Communication Networks
The University of Kansas EECS 780
Network Routing

James P.G. Sterbenz

Department of Electrical Engineering & Computer Science
Information Technology & Telecommunications Research Center
The University of Kansas

jgps@eecs.ku.edu

http://www.ittc.ku.edu/~jgps/courses/nets
Network Routing
Outline

NR.1  Functions and services
NR.2  Routing algorithms
NR.3  PSTN routing architecture and algorithms
NR.4  Internet routing architecture and protocols
NR.5  Broadcast, multicast, and anycast routing
NR.6  Overlay routing and DHTs
Network Routing

NR.1  Functions and Services

NR.1  Functions and services
NR.2  Routing algorithms
NR.3  PSTN routing architecture and algorithms
NR.4  Internet routing architecture and protocols
NR.5  Broadcast, multicast, and anycast routing
NR.6  Overlay routing and DHTs
Network Layer Control

Hybrid Layer/Plane Cube

Layer 3: routing in control plane
Network Layer
Service and Interfaces

• Network layer 3 is above link layer 2
  – *addressing*: network layer identifier for end systems (hosts)
  – *forwarding*: transfers packets hop-by-hop
    • using link layer services
    • network layer responsible for determining *which* next hop
  – *routing*: determination of path to forward packets
  – *signalling*: messages to control network layer behaviour
  – *traffic management*: management of traffic and congestion

• Network layer service to transport layer (L4)
  – deliver TPDUs to destination transport entity
Network Layer
Forwarding vs. Routing

- **Forwarding** transfers packets hop-by-hop
  - each switch (router) makes decision on which link to send
  - forwarding table (generally) used to make decision
  - forwarding is *per packet* decision
  - [analogy: determining which exits to take on a drive]

- **Routing** determines the path to take
  - routing algorithm independent of forwarding
  - forwarding table entries populated by routing
  - routing is (generally) not done per packet
  - [analogy: planning trip from source to destination]

*Forwarding and routing are very different*
Switches
Functions: Routing and Signalling

- Routing
  - asynchronous w.r.t. forwarding
  - not part of critical path
  - may use topology and link state
  - uses signalling to coördinate among nodes
Routing

Group Communication Terminology

- **Unicast (point-to-point)**
  - single source \[\leftarrow\] → single destination (uni- or bi-directional)
- **Anycast**
  - single source → any one (appropriate) destination
- **$k$-cast**
  - single source → $k$ (appropriate) destinations
Routing
Group Communication Terminology

- **Multicast**
  - point-to-multipoint
    - single source $\rightarrow$ multiple destinations
  - reverse multicast (multipoint-to-point)
    - multiple sources $\rightarrow$ single destination
  - multipoint-to-multipoint
    - multiple sources $[\leftarrow] \rightarrow$ multiple destinations

- **Broadcast**
  - single source $\rightarrow$ all destinations (in a given subnetwork)
Network Routing
NR.2 Routing Algorithms

NR.1 Functions and services
NR.2 Routing algorithms
   NR.2.1 Overview, static and source routing
   NR.2.2 Link state routing
   NR.2.3 Distance vector routing
   NR.2.4 Comparison and hierarchy
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
Routing
Network Model

• Network is modeled as a graph \( N = (V,E) \)
  
  - nodes are vertices \( v_i \in V \)
  - links are edges \( e_{ij} = (v_i, v_j) \in E \)
    
    • \( v_i \) and \( v_j \) are neighbours
    • edge weights are costs

• Example:
  
  - \( V = \{u, v, w, x, y, z\} \)
  - \( E = \{(u,v),(u,w),(u,x),(v,x),(v,w),(x,w),(x,y),(w,y),(w,z),(y,z)\} \)
Routing

Network Model

• Each link (edge) has a cost $c(v_i, v_j)$
  – cost is metric of interest, e.g. hop count, latency, capacity
  – assume $c(v_i, v_j) = c(v_j, v_i)$
    • graph is not directed
  – $\forall (v_i, v_j) \notin E, c(v_j, v_i) = \infty$

• Sequence of edges (links) is path between nodes
  – $\text{path}(v_j, v_{i+n}) = (v_i, v_{i+1}, \ldots v_{j+n})$

• Cost of a path is sum of link costs
  – $c(\text{path}(v_j, v_{i+n})) = \sum c(v_{i}, v_{i+1})$

• Routing algorithm goal (ideal): find least cost path
  – shortest path if all link costs equal (measures hops)
Network Routing

NR.2.1 Overview, Static and Source Routing

NR.1 Functions and services

NR.2 Routing algorithms
  NR.2.1 Overview, static and source routing
  NR.2.2 Link state routing
  NR.2.3 Distance vector routing
  NR.2.4 Comparison and hierarchy

NR.3 PSTN routing architecture and algorithms

NR.4 Internet routing architecture and protocols

NR.5 Broadcast, multicast, and anycast routing

NR.6 Overlay routing and DHTs
Routing Algorithms

Classification by State

- State (topology and link costs)
- Global
  - complete* state located in each switch
    * this will be relaxed for hierarchical routing
  - example: link state routing
Routing Algorithms
Classification by State

• State (topology and link costs)
• Global
  – complete* state located in each switch
    * this will be relaxed for hierarchical routing
  – example: link state routing
• Decentralised
  – neighbourhood state only in each switch
  – iterative process of route computation with neighbors
  – example: distance vector routing
Routing Algorithms
Classification by Dynamicity

- **Dynamicity**: response to topology and traffic changes
- **Static**
  - routes do not change (directly)
  - changed by human operators or network management
- **Dynamic (adaptive)**:
  - routes change in response to topology and traffic
  - route computation may occur
    - periodically
    - in direct response to changes in topology and traffic
Routing Algorithms
Classification by Load-Sensitivity

- Load-sensitivity
- Load-sensitive: routing responds to loads on links
  - load contributes to link weight
  - paths chosen along uncongested links
- Load-insensitive: routing doesn’t respond to link load
Routing Algorithms
Characteristics and Goals

- Stability
  - routes to do not oscillate between paths
  - harder to maintain with capacity-based dynamic routing

- Low complexity and overhead
  - processing load in switches
  - signalling message overhead on links

- Optimality
  - optimal paths for communicating nodes
  - optimal use of network resources

Very difficult to achieve all of these!
Static Routing
Overview

- Network engineering determines routes
  - frequently based on traffic matrix
- Forwarding tables are *provisioned* by an operator

*When does this make sense?*
Static Routing

Overview

• Network engineering determines routes
  – frequently based on traffic matrix
• Forwarding tables are *provisioned* by an operator
• Logical choice for:
  – small networks
  – networks with predictable stable traffic
Source Routing
Overview

• Node constructs entire path of packet
  – end system or edge node
  – *source route* carried in packet headers
  – routers simply pop source route stack to forward
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end (*why?*)
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end
Source Routing Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end (why?)
Source Routing

Example

- Source puts route in header
  - each hop forwards to first address
  - rotates to end
  - reverse path accumulated so D can reply to S
Source Routing

Issues?
Source Routing

Issues

• Simple in theory
  – but some entity has to create the source route
  – edge needs significant knowledge of topology

• Rarely used
  – almost never used IP option
  – some specialised problem domains
    • e.g. MANET DSR  *Lecture MW*

• Proposed for future Internet, e.g. PoMo
Dynamic Routing
Overview

- Routes change in response to topology and traffic
- Classes of dynamic routing algorithms
  - link state
  - distance vector
Network Routing

NR.2.2  Link State Routing Algorithms

NR.1  Functions and services

NR.2  Routing algorithms
   NR.2.1  Overview, static and source routing
   NR.2.2  Link state routing
   NR.2.3  Distance vector routing
   NR.2.4  Comparison and hierarchy

NR.3  PSTN routing architecture and algorithms

NR.4  Internet routing architecture and protocols

NR.5  Broadcast, multicast, and anycast routing

NR.6  Overlay routing and DHTs
Link State Routing
Overview

• Use *link* state to compute optimal paths

• Steps:
  – neighbor discovery
  – determine cost of each link
    • e.g. measure delay, load
  – build local link state (for distribution)
  – flood link state
  – compute routes (e.g. Dijkstra algorithm)
Link State Routing

Dijkstra Algorithm

- Network topology and link costs known to all nodes
  - accomplished via link state flooding (broadcast)
  - all nodes have consistent information
- 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 $k$ destinations
Link State Routing

Dijkstra Algorithm Notation

- **Notation:**
  - \( c(x,y) \): link cost from node \( x \) to \( y \); \( = \infty \) if not neighbours
  - \( D(v) \): 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

[Kurose–Ross p.356]
Link State Routing

Dijkstra Algorithm

1. Initialisation:
2. \( N' = \{u\} \)
3. for all nodes \( v \)
4. if \( v \) is neighbour of \( u \)
5. then \( D(v) = c(u,v) \) /* cost of neighbour known */
6. else \( D(v) = \infty \) /* cost of others unknown */

[Kurose–Ross p.355]
Link State Routing

Dijkstra Algorithm

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 \( N' \)

[Kurose–Ross p.355]
## Link State Routing
### Dijkstra Example

<table>
<thead>
<tr>
<th>step</th>
<th>$N'$</th>
<th>$D_r(p)(v)$</th>
<th>$D_r(p)(w)$</th>
<th>$D_r(p)(x)$</th>
<th>$D_r(p)(y)$</th>
<th>$D_r(p)(z)$</th>
<th>notes</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>
<td>5 not min to $w$</td>
</tr>
</tbody>
</table>

![Diagram of Link State Routing example](attachment:image.png)

[Kurose–Ross p.356]

29 October 2018  
KU EECS 780 – Comm Nets – Routing  
NET-NR-39
Link State Routing
Dijkstra Example

<table>
<thead>
<tr>
<th>step</th>
<th>$N'$</th>
<th>$D_{r,p}(v)$</th>
<th>$D_{r,p}(w)$</th>
<th>$D_{r,p}(x)$</th>
<th>$D_{r,p}(y)$</th>
<th>$D_{r,p}(z)$</th>
<th>notes</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>
<td>5 not min to $w$</td>
</tr>
<tr>
<td>1</td>
<td>$ux$</td>
<td>2,$u$</td>
<td>4,$x$</td>
<td>1,$u$</td>
<td>2,$x$</td>
<td>$\infty$</td>
<td>shorter to $w$</td>
</tr>
</tbody>
</table>

[Link State Routing Dijkstra Example]

