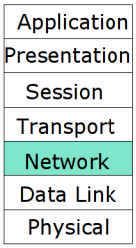
CSE/EE 461: Introduction to Computer Communications Networks
Winter 2009

Module 8
Internet Routing

John Zahorjan
zahorjan@cs.washington.edu
534 Allen Center

This Module

- Distance Vector Routing
- Link State Routing



Kinds of Routing Schemes

- Many routing schemes have been proposed/explored!

- Distributed or centralized
- Hop-by-hop or source-based
- Deterministic or stochastic
- Single or multi-path
- Static or dynamic route selection

- Internet is to the left…

Routing Questions

- How to choose best path?
  - Defining "best" is 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
  - We'll use distributed algorithms

Some Pitfalls

- Using global knowledge is challenging
  - Hard to collect
  - Can be out-of-date
  - Needs to summarize in a locally-relevant way
- Inconsistencies in local /global knowledge can cause:
  - Loops (black holes)
  - Oscillations, especially when adapting to load

First Approach: Distance Vector Routing

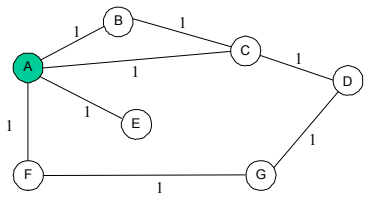
- Assume:
  - Each router knows only address of / cost to send to neighbors
- Goal:
  - Calculate routing table of next hop information for each destination at each router
- Idea:
  - Bellman-Ford
    - Tell neighbors about current distances to all destinations
    - Update cost/next hop to each destination based on your neighbors' costs
  - Very similar to the bridge spanning tree algorithm

DV Algorithm

- Each router maintains a vector of costs to *all* destinations, as well as a routing table
  - Initialize neighbors with known cost, others with infinity
- Periodically send distance vector to neighbors
  - On reception of a vector, if neighbor's path to a destination plus cost to neighbor is better, switch to better path
    - update cost in vector and next hop in routing table
- Assuming no changes, will converge to shortest paths
  - But what happens if there are changes?

Distance Vector Example

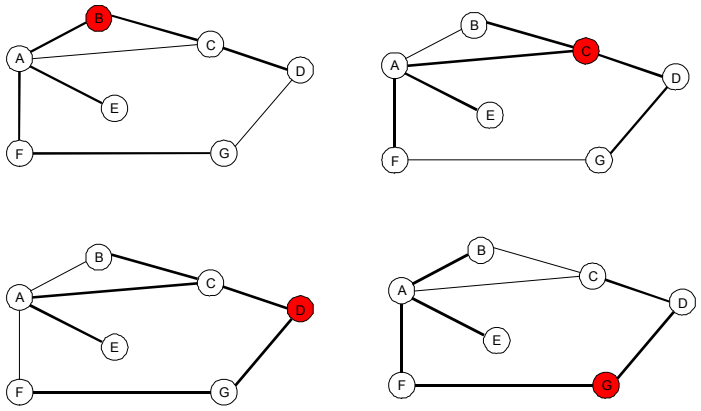
- Using hop count as the metric



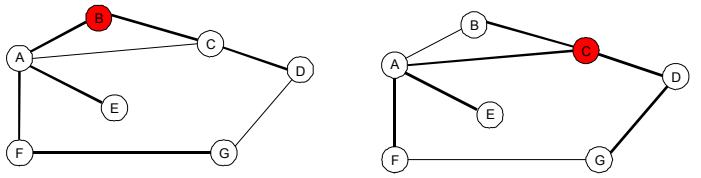
Final Table at A

Dest	Cost	Next
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

A's Routing Table: Edges on Spanning Trees Rooted at Destinations



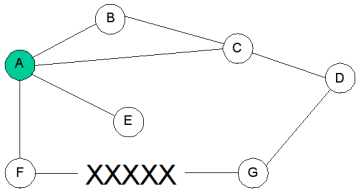
The Trees Are "Consistent"



- If A routes to C to reach D, then C's route to D has the cost A had in mind when choosing C
- No loops
  - If A routes to C to reach some destination D, C cannot think A is closer to D than C is itself

What if there are changes?

