计网虚拟路由实验报告

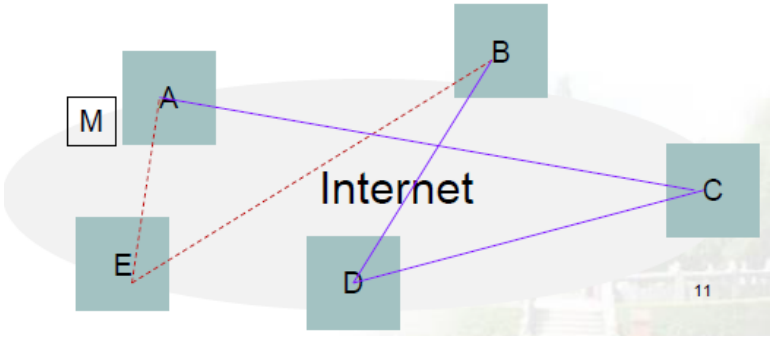
小组成员：黄建武、骆铭涛、蔡桐钊、陈海涛
陈平永、何晋豪、洪创煌、童云钊

1. 实验简介

本次实验分为两个任务：**自组织路由、集中式路由**。

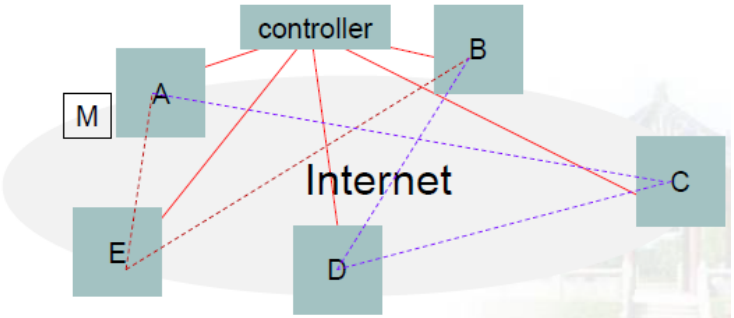
任务 1：自组织路由

- 为所有成员的电脑选择一个虚拟拓扑；
- 根据虚拟拓扑为这些电脑构建虚拟连接；
- 每一台电脑既是客户机也是路由器；
- 每台电脑周期性地交互和更新路由表；
- 一台电脑可以对其他电脑发送信息。



任务 2：集中式路由

- 满足上述自组织路由的要求；
- 控制器决定路由表并分发给每一个成员。



2. 实验设计

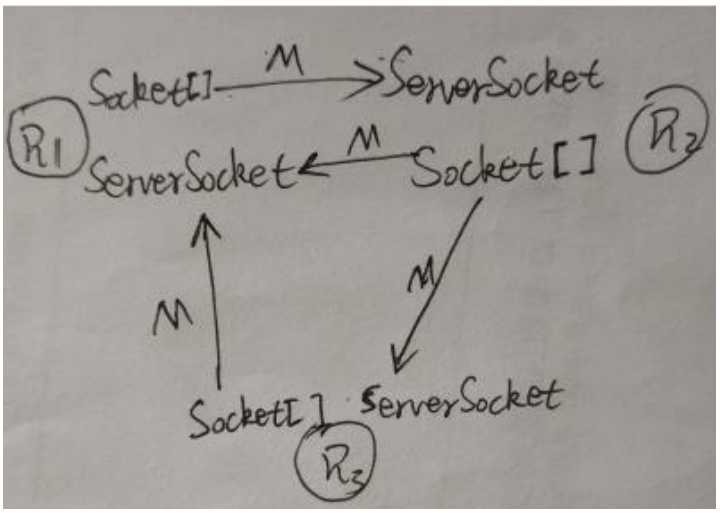
2.1 自组织路由设计思路

算法（DV）：

1) 初始化每个节点的路由表为空，添加直连路由之后在路由表加入新的 item；
2) 路由表产生变化则向邻居转发路由表；
3) 收到邻居的路由表之后与自己的路由表对比，计算新的路由表，计算结束之后如果路由表有新变化，则向邻居继续转发，否则不转发；
4) 步骤 2 和 3 不断重复，最终路由表会达到稳定状态，每个节点都不会继续转发路由表。

