1. Initiallizesd the cloudsim simulation environment.

2. Create a data center.

3. Create a broker which acts as a mediator between user and data center.

4. Define and create 4 virtual machines.

5. Define and create 8 cloudlet.

6. Bind specific cloudlet to the virtual machines.

7. Start the simulation.

8. Print Result.

import java.text.DecimalFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.LinkedList;

import java.util.List;

import org.cloudbus.cloudsim.Cloudlet;

import org.cloudbus.cloudsim.CloudletSchedulerTimeShared;

import org.cloudbus.cloudsim.Datacenter;

import org.cloudbus.cloudsim.DatacenterBroker;

import org.cloudbus.cloudsim.DatacenterCharacteristics;

import org.cloudbus.cloudsim.Host;

import org.cloudbus.cloudsim.Log;

import org.cloudbus.cloudsim.Pe;

import org.cloudbus.cloudsim.Storage;

import org.cloudbus.cloudsim.UtilizationModel;

import org.cloudbus.cloudsim.UtilizationModelFull;

import org.cloudbus.cloudsim.Vm;

import org.cloudbus.cloudsim.VmAllocationPolicySimple;

import org.cloudbus.cloudsim.VmSchedulerTimeShared;

import org.cloudbus.cloudsim.core.CloudSim;

import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;

import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;

import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

// A simple example showing how to create a data center

// with one host and run eight cloudlets on it

public class CloudSimExample1 {

// The cloudlet list

private static List<Cloudlet> cloudletList;

// The vmlist

private static List<Vm> vmlist;

@SuppressWarnings("unused")

public static void main(String[] args)

{

Log.printLine("Starting CloudSimExample2...");

try {

// First step: Initialize the CloudSim package.

// It should be called before creating any

// entities. number of cloud users

int num\_user = 1;

// Calendar whose fields have been initialized

// with the current date and time.

Calendar calendar = Calendar.getInstance();

// trace events

boolean trace\_flag = false;

CloudSim.init(num\_user, calendar, trace\_flag);

// Second step: Create Datacenters

// Datacenters are the resource providers in

// CloudSim. We need at list one of them to run

// a CloudSim simulation

Datacenter datacenter0

= createDatacenter("Datacenter\_0");

// Third step: Create Broker

DatacenterBroker broker = createBroker();

int brokerId = broker.getId();

// Fourth step: Create four virtual machine

vmlist = new ArrayList<Vm>();

// VM description

int vmid = 0;

int mips = 1000;

long size = 10000; // image size (MB)

int ram = 512; // vm memory (MB)

long bw = 1000; // bandwidth

int pesNumber = 1; // number of cpus

String vmm = "Xen"; // VMM name

// create 4 VMs

Vm vm1

= new Vm(vmid, brokerId, mips, pesNumber,

ram, bw, size, vmm,

new CloudletSchedulerTimeShared());

vmid++;

Vm vm2 = new Vm(

vmid, brokerId, mips \* 2, pesNumber,

ram - 256, bw, size \* 2, vmm,

new CloudletSchedulerTimeShared());

vmid++;

Vm vm3 = new Vm(

vmid, brokerId, mips / 2, pesNumber,

ram + 256, bw, size \* 3, vmm,

new CloudletSchedulerTimeShared());

vmid++;

Vm vm 4

= new Vm(vmid, brokerId, mips \* 4,

pesNumber, ram, bw, size \* 4, vmm,

new CloudletSchedulerTimeShared());

vmid++;

// add the VM to the vmList

vmlist.add(vm1);

vmlist.add(vm2);

vmlist.add(vm3);

vmlist.add(vm4);

// submit vm list to the broker

broker.submitVmList(vmlist);

// Fifth step: Create eight Cloudlets

cloudletList = new ArrayList<Cloudlet>();

// Cloudlet properties

int id = 0;

long length = 400000;

long fileSize = 300;

long outputSize = 300;

UtilizationModel utilizationModel

= new UtilizationModelFull();

Cloudlet cloudlet1 = new Cloudlet(

id, length, pesNumber, fileSize, outputSize,

utilizationModel, utilizationModel,

utilizationModel);

cloudlet1.setUserId(brokerId);

id++;

