COMP3331 Week 8 Lecture 2

Network Layer

# Distance vector: Bellman-Ford equation

Let

dx(y): = cost of least-cost path from x to y

then

dx(y) = dx(y)=minv{c(x,v)+dv(y)},

where:

* c is the cost to a neighbour v
* d is the cost from neighbour to destination y

Graphical user interface, text, application, email

Description automatically generated

Exam:

The question will be asked on a very small network, probably <= 5 routers in the network. We will be asked to fill in the table for a particular update of table, and we will need to be able to fill in the initialisation table.

**For each neighbour *v*, the cost *c*(*x, v*) from *x* to directly attached neighbour, *v*. In DV, routers do not have knowledge of the entire topology, just their direct neighbours**

Diagram

Description automatically generated

Initially, each forwarding table contains the information of its neighbours. For example, from Node B, we can reach A, C, and D, and we fill in 2, 1 and 3 respectively (direct link). For the rest of the table we just use infinity.

Diagram

Description automatically generated

Now C sends update to A, C has a min distance vector of {7,1,0,1}. In table A, we only look at via C column. We know that A can reach C via C with a cost of 7, we add this to the distance vector sent by C, we have {14,8,7,8}, we only need to update row B and row D.

Diagram, schematic

Description automatically generated

Similarly, when B sends update to A, we are only looking at column via B, the cost to B via B is 2, we add this to the min distance vector sent by B, we have {4,2,3,5}. We then update table A to get {-,2,3,5}

* Initial state: best one-hop paths
* One simultaneous round: best two-hop paths
* Two simultaneous rounds: best three-hop paths
* Kth simultaneous round: best (k+1) hop paths

**Convergence** is the time during which all routers come to an agreement about the best paths through the internetwork

## Link Cost Change

**Good news travels fast: the tables can converge very quickly**

Diagram

Description automatically generated

**Bad news causes “counting to infinity” problem: table takes a very long time to converge**

Diagram

Description automatically generated with medium confidence

**We can use poisoned reverse to partially solve the counting to infinity problem, however, it is not guaranteed.**

Diagram

Description automatically generated

## Comparison of link state and distance vector

We have seen that LS requires each node to know the cost of each link in the network. This requires O(|N| |E|) messages to be sent. Also, whenever a link cost changes, the new link cost must be sent to all nodes. The DV algorithm requires message exchanges between directly connected neighbors at each iteration. We have seen that the time needed for the algorithm to converge can depend on many factors. When link costs change, the DV algorithm will propagate the results of the changed link cost only if the new link cost results in a changed least-cost path for one of the nodes attached to that link.

The convergence delay depends on the number of nodes and edges, as well as the link weight for DV. If a router goes down, the LS can advertise incorrect link cost and it converges quickly. For DV the error propagate through the network and it will take long time to converge.

Graphical user interface, text

Description automatically generated with medium confidence

# ICMP: Internet Control Message Protocol

The Internet Control Message Protocol (ICMP), specified in [RFC 792], is used by hosts and routers to communicate network-layer information to each other. The most typical use of ICMP is for **error reporting**.

Timeline

Description automatically generated

# Data Link Layer Introduction

data-link layer has responsibility of transferring datagram from one node to physically adjacent node over a link, concepts to be covered in this section

* error detection, correction
* sharing a broadcast channel: multiple access
* link layer addressing
* local area networks: Ethernet

## Link layer services

* framing, link access:
  + encapsulate datagram into frame, adding header, trailer.
  + “MAC” addresses used in frame headers to identify source, destination.
* reliable delivery between adjacent nodes: wireless links: high error rates
  + we need reliable delivery for wireless links, therefore we have both link-level and end-end reliability
* flow control: pacing between adjacent sending and receiving nodes
* error detection: receiver detects presence of errors, e.g., CRC (next week)
* error correction: receiver identifies and corrects bit error(s) without resorting to retransmission
* half-duplex and full-duplex: with half duplex, nodes at both ends of link can transmit, but not at same time