# T06 Fundamental Network Concepts

School of Electrical and Computer Engineering  
Cornell University  

revision: 2015-10-21-11-08

## 1 Network/Roadway Analogy

1.1. Running Errands ........................................... 4  
1.2. Network Technology ...................................... 5  
1.3. Networks in Computer Architecture .................. 6

## 2 Network Topology

2.1. Single-Stage Bus Topology ............................... 8  
2.2. Single-Stage Crossbar Topology ....................... 9  
2.3. Multi-Stage Butterfly Topology ...................... 10  
2.4. Multi-Stage Torus Topology .......................... 12  
2.5. Terminology ............................................. 14

## 3 Network Routing

3.1. Oblivious Deterministic Routing ..................... 18  
3.2. Oblivious Non-Deterministic Routing ............... 18  
3.3. Adaptive Routing ..................................... 19  
3.4. Deadlock ............................................. 20
4 Analyzing Network Performance

4.1. Traffic Patterns ...................................................... 21
4.2. Ideal Throughput .................................................... 23
4.3. Zero-Load Latency .................................................... 25
4.4. Comparing Topologies .............................................. 28
4.5. Comparing Routing Algorithms ................................. 29
1. Network/Roadway Analogy

Our goal is to run some errands around town using our bike. Assume we are studying in the Duffield Atrium and we need to: (1) do some laundry in Collegetown (maybe pickup some coffee?), and (2) pick up some books about computer architecture from the library.

- Duffield Atrium
- Coffee/Laundry
- Library
1.1. Running Errands

- **Network Topology**
  - Arrangement of roads and intersections to interconnect sources and destinations
  - Wide vs. narrow roads
  - Long vs. short roads
  - Small vs. large intersections

- **Network Routing**
  - Path from source to destination along roads and intersections
  - Short vs. long paths
  - Common vs. rare paths

- **Network Microarchitecture**
  - Managing long line of bikes and cars on road
  - Managing many cards and bikes at same intersection
1.2. Network Technology

<table>
<thead>
<tr>
<th>AWG24 Twisted Pair</th>
<th>PCB Trace</th>
<th>On-Chip M6 Wire in 0.18µm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resistance</td>
<td>0.08 Ω/m</td>
<td>5 Ω/m</td>
</tr>
<tr>
<td>Inductance</td>
<td>400 nH/m</td>
<td>300 nH/m</td>
</tr>
<tr>
<td>Capacitance</td>
<td>40 pF/m</td>
<td>100 pF/m</td>
</tr>
<tr>
<td>Data Rate</td>
<td>≈Gb/s</td>
<td>≈Gb/s</td>
</tr>
<tr>
<td>Critical Length</td>
<td>1m</td>
<td>10cm</td>
</tr>
<tr>
<td>Pitch</td>
<td>≈mm</td>
<td>&lt;mm</td>
</tr>
</tbody>
</table>

On-Chip Wires

Distributed Wire Resistance and Capacitance

Because both wire resistance and wire capacitance increase with length, wire delay grows quadratically with length.

On-Chip Buffers

On-chip buffers are 1r1 FIFOs implemented using either a register file or SRAM.

On-chip network technology constraints very different from off-chip technology constraints.
1.3. Networks in Computer Architecture

- **Network Topology**
  - Arrangement of channels and routers to interconnect sources and destinations
  - Roads = channels (implemented with cables, traces, wires)
  - Width of road = channel bw
  - Intersections = routers
  - Size intersection = router radix

- **Network Routing**
  - Path from source to destination along channels and routers
  - Short vs. long paths = minimal vs. non-minimal paths
  - Common vs. rare paths = channel congestion

- Topology is constrained by packaging (geography)
Network Transactions

We will use processor-memory networks as a driving example throughout the course. Processor-memory networks allow many processors to perform memory read/write transactions on many memories.

