计算机网络实验报告

本文是计算机网络——路由算法实验的实验报告。

1 实验目的

学习和掌握距离向量算法

2 实验环境

CentOS 6.2 + GCC 4.4.6
Win10+Python3.7

3 实验内容

3.1 内容

编程实现并分析以下过程:
模拟路由收敛
模拟拓扑变化
制造路由回路
抑制路由回路

3.2 实验原理

3.2.1 路由器的功能

1、路由选择（Routing）

选择一条正确的路径——寻找下一跳（next hop），同时使目的可达且路径最优（距离最短、延迟最小、费用最低……）。最后建立路由表（包括距离向量、链路状态、路径向量等）

2、转发（Forwarding）

根据路由选择的结果，将数据包从输入接口转发至输出接口

3.2.2 DV 算法基本思想

1、使用“距离”度量路由

路由表保存到达各目标的最短距离及下一跳

2、使用“距离向量”交换路由信息

相邻路由器之间交换路由表，各自计算最佳路由——到达目标的最短距离及下一跳

3、所有路由器两两定时交换，将路由信息扩散至全网，最后达到收敛状态

3.2.3 DV 算法

1、定义：

adj(i) 为节点 i 的所有相邻节点的集合

c(i, n) 为一对相邻节点 i 和 n 之间的距离

d(i, j) 为从节点 i 到节点 j 之间的最短距离

2、公式：

$d(i, i) = 0$

$d(i, j) = \min[c(i, n) + d(n, j)]$。其中 $n \in adj(i)$

c(i, n) 为初始条件，已知

d(n, j) 为节点 i 从邻居节点 n 获知的路由信息，已知

4 实验内容

4.1 任务 1：模拟路由收敛

已知网络拓扑如图所示，请使用 DV 算法模拟该网络的迭代收敛过程。

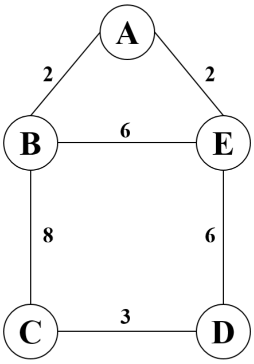


图 1　一个简单的网络拓扑结构

4.1.1　程序要求

建议使用 python3 编程；使用 socket 编程实现分布式；每次迭代后（每隔 Interval，如 30s），各节点输出路由表，输出格式可参考本课件“算法示例：一次迭代后的路由表”；输出收敛后的路由表，即输出每对节点间的最短距离和下一跳。

