1.已经实现的功能列表：

1）找到他的邻居，了解他们的端口和链接的成本。

2）产生序列号和报文，报文包含其链路状态，将此包发送到其他路由器。

3）定时像邻居节点发送心跳报文

4）收到包后，检查是否是重复的包，将被丢弃，否则将被发送到源以外的路由器。

5）计算每个路由器的最短路径。例如，任何使用dijkstra算法的路由器都可以找到从它到另一个路由器的最短路径。

6）在一段时间内未接收到相邻点的信息后，确定连接失败并发送到其他路由器。

2。数据结构：

对象：edge存储相邻节点的成本信息，是一个字典，相邻点之间保存一个成本。

Edge(v1,v2)=c

当v1，v2代表相邻点时，c代表成本。

三。包格式的数据结构：

序列号+相邻路由信息

你如何处理错误？

当路由器在连续一段时间内没有收到来自相邻路由器的心跳时，确定两个路由器之间的连接错误，生成新消息并发送到相邻路由器，更新本地网络拓扑。

我们如何避免发送太多广播？

发送方接收有关链接状态的消息，并将其发送到源以外的路由器。

记录收到的序列号。如果消息重复，则会在不发送的情况下丢弃该消息。

三

使用字典的数据结构，查找快，空间大。

可能的改进和扩展

在多线程操作本地保存的全局链路状态信息时，没有对全局变量进行加锁。可能会导致计算最短路径时读入脏数据，结果错误。

可能的改进就是多线程对全局变量加锁。

4.代码参考

广播的发送方：

#!usr/bin/env python

import socket

host=''

port=10000

s=socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM)

s.setsockopt(socket.SOL\_SOCKET,socket.SO\_REUSEADDR,1)

s.setsockopt(socket.SOL\_SOCKET,socket.SO\_BROADCAST,1)

s.bind((host,port))

while 1:

try:

data,addr=s.recvfrom(1024)

print "got data from",addr

s.sendto("broadcasting",addr)

print data

except KeyboardInterrupt:

raise

广播的接收方：

#!/usr/bin/env python

import socket,sys

addr=('<broadcast>',10000)

s=socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM)

s.setsockopt(socket.SOL\_SOCKET,socket.SO\_BROADCAST,1)

s.sendto("hello from client",addr)

while 1:

data=s.recvfrom(1024)

if not data:

break

print data