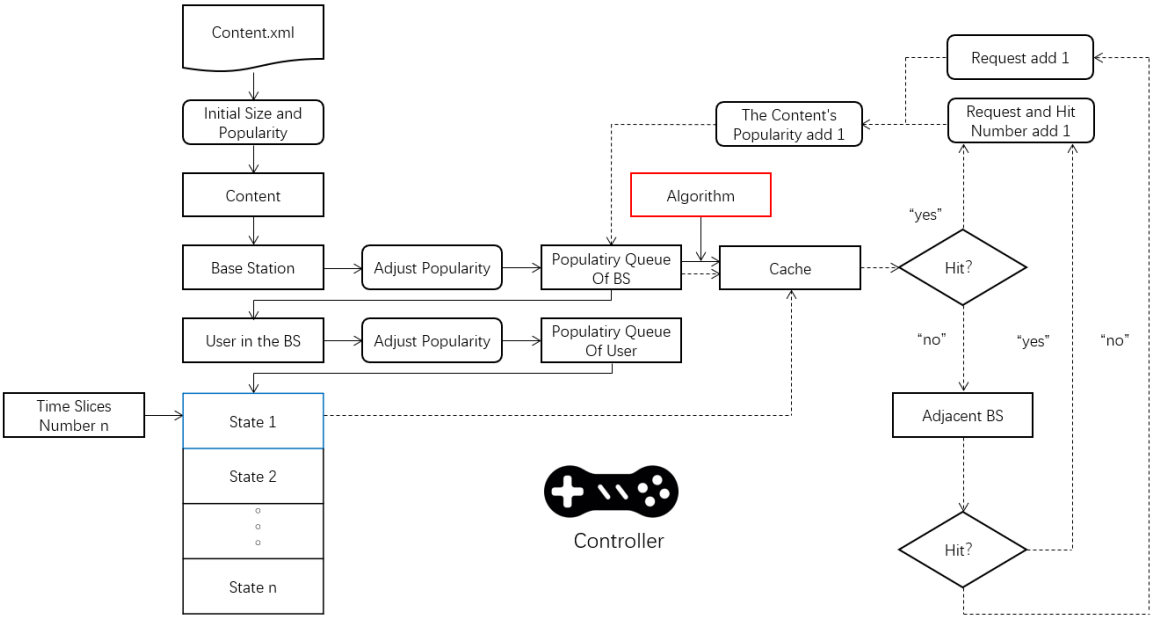
EdgeSim Operation Manual

一、 Introduction

EdgeSim is a Java SE based edge caching/computing simulator that simulates a layered wireless network architecture. It mainly includes base station model, user mobile model, content model and communication model, and provides a convenient method of algorithm import, which can be used for algorithm comparison experiment quickly. Most importantly, it will continue to be updated.

二、 System flow



Before we introduce the system flow, we need to introduce a few important concepts.

Local popularity: in the real network environment, the popularity of multimedia content varies with the coverage of different base stations.Similarly, there are differences in how much each user likes the same content.Therefore, this concept can reflect the

differences of different regions.

Request queue: our entire experiment was based on a time slice, with a large number of requests arriving at the base station in each time slice, and the effect (popularity) was superimposed.

Real-time algorithm and non-real-time algorithm: the real-time algorithm stipulates that the cache should be adjusted every time a request is made; The non-real-time base station only adjusts the base station after each time slice and before the next slice arrives.

The whole experimental process can be summarized as: initialization, request arrival, calculation results, cache adjustment, request arrival.....The detailed system flow is as follows:

1. At the beginning of system operation, the system initializes the size and popularity of all contents;
2. Local popularity is reflected between base stations and users through "fluctuation adjustment" strategy;
3. The algorithm adjusts the cache in the base station;
4. The requests in each time slice arrive at the base station, and the hit rate, delay and traffic status are counted.

三、 Controller

The controller is a tool that developers use for any model in the operating system.In theory, developers can get any object they want from the controller, especially in algorithmic design, which is critical.

Controller controller = controller.getinstance ().

In the non-real-time algorithm, we can obtain the base station (including cache, etc.) and the popularity of all contents in the current base station through the controller, so as to conduct cache configuration.

In the real-time algorithm, we will pass the base station and request content of the response to the algorithm when each request arrives.So, in general, we don't use controllers in real-time algorithms.

