network simulators do not consider cloud computing elements
- Datacenters
- Hosts
- VMs
- Brokers

Motivation of Developing a Simulator

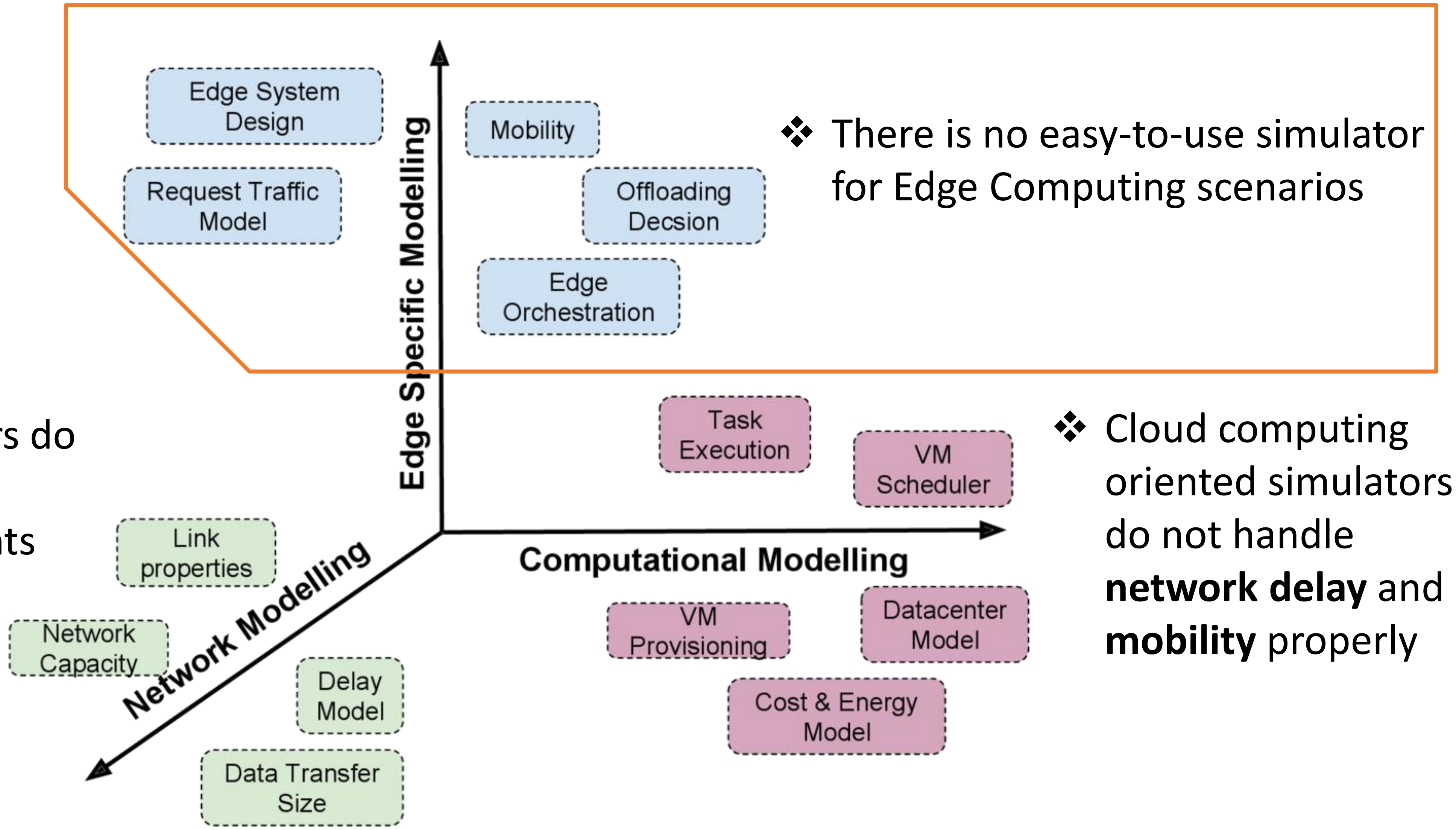


❖ Network simulators do not consider cloud computing elements

- Datacenters
- Hosts
- VMs
- Brokers

❖ Cloud computing oriented simulators do not handle **network delay** and **mobility** properly

Motivation of Developing a Simulator



❖ There is no easy-to-use simulator for Edge Computing scenarios

❖ Network simulators do not consider cloud computing elements
- Datacenters
- Hosts
- VMs
- Brokers

❖ Cloud computing oriented simulators do not handle **network delay** and **mobility** properly

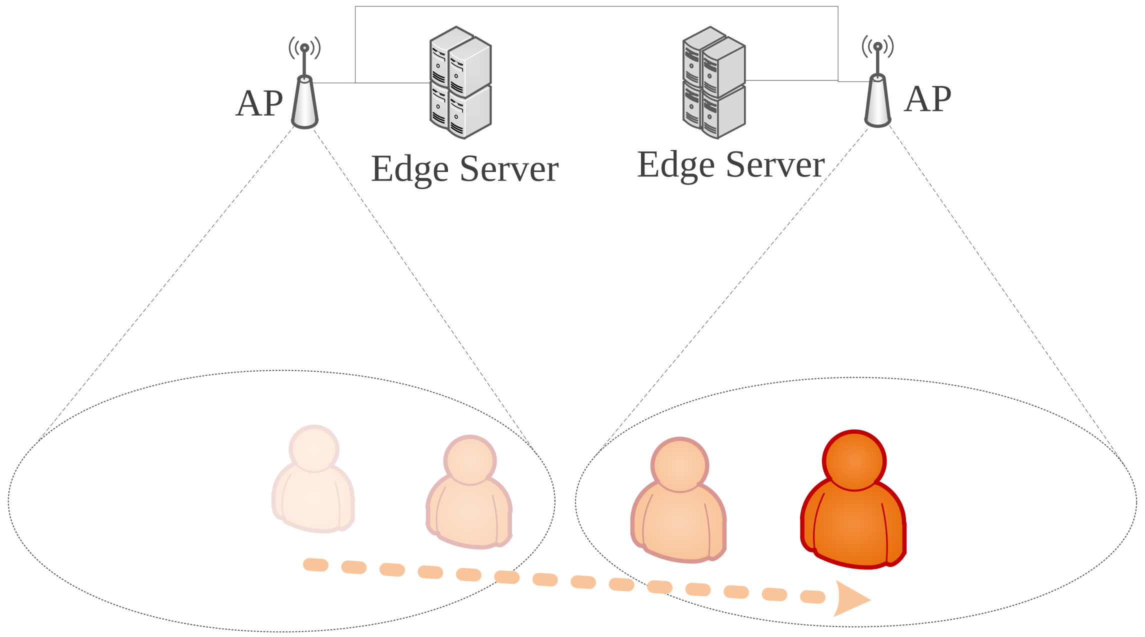
EdgeCloudSim Core Modules

- Mobility Module
- Networking Module
- Load Generator Module
- Edge Orchestrator Module

EdgeCloudSim Core Modules cont.

Mobility Module

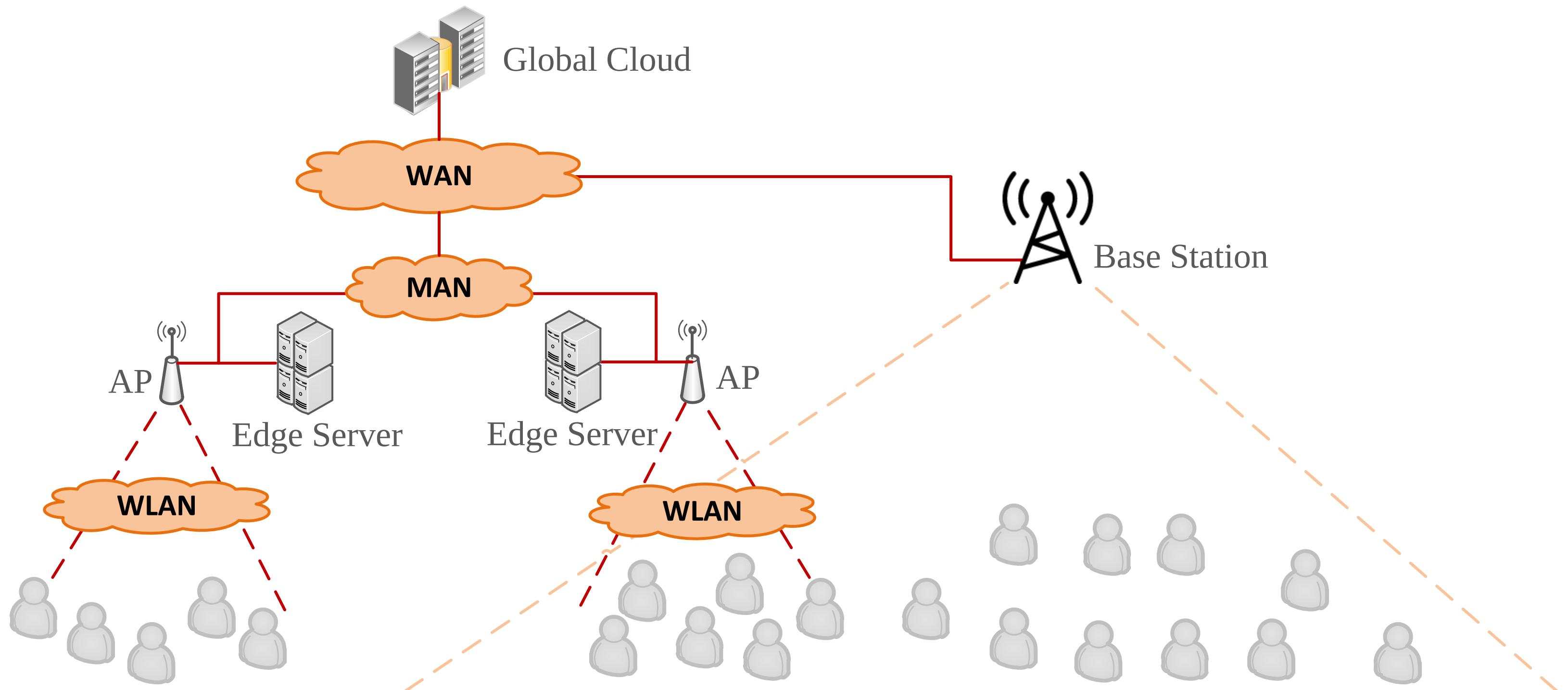
- Manages the location of edge devices and clients



EdgeCloudSim Core Modules cont.

Networking Module

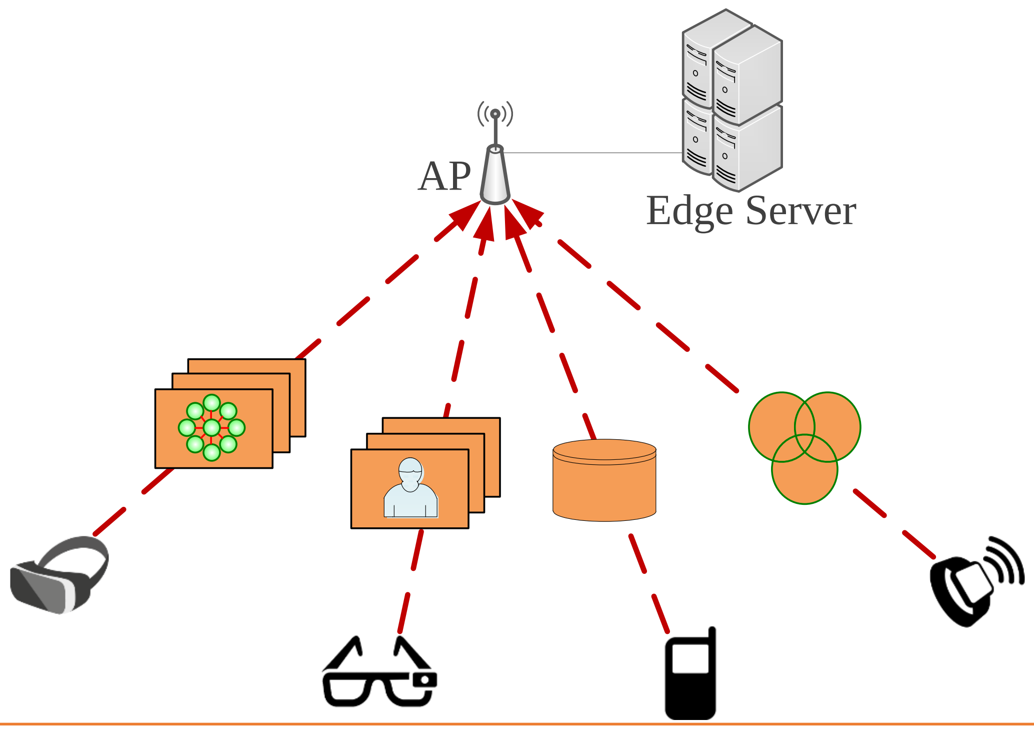
- Adds link delays between the network components



EdgeCloudSim Core Modules cont.

Load Generator Module

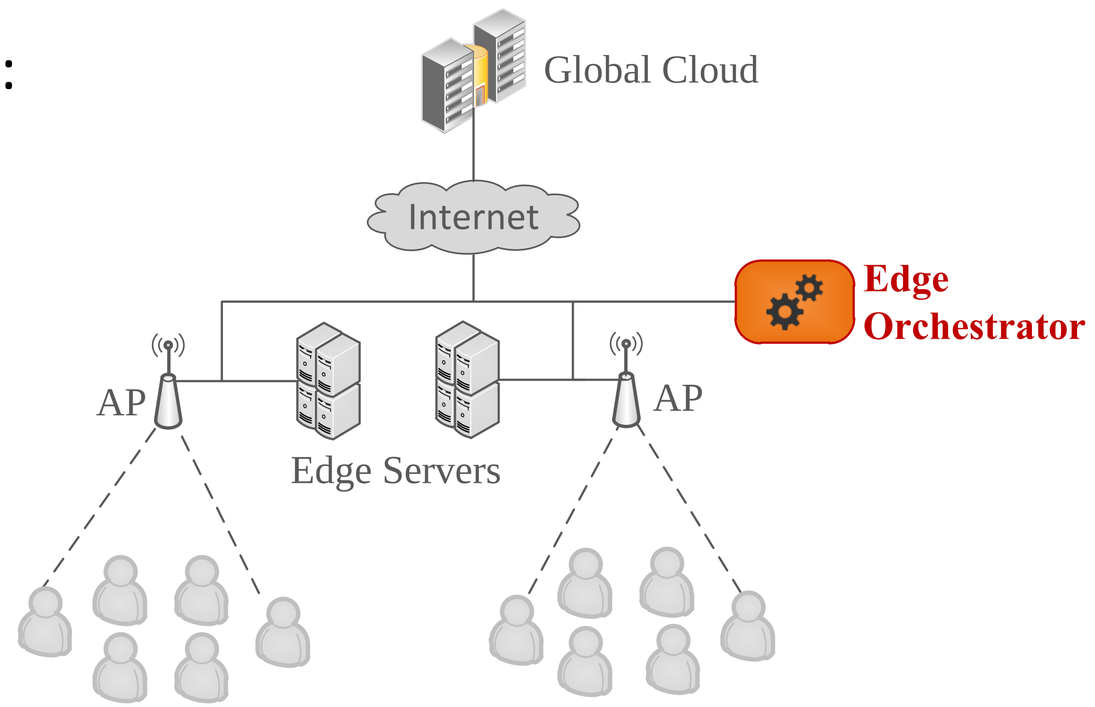
- Generates tasks based on the simulated scenario



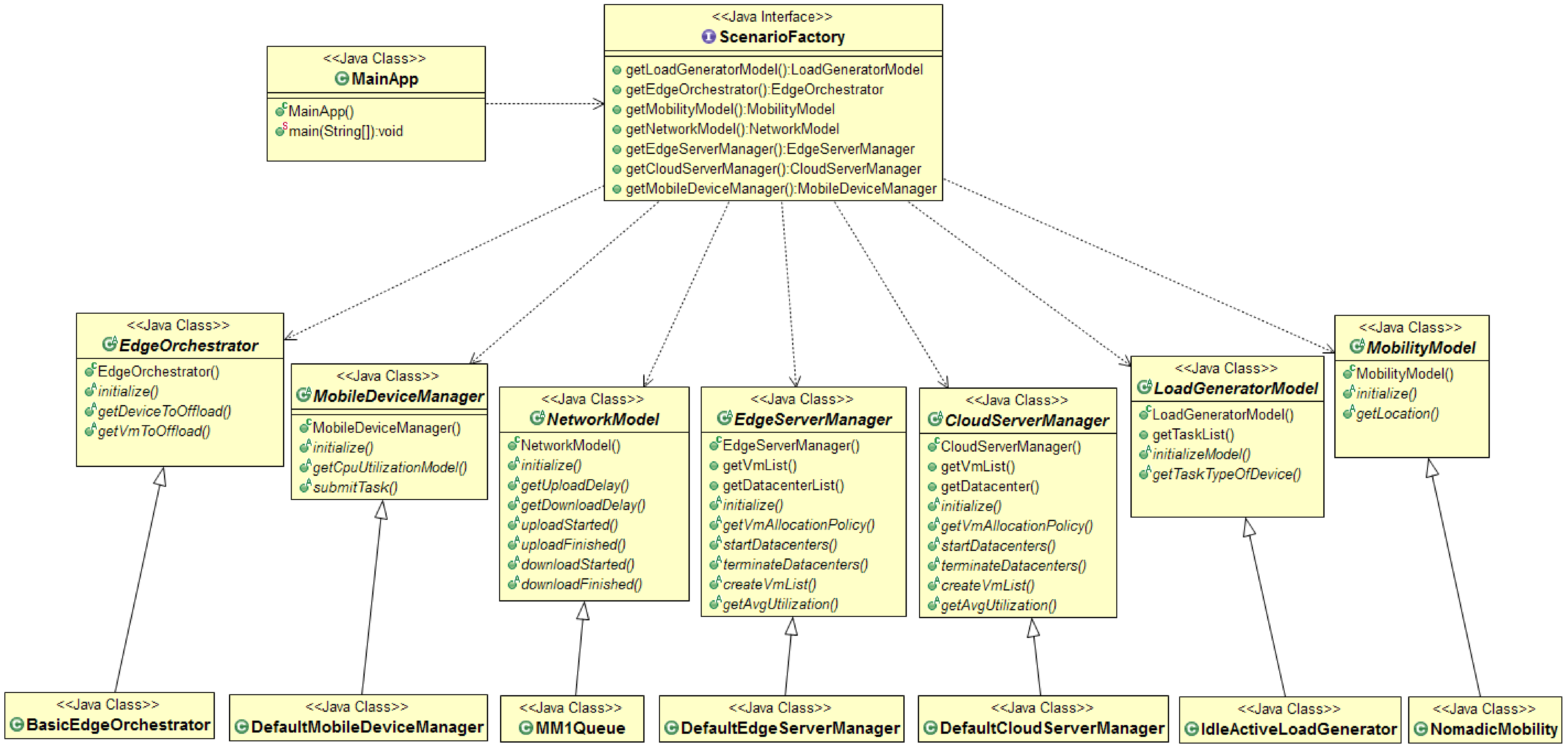
EdgeCloudSim Core Modules cont.

Edge Orchestrator Module

- The edge orchestrator can be considered as the central nervous system
- It makes critical decisions, such as:
  - Resource provisioning
  - Scales up/down the servers
  - Generates/terminates VMs
  - Migrates tasks
  - Coordinates services



Extensibility



Sample Factory Class

```java
public class VehicularScenarioFactory implements ScenarioFactory {
    ...
    @Override
    public LoadGeneratorModel getLoadGeneratorModel() {
        return new VehicularLoadGenerator(numOfMobileDevice, simulationTime, simScenario);
    }
    @Override
    public EdgeOrchestrator getEdgeOrchestrator() {
        return new VehicularEdgeOrchestrator(numOfMobileDevice, orchestratorPolicy, simScenario);
    }
    @Override
    public MobilityModel getMobilityModel() {
        return new VehicularMobilityModel(numOfMobileDevice,simulationTime);
    }
    @Override
    public NetworkModel getNetworkModel() {
        return new VehicularNetworkModel(numOfMobileDevice, simScenario, orchestratorPolicy);
    }
    @Override
    public EdgeServerManager getEdgeServerManager() {
        return new VehicularEdgeServerManager();
    }
    @Override
    public CloudServerManager getCloudServerManager() {
        return new DefaultCloudServerManager();
    }
    @Override
    public MobileDeviceManager getMobileDeviceManager() throws Exception {
        return new VehicularMobileDeviceManager();
    }
    @Override
    public MobileServerManager getMobileServerManager() {
        return new VehicularMobileServerManager(numOfMobileDevice);
    }
```

Extending Mobility Model...

```
public class VehicularMobilityModel extends MobilityModel {
        private final double SPEED_FOR_PLACES[] = {20, 40, 60}; //km per hour

        private int lengthOfSegment;
        private double totalTimeForLoop; //seconds
        private int[] locationTypes;

        //prepare following arrays to decrease computation on getLocation() function
        //NOTE: if the number of clients is high, keeping following values in RAM
        //      may be expensive. In that case sacrifice computational resources!
        private int[] initialLocationIndexArray;
        private int[] initialPositionArray; //in meters unit
        private double[] timeToDriveLocationArray;//in seconds unit
        private double[] timeToReachNextLocationArray; //in seconds unit

        public VehicularMobilityModel(int _numberOfMobileDevices, double _simulationTime) {
                super(_numberOfMobileDevices, _simulationTime);
        }

        @Override
        public void initialize() {
                //Find total length of the road
                Document doc = SimSettings.getInstance().getEdgeDevicesDocument();
                NodeList datacenterList = doc.getElementsByTagName("datacenter");
        ...
```

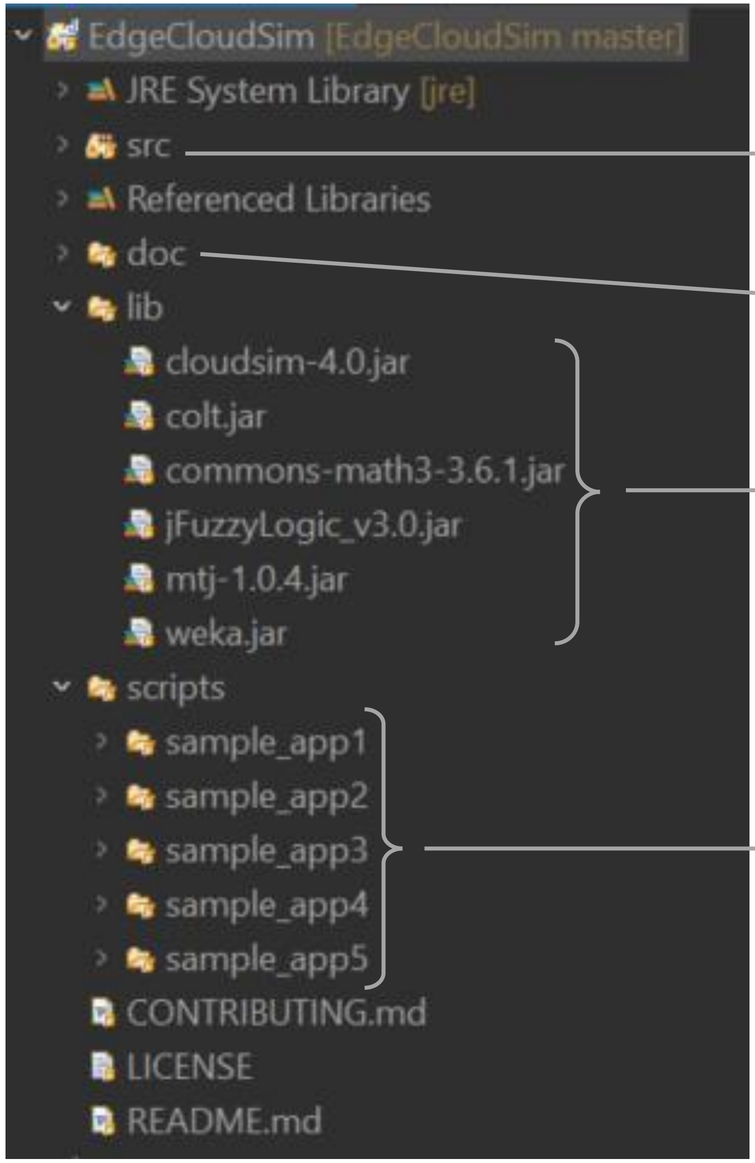
Extending Network Model…

```java
public class VehicularNetworkModel extends NetworkModel {
        public static double maxWlanDelay = 0;
        public static double maxWanDelay = 0;
        public static double maxGsmDelay = 0;

        private class MMPPWrapper {
                private double currentPoissonMean;
                private double currentTaskSize;

                //record last values used for successful packet transmission
                private double lastPoissonMean;
                private double lastTaskSize;

                //record last n task statistics during MM1_QUEUE_MODEL_UPDATE
                private double numOfTasks;
                private double totalTaskSize;

                public MMPPWrapper() {
                        currentPoissonMean = 0;
                        currentTaskSize = 0;

                        lastPoissonMean = 0;
                        lastTaskSize = 0;

                        numOfTasks = 0;
                        totalTaskSize = 0;
                }
          ...
```

Downloading EdgeCloudSim

- EdgeCloudSim is publicly available in GitHub
- You should clone EdgeCloudSim repository to start development
    - `$git clone https://github.com/CagataySonmez/EdgeCloudSim.git`
- After clonning the repository, you can use your favorite IDE such as Eclipse and NetBeans
- Please check EdgeCloudSim wiki page for more
    - Using command line
        - https://github.com/CagataySonmez/EdgeCloudSim/wiki/How-to-compile-EdgeCloudSim-application
    - Using Eclipse IDE
        - https://github.com/CagataySonmez/EdgeCloudSim/wiki/EdgeCloudSim-in-Eclipse:-step-by-step-installation-&-running-sample-application
    - Using Netbeans IDE
        - https://github.com/CagataySonmez/EdgeCloudSim/wiki/EdgeCloudSim-in-NetBeans:-step-by-step-installation-&-running-sample-application

EdgeCloudSim Folder Hierarchy



```
EdgeCloudSim [EdgeCloudSim master]
  JRE System Library [jre]
  src
  Referenced Libraries
  doc
  lib
    cloudsim-4.0.jar
    colt.jar
    commons-math3-3.6.1.jar
    jFuzzyLogic_v3.0.jar
    mtj-1.0.4.jar
    weka.jar
  scripts
    sample_app1
    sample_app2
    sample_app3
    sample_app4
    sample_app5
  CONTRIBUTING.md
  LICENSE
  README.md
```

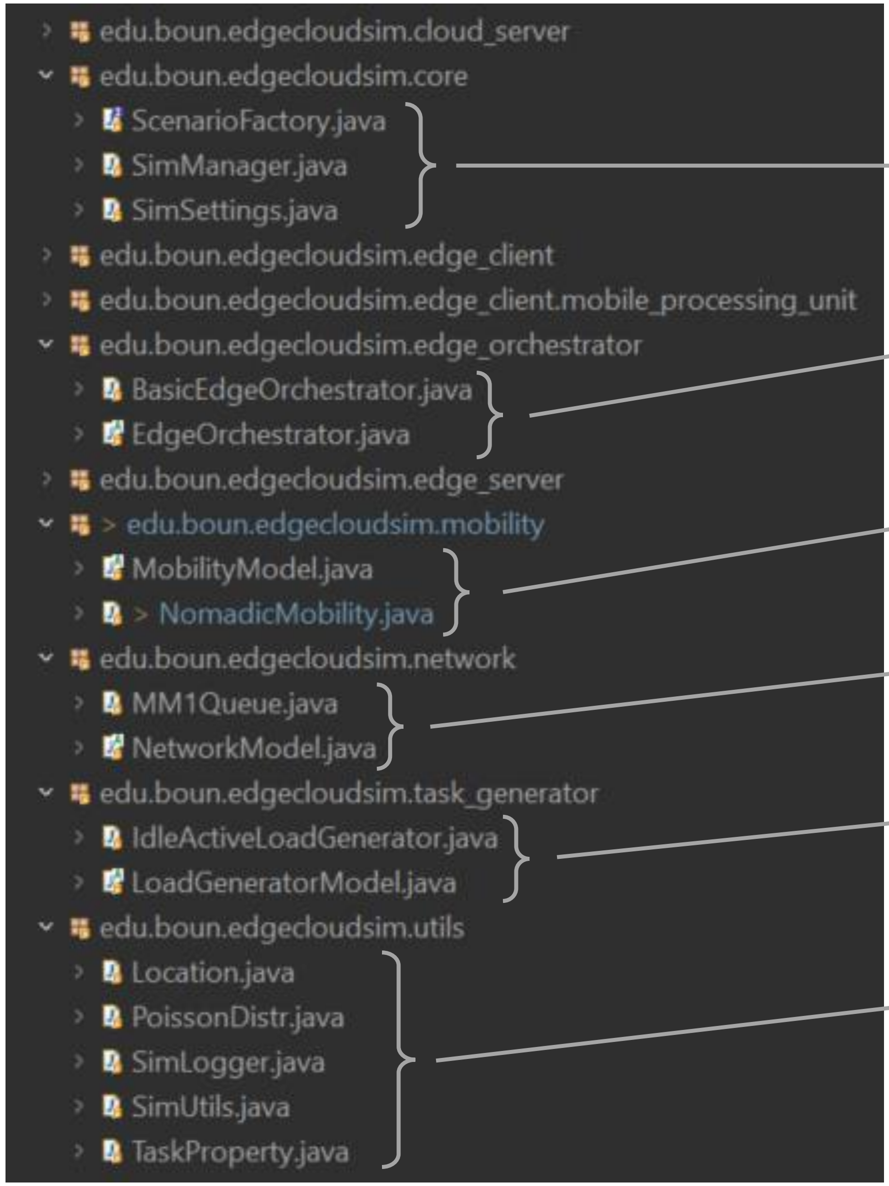
Configuration files are mostly provided via command line arguments (especially when running through terminal).

Configuration files are mostly provided via command

Sometimes developers can use static file path for convenience (especially when running the simulations through IDEs during development).