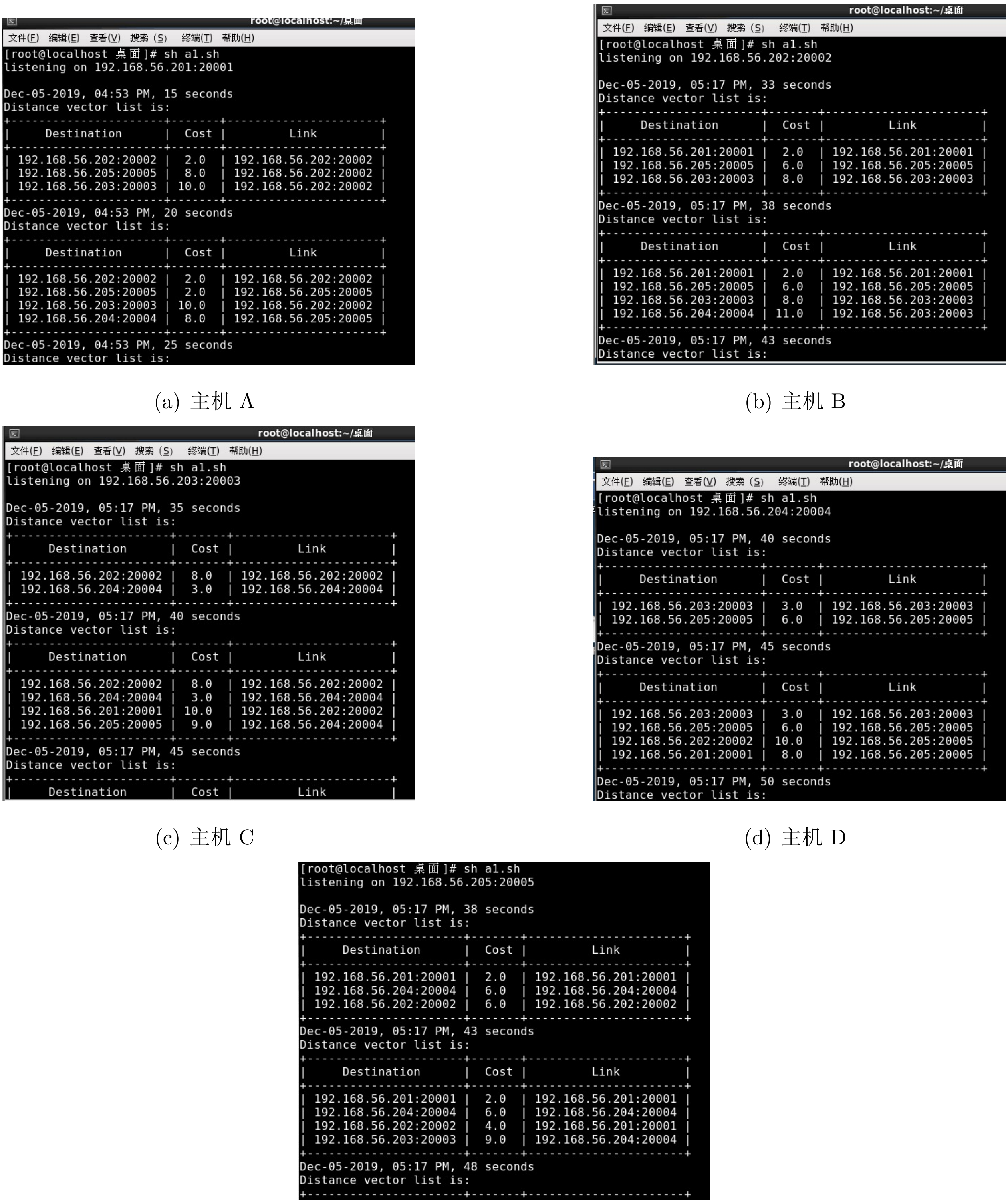
4.1.2　源程序

```python
import socket
import sys
import time
import threading
import copy

""" thread that will call a function every interval seconds """
class RepeatTimer(threading.Thread):

    def __init__(self, interval, target):
        threading.Thread.__init__(self)
        self.target = target
        self.interval = interval
        self.daemon = True
        self.__flag = threading.Event()
        self.__flag.set()  # True
        self.__running = threading.Event()
        self.__running.set()  # running--True

    def run(self):
        while self.__running.isSet():
            self.__flag.wait()  # True--return, False--blocking until True
            self.target()
            time.sleep(self.interval)

    def pause(self):
        self.__flag.clear()  # False--set blocking

    def resume(self):
        self.__flag.set()  # True--stop blocking

    def stop(self):
        self.__flag.set()
        self.__running.clear()  # False


’’’ getting the program parameters ’’’
def parse_argv():
    # Get the parameter list
    s = sys.argv[1:]
    length = len(s)
    # Incomplete information
    if length <= 0 or (length - 1) % 3 != 0:
        print(”error: parameters must be :”
              ”python dvroute.py <listening-port> <ip-address1 port1 distance1> <ip-address2 port2 distance2> ……”)
        return False

    parsed1 = {}
    port = s.pop(0)
    # Get the listening port number
    try:
        parsed1[’port’] = int(port)
    except ValueError:
        print(”error: port values must be integers. {0} is not an int.”.format(port))
        return False

    # {’port’: xxx, ’neighbors’:[addr1,addr2,addr3],  ’costs’:[cost1,cost2,cost3]}
    parsed1[’neighbors’] = []
    parsed1[’costs’] = []
    while len(s):
        ip = s.pop(0)
        port = s.pop(0)
        try:
            port = int(port)
            parsed1['neighbors'].append((ip, port))
        except ValueError:
            print("error: port values must be integers. {0} is not an int.".format(port))
            return False
        distance = s.pop(0)
        try:
            distance = float(distance)
            parsed1['costs'].append(distance)
        except ValueError:
            print("error: link distance values must be numbers. {0} is not a number.".format(distance))
            return False
    return parsed1


""" recalculate inter-node path costs using bellman ford algorithm """
def update_costs(data, addr):
    # Gets the distance from the adjacent router
    dis = neighbors[addr][0]
    # Traverse the route table received
    for address in data.keys():
        if address == host_addr:
            # we don't need to update the distance to ourselves
            continue
        else:
            # iterate through neighbors and find cheapest route
            if address not in routing.keys():
                # If a node listed in costs is not in our list of nodes
                # join the routing table
                routing[address] = [dis + data[address][0], addr]
            else:
                if routing[address][1] == addr:
                    # The next hop is 'addr'
                    # update route table
                    routing[address][0] = dis + data[address][0]
                else:
                    # To the destination network 'address', but the next hop address is not 'addr'
                    if data[address][0] + dis < routing[address][0]:
                        # 'addr' is a closer link
                        # update route table
                        routing[address] = [data[address][0] + dis, addr]


''' Receive routing information'''
def recv_costs():
    while True:
        try:
            data, addr = skt.recvfrom(4096)
            data = eval(data.decode('utf-8'))
            if isinstance(data, dict):
                # DICTIONARY is receved
                # it is routing information
                # update route table
                if addr not in neighbors.keys():
                    # Do not process messages from routers not a neighbor
                    # (consider link disconnection)
                    continue
                # update route table
                update_costs(data, addr)
                # update the updated-time of the neighbors
                neighbors[addr][1] = time.time()
            else:
                # LIST is receved
                # it is a command
                # call to the modify link function
                r_cmd = data[0]
                r_parsed = data[1]
                if r_cmd == 'linkdown':
                    linkdown(r_parsed)
                elif r_cmd == 'linkup':
                    linkup(r_parsed)
                else:
                    linkchange(r_parsed)
        except ConnectionError:
            # print(skt.gettimeout())
            # print("远程主机强迫关闭了一个现有的连接。")
            pass


''' Send routing information '''
```

```python
def send_costs():
    for address in neighbors.keys():
        # Dictionary -> byte stream
        skt.sendto(str(routing).encode('utf-8'), address)


''' Check the update time of your neighbor's router every THREE seconds '''
def check_neighbors():
    while True:
        # The present time
        now_time = time.time()
        # Traverse the adjacent router table
        for address in list(neighbors.keys()):
            if now_time - neighbors[address][1] > 6 * Interval:
                # The last update of a router was too long
                # removes it from the adjacent router table
                neighbors.pop(address)
                # Traverse
                for add in list(routing.keys()):
                    # 'Address is the next hop address of a destination network in the routing table
                    # the routing item needs to be removed
                    if routing[add][1] == address:
                        routing.pop(add)
        # sleep
        time.sleep(3)

""" display routing info: cost to destination; route to take """
def showrt():
    print(formatted_now())
    print("Distance vector list is:")
    print("+----------------------+-------+----------------------+")
    print("|      Destination     |  Cost |         Link         |")
    print("+----------------------+-------+----------------------+")

    for address in routing.keys():
        print ("|{destination:^22}|{cost:^7}|{nexthop:^22}|".format(
            destination=str(address),
            cost=routing[address][0],
            next=str(routing[address][1])))
    print("+----------------------+-------+----------------------+") # extra line
```

```python

if __name__ == '__main__':
    # Verify that parameters are correct
    parsed = parse_argv()
    if parsed == False:
        sys.exit(1)
    # print(parsed)

    # Different programs have different localhost, which needs to be manually modified within the program
    # 不同的程序对应的localhost不同，需要在程序内手动修改
    localhost = '127.0.0.1'
    # time between two transmissions, interval
    Interval = 30
    skt = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    host_addr = (localhost, parsed['port'])
    skt.bind(host_addr)
    # skt.setblocking(True)
    print('UDP Server on %s:%s...' % (host_addr[0], host_addr[1]))

    # Adjacent router: {adjacent router address: [distance, last update time]}
    neighbors = {}
    for i in range(len(parsed['neighbors'])):
        neighbors[parsed['neighbors'][i]] = [parsed['costs'][i], time.time()]  # + timeout * 3

    # Routing table dictionary, {destination address: [distance, next hop address]}
    routing = {host_addr: [0, host_addr]}

    for i in range(len(parsed['neighbors'])):
        # Routing table dictionary, {destination address: [distance, next hop address]}
        routing[parsed['neighbors'][i]] = [parsed['costs'][i], parsed['neighbors'][i]]
    # print(routing)

    # Send routing table information regularly
    ts = RepeatTimer(interval=Interval, target=send_costs)
    # Receive routing table information regularly
    tr = threading.Thread(target=recv_costs, daemon=True)  # 守护线程，当主线程结束时，停止接收子线程
    # Print routing table information regularly
    t_showrt = RepeatTimer(interval=Interval, target=showrt)
    # Periodically check routing table information
    t_check = threading.Thread(target=check_neighbors, daemon=True)  # 守护线程，当主线程结束时，停止检查
    ts.start()
    tr.start()
    t_showrt.start()
    t_check.start()

    cmds = (’linkdown’, ’linkup’, ’linkchange’)
    while True:
        cmd = input()
        # print(cmd)
        if cmd in cmds:
            # Pause printing routing information
            t_showrt.pause()
            if cmd == ’linkdown’:
                parsed = input(”link down : ”)
                ad_other = linkdown(parsed)
            elif cmd == ’linkup’:
                parsed = input(”link up : ”)
                ad_other = linkup(parsed)
            else:
                parsed = input(”link change : ”)
                ad_other = linkchange(parsed)
            if ad_other != False:
                # Send it to another router and change the routing table of the other party
                skt.sendto(str([cmd, ad_other[1]]).encode(’utf-8’), ad_other[2])
            # Restore printing routing information
            t_showrt.stopped = False
            t_showrt.resume()
        if cmd == ’close’:
            break
    skt.close()
```

