Routing Example

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

  \[ 0 \rightarrow 3 \]

  \[ 1 \rightarrow 0 \rightarrow 3 \]

- **Uniform Random**: Randomly pick direction

  for \( 0 \rightarrow 3 \)

  \[ 50\% \] take 4 nop path

  \[ 50\% \] take 6 nop path

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

  for \( 0 \rightarrow 3 \)

  \[ 65\% \] take 3 nop path

  \[ 35\% \] take 5 nop path

Probability of taking short path is

\[ \frac{N - M_{\min} + 1}{N} \]

- **Adaptive**: Look at queues in either direction, select in direction of 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></th>
<th>Tornado</th>
<th>Uniform Random</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greedy</td>
<td>0.33</td>
<td>1.9</td>
</tr>
<tr>
<td>URand</td>
<td>0.40</td>
<td>0.57</td>
</tr>
<tr>
<td>WRand</td>
<td>0.53</td>
<td>0.76</td>
</tr>
<tr>
<td>Adaptive</td>
<td>0.53</td>
<td>1.3</td>
</tr>
</tbody>
</table>

![Tornado and Uniform Random Graphs](image-url)
Tornado on Ring

Node i sends to \( \frac{(i + (N-1) \mod N)}{2} \) or \( \frac{(i + 3 \mod N)}{2} \)

Greedy Algo

Clockwise channels not used at all!
Poor load balancing

Random Algo

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

What about clockwise channels though?
5 paths cross each CW channel each is 0.5 unit of traffic
\( N_{\text{max,CW}} = 2.5 > N_{\text{max,CW}} = 1.5 \)
\( \Theta_{\text{CW}} = 0.4 \)

Now the clockwise channel is the bottleneck!
\( T_\Phi = 0.5 \times 4 + 0.5 \times 6 = 5 \)
Weighted RoundRobin Algo

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

\[
\begin{align*}
\text{5 paths each with } \frac{3}{8} \text{ traffic} \\
N_{\text{max, cw}} &= \frac{3}{\frac{3}{8}} = 1.875 \\
N_{\text{max, ccw}} &= \frac{2}{\frac{3}{8}} = 1.875 \\
\Theta_{\text{cw}} &= \frac{1}{1.875} = 0.53 \\
T_0 &= \frac{5}{8} \times 4 + \frac{3}{8} \times 6 = 2.5 + 2.25 = 4.75
\end{align*}
\]

Balanced!

Adaptive

Minimal Latency at light load
Max throughput at high load
## Routing Taxonomy

- **Oblivious**
  - Deterministic
  - Non-Deterministic
  - Adaptive

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Non-Deterministic</th>
<th>Adaptive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dest Tag</td>
<td>VATAG</td>
<td>ALGO</td>
</tr>
<tr>
<td>Routing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Non-Deterministic</th>
<th>Adaptive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dest Tag</td>
<td>RANDOMLY CHOOSE MID</td>
<td>ROUTER</td>
</tr>
<tr>
<td>Routing</td>
<td>ROUTER</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deterministic</th>
<th>Non-Deterministic</th>
<th>Adaptive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim Essays</td>
<td>ORIGIN</td>
<td>VATAG</td>
</tr>
<tr>
<td>Routing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Note: The table and diagram illustrate various routing taxonomy categories and methods, including deterministic and non-deterministic approaches, as well as adaptive routing techniques.
DEADLOCK

Cyclic dependency in "waits for" and "needs" relation.

Assume
\[ \text{forward } i \rightarrow i+2 \]

\[
\begin{align*}
\text{CWSR} & \quad \text{R1} & \quad \text{R2} & \quad \text{R3} \\
\text{CWSR} & \quad \text{CWSR} & \quad \text{CWSR} & \quad \text{CWSR} & \quad \text{CWSR}
\end{align*}
\]

\[
\begin{align*}
0: & \quad D & \quad A & \quad A & \quad B & \quad B & \quad C & \quad D \\
& \quad \text{Cyclical dependency}
\end{align*}
\]

ACTORS

\[
\begin{align*}
\text{D} & \quad \text{A} & \quad \text{B} & \quad \text{C} \\
\text{wait} & \quad \text{wait} & \quad \text{wait} & \quad \text{wait}
\end{align*}
\]

RESOURCES

\[
\begin{align*}
\text{CWSR} & \quad \text{CWSR} & \quad \text{CWSR} & \quad \text{CWSR} \\
\text{wait} & \quad \text{wait} & \quad \text{wait} & \quad \text{wait}
\end{align*}
\]

- Deadlock avoidance vs. Deadlock detection/recovery

Contrast to Livelock Chaos Routing Algorithm
Activity: Vaian's Algorithm

Consider a 2-ary 2-HHy destination-tag routing with the given permutation traffic pattern. Calculate the ideal terminal throughput and the zero load latency in route hops if we use Vaian's routing algorithm.

Vaian's Algorithm:
First, route to a random destination and then route to the desired destination.

Example route for src 0 to dest 1:

* How would Vaian's do on 8-node rings?
* Try on your own similar activity for 2-ary 3-fly.