SimSettings is initialized with the configuration files

EdgeCloudSim Core Classes



```
> edu.boun.edgecloudsim.cloud_server
v edu.boun.edgecloudsim.core
    > ScenarioFactory.java
    > SimManager.java
    > SimSettings.java
> edu.boun.edgecloudsim.edge_client
> edu.boun.edgecloudsim.edge_client.mobile_processing_unit
v edu.boun.edgecloudsim.edge_orchestrator
    > BasicEdgeOrchestrator.java
    > EdgeOrchestrator.java
> edu.boun.edgecloudsim.edge_server
v > edu.boun.edgecloudsim.mobility
    > MobilityModel.java
    > > NomadicMobility.java
v edu.boun.edgecloudsim.network
    > MM1Queue.java
    > NetworkModel.java
v edu.boun.edgecloudsim.task_generator
    > IdleActiveLoadGenerator.java
    > LoadGeneratorModel.java
v edu.boun.edgecloudsim.utils
    > Location.java
    > PoissonDistr.java
    > SimLogger.java
    > SimUtils.java
    > TaskProperty.java
```

core package of EdgeCloudSim and related classes

Abstract edge orchestrator class and its default implementation

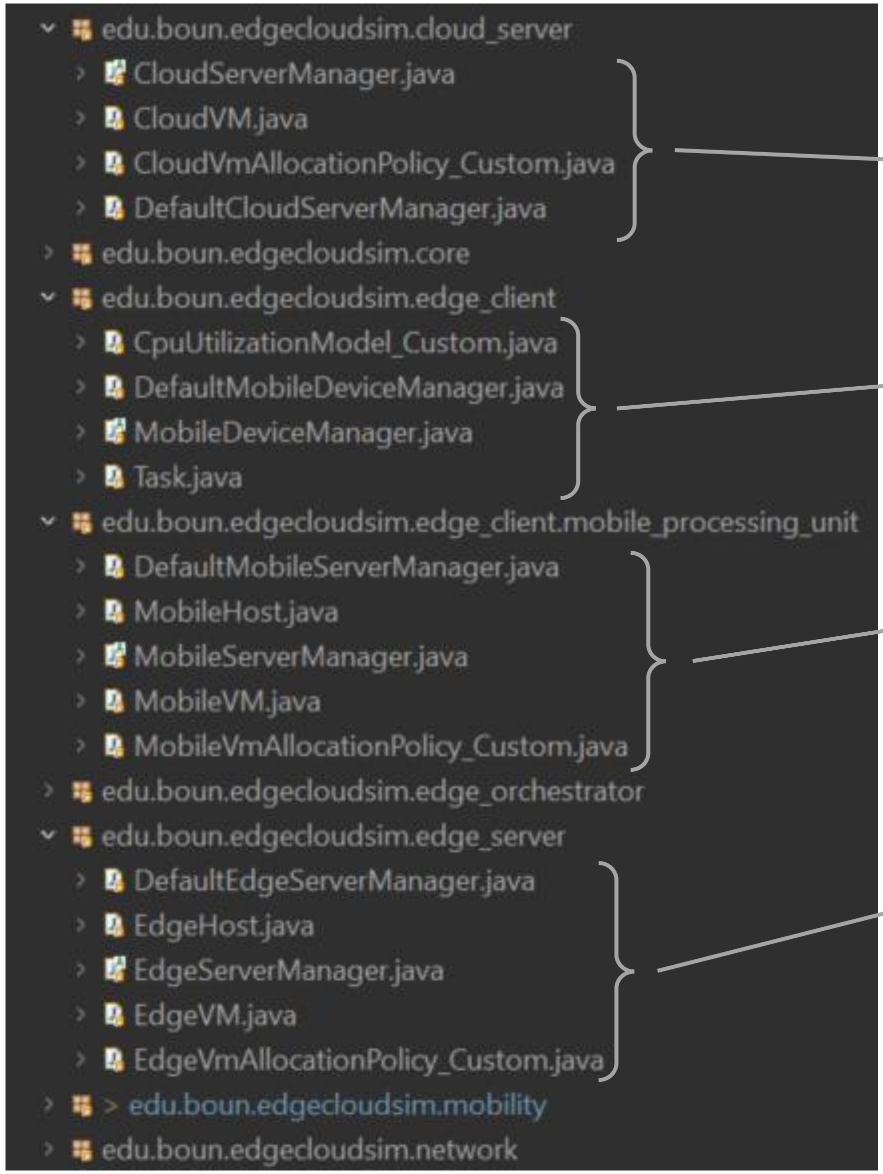
Abstract mobility model class and its default implementation

Abstract network model class and its default implementation

Abstract load generator class and its default implementation

util package of EdgeCloudSim and related classes

EdgeCloudSim Computational Classes



```
edu.boun.edgecloudsim.cloud_server
    CloudServerManager.java
    CloudVM.java
    CloudVmAllocationPolicy_Custom.java
    DefaultCloudServerManager.java
edu.boun.edgecloudsim.core
edu.boun.edgecloudsim.edge_client
    CpuUtilizationModel_Custom.java
    DefaultMobileDeviceManager.java
    MobileDeviceManager.java
    Task.java
edu.boun.edgecloudsim.edge_client.mobile_processing_unit
    DefaultMobileServerManager.java
    MobileHost.java
    MobileServerManager.java
    MobileVM.java
    MobileVmAllocationPolicy_Custom.java
edu.boun.edgecloudsim.edge_orchestrator
edu.boun.edgecloudsim.edge_server
    DefaultEdgeServerManager.java
    EdgeHost.java
    EdgeServerManager.java
    EdgeVM.java
    EdgeVmAllocationPolicy_Custom.java
> edu.boun.edgecloudsim.mobility
edu.boun.edgecloudsim.network
```

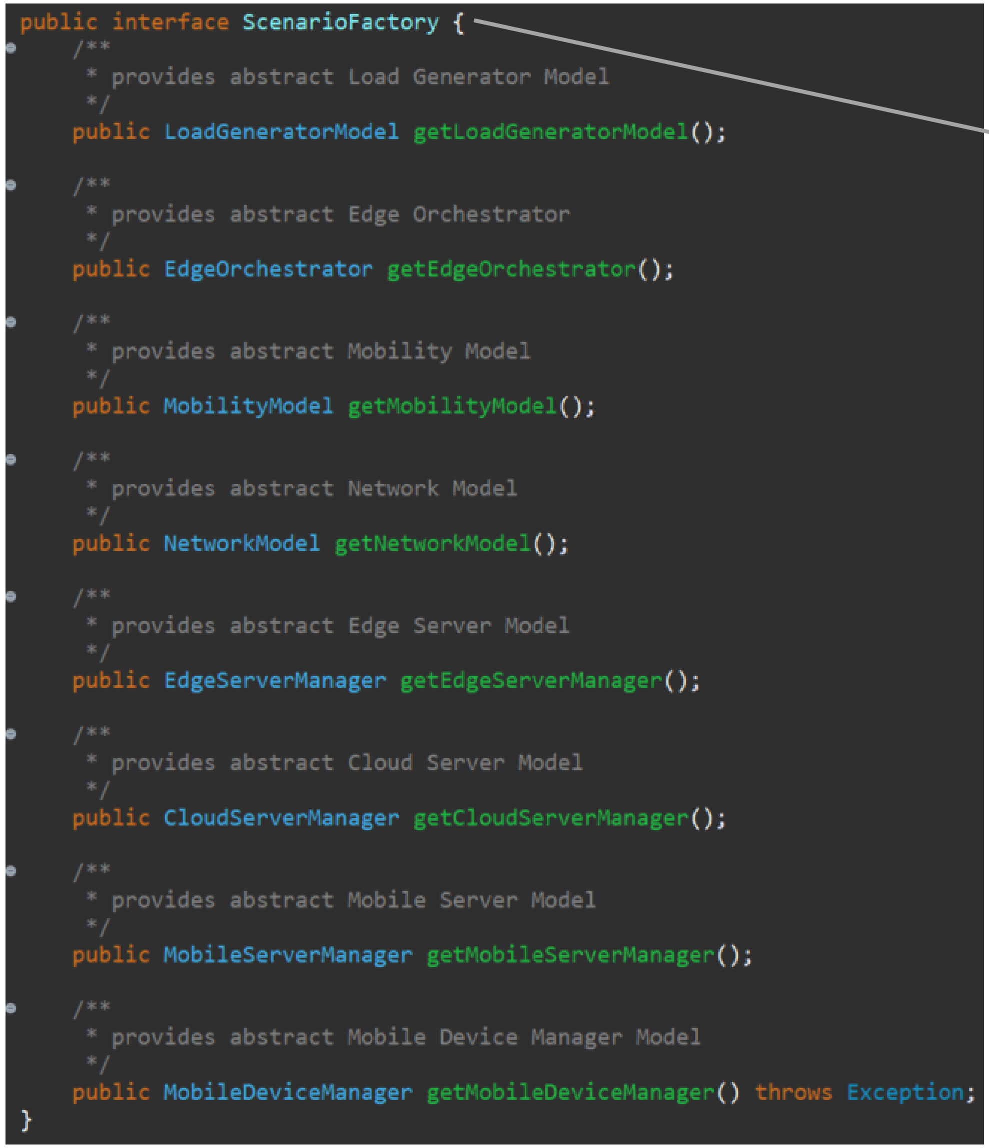
cloud_server package includes classes for the cloud datacenter and associated hosts and VMs

edge_client package includes classes that represents a broker (user)

mobile_processing_unit package includes classes for the mobile datacenter and associated hosts and VMs

edge_server package includes classes for the edge datacenter and associated hosts and VMs

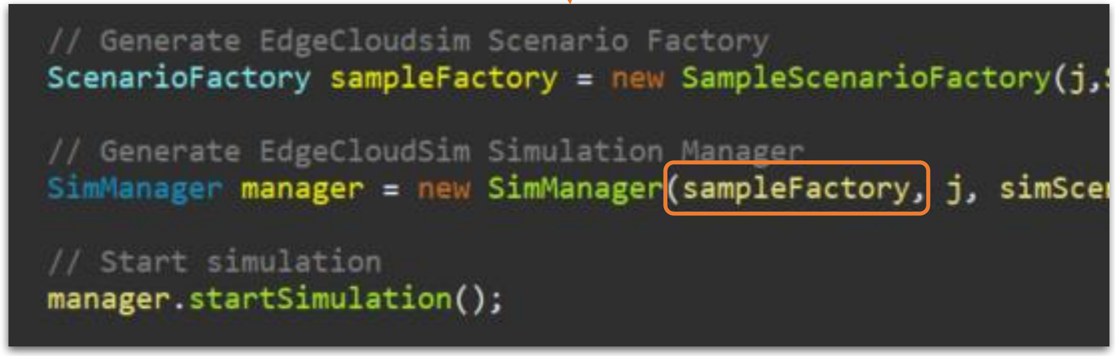
Important Classes: ScenarioFactory



```java
public interface ScenarioFactory {
    /**
     * provides abstract Load Generator Model
     */
    public LoadGeneratorModel getLoadGeneratorModel();

    /**
     * provides abstract Edge Orchestrator
     */
    public EdgeOrchestrator getEdgeOrchestrator();

    /**
     * provides abstract Mobility Model
     */
    public MobilityModel getMobilityModel();

    /**
     * provides abstract Network Model
     */
    public NetworkModel getNetworkModel();

    /**
     * provides abstract Edge Server Model
     */
    public EdgeServerManager getEdgeServerManager();

    /**
     * provides abstract Cloud Server Model
     */
    public CloudServerManager getCloudServerManager();

    /**
     * provides abstract Mobile Server Model
     */
    public MobileServerManager getMobileServerManager();

    /**
     * provides abstract Mobile Device Manager Model
     */
    public MobileDeviceManager getMobileDeviceManager() throws Exception;
}
```

The ScenarioFactory class provides extensibility. You can provide versions of most of the important classes with different behaviors to implement your simulation scenario.

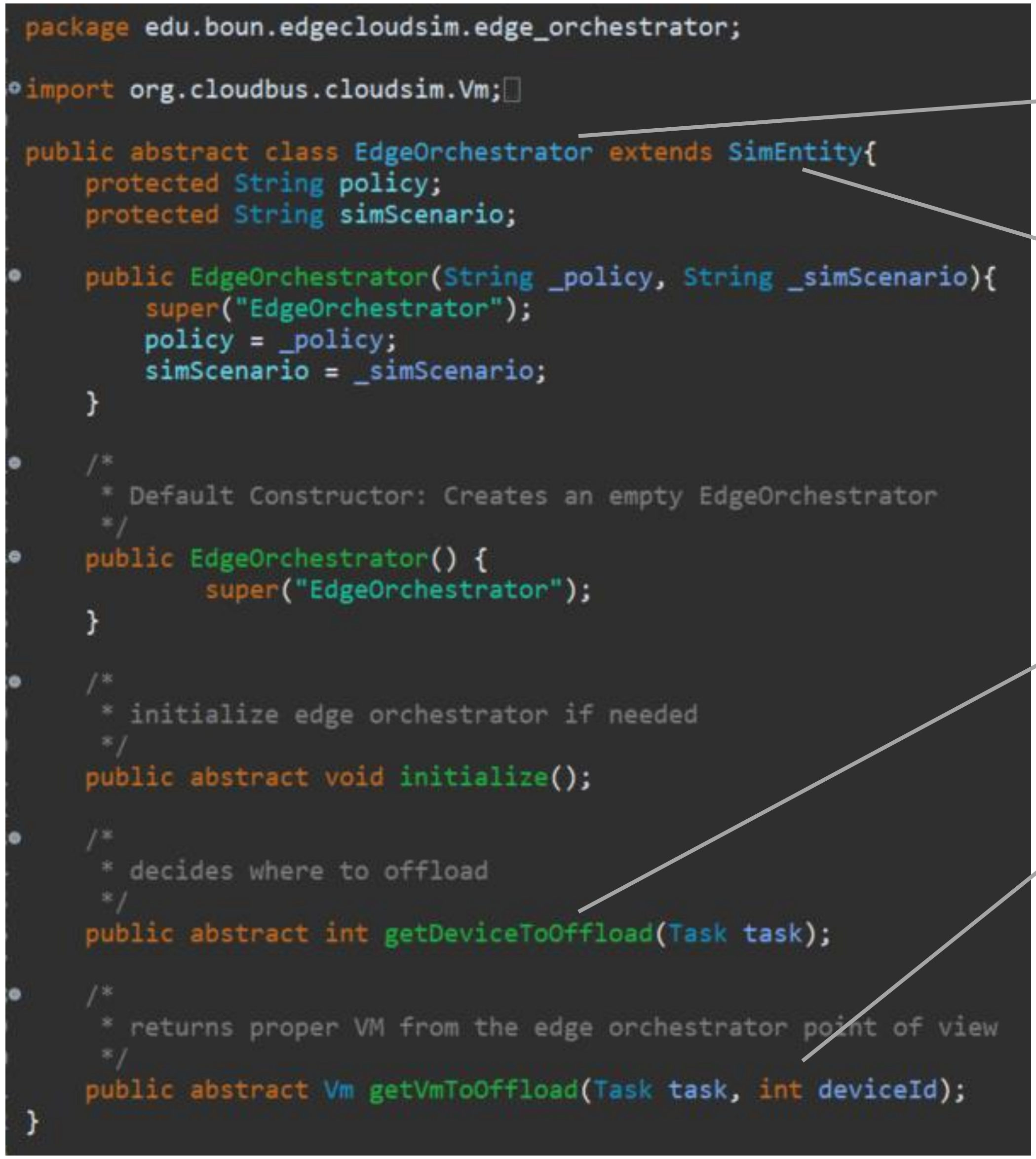
You have to provide a concrete implementation of ScenarioFactory class to create the SimManager class and run the simulation in your java main method



```java
// Generate EdgeCloudsim Scenario Factory
ScenarioFactory sampleFactory = new SampleScenarioFactory(j,
// Generate EdgeCloudSim Simulation Manager
SimManager manager = new SimManager(sampleFactory, j, simSce
// Start simulation
manager.startSimulation();
```

Sample code snippet from Java main method

Important Classes: EdgeOrchestrator



```
package edu.boun.edgecloudsim.edge_orchestrator;

import org.cloudbus.cloudsim.Vm;

public abstract class EdgeOrchestrator extends SimEntity{
    protected String policy;
    protected String simScenario;

    public EdgeOrchestrator(String _policy, String _simScenario){
        super("EdgeOrchestrator");
        policy = _policy;
        simScenario = _simScenario;
    }

    /*
     * Default Constructor: Creates an empty EdgeOrchestrator
     */
    public EdgeOrchestrator() {
          super("EdgeOrchestrator");
    }

    /*
     * initialize edge orchestrator if needed
     */
    public abstract void initialize();

    /*
     * decides where to offload
     */
    public abstract int getDeviceToOffload(Task task);

    /*
     * returns proper VM from the edge orchestrator point of view
     */
    public abstract Vm getVmToOffload(Task task, int deviceId);
}
```

The edge orchestrator can be considered as the central nervous system. Make all critical decisions here in this class.

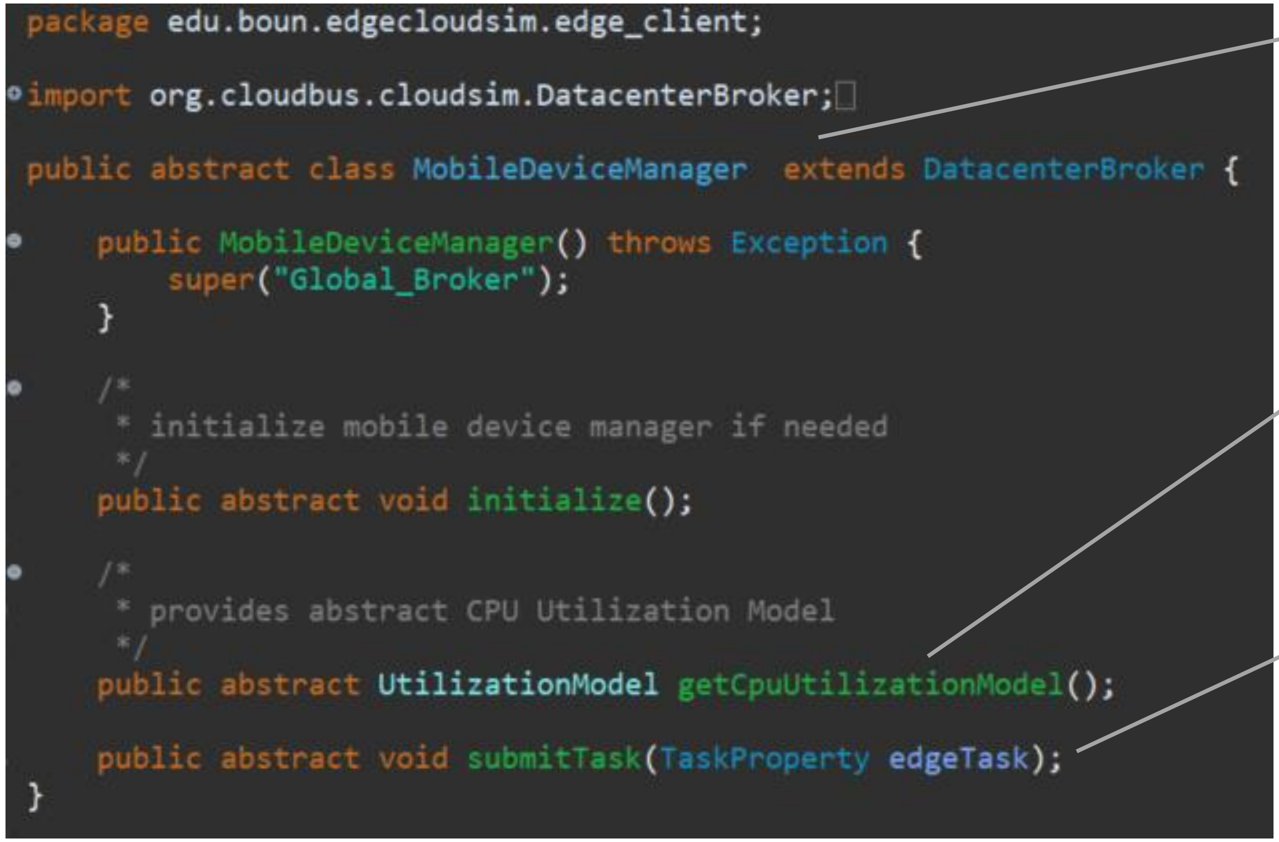
You need to extend SimEntity to create custom events!

This generic function is mostly used to decide which task should be sent to which datacenter.

This generic function is mostly used to decide which VM the task should be assigned to.

★ You can add other helper functions in your concrete implementation of this class to handle your business logic!

Important Classes: MobileDeviceManager



```
package edu.boun.edgecloudsim.edge_client;

import org.cloudbus.cloudsim.DatacenterBroker;

public abstract class MobileDeviceManager  extends DatacenterBroker {

    public MobileDeviceManager() throws Exception {
        super("Global_Broker");
    }

    /*
     * initialize mobile device manager if needed
     */
    public abstract void initialize();

    /*
     * provides abstract CPU Utilization Model
     */
    public abstract UtilizationModel getCpuUtilizationModel();

    public abstract void submitTask(TaskProperty edgeTask);
}
```

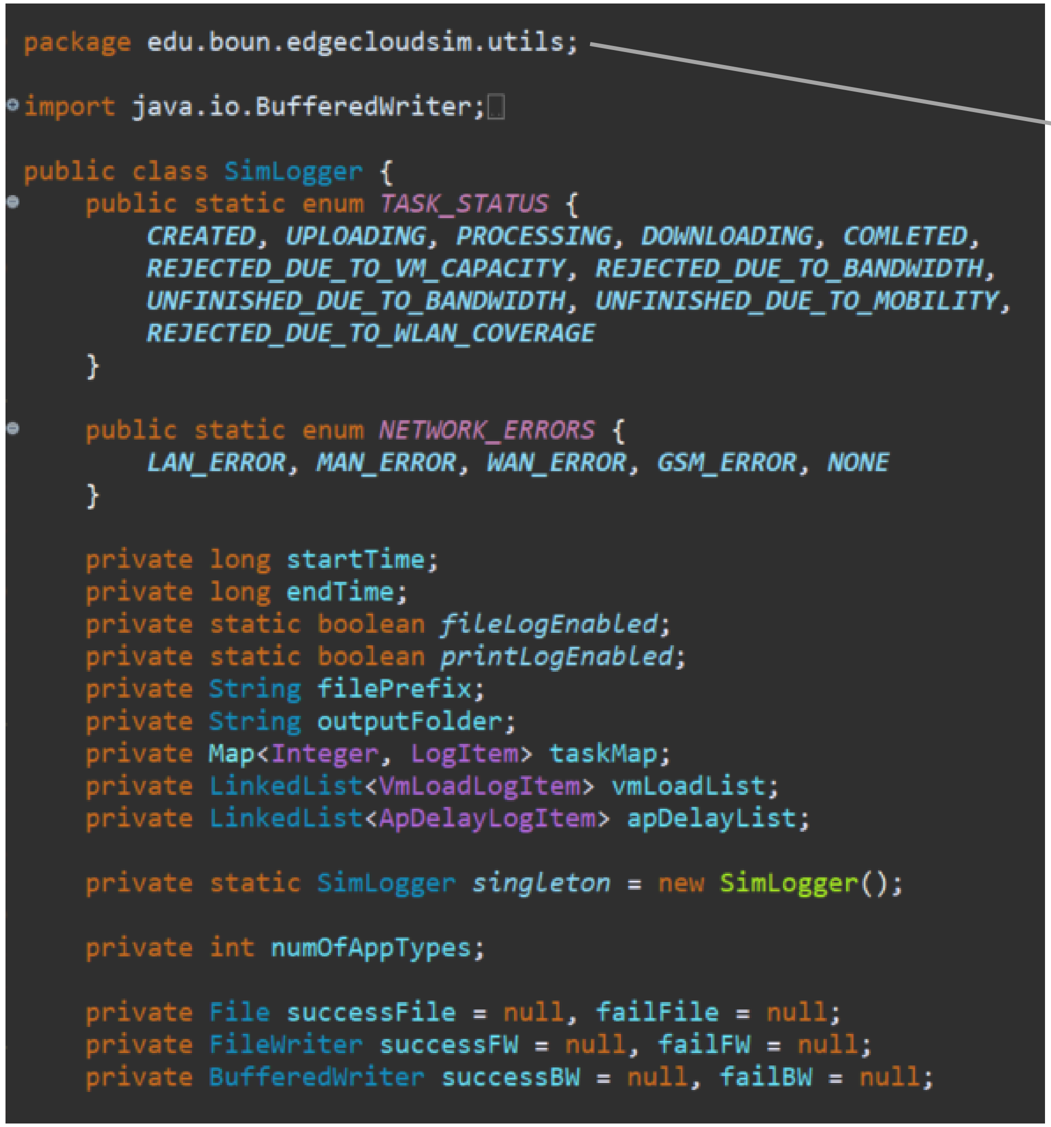
MobileDeviceManager extends CloudSim's DatacenterBroker class which represents a broker (user).

CloudSim does not provide realistic VM CPU utilization model, so that this class is responsible for providing a CPU utilization model based on the simulation scenario

It is basically responsible for submitting VM provisioning requests to data centers (submitting tasks to VMs)

★ For convenience, this class often handles (simulates) the movement of tasks between the entities.

Important Classes: SimLogger



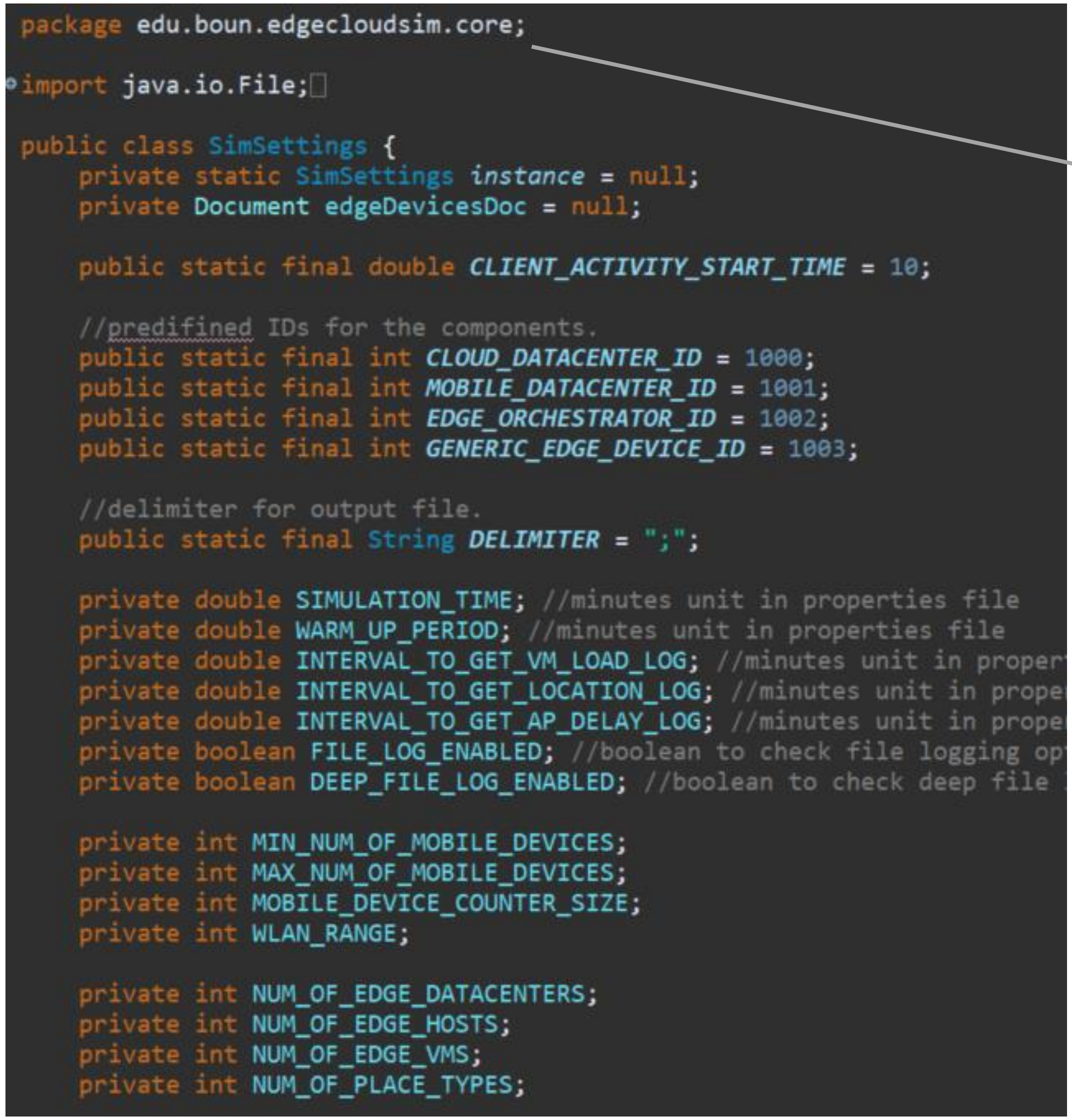
```
package edu.boun.edgecloudsim.utils;

import java.io.BufferedWriter;

public class SimLogger {
    public static enum TASK_STATUS {
        CREATED, UPLOADING, PROCESSING, DOWNLOADING, COMLETED,
        REJECTED_DUE_TO_VM_CAPACITY, REJECTED_DUE_TO_BANDWIDTH,
        UNFINISHED_DUE_TO_BANDWIDTH, UNFINISHED_DUE_TO_MOBILITY,
        REJECTED_DUE_TO_WLAN_COVERAGE
    }

    public static enum NETWORK_ERRORS {
        LAN_ERROR, MAN_ERROR, WAN_ERROR, GSM_ERROR, NONE
    }

    private long startTime;
    private long endTime;
    private static boolean fileLogEnabled;
    private static boolean printLogEnabled;
    private String filePrefix;
    private String outputFolder;
    private Map<Integer, LogItem> taskMap;
    private LinkedList<VmLoadLogItem> vmLoadList;
    private LinkedList<ApDelayLogItem> apDelayList;

    private static SimLogger singleton = new SimLogger();

    private int numOfAppTypes;

    private File successFile = null, failFile = null;
    private FileWriter successFW = null, failFW = null;
    private BufferedWriter successBW = null, failBW = null;
```

The SimSettings class is located under util package, and responsible for saving simulation results to files.

The first version of SimLogger was logging events to files as they happened, but this approach requires a lot of I/O. Therefore, it was updated to collect the results of the events in data structures and write them to the files at the end of the simulation!

★ Unfortunately, the SimLogger class is not designed to be extended via the factory pattern; if you need to log different simulation results, you will need to modify this core class!

Important Classes: SimSettings



```
package edu.boun.edgecloudsim.core;

import java.io.File;

public class SimSettings {
    private static SimSettings instance = null;
    private Document edgeDevicesDoc = null;

    public static final double CLIENT_ACTIVITY_START_TIME = 10;

    //predifined IDs for the components.
    public static final int CLOUD_DATACENTER_ID = 1000;
    public static final int MOBILE_DATACENTER_ID = 1001;
    public static final int EDGE_ORCHESTRATOR_ID = 1002;
    public static final int GENERIC_EDGE_DEVICE_ID = 1003;

    //delimiter for output file.
    public static final String DELIMITER = ";";

    private double SIMULATION_TIME; //minutes unit in properties file
    private double WARM_UP_PERIOD; //minutes unit in properties file
    private double INTERVAL_TO_GET_VM_LOAD_LOG; //minutes unit in proper
    private double INTERVAL_TO_GET_LOCATION_LOG; //minutes unit in prope
    private double INTERVAL_TO_GET_AP_DELAY_LOG; //minutes unit in prope
    private boolean FILE_LOG_ENABLED; //boolean to check file logging op
    private boolean DEEP_FILE_LOG_ENABLED; //boolean to check deep file

    private int MIN_NUM_OF_MOBILE_DEVICES;
    private int MAX_NUM_OF_MOBILE_DEVICES;
    private int MOBILE_DEVICE_COUNTER_SIZE;
    private int WLAN_RANGE;

    private int NUM_OF_EDGE_DATACENTERS;
    private int NUM_OF_EDGE_HOSTS;
    private int NUM_OF_EDGE_VMS;
    private int NUM_OF_PLACE_TYPES;
```

The SimSettings class is located under core package, and responsible for storing all simulation settings by reading the values in the configuration files.

Unfortunately, the SimSettings class is not designed to be extended via the factory pattern; if you need to store different simulation settings for your simulation scenario, you will need to modify this core class!

