Interconnection Networks

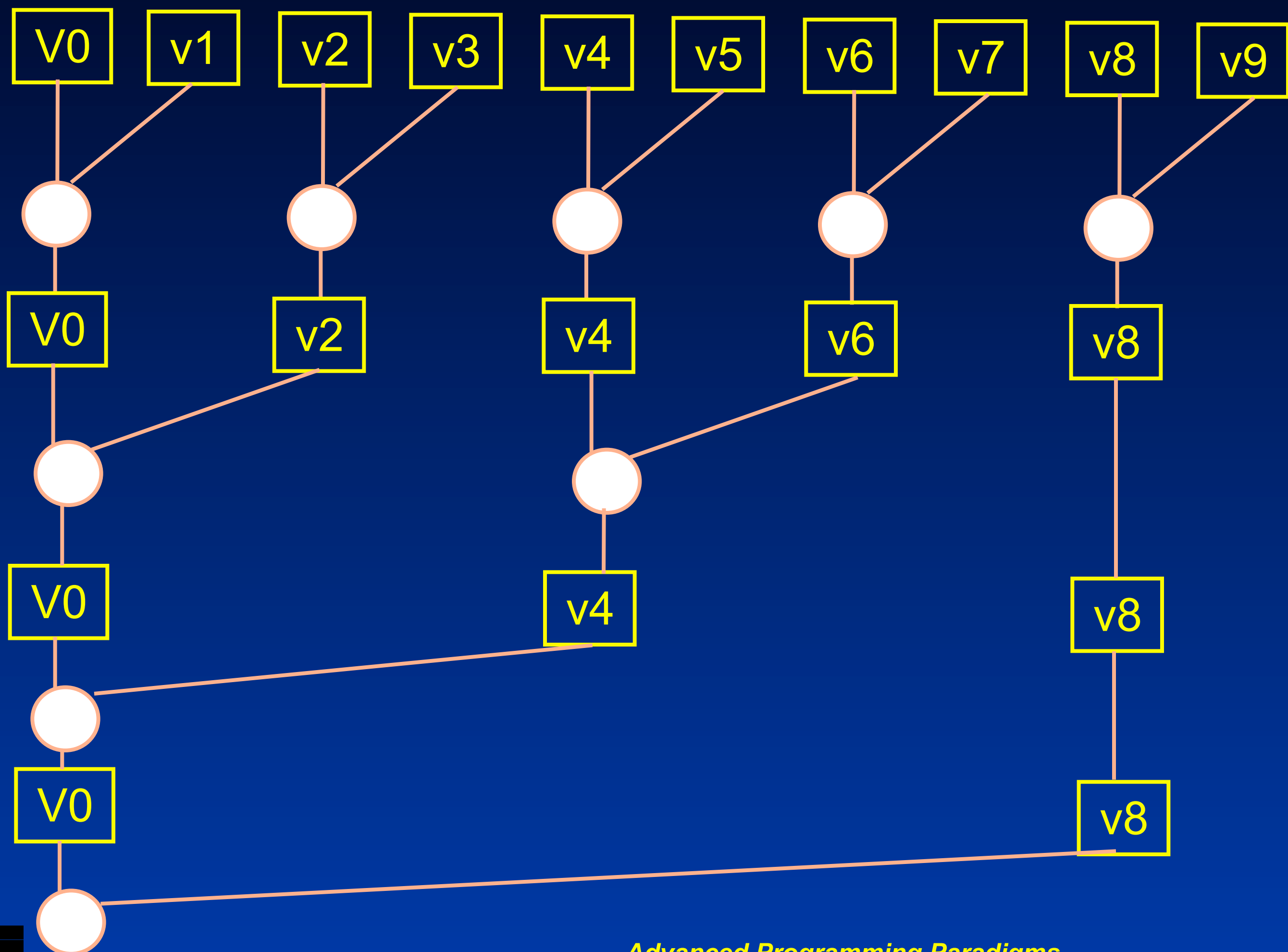
Chapter 3 - El Rewini and Lewis

Processors and the IN

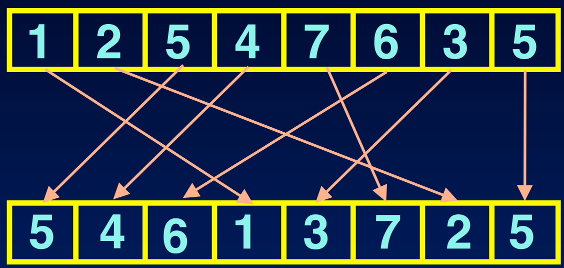
- A parallel processor has two major components:
  - The processors
  - The interconnection network (IN)
- The IN is very important in *all* high-performance machines
  - Static IN
  - Dynamic IN
- We look briefly at some typical examples
  - Not an exhaustive study

Recall ...

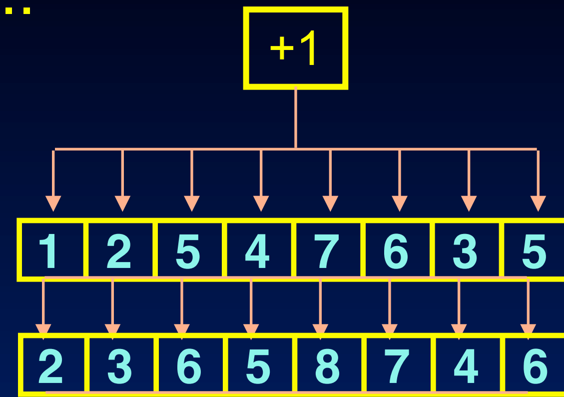
- This reduce operation



- and other data parallel operations ...