[Kurose–Ross p.356]
Link State Routing

Dijkstra Example

<table>
<thead>
<tr>
<th>step</th>
<th>$N'$</th>
<th>$D_r(p)(v)$</th>
<th>$D_r(p)(w)$</th>
<th>$D_r(p)(x)$</th>
<th>$D_r(p)(y)$</th>
<th>$D_r(p)(z)$</th>
<th>notes</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>
<td>5 not min to $w$</td>
</tr>
<tr>
<td>1</td>
<td>$ux$</td>
<td>$2,u$</td>
<td>$4,x$</td>
<td>$1,u$</td>
<td>$2,x$</td>
<td>$\infty$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>2</td>
<td>$uxy$</td>
<td>$2,u$</td>
<td>$3,y$</td>
<td>$1,u$</td>
<td>$2,x$</td>
<td>$4,y$</td>
<td>shorter to $w$</td>
</tr>
</tbody>
</table>

[Link State Routing]

[Dijkstra Example]

[Kurose–Ross p.356]
# Link State Routing

## Dijkstra's Example

<table>
<thead>
<tr>
<th>step</th>
<th>$N'$</th>
<th>$D_s p_v$</th>
<th>$D_s p_w$</th>
<th>$D_s p_x$</th>
<th>$D_s p_y$</th>
<th>$D_s p_z$</th>
<th>notes</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>
<td>5 not min to $w$</td>
</tr>
<tr>
<td>1</td>
<td>$ux$</td>
<td>2, $u$</td>
<td>4, $x$</td>
<td>1, $u$</td>
<td>2, $x$</td>
<td>$\infty$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>2</td>
<td>$uxy$</td>
<td>2, $u$</td>
<td>3, $y$</td>
<td>1, $u$</td>
<td>2, $x$</td>
<td>4, $y$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>3</td>
<td>$uxyv$</td>
<td>2, $u$</td>
<td>3, $y$</td>
<td>1, $u$</td>
<td>2, $x$</td>
<td>4, $y$</td>
<td>$w$ and $x$ stay</td>
</tr>
</tbody>
</table>

---

[Kurose–Ross p.356]
## Link State Routing
### Dijkstra Example

<table>
<thead>
<tr>
<th>step</th>
<th>$N'$</th>
<th>$D_p(v)$</th>
<th>$D_p(w)$</th>
<th>$D_p(x)$</th>
<th>$D_p(y)$</th>
<th>$D_p(z)$</th>
<th>notes</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>
<td>5 not min to $w$</td>
</tr>
<tr>
<td>1</td>
<td>$ux$</td>
<td>$2,u$</td>
<td>$4,x$</td>
<td>$1,u$</td>
<td>$2,x$</td>
<td>$\infty$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>2</td>
<td>$uxy$</td>
<td>$2,u$</td>
<td>$3,y$</td>
<td>$1,u$</td>
<td>$2,x$</td>
<td>$4,y$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>3</td>
<td>$uxvy$</td>
<td>$2,u$</td>
<td>$3,y$</td>
<td>$1,u$</td>
<td>$2,x$</td>
<td>$4,y$</td>
<td>$w$ and $x$ stay</td>
</tr>
<tr>
<td>4</td>
<td>$uxyvw$</td>
<td>$2,u$</td>
<td>$3,y$</td>
<td>$1,u$</td>
<td>$2,x$</td>
<td>$4,y$</td>
<td>$x$, $y$, and $z$ stay</td>
</tr>
</tbody>
</table>

[Link State Routing](Kurose–Ross p.356)
Link State Routing

Dijkstra Example

<table>
<thead>
<tr>
<th>step</th>
<th>( N' )</th>
<th>( D,u_p(v) )</th>
<th>( D,u_p(w) )</th>
<th>( D,u_p(x) )</th>
<th>( D,u_p(y) )</th>
<th>( D,u_p(z) )</th>
<th>notes</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>
<td>5 not min to ( w )</td>
</tr>
<tr>
<td>1</td>
<td>( ux )</td>
<td>2,( u )</td>
<td>4,( x )</td>
<td>1,( u )</td>
<td>2,( x )</td>
<td>( \infty )</td>
<td>shorter to ( w )</td>
</tr>
<tr>
<td>2</td>
<td>( uxy )</td>
<td>2,( u )</td>
<td>3,( y )</td>
<td>1,( u )</td>
<td>2,( x )</td>
<td>4,( y )</td>
<td>shorter to ( w )</td>
</tr>
<tr>
<td>3</td>
<td>( uxyv )</td>
<td>2,( u )</td>
<td>3,( y )</td>
<td>1,( u )</td>
<td>2,( x )</td>
<td>4,( y )</td>
<td>( w ) and ( x ) stay</td>
</tr>
<tr>
<td>4</td>
<td>( uxyvw )</td>
<td>2,( u )</td>
<td>3,( y )</td>
<td>1,( u )</td>
<td>2,( x )</td>
<td>4,( y )</td>
<td>( x, y, ) and ( z ) stay</td>
</tr>
<tr>
<td>5</td>
<td>( uxyvwz )</td>
<td>2,( u )</td>
<td>3,( y )</td>
<td>1,( u )</td>
<td>2,( x )</td>
<td>4,( y )</td>
<td>( w ) stays</td>
</tr>
</tbody>
</table>

[Kurose–Ross p.356]
## Link State Routing

### Dijkstra Example

<table>
<thead>
<tr>
<th>step</th>
<th>$N'$</th>
<th>$D,v_p$</th>
<th>$D,w_p$</th>
<th>$D,x_p$</th>
<th>$D,y_p$</th>
<th>$D,z_p$</th>
<th>notes</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>
<td>5 not min to $w$</td>
</tr>
<tr>
<td>1</td>
<td>$ux$</td>
<td>2,$u$</td>
<td>4,$x$</td>
<td>1,$u$</td>
<td>2,$x$</td>
<td>$\infty$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>2</td>
<td>$uxy$</td>
<td>2,$u$</td>
<td>3,$y$</td>
<td>1,$u$</td>
<td>2,$x$</td>
<td>4,$y$</td>
<td>shorter to $w$</td>
</tr>
<tr>
<td>3</td>
<td>$uxyv$</td>
<td>2,$u$</td>
<td>3,$y$</td>
<td>1,$u$</td>
<td>2,$x$</td>
<td>4,$y$</td>
<td>$w$ and $x$ stay</td>
</tr>
<tr>
<td>4</td>
<td>$uxyvw$</td>
<td>2,$u$</td>
<td>3,$y$</td>
<td>1,$u$</td>
<td>2,$x$</td>
<td>4,$y$</td>
<td>$x$, $y$, and $z$ stay</td>
</tr>
<tr>
<td>5</td>
<td>$uxyvwz$</td>
<td>2,$u$</td>
<td>3,$y$</td>
<td>1,$u$</td>
<td>2,$x$</td>
<td>4,$y$</td>
<td>$w$ stays</td>
</tr>
</tbody>
</table>

Also keep track of next hop for shortest paths

[Kurose–Ross p.356]
Link State Routing
Oscillations

- Oscillations between paths
  - link cost = load carried on link in a given direction
  - asymmetric: \( c(x,y) \) not necessarily equal to \( c(y,x) \)

- Traffic on a given flow changes link state
  - and may cause it to switch to the other path
  - and back-and-forth again

todo: figure
Network Routing

NR.2.3 Distance Vector Routing Algorithms

NR.1 Functions and services
NR.2 Routing algorithms
   NR.2.1 Overview, static and source routing
   NR.2.2 Link state routing
   NR.2.3 Distance vector routing
   NR.2.4 Comparison and hierarchy
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
Distance Vector Routing

Overview

• In a fully distributed manner nodes:
  – periodically send *distance vector* estimates to neighbours
  – update their own estimate when received from neighbours
  – algorithm converges and is self terminating

• Bellman-Ford equation:
  – define \( d_x(y) \) as cost of least-cost path from \( x \) to \( y \)
  – \( d_x(y) = \min_v\{c(x,v) + d_v(y)\} \) over all neighbours of \( x \)

[Kurose–Ross p.358]
Distance Vector Routing
Bellman-Ford Example

\[ d_v(z) = 5, \quad d_x(z) = 3, \quad d_w(z) = 3 \quad [\text{by inspection}] \]

\[ d_u(z) = \min \{ \ c(u,v)+d_v(z) , \ c(u,x)+d_x(z) , \ c(u,w)+d_w(z) \} \]
\[ = \min \{ \ 2 + 5 , \ 1 + 3 , \ 5 + 3 \} \]
\[ = 4 \]
Distance Vector Routing

Notation

- **Notation:**

  - $N$ set of all nodes
  - $c(x, v)$ link cost from node $x$ to $v$
  - $D_x(y)$ estimate of least cost from $x$ to $y$
  - $D_x(y)$ distance vector = $[D_x(y)]$ for all $y \in N$
Distance Vector Routing Algorithm

- Node $x$ knows cost to each neighbor $v$: $c(x,v)$
- Node $x$ maintains $D_x(y) = [D_x(y)]$ for all $y \in N$
- Node $x$ also maintains neighbours’ distance vectors
  - for each neighbour $v$, $x$ maintains $D_v(y) = [D_v(y)]$ for all $y \in N$
- Each node:
  - **wait** for change in local link cost message from neighbour
  - **recompute** estimates
  - if $D$ to any destination has changed, **notify** neighbours

[Kurose–Ross p.359]
## Distance Vector Routing

### Example

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>y</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>z</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>x</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>z</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Z</td>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>x</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>y</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>

[Kurose–Ross p.362]
## Distance Vector Routing

### Example

<table>
<thead>
<tr>
<th>X</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>y</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>z</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Y</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Y</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Z</th>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>y</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Note: node only have estimates for all paths in second step because max path between nodes is 2 in this small example.