Ease of Use

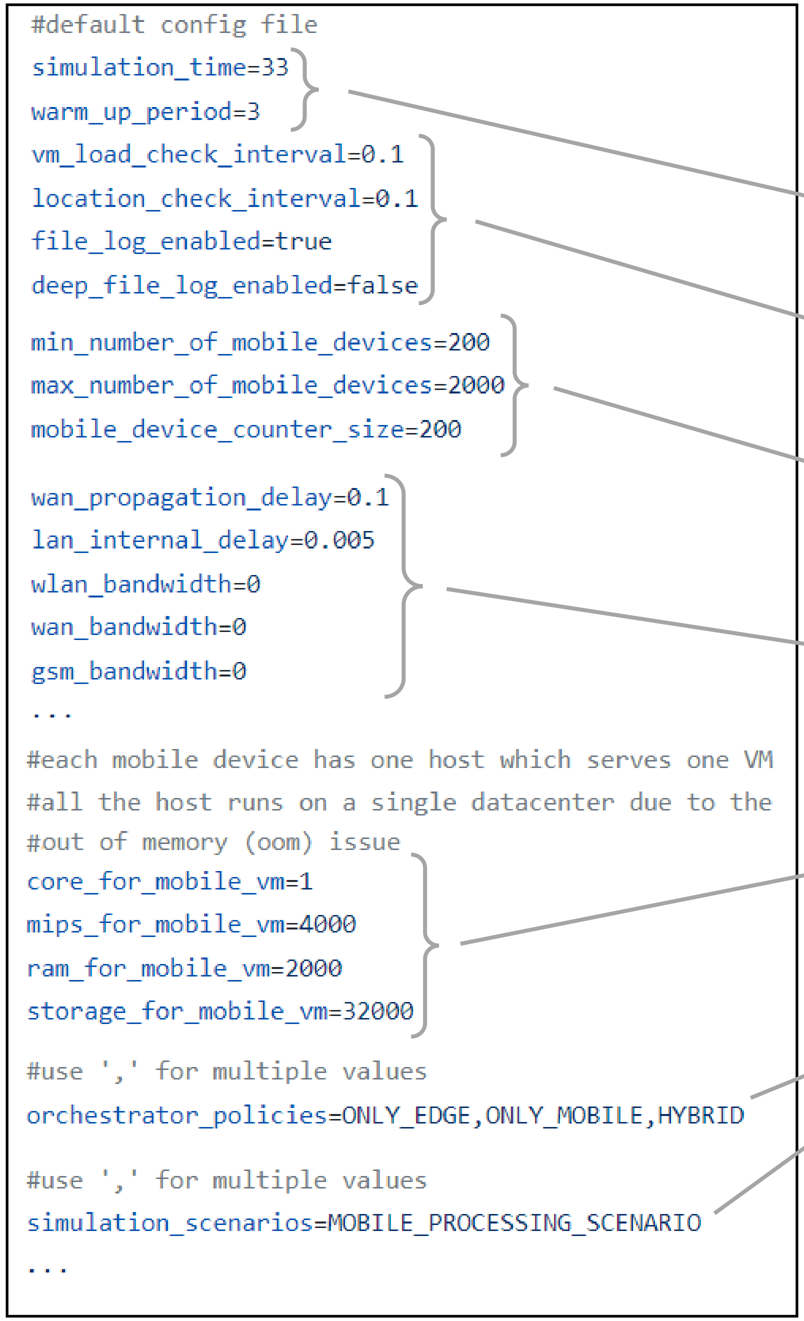
Problems

- Too many parameters are used in the simulations
- Managing parameters programmatically is difficult

Solution

- EdgeCloudSim reads parameters dynamically
  - ✓Simulation settings are managed in configuration file
  - ✓Application properties are stored in xml file
  - ✓Edge devices (datacenters, hosts, VMs etc.) are defined in xml file

config.properties file



```
#default config file
simulation_time=33
warm_up_period=3
vm_load_check_interval=0.1
location_check_interval=0.1
file_log_enabled=true
deep_file_log_enabled=false

min_number_of_mobile_devices=200
max_number_of_mobile_devices=2000
mobile_device_counter_size=200

wan_propagation_delay=0.1
lan_internal_delay=0.005
wlan_bandwidth=0
wan_bandwidth=0
gsm_bandwidth=0
...
#each mobile device has one host which serves one VM
#all the host runs on a single datacenter due to the
#out of memory (oom) issue
core_for_mobile_vm=1
mips_for_mobile_vm=4000
ram_for_mobile_vm=2000
storage_for_mobile_vm=32000

#use ',' for multiple values
orchestrator_policies=ONLY_EDGE,ONLY_MOBILE,HYBRID

#use ',' for multiple values
simulation_scenarios=MOBILE_PROCESSING_SCENARIO
...
```

Simulation and warm-up time

Values to use for file logging

Number of mobile devices to be used in the simulation

Values to be used in the network model

CPU, memory and storage specifications for mobile devices

Orchestrator policies and scenarios

Any variables related to simulation settings are kept in this file

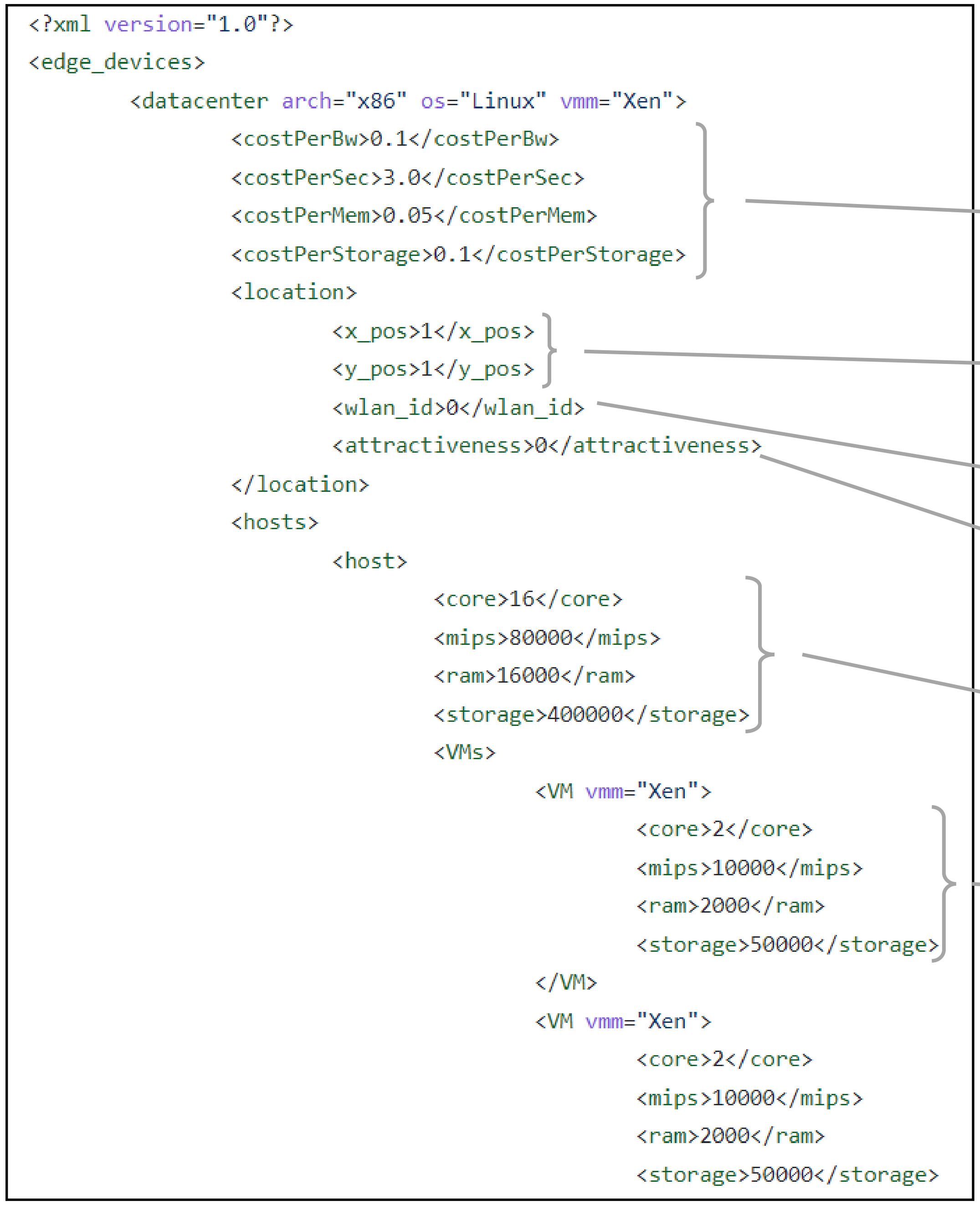
applications.xml file

```xml
<?xml version="1.0"?>
<applications>
  <application name="AUGMENTED_REALITY">
    <usage_percentage>30</usage_percentage>
    <prob_cloud_selection>20</prob_cloud_selection>
    <poisson_interarrival>2</poisson_interarrival>
    <delay_sensitivity>0</delay_sensitivity>
    <active_period>40</active_period>
    <idle_period>20</idle_period>
    <data_upload>1500</data_upload>
    <data_download>250</data_download>
    <task_length>12000</task_length>
    <required_core>1</required_core>
    <vm_utilization_on_edge>8</vm_utilization_on_edge>
    <vm_utilization_on_cloud>0.8</vm_utilization_on_cloud>
    <vm_utilization_on_mobile>20</vm_utilization_on_mobile>
  </application>
  <application name="HEALTH_APP">
     <usage_percentage>20</usage_percentage>
```

- How much this app is used by mobile devices
- Possibility of offloading to the cloud (for some scenarios)
- Interarrival time for task generation (Poisson process)
- How sensitive the app is to latency (for some scenarios)
- Active period for Active/Idle task generation model
- Idle period for Active/Idle task generation model
- Average size of data loaded by the task
- Average size of data downloaded after the task execution
- Average task length in millions of instructions (MI)
- How much CPU core is used by tasks created by this app
- CPU utilization ratio of related tasks on compute nodes

★ Applications create tasks to be offloaded with the properties specified in this file

edge_devices.xml file



```
<?xml version="1.0"?>
<edge_devices>
        <datacenter arch="x86" os="Linux" vmm="Xen">
                <costPerBw>0.1</costPerBw>
                <costPerSec>3.0</costPerSec>
                <costPerMem>0.05</costPerMem>
                <costPerStorage>0.1</costPerStorage>
                <location>
                        <x_pos>1</x_pos>
                        <y_pos>1</y_pos>
                        <wlan_id>0</wlan_id>
                        <attractiveness>0</attractiveness>
                </location>
                <hosts>
                        <host>
                                <core>16</core>
                                <mips>80000</mips>
                                <ram>16000</ram>
                                <storage>400000</storage>
                                <VMs>
                                        <VM vmm="Xen">
                                                <core>2</core>
                                                <mips>10000</mips>
                                                <ram>2000</ram>
                                                <storage>50000</storage>
                                        </VM>
                                        <VM vmm="Xen">
                                                <core>2</core>
                                                <mips>10000</mips>
                                                <ram>2000</ram>
                                                <storage>50000</storage>
```

Bandwidth, CPU, memory and storage cost values for this datacenter (based on CloudSim specifications)

x, y position of the datacenter (will be important for your mobility model)

Each WLAN should have a unique id

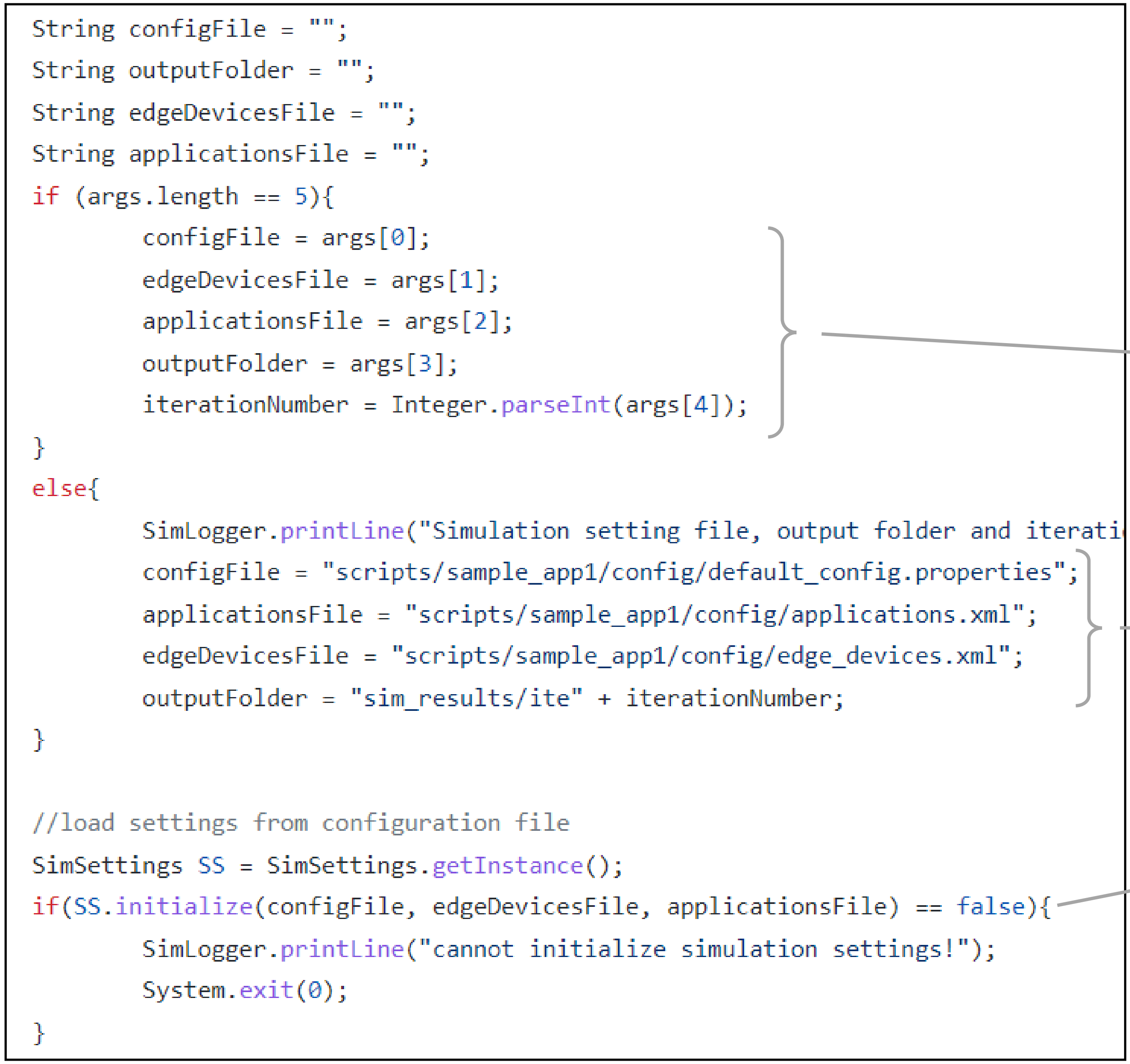
Attractiveness level of this location (for some scenarios)

CPU, memory and storage specifications for the corresponding host

CPU, memory and storage specifications for the corresponding host

★ Edge servers are created with properties specified in this file

Using Conf. Files



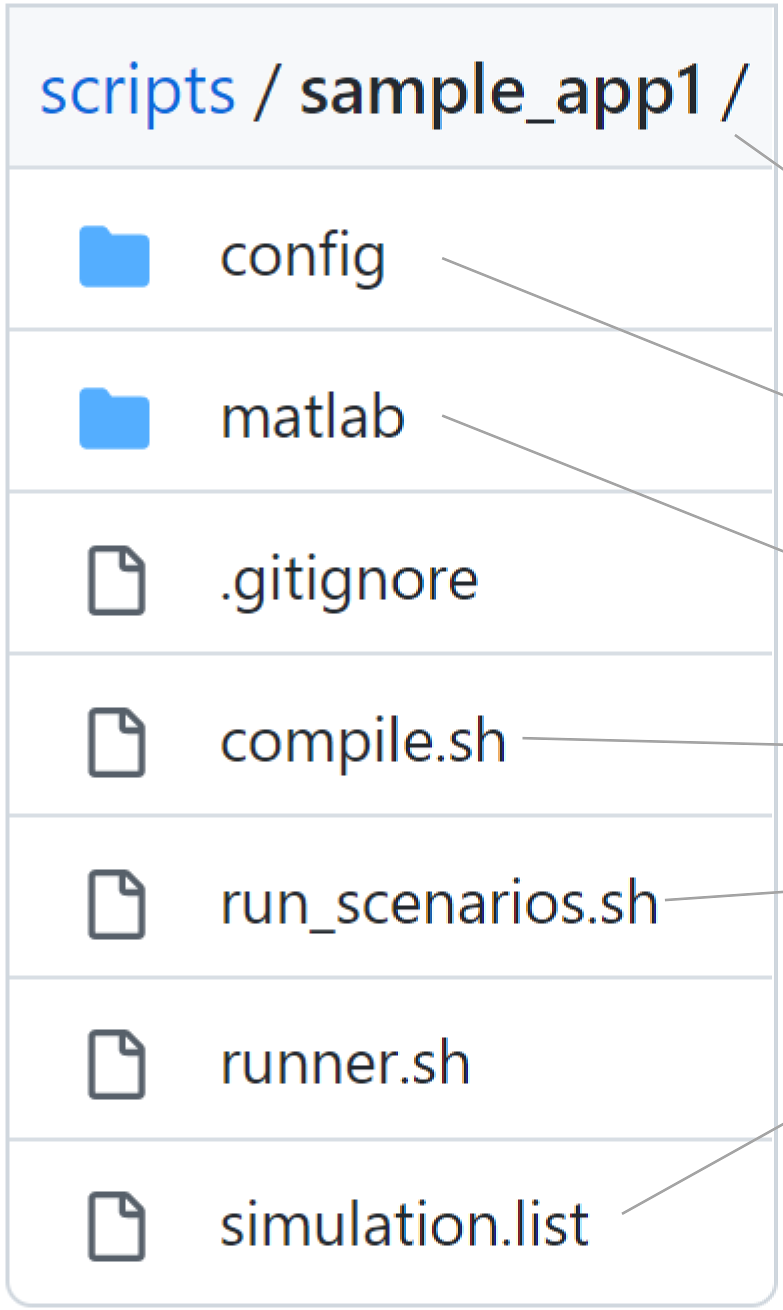
```
String configFile = "";
String outputFolder = "";
String edgeDevicesFile = "";
String applicationsFile = "";
if (args.length == 5){
        configFile = args[0];
        edgeDevicesFile = args[1];
        applicationsFile = args[2];
        outputFolder = args[3];
        iterationNumber = Integer.parseInt(args[4]);
}
else{
        SimLogger.printLine("Simulation setting file, output folder and iterati
        configFile = "scripts/sample_app1/config/default_config.properties";
        applicationsFile = "scripts/sample_app1/config/applications.xml";
        edgeDevicesFile = "scripts/sample_app1/config/edge_devices.xml";
        outputFolder = "sim_results/ite" + iterationNumber;
}

//load settings from configuration file
SimSettings SS = SimSettings.getInstance();
if(SS.initialize(configFile, edgeDevicesFile, applicationsFile) == false){
        SimLogger.printLine("cannot initialize simulation settings!");
        System.exit(0);
}
```

Configuration files are mostly provided via command line arguments (especially when running through terminal).

Sometimes developers can use static file path for convenience (especially when running the simulations through IDEs during development).

SimSettings is initialized with the configuration files

Helper Scripts


scripts / sample_app1 /

- config
- matlab
- .gitignore
- compile.sh
- run_scenarios.sh
- runner.sh
- simulation.list

Each application has utilities in its scripts folder to run simulations and plot results easily

Configuration files located in config folder

MATLAB scripts to generate graphs

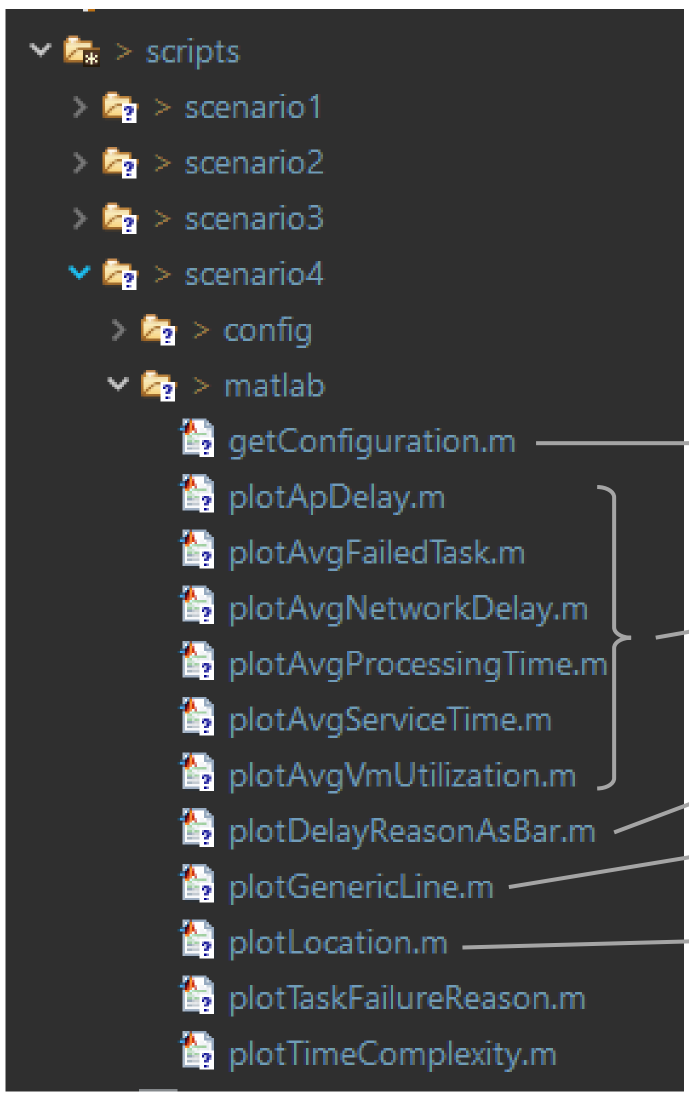
Script to compile sample app

Script to run sample app

Used to run multiple simulations with different configurations (config files)

★ Scripts are tested & verified on Linux based operating systems including Mac OS

Plotting Simulation Results


```
scripts
  scenario1
  scenario2
  scenario3
  scenario4
    config
    matlab
      getConfiguration.m
      plotApDelay.m
      plotAvgFailedTask.m
      plotAvgNetworkDelay.m
      plotAvgProcessingTime.m
      plotAvgServiceTime.m
      plotAvgVmUtilization.m
      plotDelayReasonAsBar.m
      plotGenericLine.m
      plotLocation.m
      plotTaskFailureReason.m
      plotTimeComplexity.m
```

★ First of all, configure getConfiguration.m file based on your simulation scenario and preferences!

getConfiguration.m → Configuration file for the other helper scripts

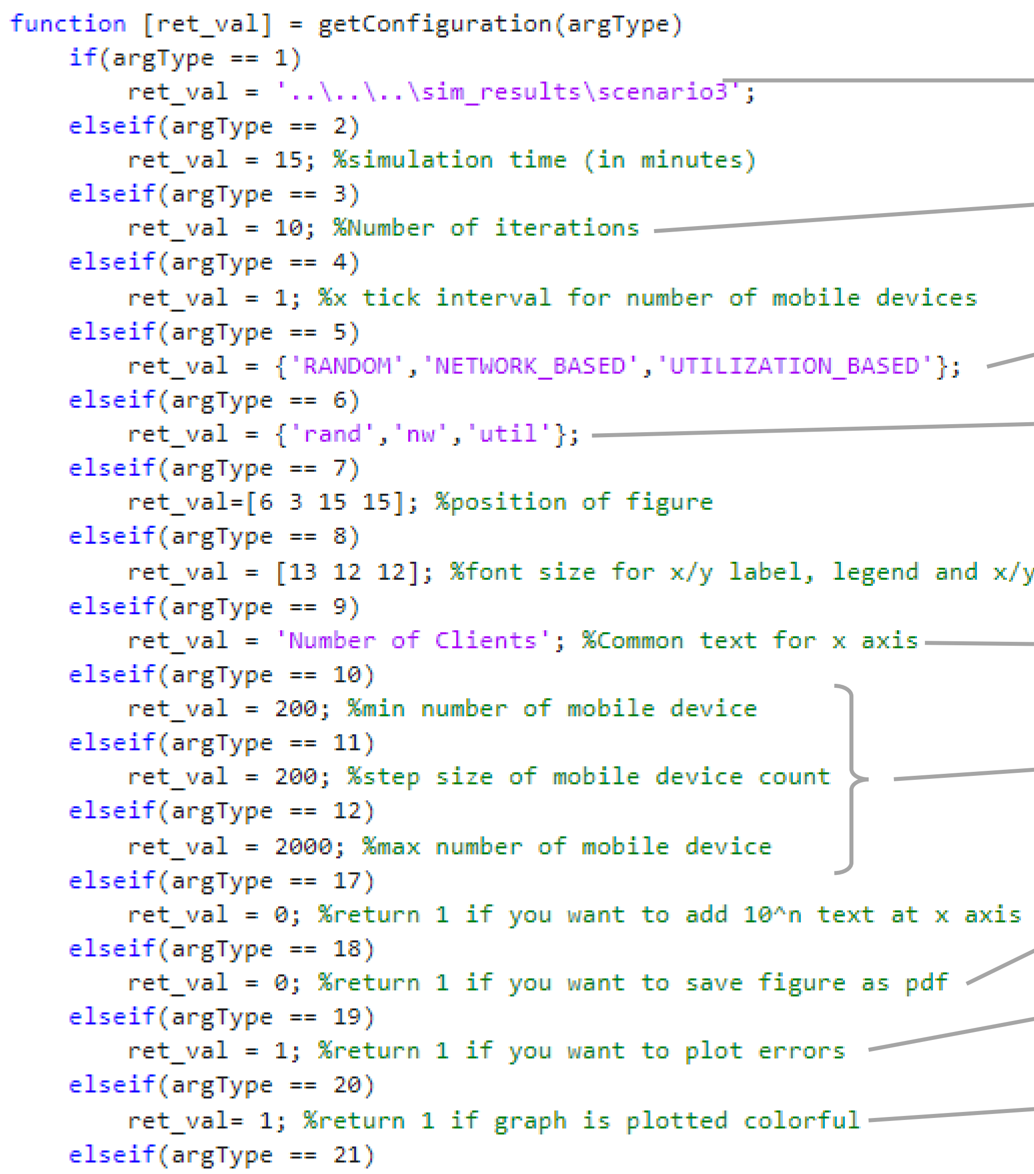
plotApDelay.m … plotAvgVmUtilization.m → Helper scripts to plot graphics using generic line plotter

plotDelayReasonAsBar.m → Plots bar graphs for delay types

plotGenericLine.m → Generic line plotter that generates most of the graphics

plotLocation.m → Plots a heat-map like graphics to analyze location of the clients

Matlab Configuration Script



```
function [ret_val] = getConfiguration(argType)
    if(argType == 1)
        ret_val = '..\..\..\sim_results\scenario3';
    elseif(argType == 2)
        ret_val = 15; %simulation time (in minutes)
    elseif(argType == 3)
        ret_val = 10; %Number of iterations
    elseif(argType == 4)
        ret_val = 1; %x tick interval for number of mobile devices
    elseif(argType == 5)
        ret_val = {'RANDOM','NETWORK_BASED','UTILIZATION_BASED'};
    elseif(argType == 6)
        ret_val = {'rand','nw','util'};
    elseif(argType == 7)
        ret_val=[6 3 15 15]; %position of figure
    elseif(argType == 8)
        ret_val = [13 12 12]; %font size for x/y label, legend and x/y
    elseif(argType == 9)
        ret_val = 'Number of Clients'; %Common text for x axis
    elseif(argType == 10)
        ret_val = 200; %min number of mobile device
    elseif(argType == 11)
        ret_val = 200; %step size of mobile device count
    elseif(argType == 12)
        ret_val = 2000; %max number of mobile device
    elseif(argType == 17)
        ret_val = 0; %return 1 if you want to add 10^n text at x axis
    elseif(argType == 18)
        ret_val = 0; %return 1 if you want to save figure as pdf
    elseif(argType == 19)
        ret_val = 1; %return 1 if you want to plot errors
    elseif(argType == 20)
        ret_val= 1; %return 1 if graph is plotted colorful
    elseif(argType == 21)
```

Folder path where simulation results are saved

Number of iterations

Scenario names used in the simulations

Corresponding legend texts in figures

Position, size and font size of graphs

Common x axis label

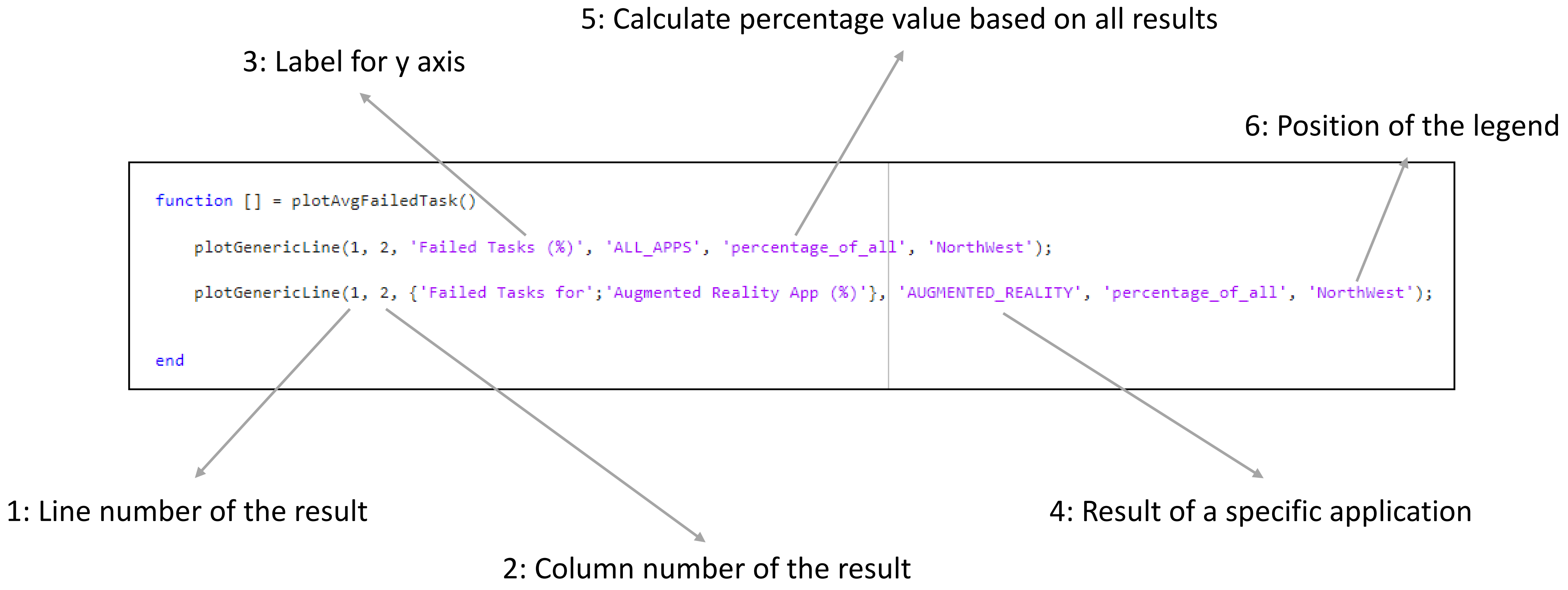
Number of clients used in the simulation

Option to save graph in pdf format

Option to plot graphs with 95% confidence interval error bars

Option to plot graph in color

Sample Plotter Script


5: Calculate percentage value based on all results
3: Label for y axis
6: Position of the legend
1: Line number of the result
2: Column number of the result
4: Result of a specific application

```
function [] = plotAvgFailedTask()

    plotGenericLine(1, 2, 'Failed Tasks (%)', 'ALL_APPS', 'percentage_of_all', 'NorthWest');

    plotGenericLine(1, 2, {'Failed Tasks for';'Augmented Reality App (%)'}, 'AUGMENTED_REALITY', 'percentage_of_all', 'NorthWest');

end
```

Live Demo Sessions

Case Study 1 – VM Scheduling

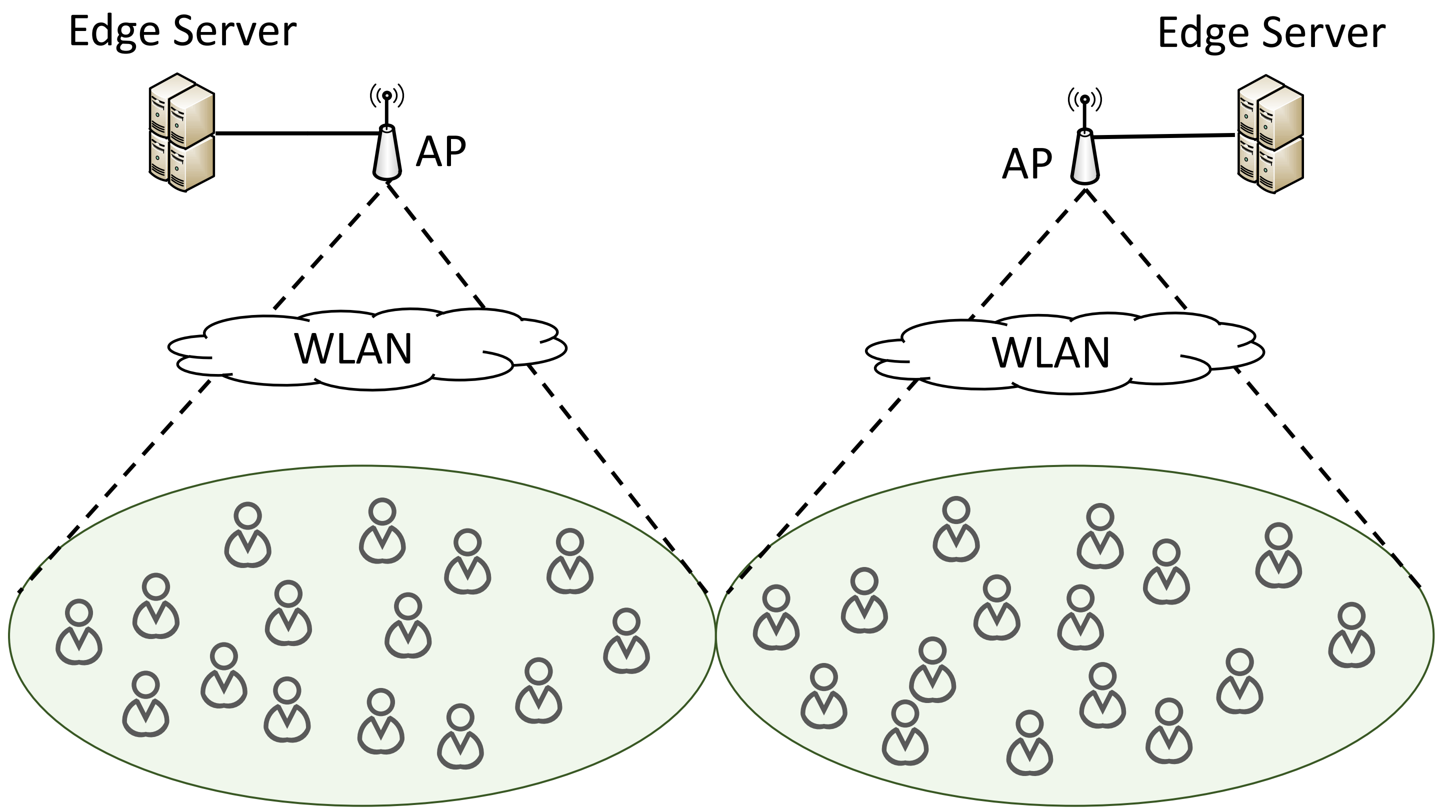
Performance Evaluation of Different VM Allocation Policies