Permutation



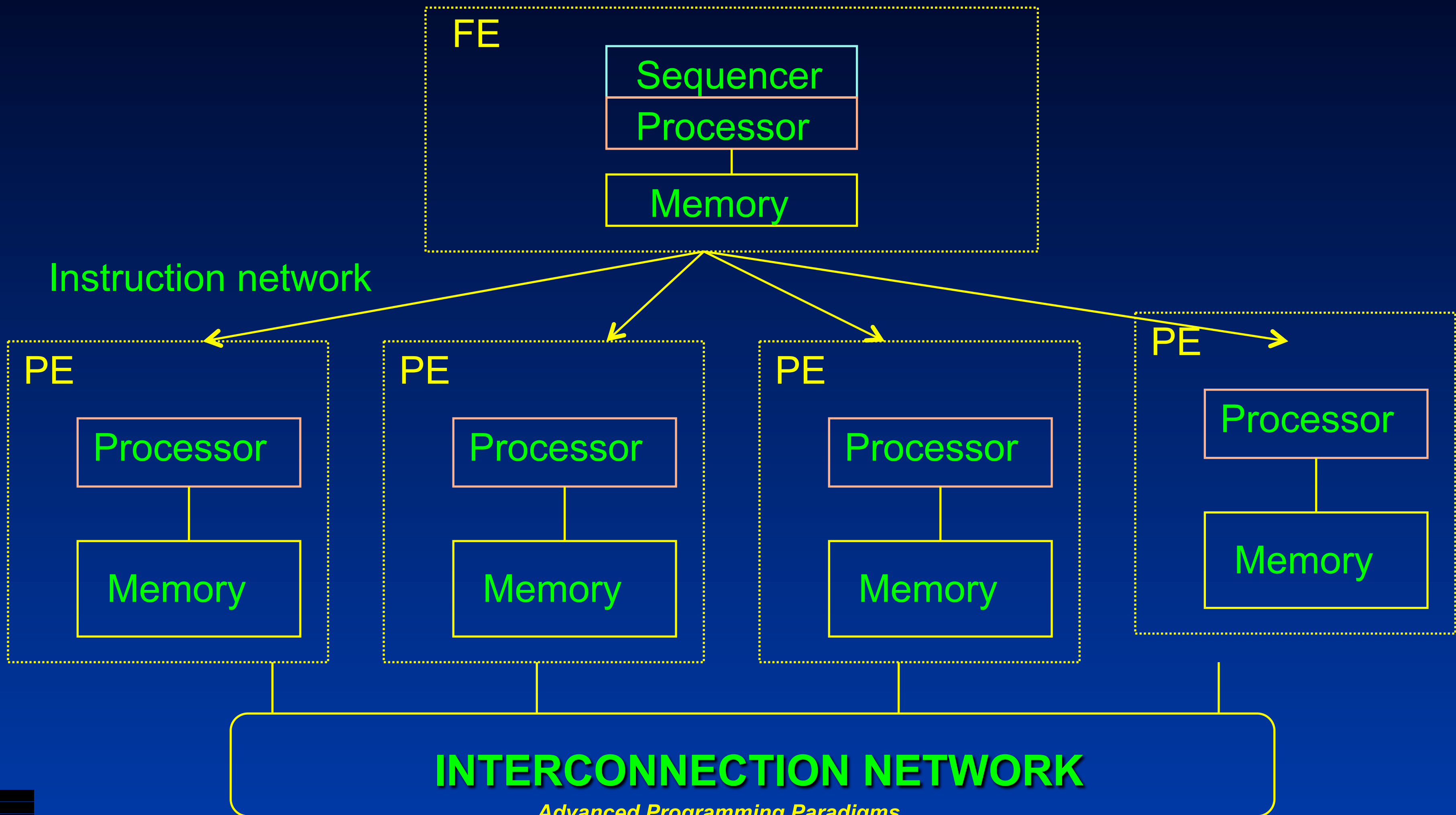
Map

- all involve *communication* via the *interconnection n/w (IN)*
  - clearly crucial to performance

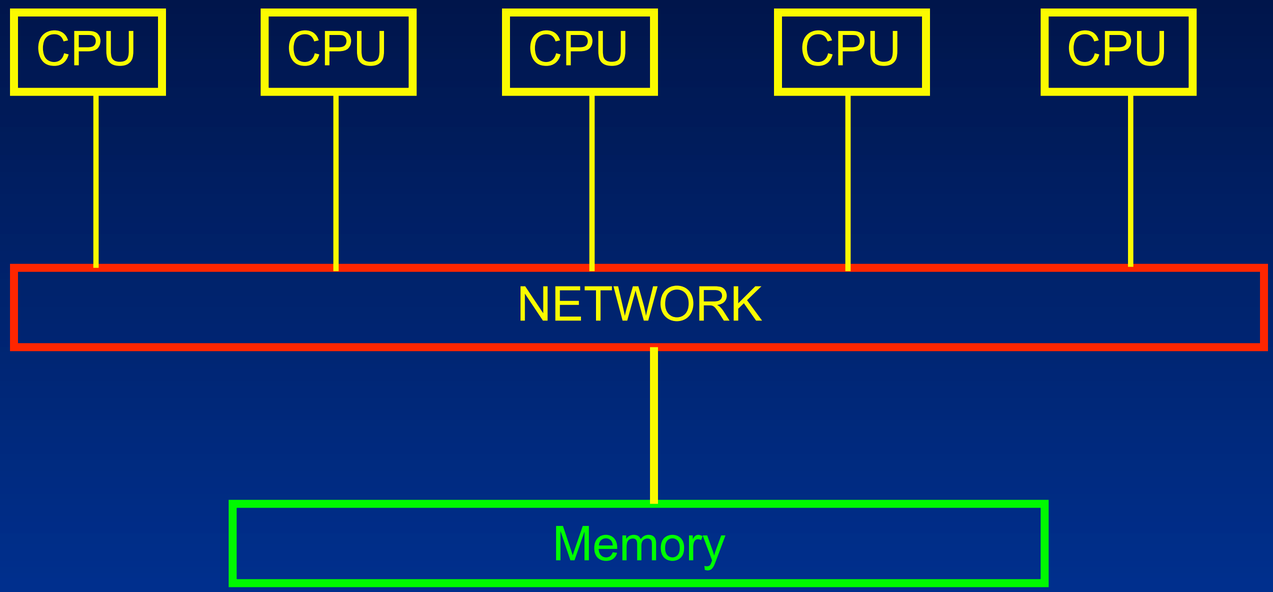
...