通信模型：

1) 每个节点都有一个 ServerSocket，多个 Socket。通信过程中，需要发送数据的节点为 client，接收数据的节点为 server。例：如 A 向 E 转发路由表，则 A 为 client，E 为 server；
2) client 向 server 发起通信，server 上面的 ServerSocket 收到之后 accept()并开启新的线程去处理相应的请求；
3) 节点之间的通信模型如下，每个节点都只有一个 ServerSocket，可以有多个普通 Socket，如此便可以实现任意两个节点之间都能发起连接进行通信，类似于现实中任意两个路由器放到一起，它们之间都能接一条网线；



4) 在代码中加入逻辑限制，使得特定节点只能与特定节点直连通信，比如 A 只能与 C 和 E 直连通信，从而实现虚拟路由按照特定的拓补图连接，类似现实中的给几个路由器拉网线的过程。其他非直连节点只能间接通信，即通过其他路由器转发消息来实现。
5) 每个节点转发的消息由两种类型，一种是普通消息 MSG，一种是路由表 RT。每个节点收到消息之后都会提取报文中的特定字段，判断该消息是否发向自己，如果是则接收，否则继续转发给下一跳节点。

模块接口：

1) GUI

GUI 线程，实现用户界面，便于操作和查看消息传送的过程。

```
private void addListeners();
private void initialGUI();
public void run();
public void updateMessage(String msgType, String message);
public void updateNeighborhood(Set<String> neighbors);
public void setVirtualRouter(VirtualRouter virtualRouter);
```

2) VirtualRouter

虚拟路由节点线程，实现虚拟路由之间的通信，消息传送等功能。同时具备向邻居转发路由表，收取邻居路由表，计算更新自己路由表等功能（DV 算法的实现）。

```
public void setGUI(GUI gui);
public void run();
public void sendSimpleMessage(String to, String simpleMsg);
public void addNeighborhood(String neigbhor);
private void sendRouteTableToAllNeighbor();
private void sendData(String to, Message message);
private void receiveData(Socket socket);
private boolean processRouteTable(Map<String, String[]> neighborTable, String neigbhor);
```

3) Message

封装了整个代码中可能使用到的消息，可以支持 RT，MSG，CM 三种消息类型。消息内容包括【type】，【from】，【to】，【simpleMsg】，【routeTable】。根据对应的消息类型提取相应字段即可得到消息内容。

```
public void setFrom(String from);
public String getFrom();
public void setTo(String to);
public String getTo();
public void setType(String type);
public String getType();
public void setSimpleMsg(String simpleMsg);
public String getSimpleMsg();
public void setRouteTable(Map<String, String[]> routeTable);
public Map<String, String[]> getRouteTable();
```

4) VRMain

虚拟路由和 GUI 的启动模块。

```
public static void main(String args[]);
```

2.2 集中式路由设计思路

算法（LS）：

1) 使用 Dijkstra 算法根据整个网络的拓补图计算最短路径，生成路由表；
2) 拓补图和路由表都存储在 Controller 中。

通信模型：

1) 每个节点都与 Controller 直连，Controller 上面有一个 ServerSocket 负责接收其他普通节点发起的连接并开辟线程去处理；
2) 每个节点初始化的时候向 Controller 发起请求，获取当前网络中的所有节点(网上邻居)；
3) 每个节点发送消息之前(不管是自己主动发送的还是转发别人的)，都先向 Controller 发出请求得到下一跳节点，然后再发给下一跳节点，以此类推；
4) 这个实验中的消息由两种类型，一种是普通消息 MSG，一种是查询消息 CM。MSG 在节点之间传送，用于发送普通的消息。CM 在节点和 Controller 之间传送，当节点向 Controller 查询网络邻居或者下一跳地址的之后就会用到该消息类型。

模块接口：

1) GUI

功能与“任务 1”一致。

```
private void addListeners();
private void initialGUI();
public void setVirtualRouter(VirtualRouter virtualRouter);
public void run();
public void updateMessage(String msgType, String message);
public void updateNeighborhood(String[] neighbors, String me);
```

2) VirtualRouter

基本通信功能与“任务 1”一致，少了传送路由表功能，增加与 Controller 通信的功能。

```
public void setGUI(GUI gui);
public void run();
public void updateNeighborhood(String ip, String port);
public void sendSimpleMessage(String to, String simpleMsg);
private void sendData(String to, Message message);
private void receiveData(Socket socket);
```