[Kurose–Ross p.362]
## Distance Vector Routing

### Example

**Initial Table:**

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>y</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>z</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

**Updated Table:**

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Neighbor Table:**

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

**Destination Table:**

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>7</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**Distance Calculation:**

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

---

[Kurose–Ross p.362]
**Distance Vector Routing**

**Example**

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0 2 7</td>
<td>∞ ∞ ∞</td>
<td>∞ ∞ ∞</td>
</tr>
<tr>
<td>Y</td>
<td>∞ ∞ ∞</td>
<td>2 0 1</td>
<td>∞ ∞ ∞</td>
</tr>
<tr>
<td>Z</td>
<td>∞ ∞ ∞</td>
<td>7 1 0</td>
<td>3 1 0</td>
</tr>
</tbody>
</table>

*Y, X, Z are nodes in the network.*

![Network Diagram](image)

**Done**

[Kurose–Ross p.362]
Distance Vector Routing
Algorithm

- Iterative and asynchronous – each local iteration caused by:
  - local link cost change
  - distance vector update message from neighbour
- Distributed:
  - each node notifies neighbors only when its vector changes
  - neighbours then notify their neighbours if necessary
- Self-terminating
  - no messages when no changes need to be propagated
Distance Vector Routing

Link Cost Changes

- Link cost changes
  - node detects local link cost change
  - updates routing information
  - recalculates distance vector
  - if distance vector changes notify neighbours

- Decrease in cost converges quickly
  - “good news travels fast”

- Increase in cost converges slowly
  - “bad news travels slowly”
  - count to infinity problem
  - routing loops
Network Routing

NR.2.4 Comparison and Hierarchy

NR.1 Functions and services

NR.2 Routing algorithms
  NR.2.1 Overview, static and source routing
  NR.2.2 Link state routing
  NR.2.3 Distance vector routing
  NR.2.4 Comparison and hierarchy

NR.3 PSTN routing architecture and algorithms

NR.4 Internet routing architecture and protocols

NR.5 Broadcast, multicast, and anycast routing

NR.6 Overlay routing and DHTs
Routing Algorithms
Comparison

- Link state vs. distance vector
- Comparison of characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Link State</th>
<th>Distance Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message complexity</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>State maintenance</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Convergence time</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robustness to errors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Routing Algorithms
Comparison

- Link state vs. distance vector
- Comparison of characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Link State</th>
<th>Distance Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message complexity</td>
<td>flooding link state</td>
<td>neighbour exchange</td>
</tr>
<tr>
<td>State maintenance</td>
<td>full (sub-)network</td>
<td>neighbours</td>
</tr>
<tr>
<td>Convergence time</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Robustness to errors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scalability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Routing Algorithms

Comparison

- Link state vs. distance vector
- Comparison of characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Link State</th>
<th>Distance Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message complexity</td>
<td>flooding link state</td>
<td>neighbour exchange</td>
</tr>
<tr>
<td>State maintenance</td>
<td>full (sub-)network</td>
<td>neighbours</td>
</tr>
<tr>
<td>Convergence time</td>
<td>fast</td>
<td>slow</td>
</tr>
<tr>
<td>Robustness to errors</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Scalability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Routing Algorithms

### Comparison

- Link state vs. distance vector
- Comparison of characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Link State</th>
<th>Distance Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message complexity</td>
<td>flooding link state</td>
<td>neighbour exchange</td>
</tr>
<tr>
<td>State maintenance</td>
<td>full (sub-)network</td>
<td>neighbours</td>
</tr>
<tr>
<td>Convergence time</td>
<td>fast</td>
<td>slow</td>
</tr>
<tr>
<td>Robustness to errors</td>
<td>errors limited to link(s)</td>
<td>errors propagate</td>
</tr>
<tr>
<td>Scalability</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Routing Algorithms
Comparison Summary

- Link state vs. distance vector
- Comparison of characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Link State</th>
<th>Distance Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message complexity</td>
<td>flooding link state</td>
<td>neighbour exchange</td>
</tr>
<tr>
<td>State maintenance</td>
<td>full (sub-)network</td>
<td>neighbours</td>
</tr>
<tr>
<td>Convergence time</td>
<td>fast</td>
<td>slow</td>
</tr>
<tr>
<td>Robustness to errors</td>
<td>errors limited to link(s)</td>
<td>errors propagate</td>
</tr>
<tr>
<td>Scalability</td>
<td>reasonable (with hierarchy)</td>
<td>poor (long convergence)</td>
</tr>
</tbody>
</table>
Routing Algorithms

Comparison Details

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

- **State maintenance**
  - LS: each node maintains state for entire (sub)network
    - significant problem in large networks
  - DV: each node maintains state only for neighbours
Routing Algorithms
Comparison Details

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

- Robustness to switch malfunction
  - LS: node can advertise incorrect link cost
    - each node computes only its own table
  - DV: node can advertise incorrect path cost
    - each node’s table used by others
    - error propagate thru network
Routing Algorithms

Scalability Challenges

• Scalability of routing algorithms
  – processing and bandwidth: message complexity
  – memory: state maintained

• Link state scalability challenges
  – flooding of link state in large network
  – large topology data bases

• Distance vector scalability challenges
  – long path lengths

Solution?
Routing Algorithms
Hierarchical Routing Concepts

- Hierarchy
  - divide network into clusters
  - isolate full topology and link state in lowest layers
  - higher layers aggregate

![Diagram of hierarchical routing concepts]
Routing Algorithms
Resilience and Security

- Routing security
  - authentication to resist denial of service attacks
  - encryption to resist traffic analysis
- Routing fault resilience
  - fault tolerant: robust to natural failures
    - e.g. redundancy
  - survivable: resilient to attack and disaster
    - e.g. geographic diversity
  - resilient: tolerant to environmental challenges and traffic
    - survivable
    - mobility, weak wireless channels, unpredictably large delay
    - unusual but legitimate traffic (e.g. flash crowd)

Lecture SR
Network Routing

NR.3 PSTN Routing Architecture and Algorithms

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
  NR.3.1 Fixed HIERarchical routing
  NR.3.2 DNHR: dynamic non-hierarchical routing
  NR.3.3 RTNR: real–time network routing
  NR.3.4 Other dynamic routing algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
PSTN Routing
Background and Overview

• PSTN evolved from a strict hierarchy
  – hierarchical E.164 addressing
  – with a (almost) single entity in technical control in the US

• Initial routing algorithm matches hierarchy
  – HIER

• Later algorithms relaxed strict hierarchy
  – better load balancing
  – necessary for NPA overlay and number portability
US PSTN
Traditional Local Network Structure

- Final trunk group
  (to class 4 long distance switching)

- Tandem trunk group
- Local loops
- Local switches
- Direct trunk group

Local office

- Local tandem switches
- Direct trunk group

Tandem office
PSTN Routing

NR.3.1 Fixed Hierarchical Routing

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
  NR.3.1 Fixed Hierarchical routing
  NR.3.2 DNHR: dynamic non-hierarchical routing
  NR.3.3 RTNR: real-time network routing
  NR.3.4 Other dynamic routing algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
PSTN Fixed Hierarchical Routing

Overview

• Fixed Hierarchical routing (HIER)
  – static routing algorithm based on strict hierarchy

• Switch class hierarchy
  5: local
  4: toll
  3: primary
  2: regional
  1: sectional

• Link types
  – final
  – high usage
PSTN Hierarchical Routing

Link Types

- **Final links**
  - each switch *homed* to single switch above
  - top level interconnected
  - *ladder*: vertical final link set
PSTN Hierarchical Routing

Link Types

- **High-usage links**
  - additional direct trunks
  - deployed as needed based on traffic matrix
  - reduce num. of hops below max. of 9
  - reduce bottlenecks in lower classes
US PSTN
Traditional Local Network Structure

- final trunk group
  (to class 4 long distance switching)

- local (class 5) switches
- direct trunk group

- local loops
- tandem office
- tandem trunk group
- direct trunk group
- local office
- final trunk group
- direct trunk group
PSTN Fixed Hierarchical Routing
Routing Design Rules

• Routing *design rules* define routing algorithm
  – constrain space of alternate choice and prevent looping

• Rules
  – two-ladder limit
  – intraladder direction
  – multiple switching function
  – one-level limit
  – switch low
  – directional routing
  – single route
  – alternate-route selection
PSTN Fixed Hierarchical Routing

Design Rules: Two-Ladder Limit

• Two-ladder limit
  – traffic only traverses origination and destination ladders
  – traffic cannot pass through a 3rd (different) ladder
PSTN Fixed Hierarchical Routing

Design Rules: Intraladder Direction

- Intraladder direction
  - traffic on originating ladder goes only upward
  - traffic on destination ladder goes only downward
  
  *why?*
PSTN Fixed Hierarchical Routing

Design Rules: Intraladder Direction

• Intraladder direction
  – traffic on originating ladder goes only upward
  – traffic on destination ladder goes only downward
  – prevents looping
PSTN Fixed Hierarchical Routing

Design Rules: Multiple Switching Function

- Multiple switching function
  - switch that performs switching for multiple classes
  - assume internal ladder spanning classes
PSTN Fixed Hierarchical Routing

Design Rules: One-Level Limit

- One-level limit
PSTN Fixed Hierarchical Routing
Design Rules: Switch Low

- Switch low
  - switch using the lowest possible level (highest class)
PSTN Fixed Hierarchical Routing

Design Rules: Directional Routing

- Directional routing
  - choose terminating ladder path first
PSTN Fixed Hierarchical Routing
Design Rules: Single Route

- Single route
  - choose direct interladder links before intraladder links
PSTN Fixed Hierarchical Routing

Design Rules: Alternate-Route Selection

• Alternate-route selection
  – alternate to loaded link is same as if link didn’t exist
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Goals
  - minimise number of hops
  - distribute load in lower classes

based on [Girad-1990 Fig 2.24]
PSTN Fixed Hierarchical Routing

Interladder Route Selection

• Order of link selection
  – assuming trunk exists
  – capacity available

• Hunt sequence
  – direct high-usage link
  • if switch LAMA capable
    (local automatic message accounting)
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
PSTN Fixed Hierarchical Routing
Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

• Order of link selection
  – assuming trunk exists
  – capacity available

• Hunt sequence
  – direct high-usage link
  – high-usage class 4 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
PSTN Fixed Hierarchical Routing
Interladder Route Selection

• Order of link selection
  – assuming trunk exists
  – capacity available

• Hunt sequence
  – direct high-usage link
  – high-usage class 4 links
  – high-usage class 3 links
  – high-usage class 2 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
PSTN Fixed Hierarchical Routing
Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing
Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing
Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

• Order of link selection
  – assuming trunk exists
  – capacity available

• Hunt sequence
  – direct high-usage link
  – high-usage class 4 links
  – high-usage class 3 links
  – high-usage class 2 links
  – high-usage class 1 links
PSTN Fixed Hierarchical Routing

Interladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available

- Hunt sequence
  - direct high-usage link
  - high-usage class 4 links
  - high-usage class 3 links
  - high-usage class 2 links
  - high-usage class 1 links
  - final links (9 hops)
    - final choice possible
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Goals
  - minimise number of hops
  - reduce final link load
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
  - reduce class 1 hops
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
  - reduce class 1 hops
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
  - reduce class 1 hops
  - reduce total/class 2 hops
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
  - reduce class 1 hops
  - reduce class 2 hops
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
  - reduce class 1 hops
  - reduce class 2 hops
PSTN Fixed Hierarchical Routing

Intraladder Route Selection

- Order of link selection
  - assuming trunk exists
  - capacity available
  - given intra-link selection

- Hunt sequence
  - direct high-usage trunk
  - reduce class 1 hops
  - reduce class 2 hops
PSTN Fixed Hierarchical Routing

Summary

Advantages?
PSTN Fixed Hierarchical Routing

Summary

• Advantages
  – simple: directly related to network topology
  – easy to implement in early relay-logic switches

