package lte;

import java.io.File;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.Arrays;

import java.util.Random;

import java.util.StringTokenizer;

import java.util.TreeMap;

import javafx.application.Platform;

import javax.swing.JOptionPane;

public class Supersoucesockreceiver extends Thread

{

public String videoname,sysno,portno,path;

SuperSourceController ssc;

SuperSourcemultiReceiver ssmr;

public TreeMap costpath = new TreeMap();

public TreeMap arrcost = new TreeMap();

Observer o;

Random r = new Random();

File df = new File("");

public Supersoucesockreceiver(String videoname,String sysno,String portno,SuperSourceController ssc,SuperSourcemultiReceiver ssmr)

{

this.videoname = videoname;

this.sysno = sysno;

this.portno = portno;

this.ssc = ssc;

this.ssmr = ssmr;

this.o = ssc.getObserver();

start();

}

public void run()

{

try

{

ServerSocket ss = new ServerSocket(Integer.parseInt(portno));

while (true)

{

Socket s = ss.accept();

ObjectInputStream ois = new ObjectInputStream(s.getInputStream());

String status = (String) ois.readObject();

if (status.equalsIgnoreCase("VideoRequesttosupersource"))

{

String choose = (String)ois.readObject();

String source = (String)ois.readObject();

String location = (String)ois.readObject();

String videoname = (String)ois.readObject();

Platform.runLater(new Runnable()

{

@Override

public void run()

{

o.setreqsupersource(source);

}

});

findpath(choose,source,location,videoname);

}

}

}

catch(Exception e)

{

e.printStackTrace();

}

}

public void findpath(String choose,String source,String location,String videoname) throws IOException

{

System.out.println("check traffic status");

File dff = new File(df.getAbsolutePath()+"\\localgateway\\"+location+"\\"+videoname);

if(dff.exists())

{

System.out.println("video status is "+dff.exists());

String trafficintensity = (String)ssmr.allstatus.get(location).toString().trim();

if(trafficintensity.equalsIgnoreCase("LOW"))

{

minimizelinkroute(choose,source,location,videoname);

}

else if(trafficintensity.equalsIgnoreCase("HIGH"))

{

minimizecostroute(choose,source,location,videoname);

}

}

else

{

minimizecostroute(choose,source,location,videoname);

}

}

public void minimizelinkroute(String choose,String source,String location,String videoname) throws IOException

{

String internode="";

JOptionPane.showMessageDialog(null,"Choose Minimize link path");

for(int i=0;i<ssmr.allgateway.size();i++)

{

String localgateway = (String)ssmr.allgateway.get(i).toString();

if(localgateway.equalsIgnoreCase(location))

{

String sysnum = (String)ssmr.gatewaysysno.get(localgateway).toString();

String portnum = (String)ssmr.gatewayportno.get(localgateway).toString();

path = localgateway;

if(!ssmr.allinnodes.isEmpty())

{

internode = ssmr.allinnodes.get(r.nextInt(ssmr.allinnodes.size())).toString();

String sysnu = ssmr.allinsysno.get(internode).toString();

String portnu = ssmr.allinport.get(internode).toString();

path = path+"->"+internode+"->"+source;

Socket s = new Socket(sysnu, Integer.parseInt(portnu));

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("pathSplit");

oos.writeObject(localgateway);

oos.writeObject(internode);

oos.writeObject(source);

oos.writeObject(videoname);

oos.writeObject(path);

oos.close();

s.close();

}

else

{

path = localgateway+"->"+source;

}

Socket s = new Socket(sysnum, Integer.parseInt(portnum));

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("Costbasedpath");

oos.writeObject(choose);

oos.writeObject(path);

oos.writeObject(localgateway);

oos.writeObject(internode);

oos.writeObject(source);

oos.writeObject(videoname);

oos.close();

s.close();

System.out.println("costbasedpath select");

}

}

}

public void minimizecostroute(String choose,String source,String location,String videoname) throws IOException

{

JOptionPane.showMessageDialog(null,"Choose Minimize cost path");

int c=0;

String pdest="";

String pinternode="";

String psource="",pnextinernode="";

String path="";

int[] inputArr = new int[ssmr.allgateway.size()];

for(int i=0;i<ssmr.allgateway.size();i++)

{

String gate = (String) ssmr.allgateway.get(i).toString();

String repla = (String)ssmr.alltraffic.get(gate);

String tra = repla.replace("%", "");

int a = Integer.valueOf(tra);

path = gate+"->"+source;

if(!ssmr.allinnodes.isEmpty())

{

for(int j=0;j<ssmr.allinnodes.size();j++)

{

String internode = (String)ssmr.allinnodes.get(j);

String replac = (String)ssmr.allintraffic.get(internode);

String intertra = replac.replace("%", "");

int b = Integer.valueOf(intertra);

c = a+b;

path = gate+"->"+internode+"->"+source;

costpath.put(c, path);

}

}

else

{

costpath.put(a, path);

}

if(c!=0)

{

inputArr[i]=c;

}

else

{

inputArr[i]=a;

}

}

Arrays.sort(inputArr);

// ===============binary search algorithm===================

int pos=0;

for(int i=0;i<inputArr.length;i++)

{

int key = inputArr[i];

int start=0;

int end=inputArr.length-1;

while (start <= end)

{

int midd = (start + end) / 2;

if (key == inputArr[midd])

{

pos = midd;

arrcost.put(pos, key);

}

if (key < inputArr[midd])

{

end = midd - 1;

}

else

{

start = midd+1;

}

}

}

int lowkey = (int)arrcost.firstKey();

int lowtraffic = (int)arrcost.get(lowkey);

path = (String) costpath.get(lowtraffic);

StringTokenizer pathsplit = new StringTokenizer(path, "->");

pdest = pathsplit.nextToken();

pinternode = pathsplit.nextToken();

psource = pathsplit.nextToken();

String sysnum = (String)ssmr.gatewaysysno.get(pdest).toString();

String portnum = (String)ssmr.gatewayportno.get(pdest).toString();

Socket s = new Socket(sysnum, Integer.parseInt(portnum));

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("Costbasedpath");

oos.writeObject(choose);

oos.writeObject(path);

oos.writeObject(pdest);

oos.writeObject(pinternode);

oos.writeObject(psource);

oos.writeObject(videoname);

oos.close();

s.close();

JOptionPane.showMessageDialog(null,"SuperSource select "+pdest+" gateway");

String sysnu = (String)ssmr.allinsysno.get(pinternode).toString();

String portnu = (String)ssmr.allinport.get(pinternode).toString();

s = new Socket(sysnu, Integer.parseInt(portnu));

oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject("Costbasedpathtointermediate");

oos.writeObject(path);

oos.writeObject(pdest);

oos.writeObject(pinternode);

oos.writeObject(psource);

oos.writeObject(videoname);

oos.close();

s.close();

}

}