Lecture 14: Link state and Distance Vector Routing

CSE 123: Computer Networks
Aaron Schulman
Lecture 14 Overview

- Link-state routing
- Distance vector
- Bellman-Ford algorithm
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)

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

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

- 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

CSE 123 – Lecture 13: Link-state Routing
Making Something Disappear

- 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
  - 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
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
Transient Disruptions

- 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 hellos

CSE 123 – Lecture 14: Distance-vector Routing
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
Convergence Delay

- Sources of convergence delay
  - Detection latency
  - Flooding of link-state information
  - Shortest-path computation
  - Creating the forwarding table

- 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 hello 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…

- Read Ch. 3.4 in P&D