Find the source code below:

https://github.com/CagataySonmez/PerformanceEvaluationWithEdgeCloudSim/tree/main/src/edu/boun/edgecloudsim/applications/scenario1

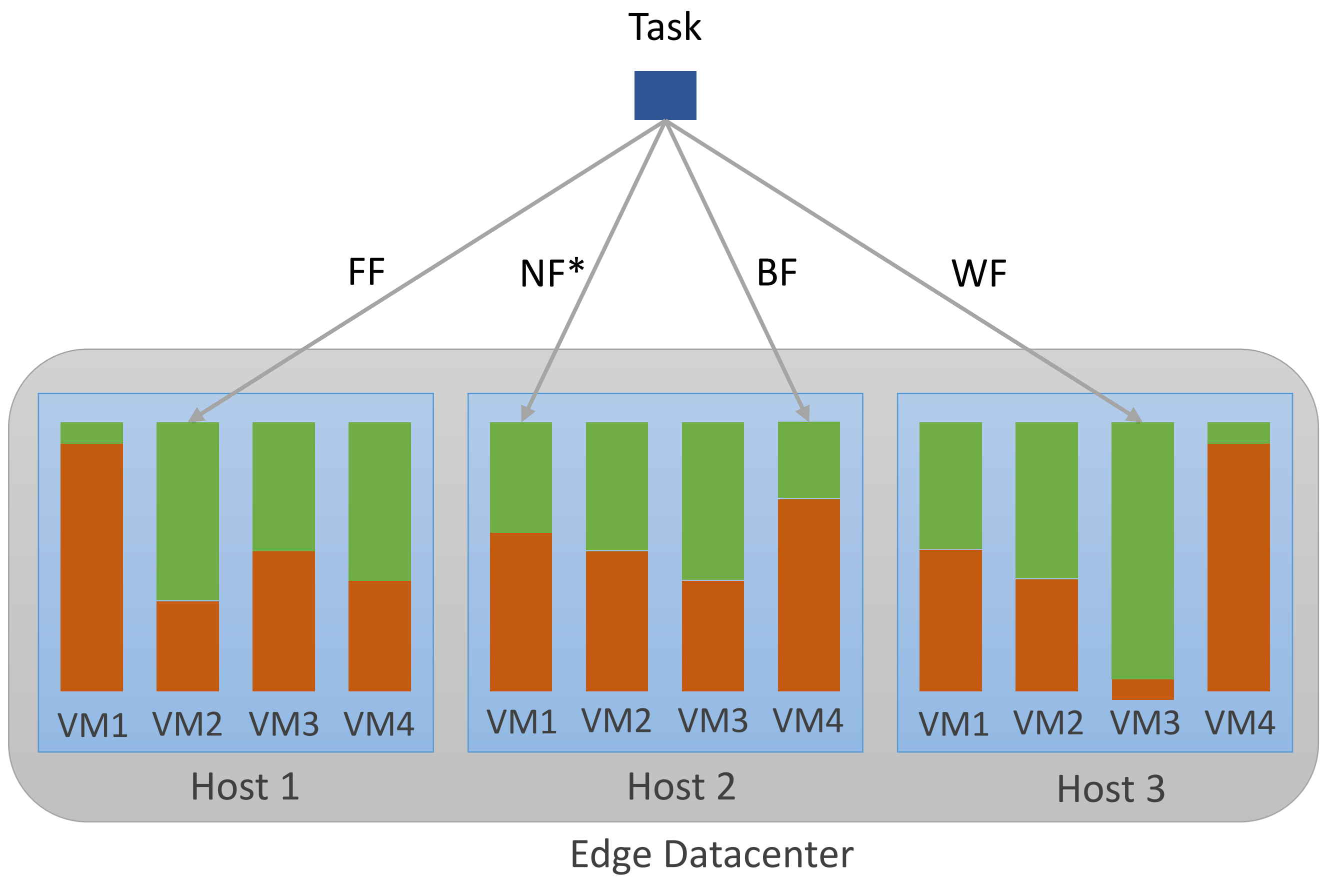
Simulation Scenario

- Mobile devices can only offload task to the edge servers connected to the serving access point
- Edge servers operates a variable number of VMs
- This scenario compares different VM provisioning algorithms



Compatitor VM Provisioning Algorithms

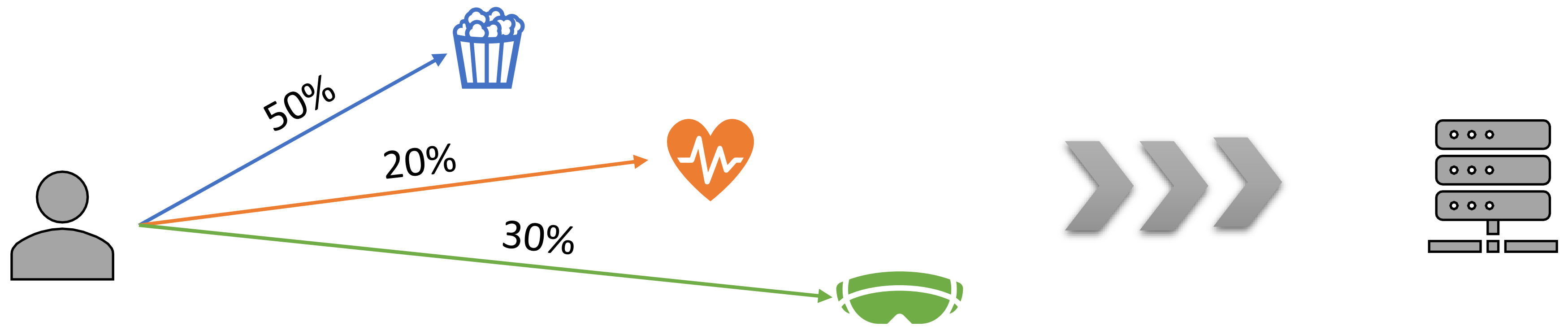
- **Random (RND)**: A random VM is selected
- **First-Fit (FF)**: First available VM is selected
- **Next-Fit (NF)**: The hosts are visited in order and the first suitable VM is selected.
- **Best-Fit (BF)**: The VM with the highest CPU utilization is selected
- **Worst-Fit (WF)**: The VM with the least CPU utilization is selected



* Assuming the previous selection of the NF algorithm was one of the VM in Host1.

Applications Used in This Simulation

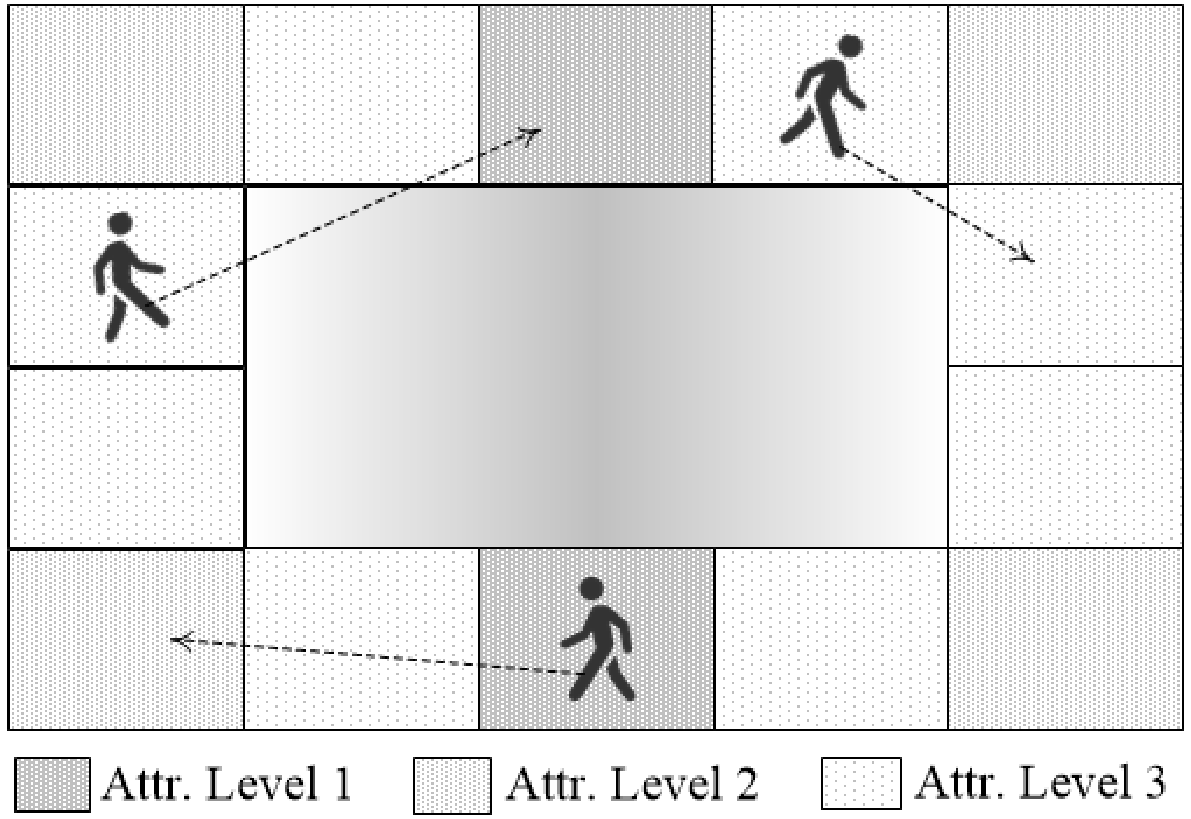
Parameter	Aug. Reality	Health	Infotainment
Usage Percentage (%)	30	20	50
Task Interarrival (sec)	2	3	7
Active/Idle Period (sec)	40/20	45/90	30/45
VM Utilization on Edge Server (%)	6	2	10
Task Length (GI)	15	3	9
Upload/Download Data (KB)	1500/50	50/1250	250/1000



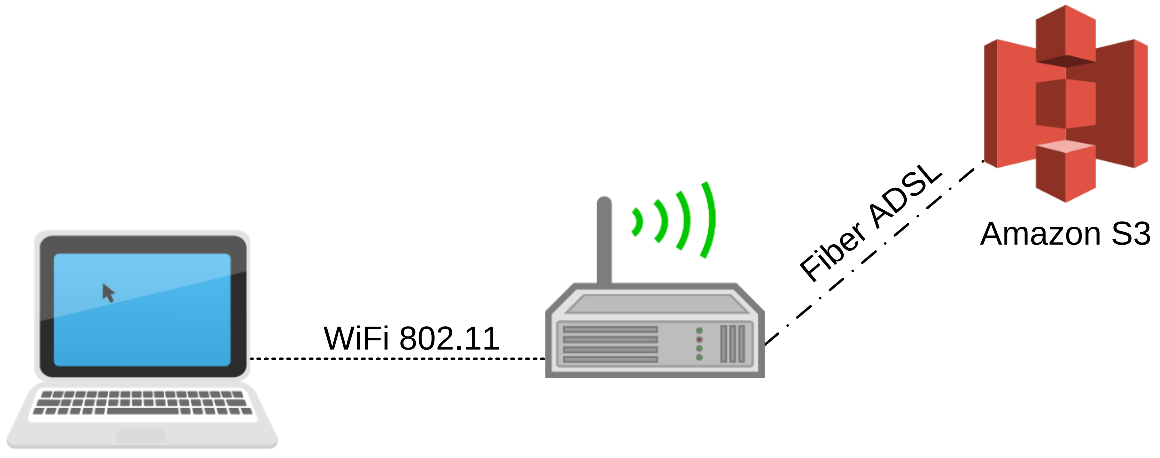
★ Clients are utilizing an application that generates task according to a Poisson process

Nomadic Mobility Model

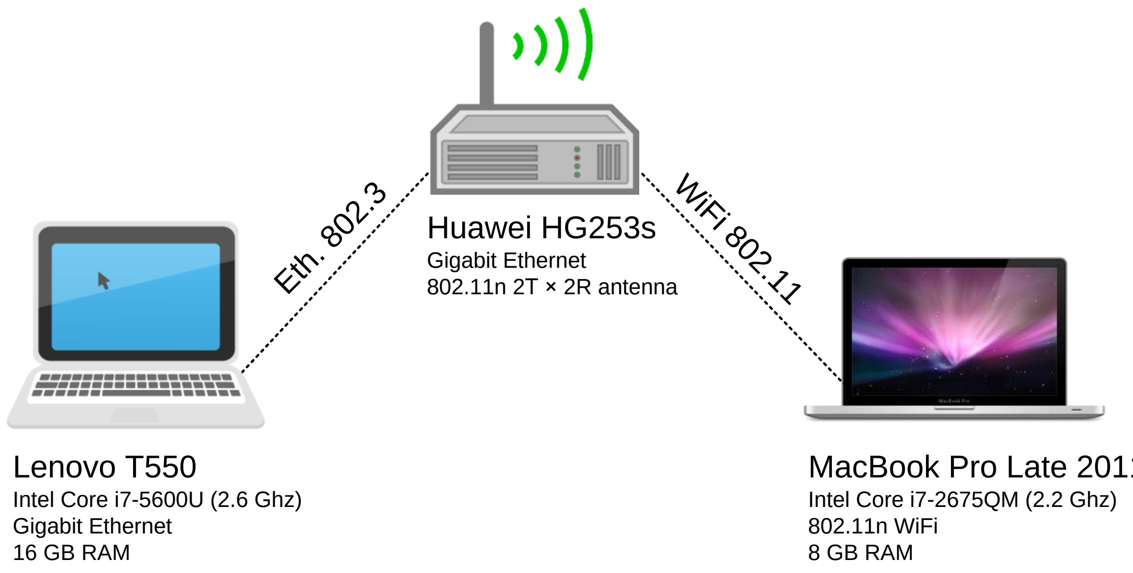
- There is no real time walking pattern, user location is updated at random time intervals
- Probability of selecting a new location is same for all locations
- We use variable locations with different attractiveness levels in simulations
- The attractiveness level determines how much time the user will spend (dwell time) in the corresponding place



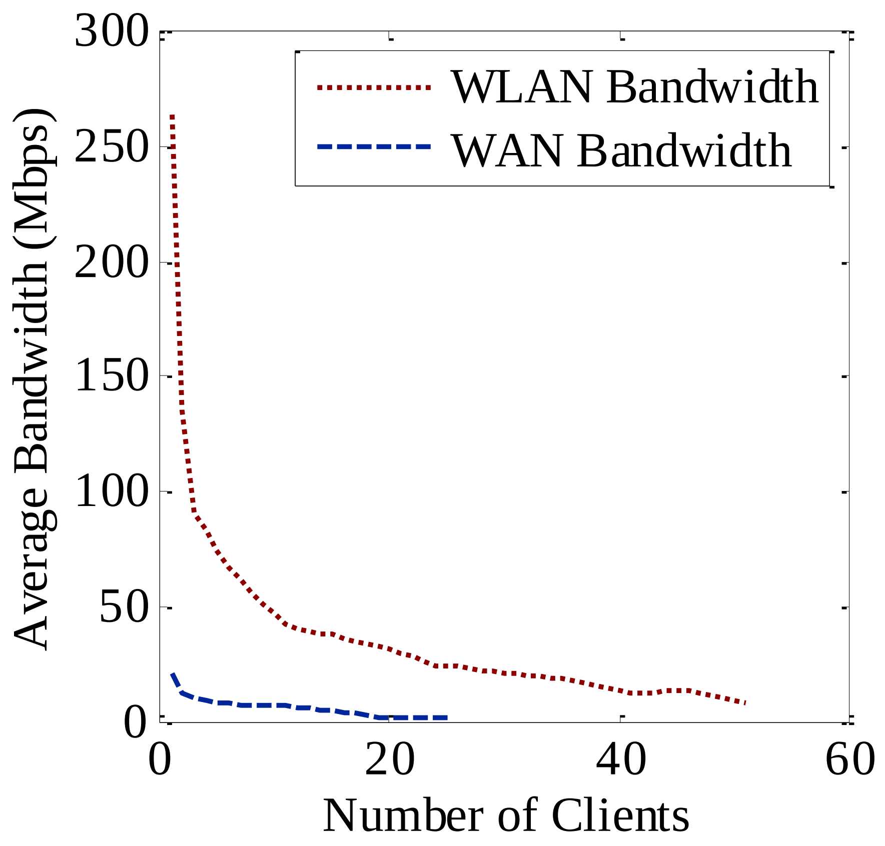
Empirical WAN/WLAN Model



WAN Bandwidth Analysis Setup



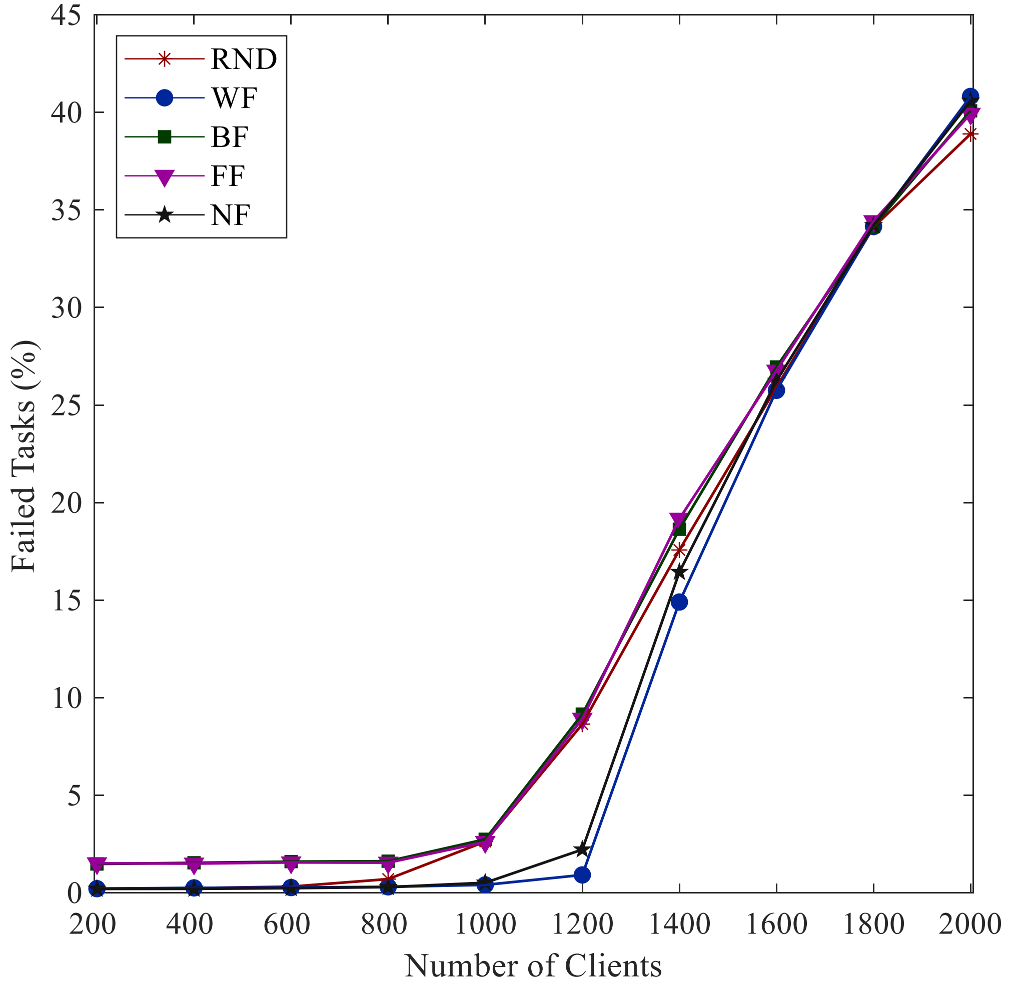
WLAN Bandwidth Analysis Setup

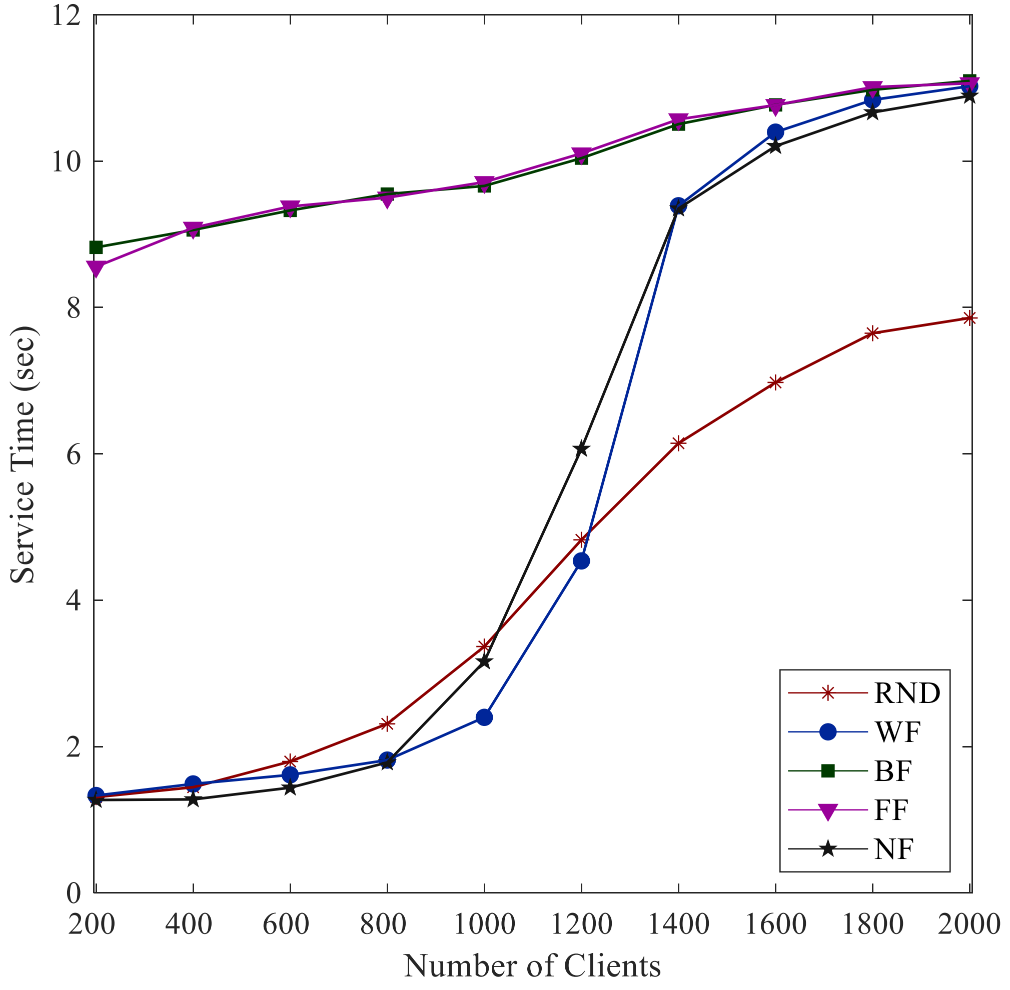


Simulation Parameters

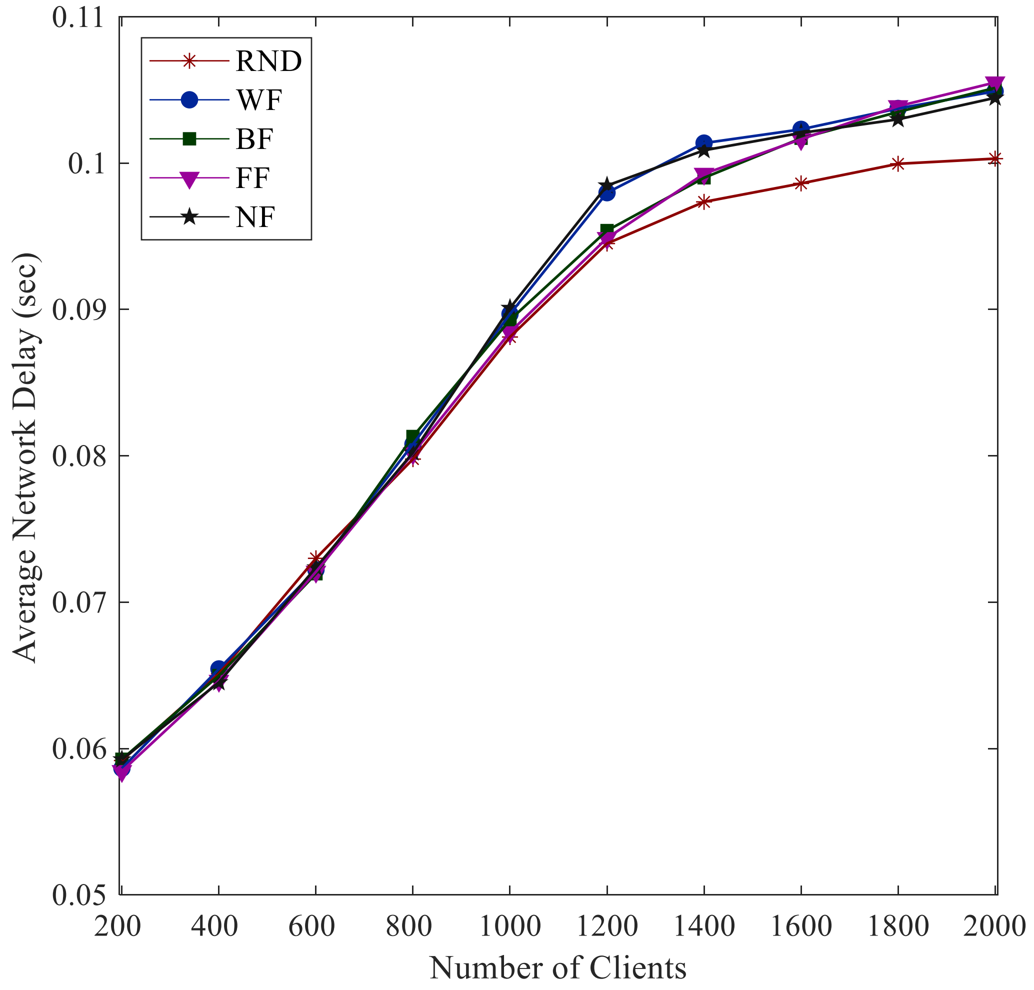
Parameter	Value
Simulation Time/Warm-up Period	15/1 minutes
Number of Repetition	10
WLAN Delay Model	Empirical
MAN Delay	Fixed (5 ms)
Number of VM per Edge Host	8
Number of Cores per Edge VM CPU	2
VM Processor Speed per Edge	10 GIPS
Mobility Model	Nomadic Mobility
Number of Locations for Type 1/2/3 places	2/4/8
Mean waiting time in Type 1/2/3 places	10/10/10 minutes

