Computer Networks

Lecture 22

Distance Vector and Link State Routing Protocols;
Intro to Internet Routing

11/19/2013

Admin

- Assignment 4 questions
Recap: Basic Routing Setting: Shortest Path

- Represent network as a graph, with positive costs assigned to network links
- Each node computes the shortest (lowest cost) paths to all destinations

Recap: Distributed Distance-Vector

- Distributed shortest path alg. based on simple update rule:

\[ d_i(h + 1) = \min_{j \in N(i)} (d_{ij} + d_j(h)) \]

- Key property: monotonicity
  - Consider two configurations \( d(t) \) and \( d'(t) \):
    - if \( d(t) \geq d'(t) \), then \( d(t+1) \geq d'(t+1) \)
  - Tool box: a key technique for proving convergence (liveness) of distributed protocols: two extreme configurations to sandwich any real configurations
Recap: Distributed Distance-Vector

- Good news propagate quickly
- Bad news propagate slowly (counting to infinity)

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4  Initially</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
<td>4  After 1 exchange</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4  After 2 exchanges</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4  After 3 exchanges</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>5</td>
<td>6  After 4 exchanges</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>7</td>
<td>6  After 5 exchanges</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>7</td>
<td>8  After 6 exchanges</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

Recap: Routing Loops

- Counting-to-infinity is due to routing loops:
  - A global state (consisting of the nodes' local states) at a global moment (observed by an oracle) such that there exist nodes A, B, C, ... E such that A (locally) thinks B as next hop, B thinks C as next hop, ... E thinks A as next hop

- Solution 1: Reverse-poison
  - Removes two-node loops but may not remove more-node loops
Discussion

- Possibilities to avoid routing loops?

Outline

- Recap
  - Distance vector protocols
    - synchronous Bellman-Ford (SBF)
    - asynchronous Bellman-Ford (ABF)
  - destination-sequenced distance vector (DSDV)
Destination-Sequenced Distance Vector protocol (DSDV)

- One extension of distance vector protocol to address the counting-to-infinity problem

Extension
- DSDV tags each route with a sequence number
- Each destination node D periodically advertises monotonically increasing even-numbered sequence numbers
- When a node realizes that the link it uses to reach destination D is broken, it advertises an infinite metric and a sequence number which is one greater than the previous route (i.e., an odd seq. number)
  - The route is repaired by a later even-number advertisement from the destination

DSDV: More Detail

- Let’s assume the destination node is D

- There are optimizations but we present a simple version:
  - Each node B maintains \((S^B, d^B)\), where \(S^B\) is the sequence number at B for destination D and \(d^B\) is the best distance using a neighbor from B to D

- Both periodical and triggered updates
  - Periodically: D increases its seq. by 2 and broadcasts with \((S^D, 0)\)
  - If B is using C as next hop to D and B discovers that C is no longer reachable
    - B increases its sequence number \(S^B\) by 1, sets \(d^B\) to \(\infty\), and sends \((S^B, d^B)\) to all neighbors
Example

Will this trigger an update?

Example

Will this trigger an update?
**DSDV: Simple Update**

- Consider simple version, no optimization
- Update after receiving a message
  - assume B sends to A its current state \((S^B, d^B)\)
  - when A receives \((S^B, d^B)\)
    - if \(S^B > S^A\), then
      - **always update if a higher seq#**
      - \(S^A = S^B\)
      - if \((d^B = \infty)\) \(d^A = \infty\); else \(d^A = d^B + d(A,B)\)
    - else if \(S^A = S^B\), then
      - if \(d^A > d^B + d(A,B)\)
        - **update for the same seq# only if better route**
        - \(d^A = d^B + d(A,B)\) and uses B as next hop

---

**Example: CD link down**

- C-D link is down, it increases its seq# and broadcasts its cost to be \(\infty\)
Claim: DSDV Does Not Form Loop

- Initially no loop (no one has next hop so no loop)
- Derive contradiction if a loop forms after a node processes an update,
  - e.g., when A receives the update from B, A decides to use B as next hop and forms a loop

Technique: Global Invariants

- Global Invariant is a very effective method in understanding safety of distributed asynchronous protocols
- Invariants are defined over the states of the distributed nodes

- Consider any node B.
- Let’s identify some invariants over the state of node B, i.e., \((S^B, d^B)\).
**Invariants of a Single Node B**