4.1.3 运行结果



```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.201:20001

Dec-05-2019, 04:53 PM, 15 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.203:20003 | 10.0  | 192.168.56.202:20002 |
+----------------------+-------+----------------------+
Dec-05-2019, 04:53 PM, 20 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  2.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 | 10.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  8.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
Dec-05-2019, 04:53 PM, 25 seconds
Distance vector list is:
```

(a) 主机 A

```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.202:20002

Dec-05-2019, 05:17 PM, 33 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 |  8.0  | 192.168.56.203:20003 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 38 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 |  8.0  | 192.168.56.203:20003 |
| 192.168.56.204:20004 | 11.0  | 192.168.56.203:20003 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 43 seconds
Distance vector list is:
```

(b) 主机 B

```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.203:20003

Dec-05-2019, 05:17 PM, 35 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  3.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 40 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  3.0  | 192.168.56.204:20004 |
| 192.168.56.201:20001 | 10.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  9.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 45 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
```

(c) 主机 C

```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.204:20004

Dec-05-2019, 05:17 PM, 40 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 45 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.202:20002 | 10.0  | 192.168.56.205:20005 |
| 192.168.56.201:20001 |  8.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 50 seconds
Distance vector list is:
```

(d) 主机 D

```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.205:20005

Dec-05-2019, 05:17 PM, 38 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  6.0  | 192.168.56.202:20002 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 43 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      | Cost  |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  4.0  | 192.168.56.201:20001 |
| 192.168.56.203:20003 |  9.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
Dec-05-2019, 05:17 PM, 48 seconds
Distance vector list is:
+----------------------+-------+----------------------+
```

(e) 主机 E

图 2 模拟路由收敛运行结果

4.2 任务 2：模拟拓扑变化

在任务 1 的网络收敛后，将 B 和 E 之间的距离由 6 改为 2（好消息！），模拟该变化导致的重新收敛过程。

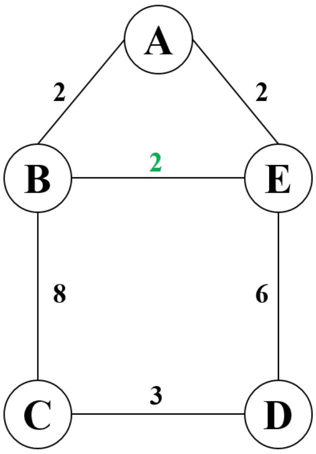


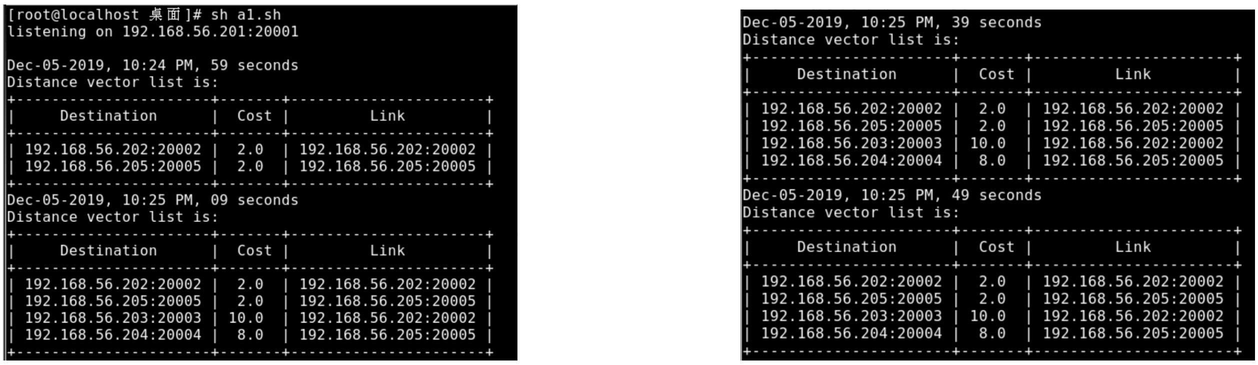
图 3 网络拓扑结构变化

4.2.1 源程序

添加函数如下：

```python
''' change the link '''
def linkchange(parsed):
    parsed = parsed.split()
    length = len(parsed)
    # incompleted information
    if length != 3:
        print("error: parameters must be :<neighbor-ip> <port> <link-cost> ")
        return False

    ip = parsed[0]
    port = parsed[1]
    distance = parsed[2]
    try:
        port = int(port)
    except ValueError:
        print("error: port values must be integers. {0} is not an int.".format(port))
        return False
    try:
        distance = float(distance)
    except ValueError:
        print("error: link distance values must be numbers. {0} is not a number.".format(distance))
        return False

    ad_temp = (ip, port)
    if ad_temp in neighbors.keys():
        neighbors[ad_temp] = [distance, time.time()]
        # comparison
        # change the routing table if the next hop address of the original routing path is ad_temp
        if routing[ad_temp][1] == ad_temp:
            # Modify the routing table dictionary, {destination address: [distance, next hop address]}
            routing[ad_temp] = [distance, ad_temp]
        elif routing[ad_temp][0] >= distance:
            # Compare the distance if the next hop address of the original routing path is not ad_temp (that is, arrive indirectly)
            # update if the distance is smaller
            # {destination address: [distance, next hop address]}
            routing[ad_temp] = [distance, ad_temp]
        print("the cost of {0} has been changed.".format(ad_temp))
        return ["linkchange", str(host_addr[0]) + ' ' + str(host_addr[1]) + ' ' + str(distance), ad_temp]
    else:
        print("there is no {0} in neighbors.".format(ad_temp))
        return False
```