3) Message 和 VRMain

均与“任务 1”一致。

4) ControllerLogic

使用 Dijkstra 算法计算路由表，提供路由表查询功能。

```
public ArrayList<Tableitem> getLinkState(int index);
private int selectMinCostNode(HashSet<Integer> Nset, ArrayList<Tableitem> mLinkStates);
private ArrayList<Integer> getNeighbour(HashSet<Integer> Nset, int index);
public static class Tableitem;
```

5) ControllerMain

Controller 的启动模块，同时负责监听来自虚拟路由的请求，调用 ControllerLogic 的接口查询响应的下一跳地址并响应虚拟路由。

```
public static void main(String[] args);
public static class HandlerThread extends Thread;
```

3. 项目启动

任务 1：自组织路由

编译所有 Java 文件，启动 VRMain 运行虚拟路由和 GUI 界面。在命令行参数传入该虚拟路由的地址，以 ip:port 的形式。在局域网中进行试验需要将 127.0.0.1 改为局域网 IP，例如 192.168.1.11。

```
javac *.java -classpath .:beautyeye_lnf.jar
java -classpath .:beautyeye_lnf.jar VRMain 127.0.0.1:8080
```

运行之后，在 GUI 界面中为某些添加直连路由。

任务 2：集中式路由

启动虚拟路由的方式与“任务 1”一致，同样需要传入该路由的地址。

```
javac *.java -classpath .:beautyeye_lnf.jar
java -classpath .:beautyeye_lnf.jar VRMain 127.0.0.1:8080
```

