Lecture 13: Link-state Routing
Lecture 13 Overview

- Routing overview
- Intra vs. Inter-domain routing
- Link-state routing protocols
Router Tasks

● Forwarding
  ◆ Move packet from input link to the appropriate output link
  ◆ Purely local computation
  ◆ Must go be very fast (executed for every packet)

● Routing
  ◆ Make sure that the next hop actually leads to the destination
  ◆ Global decisions; distributed computation and communication
  ◆ Can go slower (only important when topology changes)
Forwarding Options

- **Source routing**
  - Complete path listed in packet

- **Virtual circuits**
  - Set up path out-of-band and store path identifier in routers
  - Local path identifier in packet

- **Destination-based forwarding**
  - Router looks up address in forwarding table
  - Forwarding table contains (address, next-hop) tuples
Source Routing

- **Routing**
  - Host computes path
    - Must know global topology and detect failures
  - Packet contains complete ordered path information
    - I.e. node A then D then X then J…
  - Requires variable length path header

- **Forwarding**
  - Router looks up next hop in packet header, strips it off and forwards remaining packet
    - Very quick forwarding, no lookup required

- **In practice**
  - ad hoc networks (DSR), some HPC networks (Myrinet), and for debugging on the Internet (LSR, SSR)
Virtual Circuits

- **Routing**
  - Hosts sets up path out-of-band, requires connection setup
  - Write (input id, output id, next hop) into each router on path
  - Flexible (one path per flow)

- **Forwarding**
  - Send packet with path id
  - Router looks up input, swaps for output, forwards on next hop
  - Repeat until reach destination
  - Table lookup for forwarding (why faster than IP lookup?)

- **In practice**
  - ATM: fixed VC identifiers and separate signaling code
  - MPLS: ATM meets the IP world (why? traffic engineering)
Routing
- All addresses are globally known
  » No connection setup
- Host sends packet with destination address in header
  » No path state; only routers need to worry about failure
- Distributed routing protocol used to routing tables

Forwarding
- Router looks up destination in table
  » Must keep state proportional to destinations rather than connections
- Lookup address, send packet to next-hop link
  » All packets follow same path to destination

In Practice: IP routing
The routing table at A, lists – *at a minimum* – the next hops for the different destinations.

<table>
<thead>
<tr>
<th>Dest</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>G</td>
<td>F</td>
</tr>
</tbody>
</table>
Routing on a Graph

- Essentially a graph theory problem
  - Network is a directed graph; routers are vertices
- Find “best” path between every pair of vertices
  - In the simplest case, best path is the shortest path

CSE 123 – Lecture 13: Link-state Routing
Routing Challenges

● How to choose best path?
  ◆ Defining “best” can be slippery

● How to scale to millions of users?
  ◆ Minimize control messages and routing table size

● How to adapt to failures or changes?
  ◆ Node and link failures, plus message loss
Intra-domain Routing

- Routing within a network/organization
  - A single administrative domain
  - The administrator can set edge costs

- Overall goals
  - Provide intra-network connectivity
  - Adapt quickly to failures or topology changes
  - Optimize use of network resources

- Non-goals
  - Extreme scalability
  - Lying, and/or disagreements about edge costs
Basic Approaches

- **Static**
  - Type in the right answers and hope they are always true
  - ...So far

- **Link state**
  - Tell *everyone* what you know about your *neighbors*
  - Today’s lecture!

- **Distance vector**
  - Tell your *neighbors* what you know about *everyone*
  - Next time…
Link-state Routing

- Two phases
  - Reliable flooding
    » Tell all routers what you know about your local topology
  - Path calculation (Dijkstra’s algorithm)
    » Each router computes best path over complete network

- Motivation
  - Global information allows optimal route computation
  - Straightforward to implement and verify
Broadcasting Link State

- **Reliable flooding**
  - Each router transmits a **Link State Packet** (LSP) on all links
  - A neighboring router forwards out all links except incoming
    » Keep a copy locally; don’t forward previously-seen LSPs

- **Challenges**
  - Packet loss
  - Out-of-order arrival

- **Solutions**
  - Acknowledgments and retransmissions
  - Sequence numbers
  - Time-to-live for each packet
Flooding Example

- LSP generated by X at T=0
- Nodes become orange as they receive it

Entire process repeats with LSPs for A, B, C, …
  - Actually in runs in parallel
Dijkstra’s Shortest Path Tree

- So you have all of these LSPs. Now what?
- Graph algorithm for single-source shortest path tree (find best route to all nodes)

\[
\begin{align*}
S & \leftarrow \{\} \\
Q & \leftarrow \text{remaining nodes keyed by distance} \\
\text{While } Q \neq \{} \\
& \quad u \leftarrow \text{extract-min}(Q) \quad u = \text{node with lowest cost} \\
& \quad S \leftarrow S \text{ plus } \{u\} \quad \leftarrow u \text{ is done} \\
& \quad \text{Within } Q: \\
& \quad \quad \text{for each node } v \text{ adjacent to } u \\
& \quad \quad \quad \text{“relax” the cost of } v \quad \text{is it cheaper to go through } u? 
\end{align*}
\]
Dijkstra Example – Step 1

- Green nodes are “confirmed”
- Yellow nodes are “tentative”
- We can add ourselves to “confirmed”
Example – Step 2

- Green nodes are “confirmed”
- Yellow nodes are “tentative”
- First look at neighbors
- “5” is cheaper than “10”
- We can confirm path with cost “5”
Example – Step 3

- Green nodes are “confirmed”
- Yellow nodes are “tentative”
- Update costs
- Look at 5’s neighbors
- 7 is cheapest
- We can confirm path with cost 7
Example – Step 4

- **Green** nodes are “confirmed”
- **Yellow** nodes are “tentative”
- Update costs
- 7 has no new neighbors
- 8 is cheapest
- We can confirm 8
Example – Step 5

- Green nodes are “confirmed”
- Yellow nodes are “tentative”
- Update costs
- No new neighbors
- 9 is cheapest
- We can confirm path with cost 9
Example – Done
For next time…

• Read Chapter 3.3.2 in P&D