Disadvantages?
PSTN Fixed Hierarchical Routing

Summary

• Advantages
  – simple: directly related to network topology
  – easy to implement in early relay-logic switches

• Disadvantages
  – inflexible: class 5 switches cannot serve as transit nodes
  – unable to support overlay area codes and number portability
  – unable to adapt to time-varying load

Alternatives?
PSTN Routing
NR.3.2  Dynamic Non-Hierarchical Routing

NR.1  Functions and services
NR.2  Routing algorithms
NR.3  PSTN routing architecture and algorithms
  NR.3.1  Fixed HIERarchical routing
  NR.3.2  DNHR: dynamic non-hierarchical routing
  NR.3.3  RTNR: real–time network routing
  NR.3.4  Other dynamic routing algorithms
NR.4  Internet routing architecture and protocols
NR.5  Broadcast, multicast, and anycast routing
NR.6  Overlay routing and DHTs
PSTN Routing
Non-Hierarchical Topology

- Modern PSTN lacks strict hierarchy of Bell System
  - divestiture and deregulation
    - multiple IXCs connecting ILECs and CLECs
  - NPA overlays
  - LNP (local number portability)
PSTN Routing
Non-Hierarchical Topology

Internet–PSTN gateways

CATV

IXCs

Internet VoIP

ILECs incumbent LECS

local ILEC loops: analog or DSL, fiber

Internet–PSTN gateways

CATV

IXCs

Internet VoIP

ILECs incumbent LECS

local ILEC loops: analog or DSL, fiber
PSTN Routing
Non-Hierarchical and Dynamic Routing

- Non-Hierarchical routing
  - removes requirement routing matches physical hierarchy
  - note: different from hierarchy for scalability (e.g. PNNI)
PSTN Routing
Non-Hierarchical and Dynamic Routing

- Non-Hierarchical routing
  - removes requirement routing matches physical hierarchy
  - note: different from hierarchy for scalability (e.g. PNNI)
  - planned before divestiture to increase efficiency of network

- Dynamic routing
  - permit routing to adapt to traffic
PSTN Non-Hierarchical Routing

DNHR Overview

• DNHR (dynamic non-hierarchical routing)
• AT&T long distance network
  – Bell Labs research in early 1980s
  – former classes 3 – 1 switches
PSTN Non-Hierarchical Routing

DNHR Topology

• Network is a mesh of switches
• Each switch pair connected by:
  – direct primary path
  – alternate two-hop paths
PSTN Non-Hierarchical Routing

DNHR Forwarding Tables

• Forwarding tables
  – max of 15 paths/destination
  – limited by switch computing power & memory of early 1980s

• Engineered paths
  – ≤ 10 preplanned paths (typically 3 or 4)
  – computed off-line every other weekend

• Real-time paths
  – used in case of congestion
  – computed by NOC every 5 minutes based on actual traffic
    • NOC = network operations center
PSTN Non-Hierarchical Routing
DNHR Time-Dependent Routing

- Forwarding tables swapped
- Time-dependent routing
  - tables changed several times per day

*why?
PSTN Non-Hierarchical Routing
DNHR Time-Dependent Routing

- Forwarding tables swapped
- Time-dependent routing
  - tables changed several times per day
  - traffic patterns change with business hours
    - e.g. mid-morning peak direct links congested within time zone
- Coarse time granularity for engineered paths
  - 10 periods during weekdays
  - 5 periods during weekend days
PSTN Non-Hierarchical Routing

DNHR Routing Algorithm

- DNHR routing algorithm
  - attempt to use primary path
  - if fully loaded (congested), hunt alternatives
    - engineered or real-time paths
  - crankback if 2nd hop loaded
  - metasatbility problems...
PSTN Non-Hierarchical Routing

DNHR Routing Stability

• DHNR alternate path problem:
  – each 2-hop path reduces capacity on *two* primary paths
  – as more alternate paths used...
    less likely primary path available

• DNHR Metastability
  – converge on majority of 2-hop alternate paths
  – sub-optimal stable state
  – solution: reserve fraction of traffic for primary paths
PSTN Non-Hierarchical Routing

DNHR Example

- Example 7-node mesh network
PSTN Non-Hierarchical Routing
DNHR Example

- NY → LA next hop
  - simplified to
    - 4 periods; 7 paths
      - engineered
      - dynamic

### Periods

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morning</td>
<td>LA, KC, DA, DC, DE, SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA, DA, KC, KC, DE, SF</td>
</tr>
<tr>
<td>Evening</td>
<td>LA, DC, DA, KC, DE, SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA, DC, LA, KC, DE, SF</td>
</tr>
</tbody>
</table>
PSTN Non-Hierarchical Routing
DNHR Example

- NY → LA next hop
  - morning period

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA  KC  DA  DC  DE  SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA  DA  KC  KC  DE  SF</td>
</tr>
<tr>
<td>Evening</td>
<td>LA  DC  DA  KC  DE  SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA  DC  LA  KC  DE  SF</td>
</tr>
</tbody>
</table>

[MR2007]
PSTN Non-Hierarchical Routing
DNHR Example

• NY → LA next hop
  1. try direct NY→LA

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA  KC  DA  DC  DE  SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA  DA  KC  KC  DE  SF</td>
</tr>
<tr>
<td>Evening</td>
<td>LA  DC  DA  KC  DE  SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA  DC  LA  KC  DE  SF</td>
</tr>
</tbody>
</table>
PSTN Non-Hierarchical Routing

DNHR Example

- **NY → LA next hop**
  1. try direct **NY → LA** else
  2. try **NY → KC**

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Morning</td>
<td>LA  KC  DA  DC  DE  SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA  DA  KC  KC  DE  SF</td>
</tr>
<tr>
<td>Evening</td>
<td>LA  DC  DA  KC  DE  SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA  DC  LA  KC  DE  SF</td>
</tr>
</tbody>
</table>
PSTN Non-Hierarchical Routing

DNHR Example

- **NY → LA next hop**
  1. try direct NY→LA else
  2. try NY→KC→LA

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA KC DA DC DE SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA DA KC KC DE SF</td>
</tr>
<tr>
<td>Evening</td>
<td>LA DC DA KC DE SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA DC LA KC DE SF</td>
</tr>
</tbody>
</table>

[MR2007]
PSTN Non-Hierarchical Routing
DNHR Example

- NY → LA next hop
  1. try direct NY→LA else
  2. try NY→KC→LA
     (attempt crankback KC→_)

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA  KC  DA  DC  DE  SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA  DA  KC  KC  DE  SF</td>
</tr>
<tr>
<td><strong>Evening</strong></td>
<td>LA  DC  DA  KC  DE  SF</td>
</tr>
<tr>
<td><strong>Weekend</strong></td>
<td>DA  DC  LA  KC  DE  SF</td>
</tr>
</tbody>
</table>
PSTN Non-Hierarchical Routing

DNHR Example

- **NY → LA next hop**
  1. try direct NY→LA else
  2. try NY→KC→LA
     (attempt crankback KC→_)

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA KC DA DC DE SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA DA KC KC DE SF</td>
</tr>
<tr>
<td><strong>Evening</strong></td>
<td>LA DC DA KC DE SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA DC LA KC DE SF</td>
</tr>
</tbody>
</table>

[MR2007]
PSTN Non-Hierarchical Routing
DNHR Example

- **NY → LA next hop**
  1. try direct NY→LA else
  2. try NY→KC→LA else
  3. try NY→DA

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA KC DA DC DE SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA DA KC KC DE SF</td>
</tr>
<tr>
<td>Evening</td>
<td>LA DC DA KC DE SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA DC LA KC DE SF</td>
</tr>
</tbody>
</table>

[MR2007]
PSTN Non-Hierarchical Routing
DNHR Example

- NY → LA next hop
  1. try direct NY → LA else
  2. try NY → KC → LA else
  3. try NY → DA → LA

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA  KC  DA  DC  DE  SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA  DA  KC  KC  DE  SF</td>
</tr>
<tr>
<td><strong>Evening</strong></td>
<td>LA  DC  DA  KC  DE  SF</td>
</tr>
<tr>
<td><strong>Weekend</strong></td>
<td>DA  DC  LA  KC  DE  SF</td>
</tr>
</tbody>
</table>
PSTN Non-Hierarchical Routing

DNHR Example

- NY → LA next hop
  1. try direct NY → LA else
  2. try NY → KC → LA else
  3. try NY → DA → LA done!

<table>
<thead>
<tr>
<th>Period</th>
<th>NY Forwarding Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Morning</strong></td>
<td>LA  KC  DA  DC  DE  SF</td>
</tr>
<tr>
<td>Afternoon</td>
<td>LA  DA  KC  KC  DE  SF</td>
</tr>
<tr>
<td><strong>Evening</strong></td>
<td>LA  DC  DA  KC  DE  SF</td>
</tr>
<tr>
<td>Weekend</td>
<td>DA  DC  LA  KC  DE  SF</td>
</tr>
</tbody>
</table>

[MR2007]
PSTN Routing
NR.3.3 Real-Time Network Routing

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
  NR.3.1 Fixed HIERarchical routing
  NR.3.2 DNHR: dynamic non-hierarchical routing
  NR.3.3 RTNR: real–time network routing
  NR.3.4 Other dynamic routing algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
PSTN Real-Time Network Routing

RTNR Overview

• RTNR (real-time network routing)
  – developed by Bell Labs in late 1980s to improve on DNHR
  – deployed in AT&T voice network 1991–present

• Motivation
  – DNHR was not adaptive enough
  – off-line computations and administration costly
  – increased switch processing and memory capabilities
PSTN Real-Time Network Routing

RTNR Forwarding and Routing

- **Forwarding tables**
  - change dynamically based on traffic conditions

- **Routing**
  - fully distributed algorithm
  - paths computed per call
    - used for *next* call to reduce call-setup delay

© James P.G. Sterbenz
PSTN Routing

NR.3.4 Other Dynamic Routing Algorithms

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
  NR.3.1 Fixed HIERarchical routing
  NR.3.2 DNHR: dynamic non-hierarchical routing
  NR.3.3 RTNR: real-time network routing
  NR.3.4 Other dynamic routing algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
PSTN Routing Algorithms

Other Dynamic Algorithms

- **DCR**: dynamically controlled routing
  - developed by BNR (Bell-Northern Research) for Canada

- **DAR**: dynamic alternate routing
  - developed by BT Labs (British Telecom)
Network Routing

NR.4 Internet Routing

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
  NR.4.1 Intradomain routing: RIP and OSPF
  NR.4.2 Interdomain routing: BGP
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
Internet Routing Topology

Current Internet Structure

tier-2 providers

IXP

IXP

peering points

tier-1 providers

local ISPs

access lines

local ISPs
Internet Routing Topology
AS Intra/Inter-Domain Hierarchy

- **Autonomous systems (ASs) or domains**
  - administrative autonomy
  - improves scalability
- Two-layer major hierarchy
  - intradomain within AS
  - interdomain between ASs
- Routing protocols loads forwarding tables
Network Routing