Typical SIMD architecture

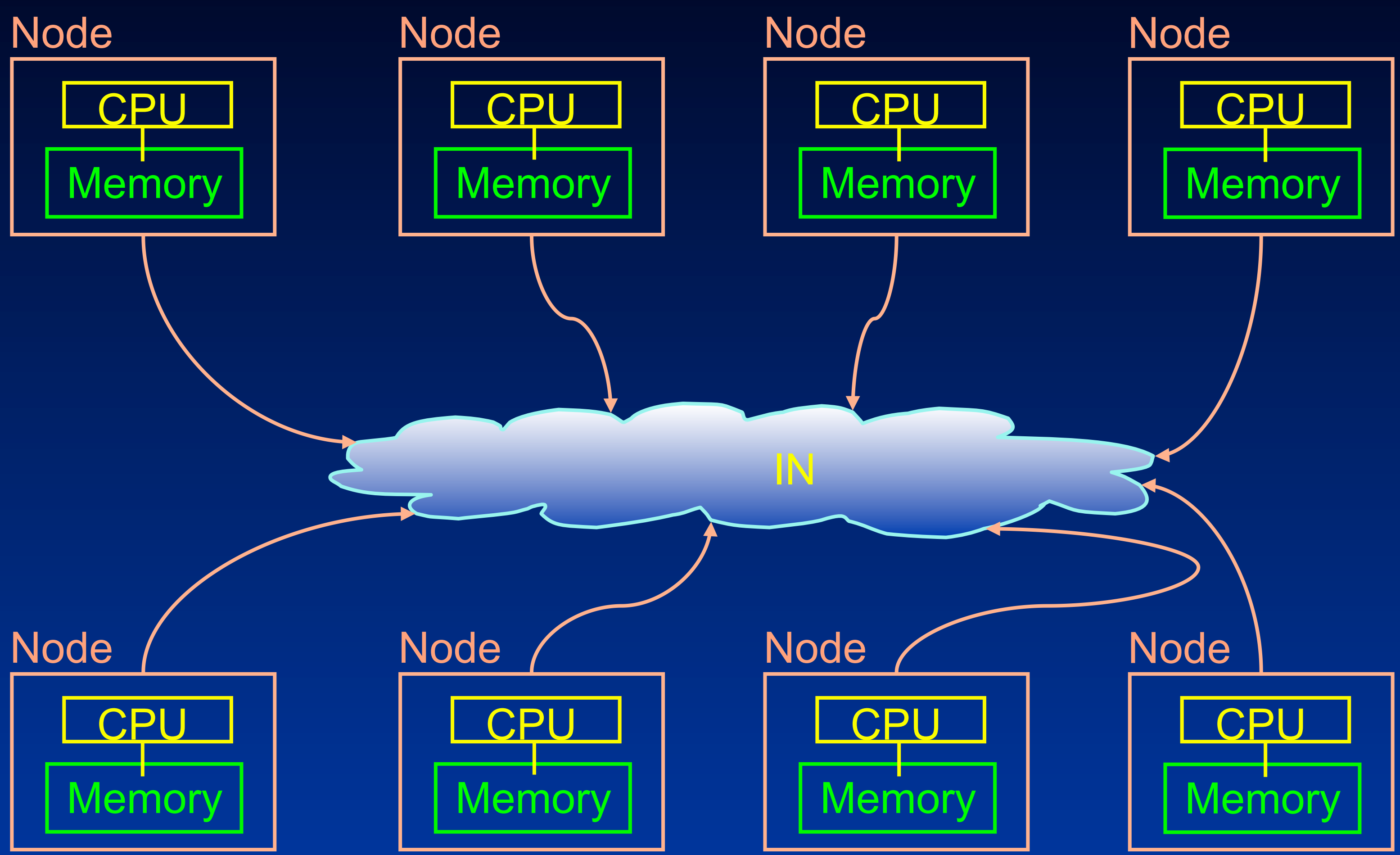
IN design influences performance



Shared Memory MIMD



Distributed Memory MIMD



hardware IN taxonomy

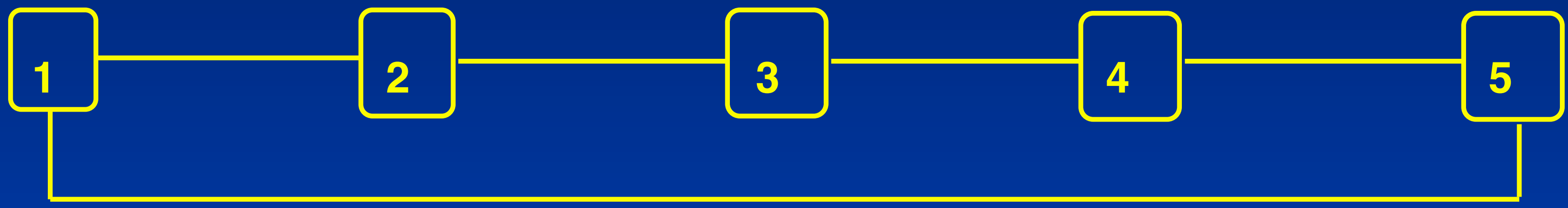
- IN Topology
  - Distributed Memory
    - Static
      - 1-D
      - 2-D
      - Hypercube
    - Dynamic
      - Single Stage
      - Multi Stage
      - Tree
      - Cross Bar
  - Shared Memory
    - Vector
    - MIMD

Static INs

- Static INs are hardwired into a machine
  - Processors connected directly to each other – no switches
    - » Simplest example: linear ring, see next slide.
  - Communication between neighbours is fast but more distant nodes must use multiple steps.
  - Communications primitives available determined by topology.
    - » ie. left and right for a 1-D mesh; left, right, up, down for 2D mesh.
- Static topologies do well on problems which have predictable communications patterns.
- Tends to make machines more special-purpose.
- Used for both SIMD and MIMD designs – not shared memory.

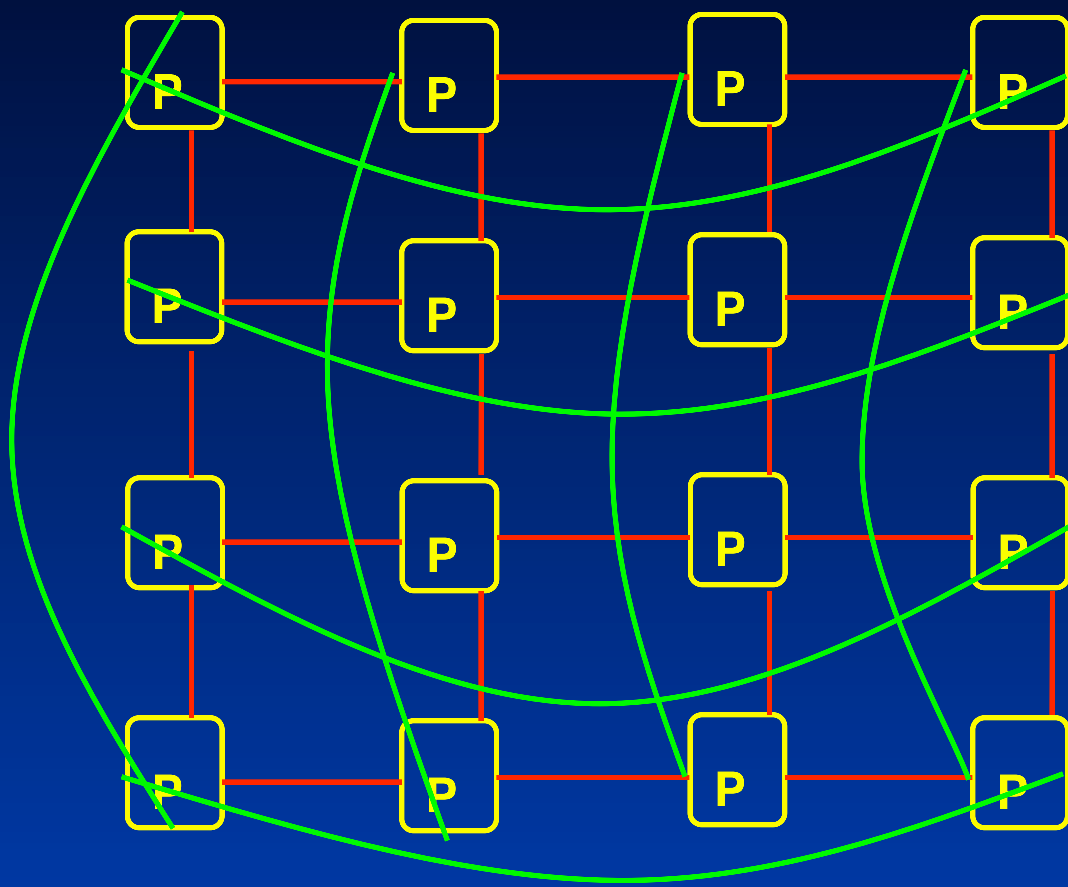
Ring (1D-Mesh)

- An example of a 1-D static IN is the *ring*
  - Conectivity: 2 nearest neighbours.
  - Average hops for message *N/3* in example below
  - There is a choice of routes
  - Cheap: cost grows with N.
  - Max Latency also grows with N ….



2D - Mesh

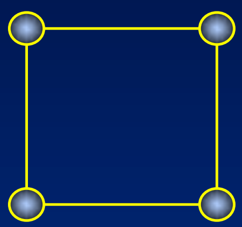
- Latency now grows with ROOT(N)
- Connectivity: 3 – 4 neighbours.
- Often join the boundaries to make rings



