MONASH UNIVERSITY
DEPARTMENT OF ELECTRICAL & COMPUTER SYSTEMS ENGINEERING

**Performance of Telecommunication Networks**

**Assignment II: Distance-Vector Routing Algorithm**

Y. Ahmet Şekercioğlu

# 1 Overview

In this assignment, you will be writing a new network layer for our packet switching network simulation model [Şek06a] to implement the *distributed, asynchronous* distance-vector (Bellman-Ford) routing algorithm [Sta02, Pages 413–414] for arbitrary network topologies of up to 100 nodes[1].

# 2 Procedure

In the *delay and queueing in packet switching networks* experiment [Şek06a], we use a set of models to simulate a network of hosts and routers. If you examine the `networklayer.cc` file, you see that I actually cheat, and use the tricks provided by the OMNeT++ framework to fill the routing tables before the simulated time starts ticking. In a real network there is no such possibility, and routers need to communicate with each other (by sending routing information packets) to progressively build their routing tables.

This means that, in our simulated packet switching network, there will be two types of packets flowing in the links: data and routing packets. the packet format we use in the experiment mentioned in the above paragraph is very simple and has no `type` or `data` fields. For this reason, we need to change the packet format to distinguish the routing packets from the ones carrying data and to include a payload field for exchanging the routing table information. The following packet format should be sufficient for the purpose of implementing the distance-vector routing algorithm in our packet switching network:

```
enum PayloadTypes
{
    routing = 1;
    data    = 2;
};

//
// Represents a packet in the network.
//
```

[1] I have placed this arbitrary limit to keep the C++ coding simple. Entire routing table information should fit in a single routing packet.

```
message Packet
{
  fields:
    int srcAddr;
    int destAddr;
    int hopCount;
    int payloadType enum(PayloadTypes);
    unsigned char data[200];
}
```

You now have a way of distinguishing the routing packets from ordinary data packets arriving at the routers by checking the value of the `payloadType` field (Chapter 5 of the OMNeT++ User Manual [Vár06] has good examples). To use this, we need to make a simple modification in the `app_gen.cc`. Since current implementation of the application layer does not mark the data packets, we need to insert the following statement

```
pk->setPayloadType(data);
```

after the statement

```
pk->setDestAddr(destAddress);
```

in the `handleMessage()` routine of `app_gen.cc`. Every data packet generated by the hosts should now be properly marked. After these changes, you can easily detect the type of every incoming packet to a router by checking the value of the `payloadType` field.

Assume that all links have an equal cost of 1 (i.e. we want to minimize the number of router hops).

You are now ready to work on your assignment. You may try your algorithm on a network topology like the one in Figure 1 of the *introductory network layer protocols: hot potato routing* experiment [Şek06b].

## 3 Submission Requirements

Please pay attention to following points:

1. At the end of the simulation, dump the contents of the routing tables. The display should conform to the format given in the following example:

```
router[1] final routing table:
destination   cost   next hop
  address
-----------   ----   --------
     0          2    router[0]
     1          1    router[0]
     2          1    host[2]
     3          1    host[3]
     4          2    router[0]
```

2. Submit only your own `networklayer.cc` as a basic *ASCII text* file through the Web submission page. Do *not* submit anything else!

3. I will test your submission over my own sample network topologies (I will not make them available).

4. You cannot use any global variables or OMNeT++'s own topology discovery routines. All the communication between routers has to be done by exchanging packets.

## References

[Şek06a] Y. A. Şekercioğlu. Delay and Queueing in Packet Switching Networks Experiment for the Performance of Telecommunication Networks Unit, 2006. (available online) http://titania.ctie.monash.edu.au/netperf/netperf-omnetpp-pkswitch.pdf.

[Şek06b] Y. A. Şekercioğlu. Introductory Network Layer Protocols: Hot Potato Routing Experiment for the Performance of Telecommunication Networks Unit, 2006. (available online) http://titania.ctie.monash.edu.au/netperf/netperf-omnetpp-tictocnet.pdf.

[Sta02] W. Stallings. *High-Speed Networks and Internets: Performance and Quality of Service*. Prentice Hall, $2^{nd}$ edition, 2002.

[Vár06] A. Várga. OMNeT++ Object-Oriented Discrete Event Simulation System User Manual. URL reference http://www.omnetpp.org/doc/manual/usman.html, 2006.