- Some invariants about the state of a node B
  - [I1] $S^B$ is non-decreasing
  - [I2] $d^B$ is non-increasing for the same sequence number

**Invariants of if A Considers B as Next Hop**

- Some invariants if A considers B as next hop
  - [I3] $S^A$ cannot be an odd number, $d^A$ is not $\infty$
  - [I4] $S^B \geq S^A$
    because A is having the seq# which B last sent to A; B's seq# might be increased after B sent its state
  - [I5] if $S^B = S^A$ then $d^B \times d^A$
    because $d^A$ is based on $d^B$ which B sent to A some time ago, $d^B \times d^A$ since all link costs are positive; $d^B$ might be decreased after B sent its state
Loop Freedom of DSDV

- Consider a critical moment
  - A starts to consider B as next hop, and we have a loop
- According to invariant I4 for each link in the loop
  \((X \text{ considers } Y \text{ as next hop}) : S^Y \geq S^X\)
- Two cases:
  - exists \(S^Y > S^X\)
    - by transition along the loop \(S^B > S^A\)
  - all nodes along the loop have the same sequence number
    - apply I5, by transition along the loop \(d^B > d^A\)

Summary: DSDV

- DSDV uses sequence number to avoid routing loops
  - seq# partitions routing updates from different outside events
  - within same event, no loop so long each node only decreases its distance
EIGRP: Diffusive Update Alg.

- EIGRP: a proprietary routing protocol by Cisco based on a Diffusive Update Algorithm (DUAL)
- Both a recomputation part and a fast reroute (switching) part
- We focus on the fast switching part
  - key idea: checking condition before switching to another route to prevent potential loops

EIGRP Key Idea: Feasible Successors

- If the reported distance of a neighbor n is lower than the total distance using primary (current shortest), the neighbor n is a feasible successor

\[ d_n + d_{i \rightarrow n} \geq d_{\text{primary}} + d_{i \rightarrow \text{primary}} > d_n \]
EIGRP: Example

Will A use B as a feasible successor?

Using Feasible Successor

- Assume that the DV alg. converges.
- The link to a node’s primary neighbor is down, and the node switches to a feasible successor.
- No loop will be formed

Discussion: Distance Vector Routing

- What do you like about distance vector routing?

- What do you not like about distance vector routing?
Churns of DV: One Example

Message sequences
1. Node 2 tells 3. Node 3 tells 4...
   Node N tells N+1. (N-1 messages)
2. Node N+1 tells N+2, N+2 tells N+3, ...
   2N. (N-1 messages)
3. Now node N-1 tells node N+1
4. Step 2 repeats
5. Now node N-2 tells node N+1
6. ...

A total of $O(N^2)$ messages

Outline

- Recap
- Distance-vector protocols
  - synchronous Bellman-Ford (SBF)
  - asynchronous Bellman-Ford (ABF)
  - destination-sequencezed distance vector (DSDV)
  - DUAL
- Link state protocol
Link-State Routing

- Net topology, link costs are distributed to all nodes
  - all nodes have same info
- Link state distribution accomplished via “link state broadcast”
- Each node computes its shortest paths from itself to all other nodes
  - e.g., use Dijkstra’s algorithm, not distributed shortest path computation
- The only distributed part is link state broadcast

Link State Broadcast

- This is the hard part

- Basic event structure at node n
  - on initialization:
  - on state change to a link connected to n:
  - on receiving an LS:
    - forwards a link state broadcast (link ID, link head, link tail, link up/down) to all links except the incoming link
Link State Broadcast

Node S updates link states connected to it.

To avoid forwarding the same link state announcement (LSA) multiple times (forming a loop), each node remembers the received LSAs.
- Second LSA[e] received by E from C is discarded
- Second LSA[e] received by C from E is discarded as well
- Node H receives LSA[e] from two neighbors, and will discard one of them
Link State Broadcast

Basic event structure at node n

- on initialization:
  - broadcast LSA[e] for each link e connected to n

- on state change to a link e connected to n:
  - broadcast LSA[e]

- on receiving an LSA[e]:
  - if (does not have LSA[e])
    forwards LSA[e] to all links except the incoming link

Does this protocol allow each node to have a complete link state map?