启动 Controller。Controller 启动之后需要输入其地址，同样以 ip:port 的形式输入。

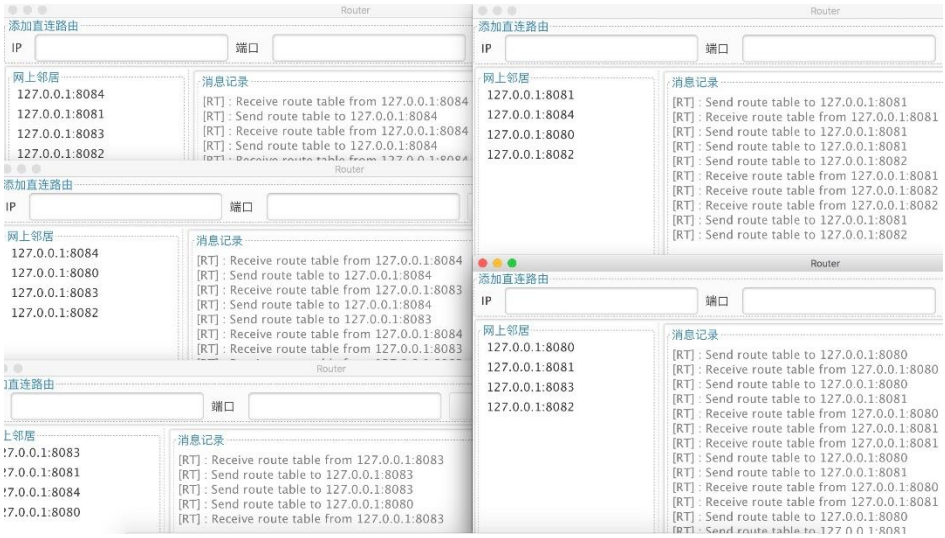
```
java ControllerMain
```

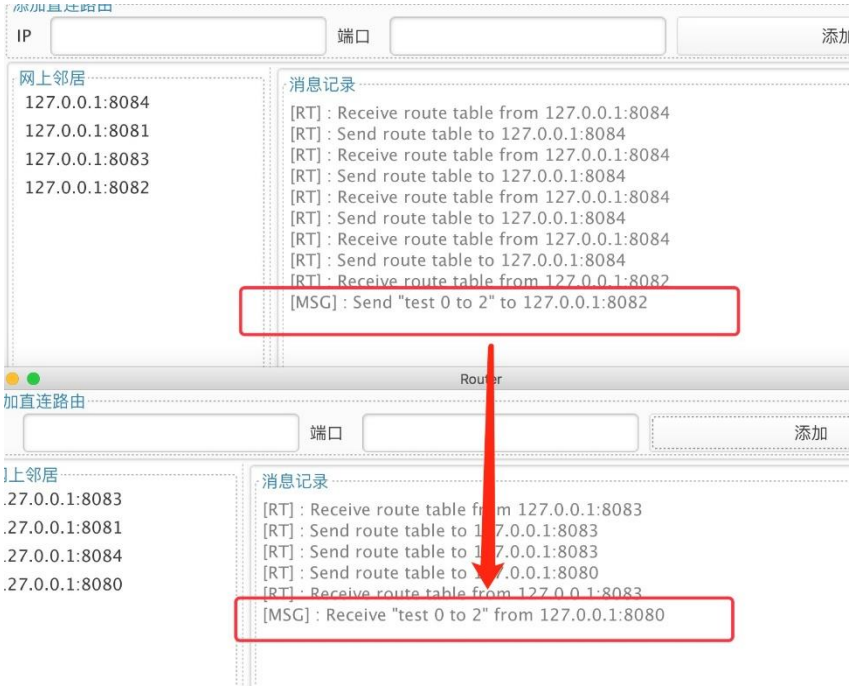
都完成之后，在 GUI 界面中为每个路由器设置 Controller 的地址信息以便访问到 Controller。

