Session 7 - Main Theme
Networks: Part II
Routing Algorithms and Routing Protocol

Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Agenda

- Routing Strategies
- Interplay Between Routing and Forwarding
- Graph Abstraction
- Routing Algorithm Classification
- Link-State Routing (LS) Algorithm
- Distance Vector (DV) Algorithm
- Comparison of LS and DV Algorithms
- Hierarchical Routing
- Interconnected ASes
- Inter-AS Tasks
- Intra-AS Routing
- Internet Inter-AS Routing – Border Gateway Protocol
- Why Different Intra- and Inter-AS Routing?
Part I

Routing Strategies

- Fixed
- Flooding
- Random
- Adaptive
Fixed Routing

- Single permanent route for each source to destination pair
- Determine routes using a least cost algorithm
- Route fixed, at least until a change in network topology

Fixed Routing Tables

<table>
<thead>
<tr>
<th>From Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>5</td>
<td>2</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td></td>
<td>4</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>To Node</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td></td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

Central Routing Directory

<table>
<thead>
<tr>
<th>Node 1 Directory</th>
<th>Destination</th>
<th>Next Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Node 2 Directory</td>
<td>Destination</td>
<td>Next Node</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Node 3 Directory</td>
<td>Destination</td>
<td>Next Node</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>
Flooding

- No network info required
- Packet sent by node to every neighbor
- Incoming packets retransmitted on every link except incoming link
- Eventually a number of copies will arrive at destination
- Each packet is uniquely numbered so duplicates can be discarded
- Nodes can remember packets already forwarded to keep network load in bounds
- Can include a hop count in packets

Flooding Examples
Properties of Flooding

- All possible routes are tried
  - Very robust
- At least one packet will have taken minimum hop count route
  - Can be used to set up virtual circuit
- All nodes are visited
  - Useful to distribute information (e.g. routing)

Random Routing

- Node selects one outgoing path for retransmission of incoming packet
- Selection can be random or round robin
- Can select outgoing path based on probability calculation
- No network info needed
- Route is typically not least cost nor minimum hop
Adaptive Routing

- Used by almost all packet switching networks
- Routing decisions change as conditions on the network change
  - Failure
  - Congestion
- Requires info about network
- Decisions more complex
- Tradeoff between quality of network info and overhead
- Reacting too quickly can cause oscillation
- Reacting too slowly to be relevant

Part II

*Interplay Between Routing and Forwarding*
Interplay Between Routing & Forwarding

<table>
<thead>
<tr>
<th>header value</th>
<th>output link</th>
</tr>
</thead>
<tbody>
<tr>
<td>0100</td>
<td>3</td>
</tr>
<tr>
<td>0101</td>
<td>2</td>
</tr>
<tr>
<td>0111</td>
<td>2</td>
</tr>
<tr>
<td>1001</td>
<td>1</td>
</tr>
</tbody>
</table>

value in arriving packet’s header

Part III

Graph Abstraction
Graph Abstraction

Graph: \( G = (N,E) \)

- \( N \) = set of routers = \{ u, v, w, x, y, z \}
- \( E \) = set of links = \{ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) \}

Remark: Graph abstraction is useful in other network contexts
Example: P2P, where \( N \) is set of peers and \( E \) is set of TCP connections

Graph Abstraction: Costs

- \( c(x,x') = \text{cost of link } (x,x') \)
  - e.g., \( c(w,z) = 5 \)
  - cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

Cost of path \( (x_1, x_2, x_3, ..., x_p) = c(x_1, x_2) + c(x_2, x_3) + ... + c(x_{p-1}, x_p) \)

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path
Part IV

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
Part V

*Link-State Routing Algorithm*

---

## Link-State Routing Algorithm

### Dijkstra’s algorithm
- Net topology, link costs known to all nodes
  - Accomplished via "link state broadcast"
  - All nodes have same info
- Computes least cost paths from one node ("source") to all other nodes
  - Gives forwarding table for that node
- Iterative: after k iterations, know least cost path to k dest.'s

