《现代交换原理》实验报告

实验名称 **MPLS 编程实验**

班　　级 **2020211314**

学　　号 **2020211502**

姓　　名 王小龙

指导教师 赵　学　达

实验三　　MPLS 编程实验

一、实验目的

三个编程实验主要用于加强学生对 MPLS 交换中标记请求、标记分配与分发、标记分组转发的理解。

二、实验内容和实验步骤（简写）

实验内容：

多协议标记交换 MPLS（Multiple Protocol Labeled Switching）技术是将第二层交换和第三层路由结合起来的一种 L2/L3 集成数据传输技术。MPLS 是一项面向连接的交换技术，因此有建立连接的过程。各个 MPLS 设备运行 路由协议，在标记分发协议 LDP 的控制下根据计算得到的路由在相邻的路由器进行标记分配和分发，从而通过标记的拼接建立起从网络入口到出口的标记交换路径 LSP。

在数据转发过程中，入口标记路由器 LER 根据数据流的属性比如网络层目的地址等将分组映射到某一转发等价类 FEC，并为分组绑定标记。核心标记交换路由器 LSR 只需根据分组中所携带的标记进行转发即可。出口标记路由器 LER 弹出标记，根据分组的网络层目的地址将分组转发到下一跳。MPLS 节点（MPLS 标记交换路由器 LSR 或 MPLS 边缘路由器 LER）均要创建和维护传统的路由表和标记信息库 LIB。

路由表记录记录路由信息，用于转发网络层分组和标记分发从而建立标记交换路径。LIB 记录了本地节点分配的标记与从邻接 MPLS 节点收到的标记之间的映射关系，用于标记分组的转发。

MPLS 技术的核心实质在于：（1）网络中分组基于标记的转发（2）LDP 协议控制下的进行标记分发从而建立标记交换路径 LSP。

本次实验中的三个小实验就是基于上述原理设计的。

实验步骤：

实验一　标记请求实验：

1、理解实验要求和所给数据结构定义：

```
1：标记请求实验要求函数：

extern  "C" _declspec(dllexport) struct ReqType req_process(int idnow,struct routertype routenow)

{

 struct ReqType reqtemp;

  return reqtemp;

}

参数意义：

int idnow：当前的节点号；

struct routertype routenow：当前所指的路由表的表项；

函数要求：根据提供的当前节点号和路由表表项值产生标记请求包；

过程描述：

   标记请求包的源节点号由当前节点号提供，目的节点号和ip地址前缀由当前所指的路由表表项的下一跳节点和ip地址前缀提供；
```

通过所给实验内容和所给函数及其相关数据结构可知，我们需要完成对类型为 ReqType 的结构体 reqtemp 的赋值操作，该结构体为发送的请求信息包数据结构，包括了 iFirstNode，iEndNode，ipaddress 三个成员变量，分别代表请求信息包的源节点，请求信息包的目的节点，请求信息包包含的网络层目的 IP 地址前缀。实验的目的就是完成该结构体变量的赋值。

```
//发送的请求信息包数据结构
struct ReqType
{
       int iFirstNode;      //请求信息包的源节点
       int iEndNode;      //请求信息包的目的节点
      double ipaddress;  //请求信息包包含的网络层目的IP地址前缀（例如197.42）
};
```

2.编写代码

所给函数中有两个参数，分别为 idnow 和 routenow。idnow 是当前的节点号，利用它可以完成对请求信息包的源节点 reqtemp.iFirstNode 的赋值；routenow 是一个结构体，它当中包含的 nexthop 成员变量代表下一跳节点，通过它可以为请求信息包的目的节点 reqtemp.iEndNode 进行赋值，另外一个成员变量 ipaddress 代表网络层目的地址前缀，利用它可以完成对请求信息包包含的网络层目的 IP 地址前缀 reqtemp.ipaddress 的赋值。由上述就可以写出代码。

实验二　标记分配与分发实验

1、理解实验要求和所给数据结构定义

```
2：标记分配与分发实验：
extern  "C" _declspec(dllexport) struct funcusedtype label_process(struct routertype routenow,int labelout,int idnow)
{
  struct funcusedtype tempstruct;
  return tempstruct;
}
```

通过所给实验内容和所给函数及其相关数据结构可知，我们需要完成对类型为 funcusedtype 的结构体 tempstruct 的赋值操作，该结构体包含两个成员变量，分别为 libinfo，labelinfo，上述两个参数分别由 libtype 结构体和 LabelPack 结构体定义，这两个结构体分别表示标记信息表表项的数据结构和发送的标记信息包数据结构。

```
struct  libtype
{
     double ipaddress;   //网络层目的地址前缀
      int  inpoint;     //入端口号
      int  outpoint;    //出端口号
      int  inlabel;     //入标记值
      int  outlabel;    //出标记值
};
```

```
//发送的标记信息包数据结构
struct  LabelPack
{
  int iFirstNode;  //源节点号
  int iEndNode;  //目的节点号
  int labelvalue;  //标签值
};
```

```
struct  funcusedtype
{
struct libtype libinfo;    //包含的标记信息表项
struct LabelPack labelinfo; //包含的标记信息包数据结构
};
```

