Datagram vs. Virtual Circuit

In a datagram network

- Two packets of the same user pair can travel along different routes.
- The packets can arrive out of sequence.
- Packets contain full Src, Dst addresses.
- Each host occupies routing table entries.
- Requires no connection setup.

In a virtual circuit network

- All packets of the same virtual circuit travel along the same path.
- Packet sequencing is guaranteed.
- Packets contain short VC Id. (VCI).
- Each VC occupies routing table entries.
- Requires VC setup. First packet has large delay.
### Virtual Circuit and Datagram Implementation

<table>
<thead>
<tr>
<th>External Service</th>
<th>Datagram</th>
<th>Virtual Circuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datagram</td>
<td>UDP over IP (packet)</td>
<td>IP over ATM</td>
</tr>
<tr>
<td>Virtual Circuit</td>
<td>TCP over IP (message, packet)</td>
<td>TYMNET, SNA over ATM (Virtual and explicit route)</td>
</tr>
</tbody>
</table>
Routing Algorithm

- Select routes for various origin-destination (OD) pairs via shortest or optimization calculation (accommodate more OD pairs).
- Delivery of messages to the correct destination once routes are selected.
  → Use routing tables.

Performance Measures Affected By Routing

**Delay**

- All links have $C=10$ pkts/s.
- If all traffic is via (4,6), congestion occurs.
- Via (1,3,6) and (2,5,6), the delay is small.

**Throughput**

- Traffic can be accommodated by multi-path routing.
- What is the max. throughput from nodes 1 and 2 to 6?
- How about the worst case?
Classification of Routing Algorithms

- Centralized (all routing decisions at a single node) or Distributed (computation of routes shared by nodes)
- Static (routes are fixed for each OD pair regardless of traffic pattern) or Adaptive (responsive to traffic pattern)

Desirable Properties of Routing Algorithms

- correctness
- simplicity
- robustness
- stability
- fairness
- optimality

conflicts
(Routing) Optimal Principle

- If router J is on the optimal path from route I to route K, then the optimal path from J to K also falls along the same route.
- The optimal routes to a router form a sink tree.
Dijkstra’s Shortest Path Algorithm [DIJK59]

Find the shortest paths form a given source node to all other nodes by developing paths in order of increasing path length.

Let the set of nodes in a network be \( N \).

1. start with a source node \( S \) in node set \( M \). Let the nodes not in \( M \) be \( M' \).
2. Let \( L \) with the set of links connecting \( M \) and \( M' \).
   - Among the nodes in \( M' \) connected to \( M \) via \( L \), find the node, \( n \), with the lowest path cost to \( S \). Move \( n \) to \( M \). (Use hopcount, nodeId for tie breaker.)
   - Update the cost from \( S \) to other nodes in \( M' \) taking into consideration of the new path via \( n \).
3. Repeat step 2 until \( M = N \).
Computing Shortest Path

(a)

(b)

(c)

(d)

(e)

(f)
Routing Strategies

- **Fixed routing**
  Network Operation Center (NOC) collects information from individual nodes.
  NOC carries out the least cost routing algorithms.
  NOC distributed the routing information to individual nodes.
  The above steps are carried out periodically.

- **Flooding**
  A node sends/relays a message along all its outgoing links.
  Rely on hop counts or time-stamps to terminate the flooding.
  Disadvantage: a lot of redundant msgs, waste bandwidth.
  Advantage: does not require NOC, reliable,
  message may arrive via a minimum hop route.
Routing in ARPANET (old version)

Use distributed, shortest path, and adaptive routing algorithm. Periodically, each IMP calculates delay to other IMPs, exchanges delay vectors with their neighbors, based on the delay vectors received, compute the new routing table.

Problems: 1) low throughput, 2) susceptible to oscillations, 3) good news spread fast; bad news spread slow.
Exercise on ARPANET Routing

Consider the mesh network shown below. Assume that the ARPA.net routing algorithm is used. Node B receives three routing vectors from A, E, and C. With the above link delay to A, E, and C, calculate B’s new routing table and fill your result into the above vacant entries.
ARPANET Routing 1979

The old ARPANET routing algorithm has the following shortcomings:
- only consider queue length, not favor high speed lines.
- it responded slowly to congestion and delay increases.

The new ARPANET routing algorithm
- measure the delay directly using time-stamp.
- every 10 seconds, delay on each link is calculated and flood to all other nodes (not just the neighboring node).
- each node compute the new routing information based on new delay info and Dijkstra algorithm.
- it is more responsive.
- but new problems appear (Oscillations are possible and very damaging)
Count to Infinity Problem

(a) Initially
   1
   1 2
   1 2 3
   1 2 3 4

   After 1 exchange
   3 2 3 4
   3 4 3 4
   5 4 5 4
   5 6 5 6

   After 2 exchanges
   3
   3 4
   5 4
   5

   After 3 exchanges
   7 6
   6 7
   7

   After 4 exchanges
   7 8
   7

(b) Initially
   1 2 3 4
   3 4 3 4
   5 4 5 4
   4

   After 1 exchange
   1 2
   3 4
   4

   After 2 exchanges
Good news travels quickly; bad news travels slowly
Split Horizon Update

**Routing Update**

**Round 1**

**Round 2**
Problem can’t be solved by Split Horizon
Link State Routing

(a)

(b)

<table>
<thead>
<tr>
<th>Link</th>
<th>State</th>
<th>Packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Seq.</td>
<td>Age</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>D</td>
<td>7</td>
</tr>
<tr>
<td>F</td>
<td>E</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Seq.</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
</tr>
</tbody>
</table>
Oscillation in new ARPANET Routing

Oscillations are possible and very damaging.
Oscillation in ARPANET Routing
Damping the Oscillation in ARPANET Routing

Oscillations can be damped by using
1) A large bias factor (a constant that make link lengths large at zero flow).
   Side-effect: makes it less sensitive to congestion.
2) Average of several routing updates.
3) Asynchronous execution of the routing algorithm.
Traffic control

Deal with the control of the number of packets entering the network.

- Flow control: enable the receiver to control the msg receiving rate. e.g., use slide-window protocol.
- Congestion control: maintaining the number of packets within the network below the level where the network throughput starts to decrease.
  1. congested node sends choke packets to the sources.
  2. rely routing algorithms (passive)
  3. use the probing message between end points.
  4. piggyback the congestion information back to the source.
- Deadlock avoidance.
  - use structured buffer pool
  - use setup packet to reserve enough buffer and use ack msg to release buffer.
Interaction of Routing and Flow Control

As good routing keeps delay low, flow control allows more traffic into the network.

Routing determines the delay/throughput curve along which flow control operates.