4.2.2 运行结果



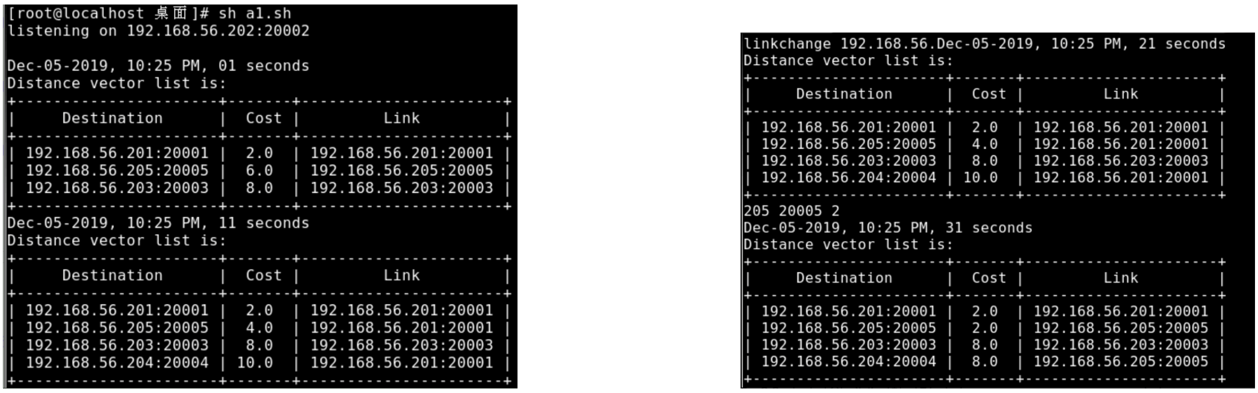
```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.201:20001

Dec-05-2019, 10:24 PM, 59 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  2.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
Dec-05-2019, 10:25 PM, 09 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  2.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 | 10.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  8.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
```

(a) 初始收敛

```
Dec-05-2019, 10:25 PM, 39 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  2.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 | 10.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  8.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
Dec-05-2019, 10:25 PM, 49 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.205:20005 |  2.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 | 10.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  8.0  | 192.168.56.205:20005 |
+----------------------+-------+----------------------+
```

(b) 改变路径长度后

图 4　主机 A



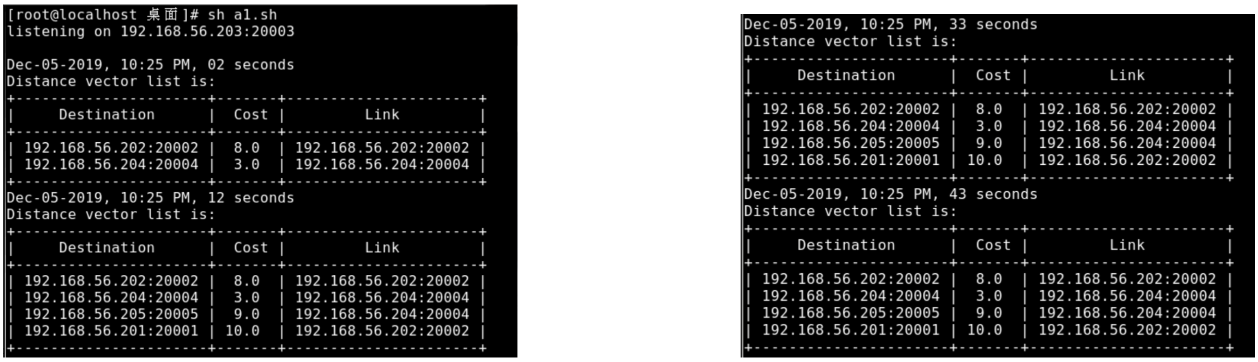
```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.202:20002

Dec-05-2019, 10:25 PM, 01 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 |  8.0  | 192.168.56.203:20003 |
+---------------------+-------+---------------------+
Dec-05-2019, 10:25 PM, 11 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.205:20005 |  4.0  | 192.168.56.201:20001 |
| 192.168.56.203:20003 |  8.0  | 192.168.56.203:20003 |
| 192.168.56.204:20004 | 10.0  | 192.168.56.201:20001 |
+---------------------+-------+---------------------+
```

(a) 初始收敛

```
linkchange 192.168.56.Dec-05-2019, 10:25 PM, 21 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.205:20005 |  4.0  | 192.168.56.201:20001 |
| 192.168.56.203:20003 |  8.0  | 192.168.56.203:20003 |
| 192.168.56.204:20004 | 10.0  | 192.168.56.201:20001 |
+---------------------+-------+---------------------+
205 20005 2
Dec-05-2019, 10:25 PM, 31 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.205:20005 |  2.0  | 192.168.56.205:20005 |
| 192.168.56.203:20003 |  8.0  | 192.168.56.203:20003 |
| 192.168.56.204:20004 |  8.0  | 192.168.56.205:20005 |
+---------------------+-------+---------------------+
```

(b) 改变路径长度后

图 5　主机 B



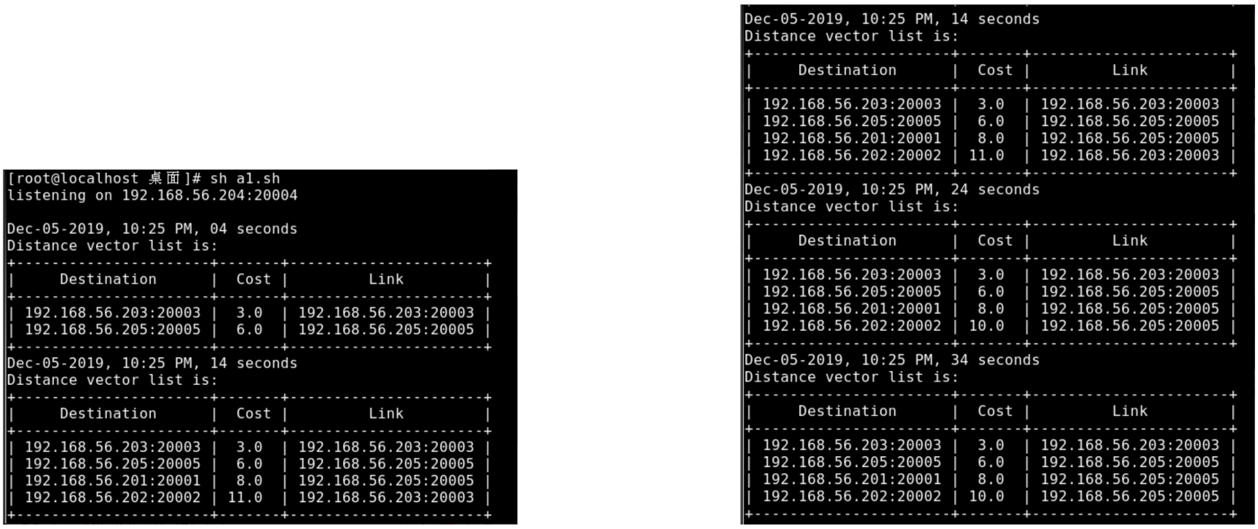
```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.203:20003

Dec-05-2019, 10:25 PM, 02 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.202:20002 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  3.0  | 192.168.56.204:20004 |
+---------------------+-------+---------------------+
Dec-05-2019, 10:25 PM, 12 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.202:20002 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  3.0  | 192.168.56.204:20004 |
| 192.168.56.205:20005 |  9.0  | 192.168.56.204:20004 |
| 192.168.56.201:20001 | 10.0  | 192.168.56.202:20002 |
+---------------------+-------+---------------------+
```