### Notation:
- $c(x,y)$: link cost from node $x$ to $y$; $\infty$ if not direct neighbors
- $D(v)$: current value of cost of path from source to dest. $V$
- $p(v)$: predecessor node along path from source to $v$
- $N'$: set of nodes whose least cost path definitively known
Dijskstra’s Algorithm

1 \textit{Initialization:}
2 \hspace{1em} N' = \{u\}
3 \hspace{1em} \text{for all nodes } v
4 \hspace{1em} \text{if } v \text{ adjacent to } u
5 \hspace{1em} \text{then } D(v) = c(u,v)
6 \hspace{1em} \text{else } D(v) = \infty
7
8 \textit{Loop}
9 \hspace{1em} \text{find } w \text{ not in } N' \text{ such that } D(w) \text{ is a minimum}
10 \hspace{1em} \text{add } w \text{ to } N'
11 \hspace{1em} \text{update } D(v) \text{ for all } v \text{ adjacent to } w \text{ and not in } N':
12 \hspace{1em} D(v) = \min( D(v), D(w) + c(w,v) )
13 \hspace{1em}/* \text{ new cost to } v \text{ is either old cost to } v \text{ or known}
14 \hspace{1em} \text{shortest path cost to } w \text{ plus cost from } w \text{ to } v */
15 \hspace{1em} \text{until all nodes in } N'\n
Dijkstra’s Algorithm: Example

<table>
<thead>
<tr>
<th>Step</th>
<th>N'</th>
<th>D(v),p(v)</th>
<th>D(w),p(w)</th>
<th>D(x),p(x)</th>
<th>D(y),p(y)</th>
<th>D(z),p(z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>u</td>
<td>2,u</td>
<td>5,u</td>
<td>-1,u</td>
<td>\infty</td>
<td>\infty</td>
</tr>
<tr>
<td>1</td>
<td>ux</td>
<td>2,u</td>
<td>4,x</td>
<td>2,x</td>
<td>\infty</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>uxy</td>
<td>2,u</td>
<td>3,y</td>
<td></td>
<td>4,y</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>uxyv</td>
<td></td>
<td>3,y</td>
<td></td>
<td>4,y</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>uxyvw</td>
<td></td>
<td></td>
<td></td>
<td>4,y</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>uxyvwz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\begin{center}
\includegraphics[width=0.5\textwidth]{diagram.png}
\end{center}
Dijkstra’s Algorithm: Discussion

Algorithm Complexity: n nodes
- each iteration: need to check all nodes, w, not in N
- \( n(n+1)/2 \) comparisons: \( O(n^2) \)
- more efficient implementations possible: \( O(n \log n) \)

Oscillations Possible:
- e.g., link cost = amount of carried traffic

Part VI

Distance Vector Algorithm
Distance Vector Algorithm

Bellman-Ford Equation (dynamic programming)

Define
d_x(y) := cost of least-cost path from x to y

Then

d_x(y) = min \{c(x,v) + d_v(y) \}

where min is taken over all neighbors of x

Bellman-Ford Example

Clearly, d_u(z) = 5, d_x(z) = 3, d_w(z) = 3

B-F equation says:

d_u(z) = min \{ c(u,v) + d_v(z), c(u,x) + d_x(z), c(u,w) + d_w(z) \}

= min \{2 + 5, 1 + 3, 5 + 3\} = 4

Node that achieves minimum is next hop in shortest path ➔ forwarding table
Distance Vector Algorithm (cont.)

- $D_x(y)$ = 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
  - For each neighbor $v$, $x$ maintains $D_v = [D_v(y): y \in N]$

Basic Idea:

- Each node periodically sends its own distance vector estimate to neighbors
- When node 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 $D_x(y)$ converge the actual least cost $d_x(y)$
Distance Vector Algorithm (cont.)