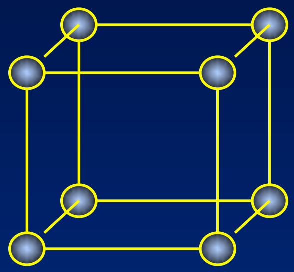
Hypercube

0-Dimensions

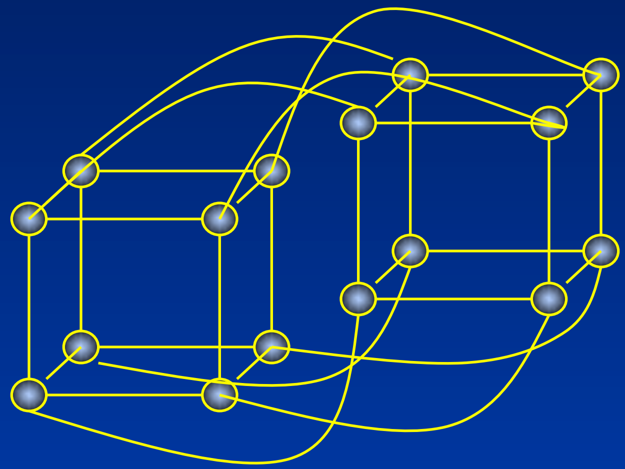
1-Dimension



2-Dimensions



3-Dimensions



4-Dimensions

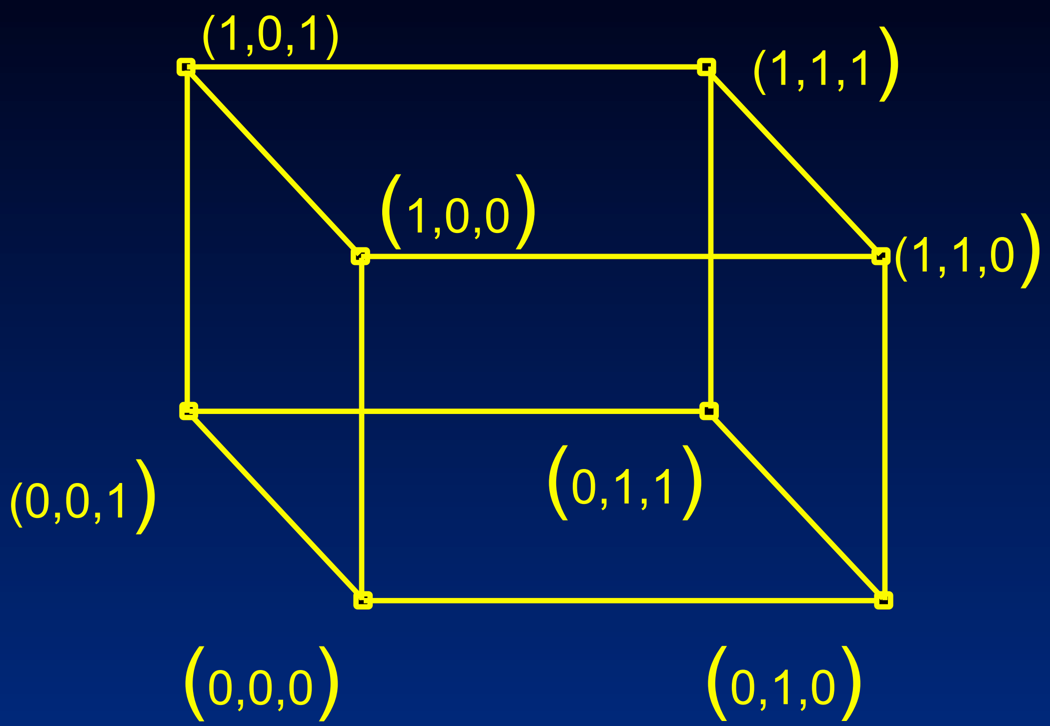
CM2 has 12-D hypercube
with sub-grids at each vertex

..hypercubes

- Properties of *d*-dimensional hypercube
- Latency grows with log(N)
- Wire cost also N log(N) as opposed to N for mesh
  - 2^d nodes
  - *d* links per node
  - max path length is *d*
  - shortest path between two nodes is Hamming distance
    - » number of binary digits that are different *(exclusive-or)*

Example – 3D cube

- d=3
- 2^d = 2^3 = 8 nodes
- d links per node = 3
- max path length = d = 3

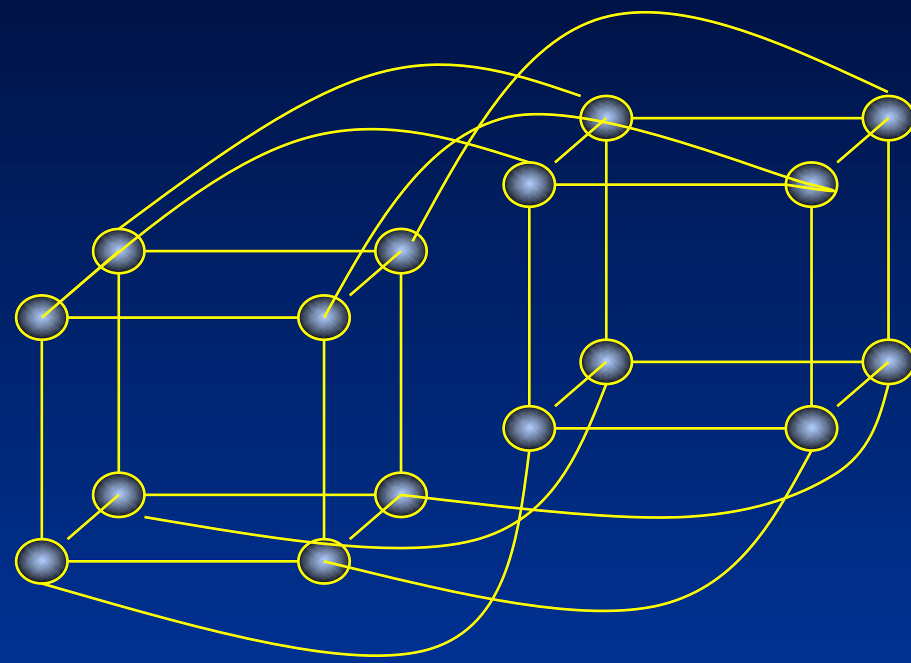


direct connection == Hamming distance = 1

Node (1,1,0) connected to: (0,1,0), (1,0,0) and (1,1,1)

Example – 4D cube

- d = 4
- 2^4 = 16 nodes
- 4 links per node
- max path length = 4



Caltech Cosmic Cube

- Developed from 1981 onwards
- 64 Intel 8086/87 processors (4-5 Mhz)
- 128kB ram per processor
- 6D hypercube network => each processor directly connected to six others
- Peak performance: 3 MFLOPS

Hypercube - Routing

- Routing is very simple in a hypercube.
  - Calculate SRC XOR DEST
  - number of bits = hamming distance = number of hops needed.
  - also equals number of different ways of routing the message (along shortest paths).
  - to route message, we change each XOR'd bit one at a time.
- Example:
  - Route (0,1,0) -> (1, 1, 1)