2、编写代码

所给函数中有三个参数，分别为 routenow，labelout，idnow。

routenow 结构体变量表示当前所指的路由表表项，它当中包含的 ipaddress 代表网络层目的地址前缀，inpoint 代表入端口号，outpoint 代表出端口号，利用它们可以分别完成对 libinfo.ipaddress，lininfo.inpoint，lininfo.outpoint 的赋值。入标记值 libinfo.inlabel 可以再 1-9 随机选取一个值。

出标记值 lininfo.outlabel 可以通过参数 labelout 进行赋值。

labelinfo.iFirstNode 值为可由参数 idnow 进行赋值，目的节点号 labelinfo.iEndNode 可由上一跳节点 routenow.lasthop 进行赋值，因为该报文是由下游返回上游的，所以标签值 labelinfo.labelvalue 可由 libinfo.inlabel 进行赋值。

由上述即可写出该函数。

实验三　标记分组转发实验

1、理解实验要求和所给数据结构定义

```
3. 标记分组转发实验

extern  "C"  _declspec(dllexport) struct LabelledDataPack pack_process(struct routertype routenow,struct libtype
libnow,int idnow)

{

 struct LabelledDataPack packtemp;

 return packtemp;

}
```

通过所给实验内容和所给函数及其相关数据结构可知，我们需要完成对类型为 LabelledDataPack 的结构体 someLabelledDataPack 的赋值操作，该结构体包含三个成员变量，分别为 iFirstNode，iEndNode，DataInfo，上述中的 DataInfo 由结构体 MessageType 定义，该结构体表示标记分组类型。

```
//发送的标记分组信息包类型

struct  LabelledDataPack

{

  int iFirstNode;                //源节点号

  int iEndNode;                //目的节点号

  struct MessageType  DataInfo;  //包含的标记分组类型信息

};
```

```
//标记分组类型

struct MessageType

{

  double ipaddress;   //网络层目的地址前缀

  int labelvalue;     //输出标签值

};
```

2、编写代码

所给函数中有三个参数 routenow,libnow,idnow。

routenow 代表当前所指的路由表表项，可以通过它包含的下一跳节点 nexthop 和网络层目的地址前缀 ipaddress 来对 someLabelledDataPack.iEndNode 和 DataInfo.ipaddress 进行赋值。