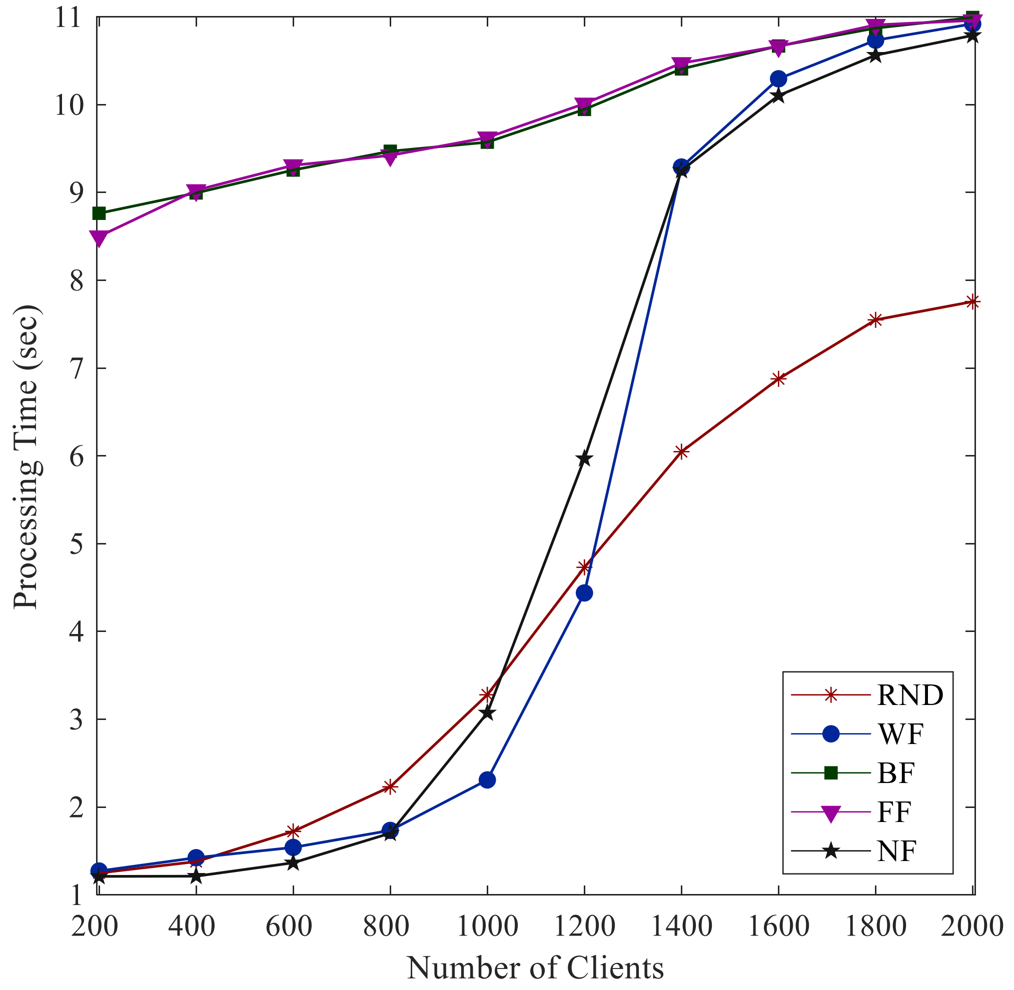
Important Simulation Results





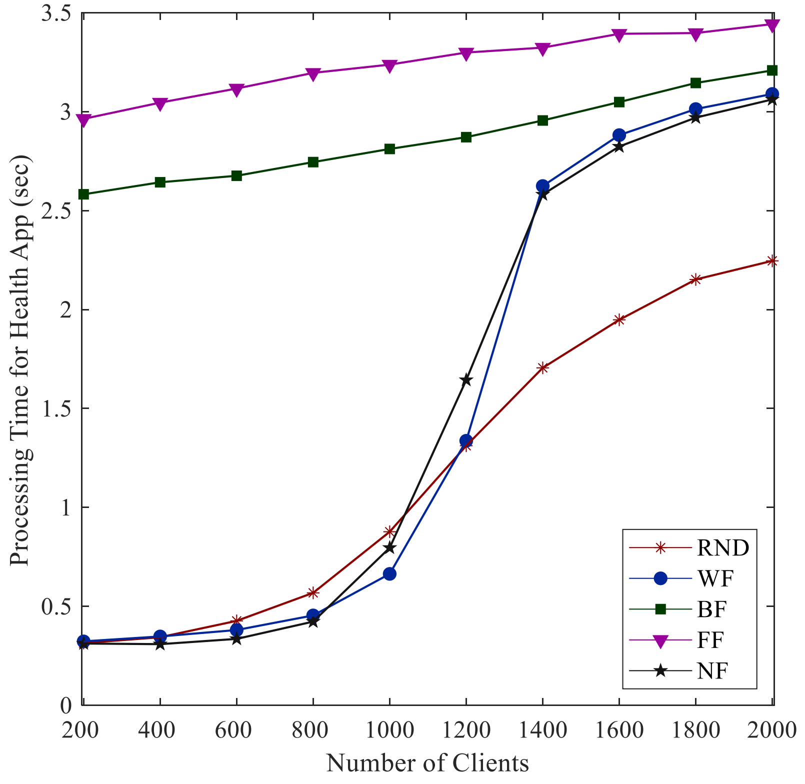
Service Time Analysis



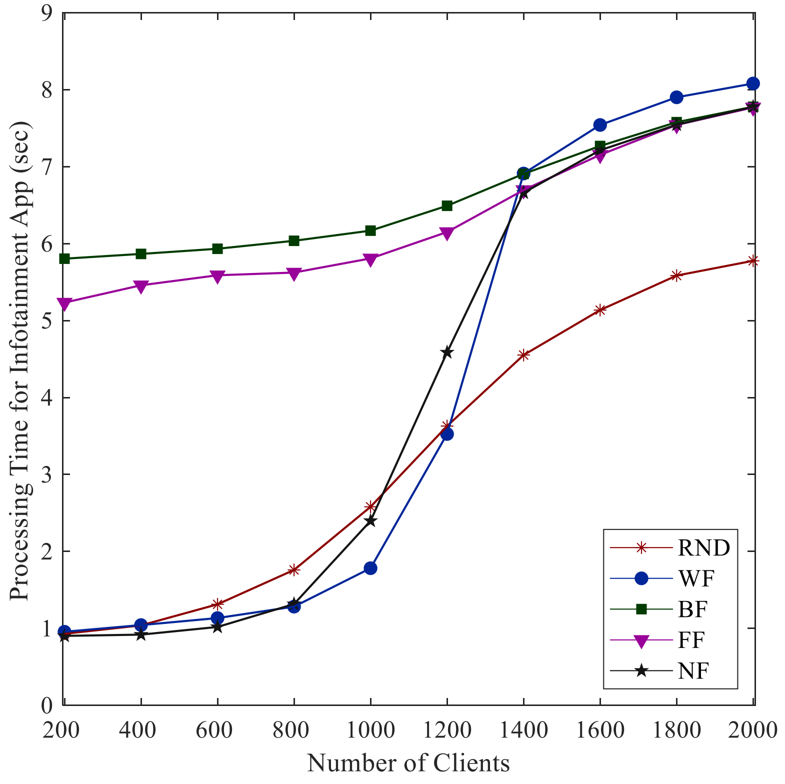


Processing time dominates the average service time

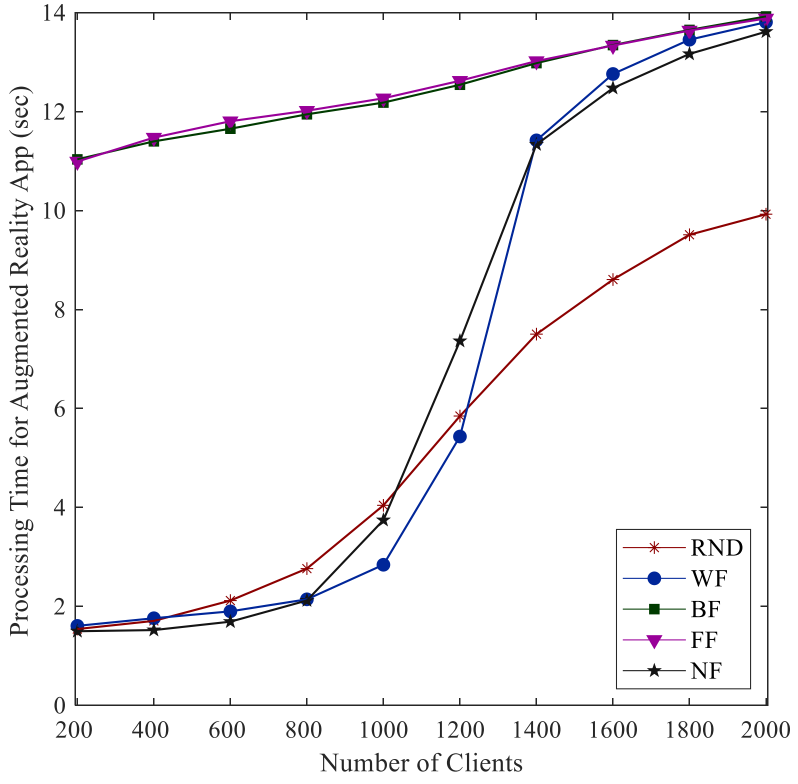
Processing Time per Task Types



Task Size: 3 GI



Task Size: 9 GI



Task Size: 15 GI

Case Study 2 – What to Offload

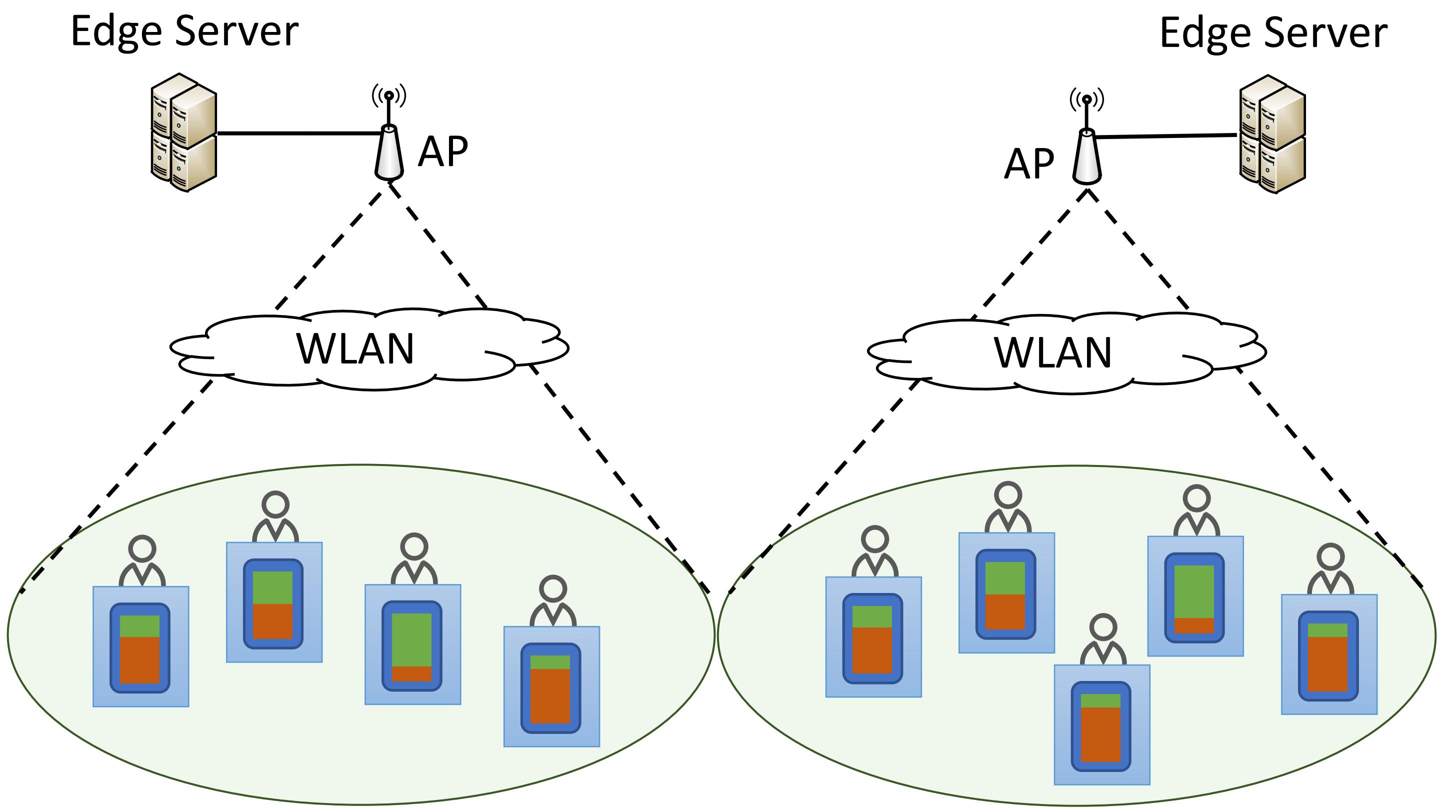
Performance Evaluation of Different Approaches that Decide Granularity of Task Offloading

Find the source code below:

https://github.com/CagataySonmez/PerformanceEvaluationWithEdgeCloudSim/tree/main/src/edu/boun/edgecloudsim/applications/scenario2

Simulation Scenario

- Mobile devices can operate tasks locally or offload them to the edge servers connected to the serving access point
- Edge servers operates a variable number of VMs
- Worst-fit VM provisioning (least loaded first) algorithm is used



Compatitor Task Offloading Algorithms

- **Random**: A random VM is selected
- **Mobile Device Utilization Heuristic**

```
If average mobile device CPU utilization < 75 execute task locally
Otherwise, offload to edge server
```

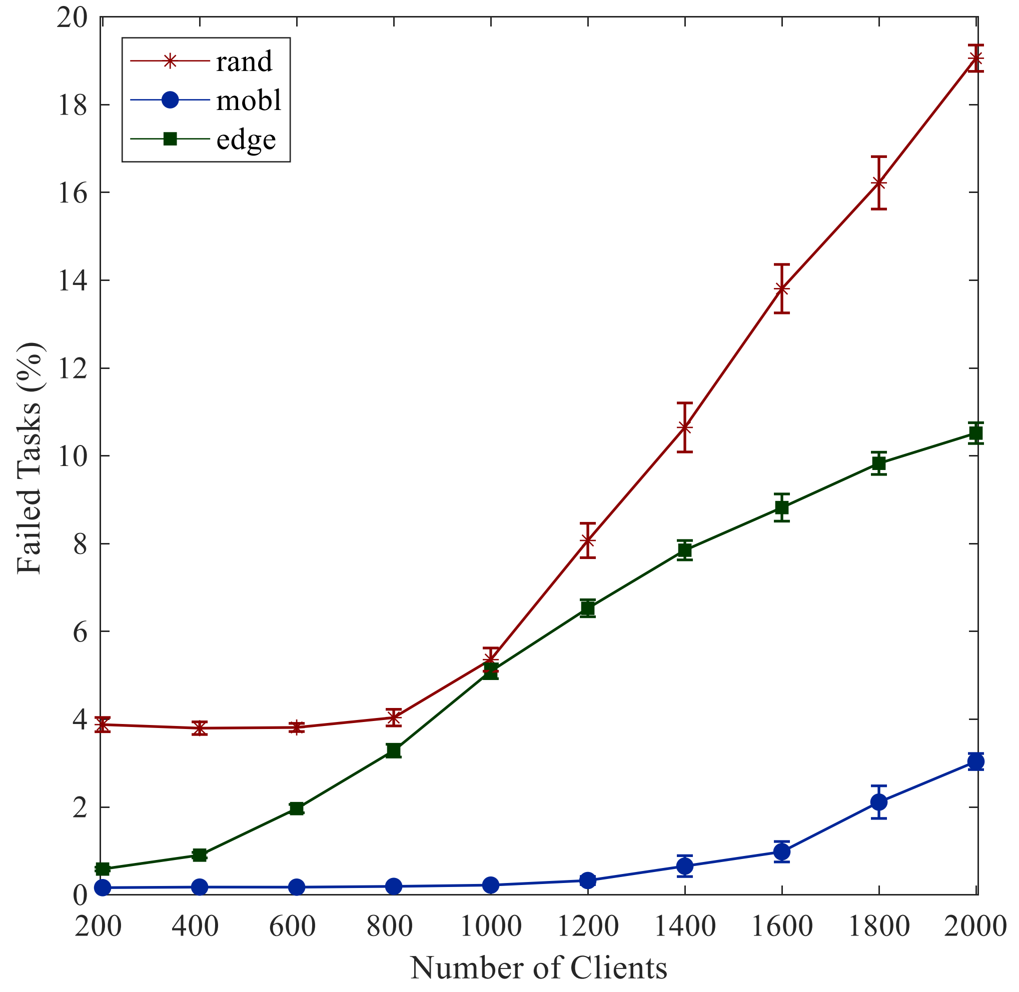
- **Edge Utilization Heuristic**

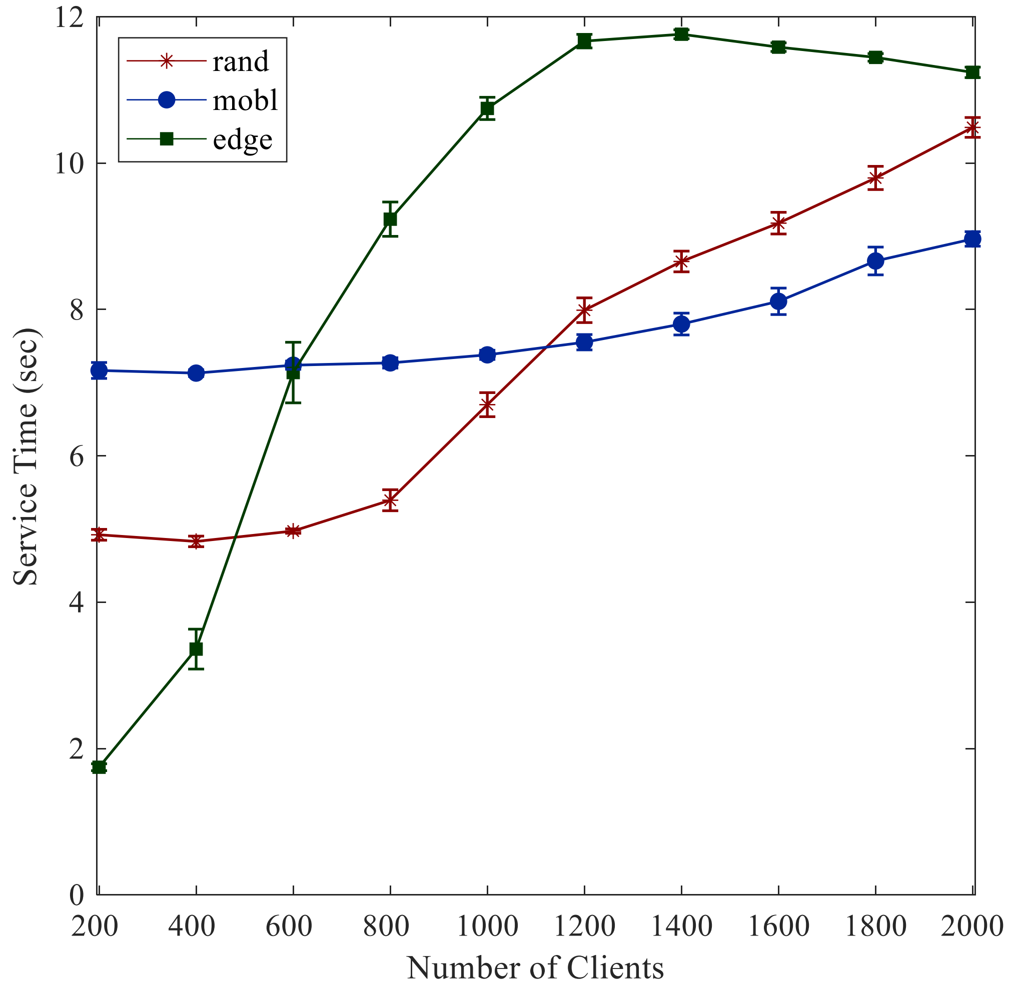
```
If average edge server CPU utilization < 90 offload task to edge server
Otherwise, execute task locally
```

Applications Used in This Simulation

Parameter	Aug. Reality	Health	Infotainment
Usage Percentage (%)	30	30	40
Task Interarrival (sec)	2	3	5
Active/Idle Period (sec)	40/20	60/30	30/30
VM Utilization on Edge/Client (%)	5/20	2/8	4/1
Task Length (GI)	20	8	16
Upload/Download Data (KB)	3000/1000	900/500	2000/4000

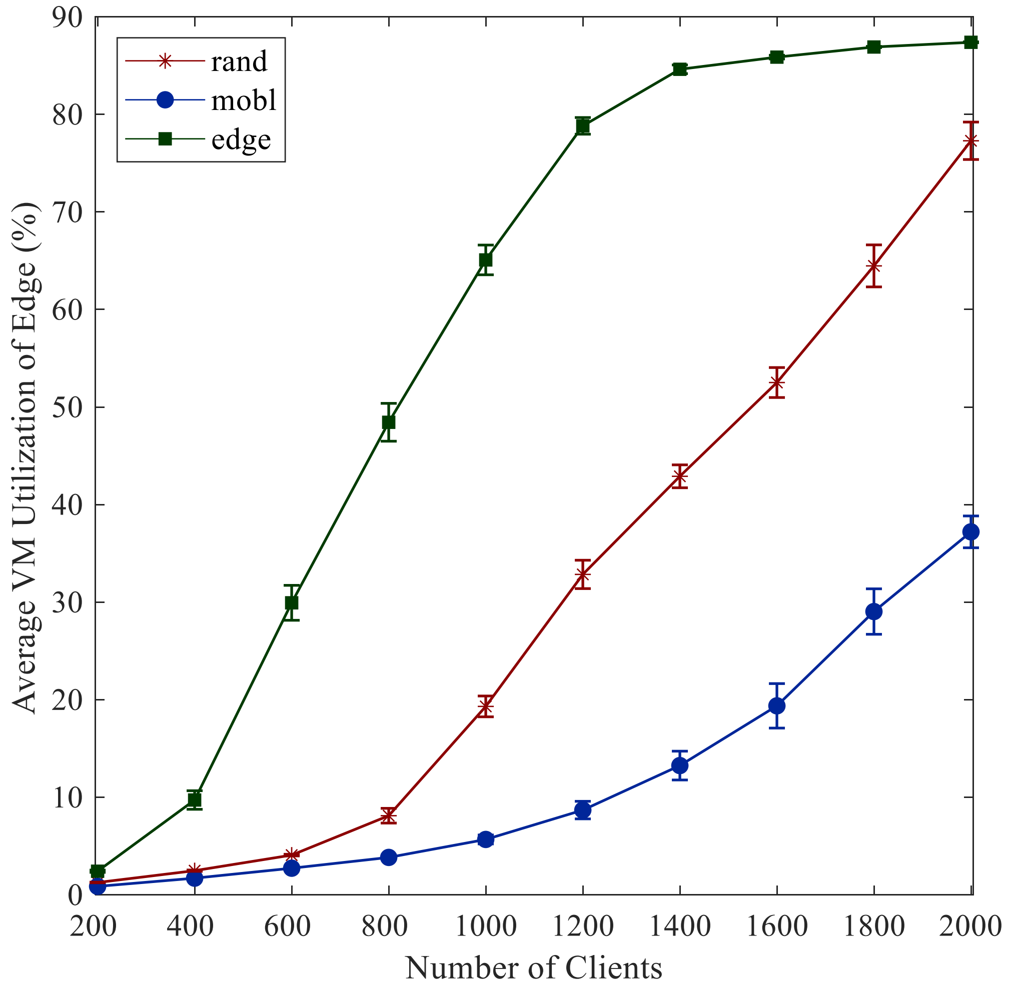
Important Simulation Results

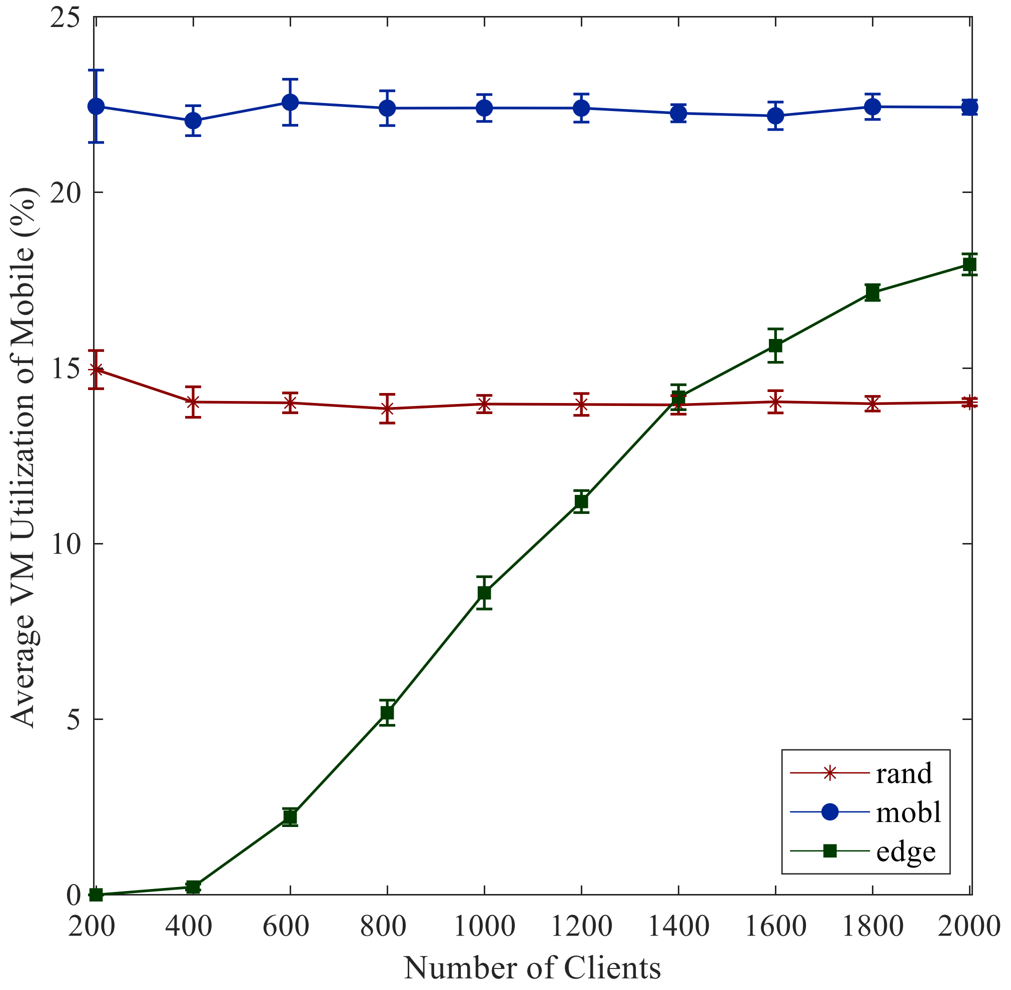




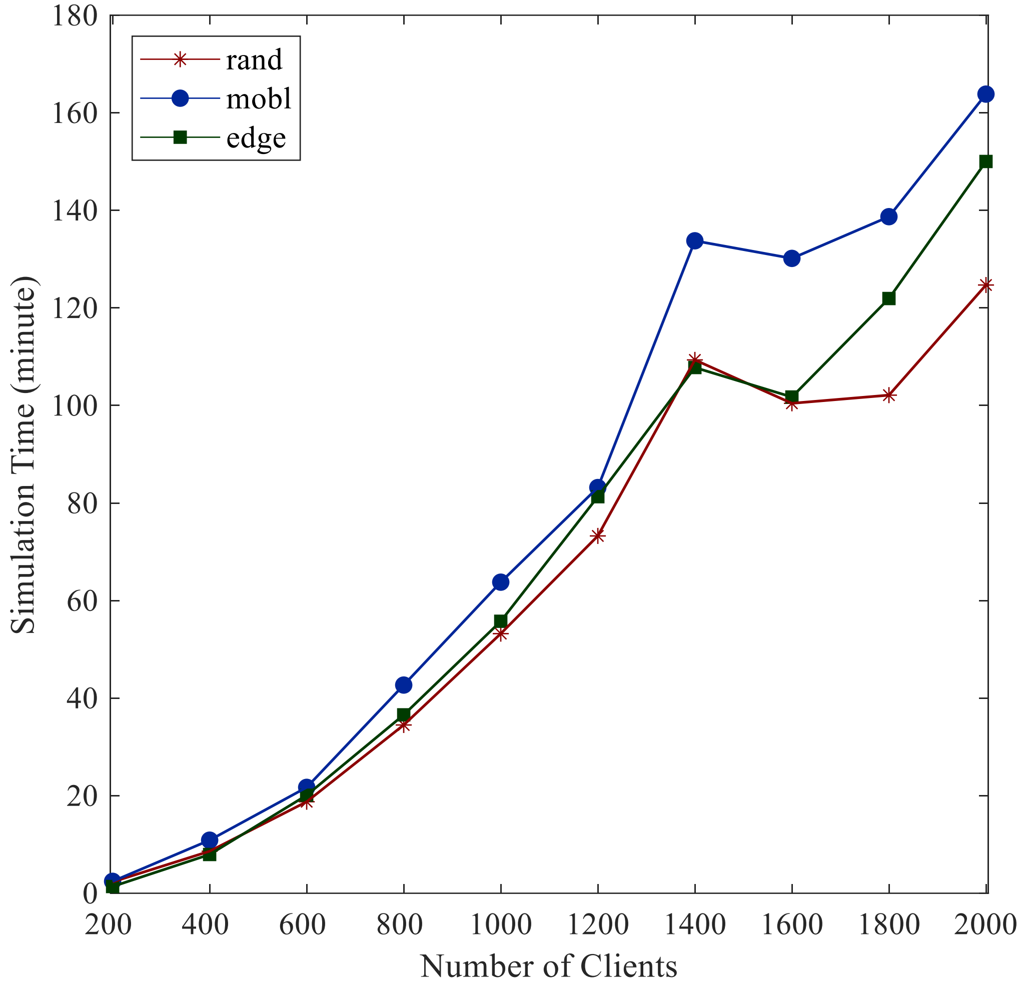
Graph is plotted with 95% confidence interval error bars

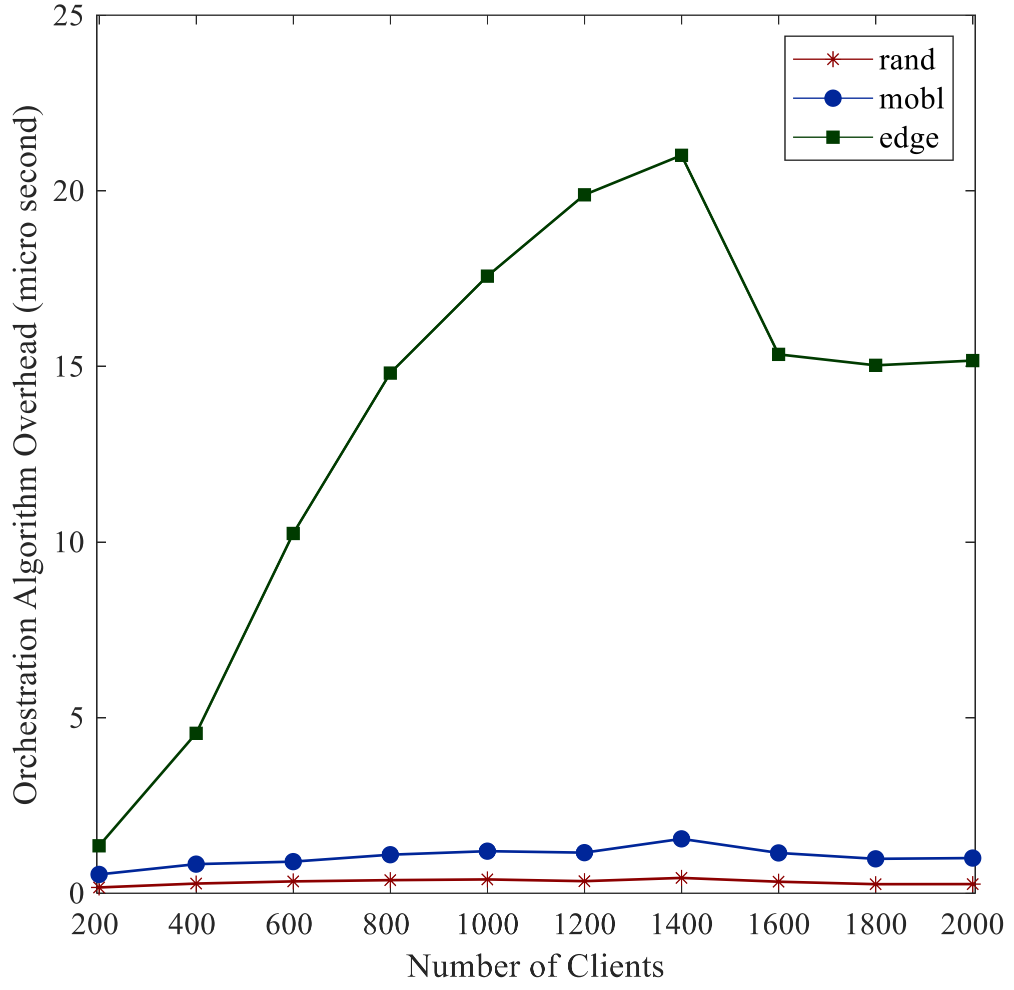
VM Utilization Analysis





Complexity Analysis





★ The results are highly dependent to host machine load while running the simulation!

