Politenico di Milano

Dipartimento Elettronica, Informazione e Bioingegneria

HEAPLab Project Report

EdgeCloudSim Report

Author:
Zhang Qiaolun

Supervisor:
Michele Zanella

February 15, 2019

Abstract

The simulation tool EdgeCloudSim provides some basic features of simulating edge computing scenarios. But it lacks some further features. This project extends basic task application definition to support task-based application.

1 Introduction

EdgeCloudSim is a simulation tool which can simulate Edge Computing scenarios. But this tool can not simulate task-based application. The project adds the following features to the simulation tool:

1. Extend basic task application definition to support task-based application
2. Task migration among the Edge or Cloud VMs
3. Add probabilistic network failure model by considering the congestion or other parameters such as the distance between mobile device and the WiFi access point.
4. Visual tool for displaying the network topology

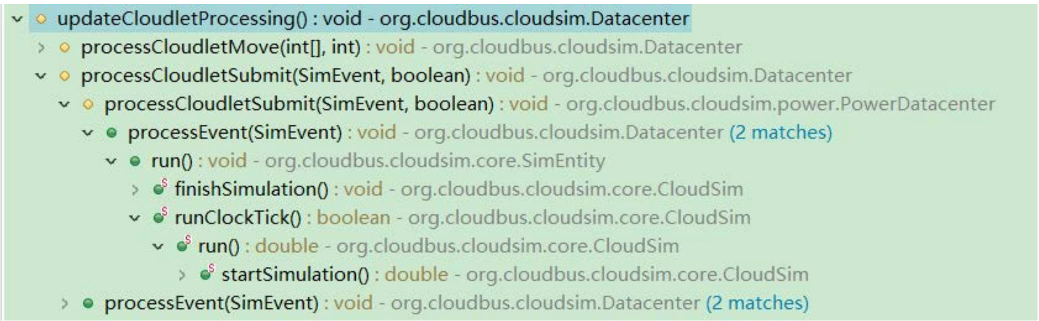
In the end, this project gives a detailed simulation and provides the simulation results.

2 Relation between EdgeCloudSim and Cloudsim

2.1 Simulation Framework of CloudSim

2.1.1 Simulation Data Flow

CloudSim is a discrete event management framework. The core simulation process is a loop function which checks the events related to all the entities and processes the event. Each entities can process different kinds of events. So figuring out all the



```
⌄ ○ updateCloudletProcessing() : void - org.cloudbus.cloudsim.Datacenter
  › ○ processCloudletMove(int[], int) : void - org.cloudbus.cloudsim.Datacenter
  ⌄ ○ processCloudletSubmit(SimEvent, boolean) : void - org.cloudbus.cloudsim.Datacenter
    ⌄ ○ processCloudletSubmit(SimEvent, boolean) : void - org.cloudbus.cloudsim.power.PowerDatacenter
      ⌄ ● processEvent(SimEvent) : void - org.cloudbus.cloudsim.Datacenter (2 matches)
        ⌄ ● run() : void - org.cloudbus.cloudsim.core.SimEntity
          › ● finishSimulation() : void - org.cloudbus.cloudsim.core.CloudSim
          ⌄ ● runClockTick() : boolean - org.cloudbus.cloudsim.core.CloudSim
            ⌄ ● run() : double - org.cloudbus.cloudsim.core.CloudSim
              › ● startSimulation() : double - org.cloudbus.cloudsim.core.CloudSim
    › ● processEvent(SimEvent) : void - org.cloudbus.cloudsim.Datacenter (2 matches)
```

Figure 1: Call hierarchy of function updateCloudletProcessing.

2.1.2 Data center internal processing

Figure 1 shows the call hierarchy of member function updateCloudletProcessing in class DataCenter. From this figure, we can figure out that the Cloudlet processing update process figure in the paper about CloudSim is wrong. Figure 2 is the fixed Cloudlet processing update process. At the end of the function updateCloudletProcessing, it will add a new VM_DATACENTER_EVENT to the future queue using function schedule. When we go back to the loop of Run function, the updateCloudletProcessing function will be triggered again.

2.2 Simulation Framework of EdgeCloudSim

As is shown in the picture1, there are two important classes in core package: ScenarioFactory and SimManager. The ScenarioFactory gets the parameters for the scenarios. And the SimManager receives the object of type ScenarioFactory. Moreover, the SimManager is a class extended from SimEntity class. The SimEntity is a class defined in CloudSim. And it has a function called startEntity(), which will schedule the task.