Cloudlet cloudlet2 = new Cloudlet(

id, length \* 2, pesNumber, fileSize \* 2,

outputSize / 3, utilizationModel,

utilizationModel, utilizationModel);

cloudlet2.setUserId(brokerId);

id++;

Cloudlet cloudlet3 = new Cloudlet(

id, length / 2, pesNumber, fileSize \* 3,

outputSize \* 3, utilizationModel,

utilizationModel, utilizationModel);

cloudlet3.setUserId(brokerId);

Cloudlet cloudlet4 = new Cloudlet(

id, length / 3, pesNumber, fileSize / 3,

outputSize / 2, utilizationModel,

utilizationModel, utilizationModel);

cloudlet4.setUserId(brokerId);

Cloudlet cloudlet5 = new Cloudlet(

id, length \* 3, pesNumber, fileSize / 2,

outputSize / 4, utilizationModel,

utilizationModel, utilizationModel);

cloudlet5.setUserId(brokerId);

Cloudlet cloudlet6 = new Cloudlet(

id, length / 4, pesNumber, fileSize \* 4,

outputSize \* 4, utilizationModel,

utilizationModel, utilizationModel);

cloudlet6.setUserId(brokerId);

Cloudlet cloudlet7 = new Cloudlet(

id, length \* 4, pesNumber, fileSize,

outputSize \* 2, utilizationModel,

utilizationModel, utilizationModel);

cloudlet7.setUserId(brokerId);

Cloudlet cloudlet8 = new Cloudlet(

id, length, pesNumber, fileSize / 4,

outputSize / 3, utilizationModel,

utilizationModel, utilizationModel);

cloudlet8.setUserId(brokerId);

// add the cloudlet to the list

cloudletList.add(cloudlet1);

cloudletList.add(cloudlet2);

cloudletList.add(cloudlet3);

cloudletList.add(cloudlet4);

cloudletList.add(cloudlet5);

cloudletList.add(cloudlet6);

cloudletList.add(cloudlet7);

cloudletList.add(cloudlet8);

// submit cloudlet list to the broker

broker.submitCloudletList(cloudletList);

// bind the cloudlets to the vms,This way the

// broker will submit the bound cloudlets only

// to the specific VM

broker.bindCloudletToVm(

Cloudlet1.getCloudletId(), vm1.getId());

broker.bindCloudletToVm(

Cloudlet2.getCloudletId(), vm2.getId());

broker.bindCloudletToVm(

Cloudlet3.getCloudletId(), vm3.getId());

broker.bindCloudletToVm(

Cloudlet4.getCloudletId(), vm4.getId());

broker.bindCloudletToVm(

Cloudlet5.getCloudletId(), vm1.getId());

broker.bindCloudletToVm(

Cloudlet6.getCloudletId(), vm2.getId());

broker.bindCloudletToVm(

Cloudlet7.getCloudletId(), vm3.getId());

broker.bindCloudletToVm(

Cloudlet8.getCloudletId(), vm4.getId());

// Sixth step: Starts the simulation

CloudSim.startSimulation();

CloudSim.stopSimulation();

// Final step: Print results when simulation is

// over

List<Cloudlet> newList

= broker.getCloudletReceivedList();

printCloudletList(newList);

Log.printLine("CloudSimExample1 finished!");

}

catch (Exception e) {

e.printStackTrace();

Log.printLine("Unwanted errors happen");

}

}

private static Datacenter createDatacenter(String name)

{

// Here are the steps needed to create a

// PowerDatacenter:

// 1. We need to create a list to store

// our machine

List<Host> hostList = new ArrayList<Host>();

9. This will initialize the simulation.

10. This will give number of user.

11. 1 cpu with thousand mips.

12.

13. VM schedular time schedule.

14. 8 cloudlets are created.

#Outcome of this program

-Cloudlet id

-Status

-Data center id

-virtual machine Id-

-Ececution time

-End time

A task for job that simulates the user workloads.

VM: Virtual istances where cloudlets are running

HOST: Host is nothing but physical machines in the data centers.

Data centers: A collection of host provider resources.

Broker: Assign cloudlets to virtual machines.

P - Processing element representing a real element.