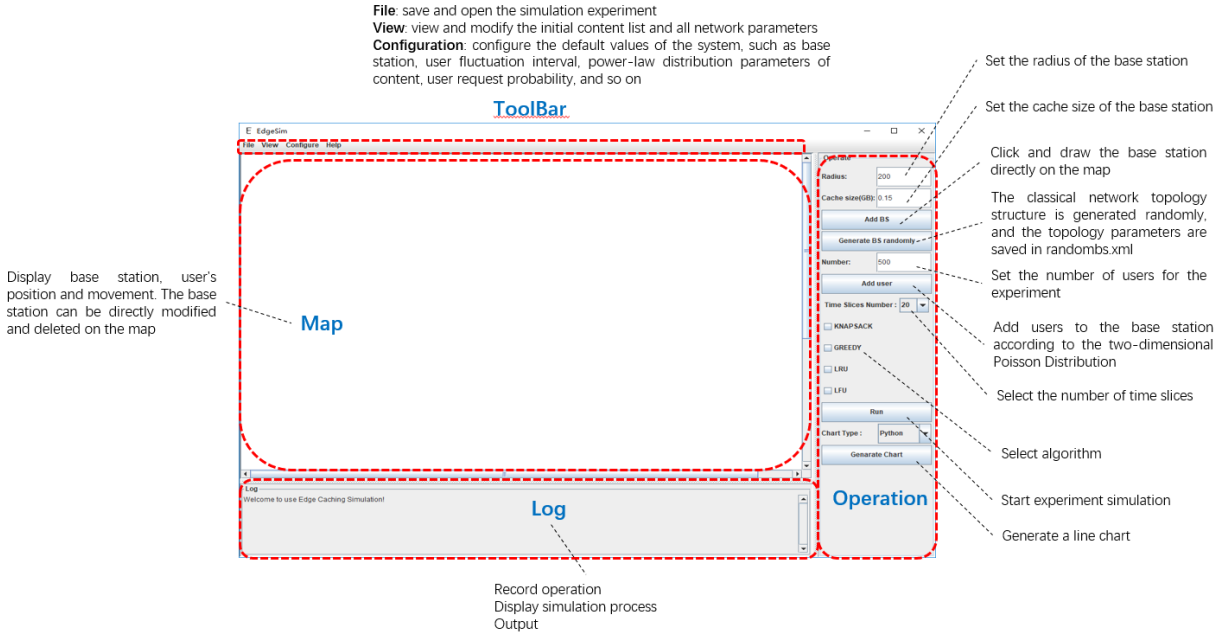
Parameters that the controller can call (part) :

Base station: coverage, cache size, current cache queue, past request queue, number of requests, hit number, popularity distribution in the cell, associated base station, all users in the coverage and so on;

Users: mobile model, speed, acceleration, current coordinates, base station, distance from base station, past request queue, personal preference, probability of request generation, etc.;

Content: popularity distribution.

四、 Main interface



五、 Algorithm imports method

Algorithm imports method is very simple, only need three steps. Most importantly, we give detailed examples, such as backpack and greedy algorithm are non-real-time algorithms, while LRU and LFU are real-time algorithms.

A. A. As mentioned above, the cache algorithm includes real-time algorithm and non-real-time algorithm. Before importing the algorithm, first implement the corresponding interface according to your own needs. RealTimeAlgorithm is

implemented in real-time algorithm, while non-real-time algorithm is implemented in OnetTimeAlgorithm and added its own algorithm in setCache().

B. B. Add labels in swing/operator to select the algorithm in the visual interface. There are three things to add, so follow the example closely:

```
/**
 * Text Field: amount of users
 */
private JTextField userAmountText;
private JCheckBox knapsackAlgorithm;
private JCheckBox greedyAlgorithm;
private JCheckBox lruAlgorithm;
private JCheckBox lfuAlgorithm;
```

```
{
    knapsackAlgorithm = new JCheckBox();
    greedyAlgorithm = new JCheckBox();
    lruAlgorithm = new JCheckBox();
    lfuAlgorithm = new JCheckBox();
    knapsackAlgorithm.setText("KNAPSACK");
    greedyAlgorithm.setText("GREEDY");
    lruAlgorithm.setText("LRU");
    lfuAlgorithm.setText("LFU");
}
```

```
{
    this.add(Radius);
    this.add(Cache);
    this.add(addBS);
    this.add(addBSRandomly);
    this.add(User);
    this.add(addUser);
    this.add(timeSlicesPanel);
    this.add(knapsackAlgorithm);
    this.add(greedyAlgorithm);
    this.add(lruAlgorithm);
    this.add(lfuAlgorithm);
    this.add(run);
    this.add(chartTypePanel);
    this.add(generateChart);

}
```

C. In the run method of the swing/operator class, add the response. Again, follow the example:

```
if (knapsackAlgorithm.isSelected()) {
    algorithmSelected = true;
    controller.getWirelessNetworkGroup().clearAllCache();
    controller.getRequestHandler().processRequest(new KnapsackAlgorithm(), knapsackAlgorithm.getText(), (Integer)(timeSlices.getSelectedItem()));
}

if(greedyAlgorithm.isSelected()) {
    algorithmSelected = true;
    ContentService.resetMyHobby(initialMyHobbyMap,controller.getWirelessNetworkGroup().BS);
    controller.getWirelessNetworkGroup().clearAllCache();
    controller.getRequestHandler().processRequest(new GreedyAlgorithm(), greedyAlgorithm.getText(), (Integer)(timeSlices.getSelectedItem()));
}

if(lruAlgorithm.isSelected()){
    algorithmSelected = true;
    ContentService.resetMyHobby(initialMyHobbyMap,controller.getWirelessNetworkGroup().BS);
    controller.getWirelessNetworkGroup().clearAllCache();
    controller.getRequestHandler().processRequest(new LRUAlgorithm(), lruAlgorithm.getText(), (Integer)(timeSlices.getSelectedItem()));
}

if(lfuAlgorithm.isSelected()){
    algorithmSelected = true;
    ContentService.resetMyHobby(initialMyHobbyMap,controller.getWirelessNetworkGroup().BS);
    controller.getWirelessNetworkGroup().clearAllCache();
    controller.getRequestHandler().processRequest(new LFUAlgorithm(), lfuAlgorithm.getText(), (Integer)(timeSlices.getSelectedItem()));
}
```

六、 Statement

If you are interested in edge computing/caching, we are looking forward to your participation. Finally, our fully interfacing Web emulator is under development. If you would like to participate, please contact me!