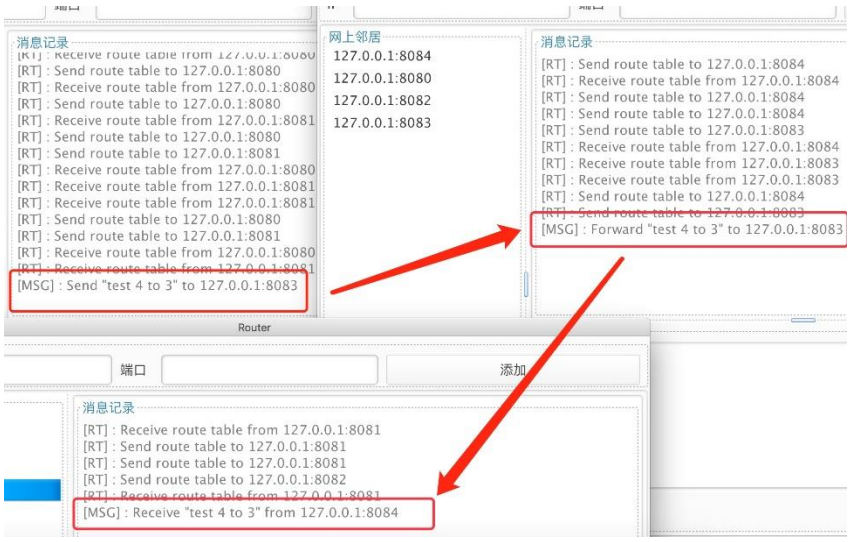
4. 实验结果

实验结果如下图所示，详细过程请见视频。

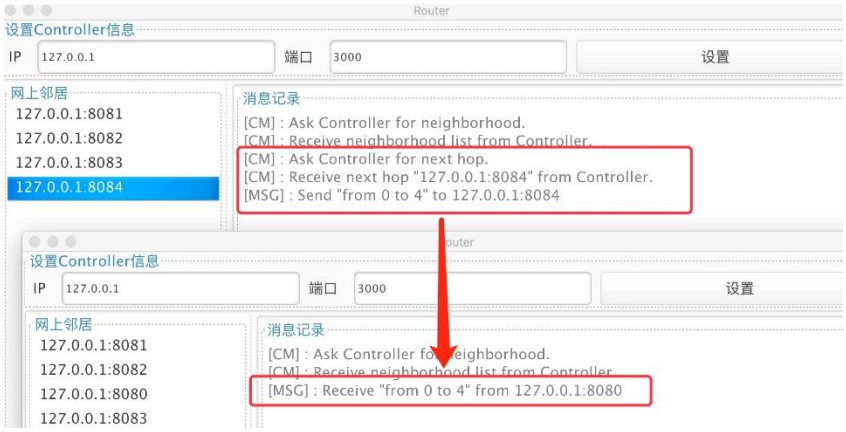
任务 1：自组织路由

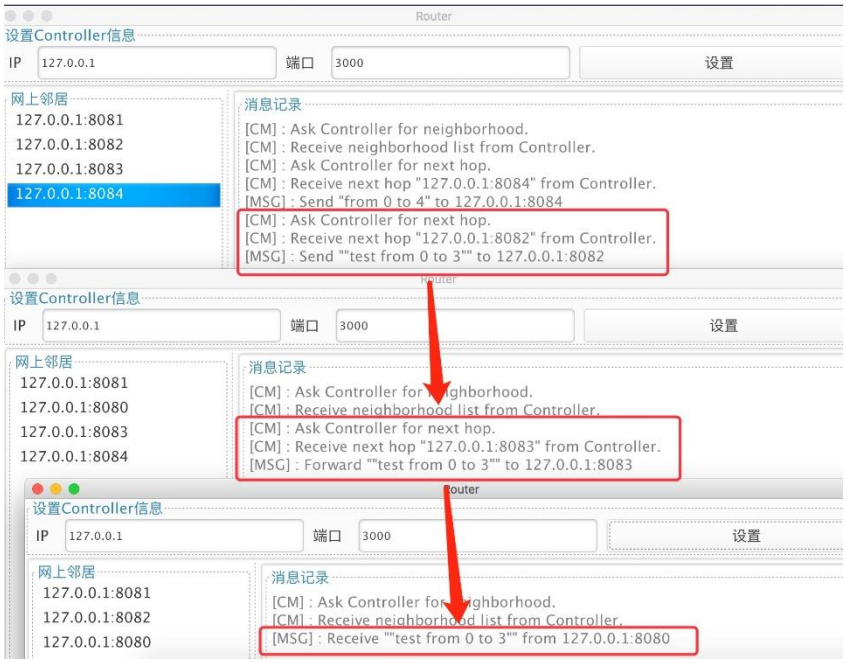


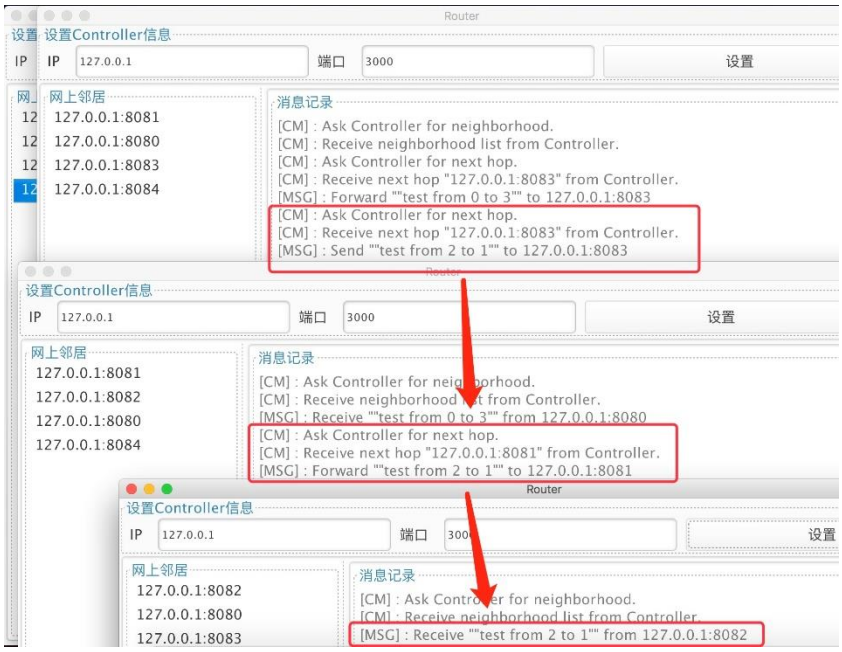

IP
端口
添加
网上邻居
127.0.0.1:8084
127.0.0.1:8081
127.0.0.1:8083
127.0.0.1:8082
消息记录
[RT] : Receive route table from 127.0.0.1:8084
[RT] : Send route table to 127.0.0.1:8084
[RT] : Receive route table from 127.0.0.1:8084
[RT] : Send route table to 127.0.0.1:8084
[RT] : Receive route table from 127.0.0.1:8084
[RT] : Send route table to 127.0.0.1:8084
[RT] : Receive route table from 127.0.0.1:8084
[RT] : Send route table to 127.0.0.1:8084
[MSG] : Send "test 0 to 2" to 127.0.0.1:8082
Router
[MSG] : Receive "test 0 to 2" from 127.0.0.1:8080