(a) 初始收敛

```
Dec-05-2019, 10:25 PM, 33 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.202:20002 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  3.0  | 192.168.56.204:20004 |
| 192.168.56.205:20005 |  9.0  | 192.168.56.204:20004 |
| 192.168.56.201:20001 | 10.0  | 192.168.56.202:20002 |
+---------------------+-------+---------------------+
Dec-05-2019, 10:25 PM, 43 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.202:20002 |  8.0  | 192.168.56.202:20002 |
| 192.168.56.204:20004 |  3.0  | 192.168.56.204:20004 |
| 192.168.56.205:20005 |  9.0  | 192.168.56.204:20004 |
| 192.168.56.201:20001 | 10.0  | 192.168.56.202:20002 |
+---------------------+-------+---------------------+
```

(b) 改变路径长度后

图 6　主机 C



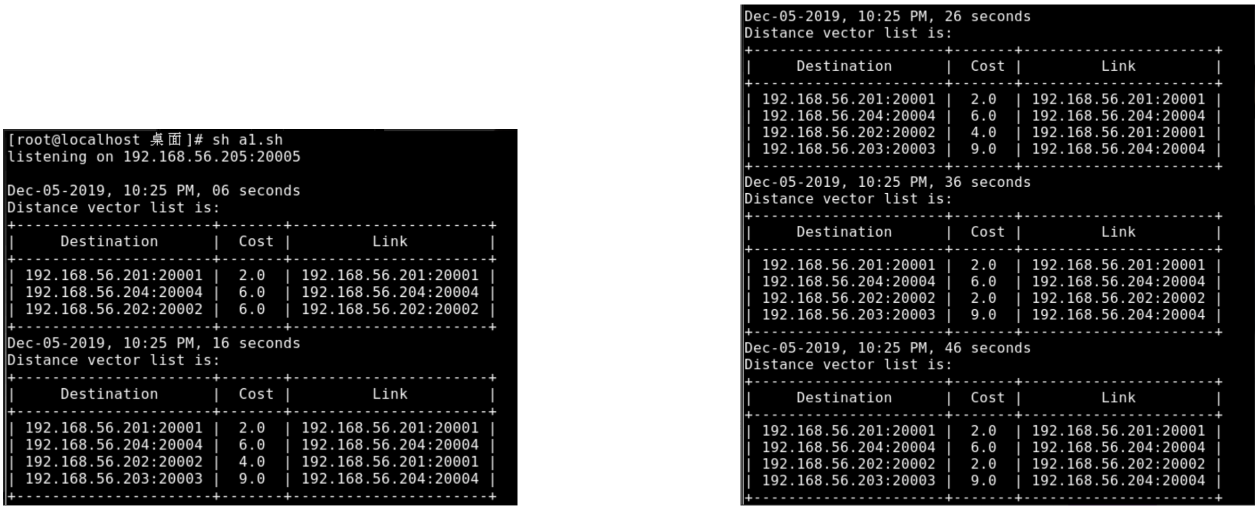
```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.204:20004

Dec-05-2019, 10:25 PM, 04 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
+---------------------+-------+---------------------+
Dec-05-2019, 10:25 PM, 14 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.201:20001 |  8.0  | 192.168.56.205:20005 |
| 192.168.56.202:20002 | 11.0  | 192.168.56.203:20003 |
+---------------------+-------+---------------------+
```

(a) 初始收敛

```
Dec-05-2019, 10:25 PM, 14 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.201:20001 |  8.0  | 192.168.56.205:20005 |
| 192.168.56.202:20002 | 11.0  | 192.168.56.203:20003 |
+---------------------+-------+---------------------+
Dec-05-2019, 10:25 PM, 24 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.201:20001 |  8.0  | 192.168.56.205:20005 |
| 192.168.56.202:20002 | 10.0  | 192.168.56.205:20005 |
+---------------------+-------+---------------------+
Dec-05-2019, 10:25 PM, 34 seconds
Distance vector list is:
+---------------------+-------+---------------------+
|     Destination     | Cost  |         Link        |
+---------------------+-------+---------------------+
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
| 192.168.56.205:20005 |  6.0  | 192.168.56.205:20005 |
| 192.168.56.201:20001 |  8.0  | 192.168.56.205:20005 |
| 192.168.56.202:20002 | 10.0  | 192.168.56.205:20005 |
+---------------------+-------+---------------------+
```

(b) 改变路径长度后

图 7　主机 D



```
[root@localhost 桌面]# sh a1.sh
listening on 192.168.56.205:20005

Dec-05-2019, 10:25 PM, 06 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  6.0  | 192.168.56.202:20002 |
+----------------------+-------+----------------------+
Dec-05-2019, 10:25 PM, 16 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  4.0  | 192.168.56.201:20001 |
| 192.168.56.203:20003 |  9.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
```

(a) 初始收敛

```
Dec-05-2019, 10:25 PM, 26 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  4.0  | 192.168.56.201:20001 |
| 192.168.56.203:20003 |  9.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
Dec-05-2019, 10:25 PM, 36 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.203:20003 |  9.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
Dec-05-2019, 10:25 PM, 46 seconds
Distance vector list is:
+----------------------+-------+----------------------+
|     Destination      |  Cost |         Link         |
+----------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.204:20004 |  6.0  | 192.168.56.204:20004 |
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.203:20003 |  9.0  | 192.168.56.204:20004 |
+----------------------+-------+----------------------+
```