Simulation Parameters

Parameter	Value
Simulation Time/Warm-up Period	15/1 minutes
Number of repetition	10
WLAN Delay Model	Empirical
MAN Delay	Fixed (5 ms)
Number of VM per Edge/Client Host	8/1
Number of Cores per Edge/Client VM CPU	2/1
VM Processor Speed per Edge/Client	10/4 GIPS
Mobility Model	Nomadic Mobility
Number of locations for Type 1/2/3 places	2/4/8
Mean waiting time in Type 1/2/3 places	10/6.6/3.3 minutes

Case Study 3 – Where to Offload

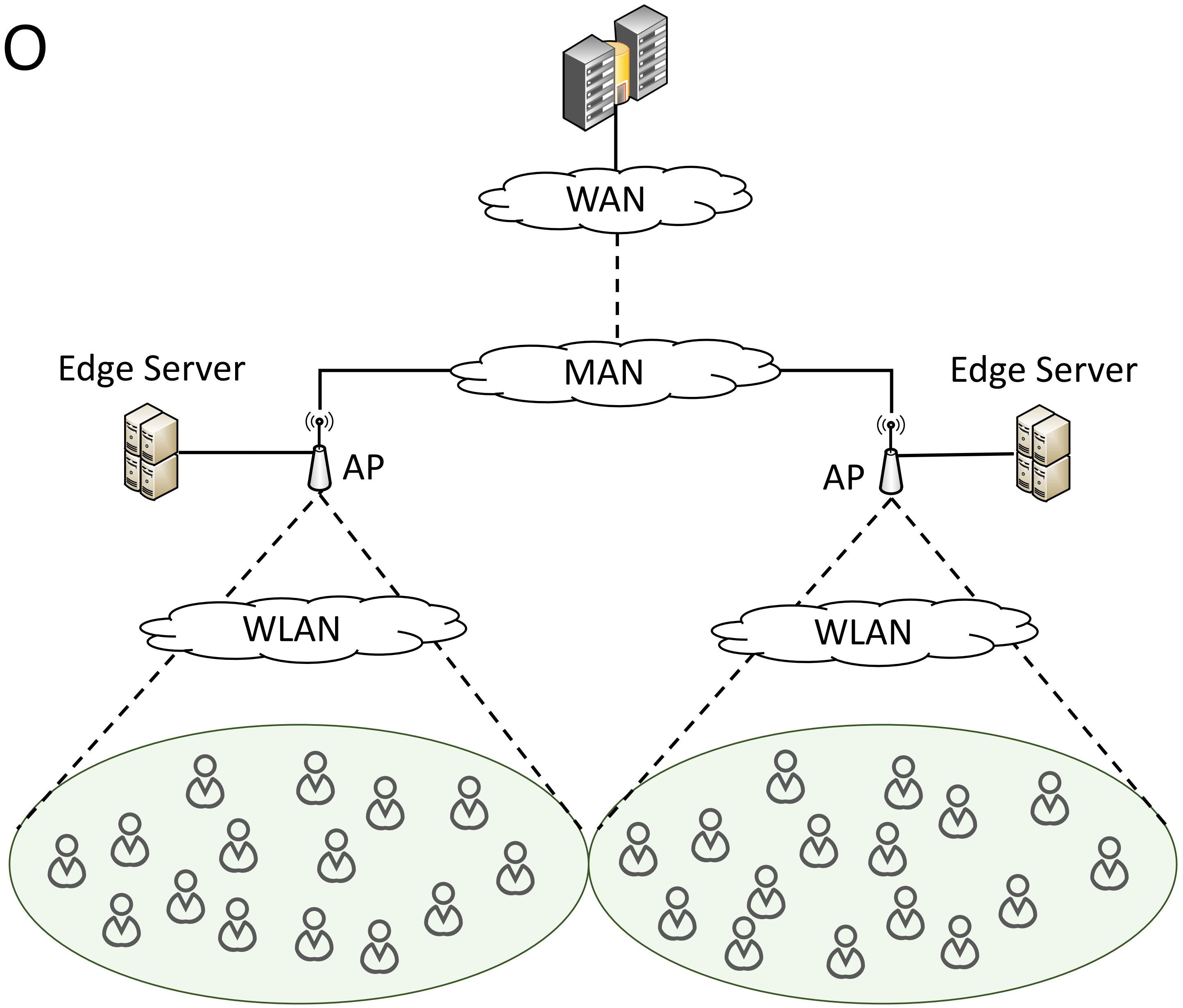
Performance Evaluation of Different Workload Orchestration Policies

Find the source code below:

https://github.com/CagataySonmez/PerformanceEvaluationWithEdgeCloudSim/tree/main/src/edu/boun/edgecloudsim/applications/scenario3

Simulation Scenario

- Mobile can offload tasks to edge or cloud servers
- Worst-fit VM provisioning (least loaded first) algorithm is used
- If the task is sent to another edge server outside the connected network, it is transmitted via MAN
- WLAN and WAN delays are modeled independently, so the WLAN is not affected if task is sent to remote server



Compatitor Edge Orchestration Algorithms

- **Random**: A random server is selected to offload task
- **Edge Server Utilization Heuristic**

```
If average edge servers CPU utilization > 75, offload task to cloud server
Otherwise, offload task to edge servers
```

- **Network Utilization Heuristic**

```
If WAN bandwidth > 5 Mbps, offload task to cloud server
Otherwise, offload task to edge servers
```

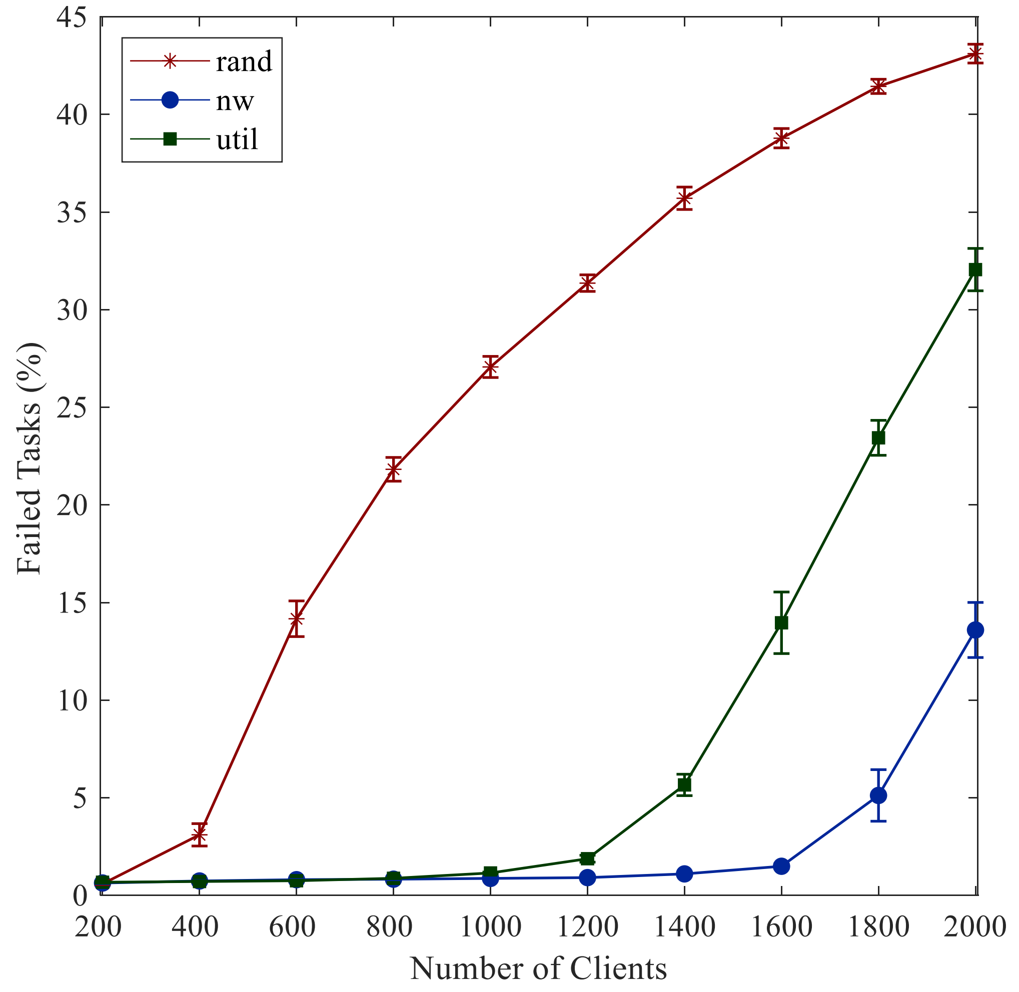
Applications Used in This Simulation

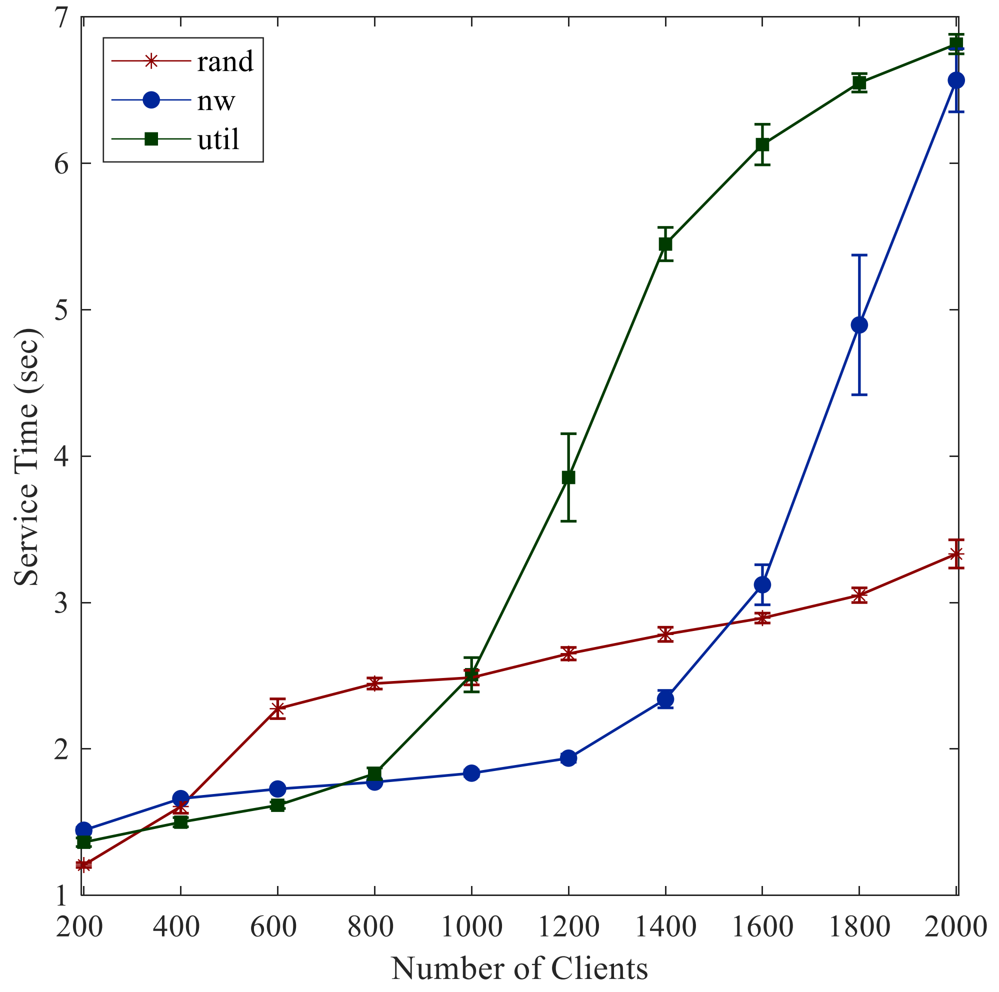
Parameter	Aug. Reality	Health	Infotainment
Usage Percentage (%)	30	20	50
Task Interarrival (sec)	2	3	7
Active/Idle Period (sec)	40/20	45/90	30/45
VM Utilization on Edge/Cloud (%)	6/0.6	2/0.2	10/1
Task Length (GI)	15	3	9
Upload/Download Data (KB)	1500/50	50/1250	250/1000

Simulation Parameters

Parameter	Value
Simulation Time/Warm-up Period	15/1 minutes
Number of repetition	10
WAN/WLAN Delay Model	Empirical
MAN Delay	Fixed (5 ms)
Number of VM per Edge/Cloud Host	8/4
Number of Cores per Edge/Cloud VM CPU	2/4
VM Processor Speed per Edge/Cloud	10/100 GIPS
Mobility Model	Nomadic Mobility
Number of Locations for Type 1/2/3 Places	2/4/8
Mean waiting time in Type 1/2/3 Places	8/5/2 minutes

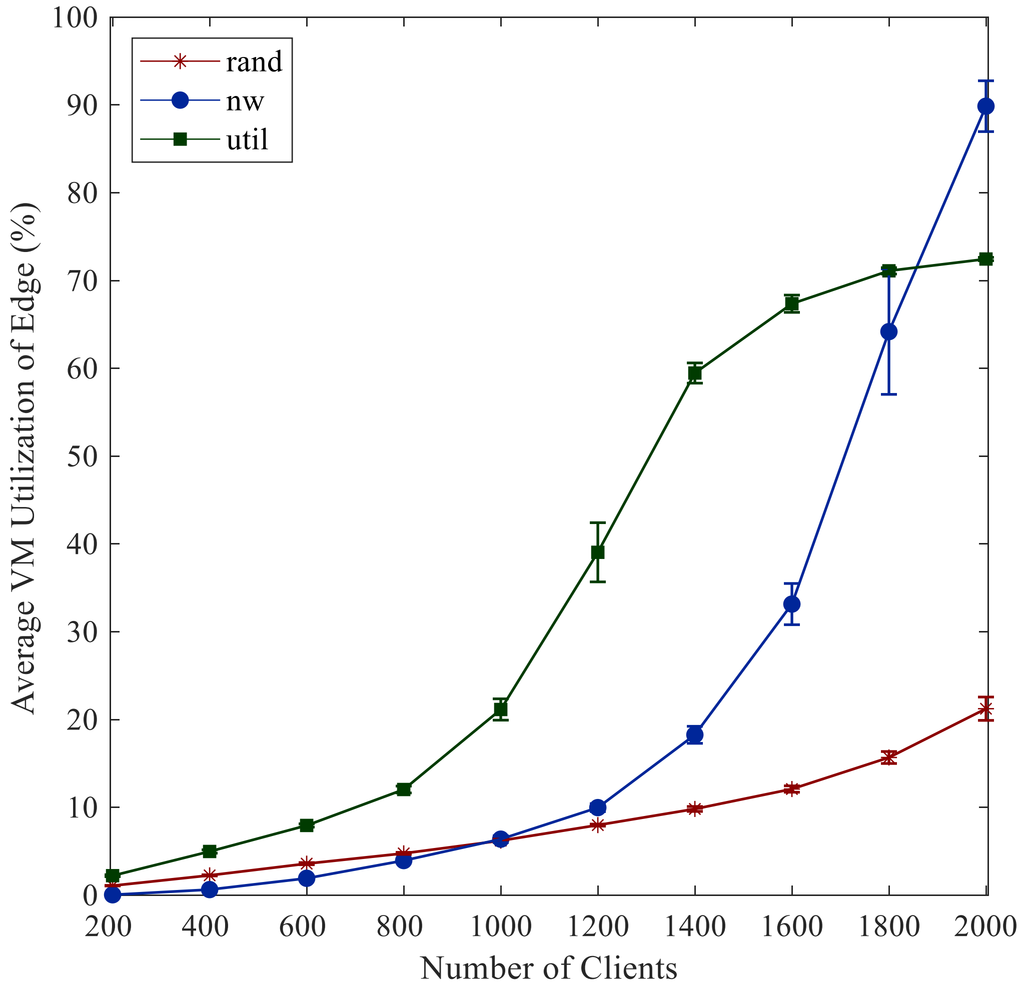
Important Simulation Results

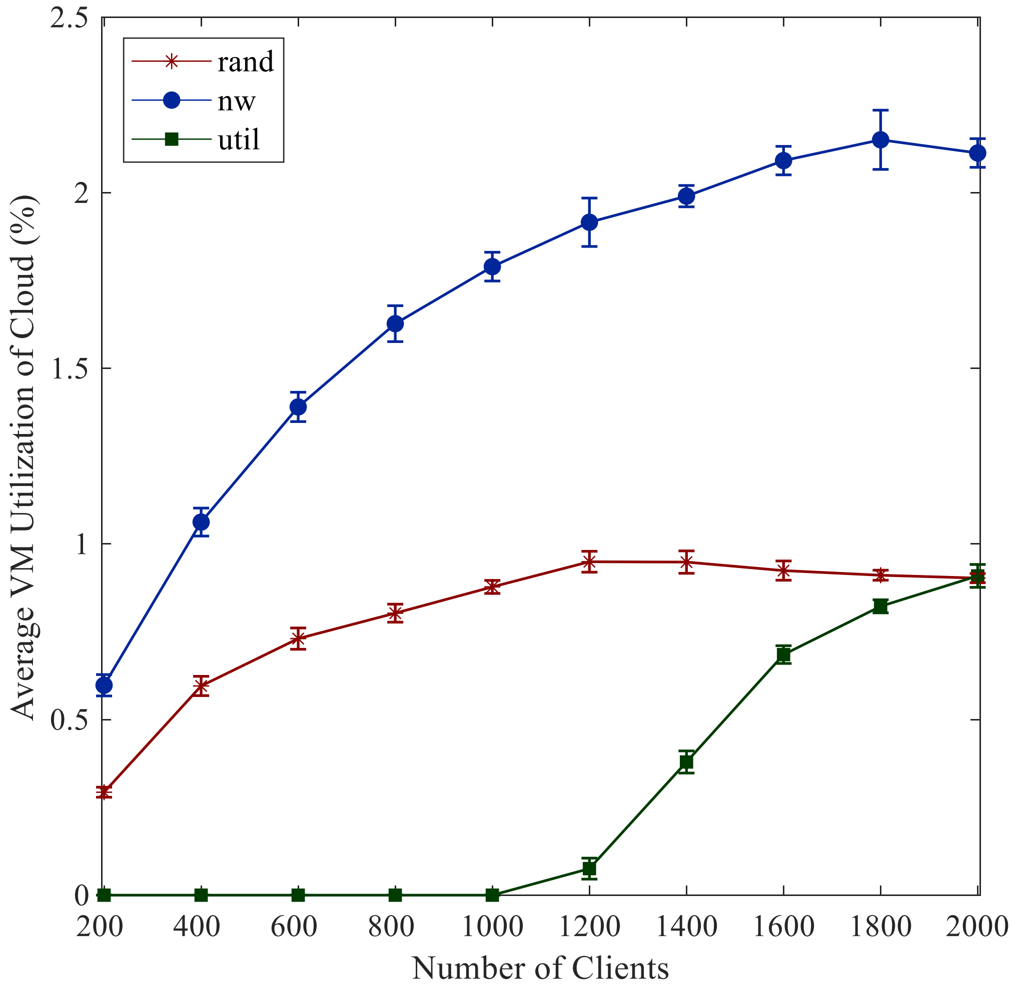




Graph is plotted with 95% confidence interval error bars

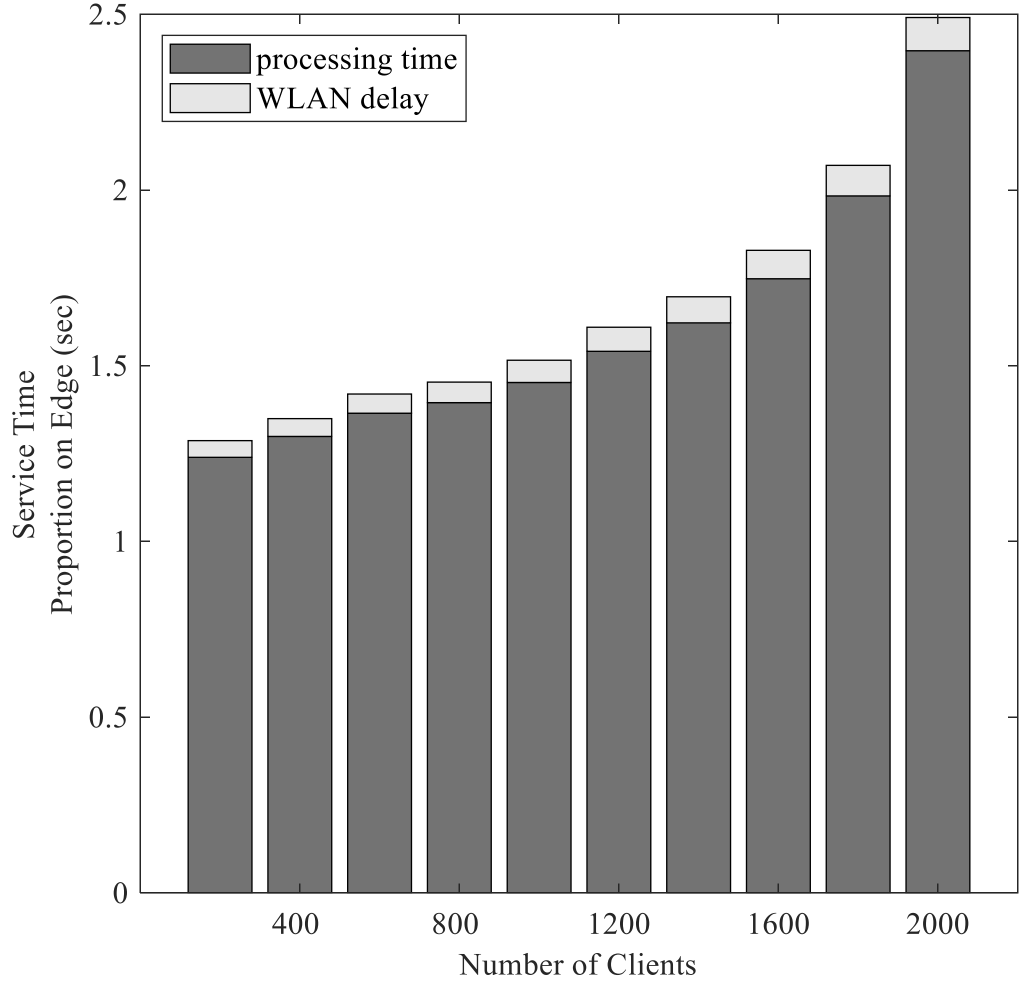
VM Utilization Analysis

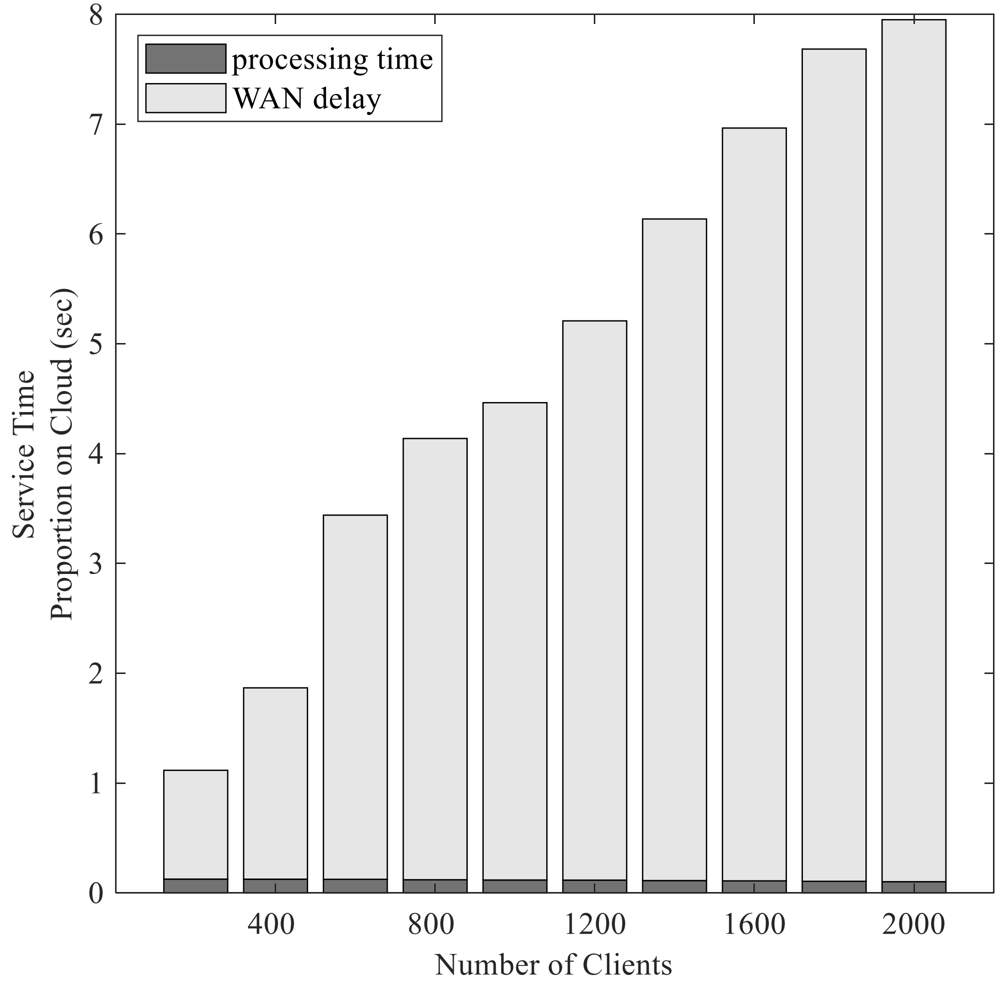




★ The cloud is mostly considered to be an endless pool of resources

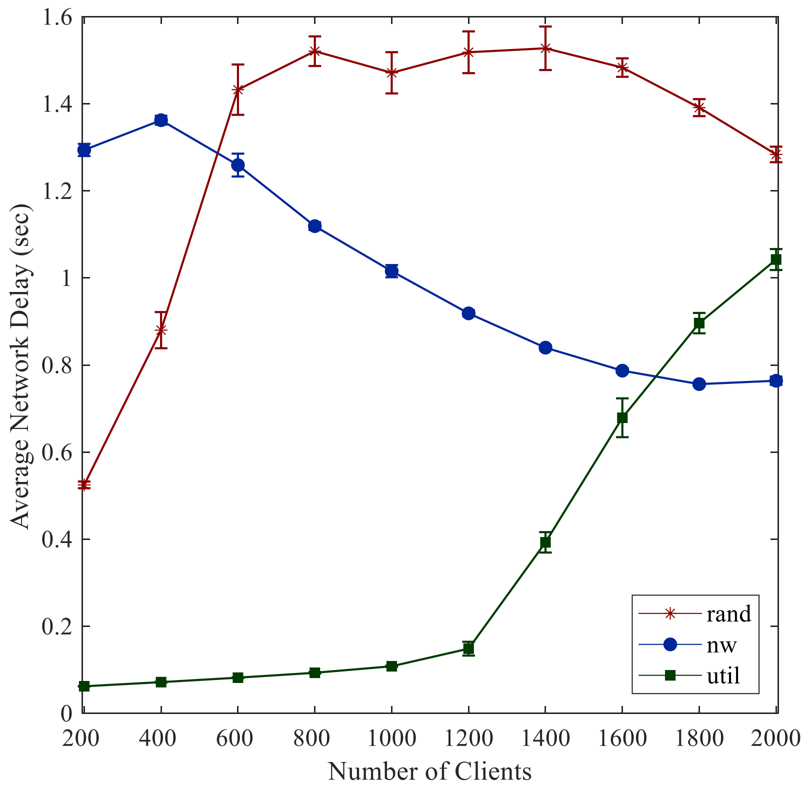
Service Time Analysis

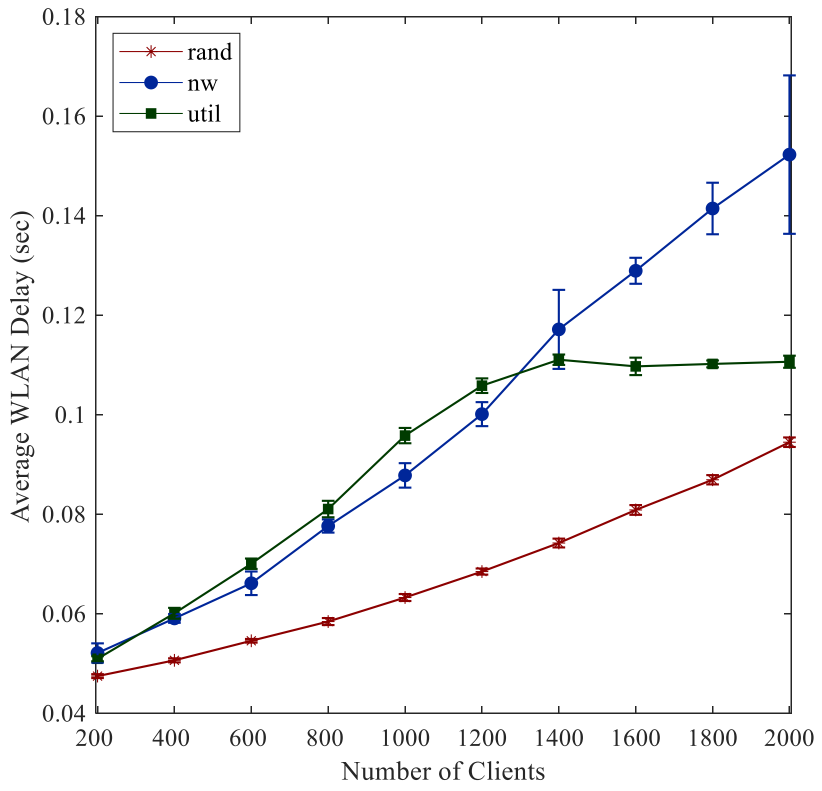


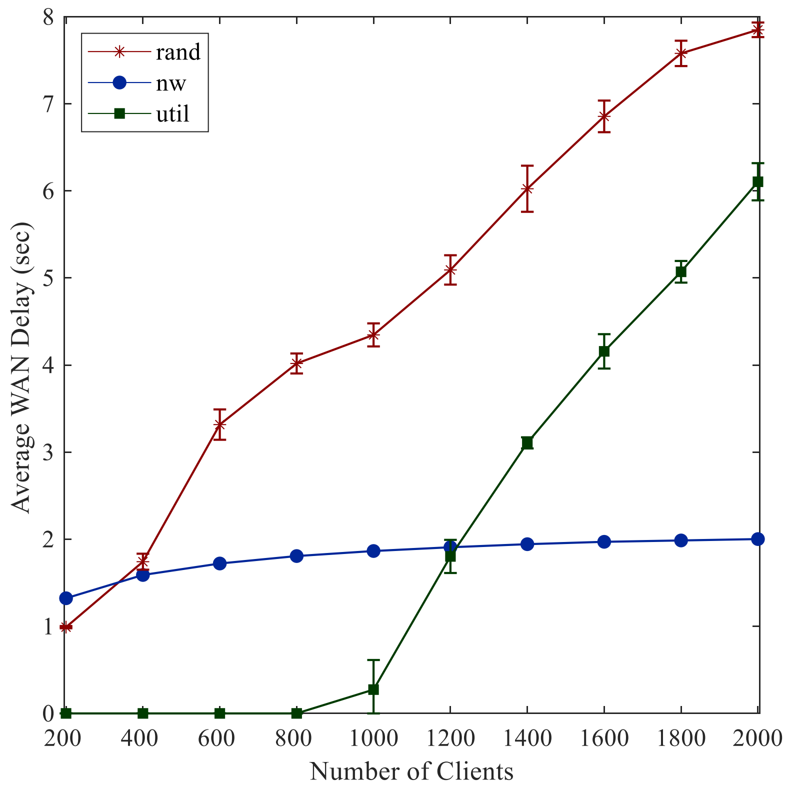


Processing time is a problem for the Edge and WAN delay for the Cloud servers

Network Delay Analysis







★ WAN delay dominates the average network delay

Case Study 4 – Server Capacity Planning

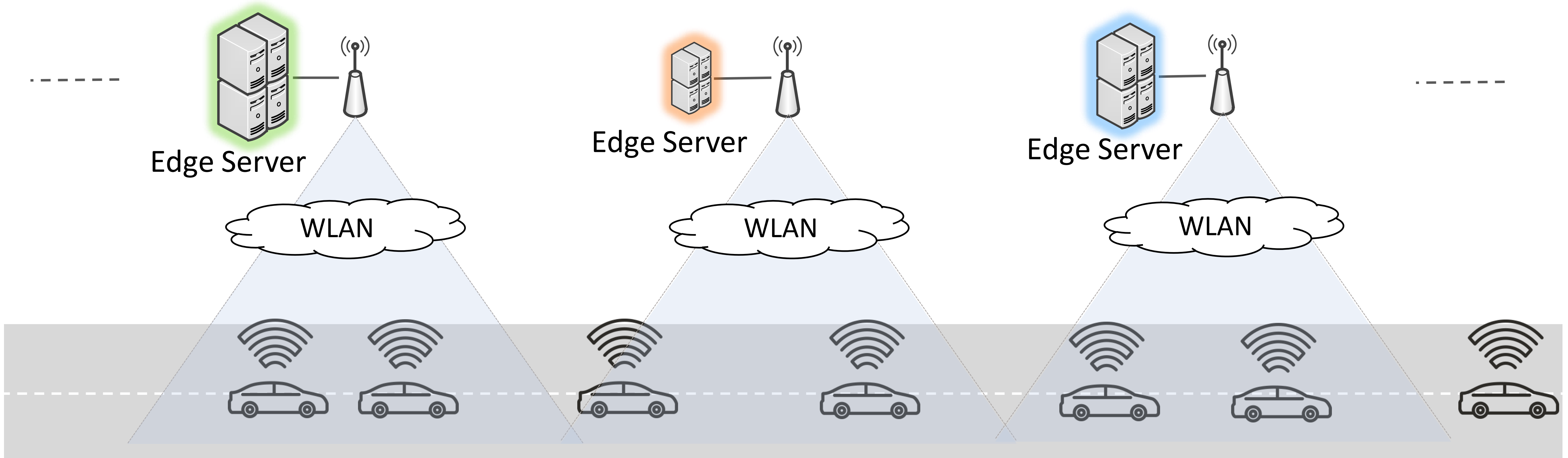
Performance Evaluation of Different Capacity Planning Approaches

Find the source code below:

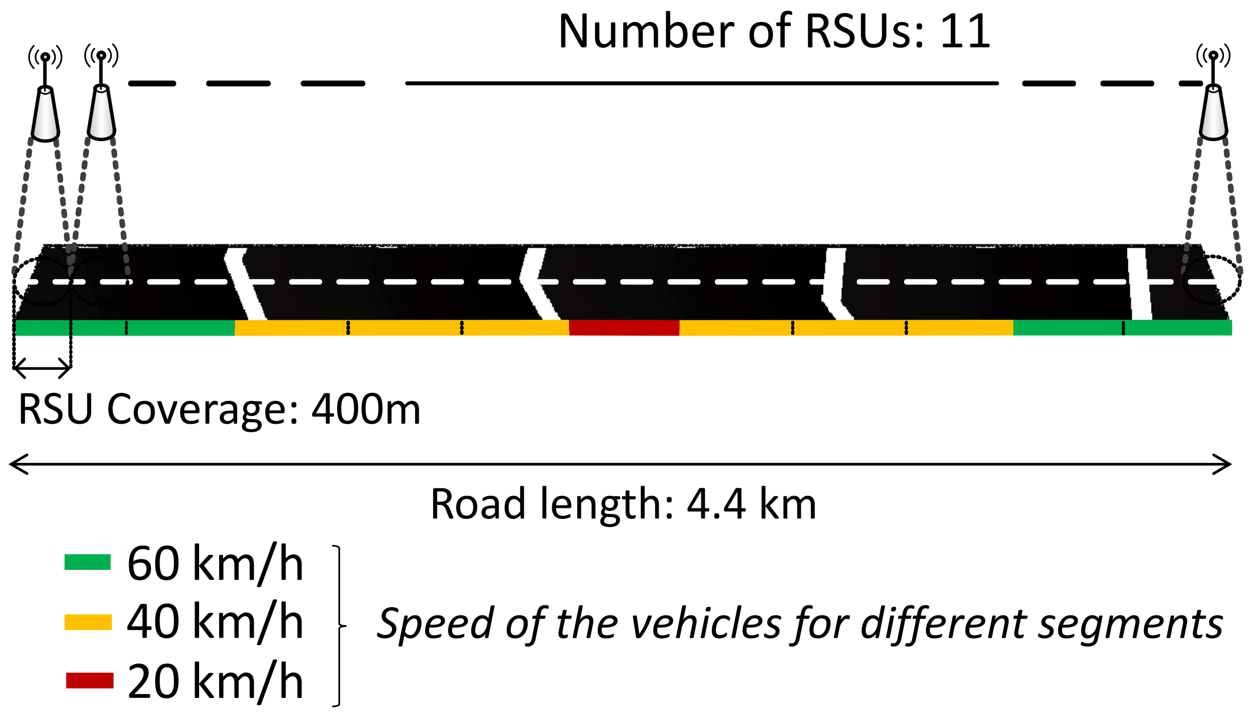
https://github.com/CagataySonmez/PerformanceEvaluationWithEdgeCloudSim/tree/main/src/edu/boun/edgecloudsim/applications/scenario4

Simulation Scenario

- Vehicles can only offload task to the edge servers connected to the serving access point
- Edge servers run host machines of varying capacity
- This scenario compares different edge server capacity planning algorithms



Vehicular Mobility Model



- A **smart highway** environment is simulated
- 1000 to 2000 vehicles traveling on a circular road
- Dynamic velocity values based on the vehicle position is used

