Internet Protocol: Routing Algorithms

Srinidhi Varadarajan
Routing

Routing protocol

Goal: determine “good” path (sequence of routers) thru network from source to dest.

Graph abstraction for routing algorithms:

- graph nodes are routers
- graph edges are physical links
  - link cost: delay, $ cost, or congestion level

“good” path:
- typically means minimum cost path
- other def’s possible
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
A 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 routing table for that node
- iterative: after k iterations, know least cost path to k dest.’s

Notation:

- \( c(i,j) \): link cost from node i to j. cost infinite 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, that is next v
- \( N \): set of nodes whose least cost path definitively known
Dijsktra’s Algorithm

1. **Initialization:**
   2.  \( N = \{A\} \)
   3.  for all nodes \( v \)
   4.     if \( v \) adjacent to \( A \)
   5.       then \( D(v) = c(A,v) \)
   6.     else \( D(v) = \text{infty} \)

8. **Loop**
   9.  find \( w \) not in \( N \) such that \( D(w) \) is a minimum
   10. add \( w \) to \( N \)
   11. update \( D(v) \) for all \( v \) adjacent to \( w \) and not in \( N \):
   12.  \( D(v) = \min( D(v), D(w) + c(w,v) ) \)
   13.  /* new cost to \( v \) is either old cost to \( v \) or known
   14.  shortest path cost to \( w \) plus cost from \( w \) to \( v \) */

15. **until all nodes in \( N \)**
# Dijkstra’s algorithm: example

<table>
<thead>
<tr>
<th>Step</th>
<th>start N</th>
<th>D(B),p(B)</th>
<th>D(C),p(C)</th>
<th>D(D),p(D)</th>
<th>D(E),p(E)</th>
<th>D(F),p(F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>2,A</td>
<td>5,A</td>
<td>1,A</td>
<td>infinity</td>
<td>infinity</td>
</tr>
<tr>
<td>1</td>
<td>AD</td>
<td>2,A</td>
<td>4,D</td>
<td>2,D</td>
<td>infinity</td>
<td>infinity</td>
</tr>
<tr>
<td>2</td>
<td>ADE</td>
<td>2,A</td>
<td>3,E</td>
<td></td>
<td>4,E</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ADEB</td>
<td></td>
<td>3,E</td>
<td></td>
<td>4,E</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ADEBC</td>
<td></td>
<td></td>
<td></td>
<td>4,E</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>ADEBCF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Dijkstra’s algorithm, discussion

**Algorithm computational 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:
  - \( O(n \log n) \): Use a heap (sorted) to maintain interim table

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

![Graph Diagram](image)
Distance Vector Routing Algorithm

**iterative:**
- continues until no nodes exchange info.
- *self-terminating:* no “signal” to stop

**asynchronous:**
- nodes need *not* exchange info/iterate in lock step!

**distributed:**
- each node communicates *only* with directly-attached neighbors

**Distance Table data structure**
- each node has its own
- row for each possible destination
- column for each directly-attached neighbor to node
Distance Table: example

\[ D^E(C,D) = c(E,D) + \min_w \{D^D(C,w)\} = 2 + 2 = 4 \]

\[ D^E(A,D) = c(E,D) + \min_w \{D^D(A,w)\} = 2 + 3 = 5 \quad \text{loop!} \]

\[ D^E(A,B) = c(E,B) + \min_w \{D^B(A,w)\} = 8 + 6 = 14 \quad \text{loop!} \]
Distance table gives routing table

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>14</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>6</td>
<td>9</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>4</td>
<td>11</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Destination</th>
<th>Outgoing link to use, cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A,1</td>
</tr>
<tr>
<td>B</td>
<td>D,5</td>
</tr>
<tr>
<td>C</td>
<td>D,4</td>
</tr>
<tr>
<td>D</td>
<td>D,4</td>
</tr>
</tbody>
</table>

Distance table \( \rightarrow \) Routing table
Distance Vector Routing: overview

Iterative, asynchronous:
- each local iteration caused by:
  - local link cost change
  - message from neighbor: its least cost path change from neighbor

Distributed:
- each node notifies neighbors only when its least cost path to any destination changes
  - neighbors then notify their neighbors if necessary

Each node:
- wait for (change in local link cost of msg from neighbor)
- recompute distance table
- if least cost path to any dest has changed, notify neighbors
Distance Vector: link cost changes

Link cost changes:
- node detects local link cost change
- updates distance table (line 15)
- if cost change in least cost path, notify neighbors (lines 23, 24)

“good news travels fast”
Distance Vector: link cost changes

Link cost changes:
- good news travels fast
- bad news travels slow

Algorithm continues on!
Distance Vector: 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)
- Does not work on larger loops

![Diagram of distance vector algorithm](chart)
Comparison of LS and DV algorithms

Message complexity
- **LS:** with $n$ nodes, with an average of $l$ links/node, each node sends $O(nl)$. Total messages $O(n^2l)$
- **DV:** exchange between neighbors only
  - 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
Hierarchical Routing

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

scale: with 50 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 can run different intra-AS routing protocol

**gateway routers**
- special routers in AS
- run intra-AS routing protocol with all other routers in AS
- *also* responsible for routing to destinations outside AS
  - run *inter-AS routing* protocol with other gateway routers
Why different Intra- and Inter-AS routing?

