Routing Example

- **Greedy**: Always use minimal path, equidistant choose randomly

  ![Diagram of a network with nodes 0 to 7, one path from 0 to 7 highlighted.](image)

  1. \[ 0 \rightarrow 3 \]
  2. \[ 0 \rightarrow 5 \]

- **Uniform Random**: Randomly pick direction

  ![Diagram with a network and a note.](image)

  - For 0 → 3
    - 50% take 4 hops
    - 50% take 6 hops

- **Weighted Random**: Randomly pick direction but weight probability by distance

  ![Diagram with a network and a note.](image)

  - For 0 → 3
    - 5/8 chance take 3 hop path
    - 3/8 chance take 5 hop path

  Probability of taking short path is

  \[
  N = \frac{H_{\text{min}} + 1}{N}
  \]

- **Adaptive**: Look at queues in either direction.
  Seed in direction if queue that has most free entries.

  *Do not change direction after initial choice.*
Goal is to evaluate two traffic patterns running on ring network with each of four routing algo.

<table>
<thead>
<tr>
<th>Traffic Pattern</th>
<th>( T_{0\text{om}} )</th>
<th>( T_{\Phi} )</th>
<th>( T_{0\text{om}} )</th>
<th>( T_{\Phi} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greedy</td>
<td>0.33</td>
<td>4</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Uniform</td>
<td>0.57</td>
<td>4.5</td>
<td>0.76</td>
<td>3.12</td>
</tr>
<tr>
<td>WRand</td>
<td>0.53</td>
<td>4.75</td>
<td>0.76</td>
<td>3.12</td>
</tr>
<tr>
<td>Adaptive</td>
<td>0.53</td>
<td>4</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>
Tornado on Ring

Node i sends to \( (i + (N - 1) \mod N) \) or \( (i + 1 \mod N) \)

Greedy Alg

Clockwise channels NOT used at ALL!
Poor load balancing

\( T_0 = 4 \)

Random Alg

Still 3 paths cross this channel but each path only carries 0.5 units of traffic

\( N_{\text{max, cw}} = 1.5 \)

What about clockwise channels though?

5 paths cross each CW channel each is 0.5 unit of traffic

\( N_{\text{max, cw}} = 2.5 > N_{\text{max, ccw}} = 1.5 \)

Theta_max = 0.4

Now the clockwise channels are the bottleneck!

\( T_D = 0.5 \times 4 + 0.5 \times 6 = 5 \)
Weigted Round Algo

Same number of paths cross CW and CCW directions as before except now with different amounts of traffic per path.

3 paths each with 5/8 traffic

\[ n_{\text{max,ccw}} = 3 \cdot \frac{5}{8} = 1.875 \]

5 paths each with 3/8 traffic

\[ n_{\text{max,cw}} = 5 \cdot \frac{3}{8} = 1.875 \]

\[ n_{\text{max,cw}} = n_{\text{max,ccw}} \quad \text{BALANCED!} \]

\[ q_{\text{rem}} = \frac{1}{1.875} = 0.53 \]

\[ T_0 = \frac{5}{8} \times 4 + \frac{3}{8} \times 6 = 2.5 + 2.25 = 4.75 \]

Adaptive

Minimal latency at light load
Max throughput at high load
Activity: Vaiait's Algorithm

Consider a 2-ary 2-hy u destination-tag routing with the given permuted traffic pattern.

\[ y_{\text{max}} = 2 \]
\[ \Theta_{\text{rem}} = 0.56_e \]
\[ T_{\Phi} = 2 \text{ route hops} \]

Calculate the ideal terminal throughput and the zero load latency in route hops if we use Vaiait's routing algorithm.

**Vaiait's Algorithm**

First, routes to a random destination and then routes to the desired destination.

**Example route for**

src 0 to dest 1

* How would Vaiait's do on a node ring?

* Try on your own similar activity for 2-ary 3-fly
Routing Taxonomy

- Obvious
  - Deterministic
  - Non-Deterministic
  - Adaptive

<table>
<thead>
<tr>
<th></th>
<th>Deterministic</th>
<th>Non-Deterministic</th>
<th>Adaptive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Destination Tag Routing</td>
<td>Valuants Alg.</td>
<td>Adapively Choose Group Dest. Tag + Valuants</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Valuants Alg.</td>
<td>Randomly Choose MID Router</td>
<td>Adapively Choose MID Router</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minimal Adaptive Routing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dim. Outs. Routing Valuants</td>
<td>Romm</td>
</tr>
</tbody>
</table>
DEADLOCK

Cyclic dependency in "waits for" and "hogs" relation

Assume forward
i -> i+2

Resources: cw q1, q2, q3

ACTORS
D -> wait
A -> wait
B -> wait
C -> wait

- Deadlock avoidance vs Deadlock Detection/Recovery

Contrast to Livelock
Cuous Routing Algorithm