Competitor Capacity Planning Algorithms

- RANDOM CAPACITY
  - Total capacity is randomly assigned (allocated) to hosts
  - Total capacity is 220 GIPS
- EQUAL CAPACITY
  - Total capacity is equally distributed to the hosts
  - 20 GIPS computing capacity for all hosts
- TRAFFIC DENSITY HEURISTIC
  - Total capacity is distributed to the hosts in proportion to the intensity of the traffic
  - 44 GIPS computing capacity for the hosts in high density areas
  - 20 GIPS computing capacity for the hosts in medium density areas
  - 14 GIPS computing capacity for the hosts in low density areas

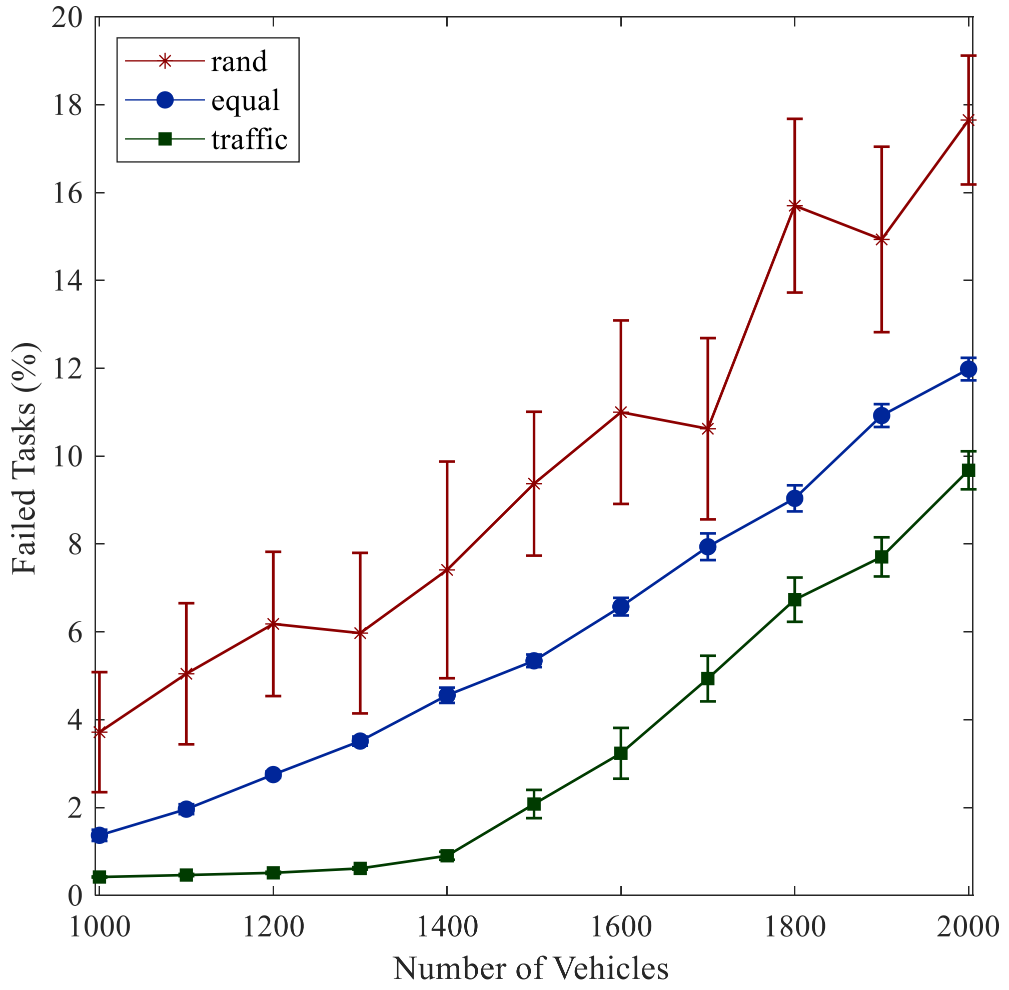
Applications Used in This Simulations

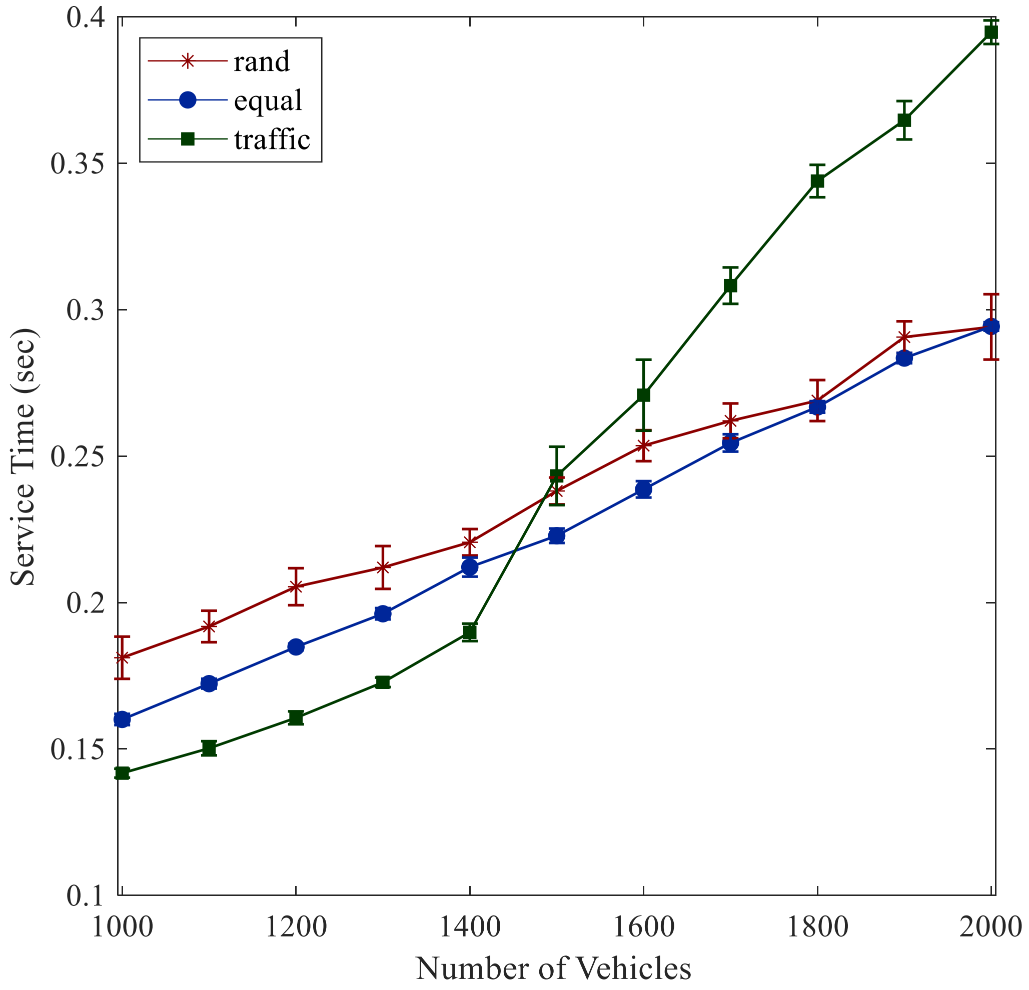
Parameter	Navigation App	Danger Assessment	Infotainment App
Usage Ratio (%)	50	25	25
Task Interarrival Time (sec)	3	5	15
Active/Idle Period (min)	always/0	always/0	always/0
Upload/Download Data (KB)	350/350	500/350	350/500
Task Length (MI)	600	1000	1600
RSU/Cloud VM Utilization(%)	6/1.6	20/4	40/8

Simulation Parameters

Parameter	Value
Simulation Time/Warm-up Period	15/1 minutes
Number of repetition	10
WLAN Delay Model	Empirical
MAN Delay	Fixed (10 ms)
Number of VM per Edge Host	2
Number of Cores per Edge VM CPU	2
VM Processor Speed per Edge	10-44 GIPS
Mobility Model	Vehicular Mobility
Number of locations for Type 1/2/3 Places	1/4/6
Speed of Vehicles in Type 1/2/3 Places	20/40/60 km/hour

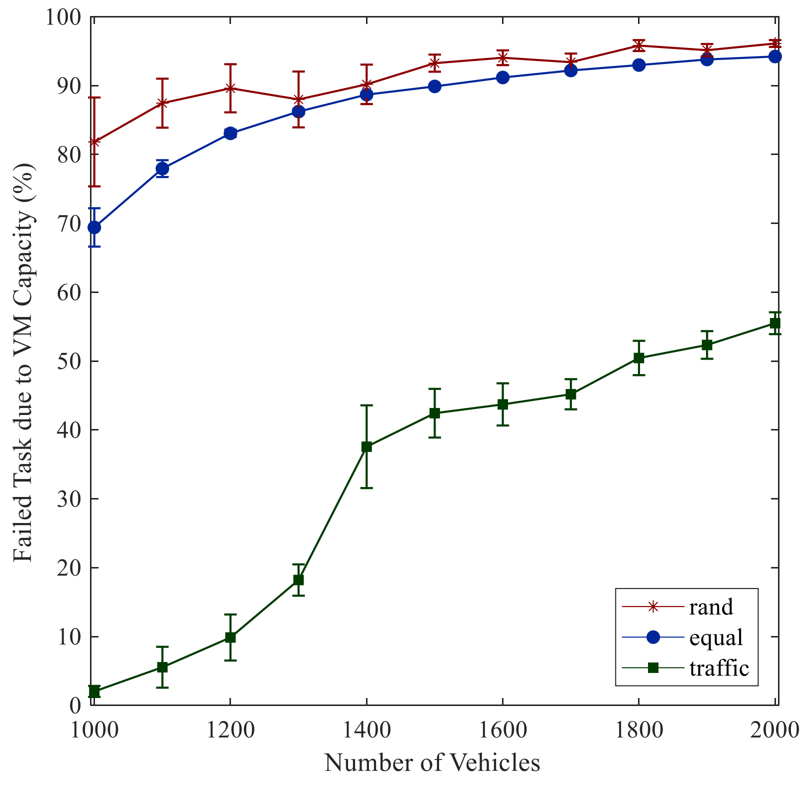
Important Simulation Results

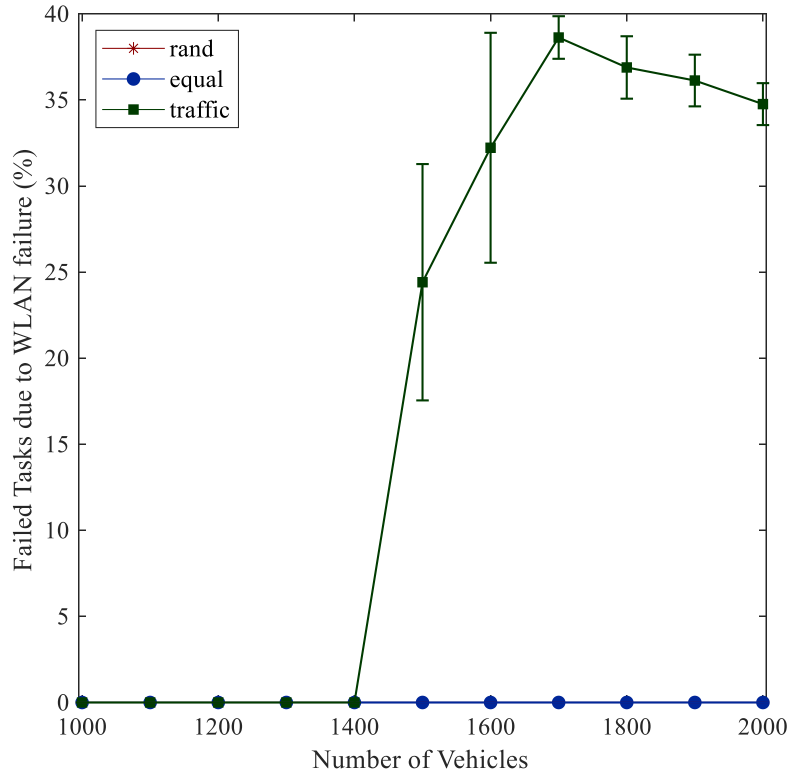


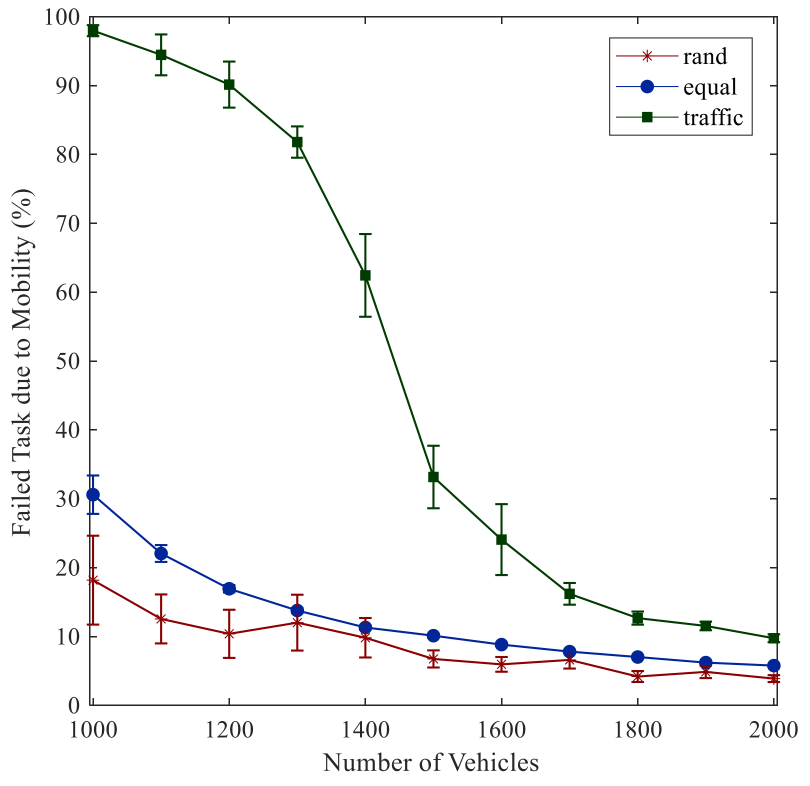


Graph is plotted with 95% confidence interval error bars

Task Failure Reason Analysis

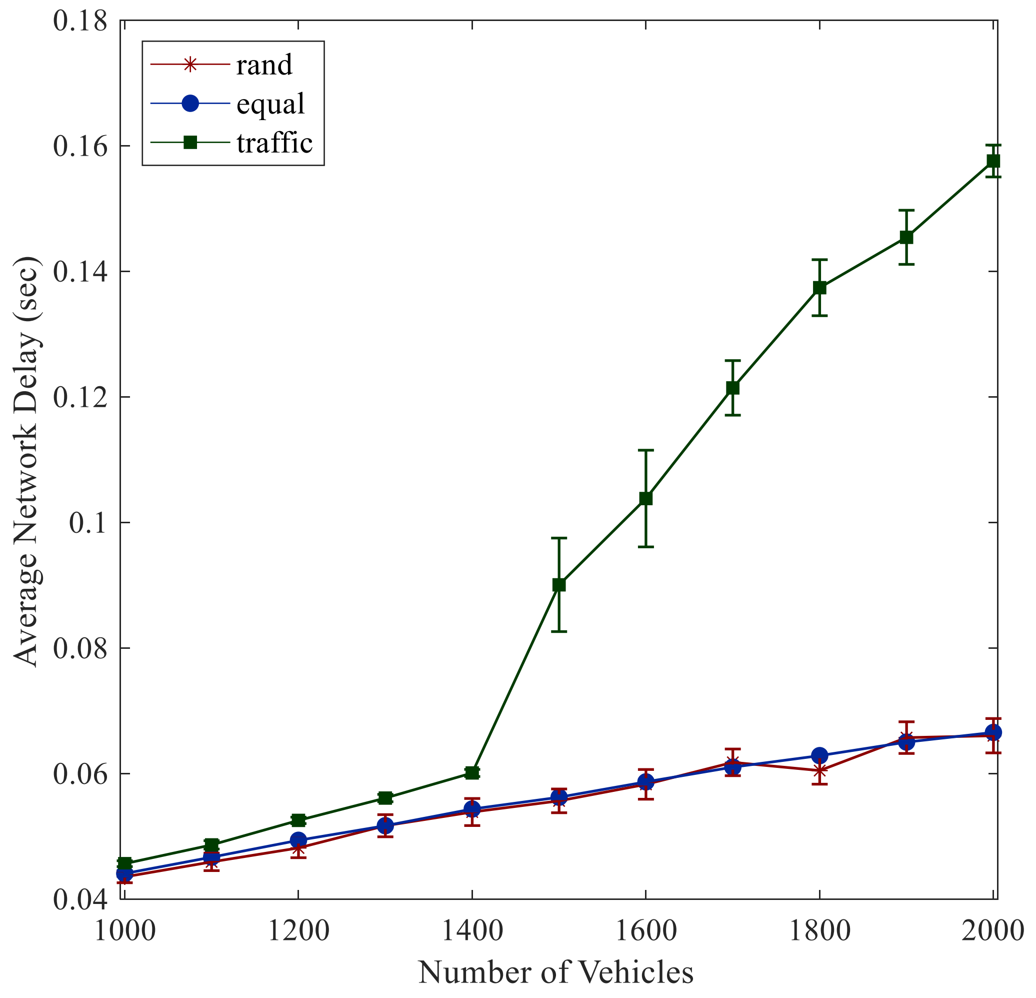


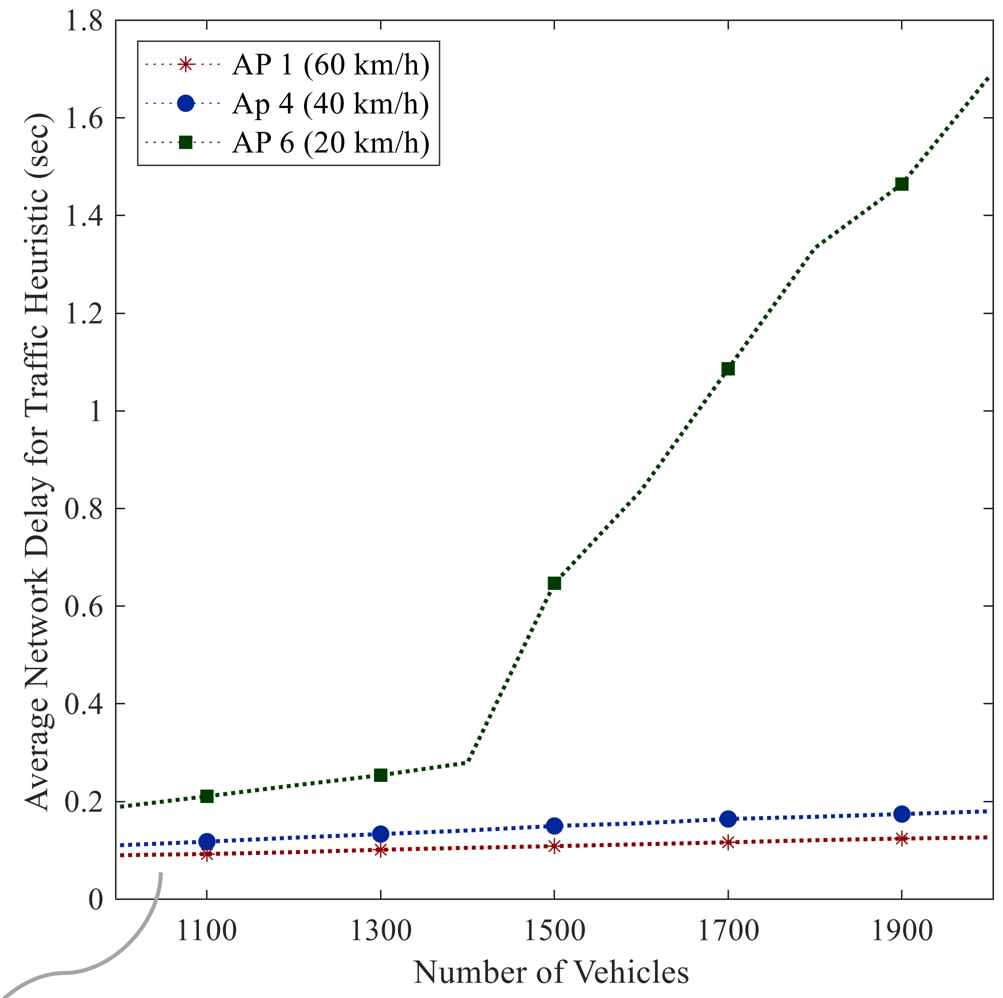




Failed tasks due to mobility dominates the failure reasons

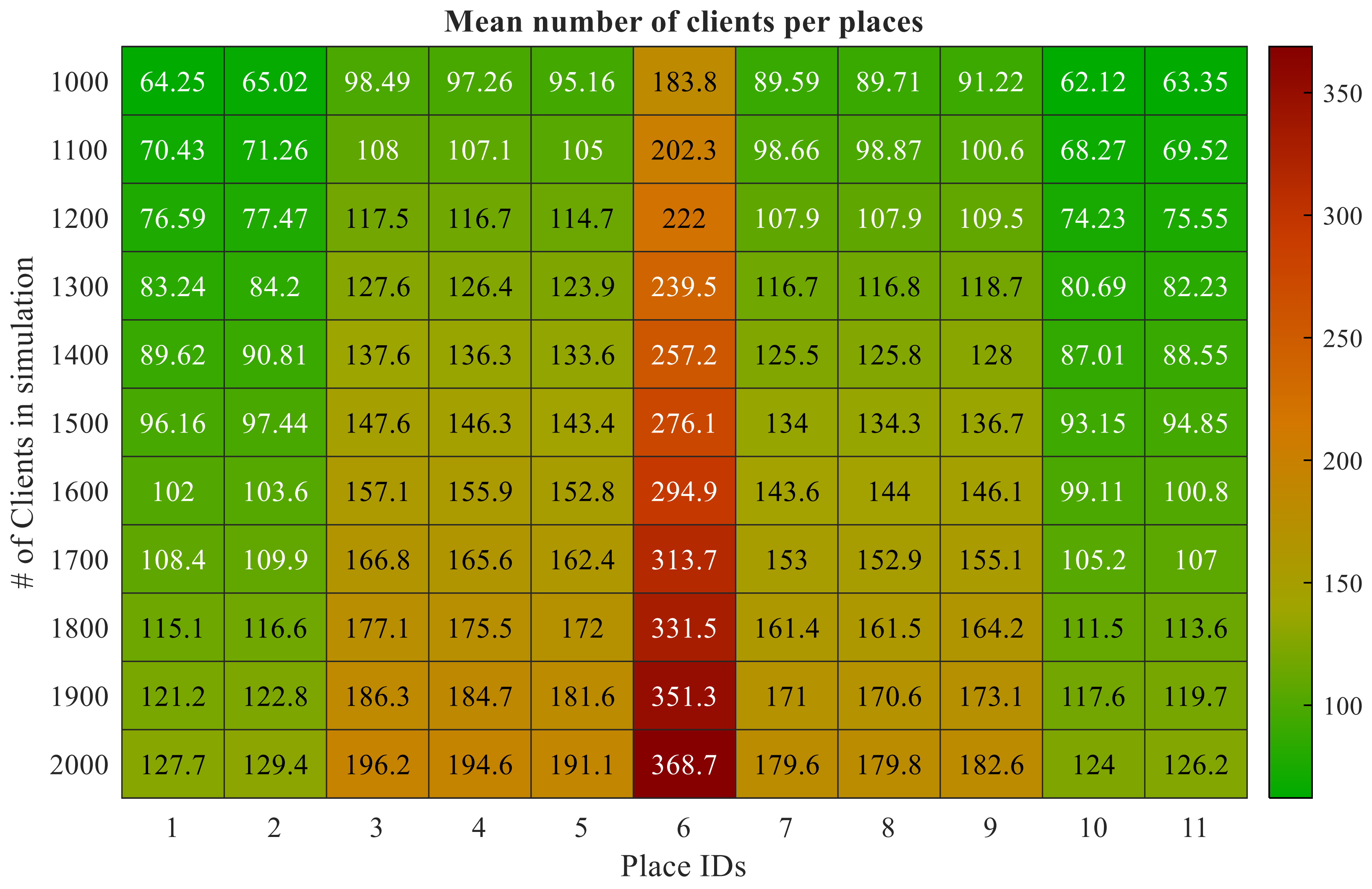
Network Delay Analysis





Network delay values of different access points

Mobility Model Verification



# of Clients in simulation	1	2	3	4	5	6	7	8	9	10	11
1000	64.25	65.02	98.49	97.26	95.16	183.8	89.59	89.71	91.22	62.12	63.35
1100	70.43	71.26	108	107.1	105	202.3	98.66	98.87	100.6	68.27	69.52
1200	76.59	77.47	117.5	116.7	114.7	222	107.9	107.9	109.5	74.23	75.55
1300	83.24	84.2	127.6	126.4	123.9	239.5	116.7	116.8	118.7	80.69	82.23
1400	89.62	90.81	137.6	136.3	133.6	257.2	125.5	125.8	128	87.01	88.55
1500	96.16	97.44	147.6	146.3	143.4	276.1	134	134.3	136.7	93.15	94.85
1600	102	103.6	157.1	155.9	152.8	294.9	143.6	144	146.1	99.11	100.8
1700	108.4	109.9	166.8	165.6	162.4	313.7	153	152.9	155.1	105.2	107
1800	115.1	116.6	177.1	175.5	172	331.5	161.4	161.5	164.2	111.5	113.6
1900	121.2	122.8	186.3	184.7	181.6	351.3	171	170.6	173.1	117.6	119.7
2000	127.7	129.4	196.2	194.6	191.1	368.7	179.6	179.8	182.6	124	126.2

Case Study 5 – Network & Server Capacity P.

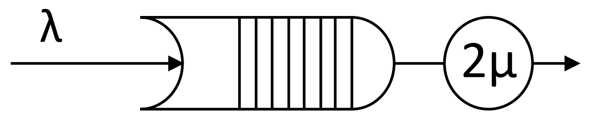
Performance Evaluation of Different Capacity Planning Approaches

Find the source code below:

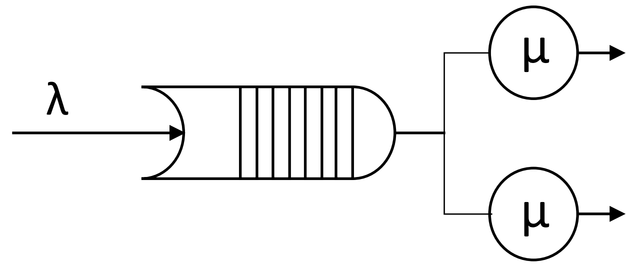
https://github.com/CagatavSonmez/PerformanceEvaluationWithEdgeCloudSim/tree/main/src/edu/boun/edgecloudsim/applications/scenario5

Which One Provides the Best Network Delay?

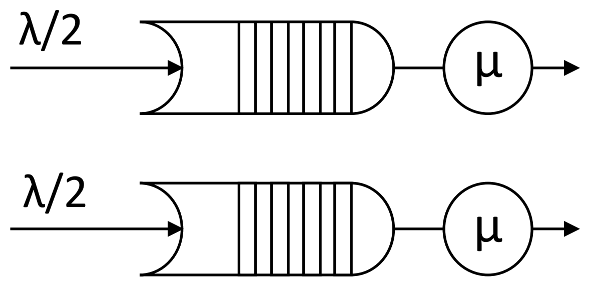
Case 1



Case 2

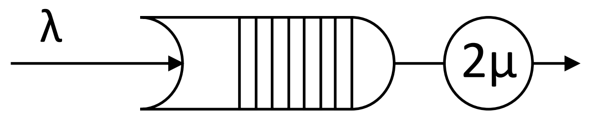


Case 3



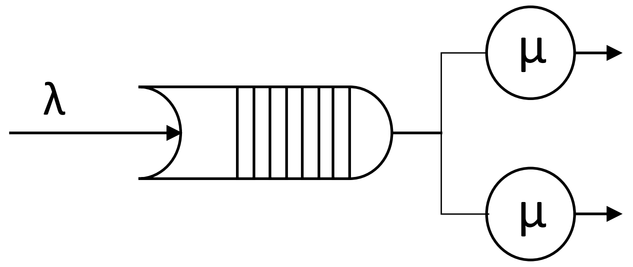
Which One Provides the Best Network Delay?