- **Message** = Logical unit of data transfer provided by network interface
- **Packet** = Unit of routing within a network
- **Flit** = Smallest unit of resource allocation in channel/router (flow-control digit)
- **Phit** = Smallest unit of data processed by a channel/router (physical digit)

For the processor-memory networks we will primarily study:

\[
\text{Message} = \text{Packet} = \text{Flit}
\]

The processor will send memory request messages/packets over the network, and the memories will send memory response messages/packets back to the processor. In this context, we consider the network messages/packets to be the “transactions”.

<table>
<thead>
<tr>
<th>Processors : Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memories : Memory accesses</td>
</tr>
<tr>
<td>Networks : Network packets</td>
</tr>
</tbody>
</table>
2. Network Topology

Network topology is the arrangement of channels and routers to interconnect sources and destinations. We will explore four different topology classes:

- Single-stage bus topologies
- Single-stage crossbar topologies
- Multi-stage butterfly topologies
- Multi-stage torus topologies

2.1. Single-Stage Bus Topology

SWMR Bus

MWSR Bus

MWMR Bus
Integrating Multiple Buses

2.2. Single-Stage Crossbar Topology

- Single-stage topologies are difficult to scale in terms of cycle time, energy, and area
- Multi-stage topologies improve scalability but raise many other interesting challenges
2.3. Multi-Stage Butterfly Topology

The Multi-Stage Butterfly Topology is characterized by a hierarchical structure where nodes are organized in stages. Each stage consists of routers connected in a specific pattern, allowing for efficient communication among terminals.

- **k-ary**: Refers to the number of inputs or outputs at each stage.
- **n-fly**: Represents the number of stages in the network.

Each router is similar to a 2x2 crossbar, facilitating unidirectional communication channels.

Mathematically, the number of nodes in the network can be calculated using the formula $2^n$, where $n$ is the number of stages.

Each stage is connected to the next in a manner that ensures every terminal is reachable from any other terminal through a sequence of stages.

![Diagram of Multi-Stage Butterfly Topology](image-url)
Example Butterfly Topology: 4-ary 2-fly

Example Butterfly Topology: 3-ary 2-fly

Sketch a 3-ary 2-fly. Use circles for the terminals, squares for the routers, and lines for one uni-directional channel.
2.4. Multi-Stage Torus Topology

\[ \text{1-Ary N-Cube} \]

Nodes in each dimension

\[ \text{N-Dim Grid} \]

Input + Output Terminal

Two unidirectional channels in opposite directions

Router

4-Ary 2-Cube

4-Ary 2-Cube

MESH

4-Ary 2-Cube

Torus

"Wrap around" channels

3-Ary 3-Cube MESH
Example Torus Topology: 8-ary 1-cube torus

Sketch a 8-ary 1-cube torus. Use circles for a terminal+router and lines for the two uni-directional channels in opposite directions.

Constructing k=ary n-cube from k k-ary (n-1) cubes
2.5. Terminology

Nodes and Channels

- Uni-directional channels in butterfly
- Bi-directional channels in mesh
- Indirect Network: node is either a terminal or router (butterfly)
- Direct Network: node combines a terminal and router (mesh)

Channel parameters

- $w_c$ = channel width (number of pins/wires)
- $f_c$ = channel frequency
- $t_c$ = channel latency
- $l_c$ = channel length
- $b_c$ = channel bandwidth ($w_c \times f_c$)
Bisection Cuts

- **Cut**: set of channels that partitions terminals into two sets
- **Bisection Cut**: cut that partitions terminals in two equal halves
- **Min Bisection Cut** ($B_c$): min channel count over all bisection cuts
- **Bisection Bandwidth** ($B_B$): min bandwidth over all bisection cuts
- For networks with uniform channel bandwidth: $B_B = B_c \times b_c$
- Bisection bandwidth is a good way to estimate global wiring resources (i.e., technology constraints)
- Example from previous butterfly/mesh topologies
  - Cut A on butterfly is a minimum bisection cut
  - Both cut A and B on mesh are bisection cuts
  - Cut A on mesh is a minimum bisection cut