- Suppose link between F and G fails
  - F notices failure, sets its cost to G to infinity
  - A (eventually) receives costs to G from B (3), C (2), and F (∞) and updates its routing table and cost to use C
  - F hears cost updated cost from A (3) and adopts A as next hop

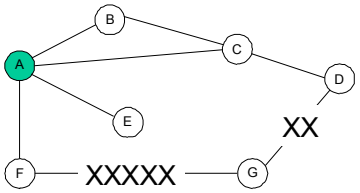


Final Table at A

Dest	Cost	Next
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	3	C

Trouble Looms

- Now link between D and G fails
  1. D notices failure, sets its cost to G to infinity
  2. D hears from C that its cost to G is 2, updates to use C
  3. C hears cost from A (3), B (3), and D (3), chooses A
  4. A updates to B
  5. B updates to C
  6. …

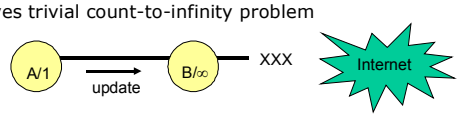


"Count to infinity" problem

Why does this happen?

Mitigation

- Split Horizon
  - Router never advertises the cost of a destination back to its next hop – that's where it learned it from!
  - Solves trivial count-to-infinity problem



- Poison reverse
  - go even further – advertise infinity back to your next hop
- Hold down
  - If you set cost to infinity, don't change it until some timer expires

Mitigation (cont.)

- However, distance vector protocols still subject to the same problem with more complicated topologies
  - Many enhancements suggested
- Make infinity small
  - Reduces time to convergence (to infinity)

RIP: Routing Information Protocol

- DV protocol with hop count as metric
  - Infinity = 16 hops
    - limits size network size
  - Includes split horizon with poison reverse
- Routers send vectors every 30 seconds
  - With triggered updates for link failures
  - Time-out in 180 seconds to detect failures
- RIPv1 specified in RFC1058
  - www.ietf.org/rfc/rfc1058.txt
- RIPv2 (adds authentication etc.) in RFC1388
  - www.ietf.org/rfc/rfc1388.txt

RIP is an "Interior Gateway Protocol"

- Suitable for small- to medium-sized networks
  - such as within a campus, business, or ISP
- Unsuitable for Internet-scale routing
  - hop count metric poor for heterogeneous links
  - 16-hop limit places max diameter on network

Later, we'll talk about "Exterior Gateway Protocols"

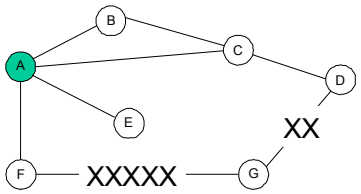
- used between organizations to route across Internet

Second Approach: Link State Routing

- Same assumptions/goals, but different idea than DV:
  - Each router acquires information on the full network topology and computes a minimum cost spanning tree with itself as root
    - Why does this work? (How do we know there will be no loops?)
- Two components to implementation:
  1. Topology dissemination
     - Flooding
  2. Shortest-path calculation
     - Dijkstra's algorithm

Link State: Dijkstra's Algorithm

- Why Dijkstra?
  - Why not?
    - It's fast
    - Link weights are non-negative
- What about behavior under failure?



Final Table at A

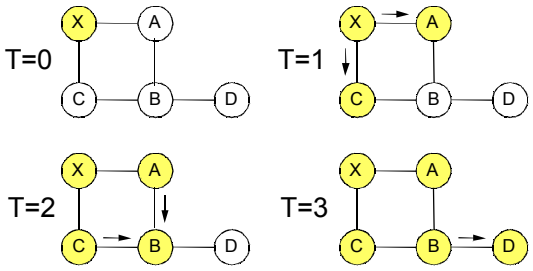
Dest	Cost	Next
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F