Case 1



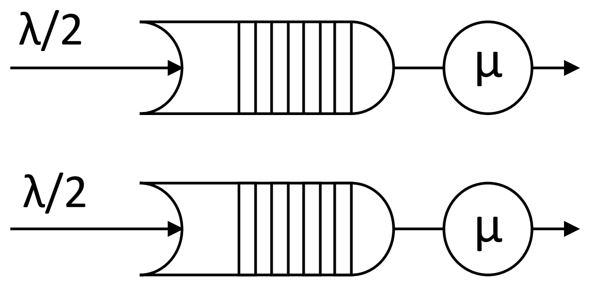
M/M/1 Queue

Case 2



M/M/2 Queue

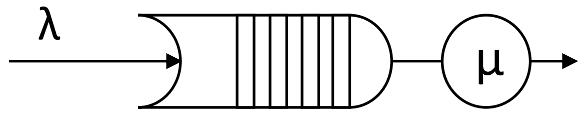
Case 3



Parallel M/M/1 Queues

Case 1: M/M/1 Queue

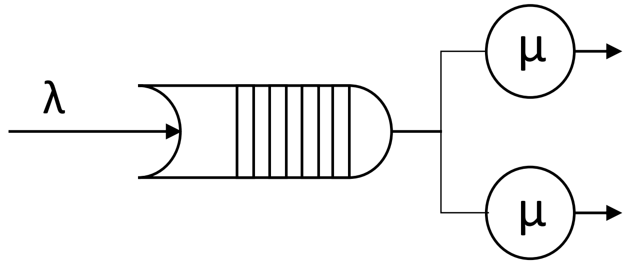
- Arrivals occur at rate λ according to a Poisson process
- Service times have an exponential distribution with rate parameter μ
- A **single** server serving with first-come first-served discipline



- Response time $$E(T) = \frac{1}{\underset{\text{capacity/packet length (p/s)}}{\underset{\swarrow}{\mu}} - \underset{\text{rate of the traffic (p/s)}}{\underset{\searrow}{\lambda}}} \quad , \mu > \lambda$$

Case 2: M/M/2 Queue

- Arrivals occur at rate λ according to a Poisson process
- Service times have an exponential distribution with rate parameter μ
- A single queue with **multiple** servers



- Response time can be calculated with birth-death process model

Case 2: M/M/2 Queue cont.

$$\begin{tikzcd}
0 \arrow[r, bend left, "\lambda"] & 1 \arrow[l, bend left, "\mu"] \arrow[r, bend left, "\lambda"] & 2 \arrow[l, bend left, "2\mu"] \arrow[r, bend left, "\lambda"] & 3 \arrow[l, bend left, "2\mu"] \arrow[r, bend left, "\lambda"] & {} \arrow[l, bend left, "2\mu"]
\end{tikzcd}$$

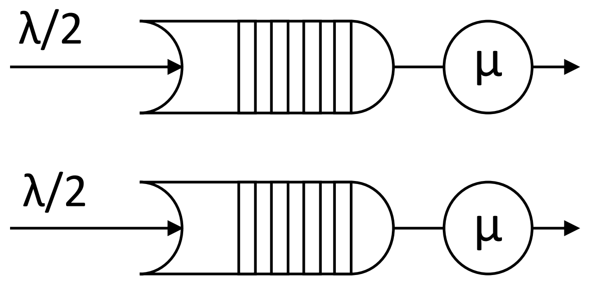
$$P_0 \lambda = P_1 \mu$$

$$P_1 (\lambda + \mu) = P_0 \lambda + P_2 2\mu \qquad \xrightarrow{\text{After some math...}} \qquad E(T) = \frac{4\mu}{(2\mu - \lambda)(2\mu + \lambda)}$$

$$P_n (\lambda + 2\mu) = P_{n-1} \lambda + P_{n+1} 2\mu$$

$$\Sigma P_n = 1$$

Case 3: Two Parallel M/M/1 Queues

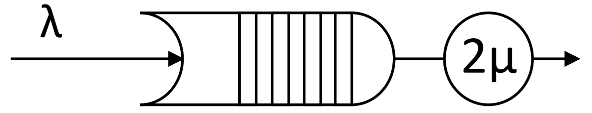


$$E(T) = \frac{1}{\mu - \lambda/2} \times 2$$

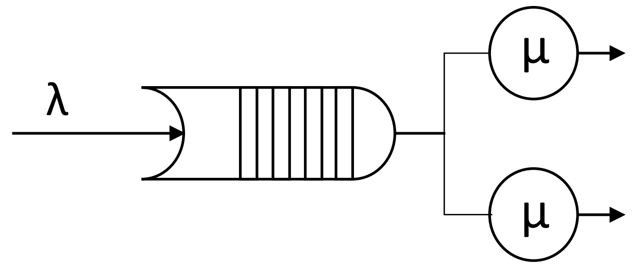
$$E(T) = \frac{4}{2\mu - \lambda}$$

Expected Network Delay for All Cases

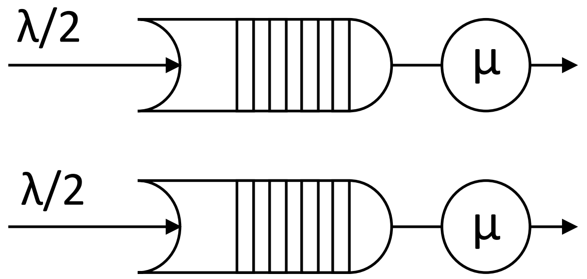
Case 1



Case 2

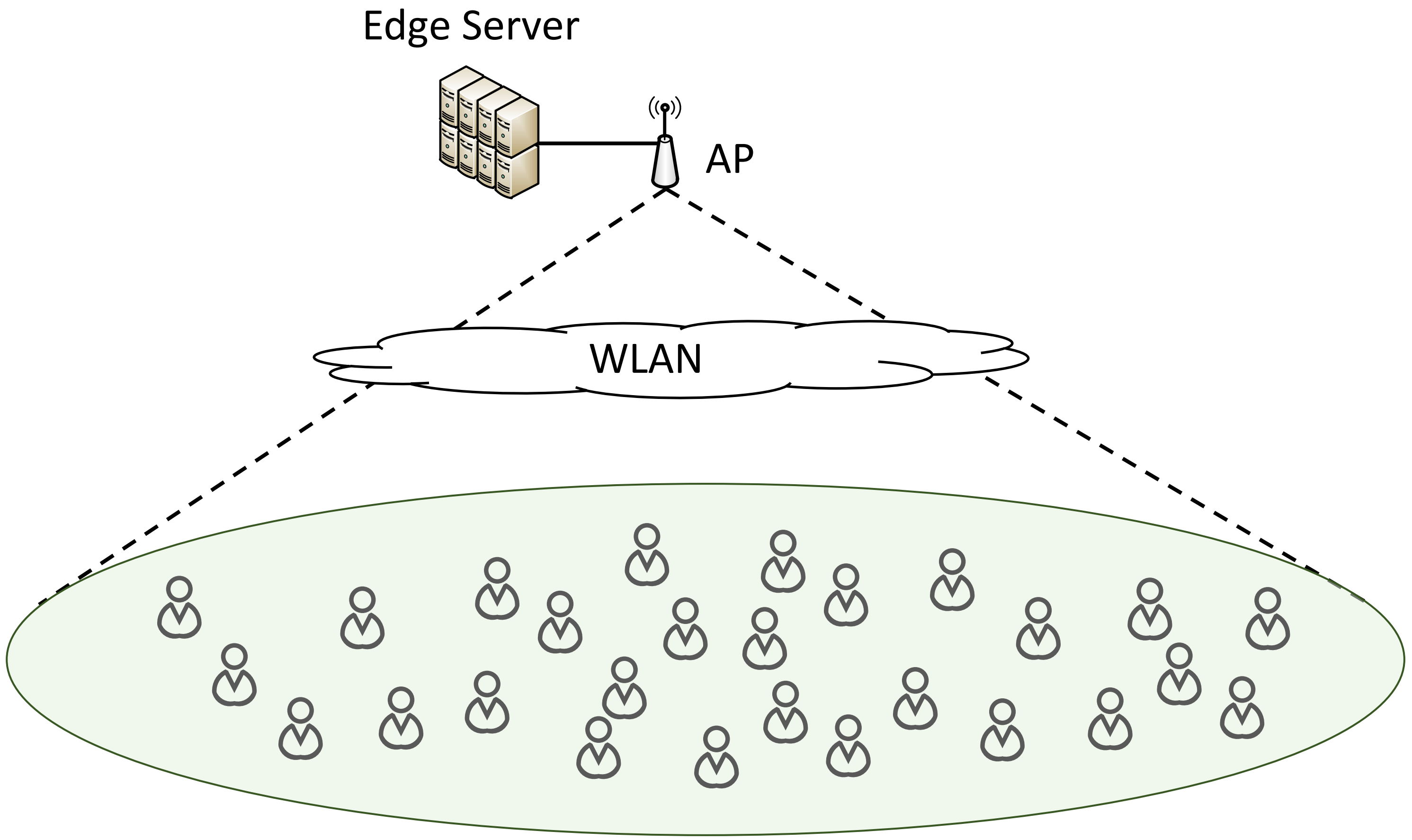


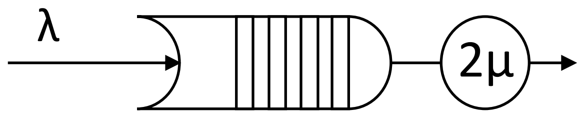
Case 3



$$E(T) = \frac{1}{2\mu - \lambda} \quad < \quad E(T) = \frac{4\mu}{(2\mu - \lambda)\,(2\mu + \lambda)} \quad < \quad E(T) = \frac{4}{2\mu - \lambda}$$

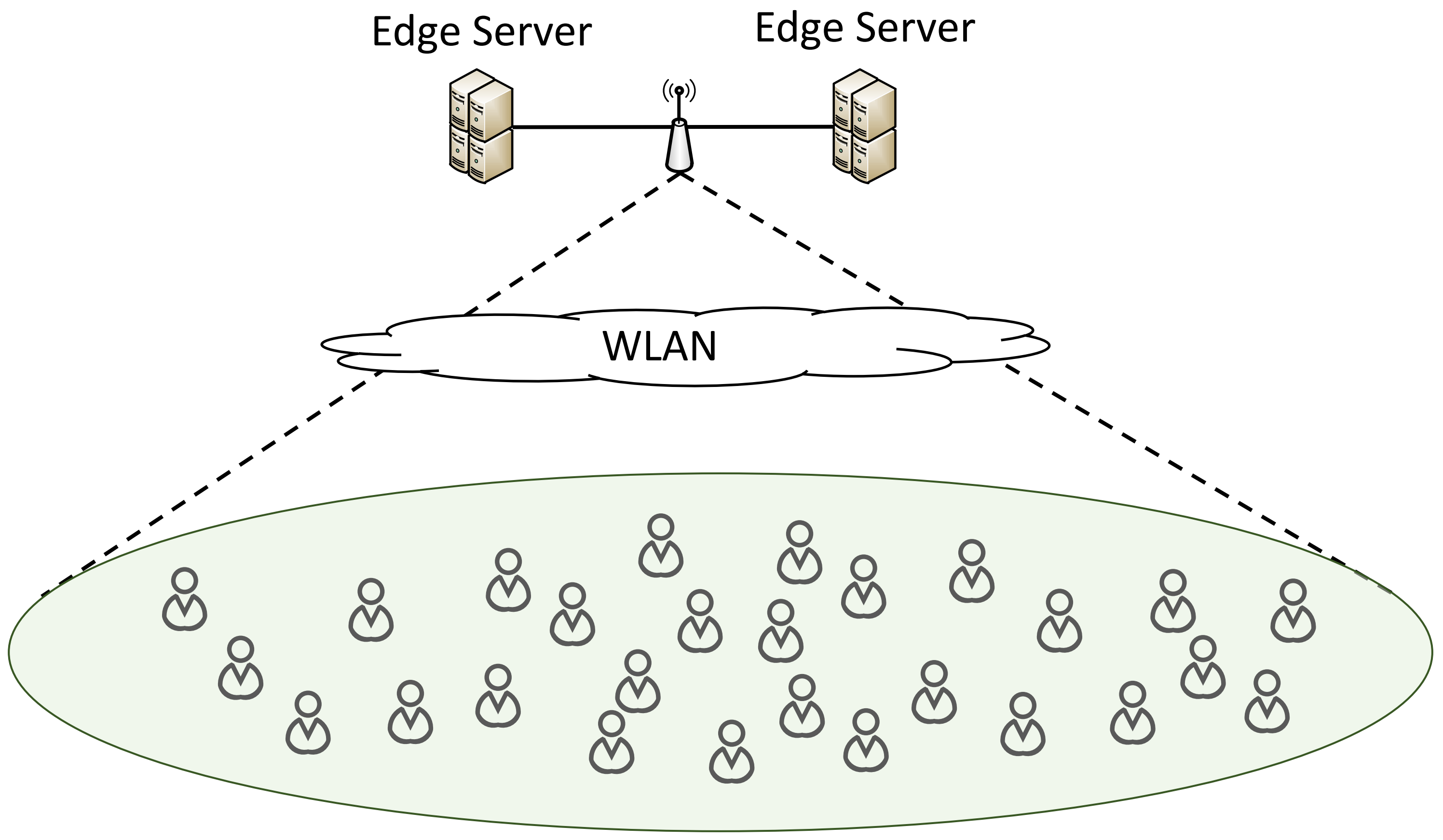
Implementation of Case1

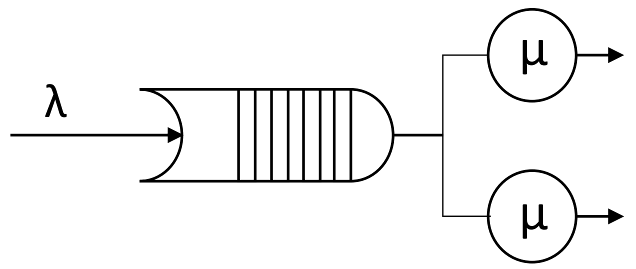




Parameter	Value
WLAN Bandwidth	100 Mbps
Number of Core Edge Server	4
Capacity of Edge Server	20 GIPS

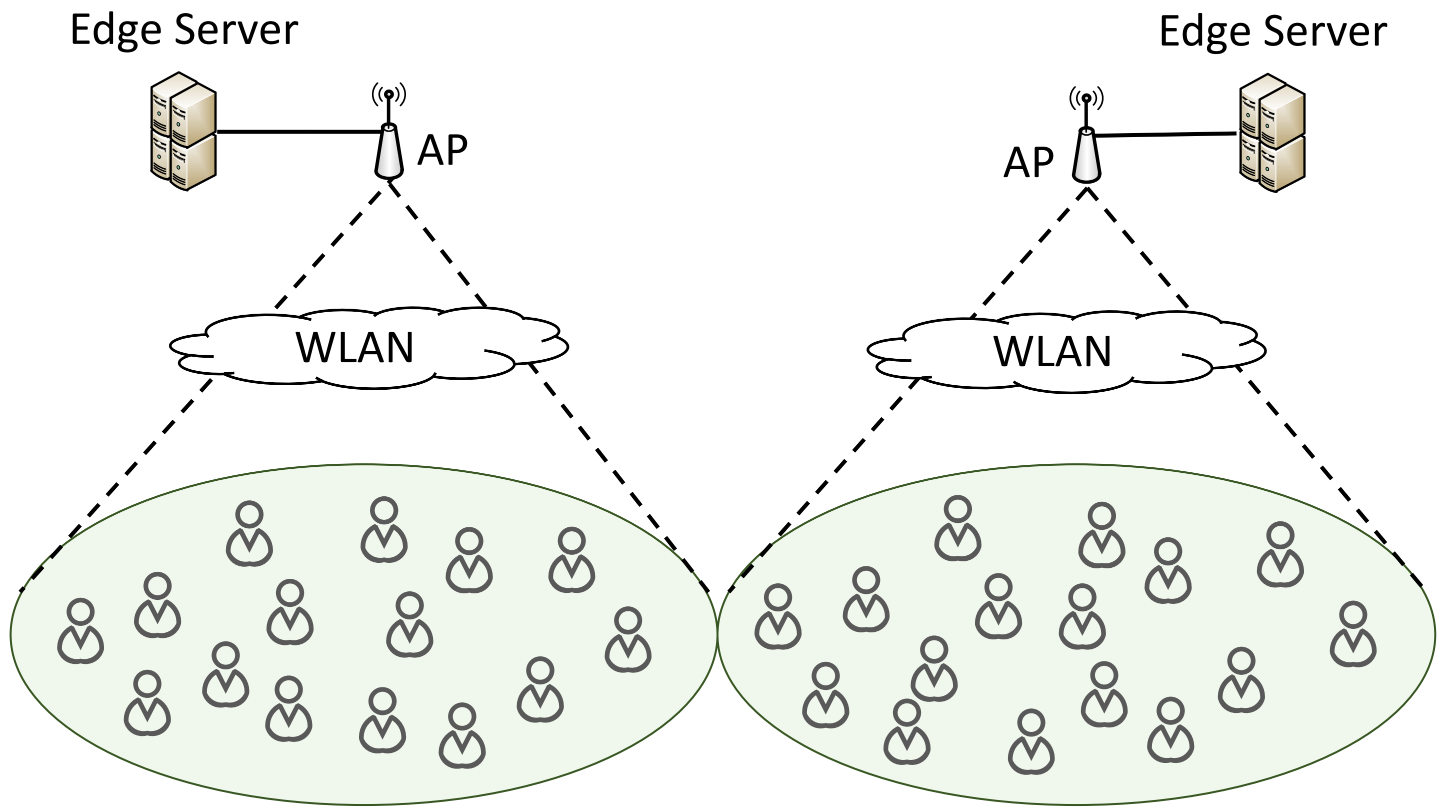
Implementation of Case2

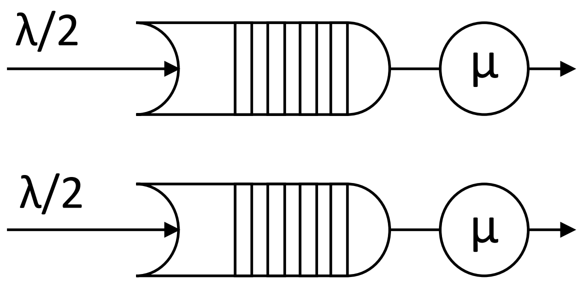




Parameter	**Value**
WLAN Bandwidth	50 Mbps
Number of Core Edge Server	2
Capacity of Edge Server	10 GIPS

Implementation of Case3





Parameter	**Value**
WLAN Bandwidth	50 Mbps
Number of Core Edge Server	2
Capacity of Edge Server	10 GIPS

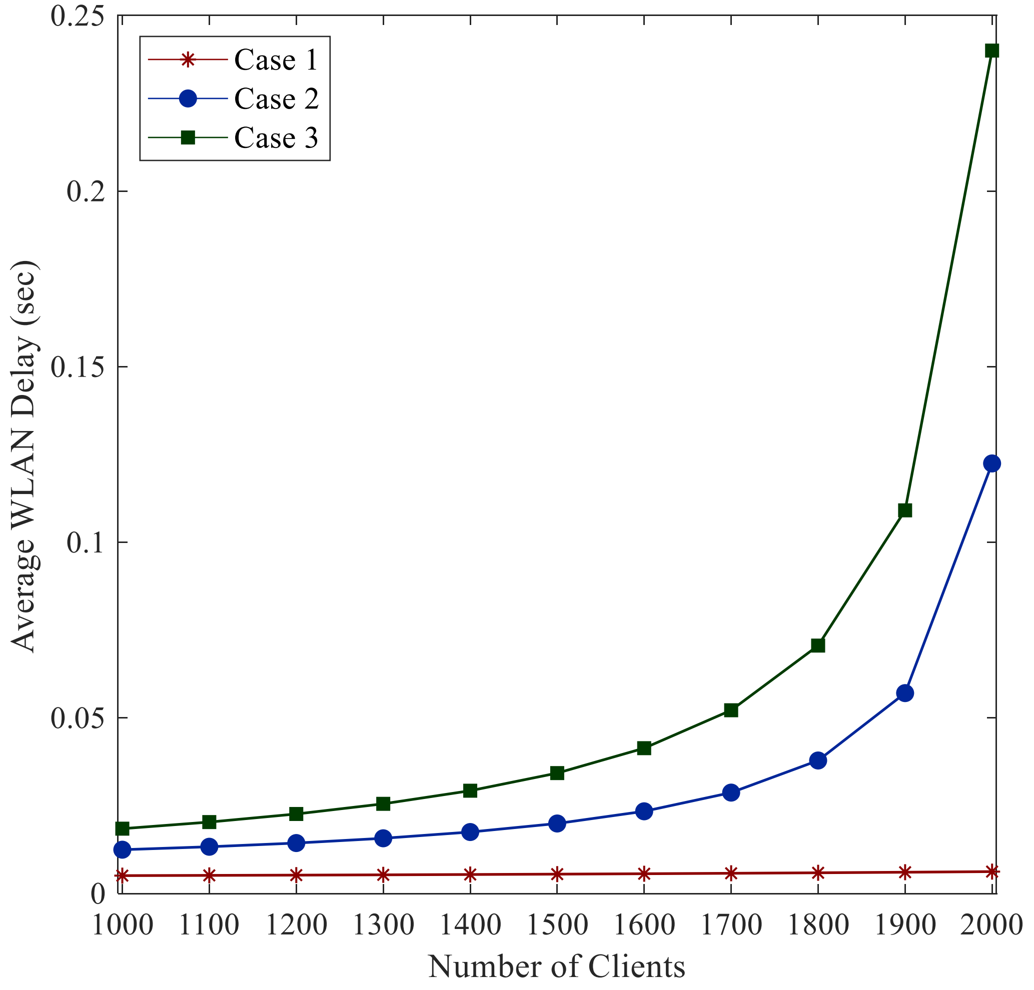
Application Used in Simulations

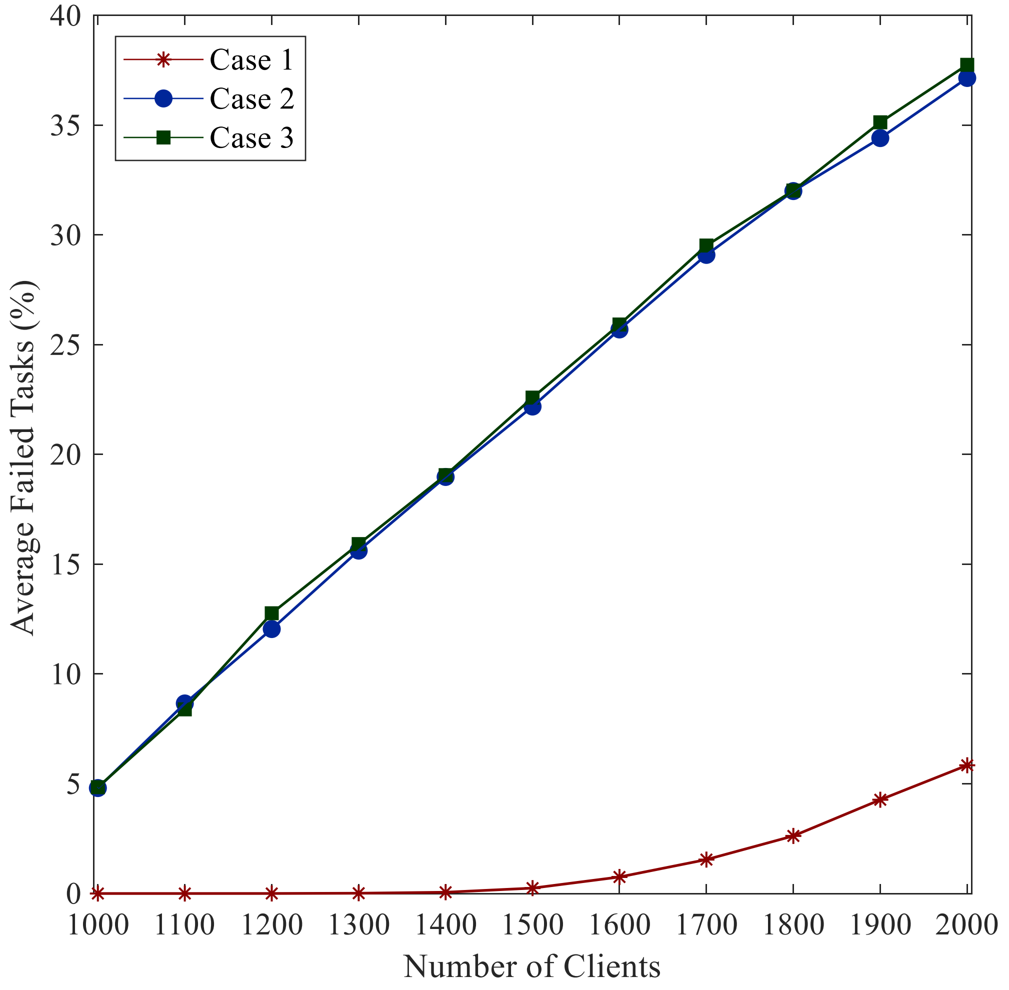
Parameter	Sample App
Task Interarrival (sec)	5
Active/Idle Period (sec)	30/1
VM Utilization on Edge/Cloud (%)	3
Task Length (GI)	500
Upload Data Size (KB)	30
Download Data Size (KB)	30

Simulation Parameters

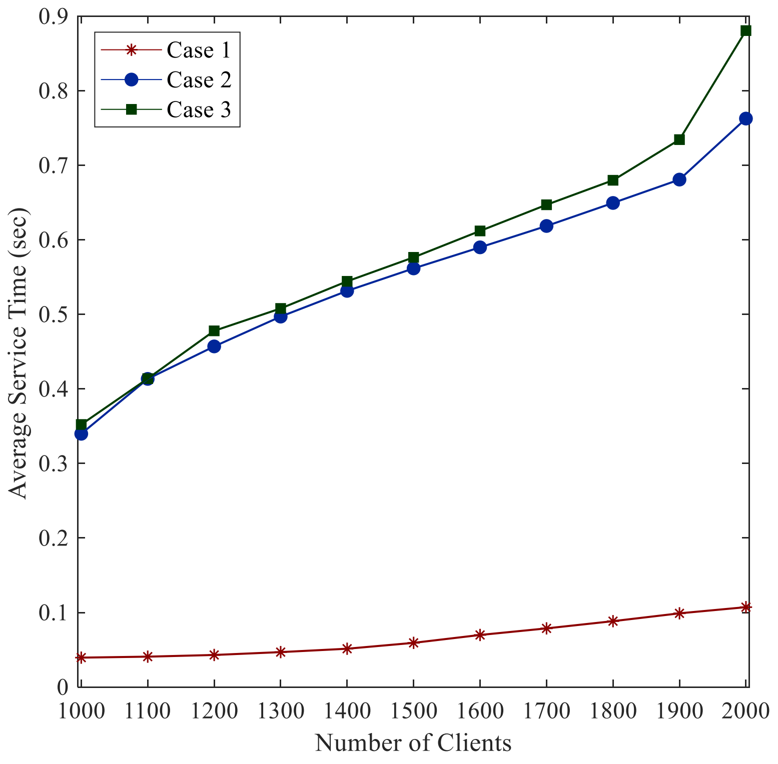
Parameter	**Value**
Simulation Time	30 minutes
Warm-up Period	5 minutes
Number of repetition	25
Mobility Model	Nomadic Mobility
Number of Mobile Clients	1000 to 2000
Length of the Simulated Area	6 KM

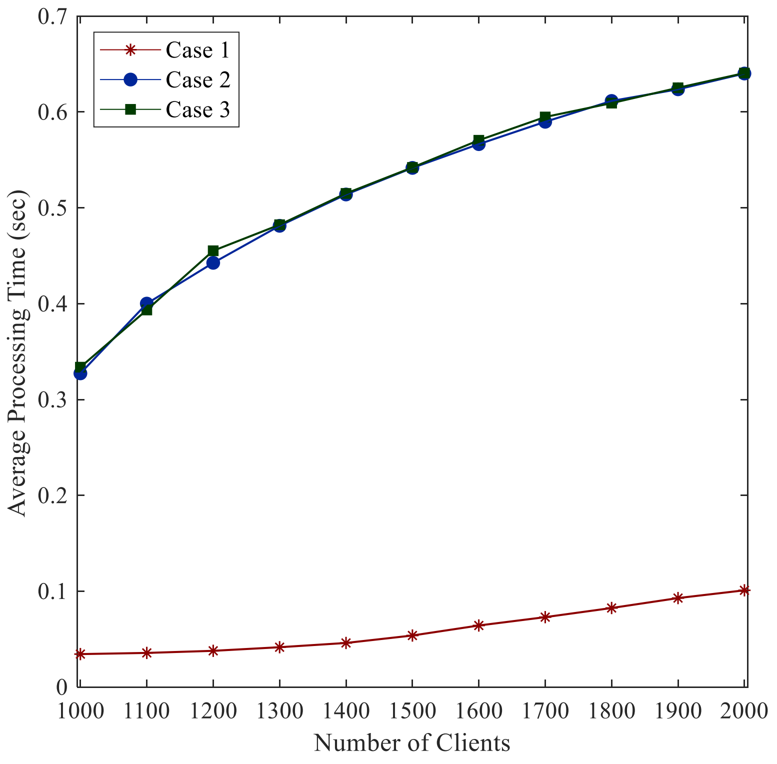
Average WLAN Delay & Success Rate

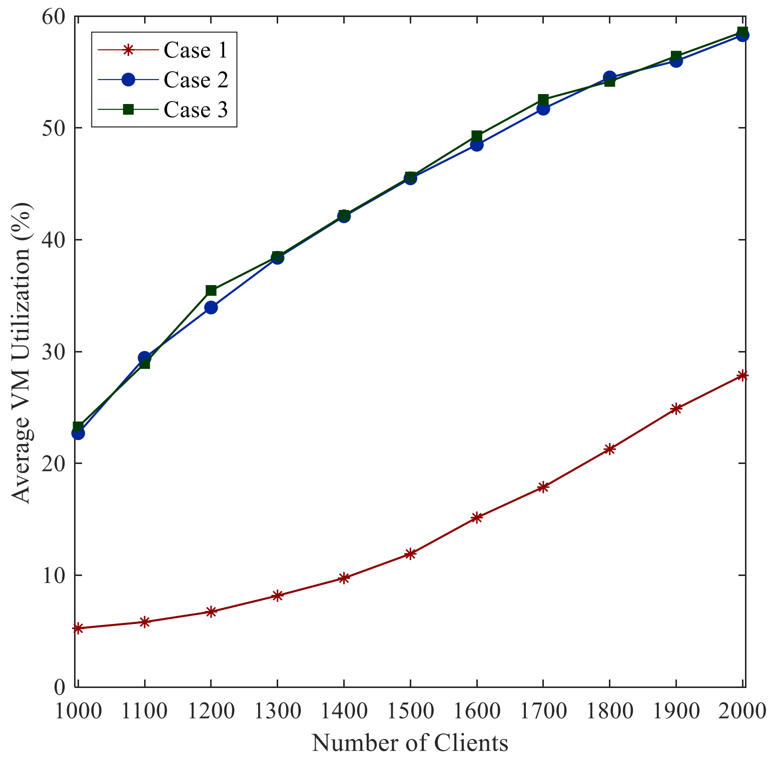




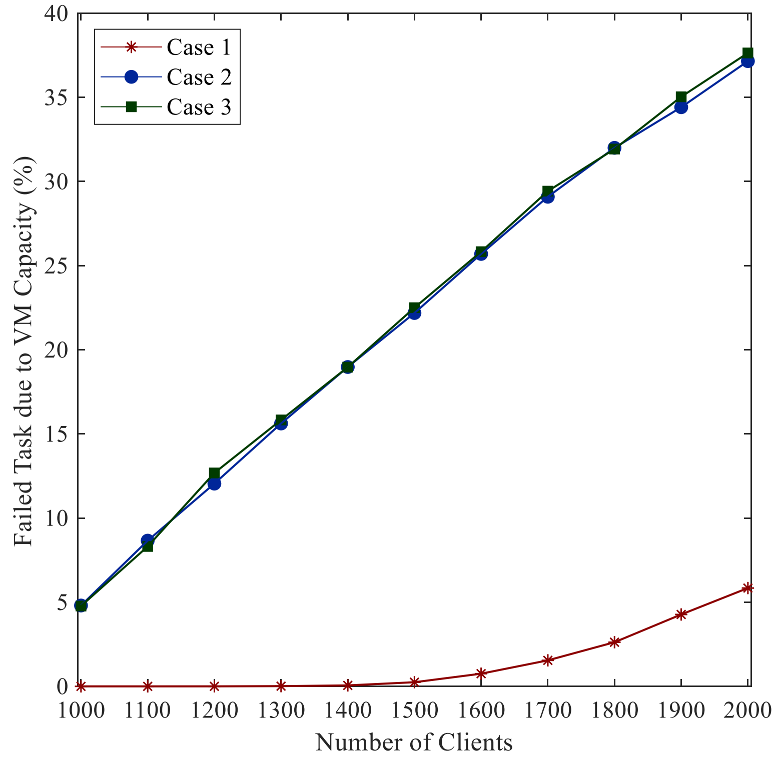
Edge Server Side Statistics

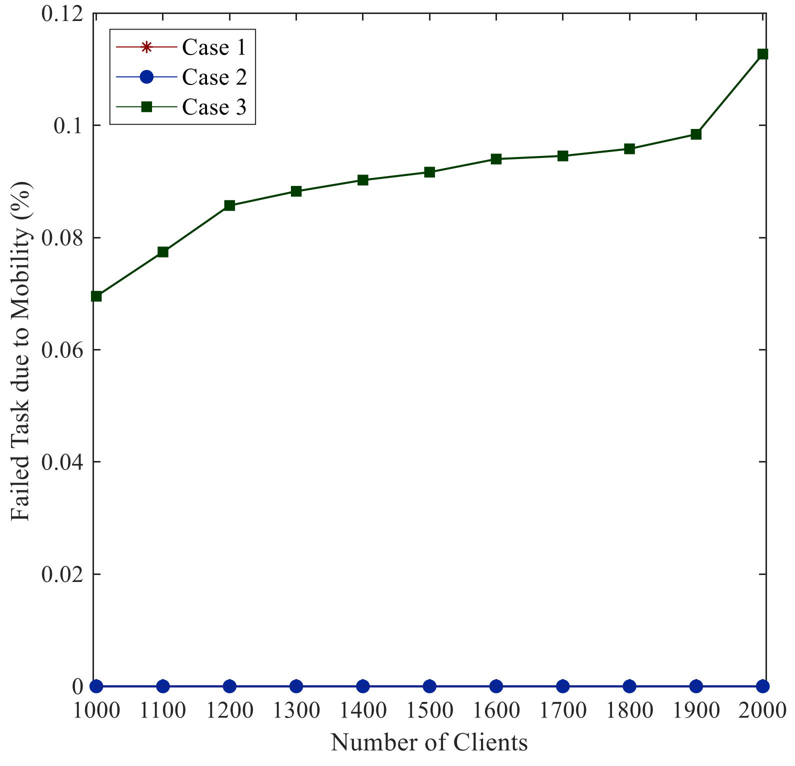


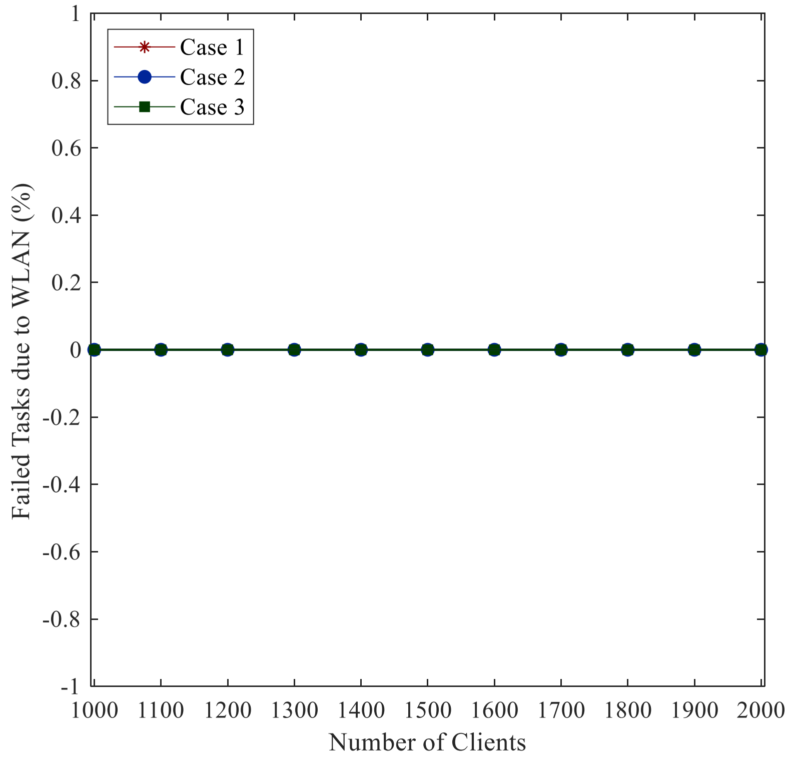




Task Failure Reasons







Thank You