Paths

- **Channel Hop Count** ($H_c$): number of channels on a path
- Channels from terminal to first router may or may not be included
- **Router Hop Count** ($H_r$): number of routers on a path
- **Minimal Path**: smallest hop count between two terminals
- **Diameter** ($H_{\text{max}}$): largest minimal path between all terminal pairs
- **Average min hop count** ($H_{\text{min}}$): average over all terminal pairs
- Example from previous butterfly/mesh topologies
  - Path from A to B on mesh: $H_r = 5$, $H_c = 4$
  - $H_{\text{max,bfly}} = 4$, $H_{\text{max,mesh}} = 8$
  - $H_{\text{min,bfly}} = 4$
Calculating $H_{min}$ usually requires enumerating the minimal hop count for every source/destination pair. The book states that $H_{min}$ for a torus with even $k$ is $nk/4$ and for a mesh with even $k$ is $nk/3$. Where does this come from?

**Calculating $H_{min,n}$**

$\begin{array}{ccc}
\text{src} & 0 & 1 & 2 & 3 \\
0 & 1 & 2 & 3 & 2 \\
1 & 2 & 1 & 2 & 3 \\
2 & 3 & 2 & 1 & 2 \\
3 & 2 & 3 & 1 & 2 \\
\end{array}$

$1 \times 4 = 4$ including $i \rightarrow i$ $32/16 = 2$

$2 \times 8 = 16$ excluding $i \rightarrow i$ $28/12 = 2.3$

**Calculating $H_{min,c}$ with terminal channels**

$\begin{array}{ccc}
\text{src} & 0 & 1 & 2 & 3 \\
0 & 3 & 4 & 3 & 1 \\
1 & 3 & 2 & 5 & 4 \\
2 & 4 & 2 & 5 & 3 \\
3 & 4 & 3 & 2 & 1 \\
\end{array}$

$2 \times 4 = 8$ including $i \rightarrow i$ $40/16 = 2.5$

$3 \times 8 = 24$ excluding $i \rightarrow i$ $40/12 = 3.3$

**Calculating $H_{min,c}$ without terminal channels**

$\begin{array}{ccc}
\text{src} & 0 & 1 & 2 & 3 \\
0 & 1 & 2 & 1 & 2 \\
1 & 0 & 1 & 2 & 1 \\
2 & 1 & 0 & 1 & 2 \\
3 & 1 & 2 & 0 & 1 \\
\end{array}$

$1 \times 8 = 8$ including $i \rightarrow i$ $16/16 = 1$

$2 \times 4 = 8$ excluding $i \rightarrow i$ $16/12 = 1.3$

Equation from Book is $\frac{nk}{3}$ for even $n$

$\frac{nk}{3} = \frac{2 \times 2}{3} = 1.33$

This is for $H_{max}$ without terminal channels, ignoring $i \rightarrow i$.
### Path Diversity

- Path diversity is number of paths between any two terminals
- Mesh has high path diversity, butterfly has no path diversity

### Example Topologies

![Mesh and SMesh topologies](image)

Assume $b_c$ for mesh is 32 bits/cycle, while $b_c$ for smesh is 16 bits/cycle. Smesh has reduced channel bandwidth due to increased number of channels, assumes specific technology constraint. Calculate the following parameters for each topology.

<table>
<thead>
<tr>
<th></th>
<th>$B_C$</th>
<th>$B_B$</th>
<th>$H_{max}$</th>
<th>$H_{min}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>mesh</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>smesh</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3. Network Routing

Network routing is the choice of path from source to destination along channels and routers. Routing algorithms can be oblivious (not factor in network state) or adaptive (factor in network state), and can also be deterministic or non-deterministic.