Policy:
- Inter-AS: admin wants control over how its traffic is routed and 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
Intra-AS and Inter-AS routing

Gateways:
- perform inter-AS routing amongst themselves
- perform intra-AS routers with other routers in their AS

inter-AS, intra-AS routing in gateway A.c

network layer
link layer
physical layer
Intra-AS and Inter-AS routing

Inter-AS routing between A and B

Intra-AS routing within AS A

Intra-AS routing within AS B

Host h1

Host h2
Routing in the Internet

- The Global Internet consists of Autonomous Systems (AS) interconnected with each other:
  - Stub AS: small corporation
  - Multihomed AS: large corporation (no transit)
  - Transit AS: provider

- Two-level routing:
  - Intra-AS: administrator is responsible for choice
  - Inter-AS: unique standard
Internet AS Hierarchy

Intra-AS border (exterior gateway) routers

Inter-AS interior (gateway) routers
Intra-AS Routing

- Also known as **Interior Gateway Protocols (IGP)**
- Most common IGPs:
  - 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)
  - *Can you guess why?*

- Distance vectors: exchanged every 30 sec via Response Message (also called advertisement)

- Each advertisement: routes for up to 25 destination nets
### RIP (Routing Information Protocol)

#### Routing Table in D

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

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 routed (daemon)**
- **Advertisements sent in UDP packets, periodically repeated**

![Diagram of network layers with routed process]
RIP Table example (continued)

Router: *giroflee.eurocom.fr*

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>Flags</th>
<th>Ref</th>
<th>Use</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>UH</td>
<td>0</td>
<td>26492</td>
<td>lo0</td>
</tr>
<tr>
<td>192.168.2.1</td>
<td>192.168.2.5</td>
<td>U</td>
<td>2</td>
<td>13</td>
<td>fa0</td>
</tr>
<tr>
<td>193.55.114.1</td>
<td>193.55.114.6</td>
<td>U</td>
<td>3</td>
<td>58503</td>
<td>le0</td>
</tr>
<tr>
<td>192.168.3.1</td>
<td>192.168.3.5</td>
<td>U</td>
<td>2</td>
<td>25</td>
<td>qaa0</td>
</tr>
<tr>
<td>224.0.0.0</td>
<td>193.55.114.6</td>
<td>U</td>
<td>3</td>
<td>0</td>
<td>le0</td>
</tr>
<tr>
<td>default</td>
<td>193.55.114.129</td>
<td>UG</td>
<td>0</td>
<td>143454</td>
<td></td>
</tr>
</tbody>
</table>

- Three attached class C networks (LANs)
- Router only knows routes to attached LANs
- Default router used to “go up”
- Route multicast address: 224.0.0.0
- Loopback interface (for debugging)
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)
OSPF “advanced” features (not in RIP)

- **Security**: all OSPF messages authenticated (to prevent malicious intrusion); TCP connections used
- **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
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 ASs.
IGRP (Interior Gateway Routing Protocol)

- CISCO proprietary; successor of RIP (mid 80s)
- Distance Vector, like RIP
  - Hold time
  - Split Horizon
  - Poison Reverse
- several cost metrics (delay, bandwidth, reliability, load etc)
- uses TCP to exchange routing updates
- EIGRP (Garcia-Luna): Loop-free routing via Distributed Updating Alg. (DUAL) based on *diffused computation*
  - Uses a mix of link-state and distance vector
Inter-AS routing
Internet inter-AS routing: BGP

- **BGP (Border Gateway Protocol):** *the de facto standard*
- **Path Vector protocol:**
  - similar to Distance Vector protocol
  - each Border Gateway broadcast to neighbors (peers) *entire path* (i.e, sequence of ASs) to destination
  - E.g., Gateway X may send its path to dest. Z:

  \[
  \text{Path (X,Z)} = X,Y_1,Y_2,Y_3,\ldots,Z
  \]
Internet inter-AS routing: BGP

**Suppose:** gateway X send its path to peer gateway W

- W may or may not select path offered by X
  - cost, policy (don’t route via competitors AS), loop prevention reasons.

- If W selects path advertised by X, then:
  \[ \text{Path (W,Z)} = w, \text{Path (X,Z)} \]

- Note: X can control incoming traffic by controlling it route advertisements to peers:
  - e.g., don’t want to route traffic to Z -> don’t advertise any routes to Z
Internet inter-AS routing: BGP

- 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
Other Routing Techniques

- **Hot-Potato Routing a.k.a Deflection Routing**
  - Use the first available link irrespective of whether it leads to the destination or not.

- **Cut Through routing**
  - Non-store and forward: Routes before entire packet is received at the router.
  - Outgoing link is reserved. What happens if a fast link succeeds a slow link?
Reading

● Required
  – *End-To-End Routing Behavior in the Internet*, V. Paxson, SIGCOMM 1996.
    • Due: 3/26/01
  – *Persistent Route Oscillations in Inter-Domain Routing*, K. Varadhan, R. Govindan, D. Estrin,
    • Due: 3/28/01

● Recommended