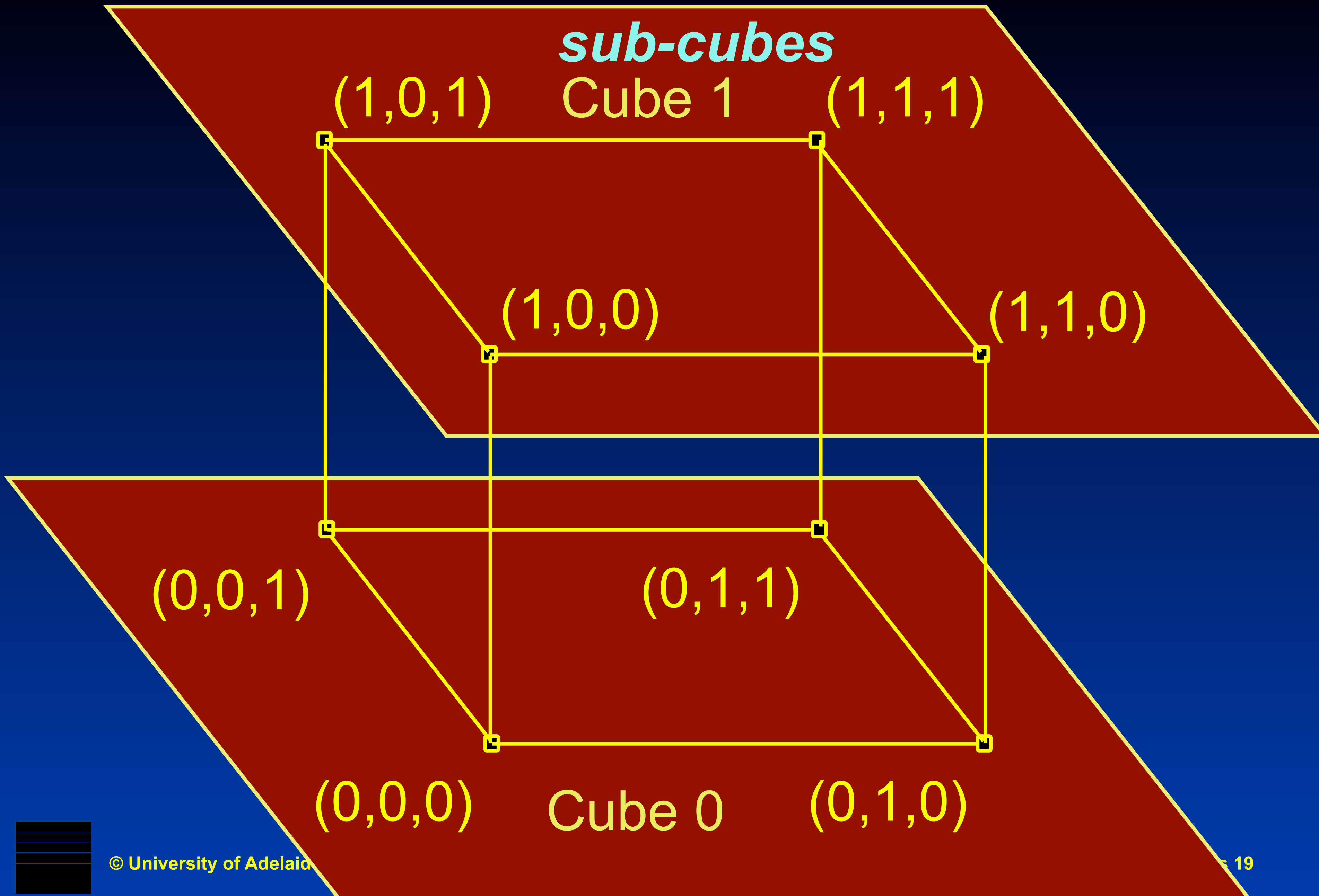
```
      0 1 0
      1 1 1
  XOR 1 0 1
```

- Two bits in XOR -> two hops needed and two possible paths:
- (0, 1, 0) -> (0, 1, 1) -> (1, 1, 1) AND (0, 1, 0) -> (1, 1, 0) -> (1, 1, 1)

Partitioning and mapping

- Hypercubes may be partitioned into sub-cubes
  - e.g. 3-D cube broken down into 2 2-D cubes
- Take first digit as the sub-cube identifier
  - Remaining digits identify a node in the sub-cube
- Lower dimensional structures can be mapped onto a hypercube
- e.g. 2-D cube with nodes (0,0), (0,1), (1,1), (1,0)
  - *Ring* imposed by
    - » Always routing messages in the above sequence
      - (with wrap-around)

sub-cubes



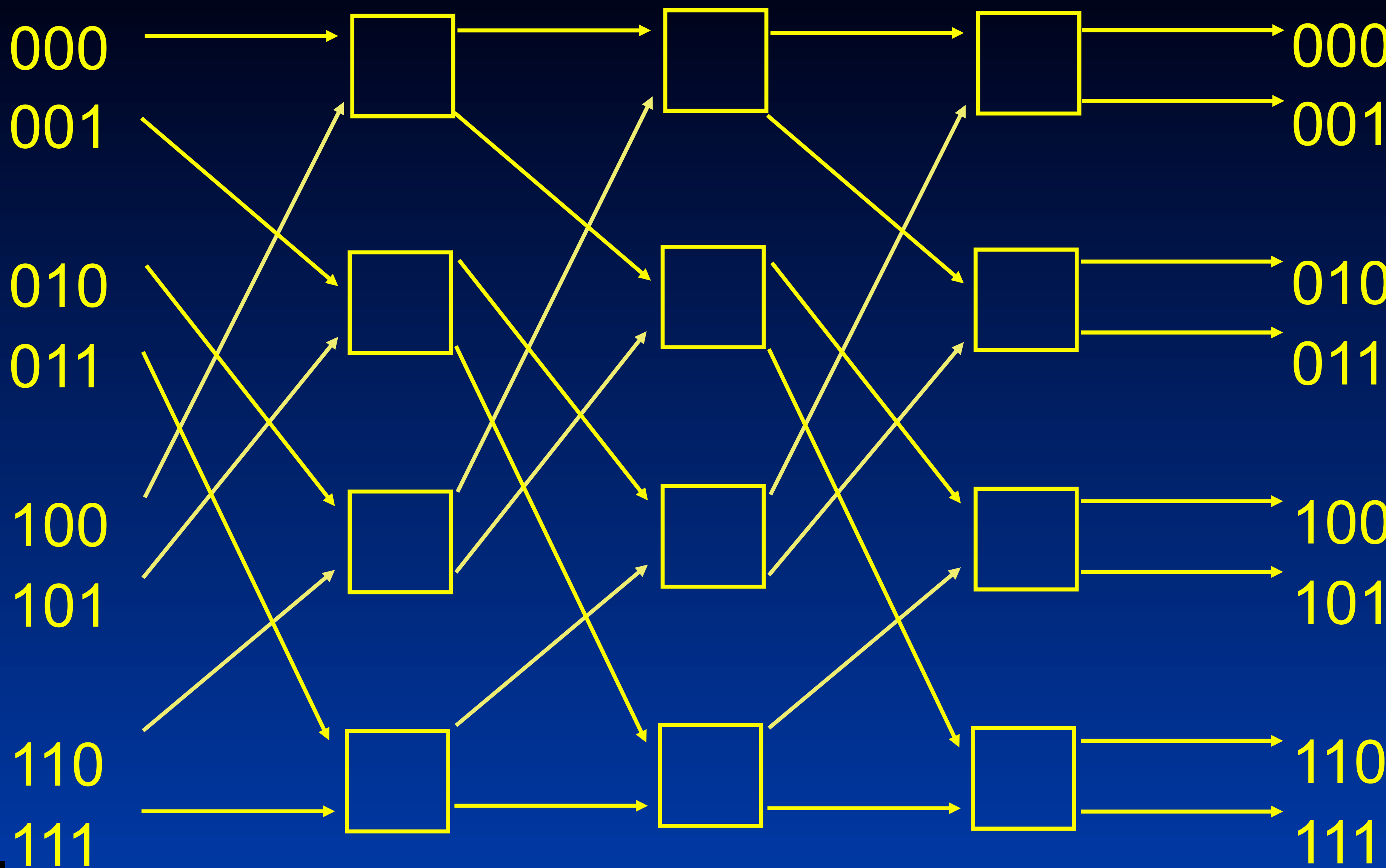
Hypercubes cont.

- Advantages
  - Small diameter – ie. maximum distance between two nodes grows slowly – log(N)
  - Easy, well defined routing strategy
  - Easily divided into sub-networks
  - More than one shortest path between two nodes.
    - » less contention
- Disadvantages
  - Fairly high wire cost: N logN. Also number of connections per node = d.
  - Lack of scalability on existing small systems – to increase size, we must add a connection to each existing node.