NR.4.1  Internet Intradomain Routing

NR.1  Functions and services
NR.2  Routing algorithms
NR.3  PSTN routing architecture and algorithms
NR.4  Internet routing architecture and protocols
   NR.4.1  Intradomain Routing: RIP and OSPF
   NR.4.2  Interdomain Routing: BGP
NR.5  Broadcast, multicast, and anycast routing
NR.6  Overlay routing and DHTs
Internet Routing
Intradomain Routing and Forwarding

- Intra-AS or *intradomain* routing
  - IGP (interior gateway protocol)
- IGP determines intra-AS entries
  - determines paths within AS
  - loads forwarding tables within AS
  - IP forwards
- Each AS can choose the IGP of its choice
  - may be hierarchical *within* AS
- Example IGPs
  - RIP, OSPF, ISIS, IGRP/EIGRP
Internet Routing

Interdomain Routing and Forwarding

• Inter-AS or *interdomain* routing
  – EGP (exterior gateway protocol)

• EGP determines inter-AS entries in edge nodes
  – determines inter-AS reachability for particular AS IP blocks
  – loads edge router forwarding tables
  – must also propagate within AS and load forwarding tables

• All Internet ASs *must* agree on the EGP
  – BGP-4: border gateway protocol is *de facto* standard
  – evolution difficult; replacement probably impossible
Intradomain Routing

Intradomain Protocols

- **RIP**: routing information protocol
  - original IGP
- **EIGRP**: (enhanced) interior gateway routing protocol
  - Cisco proprietary replacement for RIP
- **ISIS**: intermediate system – intermediate system
  - OSI IGP
- **OSPF**: open shortest path first
  - IETF replacement for RIP
- **OSPF or ISIS** used by virtually all service providers
  - distance vector convergence too slow
Intradomain Routing

RIP Overview

- RIP: routing information protocol [RFC 2453]
- Early intradomain routing protocol
  - derived from XNS (Xerox Network Systems) architecture
  - included in BSD Unix in 1982 as routed
  - largely replaced by L2 switching
- Distance vector routing protocol
  - distance metric: number of subnet hops
    - maximum of 15
    - limits RIP to small networks
  - distance vectors advertised every 30 sec.
  - transported over UDP
Intradomain Routing
IGRP/EIGRP Overview

• IGRP: interior gateway routing protocol
• Cisco replacement for RIP
  – initially developed before OSPF
  – more complex “distance” metrics
    • delay, bandwidth, load, reliability, hop count, MTU
  – multiple paths to destination
  – improved convergence
    • triggered updates, split horizon, holddown
  – runs directly over IP: protocol ID = 88
• EIGRP: enhanced IGRP (E ≠ exterior!)
  – CIDR route aggregation and other features
Intradomain Routing

ISIS Overview

• ISIS: intermediate system to intermediate system
  – used by many service providers
  – was available before OSPF

• OSI routing protocol
  – derived from DECnet phase V routing
  – adopted by ISO for CLNP (connectionless networking)

• Link state routing protocol
  – messages transported directly over IP
    – ISIS protocol ID = 124
  – very similar to OSPF in operation
Intradomain Routing

OSPF Link State

- **OSPF** is a link state routing protocol [RFC 2328/5340\textsubscript{v6}]
  - IETF open development process
  - messages transported directly over IP: protocol ID = 89

- Dijkstra least-cost path algorithm
  - *shortest path first* (SPF)
  - link costs
    - configured by network administrator
    - set to 1 for minimum hop routing
Intradomain Routing
OSPF Initialisation and Operation

• Initialisation
  – hello establishes link adjacencies to all neighbours
    – and maintains liveness
  – database description loads topology information

• Operation
  – link state advertisements *flooded* to AS
    • directed link weights use multiple parameters
    • link costs configured by network administrator
    • 1 for minimum hop routing
  – Dijkstra least-cost path algorithm
Intradomain Routing
OSPF Common Header

- Common header (20 B)
  - version #
  - type:
    1 = hello
    2 = database description (initialisation)
    3 = link state request
    4 = link state update
    5 = link state acknowledgement (for reliable flooding)
  - packet length (OSPF header + payload)
  - router ID of message source
  - area ID
  - checksum
  - authentication type
  - authentication

todo: packet format figure
Intradomain Routing
OSPF Link State Advertisements

- LSAs (link state advertisements)
  - reliable flooding within AS
    - ACKs returned
- LSAs flooded
  - whenever link state changes
  - periodically (at least 30 min. interval)
Intradomain Routing

OSPF LSA Header

- LSA header (after common header)
  - LS age: time since LSA originated [s]
  - options
  - type
  - link state ID
  - advertising router
  - LS sequence number
  - LS checksum
  - length
Intradomain Routing

OSPF Features

- **Security:**
  - all OSPF messages authenticated

- **Multipath routing**
  - multiple equal-cost paths supported
  - can cause problems to end-to-end protocols
    - effect on TCP packet ordering

- **Hierarchy**
  - 2-level scales to larger ASs

- **Multicast routing**
  - MOSPF: multicast OSPF [RFC 1584]
Intradomain Routing

OSPF Hierarchy

- OSPF supports 2-layer hierarchy
- AS divided into *areas* connected by the *backbone*
  - backbone area ID 0.0.0.0
- LSAs flooded only within area
  - or within backbone
- Border routers summarise costs within area
  - advertise to other border routers
- Boundary routers connect to other ASs
  - results in 3-layer Internet hierarchy:
    BGP / OSPF backbone / OSPF area
Network Routing

NR.4.1 Internet Interdomain Routing

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
  NR.4.1 Intradomain Routing: RIP and OSPF
  NR.4.2 Interdomain Routing: BGP
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
AS Routing Hierarchy
Motivation for Different Protocols

• Motivation for different inter- and intra-AS protocols
• Scalability
  – additional EGP over IGP hierarchy enhances scalability
  – but fixed 2-level hierarchy limits benefits
• Flexibility
• Policy
AS Routing Hierarchy
Motivation for Different Protocols

• Motivation for different inter- and intra-AS protocols
• Scalability
• Flexibility
  – AS boundaries are administrative reality
  – allows each AS to choose its own IGP
  – but doesn’t allow integrated hierarchy with IGPs
• Policy
AS Routing Hierarchy
Motivation for Different Protocols

- Motivation for different inter- and intra-AS protocols
- Scalability
- Flexibility
- Policy
  - inter-AS peering and transit a business decision
  - BGP policy not well designed from beginning
    - policies may never converge
    - convergence dependent on message order and link state
    - not possible to compute if policies will converge
Interdomain Routing

BGP Overview

- Border gateway protocol (BGP) [RFC 2328 / STD 0054]
- IETF interdomain routing protocol (EGP)
  - de facto standard; all ASs must use to interconnect
  - current version is BGP4
  - BGP replaced initial EGP [RFC 0904]
- BGP has many flaws
  - many papers on how BGP is broken and proposals to fix
  - but BGP is part of the (interdomain) Internet hourglass
    - as hard to evolve or replace as IP
Interdomain Routing
Current Internet Structure

tier-2 providers

IXP

Access lines

Tier-1 providers

peering points

IXP

Local ISPs
Interdomain Routing

BGP Path Vector

- *Path vector* routing protocol
- Some similarity to distance vector
  - paths advertised instead of distance metrics
  - sequence of AS numbers
    - assigned by RIPE, ARIN, etc.
- Propagates *reachability* information
  - ASs advertise their existence to Internet
  - policy determines which choices an AS makes
- Messages transported over TCP
Interdomain Routing

BGP Sessions

- BGP **session** is association between **peers**
  - **eBGP**: exterior BGP session between ASs
  - **iBGP**: interior BGP session within AS
    - fully connected mesh
    - **multihop** TCP if direct link doesn’t exist *(app layer protocol?)*
    - allows route advertisements to reach every node

[Kurose–Ross p.380]
Interdomain Routing
BGP Route Advertisements

- Route *advertisements*
  - promise to forward to destination
  - IP prefixes are aggregated
Interdomain Routing
BGP Reachability Example

- BGP advertisement example
  - propagation of AS3:3A reachability into AS2
Interdomain Routing
BGP Reachability Example

- BGP advertisement example
  - propagation of AS3:3A reachability into AS2
- AS3 prefix reachability to AS1 using eBGP 3A→1C
Interdomain Routing
BGP Reachability Example

- BGP advertisement example
  - propagation of AS3:3A reachability into AS2
- AS3 prefix reachability to AS1 using eBGP 3A→1C
- 1C distributes new prefix route using iBGP in AS1
Interdomain Routing
BGP Reachability Example

- BGP advertisement example
  - propagation of AS3:3A reachability into AS2
- AS3 prefix reachability to AS1 using eBGP 3A→1C
- 1C distributes new prefix route using iBGP in AS1
- 1B re-advertise new prefix to AS2 using eBGP 1B→2A
Interdomain Routing
BGP Reachability Example

- BGP advertisement example
  - propagation of AS3:3A reachability into AS2
- AS3 prefix reachability to AS1 using eBGP 3A→1C
- 1C distributes new prefix route using iBGP in AS1
- 1B re-advertise new prefix to AS2 using eBGP 1B→2A
- New prefix received creates fwd table entry to AS3
Intradomain Routing
Routes and Path Attributes

- BGP prefix advertisements include BGP attributes
  - prefix + attributes = route
- Important attributes:
  - AS-PATH
    - AS numbers through which advertisement passed
  - NEXT-HOP
    - specific internal-AS router to next-hop AS
    - may be multiple links from current AS to next-hop-AS
- When edge router receives route advertisement
  - use import policy to accept/decline
Intradomain Routing

BGP Route Selection

• Router may learn about multiple routes to a prefix

• Elimination rules:
  – local preference value attribute
    • policy decision
  – shortest AS-PATH
  – closest NEXT-HOP router
    • hot potato routing
  – additional policy criteria
Intradomain Routing

BGP Messages

- BGP messages exchanged over TCP
  - OPEN
    - opens TCP connection to peer & authenticates sender
  - UPDATE
    - advertises new path (or withdraws old)
  - KEEPALIVE
    - keeps connection alive in absence of UPDATES
    - also acknowledges OPEN request
  - NOTIFICATION
    - reports errors in previous message
    - also used to close connection

todo: packet format figure
Intradomain Routing

BGP Example

• Topology
  – provider networks AS 1, 2, 3
    • e.g. tier-1 or tier-2
  – subscriber networks AS 70, 80, 90
    • e.g. tier-3, enterprise, campus
  – AS80 dual-homed
    why?

