Network Routing - I
(Without Failures)
Lecture 18
6.02 Spring 2010
April 12, 2010
• Forwarding. Distributed routing.
• Distance-vector protocol with Bellman-Ford step
• Link-state protocol with Dijkstra’s shortest-paths

How to find a good path (or paths) between any two nodes?

The Problem: Finding Paths
• Addressing (how to name nodes)
• Forwarding (how does a switch process a packet)
• Routing (building and updating data structures to ensure that forwarding works)
• Functions of the network layer

Routing Table Structure

<table>
<thead>
<tr>
<th>Destination</th>
<th>Link</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>L1</td>
<td>18</td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>L1</td>
<td>11</td>
</tr>
<tr>
<td>D</td>
<td>L0</td>
<td>4</td>
</tr>
<tr>
<td>E</td>
<td>L1</td>
<td>16</td>
</tr>
</tbody>
</table>

Table @ B

Shortest Path Routing
• Each node wants to find the path with minimum total cost to other nodes
• We use the term “shortest path” even though we’re interested in min cost (and not min #hops)
• Several possible distributed approaches
  • Vector protocols, esp. distance vector (DV)
  • Link-state protocols (LS)

Distributed Routing: A Common Plan
• Determining live neighbors
  • HELLO protocol (periodic) - next lecture
  • Common to both DV and LS protocols
• Advertisement step (periodic)
  • Send some information to all neighbors
• Integration step
  • Compute routing table using info from advertisements
### Distance Vector Routing

- Advertisement: Each node announces a vector of 
  \((\text{destination}: \text{pathcost})\) tuples to all its neighbors
- Integration: On hearing advertisement, run Bellman-Ford step: if \((\text{current cost to dest} + \text{cost in advertisement} + \text{linkcost})\): update cost, route

### Link-State Routing

- Advertisement step:
  - Information about its links to its neighbors
  - Neighbors re-send on their links → flooding
  - Result: Each node discovers map of the network
- Integration: Each node runs the same shortest path algorithm over its map
  - If each node implements computation correctly and each node has the same map, then routing tables will be correct
- Optimal substructure property:
  - Suppose shortest path from X to Y goes through Z. Then, the sub-path from X to Z must be a shortest path. [Why?]

### Integration Step: Dijkstra's Algorithm (Example)

Suppose we want to find paths from A to other nodes

### Link-State Advertisements and Flooding

- Periodically send LSA (Link-State Advertisement) 
  \([\text{seq}, (\text{nbhr1}, \text{linkcost1}), (\text{nbhr2}, \text{linkcost2}), ...]\) to all neighbors
- If seq > last_heard:
  - save seq, LSA; rebroadcast LSA to neighbors
- LSAs aren’t reliable messages, so periodic
- Periodic messages help handle dynamism: state in each node is “soft” and times out if not refreshed

### Summary

- The network layer implements the “glue” that achieves connectivity
  - Does addressing, forwarding, and routing
- Forwarding entails a routing table lookup; the table is built using routing protocol
- DV protocol: distributes route computation; each node advertises its best routes to neighbors
- LS protocol: distributes (floods) neighbor information; centralizes route computation using shortest-path algorithm