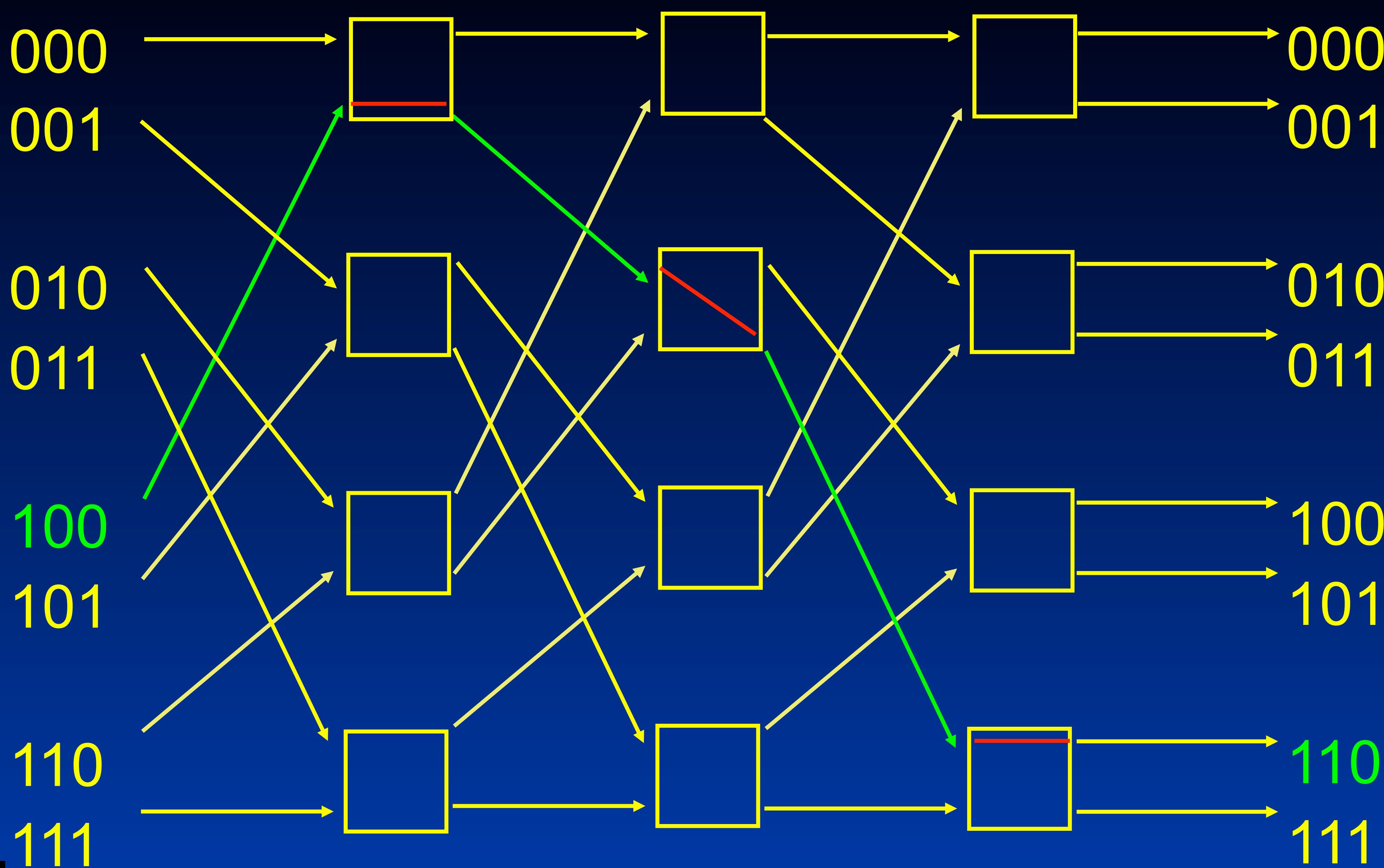
Dynamic INs

- A set of switches
  - Messages *dynamically switched* from node to node
  - Physical network of switches; firmware network for routing
  - Usually multiple stages in any point to point connection
    - at each switch the next binary digit is used
  - MINs
- Routes may be *blocked* if paths intersect