[Kurose–Ross p.382]
Intradomain Routing

BGP Example: Multihoming

• Topology
  − provider networks AS 1,2,3
    • e.g. tier-1 or tier-2
  − subscriber stub networks AS 70,80,90
    • e.g. tier-3, enterprise, campus
  − AS80 dual-homed
    • for resilience if one service provider fails or link goes down

[Kurose–Ross p.382]
Intradomain Routing

BGP Policy Example: Transit Traffic

• Subscriber network policy: transit traffic

is AS80 likely to want to route between AS2 and AS3?
why?
Intradomain Routing

BGP Policy Example: Transit Traffic

- Subscriber network policy: transit traffic
  - AS80 is unlikely to route transit traffic between AS2 and AS3
    - stub network only sources or sinks traffic
    - tier-3 not engineered to transit tier-2 traffic
    - no economic benefit

*How is this accomplished?*

[Kurose–Ross p.382]
Intradomain Routing

BGP Policy Example: Transit Traffic

- Subscriber network policy: transit traffic
  - AS80 is unlikely to route transit traffic between AS2 and AS3
- AS80 route advertisements
  - will not advertise 80→3→90 to AS2

[Kurose–Ross p.382]
Intradomain Routing
BGP Policy Example: Provider Policy

- Provider policy examples
  - AS1 advertises 1→70 to AS2
  - AS2 advertises 2→1→70 to AS80
  
  *should AS2 advertise 2→1→70 to AS3?*

[Kurose–Ross p.382]
Intradomain Routing
BGP Policy Example: Provider Policy

- Provider policy examples
  - AS1 advertises 1→70 to AS2
  - AS2 advertises 2→1→70 to AS80
  - AS2 not likely to advertise 2→1→70 to AS3
    - neither AS70 nor AS3 are AS2’s customers
    - AS2 wants to force AS3 to route to AS70 via AS1
    - AS2 wants to route only to and from its own customers

[Kurose–Ross p.382]
Network Routing

NR.5  Broadcast and Multicast Routing

NR.1  Functions and services
NR.2  Routing algorithms
NR.3  PSTN routing architecture and algorithms
NR.4  Internet routing architecture and protocols
NR.5  Broadcast, multicast, and anycast routing
  NR.5.1  Overview and motivation
  NR.5.2  Broadcast algorithms
  NR.5.3  Multicast algorithms
  NR.5.4  Internet multicast protocols
  NR.5.5  Reliable multicast
  NR.5.6  Anycast algorithms and protocols

NR.6  Overlay routing and DHTs
Network Routing

NR.5.1 Broadcast and Multicast Overview

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
   NR.5.1 Overview and motivation
   NR.5.2 Broadcast algorithms
   NR.5.3 Multicast algorithms
   NR.5.4 Internet multicast protocols
   NR.5.5 Reliable multicast
   NR.5.6 Anycast algorithms and protocols

NR.6 Overlay routing and DHTs
Broadcast and Multicast
Group Communication

- Multicast
  - point-to-multipoint
    single source $\rightarrow$ multiple destinations
  - reverse multicast (multipoint-to-point)
    multiple sources $\rightarrow$ single destination
  - multipoint-to-multipoint
    multiple sources $\leftrightarrow$ multiple destinations

- Broadcast
  - single source $\rightarrow$ all destinations (in a given subnetwork)
Broadcast and Multicast
Motivation and ALM

• Application-layer multicast (ALM) or broadcast
  advantages and disadvantages?
Broadcast and Multicast
Motivation and ALM

- **Application-layer multicast** (ALM) or broadcast
  - source sends multiple times: once to each destination
  - necessary if network-layer multicast not available
    - note: BGP does not support multicast
    - most ISPs do not enable multicast

*Alternative?*
Broadcast and Multicast

Network Multicast Motivation

- Application-layer multicast (ALM) or broadcast
- Network-layer multicast or broadcast
  - source sends once to single address
  - network duplicates packets as necessary
    - requires multicast algorithms and protocols
    - significant benefits of multicast switches *Lecture NL*
  - significant bandwidth conservation possible
    - but ISPs don’t do it
Network Routing

NR.5.2 Broadcast Algorithms

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
  NR.5.1 Overview and motivation
  NR.5.2 Broadcast algorithms
  NR.5.3 Multicast algorithms
  NR.5.4 Internet multicast protocols
  NR.5.5 Reliable multicast
  NR.5.6 Anycast algorithms and protocols

NR.6 Overlay routing and DHTs
Broadcast Algorithms

Overview

- Broadcast capable LAN
  - native broadcast transparent to layer 3

*What is the simplest broadcast algorithm? (assuming a non-broadcast mesh)*
Broadcast Algorithms

Flooding

- Broadcast capable LAN
  - native broadcast transparent to layer 3
- Flooding
  - every node duplicates to all destinations
    - except one from which packet received

**Problem?**
Broadcast Algorithms

Flooding

- Broadcast capable LAN
  - native broadcast transparent to layer 3
- Flooding
  - every node duplicates to all destinations
    - except one from which packet received
  - only suitable if no cycles in network graph
    - otherwise broadcast storm

*Alternative?*
Broadcast Algorithms

Controlled Flooding

• Broadcast capable LAN
  – native broadcast transparent to layer 3

• Flooding
  – every node duplicates to all destinations
    • except one from which packet received
  – only suitable if no cycles in network graph
    • otherwise broadcast storm

• **Controlled flooding**
  – only duplicate and forward if packet arrives for the 1st time
Broadcast Algorithms

Controlled Flooding: Sequence Number

• Sequence-number controlled-flooding
  – each node inserts its address & sequence number in packet
  – each node maintains list of previously forwarded packets
    *how long to keep the sequence number?*
  – forward only if not in list
  – examples:
    • OSPF and ISIS LSA flooding
    • Gnutella (application layer) broadcast
Broadcast Algorithms

Controlled Flooding: Sequence Number

- Sequence-number controlled-flooding
  - each node inserts its address & sequence number in packet
  - each node maintains list of previously forwarded packets
    - soft state: keep longer than expected packet life in network
  - forward only if not in list
- examples:
  - OSPF and ISIS LSA flooding
  - Gnutella (application layer) broadcast
Broadcast Algorithms
Controlled Flooding: RPF

- Reverse path forwarding
Broadcast Algorithms
Controlled Flooding: RPF

- Reverse path forwarding
- Each node in tree
  - find shortest path to root

[Kurose–Ross p.388]
Broadcast Algorithms

Controlled Flooding: RPF

- Reverse path forwarding
- Each node in tree
  - find shortest path to root
  - duplicate and forward only if received on shortest path
Broadcast Algorithms
Controlled Flooding: RPF

- Reverse path forwarding
- Each node in tree
  - find shortest path to root
  - duplicate and forward only if received on shortest path
  - otherwise discard

[Kurose–Ross p.388]
Broadcast Algorithms

Controlled Flooding: RPF

- Reverse path forwarding
- Each node in tree
  - find shortest path to root
  - duplicate and forward only if received on shortest path
  - otherwise discard

[Kurose–Ross p.388]
Broadcast Algorithms
Controlled Flooding: RPF

- Reverse path forwarding
- Each node in tree
  - find shortest path to root
  - duplicate and forward only if received on shortest path
  - otherwise discard
- Self-terminating
  - no cycles possible

[Kurose–Ross p.388]
Broadcast Algorithms

Spanning Tree

- Spanning tree
  - tree that contains every node in graph
  - no cycles

[Kurose–Ross p.388]
Broadcast Algorithms

Spanning Tree

• Spanning tree
  – tree that contains every node in graph
  – no cycles
• Packet forwarded
  – from any node in tree

[Kurose–Ross p.388]
Broadcast Algorithms
Spanning Tree

• Spanning tree
  – tree that contains every node in graph
  – no cycles

• Packet forwarded
  – from *any* node in tree
  – does not need to be *root*
Broadcast Algorithms
Spanning Tree Construction

- Spanning tree construction
  - main complexity of approach
- Various approaches
  - center-based
  - source-based

*examples later*
Network Routing

NR.5.3 Multicast Algorithms

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
  NR.5.1 Overview and motivation
  NR.5.2 Broadcast algorithms
  NR.5.3 Multicast algorithms
  NR.5.4 Internet multicast protocols
  NR.5.5 Reliable multicast
  NR.5.6 Anycast algorithms and protocols
NR.6 Overlay routing and DHTs
Multicast Overview

- Communication among *subset* of nodes is *multicast*
- Multicast addressing
  
  *why is an address needed?*
- Multicast routing
  - construct a *multicast tree* connecting group members
  - not all links among switches between participating nodes
- Multicast forwarding
  - forward packets to all group members on multicast tree
Multicast Overview

• Communication among *subset* of nodes is *multicast*

• Multicast addressing
  – group address for subset of nodes

• Multicast routing
  – construct a *multicast tree* connecting group members
  – not all links among switches between participating nodes

• Multicast forwarding
  – forward packets to all group members on multicast tree
Multicast Trees

Group Shared

- Group shared
  - all senders in group use same tree
  - permits multipoint-to-multipoint
  - difficult to implement and manage

- Source-based
  - each sender uses its own tree
Multicast Trees
Source Based

- **Group shared**
  - all senders in group use same tree

- **Source-based**
  - each sender uses its own tree
  - simpler to manage than shared
  - point-to-multipoint groups
Multicast Algorithms

Tree Alternatives: Source Based

- **Group shared**
  - all senders in group use same tree

- **Source-based**
  - each sender uses its own tree
  - simpler to manage than shared
  - point-to-multipoint groups
  - multipoint-to-multipoint
    - multiple groups needed
Multicast Algorithms

Steiner Tree

- **Steiner tree**
  - minimum cost spanning tree among group members

- **Construction algorithm is** \textit{NP-complete}
  - not solvable in polynomial time
  - but excellent heuristics exist

- **Not used in Internet practice**
  - computational complexity of heuristics
  - information about entire network needed
  - monolithic: rerun whenever a router needs to join/leave
    - proposed in the context of ATM \cite{Waxman}
Multicast Algorithms

Center-Based Tree

- Center-based tree:
  - single delivery tree shared by all
  - one router identified as *center* of tree (core)

- To join:
  - edge router sends unicast join-msg addressed to core
  - join-msg processed by intermediates & forwarded to center
  - join-msg either
    - hits existing tree branch for this center, or
    - arrives at center
  - join-msg path becomes new branch of tree for this router

todo: add K&R examples
Network Routing
NR.5.4 Internet Multicast Protocols

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
  NR.5.1 Overview and motivation
  NR.5.2 Broadcast algorithms
  NR.5.3 Multicast algorithms
  NR.5.4 Internet multicast protocols
  NR.5.5 Reliable multicast
  NR.5.6 Anycast algorithms and protocols