Not sure if I can implement the task-based application here or change the schedule function in CloudSim

2.3 Modules in EdgeCloudsim

1. core: three are there important class in this module. ScenarioFactory.java is the class for factory scheme. SimManager.java class extends SimEntity class and it is related to CREATE_TASK, CHECK_ALL_VM,

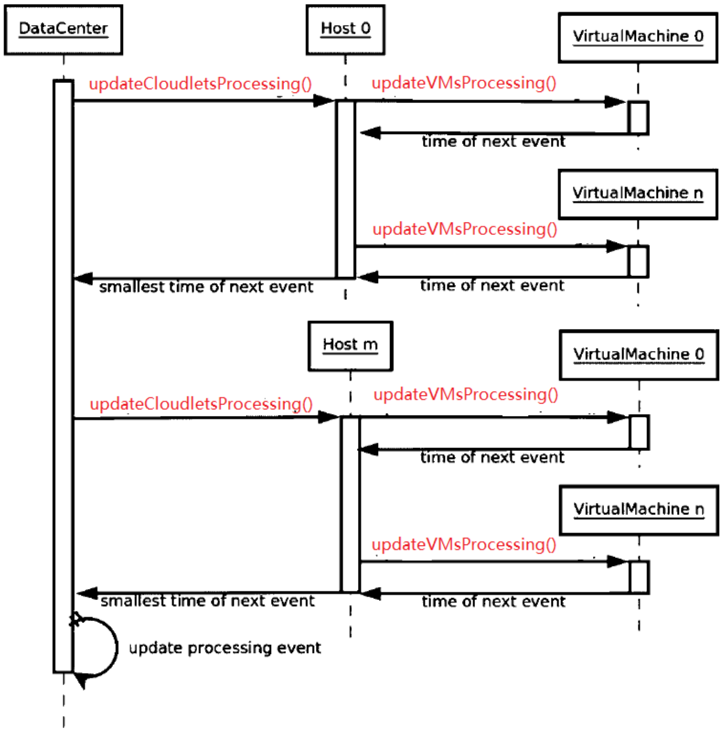


Figure 2: Cloudlet processing update process.

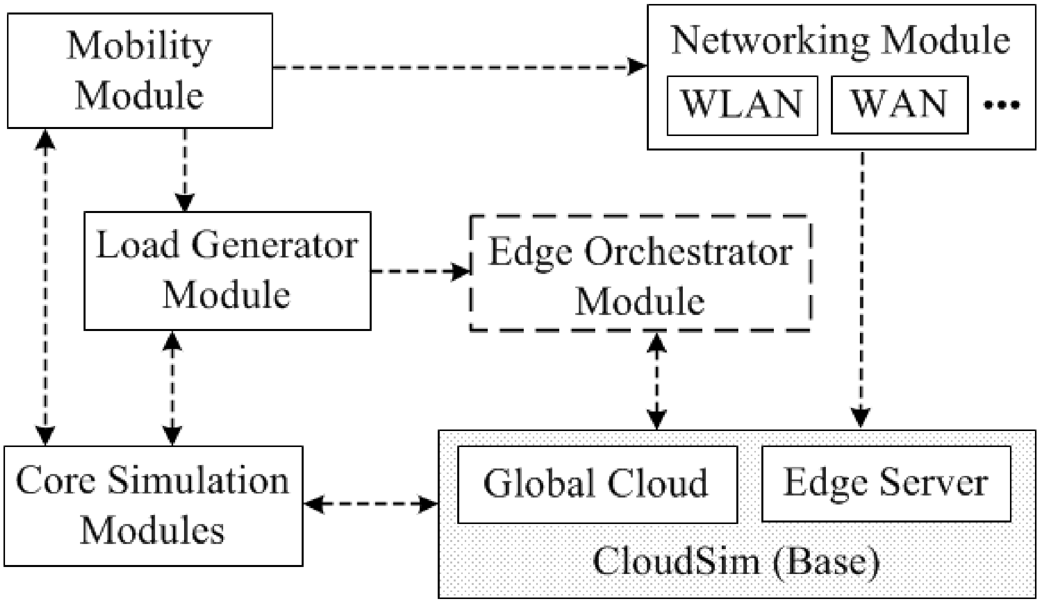


Figure 3: This is a figure caption.

GET_LOAD_LOG, PRINT_PROGRESS, STOP_SIMULATION. SimSettings.java is the class that stores all the configurations.