idnow 表示当前的节点号，利用它可以对 someLabelledDataPack.iFirstNode 进行赋值。

libnow 为当前的标签信息表表项，可以通过它包含的一个成员变量 outlabel 出标记值来对 DataInfo.labelvalue 进行赋值。

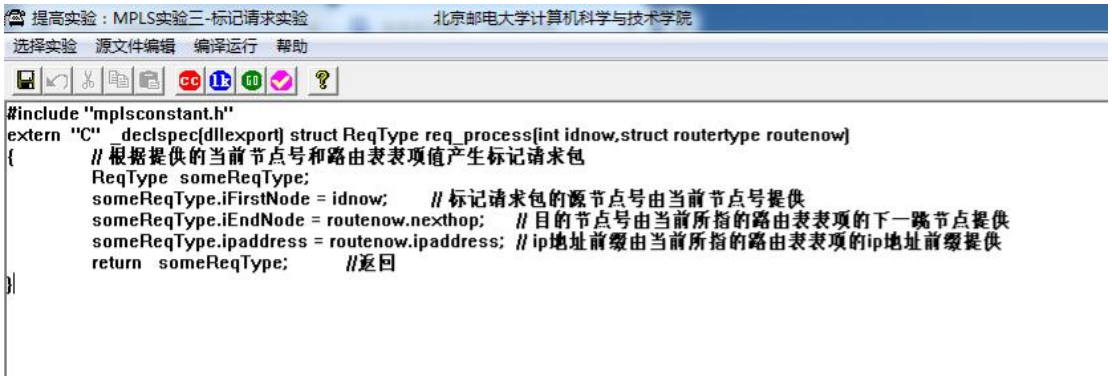
由上述就可以写出函数的代码。

三、源代码（注明代码含义）

实验一源代码：

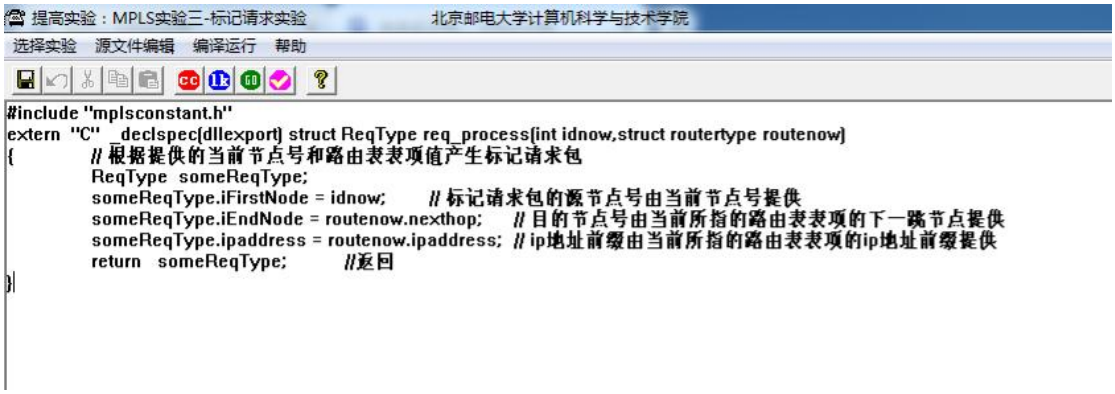

```
#include "mplsconstant.h"
extern "C" _declspec(dllexport) struct ReqType req_process(int idnow,struct routertype routenow)
{           // 根据提供的当前节点号和路由表表项值产生标记请求包
            ReqType  someReqType;
            someReqType.iFirstNode = idnow;       // 标记请求包的源节点号由当前节点号提供
            someReqType.iEndNode = routenow.nexthop;     // 目的节点号由当前所指的路由表表项的下一跳节点提供
            someReqType.ipaddress = routenow.ipaddress;  // ip地址前缀由当前所指的路由表表项的ip地址前缀提供
            return   someReqType;         //返回
}
```

实验二源代码：




```
#include "mplsconstant.h"
extern "C" _declspec(dllexport) struct ReqType req_process(int idnow,struct routertype routenow)
{           // 根据提供的当前节点号和路由表表项值产生标记请求包
            ReqType  someReqType;
            someReqType.iFirstNode = idnow;       // 标记请求包的源节点号由当前节点号提供
            someReqType.iEndNode = routenow.nexthop;     // 目的节点号由当前所指的路由表表项的下一跳节点提供
            someReqType.ipaddress = routenow.ipaddress;  // ip地址前缀由当前所指的路由表表项的ip地址前缀提供
            return   someReqType;         //返回
}
```

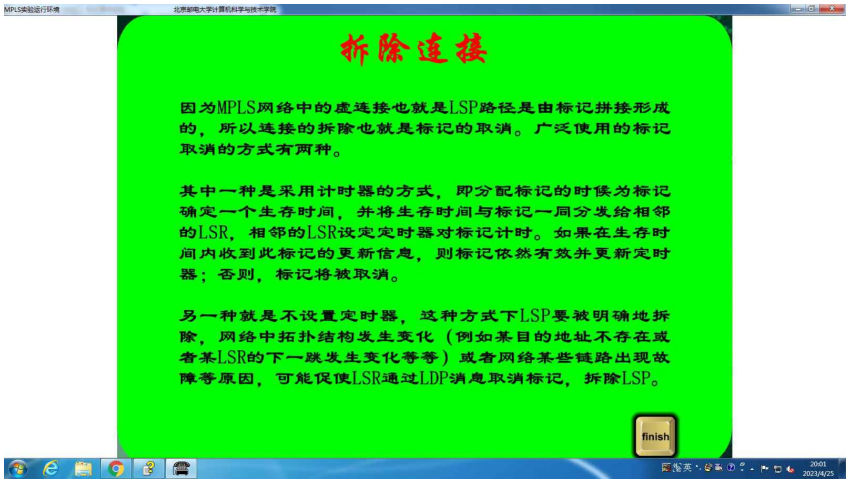
实验三源代码：




```
#include "mplsconstant.h"
extern "C" _declspec(dllexport) struct ReqType req_process(int idnow,struct routertype routenow)
{           // 根据提供的当前节点号和路由表表项值产生标记请求包
            ReqType  someReqType;
            someReqType.iFirstNode = idnow;       // 标记请求包的源节点号由当前节点号提供
            someReqType.iEndNode = routenow.nexthop;     // 目的节点号由当前所指的路由表表项的下一跳节点提供
            someReqType.ipaddress = routenow.ipaddress;  // ip地址前缀由当前所指的路由表表项的ip地址前缀提供
            return   someReqType;         //返回
}
```

四、实验结果

由于三次实验验证结果一致，故只做一次说明：



五、实验心得

通过本次实验，增强了我对 MPLS 交换中标记请求、标记分配与分发、标记分组转发的理解，对 MPLS 技术的一些关键点有了新的认识，通过三次实验的编码，我对标记请求，标记分配与分发和标记分组转发的实现过程有了更深的体会。在这次实验中，我不仅锻炼了编写代码的能力，同时也增长了很多关于时间调度表的知识，收获很多。