3.1. Oblivious Deterministic Routing

- Ring: always use minimal path, equidistant choose CW
- Butterfly: use destination to choose middle router
- Mesh: route in X first, then route in Y (Dimension-Ordered-Routing)

3.2. Oblivious Non-Deterministic Routing

- Ring: randomly choose CW vs CCW
- Butterfly: randomly choose middle router
- Mesh: randomly choose between XY-DOR and YX-DOR (O1-TURN)
3.3. Adaptive Routing

- Ring: look at queues, choose direction with least congestion
- Butterfly: look at queues, choose middle router with least congestion
- Mesh: look at queues during each hop to choose X vs Y
3.4. Deadlock

A cyclic dependency in "wants for" and "holds" relation.

Assume $i := i + 2$

- Deadlock avoidance vs deadlock detected/recovery
- Contrast to livelock chaos routing algorithm
4. Analyzing Network Performance

Similar to analyzing processors and memories, we will use simple first-order equations to help build intuition about throughput and latency trade-offs in networks.

4.1. Traffic Patterns
4. Analyzing Network Performance

4.1. Traffic Patterns

Channel 2 → 3 oversubscribed but so is ejected channel to output terminal at node 3

logical to physical mapping

Assume partition traffic pattern
Now are logical source/dest IDs in traffic pattern mapped to physical terminal IDs?

Mapping can turn any logical permutation pattern into any other pattern.
given permutation pattern usually assumes a specific mapping.
4.2. Ideal Throughput

Channel load \( n_c \) is amount of traffic that crosses channel \( c \) if each input injects one unit of traffic according to given traffic pattern.

Example traffic pattern:

```
0 1 2 3 4 5 6 7
0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0
3 0 0 0 0 0 0 0
4 0 0 0 0 0 0 0
5 0 0 0 0 0 0 0
6 0 0 0 0 0 0 0
7 0 0 0 0 0 0 0
```

Channel load ranges from 0 to 7.

Max channel load \( n_{\text{max}} \) is 7, these are the bottleneck channels that will limit throughput.

Alternative way to think about \( n_c \):

Channel load is ratio of BW demands from channel \( c \) to BW injected by one terminal.
4. Analyzing Network Performance

4.2. Ideal Throughput

**Ideal Throughput**

\[ \Theta_{term} = \frac{bc}{\gamma_{max}} \]

In previous example

\[ \Theta_{term} = \frac{b}{\gamma_{max}} = \frac{b}{2} \]

\[ \Theta_{tot} = N \cdot \frac{b}{\gamma_{max}} = B \cdot \frac{b}{2} = \gamma b \]

**Small TFLY Examples**

1. \( \gamma_{max} = 1 \)
   \[ \Theta_{term} = 6 \]

2. \( \gamma_{max} = 1 \)
   \[ \Theta_{term} = 6 \]

3. \( \gamma_{max} = 2 \)
   \[ \Theta_{term} = 6/2 \]

4. \( \gamma_{max} = 4 \)
   \[ \Theta_{term} = 6/4 \]
More generally for uniform random traffic

- on average with uniform random traffic, half the traffic crosses the direction
- N total units of traffic, N/2 cross direction
- Ideal routing will evenly balance load across all channels (this was an issue in rings example)

\[ n_{\text{max}} = \frac{N/2}{T_{E}} = \frac{N}{2B_{E}} \]

\[ \Theta_{\text{tem}} = \frac{6}{N/2B_{E}} = \frac{26B_{E}}{N} = \frac{2B_{E}}{N} \]

\[ \Theta_{\text{tot}} = N \frac{2B_{E}}{N} = 2B_{E} \]

4.3. Zero-Load Latency

---

36 B/cycle = packet length
86 B/cycle = byte PDU

Serialize packet into four PDU's

pkt 0
pkt 1
SSSSSS
SSSSS
4. Analyzing Network Performance

4.3. Zero-Load Latency

Formalized Latency Model:

\[ T = T_{\text{Head}} + \frac{L}{b} \]

- **Head Pack Latency**: Includes transfer time, hop count, and contention.