Distributing Link State Data: Flooding

- Each router must communicate the state of its outbound links to *all* other routers
  - Each router periodically sends link state packets (LSPs)
    - LSPs contain [router, neighbors, costs]
- Require:
  - New news to travel fast
    - Why?
  - Old news to eventually be forgotten
    - Why?
- Technique: flooding
  - Each router forwards LSPs not already in its database on all ports except where received
    - Each LSP will travel over the same link at most once in each direction
- Flooding is fast, and can be made reliable with ACKs

Example

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

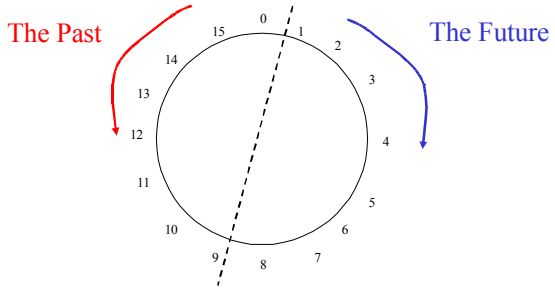


Reliability

- Want LSP to arrive everywhere soon
  - ⇒ ARQ
    - ⇒ sequence numbers
- What if a router goes down?
  - Its neighbors start advertising cost ∞ to reach it
  - Sequence number check on LSP causes other routers to update their views of the network topology
    - Perfect
- A real-world "glitch"…

ARPANET Failure

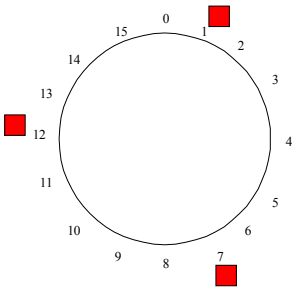
- Review: When is one sequence number bigger than another?



- 6-bit sequence numbers
  ⇒ 32 sequence numbers to go in the future
  ⇒ 16 minutes before an old packet "becomes new"
  ⇒ no problem

ARPANET Failure

- A router went berserk
- Turning off that router doesn't help
  - LSPs circulate forever, updating each other
- Eventually had to inject special code into all other routers to eliminate the bad LSPs



Reaction (OSPF)

- Sequence number field is 32-bits
  - Intended never to wrap
    - 1,361 years to exhaust at 10 seconds/sequence number
- TTL field on LSPs
  - Counts up, one per hop
  - Counts up periodically while in a router's database
  - Thrown away when exceeds some maximum

Open Shortest Path First (OSPF)

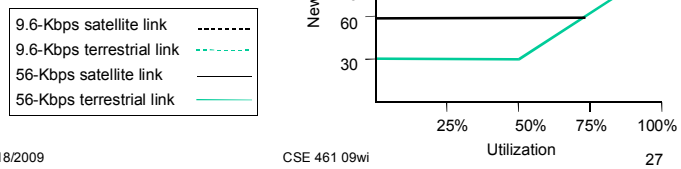
- Most widely-used Link State protocol today
- Basic link state algorithms plus many features:
  - Authentication of routing messages
  - Extra hierarchy: partition into routing areas
    - Only bordering routers send link state information to another area
      - Reduces chatter.
      - Border router "summarizes" network costs within an area by making it appear as though it is directly connected to all interior routers
    - Load balancing

Cost Metrics

- How should we choose cost?
  - To get high bandwidth, low delay or low loss?
  - Do they depend on the load?
- Static Metrics
  - Hopcount is easy but treats OC3 (155 Mbps) and T1 (1.5 Mbps)
  - Can tweak result with manually assigned costs
- Dynamic Metrics
  - Depend on load; try to avoid hotspots (congestion)
  - But can lead to oscillations (damping needed)

Revised ARPANET Cost Metric

- Based on load and link
- Variation limited (3:1) and change damped
- Capacity dominates at low load; we only try to move traffic if high load



Key Concepts

- Routing uses global knowledge; forwarding is local
- Many different algorithms address the routing problem
  - We have looked at two classes: DV (RIP) and LS (OSPF)
- Challenges:
  - Handling failures/changes
  - Defining "best" paths
  - Scaling to millions of users