Goals

- Provides the services to the transport layer
- Should be independent of the subnet design
- The transport layer should be shielded from the number, type and topology of the subnet
- A uniform numbering plan for the network addresses made available to the transport layer
Store-and-Forward Packet Switching
Implementation of Connectionless Service
Implementation of Connection-Oriented Service
<table>
<thead>
<tr>
<th>Issue</th>
<th>Datagram subnet</th>
<th>Virtual-circuit subnet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit setup</td>
<td>Not needed</td>
<td>Required</td>
</tr>
<tr>
<td>Addressing</td>
<td>Each packet contains the full source and destination address</td>
<td>Each packet contains a short VC number</td>
</tr>
<tr>
<td>State information</td>
<td>Routers do not hold state information about connections</td>
<td>Each VC requires router table space per connection</td>
</tr>
<tr>
<td>Routing</td>
<td>Each packet is routed independently</td>
<td>Route chosen when VC is set up; all packets follow it</td>
</tr>
<tr>
<td>Effect of router failures</td>
<td>None, except for packets lost during the crash</td>
<td>All VCs that passed through the failed router are terminated</td>
</tr>
<tr>
<td>Quality of service</td>
<td>Difficult</td>
<td>Easy if enough resources can be allocated in advance for each VC</td>
</tr>
<tr>
<td>Congestion control</td>
<td>Difficult</td>
<td>Easy if enough resources can be allocated in advance for each VC</td>
</tr>
</tbody>
</table>
Routing Algorithms

- Software responsible for deciding which output line an incoming packet should be transmitted on
  - Decision is made for the arrival of every datagram packet
- Desired properties:
  - Correctness, simplicity, robustness, stability, fairness and optimality
Routing Algorithms (cont.)

- Non-adaptive algorithms
  - Routes are computed in advance, and off-line
  - The routing tables are loaded into routers

- Adaptive algorithms
  - Change their routing decisions on reflect the changes in the current traffic and topology
  - Still a variety of different algorithms (e.g., WHEN and HOW to change)
Non-adaptive Algorithms

- Shortest Path Routing
  - Based on a well-known Dijkstra algorithm
  - Example
  - Pseudo program

- Flooding
  - Every incoming packet is sent out on every outgoing line (except the one it arrived on)
  - A hop counter in the header to end the process
Shortest Path Routing

(a) Shortest path routing through graph with vertices A, B, C, D, E, F, G, H, and edges with weights.

(b) Routing path from A to C via B and E.

(c) Additional nodes G and H added with weights.

(d) Further nodes added with weights.

(e) Additional nodes G and H with weights.

(f) Final routing path with added nodes.
Each router must do the following:

- Discover its neighbors, learn their network address.
- Measure the delay or cost to each of its neighbors.
- Construct a packet telling all it has just learned.
- Send this packet to all other routers.
- Compute the shortest path to every other router.
Distance Vector Routing

(a) Network diagram with nodes A through L and connections between them.

(b) Routing table for J:

- JA delay is 8
- JI delay is 10
- JH delay is 12
- JK delay is 6

New routing table for J:

- A: 8
- B: 20
- C: 28
- D: 17
- E: 30
- F: 18
- G: 12
- H: 10
- I: 0
- J: 0
- K: 6
- L: 15

New estimated delay from J:

- A: 28
- B: 36
- C: 22
- D: 24
- E: 31
- F: 19
- G: 31
- H: 19
- I: 40
- J: 0
- K: 9

Vectors received from J's four neighbors:

- JA delay is 8
- JI delay is 10
- JH delay is 12
- JK delay is 6
Distance Vector Routing (2)

The count-to-infinity problem.
Hierarchical Routing

(a) Hierarchical routing diagram showing regions and nodes.

(b) Full table for 1A:

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1B</td>
<td>1B</td>
<td>1</td>
</tr>
<tr>
<td>1C</td>
<td>1C</td>
<td>1</td>
</tr>
<tr>
<td>2A</td>
<td>1B</td>
<td>2</td>
</tr>
<tr>
<td>2B</td>
<td>1B</td>
<td>3</td>
</tr>
<tr>
<td>2C</td>
<td>1B</td>
<td>3</td>
</tr>
<tr>
<td>2D</td>
<td>1B</td>
<td>4</td>
</tr>
<tr>
<td>3A</td>
<td>1C</td>
<td>3</td>
</tr>
<tr>
<td>3B</td>
<td>1C</td>
<td>2</td>
</tr>
<tr>
<td>4A</td>
<td>1C</td>
<td>3</td>
</tr>
<tr>
<td>4B</td>
<td>1C</td>
<td>4</td>
</tr>
<tr>
<td>4C</td>
<td>1C</td>
<td>4</td>
</tr>
<tr>
<td>5A</td>
<td>1C</td>
<td>4</td>
</tr>
<tr>
<td>5B</td>
<td>1C</td>
<td>5</td>
</tr>
<tr>
<td>5C</td>
<td>1B</td>
<td>5</td>
</tr>
<tr>
<td>5D</td>
<td>1C</td>
<td>6</td>
</tr>
<tr>
<td>5E</td>
<td>1C</td>
<td>5</td>
</tr>
</tbody>
</table>

(c) Hierarchical table for 1A:

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Line</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1B</td>
<td>1B</td>
<td>1</td>
</tr>
<tr>
<td>1C</td>
<td>1C</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1B</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>1C</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1C</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>1C</td>
<td>4</td>
</tr>
</tbody>
</table>
(a) A network.  (b) A spanning tree for the leftmost router.  (c) A multicast tree for group 1.  (d) A multicast tree for group 2.
Routing for Mobile Hosts
Routing for Mobile Hosts (2)

Packet routing for mobile users.

1. Packet is sent to the mobile host's home address
2. Packet is tunneled to the foreign agent
3. Sender is given foreign agent's address
4. Subsequent packets are tunneled to the foreign agent
Routing in Ad Hoc Networks

• Military vehicles on battlefield.
  – No infrastructure.

• A fleet of ships at sea.
  – All moving all the time

• Emergency works at earthquake.
  – The infrastructure destroyed.

• A gathering of people with notebook computers.
  – In an area lacking 802.11.