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

Each node:

wait for (change in local link cost of msg from neighbor)
recompute estimates
if DV to any dest has changed, notify neighbors

\[
D_x(y) = \min\{c(x, y) + D_y(y), c(x, z) + D_z(y)\} \\
= \min\{2 + 0, 7 + 1\} = 2
\]

\[
D_x(z) = \min\{c(x, y) + D_y(z), c(x, z) + D_z(z)\} \\
= \min\{2 + 1, 7 + 0\} = 3
\]
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

"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 $z$.

Distance Vector: Link Cost Changes

- Link cost changes:
  - good news travels fast
  - bad news travels slow - "count to infinity" problem!
  - 44 iterations before algorithm stabilizes: see text

- Poissoned 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?
Part VII

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 \( O(nE) \) 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 propagate thru network
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 (cont.)

- aggregate routers into regions, “autonomous systems” (AS)
- routers in same AS run same routing protocol
  - “intra-AS” routing protocol
  - routers in different AS can run different intra-AS routing protocol

Gateway router
- Direct link to router in another AS

Part IX

*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

Part X

*Inter-AS Tasks*
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?

AS1 needs:
1. to learn which dests are reachable through AS2 and which through AS3
2. to propagate this reachability info to all routers in AS1

Job of inter-AS routing!

Example: Setting Forwarding Table in Router 1d

- Suppose AS1 learns from the inter-AS protocol that subnet x is reachable from AS3 (gateway 1c) but not from AS2.
- Inter-AS protocol propagates reachability info to all internal routers.
- Router 1d determines from intra-AS routing info that its interface I is on the least cost path to 1c.
- Puts in forwarding table entry $(x, I)$. 
Example: Choosing Among Multiple ASes

- Now suppose AS1 learns from the inter-AS protocol that subnet \( x \) is reachable from AS3 and from AS2.
- To configure forwarding table, router 1d must determine towards which gateway it should forward packets for dest \( x \).
- This is also the job on inter-AS routing protocol!
- Hot potato routing: send packet towards closest of two routers.

Part XI

*Intra-AS Routing*
Intra-AS Routing

- Also known as Interior Gateway Protocols (IGP)
- Most common Intra-AS routing protocols:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

- Distance vector algorithm
- Included in BSD-UNIX Distribution in 1982
- Distance metric: # of hops (max = 15 hops)
RIP Advertisement

- Distance vectors: exchanged among neighbors every 30 sec via Response Message (also called advertisement)
- Each advertisement: list of up to 25 destination nets within AS

RIP: Example

<table>
<thead>
<tr>
<th>Destination Network</th>
<th>Next Router</th>
<th>Num. of hops to dest.</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>y</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>z</td>
<td>B</td>
<td>7</td>
</tr>
<tr>
<td>x</td>
<td>--</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Routing table in D
RIP: Example

<table>
<thead>
<tr>
<th>Dest</th>
<th>Next</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>x</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>z</td>
<td>c</td>
<td>4</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Advertisement from A to D

Routing table in D

RIP: Link Failure and Recovery

- If no advertisement heard after 180 sec --> neighbor/link declared dead
  - routes via neighbor invalidated
  - new advertisements sent to neighbors
  - neighbors in turn send out new advertisements (if tables changed)
  - link failure info quickly propagates to entire net
  - poison reverse used to prevent ping-pong loops
    (infinite distance = 16 hops)
RIP Table Processing

- RIP routing tables managed by **application-level** process called route-d (daemon)
- advertisements sent in UDP packets, periodically repeated

```
+----------------+            +----------------+
| Transprt (UDP) |            | Transprt (UDP) |
| network (IP)   |            | network (IP)   |
| forwarding     |            | forwarding     |
| table          |            | table          |
| link           |            | link           |
| physical       |            | physical       |
```

OPSF (Open Shortest Path First)

- “open”: publicly available
- Uses Link State algorithm
  - LS packet dissemination
  - Topology map at each node
  - Route computation using Dijkstra’s algorithm

- OSPF advertisement carries one entry per neighbor router
- Advertisements disseminated to entire AS (via flooding)
  - Carried in OSPF messages directly over IP (rather than TCP or UDP)
**OPSF “advanced” features (not in RIP)**

- Security: all OSPF messages authenticated (to prevent malicious intrusion)
- Multiple same-cost paths allowed (only one path in RIP)
- For each link, multiple cost metrics for different TOS (e.g., satellite link cost set “low” for best effort; high for real time)
- Integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- Hierarchical OSPF in large domains.