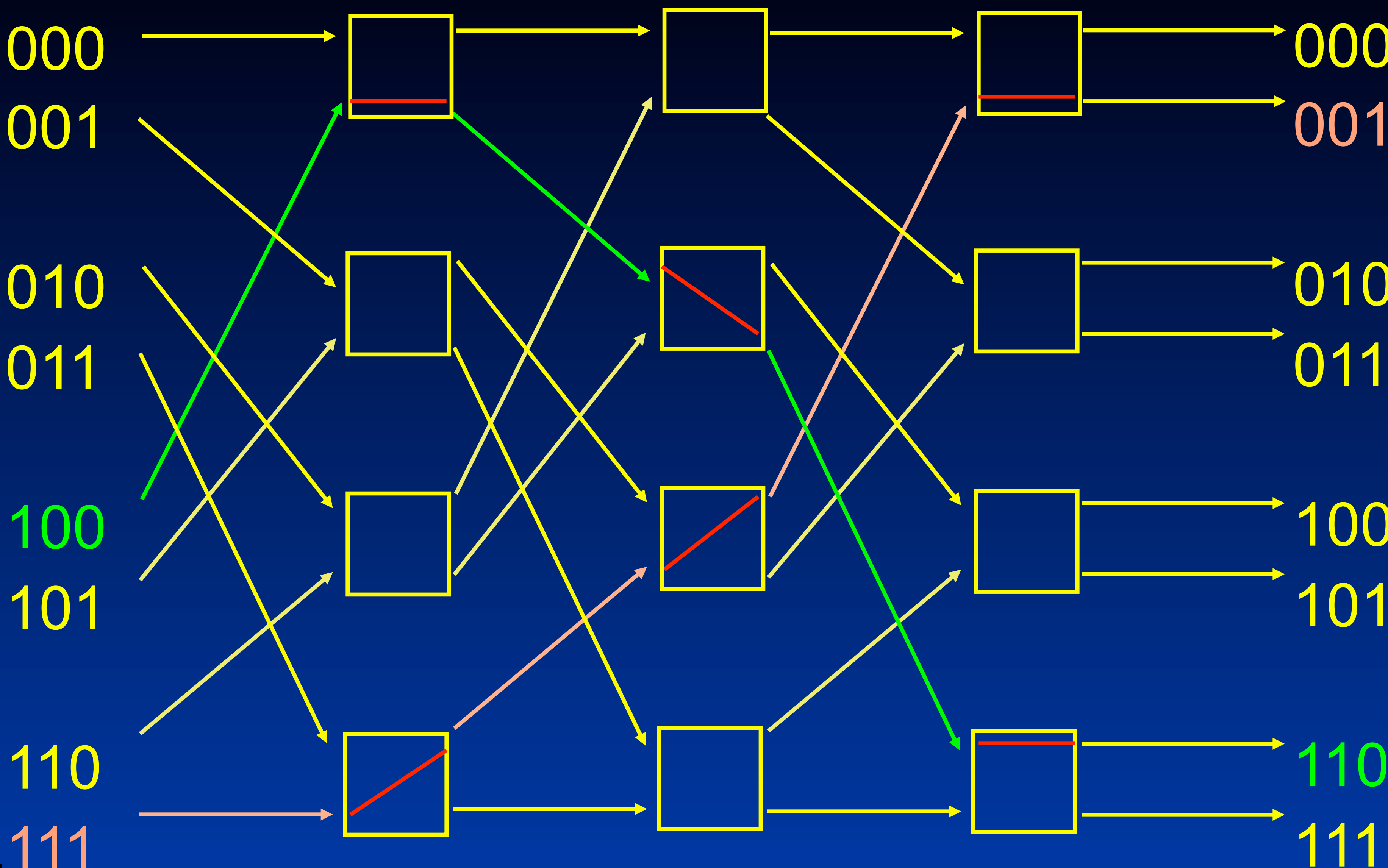
8x8 Omega network



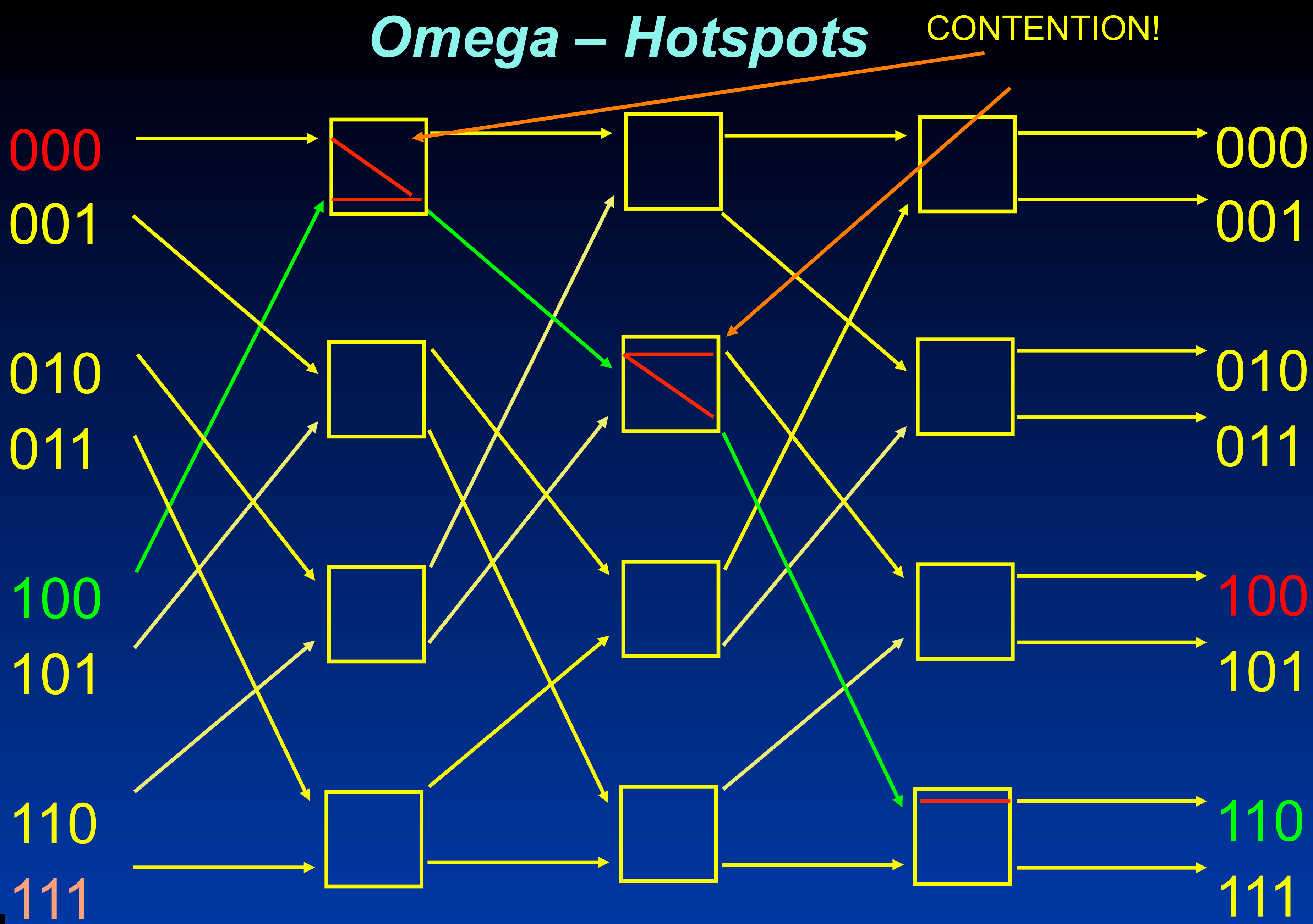
Omega – example: 100 -> 110



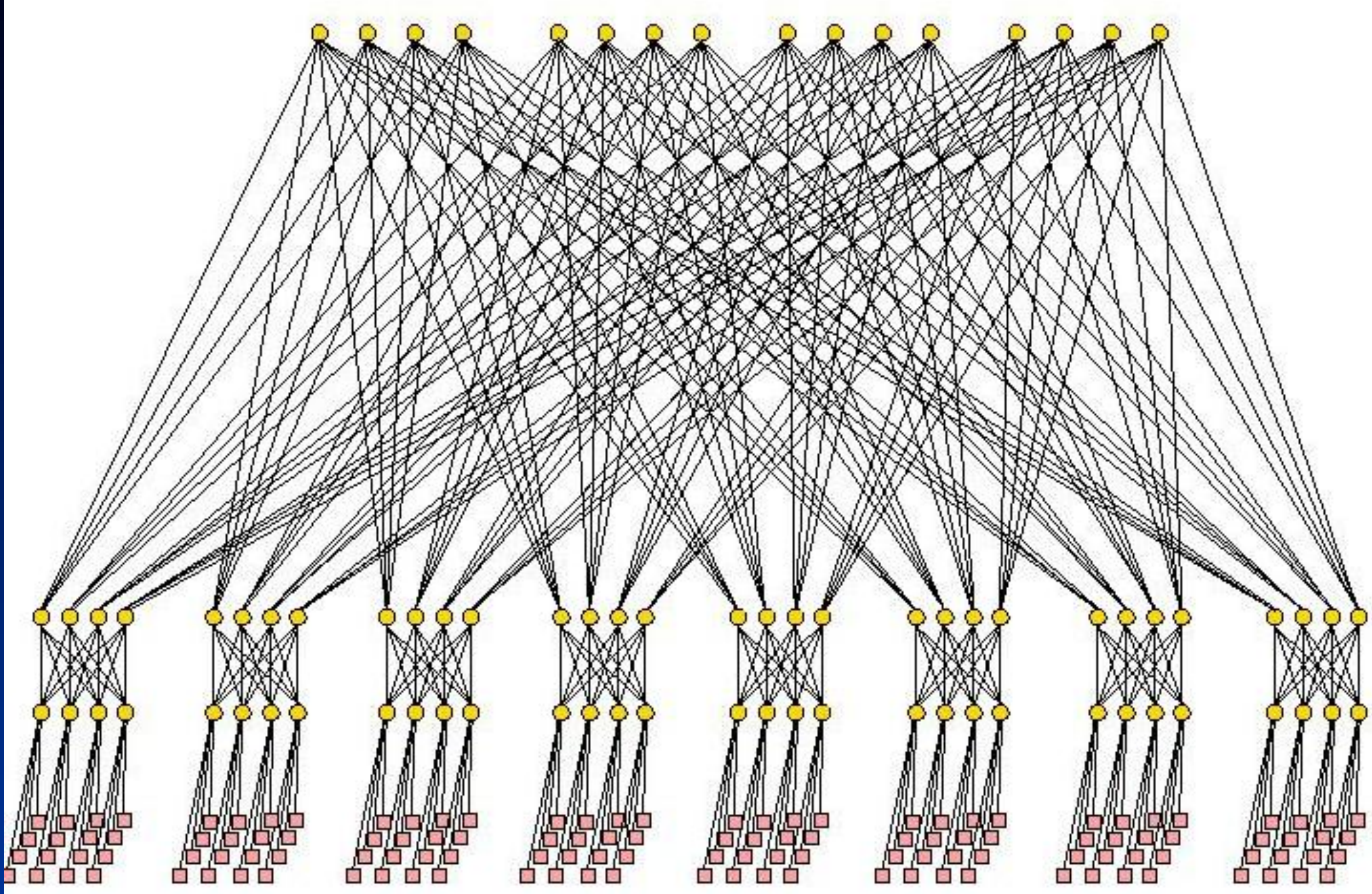
Omega – example: 111 -> 001



Omega – Hotspots



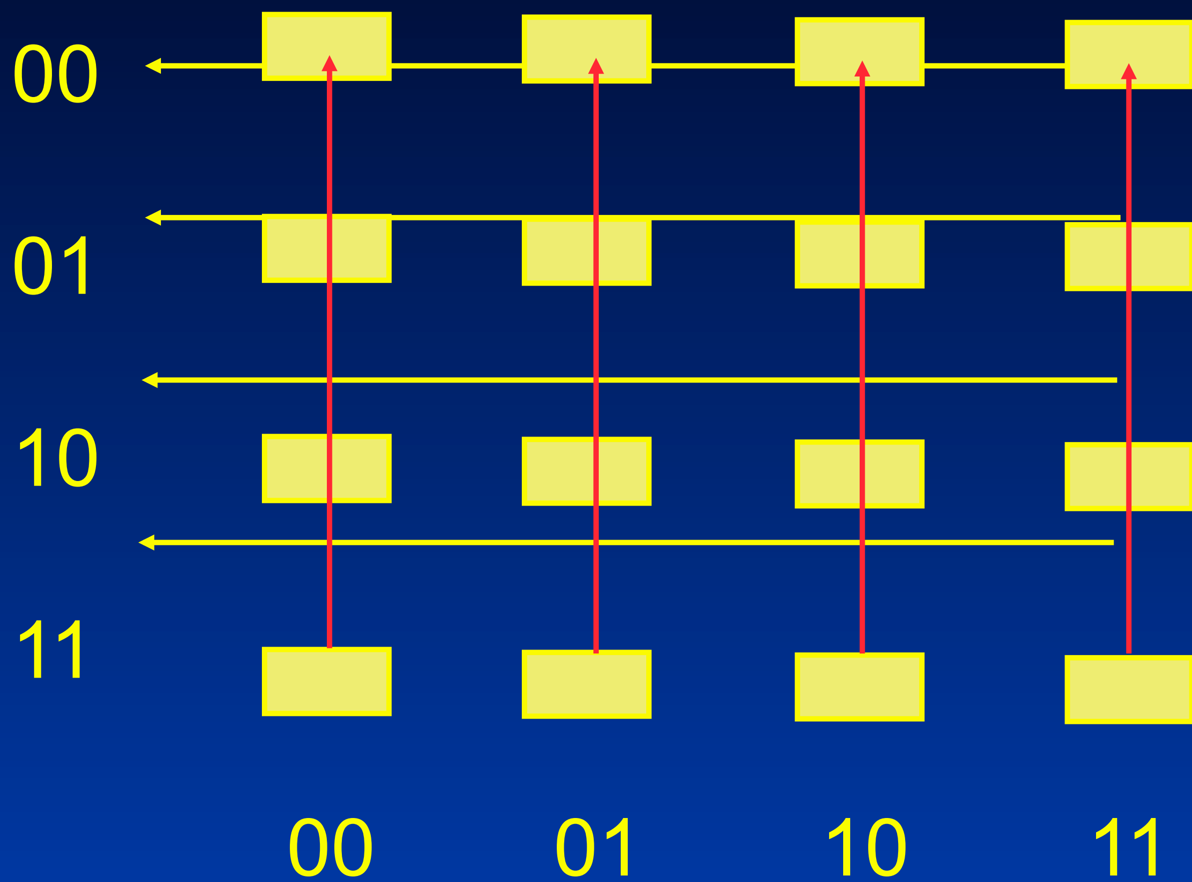
Fat-tree



(b) A 128-way fat tree.

4x4 cross-bar switch

