Routing Algorithms and Routing in the Internet

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), (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

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

Static or dynamic?
- Static:
  - 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.

Interplay between routing and forwarding

A Link-State Routing Algorithm

Dijkstra’s algorithm

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.
- $p(v)$: predecessor node along path from source to $v$
- $N$: set of nodes whose least cost path definitively known
**Dijkstra’s Algorithm**

1. **Initialization:**
   - $N' = \{ u \}$
   - for all nodes $v$
     - if $v$ adjacent to $u$
       - then $D(v) = c(u,v)$
     - else $D(v) = \infty$

2. **Loop**
   - find $w$ not in $N'$ such that $D(w)$ is a minimum
   - add $w$ to $N'$
   - update $D(v)$ for all $v$ adjacent to $w$ and not in $N'$:
     - $D(v) = \min( D(v), D(w) + c(w,v) )$
   - /* new cost to $v$ is either old cost to $v$ or known shortest path cost to $w$ plus cost from $w$ to $v$ */
   - until all nodes in $N'$

**Dijkstra’s algorithm: example**

<table>
<thead>
<tr>
<th>Step</th>
<th>$N' \setminus { u }$</th>
<th>$D(u)$</th>
<th>$D(x)$</th>
<th>$D(y)$</th>
<th>$D(z)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>${ u }$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td>${ u }$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>2</td>
<td>${ u }$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>3</td>
<td>${ u }$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>4</td>
<td>${ u }$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>5</td>
<td>${ u }$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
</tbody>
</table>

**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

**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

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

where min is taken over all neighbors of $x$

**Distance Vector Algorithm (2)**

- $D_x(z) = D_x(z) = 3$, $D_x(z) = 3$
- B-F equation says:
  - $d_x(z) = \min( c(u,v) + d_v(z), c(u,x) + d_x(z), c(u,w) + d_w(z) )$
  - $= \min( 2 + 5, 1 + 3, 5 + 3 ) = 4$

Node that achieves minimum is next hop in shortest path $\Rightarrow$ forwarding table

**Distance Vector Algorithm (3)**

- $D_x(y) = \text{estimate of least cost from } x \text{ to } y$
- 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 $D_x = \{ D_v(y); y \in N \}$
Distance vector algorithm (4)

Basic idea:
- Each node periodically sends its own distance vector estimate to neighbors.
- When a node receives a new DV estimate from neighbor, it updates its own DV using the B-F equation:
  \[ D_y(x) = \min_s \{ c(x,s) + D_y(s) \} \] for each node \( y \in N \).
- Under minor, natural conditions, the estimate \( D_y(x) \) converges to the actual distance \( d_y(x) \).

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

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

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 \( t_x \), node \( x \) detects the link-cost change, updates its DV, and informs its neighbors.
  \- At time \( t_y \), node \( y \) receives the update from node \( x \) and updates its table. It computes a new least cost to \( x \) and sends its neighbors its DV update and updates its distance table.
  \- At time \( t_z \), node \( z \) receives \( x \)'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

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?

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
  \- errors propagate thru network
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

Gateway router:
- Direct link to router in another AS

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!

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!

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 can run different intra-AS routing protocol

Interconnected ASes

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

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.

Learn from inter-AS protocol that subnet x is reachable via multiple gateways
Use routing info from inter-AS protocol to determine least-cost paths to each of the gateways
Hot potato routing: Choose the gateway that has the smallest least cost
Determine from forwarding table the interface I that leads to least-cost gateway. Enter (x,I) in forwarding table

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!
**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 advertisements**
- 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>
</tbody>
</table>

**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

OSPF (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)

OSPF "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
- 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.

Internet inter-AS routing: BGP
- BGP (Border Gateway Protocol): the de facto standard
- BGP provides each AS a means to:
  1. Obtain subnet reachability information from neighboring ASs.
  2. Propagate the reachability information to all routers internal to the AS.
  3. 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.
- With eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
- 1c can then use iBGP to 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.

**Distributing reachability info**

- BGP messages exchanged using TCP.

**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 passed 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**

- 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 (2)

- 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!

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