**Zero-Load** Latency:

\[ T_0 = H_b t_n + H_c t_c + \frac{L}{b} \]

- **Latency due to transfer hops**
- **Latency due to channel hops**
- **Serialization Latency** (no contention)
4. Analyzing Network Performance

4.3. Zero-Load Latency

Four ways to improve latency:

\[ T_\phi = H_{rc} + H_{te} + \frac{L}{b} \]

- Shorter routes
- Faster routes
- Faster encoders
- Wide channels
- Shorter msgs

Average latency vs. offered Tbw

New ideal routing + flow control

[Diagram showing latency vs. offered Tbw]
4.4. Comparing Topologies

Assume:

\[ L = 1286 \]
\[ b_c \text{ for mesh is } 32 \text{ b/cycle} \]
\[ b_c \text{ for s-mesh is } 16 \text{ b/cycle} \]
\[ t^e = 3 \]
\[ t^c = 1 \]

Which topology can achieve higher ideal throughput under uniform random traffic?

(a) \( \Theta_{term, mesh} > \Theta_{term, s-mesh} \)

(b) \( \Theta_{term, mesh} < \Theta_{term, s-mesh} \)

(c) \( \Theta_{term, mesh} = \Theta_{term, s-mesh} \)

Calculate zero load latency under uniform random traffic.
4.5. Comparing Routing Algorithms

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

- **Uniform Random**: Randomly pick direction

  - For $0 \rightarrow 3$
    - 50% take 4 hops
    - 50% take 6 hops

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

  - For $0 \rightarrow 3$
    - 5/8 take 3 hop path
    - 3/8 take 5 hop path

  Probability of taking short path is
  
  \[ N = \frac{H_{\text{min}} + 1}{N} \]

- **Adaptive**: Look at queues in either direction, send in direction of queue that has most free entries.

  Do not change direction after initial choice.
• Evaluate tornado traffic pattern on the 8-node ring
• Recall that in tornado, node $i$ sends to $i + ((N - 1)/2) \mod N$
Weights vs. Arcs

4. Analyzing Network Performance

4.5. Comparing Routing Algorithms

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

3 paths each with 5/8 traffic

$\eta_{max, ccw} = \frac{5 \cdot \frac{5}{8}}{8} = 1.875$

3 paths each with 3/8 traffic

$\eta_{max, ccw} = \frac{3 \cdot \frac{3}{8}}{8} = 1.875$

$\eta_{max, ccw} = \eta_{max, ccw}$ \textbf{Balanced!}

$\Theta_{ccw} = \frac{1}{1.875} = 0.53$

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

\textbf{Adaptive}

Minimal latency at light load
Max throughput at high load
Let’s fill in the following table to summarize the performance of the four routing algorithms on the 8-node given the tornado and uniform random traffic patterns.

<table>
<thead>
<tr>
<th>Routing Algo</th>
<th>Tornado</th>
<th>Uniform Random</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\Theta_{term}$</td>
<td>$T_0$</td>
</tr>
<tr>
<td>Greedy</td>
<td>1.00</td>
<td>3.0</td>
</tr>
<tr>
<td>Uniform Random</td>
<td>0.57</td>
<td>4.5</td>
</tr>
<tr>
<td>Weighted Random</td>
<td>0.76</td>
<td>3.1</td>
</tr>
<tr>
<td>Adaptive</td>
<td>1.00</td>
<td>3.0</td>
</tr>
</tbody>
</table>

**Tornado Traffic Pattern**  
Bandwidth vs. Latency

**Uniform Random Traffic Pattern**  
Bandwidth vs. Latency
Activity: Routing on butterfly with extra stage

Consider two routing algorithms for a 2-ary 2-fly with an extra stage:

- Destination Tag Routing: use destination to choose middle router
- Random Middle Router: randomly choose middle router

Compare these two routing algorithms in terms of ideal terminal throughput ($\Theta_{term}$) and zero-load latency ($T_0$) for the following permutation traffic pattern.

- src 0 $\rightarrow$ dest 1
- src 1 $\rightarrow$ dest 0
- src 2 $\rightarrow$ dest 3
- src 3 $\rightarrow$ dest 2

*Hint: Use the tick-mark method to calculate the max channel load for each routing algorithm.*