(b) 改变路径长度后

图 8　主机 E

4.3　任务 3：制造路由回路

将左图拓扑的 A 和 B 连接断开（坏消息!），模拟该变化导致的重新收敛过程。

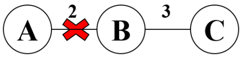


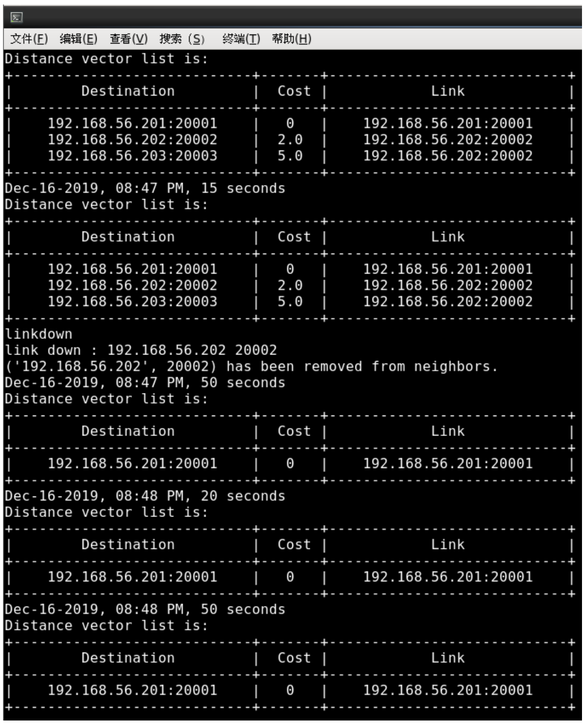
图 9　路由回路

4.3.1　源程序

添加函数如下：

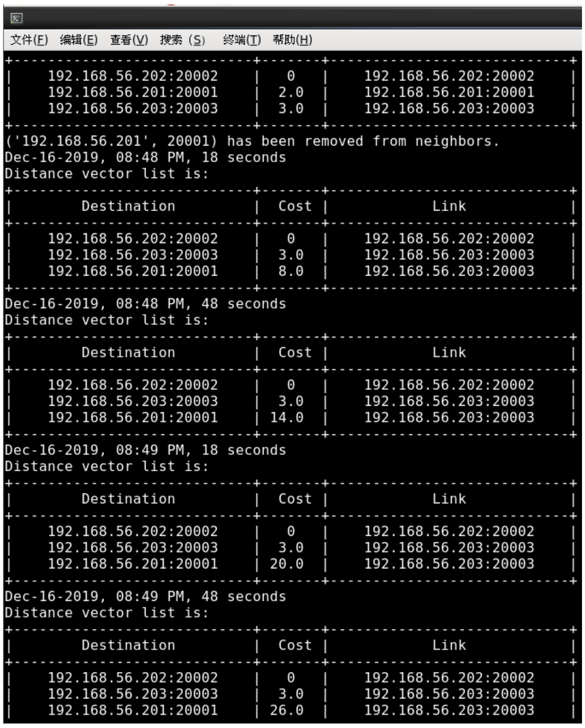
```python
''' disconnect the link'''
def linkdown(parsed):
    # split by the blank
    parsed = parsed.split()
    length = len(parsed)
    # incompleted information
    if length != 2:
        print("error: parameters must be :<neighbor-ip> <port> ")
        return False

    ip = parsed[0]
    port = parsed[1]
    try:
        port = int(port)
    except ValueError:
        print(”error: port values must be integers. {0} is not an int.”.format(port))
        return False

    ad_temp = (ip, port)
    if ad_temp in neighbors.keys():
        neighbors.pop(ad_temp)
        for address in list(routing.keys()):
            # If the next hop address of a destination network in the routing table is ad_temp
            # remove
            if routing[address][1] == ad_temp:
                routing.pop(address)
        print(”{0} has been removed from neighbors.”.format(ad_temp))
        return [”linkdown”, str(host_addr[0]) + ’ ’ + str(host_addr[1]), ad_temp]
    else:
        # no operation addresses in the adjacent router table
        print(”there is no {0} in neighbors.”.format(ad_temp))
        return False
```

4.3.2 运行结果



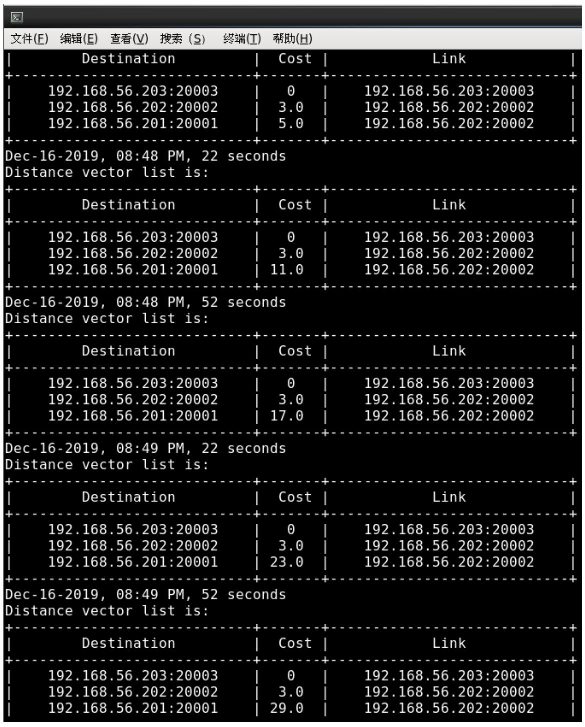
```
Distance vector list is:
+----------------------------+-------+----------------------------+
|        Destination         |  Cost |            Link            |
+----------------------------+-------+----------------------------+
|    192.168.56.201:20001    |   0   |    192.168.56.201:20001    |
|    192.168.56.202:20002    |  2.0  |    192.168.56.202:20002    |
|    192.168.56.203:20003    |  5.0  |    192.168.56.202:20002    |
+----------------------------+-------+----------------------------+
Dec-16-2019, 08:47 PM, 15 seconds
Distance vector list is:
+----------------------------+-------+----------------------------+
|        Destination         |  Cost |            Link            |
+----------------------------+-------+----------------------------+
|    192.168.56.201:20001    |   0   |    192.168.56.201:20001    |
|    192.168.56.202:20002    |  2.0  |    192.168.56.202:20002    |
|    192.168.56.203:20003    |  5.0  |    192.168.56.202:20002    |
+----------------------------+-------+----------------------------+
linkdown
link down : 192.168.56.202 20002
('192.168.56.202', 20002) has been removed from neighbors.
Dec-16-2019, 08:47 PM, 50 seconds
Distance vector list is:
+----------------------------+-------+----------------------------+
|        Destination         |  Cost |            Link            |
+----------------------------+-------+----------------------------+
|    192.168.56.201:20001    |   0   |    192.168.56.201:20001    |
+----------------------------+-------+----------------------------+
Dec-16-2019, 08:48 PM, 20 seconds
Distance vector list is:
+----------------------------+-------+----------------------------+
|        Destination         |  Cost |            Link            |
+----------------------------+-------+----------------------------+
|    192.168.56.201:20001    |   0   |    192.168.56.201:20001    |
+----------------------------+-------+----------------------------+
Dec-16-2019, 08:48 PM, 50 seconds
Distance vector list is:
+----------------------------+-------+----------------------------+
|        Destination         |  Cost |            Link            |
+----------------------------+-------+----------------------------+
|    192.168.56.201:20001    |   0   |    192.168.56.201:20001    |
+----------------------------+-------+----------------------------+
```

图 10 主机 A



```
+-------------------------------+--------+--------------------------------+
|     192.168.56.202:20002      |   0    |     192.168.56.202:20002       |
|     192.168.56.201:20001      |  2.0   |     192.168.56.201:20001       |
|     192.168.56.203:20003      |  3.0   |     192.168.56.203:20003       |
+-------------------------------+--------+--------------------------------+
('192.168.56.201', 20001) has been removed from neighbors.
Dec-16-2019, 08:48 PM, 18 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.202:20002      |   0    |     192.168.56.202:20002       |
|     192.168.56.203:20003      |  3.0   |     192.168.56.203:20003       |
|     192.168.56.201:20001      |  8.0   |     192.168.56.203:20003       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:48 PM, 48 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.202:20002      |   0    |     192.168.56.202:20002       |
|     192.168.56.203:20003      |  3.0   |     192.168.56.203:20003       |
|     192.168.56.201:20001      |  14.0  |     192.168.56.203:20003       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:49 PM, 18 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.202:20002      |   0    |     192.168.56.202:20002       |
|     192.168.56.203:20003      |  3.0   |     192.168.56.203:20003       |
|     192.168.56.201:20001      |  20.0  |     192.168.56.203:20003       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:49 PM, 48 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.202:20002      |   0    |     192.168.56.202:20002       |
|     192.168.56.203:20003      |  3.0   |     192.168.56.203:20003       |
|     192.168.56.201:20001      |  26.0  |     192.168.56.203:20003       |
```