NR.6 Overlay routing and DHTs
Internet Multicast
Protocols and Algorithms

• Internet IP multicast
• Addressing
  – IGMP: internet group multicast protocol [RFC 3376]
  – does not include multicast routing algorithm
• Multicast routing algorithm and protocol
  – e.g. DVMRP, PIM, MOSPF
• AS dependent multicast
  • recall that BGP does not support multicast
  • inter-AS multicast must be tunneled
    – but rarely is
Internet Multicast
Addressing: IGMP

- IGMP: internet group multicast protocol [RFC 3376]
  - carried in IP datagram: protocol ID = 2
- *Receiver driven*
  - multicast receivers join group
- Messages
  - membership query
  - membership report (join group)
  - leave group (optional)
Internet Multicast Routing Protocols

- Internet IP multicast consists of two components:
  - **IGMP**: internet group multicast protocol [RFC 3376]
    - way for end system (host) to join multicast tree
    - does *not* include multicast routing algorithm
      *Lecture NL*
  - **Multicast routing protocols**
    - DVMRP: distance vector multicast routing protocol
    - PIM: protocol independent multicast
    - MOSPF: multicast OSPF
Internet Multicast Protocols
Multicast Tunneling

• Not all routers support IP multicast
  – no interdomain (BGP) multicast
  – allows selective deployment or enabling within AS

*How to support multicast?*
Internet Multicast Protocols
Multicast Tunneling

- Not all routers support IP multicast
  - no interdomain (BGP) multicast
  - allows selective deployment or enabling within AS
- IGMP datagram *tunneled* between multicast routers
  - multicast *overlay* virtual topology
Internet Multicast Protocols

Multicast Tunneling

- Not all routers support IP multicast
  - no interdomain (BGP) multicast
  - allows selective deployment or enabling within AS
- IGMP datagram *tunneled* between multicast routers
  - multicast *overlay* virtual topology
  - multicast tunnel source encapsulates IGMP in IP unicast datagram
Internet Multicast Protocols
Multicast Tunneling

• Not all routers support IP multicast
  – no interdomain (BGP) multicast
  – allows selective deployment or enabling within AS

• IGMP datagram *tunneled* between multicast routers
  – multicast *overlay* virtual topology
  – multicast tunnel source encapsulates IGMP in IP unicast datagram
  – IP datagram addressed to tunnel destination
  – receiving multicast router decapsulates IGMP datagram
Internet Multicast Protocols

DVMRP Overview

• DVMRP: distance vector multicast routing protocol
  [RFC 1075]
  – first Internet multicast routing protocol
    • but not first multicast! [Kadaba 1983]
  – used in MBone (Internet multicast backbone)

• Distance vector algorithm
  – to compute shortest path from source

• Source-based trees: flood and prune
  – initial datagram flooded
  – RPF tree constructed with DVMRPs own routing tables
  – pruning to eliminate unnecessary nodes
Internet Multicast Protocols

DVMRP Soft State

• Soft state
  – DVMRP router periodically forgets branches are pruned
  – 1 min. interval
  – multicast data again flows down unpruned branches
  – downstream router: reprune or else continue to receive data

• Routers can quickly regraft to tree
  – following IGMP join at leaf
Internet Multicast Protocols

PIM Overview

- **PIM**: protocol independent multicast  [RFC 2362]
  - not dependent on any particular unicast routing algorithm
  - compatible with RIP, IGRP, OSPF, ISIS, etc.

- **Dense mode**
  - multicast group members densely located
  - example: within campus or enterprise

- **Sparse mode**
  - multicast group members widely dispersed
  - most nodes do not need to multicast (to a particular group)
  - example: users scattered about the Global Internet
Internet Multicast Protocols

PIM Dense Mode

- **PIM dense mode**
  - densely located group members
- **Data-driven tree construction**
  - RPF: underlying unicast provides forwarding information
  - flood and prune: membership assumed until pruning
  - less complicated downstream flood than DVMRP
  - protocol mechanism for router to detect it is a leaf-node
- **Bandwidth and node processing assumed plentiful**
  - within AS
  - particularly within enterprise or campus
Internet Multicast Protocols

PIM Sparse Mode

- PIM sparse mode
  - widely-scattered sparsely-connected group members

- Control-driven tree construction
  - no group membership until nodes explicitly join
  - receiver-driven to center-based tree
    - rendezvous point (RP)

- Bandwidth and node processing assumed scarce
  - across entire Internet
    - although core over-provisioned, not enough for dense-mode
  - constrained access link bandwidth

todo: add PIM examples based on K&R
Network Routing

NR.5.5 Reliable Multicast

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
  NR.5.1 Overview and motivation
  NR.5.2 Broadcast algorithms
  NR.5.3 Multicast algorithms
  NR.5.4 Internet multicast protocols
  NR.5.5 Reliable multicast
  NR.5.6 Anycast algorithms and protocols

NR.6 Overlay routing and DHTs
Reliable Multicast
Overview

- Internet multicast is not reliable
  - IP datagrams used by IGMP are unreliable
- Some applications require *reliable multicast*
  - e.g. multicast of software distributions

*Problems and issues?*
Reliable Multicast
Overview

• Internet multicast is not reliable
  – IP datagrams used by IGMP are unreliable

• Some applications require **reliable multicast**
  – e.g. multicast of software distributions

• Problems
  – reliability must be provided end-to-end (E2E arguments)
  – network support needed to get benefits of network multicast
Reliable Multicast

Alternatives

- Transport layer reliable multicast
  - end-to-end reliability
  - no benefits of network traffic aggregation
    - unless interacts directly with network layer multicast trees
Reliable Multicast

ACK Implosion

- ACK implosion problem
  - reverse of multicast bandwidth problem
  - even when routers split forward multicast

- Alternative:
  network-layer reliable multicast
  - multicast nodes manage ACKs or NAKs
    - ACK aggregation complexity
    - NAK timer management

- Lots of research has been done
  - but no consensus on solutions
Network Routing

NR.5.6 Anycast Algorithms and Protocols

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
  NR.5.1 Overview and motivation
  NR.5.2 Broadcast algorithms
  NR.5.3 Multicast algorithms
  NR.5.4 Internet multicast protocols
  NR.5.5 Reliable multicast
  NR.5.6 Anycast algorithms and protocols

NR.6 Overlay routing and DHTs
Anycast
Overview

• Communication to *any* receiver within anycast group
  – *k*-cast when destined to *k* nodes in group
  – anycast is *k*=1

• Anycast addressing
  – group address for anycast receiver nodes

• Internet anycast  [RFC 1546, 4786, 7094]
  – conceptual model
  – does not solve routing and hard problems

• Increasingly used by DNS
Network Routing

NR.6 Overlay Routing and DHTs

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
  NR.6.1 Motivation and application for overlay routing
  NR.6.2 Distributed hash tables and chord routing
Overlay Routing and DHTs

NR.6.1 Motivation and Application

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
   NR.6.1 Motivation and application for overlay routing
   NR.6.2 Distributed hash tables and chord routing
Overlay Networks
Motivating Example

• Applications frequently need unique network services
  – not supported by IP
  – not offered or allowed by ISP

• Examples
  – ALM or reliable multicast
  – other addressing schemes

Solution?
Overlay Networks

Motivating Example

• Applications frequently need unique network services
  – not supported by widely deployed protocols such as IP
  – Internet hourglass waist restricts functionality to
    • IP addressing
    • BGP interdomain routing

• Examples
  – VLANs (virtual local area networks)
  – encrypted tunnels
  – applications needing other addressing/routing/forwarding
  – reliable multicast
Overlay Networks

Definition

• Overlay
  – any instance of a protocol P over protocol Q
  – for which layer(P) $\neq$ layer(Q)

• Examples:
  – IP (L3) over ATM (L3)
  – IPv6 (L3) over IPv4 (L3)
  – P2P (L7 over L3) over TCP (L4) over IP (L3)
Overlay Networks

Motivating Example

- End systems on Internet
Overlay Networks
Motivating Example

- End systems on Internet
- Information forwarded in circular order
Overlay Networks
Motivating Example

- End systems on Internet
- Information forwarded in circular order
  - but not supported by IP

*solution*?
Overlay Networks
Motivating Example

- End systems on Internet
- Information forwarded in circular order
- Build circular overlay
  - over TCP connections between ES\(_i\) and ES\(_{i+1}\)
  - probably multiple IP hops
Overlay Networks
Motivating Example

- End systems on Internet
- Information forwarded in circular order
- Build circular overlay
  - over TCP connections between ES\(_i\) and ES\(_{i+1}\)
  - ESs perform L3 functions:
    - address
    - route
    - forward
  - as layer 7 over layer 4 (TCP)
Overlay Routing and DHTs

NR.6.2 Distributed Hash Tables and Chord

NR.1 Functions and services
NR.2 Routing algorithms
NR.3 PSTN routing architecture and algorithms
NR.4 Internet routing architecture and protocols
NR.5 Broadcast, multicast, and anycast routing
NR.6 Overlay routing and DHTs
   NR.6.1 Motivation and application for overlay routing
   NR.6.2 Distributed hash tables and chord routing
Peer-to-Peer File Sharing
Improving Growth Complexity

• Question: can we do better?
  – $O(n)$ node state of centralised and hybrid
  – poor resilience of centralised
  – $O(hn^2)$ comm. overhead of distributed flooding

*What is the goal?*
Peer-to-Peer File Sharing
Improving Growth Complexity

- **Question:** can we do better?
  - $O(n)$ node state of centralised and hybrid
  - poor resilience of centralised
  - $O(hn^2)$ comm. overhead of distributed flooding

- **Goal:** locate information by *content*
  - avoid flooding for search query

- **CAMs (content addressable memories)**
  - return value in 1 cycle by simultaneous matching all entries
  - but how to distribute?
Peer-to-Peer File Sharing

Background: Hashing

- Hashing is a mapping to a fixed-size string
  - hash function $H(x) = a$
  - important property:
    $x \neq y \Rightarrow H(x) \neq H(y)$ with high probability
    - otherwise a hash collision must be resolved

- Examples:
  - CRC (cyclic redundancy check)
  - message digest, e.g. MD5
  - hash table for content-based addressing

Can we exploit this for P2P?
Structured P2P File Sharing
DHT Overview

- Traditional hash tables index by content *locally*
- Observation:
  - divide hash-space among distributed systems
  - DHTs: *distributed hash tables* [Stoica 2001]
- Mechanisms
  - self-organisation of P2P overlay
    - node arrival, departure, failure
  - routing of DHT lookups
Circular DHT Addressing

- Nodes arranged logical circle
  - in increasing hash value $H$
  - modulo $n$

- Nodes link to successor
  - may be virtual link
    - e.g. IP address
  - sparse ring with gaps allows new node insertion
Circular DHT
Naïve Circular Routing

