Lecture 15:
Link state routing

CSE 123: Computer Networks
Aaron Schulman
Lecture 15 Overview

- Destination-based routing
- Intra-domain routing
  - Link-state routing
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 this is what we use for 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

![Graph Diagram]

X = router
\ = link
1 = cost

CSE 123 – Lecture 15: Link-state Routing
Routing Challenges

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

- How to scale to millions of destinations (addresses)?
  - Minimize control messages and routing table size

- How to adapt to failures or changes?
  - Node and link failures, plus message loss
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
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)

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

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

1. Green nodes are “confirmed”
2. Yellow nodes are “tentative”
3. First look at neighbors
4. “5” is cheaper than “10”
5. 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

CSE 123 – Lecture 15: Link-state Routing
Need to remove failed/old links from topology
- LSPs carry sequence numbers to distinguish new from old
- Routers only accept (and forward) the “newest” LSP
- Send a new LSP with cost infinity to signal a link down

But also need to remove entire routers
- Put a Time To Live (TTL) timer on each LSP
- TTL in every LSP, decremented periodically by each router
- When TTL = 0, purge the LSP and flood the network with an LSP with TTL 0 to tell everyone else to do the same
When to Flood?

- Triggered by a topology change
  - Link or node failure/recovery or
  - Configuration change like updated link metric
  - Converges quickly, but can cause flood of updates

- Periodically
  - Typically (say) every 30 minutes
  - Corrects for possible corruption of the data
  - Limits the rate of updates, but also failure recovery
Convergence

- Getting consistent routing information to all nodes
  - E.g., all nodes having the same link-state database
  - Until routing protocol converges, strange things happen…

- Consistent forwarding after convergence
  - All nodes have the same link-state database
  - All nodes forward packets on shortest paths
  - The next router on the path forwards to the next hop
Detection delay
- A node does not detect a failed link immediately
- … and forwards data packets into a black hole
- Depends on timeout for detecting lost LSPs
Transient Disruptions

- Inconsistent link-state database
  - Some routers know about failure before others
  - The shortest paths are no longer consistent
  - Can cause transient forwarding loops
Sources of convergence delay
- Detection latency
- Flooding of link-state information
- Shortest-path computation
- Creating the forwarding table

Poor performance during convergence period
- Lost packets due to black holes and TTL expiry
- Looping packets consuming resources
- Out-of-order packets reaching the destination

Very bad for VoIP, online gaming, and video
Reducing Delay

- Faster detection
  - Smaller LSP timers
  - Link-layer technologies that can detect failures

- Faster flooding
  - Flooding immediately
  - Sending link-state packets with high-priority

- Faster computation
  - Faster processors on the routers
  - Incremental Dijkstra’s algorithm

- Faster forwarding-table update
  - Data structures supporting incremental updates
Real Link-state Protocols

- OSPF (Open Shortest Path First) and IS-IS
  - Most widely used intra-domain routing protocols
  - Run by almost all ISPs and many large organizations

- Basic link state algorithm plus many features:
  - Authentication of routing messages
  - Extra hierarchy: Partition into routing areas
    - “Border” router pretends to be directly connected to all routers in an area (answers for them)
  - Load balancing: Multiple equal cost routes
Link State evaluation

- Strengths
  - Loop free as long as LS database’s are consistent
    » Can have transient routing loops – shouldn’t last long
  - Messages are small
  - Converges quickly
  - Guaranteed to converge

- Weaknesses
  - Must flood data across entire network (scalability?)
  - Must maintain state for entire topology (database)
For next time…

• Study for the Midterm
  • **Midterm will by async on Canvas:** 1 hour to complete after you start, and available online for 24 hours
  • Exam covers everything up to Monday’s lecture
  • Homework 1 and 2 solutions posted on Piazza and website

• Read Chapter 3.4 in P&D (Distance vector)