Link State Broadcast: Issues

- Problem: Out of order delivery
  - link down and then up
  - A node may receive up first and then down

- Solution
  - Each link update is given a sequence number: (initiator, seq#, link, status)
    - the initiator should increase the seq# for each new update
  - If the seq# of an update of a link is not higher than the highest seq# a router has seen, drop the update
  - Otherwise, forward it to all links except the incoming link (real implementation using packet buffer)

- Problem of solution: seq# corruption
- Solution: age field
Link State Broadcast: Issues

- Problem: network partition and then reconnect, how to sync across the reconnected components

- Solution: updates are sent periodically

OSPF (Open Shortest Path First)

- "Open": publicly available

- Uses Link State algorithm
  - link state (LS) packet dissemination
  - topology map at each node
  - route computation using Dijkstra's algorithm

http://en.wikipedia.org/wiki/Open_Shortest_Path_First
**OSPF “Advanced” Features (not in RIP)**

- Multiple same-cost paths allowed (only one path in RIP)
- For each link, multiple cost metrics for different Type Of Service (e.g., satellite link cost set “low” for best effort; high for real time)
- Security: all OSPF messages authenticated (to prevent malicious intrusion); TCP connections used
- Hierarchical OSPF

---

**Hierarchical OSPF**

- “summarize” distances to nets in own area, advertise to other Area Border routers.
- Link-state advertisements only in area each node has detailed area topology.
- Only know direction (shortest path) to nets in other areas.

Two-level hierarchy: local area, backbone.
Why Hierarchy?

- Information hiding (filtered) $\Rightarrow$ reduce computation, bandwidth, storage

Discussion: Link State Routing

- What do you like about link state routing?

- What do you not like about link state routing?
Summary

- Basic routing protocols
  - Distance vector protocols
  - Link state protocols

- Remaining question: how to design the routing infrastructure for the global Internet?

Outline

- Recap
- Distance-vector protocols
- Link state protocol
  - Routing in the Internet
Routing in the Internet

- The Global Internet consists of Autonomous Systems (AS) interconnected with each other
  - An AS is identified by an AS Number (ASN), e.g. Yale ASN is 29
  - Each AS owns part of IP address space
  - Autonomous systems connect to each other

- Try http://www.fixedorbit.com/search.htm

Internet ISP Connectivity
Routing with AS

- Internet routing is divided into intra-AS routing and inter-AS routing

- **Intra-AS**
  - A protocol running inside an AS is called an Interior Gateway Protocol (IGP)
    - RIP: Routing Information Protocol
    - E/IGRP: Interior Gateway Routing Protocol (Cisco)
    - OSPF: Open Shortest Path First
    - IS-IS: very similar to OSPF (or should we say OSPF is very similar to IS-IS?)

- **Inter-AS**
  - A protocol runs among AS's is also called an Exterior Gateway Protocol (EGP)
  - for global connectivity, a single interdomain routing protocol
**Intra-AS and Inter-AS: Example**

- **border (exterior gateway) routers**
- **interior (gateway) routers**

---

**Routing in the Internet: Example**

- **Gateway routers of same AS** share learned external routes using **iBGP**.
- **Gateway routers of diff. AS's** exchange routes using **eBGP**.
Many Routing Processes on a Single Router

BGP Setup
Internet Interdomain Routing: BGP

- **BGP (Border Gateway Protocol):** the de facto Inter-AS standard
- **Path Vector** protocol:
  - similar to Distance Vector protocol
  - a border gateway sends to a neighbor *entire path* (i.e., a sequence of ASNs) to a destination, e.g.,
    - gateway X sends to neighbor N its path to dest. Z:
      \[
      \text{path}(X,Z) = X, Y_1, Y_2, Y_3, \ldots, Z
      \]
  - if N selects path(X, Z) advertised by X, then:
    \[
    \text{path}(N,Z) = N, \text{path}(X,Z)
    \]

BGP Operations (Simplified)

1. **Establish session on TCP port 179**
2. **Exchange all active routes**
3. **Exchange incremental updates**

while (connection is ALIVE)

exchange UPDATE message
select best available route
if route changes, export to neigh.
BGP Messages

- Four types of messages
  - OPEN: opens TCP connection to peer and authenticates sender
  - UPDATE: advertises new path (or withdraws old)
  - KEEPALIVE keeps connection alive in absence of UPDATES; also ACKs OPEN request
  - NOTIFICATION: reports errors in previous msg; also used to close connection

Benefits of Including Path Vector

- Path vector prevents counting-to-infinity problem
- Path vector allows an AS to define local policies on the selection of AS paths for each destination
See BGP Paths

- http://www.fixedorbit.com/trace.htm