- Queries **travel ring**
  - each node either:
    - serves query
    - or
    - passes to successor
Circular DHT
Naïve Circular Forwarding Example

- Queries travel ring
  - each node either:
    - serves query
    - passes to successor

- Example:
  - node 3 wants $H=11$
Circular DHT
Naïve Circular Forwarding Example

• Queries travel ring
  - each node either:
    • serves query
    • passes to successor

• Example: 3 wants $H=11$
  1. $11 > \text{succ}(4)$
Circular DHT
Naïve Circular Forwarding Example

- Queries travel ring
  - each node either:
    - serves query
    - passes to successor
- Example: 3 wants $H=11$
  1. $11 > \text{succ}(4)$
  2. $11 > \text{succ}(5)$
Circular DHT
Naïve Circular Forwarding Example

• Queries travel ring
  – each node either:
    • serves query
    • passes to successor

• Example: 3 wants $H=11$
  1. $11 > \text{succ}(4)$
  2. $11 > \text{succ}(5)$
  3. $11 > \text{succ}(8)$
Circular DHT
Naïve Circular Forwarding Example

- Queries travel ring
  - each node either:
    - serves query
    - passes to successor

- Example: 3 wants $H=11$
  1. $11 > \text{succ}(4)$
  2. $11 > \text{succ}(5)$
  3. $11 > \text{succ}(8)$
  4. $11 \leq \text{succ}(10)$
Circular DHT
Naïve Circular Forwarding Example

- Queries travel ring
  - each node either:
    - serves query
    - passes to successor

- Example: 3 wants $H=11$
  1. $11 > \text{succ}(4)$
  2. $11 > \text{succ}(5)$
  3. $11 > \text{succ}(8)$
  4. $11 \leq \text{succ}(10)$
  5. return query(11) to 3
Circular DHT
Circular Forwarding Complexity

Number of query hops?
Circular DHT
Circular Forwarding Complexity

- # of hops for $N$ nodes:
  - max = $N - 1$
  - average = $N/2$
  - $O(N)$

*problem?
Circular DHT
Circular Forwarding Complexity

- # of hops for $N$ nodes:
  - max = $N - 1$
  - average = $N/2$
  - $O(N)$

- poor performance for large $N$
- single *end system* failure breaks ring

*improvement?*
Circular DHT
Chord Routing

• *Chords*: shortcuts
  – 2-hop shortcuts
    halve search time
    • average = \( N/4 \)
    • cost: add chord pointer
Circular DHT
Chord Routing

- Chords: shortcuts
  - 2-hop shortcuts
    - halve search time
      - average = \( N/4 \)
      - cost: add chord pointer
  - 4-hop shortcuts
Circular DHT
Chord Routing

- Chords: shortcuts
  - 2-hop shortcuts
    halve search time
    - average = \( N/4 \)
    - cost: add chord pointer
  - 4-hop shortcuts
    \textit{optimal solution?}
Circular DHT
Chord Routing

- Chords: shortcuts
  - optimal solution
    - multiple shortcuts of $2^i$ to half ring circumference

Note: generally based on node id rather than actual hop count shown in example diagram
Circular DHT
Chord Routing

• Chords: shortcuts
  - optimal solution
    • multiple shortcuts of $2^i$ to half ring circumference
  - finger table
    • $\log(N)$ entries
    • entry $i$ points to successor($n + 2^i$)
Circular DHT
Chord Routing and Forwarding

- Chord *routing*?
  -
  -

- Chord *forwarding*?
  -
  -
Circular DHT
Chord Routing and Forwarding

- **Chord routing**
  - *builds and maintains* finger table
  - *updates* finger table on node insertion or deletion from ring

- **Chord forwarding**
  - index *lookup* in finger table
  - forward to next hop or serve locally
Circular DHT
Chord Forwarding Example

- Queries travel ring using chord shortcuts
  - each node either:
    - serves query
    - passes to successor

- Example:
  - node 3 wants $H=1$
Circular DHT
Chord Forwarding Example

- Example: 3 wants $H=1$
  
1. $\text{lookup}(1) \Rightarrow i=2, \text{succ}=10$

[Diagram showing the Chord network with nodes labeled 0 to 15 and arrows indicating connections and successor nodes.]

$[\text{succ}(2)=10] \leq \text{mod } 1 < \text{mod } \text{nil}$

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\text{succ}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>04</td>
</tr>
<tr>
<td>1</td>
<td>05</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
</tbody>
</table>
Circular DHT
Chord Forwarding Example

• Example: 3 wants $H=1$
  1. lookup(1) $\Rightarrow i=2$, succ=10

\[ [\text{succ}(2)=10] \leq_{\text{mod}} 1 <_{\text{mod}} \text{nil} \]

<table>
<thead>
<tr>
<th>$i$</th>
<th>succ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>04</td>
</tr>
<tr>
<td>1</td>
<td>05</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
</tbody>
</table>
Circular DHT
Chord Forwarding Example

- Example: 3 wants $H=1$
  1. lookup(1) ⇒ $i=2$, succ=10
  2. lookup(1) ⇒ $i=1$, succ=15

$[\text{succ}(1)=15] \leq \text{mod} 1 < \text{mod} [\text{succ}(2)=03]$
Circular DHT
Chord Forwarding Example

• Example: 3 wants $H=1$
  1. lookup(1) $\Rightarrow i=2$, succ=10
  2. lookup(1) $\Rightarrow i=1$, succ=15

\[ \text{succ}(1) = 15 \leq \text{mod} 1 < \text{mod} \ [\text{succ}(2) = 03]\]
Circular DHT
Chord Forwarding Example

• Example: 3 wants $H=1$
  1. lookup(1) $\Rightarrow i=2$, $\text{succ}=10$
  2. lookup(1) $\Rightarrow i=1$, $\text{succ}=15$
  3. lookup(1) $\Rightarrow i=0$, $\text{succ}=01$

\[
\begin{array}{c|c}
  i & \text{succ} \\
  \hline
  0 & 01 \\
  1 & 03 \\
  2 & 05 \\
\end{array}
\]

\[\text{successors}(0)=01 \leq \mod 1 < \mod \text{successors}(1)=03\]
Circular DHT
Chord Forwarding Example

- Example: 3 wants $H=1$
  1. lookup(1) $\Rightarrow i=2$, succ=10
  2. lookup(1) $\Rightarrow i=1$, succ=15
  3. lookup(1) $\Rightarrow i=0$, succ=01

$succ(0)=01 \leq_{\text{mod } 1} 0 <_{\text{mod } 1} succ(1)=03$

<table>
<thead>
<tr>
<th>$i$</th>
<th>succ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01</td>
</tr>
<tr>
<td>1</td>
<td>03</td>
</tr>
<tr>
<td>2</td>
<td>05</td>
</tr>
</tbody>
</table>
Circular DHT
Chord Forwarding Example

- Example: 3 wants $H=1$
  1. lookup(1) $\Rightarrow i=2, \text{succ}=10$
  2. lookup(1) $\Rightarrow i=1, \text{succ}=15$
  3. lookup(1) $\Rightarrow i=0, \text{succ}=01$
  4. lookup(1) $\Rightarrow$ match
P2P File Sharing
Structured vs. Unstructured

- **Unstructured file sharing**
  - schemes without underlying address structure
    - centralised e.g. napster
    - decentralised e.g. Gnutella 0.4
    - hybrid e.g. Gnutella 0.6

- **Structured file sharing**
  - schemes that use address structure
    - DHTs e.g. Chord, Pastry, Tapestry
Unstructured P2P File Sharing

Growth Complexity

Scaling complexity measured as number of nodes $n$
Structured P2P File Sharing
Growth Complexity

- Scaling complexity measured as number of nodes $n$
  - per node state: $\log n$
  - communication overhead: $\log n$
DHT P2P
Network Layer Issues

Are servants IS or ES?
DHT P2P
Network Layer Issues

- Nodes serve *dual roles*
  - end system (servent)
    - *and*
  - intermediate system (routing and forwarding)
DHT P2P
Network Layer Issues

• Nodes serve *dual roles*
  – end system (servent)
    *and*
  – intermediate system (routing and forwarding)

• DHT generally deployed as overlay
  *why?*
DHT P2P
Network Layer Issues

- Nodes serve *dual roles*
  - end system (servent)  
  - *and*
  - intermediate system (routing and forwarding)
- DHT generally deployed as overlay
  - P2P servents are Internet-connected hosts

```
+---+   +---+   +---+
| L4 | TCP| L3 | IP |
|    |    |    |    |
| L2 | link|
```
DHT P2P
Network Layer Issues

- Nodes serve *dual roles*
  - end system (servent)
  - intermediate system (routing and forwarding)
- DHT generally deployed as overlay
  - queries use DHT overlay on TCP/IP
  - responses bypass overlay *how?*

<table>
<thead>
<tr>
<th>Layer (L)</th>
<th>Protocol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>L7</td>
<td>P2P</td>
<td>file sharing</td>
</tr>
<tr>
<td>L3</td>
<td>DHT</td>
<td>address /route / forward</td>
</tr>
<tr>
<td>L4</td>
<td>TCP</td>
<td>E2E servent–servent</td>
</tr>
<tr>
<td>L3</td>
<td>IP</td>
<td>servent–servent path</td>
</tr>
<tr>
<td>L2</td>
<td>link</td>
<td>HBH servent–servent</td>
</tr>
</tbody>
</table>
DHT P2P
Network Layer Issues

• Nodes serve *dual roles*
  – end system (servent) 
  – intermediate system (routing and forwarding) 

• DHT generally deployed as overlay 
  – queries use DHT overlay on TCP/IP 
  – responses bypass overlay 
    • directly to originating IP address 

<table>
<thead>
<tr>
<th>Layer (L)</th>
<th>Protocol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>L7</td>
<td>P2P</td>
<td>file sharing</td>
</tr>
<tr>
<td>L3</td>
<td>DHT</td>
<td>address /route / forward</td>
</tr>
<tr>
<td>L4</td>
<td>TCP</td>
<td>E2E servent–servent</td>
</tr>
<tr>
<td>L3</td>
<td>IP</td>
<td>servent–servent path</td>
</tr>
<tr>
<td>L2</td>
<td>link</td>
<td>HBH servent–servent</td>
</tr>
</tbody>
</table>
Routing
Additional References


Routing

Additional References

Communication Networks

Acknowledgements

Some material in these foils comes from the textbook supplementary materials:

  http://wps.aw.com/aw_kurose_network_5

- Sterbenz & Touch, *High-Speed Networking: A Systematic Approach to High-Bandwidth Low-Latency Communication* 
  http://hsn-book.sterbenz.org