Routing Algorithms

Link State Routing
- Routers periodically broadcast link state information
  - link state messages or LSA (link state advertisement)
  - broadcast via sequence-number controlled flooding
- All routers acquire complete connectivity + link cost information of entire network
- Each router computes routing table from that obtained topology
  - Dijkstra’s shortest path to other routers, from computing router → this becomes forwarding table

Distance Vector Algorithm (1)

Bellman-Ford Equation (dynamic programming)
Define
\[ d_x(y) := \text{cost of least-cost path from } x \text{ to } y \]

Then
\[ d_x(y) = \min (c(x,v) + d_x(y)) \]

where min is taken over all neighbors of x

Routing Algorithm Classification

Global or decentralized information?
Global:
- all routers have complete topology, link cost info
- "link state" algorithms

Decentralized:
- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Static or dynamic?
Static:
- routes change slowly over time

Dynamic:
- routes change more quickly
  - periodic update
  - in response to link cost changes

Note: Global/decentralized classification does not say where routing computation is performed, it says what information one has when computing routes

Announcements

- Project 2A due Apr 4
- Project 2B due in 2 parts:
Bellman-Ford Example (2)

Clearly, \( d_v(z) = 5 \), \( d_u(z) = 3 \), \( d_w(z) = 3 \).

B-F equation says:

\[
d_z(x) = \min \{ c(u,v) + d_z(v), \\
c(u,x) + d_z(z), \\
c(u,w) + d_z(w) \} = \min \{ 2 + 5, \\
1 + 3, \\
5 + 3 \} = 4
\]

Node that achieves minimum is next hop in shortest path → forwarding table.

Distance Vector Algorithm (3)

- \( D_x(y) \) = estimate of least cost from \( x \) to \( y \)
- Distance vector: \( D_x = [D_x(y): y \in N] \)
- Node \( x \) knows cost to each neighbor \( v \): \( c(x,v) \)
- Node \( x \) maintains \( D_x = [D_x(y): y \in N] \)
- Node \( x \) also maintains its neighbors’ distance vectors (at least temporarily)
  - For each neighbor \( v \), \( x \) maintains \( D_v = [D_v(y): y \in N] \)

Distance Vector Algorithm (4)

**Basic Idea:**
- Each node periodically sends its own distance vector estimate to neighbors.
- When a node \( x \) receives new DV estimate from neighbor, it updates its own DV using B-F equation:

\[
D_x(y) \leftarrow \min_v \{ c(x,v) + D_v(y) \} \quad \text{for each node } y \in N
\]

- Under minor, natural conditions, the estimate \( D_x(y) \) converges to actual least cost \( d_x(y) \)

Distance Vector Algorithm (5)

**Iterative, asynchronous:** each local iteration caused by:
- Local link cost change
- DV update message from neighbor

**Distributed:**
- Each node notifies neighbors only when its DV changes
  - Neighbors then notify their neighbors if necessary

Example DV Routing

Example (cont’d)
Example (cont’d)

- Converged after 3 iterations

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

- What if link cost changes?

Distance Vector: Link Cost Changes

Link cost changes:
- node detects local link cost change
- updates routing info, recalculates distance vector
- if DV changes, notify neighbors

At time $t_0$, $y$ detects the link-cost change, updates its DV, and informs its neighbors.

At time $t_1$, $z$ receives the update from $y$ and updates its table. It computes a new least cost to $x$ and sends its neighbors its DV.

At time $t_2$, $y$ receives $z$'s update and updates its distance table. $y$'s least costs do not change and hence $y$ does not send any message to $z$.

"good news travels fast"

At time $t_0$, $y$ detects the link-cost change, updates its DV, and informs its neighbors.

At time $t_1$, $z$ receives the update from $y$ and updates its table. It computes a new least cost to $x$ and sends its neighbors its DV.

At time $t_2$, $y$ receives $z$’s update and updates its distance table. $y$’s least costs do not change and hence $y$ does not send any message to $z$.

Count-To-Infinity

Before change:

After change:

Poisoned Reverse

Comparison of LS and DV algorithms

Message complexity
- **LS**: with $n$ nodes, $E$ links, $O(nE)$ msgs sent
- **DV**: exchange between neighbors only
  - convergence time varies

Speed of Convergence
- **LS**: $O(n^2)$ algorithm requires $O(n^2)$ msgs
  - may have oscillations
- **DV**: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?
- **LS**: node can advertise incorrect link cost
  - each node computes only its own table
- **DV**: DV node can advertise incorrect path cost
  - each node’s table used by others
  - error propagates thru network

Distance Vector: Link Cost Changes

Poisoned reverse:
- If $Z$ routes through $Y$ to get to $X$:
  - $Z$ tells $Y$ its (Z’s) distance to $X$ is infinite (so $Y$ won’t route to $X$ via $Z$)
- will this completely solve count to infinity problem?
Hierarchical Routing

Our routing study thus far - idealization
• all routers identical
• network “flat”
… not true in practice

scale: with 200 million destinations:
• can’t store all dest’s in routing tables!
• routing table exchange would swamp links!

administrative autonomy
• internet = network of networks
• each network admin may want to control routing in its own network

Hierarchical Routing
• aggregate routers into regions, “autonomous systems” (AS)
• routers in same AS run same routing protocol
  – “intra-AS” routing protocol
  – routers in different AS may run different intra-AS routing protocols

Gateway router
• Direct link to router in another AS
  – Example: AS1312

Interconnected ASes

• Forwarding table is configured by both intra- and inter-AS routing algorithm
  – Intra-AS sets entries for internal dests
  – Inter-AS & Intra-AS sets entries for external dests

Inter-AS tasks
• Suppose router in AS1 receives datagram for which dest is outside of AS1
  – Router should forward packet towards one of the gateway routers, but which one?

  Scenario 1: Destination x is reachable through single AS
  Inter-AS: AS3 advertises route to x
  Least cost path to 1c is interface l, add (x, l) to routing table

  Scenario 2: Destination is reachable through multiple AS:
  Hot potato routing: send packet towards closest of two routers

  Scenario 2: Destination is reachable through multiple AS:
  Hot potato routing: send packet towards closest of two routers

Inter-AS tasks (cont’d)
• Suppose router in AS1 receives datagram for which dest is outside of AS1
  – Router should forward packet towards one of the gateway routers, but which one?

Hot Potato Routing
Hot Potato Routing

• Suppose router in AS1 receives datagram for which dest is outside of AS1
  • Router should forward packet towards one of the gateway routers, but which one?

Scenario 2:
Destination is reachable through multiple AS:

AS3 advertises route to x
AS3 advertises route to x

1b is closer than 1c, interface k leads to 1b, so add (x, k) to routing table

Spread in AS1: 1c and 1b lead to x

Summary

• Routing algorithms:
  – Link State
  – Distance Vector
• Hierarchical Routing
  – AS: autonomous systems

• Next: application to Internet/IP