图 11　主机 B



```
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.203:20003      |   0    |     192.168.56.203:20003       |
|     192.168.56.202:20002      |  3.0   |     192.168.56.202:20002       |
|     192.168.56.201:20001      |  5.0   |     192.168.56.202:20002       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:48 PM, 22 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.203:20003      |   0    |     192.168.56.203:20003       |
|     192.168.56.202:20002      |  3.0   |     192.168.56.202:20002       |
|     192.168.56.201:20001      |  11.0  |     192.168.56.202:20002       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:48 PM, 52 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.203:20003      |   0    |     192.168.56.203:20003       |
|     192.168.56.202:20002      |  3.0   |     192.168.56.202:20002       |
|     192.168.56.201:20001      |  17.0  |     192.168.56.202:20002       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:49 PM, 22 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.203:20003      |   0    |     192.168.56.203:20003       |
|     192.168.56.202:20002      |  3.0   |     192.168.56.202:20002       |
|     192.168.56.201:20001      |  23.0  |     192.168.56.202:20002       |
+-------------------------------+--------+--------------------------------+
Dec-16-2019, 08:49 PM, 52 seconds
Distance vector list is:
+-------------------------------+--------+--------------------------------+
|          Destination          |  Cost  |              Link              |
+-------------------------------+--------+--------------------------------+
|     192.168.56.203:20003      |   0    |     192.168.56.203:20003       |
|     192.168.56.202:20002      |  3.0   |     192.168.56.202:20002       |
|     192.168.56.201:20001      |  29.0  |     192.168.56.202:20002       |
```

图 12　主机 C

4.4　任务 4：解决路由回路

如果使用逆向毒化技术，重新模拟 A 和 B 链接断开所导致的重新收敛过程。

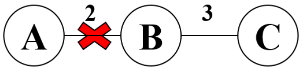


图 13　路由回路

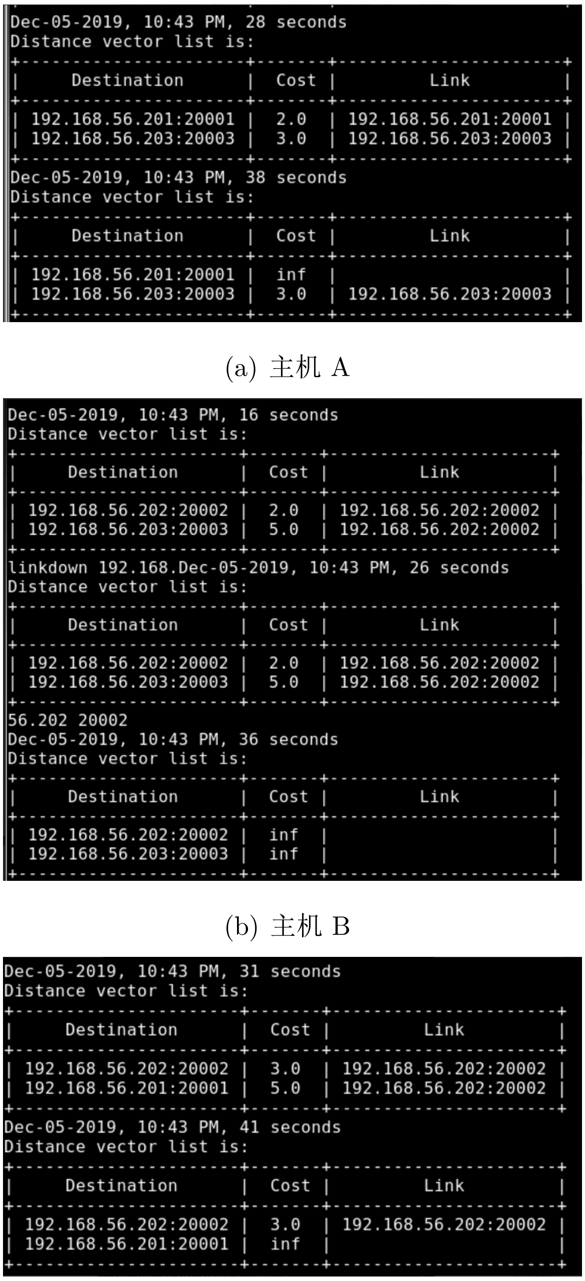
4.4.1　源程序

修改部分函数：

```
""" recalculate inter-node path costs using bellman ford algorithm """
def update_costs(data, addr):
    # Gets the distance from the adjacent router
    dis = neighbors[addr][0]
    # Traverse the route table received
    for address in data.keys():
        if address == host_addr:
            # we don't need to update the distance to ourselves
            continue
        else:
            # iterate through neighbors and find cheapest route
            if address not in routing.keys():
                # If a node listed in costs is not in our list of nodes
                # join the routing table
                routing[address] = [dis + data[address][0], addr]
            else:
                if routing[address][1] == addr:
                    # The next hop is 'addr'
                    # update route table
                    routing[address][0] = dis + data[address][0]
                else:
                    # To the destination network 'address', but the next hop address is not 'addr'
                    if data[address][0] + dis < routing[address][0]:
                        # 'addr' is a closer link
                        # update route table
                        routing[address] = [data[address][0] + dis, addr]
        if routing[address][0] == float('inf'):
            routing[address][1] = None

''' Receive routing information'''
def recv_costs():
    while True:
        try:
            data, addr = skt.recvfrom(4096)
            data = eval(data.decode('utf-8'))
            if isinstance(data, dict):
                # DICTIONARY is receved
                # it is routing information
                # update route table
                if addr not in neighbors.keys():
                    # Do not process messages from routers not a neighbor
                    # (consider link disconnection)
                    continue
                for ad in data.keys():
                    # set float'inf'
                    if data[ad][0] == 99999:
                        data[ad][0] = float('inf')

                # update route table
                update_costs(data, addr)
                # update the updated-time of the neighbers
                neighbors[addr][1] = time.time()
            else:
                # LIST is receved
                # it is a command
                # call to the modify link function
                r_cmd = data[0]
                r_parsed = data[1]
                if r_cmd == 'linkdown':
                    linkdown(r_parsed)
                elif r_cmd == 'linkup':
                    linkup(r_parsed)
                else:
                    linkchange(r_parsed)
        except ConnectionError:
            # print(skt.gettimeout())
            # print("远程主机强迫关闭了一个现有的连接。")
            pass

def send_costs():
    for address in neighbors.keys():
        # Copy
        routing_poison_reverse = copy.deepcopy(routing)
            #Set the routing table to unreachable for routing items obtained
                from adjacent routers       for ad in routing_poison_reverse.
```

```
keys():
            if routing_poison_reverse[ad][1] == address or routing_poison_reverse[ad][0] == float('inf'):
                # This routing item is obtained from the adjacent router
                # Or this route is not accessible
                # Float 'inf' converted to STR 'inf' cannot be converted back to float'inf'
                routing_poison_reverse[ad][0] = 99999
                # so it is replaced by a large number. routing_poison_reverse[ad][0] = 99999
        # Dictionary -> byte stream
        skt.sendto(str(routing_poison_reverse).encode('utf-8'), address)

''' disconnect the link'''
def linkdown(parsed):
    # split by the blank
    parsed = parsed.split()
    length = len(parsed)
    # incompleted information
    if length != 2:
        print("error: parameters must be :<neighbor-ip> <port> ")
        return False

    ip = parsed[0]
    port = parsed[1]
    try:
        port = int(port)
    except ValueError:
        print("error: port values must be integers. {0} is not an int.".format(port))
        return False

    ad_temp = (ip, port)
    if ad_temp in neighbors.keys():
        neighbors.pop(ad_temp)
        for address in list(routing.keys()):
            # If the next hop address of a destination network in the routing table is ad_temp
            # remove
            if routing[address][1] == ad_temp:
                routing[address] = [float('inf'), None]

        print("{0} has been removed from neighbors.".format(ad_temp))
        return ["linkdown", str(host_addr[0]) + ' ' + str(host_addr[1]), ad_temp]
    else:
        # no operation addresses in the adjacent router table
        print("there is no {0} in neighbors.".format(ad_temp))
        return False
```