2. cloud_server: class CloudServerManager.java. This class actually just generate the hostlist, vmlist, local data center, and function to get the average utilization of all VMS.

3. edge_server: class EdgeServerManager.java has similar functions as CloudServer.

4. mobile_processing: class MobileServerManager.java. This class enables the mobile devices have the ability to process task. We can also create data centers and virtual machines on it.

5. edge_client: MobileDeviceManager.java extends DatacenterBroker class in CloudSim. And it overwrite the processOtherEvent function and add new events: REQUEST_RECEIVED_BY_CLOUD, REQUEST_RECEIVED_BY_EDGE

2.4 Entities in EdgeCloudsim

1. SimManager: public class SimManager extends SimEntity

2. MobileDeviceManager: public abstract class MobileDeviceManager extends DatacenterBroker

3. EdgeOrchestrator: public abstract class EdgeOrchestrator extends SimEntity

2.5 Relationship of Modules between CloudSim and EdgeCloudSim

Because EdgeCloudSim is implemented on the top of CloudSim, it also relies on the discrete event management framework. MobileDeviceManager class extends DatacenterBroker class. So it implemented the following functions

3 Design and Implementation

3.1 Task-based Application Design and Implementation

3.1.1 Task-based Application Design

In EdgeCloudSim, task is atomic and cannot be divided into smaller tasks. Task-based application is an application which is composed by several smaller tasks. There may also have dependencies among these tasks. For instance, one sub-task can only starts after another sub-task.

There are two ways to implemented tasked-based application feature in EdgeCloudSim. The first one is to change the code of CloudSim and add task-based features. The second one is to change the code of EdgeCloudSim and add the feature of tasked-based application.

We can define how to divide task in XML file and store their dependencies. In the simulation, so we choose only submit a task to virtual machine when its dependencies has been met. In the SimManager class, the startEntity() function creates all the virtual machines in cloud server, edge server and mobile server. Besides, it also submits all the tasks to virtual machines. But in this project, we only submit the tasks whose dependencies has been met at this time. When a task is finished, it will send a CLOUDLET_RETURN event to DatacenterBroker. In EdgeCloudSim, this event will be sent to MobileDeviceManager. Then processCloudletReturn function will be triggered. So when a task is finished, the tasks that have not been submitted will be checked. If there are tasks whose dependencies has been met now, it will be submitted to the VM at this time. The submit of tasks are based on the schedule function. We implement these functions in MobileDeviceManager class.

In processCloudletReturn function, it will test the execution place of the task, if it is executed in GENERIC_EDGE_DEVICE_ID, the download delay will be added to the task, if it is executed in mobile device, the delay do not need to be add. We can test whether we need to submit new tasks to virtual machines at the beginning of the function.

3.1.2 Scheduling Algorithm in EdgeCloudSim

Which module can we modify to achieve task-based application? How to deal with task-dependency graph

3.1.3 Component Diagram of the Simulation tool

After adding the feature, the component diagram has changed.

3.2 Task Migration

3.3 Probabilistic Network Failure Model

4 Experimental Results

In this section, we did some experiment. We test the failure rate.

5 Conclusions

6 Some LaTeX Examples

6.1 Sections

Use section and subsection commands to organize your document. LaTeX handles all the formatting and numbering automatically. Use ref and label commands for cross-references.

Figure 4: This is a figure caption.

Item	Quantity
Widgets	42
Gadgets	13

Table 1: An example table.

6.2 Comments

Comments can be added to the margins of the document using the todo command, as shown in the example on the right. You can also add inline comments too:

Here's a comment in the margin!

This is an inline comment.

6.3 Tables and Figures

Use the table and tabular commands for basic tables — see Table 1, for example. You can upload a figure (JPEG, PNG or PDF) using the files menu. To include it in your document, use the includegraphics command as in the code for Figure 4 below.

6.4 Mathematics

LaTeX is great at typesetting mathematics. Let $X_1, X_2, \ldots, X_n$ be a sequence of independent and identically distributed random variables with $\mathrm{E}[X_i] = \mu$ and $\mathrm{Var}[X_i] = \sigma^2 < \infty$, and let

$$S_n = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n} \sum_{i}^{n} X_i$$

denote their mean. Then as n approaches infinity, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $\mathcal{N}(0, \sigma^2)$.

6.5 Lists

You can make lists with automatic numbering …

1. Like this,
2. and like this.

…or bullet points …

- Like this,
- and like this.

We hope you find writeLaTeX useful, and please let us know if you have any feedback using the help menu above.