网上邻居
127.0.0.1:8084
127.0.0.1:8080
127.0.0.1:8082
127.0.0.1:8083
消息记录
[RT] : Send route table to 127.0.0.1:8084
[RT] : Receive route table from 127.0.0.1:8084
[RT] : Send route table to 127.0.0.1:8084
[RT] : Send route table to 127.0.0.1:8083
[RT] : Receive route table from 127.0.0.1:8083
[RT] : Receive route table from 127.0.0.1:8083
[RT] : Send route table to 127.0.0.1:8084
[MSG] : Forward "test 4 to 3" to 127.0.0.1:8083
[MSG] : Send "test 4 to 3" to 127.0.0.1:8083
Router
端口
添加
[RT] : Receive route table from 127.0.0.1:8081
[RT] : Send route table to 127.0.0.1:8081
[RT] : Send route table to 127.0.0.1:8081
[RT] : Send route table to 127.0.0.1:8082
[MSG] : Receive "test 4 to 3" from 127.0.0.1:8084

任务 2：集中式路由


Router
设置Controller信息
IP 127.0.0.1
端口 3000
设置
网上邻居
127.0.0.1:8081
127.0.0.1:8082
127.0.0.1:8083
127.0.0.1:8084
消息记录
[CM] : Ask Controller for neighborhood.
[CM] : Receive neighborhood list from Controller.
[CM] : Ask Controller for next hop.
[CM] : Receive next hop "127.0.0.1:8084" from Controller.
[MSG] : Send "from 0 to 4" to 127.0.0.1:8084
127.0.0.1:8080
[MSG] : Receive "from 0 to 4" from 127.0.0.1:8080


Router
设置Controller信息
IP 127.0.0.1
端口 3000
设置
网上邻居
127.0.0.1:8081
127.0.0.1:8082
127.0.0.1:8083
127.0.0.1:8084
127.0.0.1:8080
消息记录
[CM] : Ask Controller for neighborhood.
[CM] : Receive neighborhood list from Controller.
[CM] : Ask Controller for next hop.
[CM] : Receive next hop "127.0.0.1:8084" from Controller.
[MSG] : Send "from 0 to 4" to 127.0.0.1:8084
[CM] : Ask Controller for next hop.
[CM] : Receive next hop "127.0.0.1:8082" from Controller.
[MSG] : Send ""test from 0 to 3"" to 127.0.0.1:8082
[CM] : Receive next hop "127.0.0.1:8083" from Controller.
[MSG] : Forward ""test from 0 to 3"" to 127.0.0.1:8083
[MSG] : Receive ""test from 0 to 3"" from 127.0.0.1:8080


Router
设置Controller信息
IP 127.0.0.1
端口 3000
设置
网上邻居
127.0.0.1:8081
127.0.0.1:8080
127.0.0.1:8083
127.0.0.1:8084
127.0.0.1:8082
消息记录
[CM] : Ask Controller for neighborhood.
[CM] : Receive neighborhood list from Controller.
[CM] : Ask Controller for next hop.
[CM] : Receive next hop "127.0.0.1:8083" from Controller.
[MSG] : Forward ""test from 0 to 3"" to 127.0.0.1:8083
[MSG] : Send ""test from 2 to 1"" to 127.0.0.1:8083
[MSG] : Receive ""test from 0 to 3"" from 127.0.0.1:8080
[CM] : Ask Controller for next hop.
[CM] : Receive next hop "127.0.0.1:8081" from Controller.
[MSG] : Forward ""test from 2 to 1"" to 127.0.0.1:8081
[MSG] : Receive ""test from 2 to 1"" from 127.0.0.1:8082

5. 团队分工

团队成员	工作任务	小组自评
黄建武	编写接口文档	96
陈平永	VirtualRouter 处理路由表(processRouterTable)方法的实现	96
蔡桐钊	Controller 处理节点请求	96
骆铭涛	GUI 开发	96
陈海涛	Message 类的实现	96
何晋豪	Controller 计算路由表(Dijkstra)的实现	96
洪创煌	任务 1 和任务 2 的 Socket 通信，消息传送等过程的实现	96
童云钊	实验报告、PPT	96