---

**Hierarchical OSPF**

[Diagram of hierarchical OSPF]
Hierarchical OSPF

- Two-level hierarchy: local area, backbone.
  - Link-state advertisements only in area
  - Each node has detailed area topology; only know direction (shortest path) to nets in other areas.

- **Area border routers**: “summarize” distances to nets in own area, advertise to other Area Border routers.

- **Backbone routers**: run OSPF routing limited to backbone.

- **Boundary routers**: connect to other AS’s.

---

Part XII

*Intra-AS Routing*
Internet Inter-AS Routing: BGP

- BGP (Border Gateway Protocol): *the* de facto standard
- BGP provides each AS a means to:
  - Obtain subnet reachability information from neighboring ASs.
  - Propagate the reachability information to all routers internal to the AS.
  - Determine “good” routes to subnets based on reachability information and policy.
- Allows a subnet to advertise its existence to rest of the Internet: **“I am here”**

BGP Basics

- Pairs of routers (BGP peers) exchange routing info over semi-permanent TCP connections: BGP sessions
- Note that BGP sessions do not correspond to physical links.
- When AS2 advertises a prefix to AS1, AS2 is *promising* it will forward any datagrams destined to that prefix towards the prefix.
  - AS2 can aggregate prefixes in its advertisement
Distributing Reachability Info

- With eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
- 1c can then use iBGP do distribute this new prefix reach info to all routers in AS1
- 1b can then re-advertise the new reach info to AS2 over the 1b-to-2a eBGP session
- When router learns about a new prefix, it creates an entry for the prefix in its forwarding table.

Path Attributes & BGP Routes

- When advertising a prefix, advert includes BGP attributes.
  - prefix + attributes = “route”
- Two important attributes:
  - AS-PATH: contains the ASs through which the advert for the prefix passed: AS 67 AS 17
  - NEXT-HOP: Indicates the specific internal-AS router to next-hop AS. (There may be multiple links from current AS to next-hop-AS.)
- When gateway router receives route advert, uses import policy to accept/decline.
BGP Route Selection

• Router may learn about more than 1 route to some prefix. Router must select route.
• Elimination rules:
  1. Local preference value attribute: policy decision
  2. Shortest AS-PATH
  3. Closest NEXT-HOP router: hot potato routing
  4. Additional criteria

BGP Messages

• BGP messages exchanged using TCP.
• BGP 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
BGP Routing Policy

Figure 4.5-BGPnew: a simple BGP scenario

- A, B, C are provider networks
- X, W, Y are customer (of provider networks)
- X is dual-homed: attached to two networks
  - X does not want to route from B via X to C
  - .. so X will not advertise to B a route to C

BGP Routing Policy (cont.)

Figure 4.5-BGPnew: a simple BGP scenario

- A advertises to B the path AW
- B advertises to X the path BAW
- Should B advertise to C the path BAW?
  - No way! B gets no “revenue” for routing CBAW since neither W nor C are B’s customers
  - B wants to force C to route to w via A
  - B wants to route only to/from its customers!
Part XIII

Why Different Intra- and Inter-AS Routing

Why Different Intra- and Inter-AS Routing?

Policy:
• Inter-AS: admin wants control over how its traffic routed, who routes through its net.
• Intra-AS: single admin, so no policy decisions needed

Scale:
• hierarchical routing saves table size, reduced update traffic

Performance:
• Intra-AS: can focus on performance
• Inter-AS: policy may dominate over performance
Part XIV

Conclusion

Assignment & Readings

- Assignment #3 (due)
  - Assigned at the completion of Session 6
- Readings
  - Chapter 4 (4.5, 4.6, 4.7)
Next Session:
Networks: Part 2 (Routing - Continued)