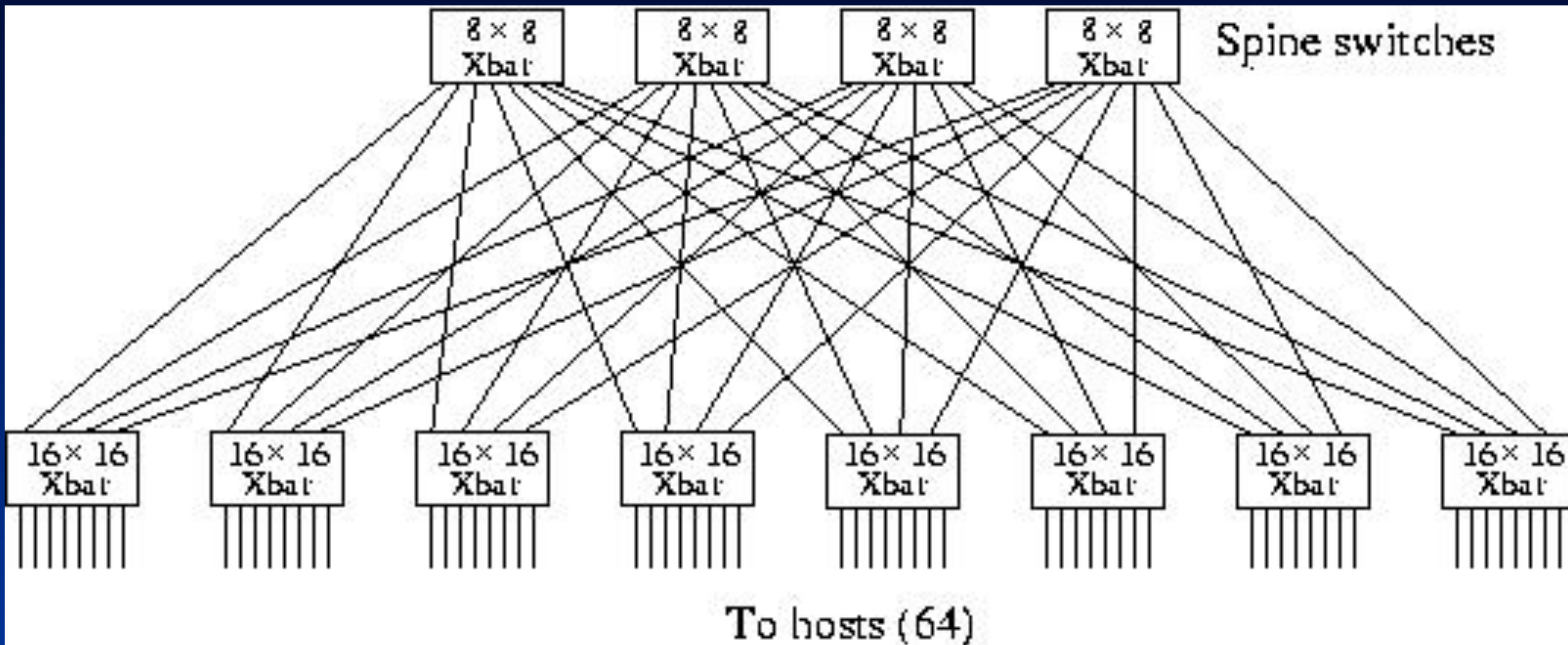
- Array of switches
  - One switch per connection
- All connections are direct point-to-point
  - Fast, but expensive



CLOS network

- Crossbar switches can be joined to form a CLOS network to increase capacity.

eg. Myrinet



relative performance

- Dynamic INs usually outperform a bus - messages may be sent in parallel.