4.4.2 运行结果



```
Dec-05-2019, 10:43 PM, 28 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.201:20001 |  2.0  | 192.168.56.201:20001 |
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
+---------------------+-------+----------------------+
Dec-05-2019, 10:43 PM, 38 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.201:20001 |  inf  |                      |
| 192.168.56.203:20003 |  3.0  | 192.168.56.203:20003 |
+---------------------+-------+----------------------+
```

(a) 主机 A

```
Dec-05-2019, 10:43 PM, 16 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.203:20003 |  5.0  | 192.168.56.202:20002 |
+---------------------+-------+----------------------+
linkdown 192.168.Dec-05-2019, 10:43 PM, 26 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.202:20002 |  2.0  | 192.168.56.202:20002 |
| 192.168.56.203:20003 |  5.0  | 192.168.56.202:20002 |
+---------------------+-------+----------------------+
56.202 20002
Dec-05-2019, 10:43 PM, 36 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.202:20002 |  inf  |                      |
| 192.168.56.203:20003 |  inf  |                      |
+---------------------+-------+----------------------+
```

(b) 主机 B

```
Dec-05-2019, 10:43 PM, 31 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.202:20002 |  3.0  | 192.168.56.202:20002 |
| 192.168.56.201:20001 |  5.0  | 192.168.56.202:20002 |
+---------------------+-------+----------------------+
Dec-05-2019, 10:43 PM, 41 seconds
Distance vector list is:
+---------------------+-------+----------------------+
|     Destination     | Cost  |         Link         |
+---------------------+-------+----------------------+
| 192.168.56.202:20002 |  3.0  | 192.168.56.202:20002 |
| 192.168.56.201:20001 |  inf  |                      |
+---------------------+-------+----------------------+
```

(c) 主机 C

图 14 逆向毒化后不会产生 loop

5 思考

5.1 演示说明

使用 UDP socket 实现 DV 信息的交换:

python client.py <listening-port> <ip-address1 port1 distance1> <ip-address2 port2 distance2> ……

本节点通过 localhost：listening-port 进行监听（采用大端口如 20000）

邻居关系由三元组定义：IP 地址、监听端口号、与本节点的距离

拓扑改变所需功能:

linkchange <neighbor-ip> <port> <link-cost> 改变边的大小（任务 2）

linkdown <neighbor-ip> <port> 取消和这个节点的链接（任务 3、4）

linkup <neighbor-ip> <port> 恢复以前使用 linkdown 取消的链接（便于调试）

本实验中在每个虚拟机中分别编写 shell 文件，并运行。

```
#命令行中键入：sh a1.sh
#A
python test2.py 20001 192.168.56.202 20002 2 192.168.56.205 20005 2

#B
python test2.py 20002 192.168.56.201 20001 2 192.168.56.205 20005 6 192.168.56.203 20003 8

#C
python test2.py 20003 192.168.56.202 20002 8 192.168.56.204 20004 3

#D
python test2.py 20004 192.168.56.203 20003 3 192.168.56.205 20005 6

#E
python test2.py 20005 10 192.168.56.201 20001 2 192.168.56.204 20004 6 192.168.56.202 20002 6

#linkchange in B
linkchange 192.168.56.205 20005 2
```

任务 3、4 中使用的 shell 文件：

```
#命令行中键入：sh a2.sh
#A
python test2.py 20001 192.168.56.202 20002 2

```

```
#B
python test2.py 20002 192.168.56.201 20001 2 192.168.56.203 20003 3

#C
python test2.py 20003 192.168.56.202 20002 3

#linkdown in A
linkdown 192.168.56.202 20002
```

5.2 思考题解答

假设现有如图 15所示的网络。[1] 现在结点 C 至结点 D 之间的链路断了，并且假设原来从 B 到 D 的最短路径为 B—A—C—D，于是 B 将会从它的角度告知 C 这条最短路径。那么，在这种情形下，即使使用了毒性逆转技术，C 还是会选取 B 作为到达 D 的下一跳。这样，一个回环形成了。

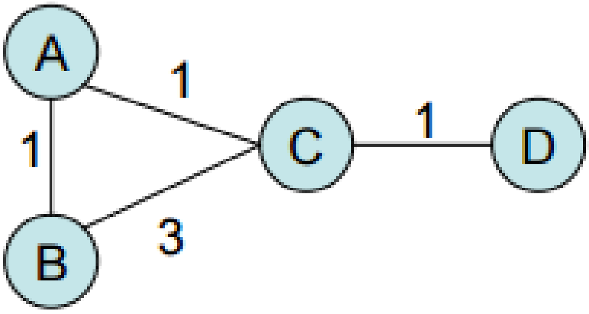


图 15　一个简单的网络拓扑结构

6 小结

通过本次实验，我熟悉了 DV 算法的执行过程以及针对 DV 算法存在的问题的解决方法。在实验过程中，主要需要注意更新前后的路由表的保存不能混乱，以尽可能模拟同步性。在编写毒性逆转时，需要对这个技术有详细的了解，在查阅相关资料后，解决了这个问题。总之，这次实验对于我理解 RIP 协议和 DV 算法很有帮助。

[1] https://people.mpi-sws.org/~gummadi/teaching/sp07/datanets/homework/homework2solution.pdf