

# Message Passing Architectures

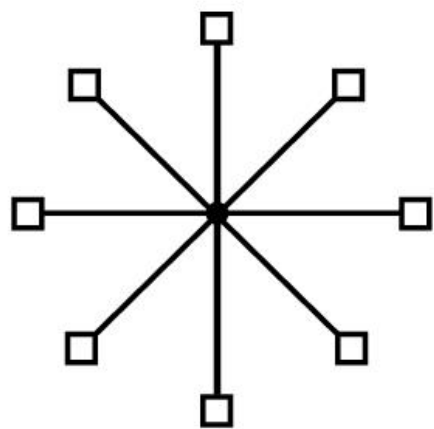
Ganesh Bikshandi

# Message Passing Architectures

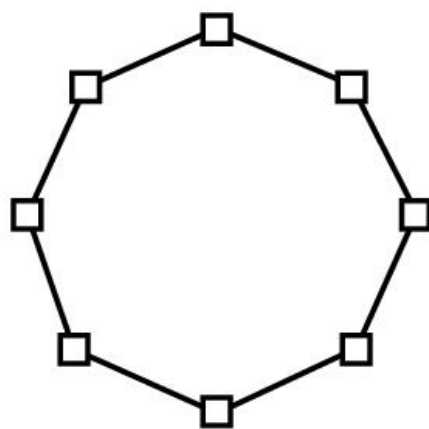
- Connected CPUs with Private Memory
- Treat network as a fast I/O Device
- Communicate via send/recv protocol
- (Network latency, bandwidth) control performance

# Network Topologies

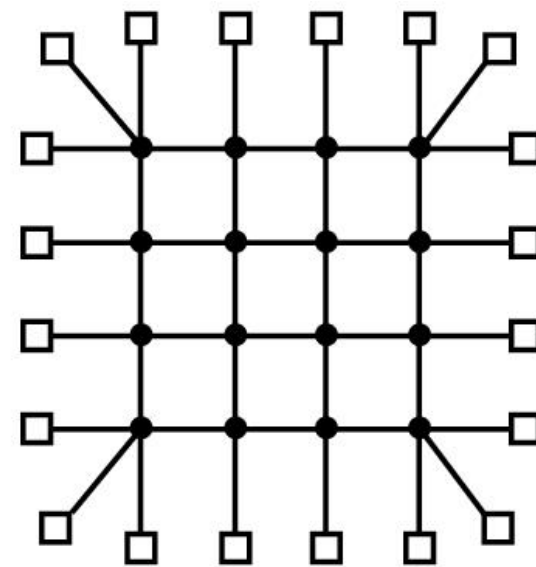
- Ring
- Double Torus
- Hypercube
  - Most preferred
  - Diameter grows linearly



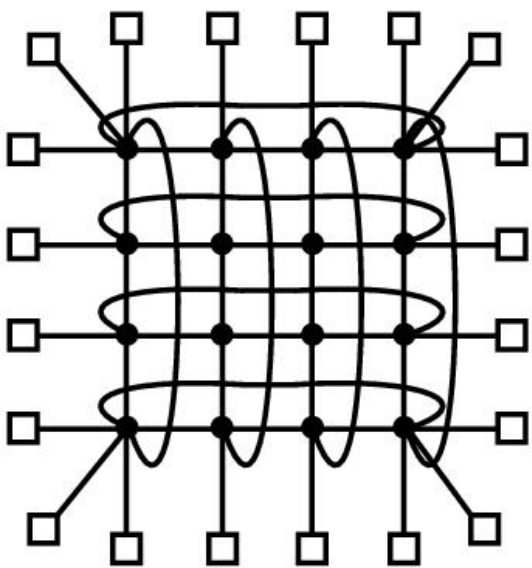
(a)



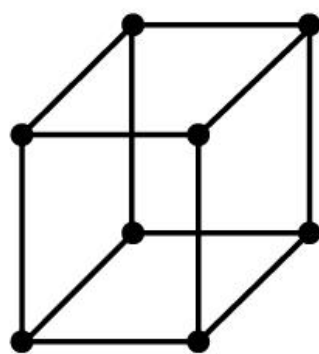
(b)



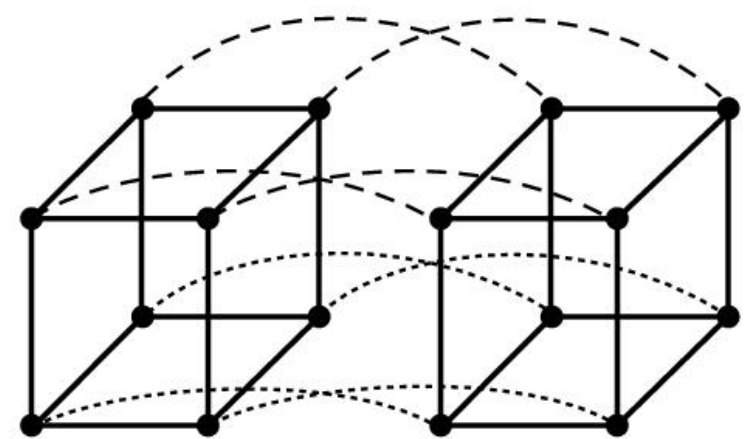
(c)



(d)



(e)



(f)

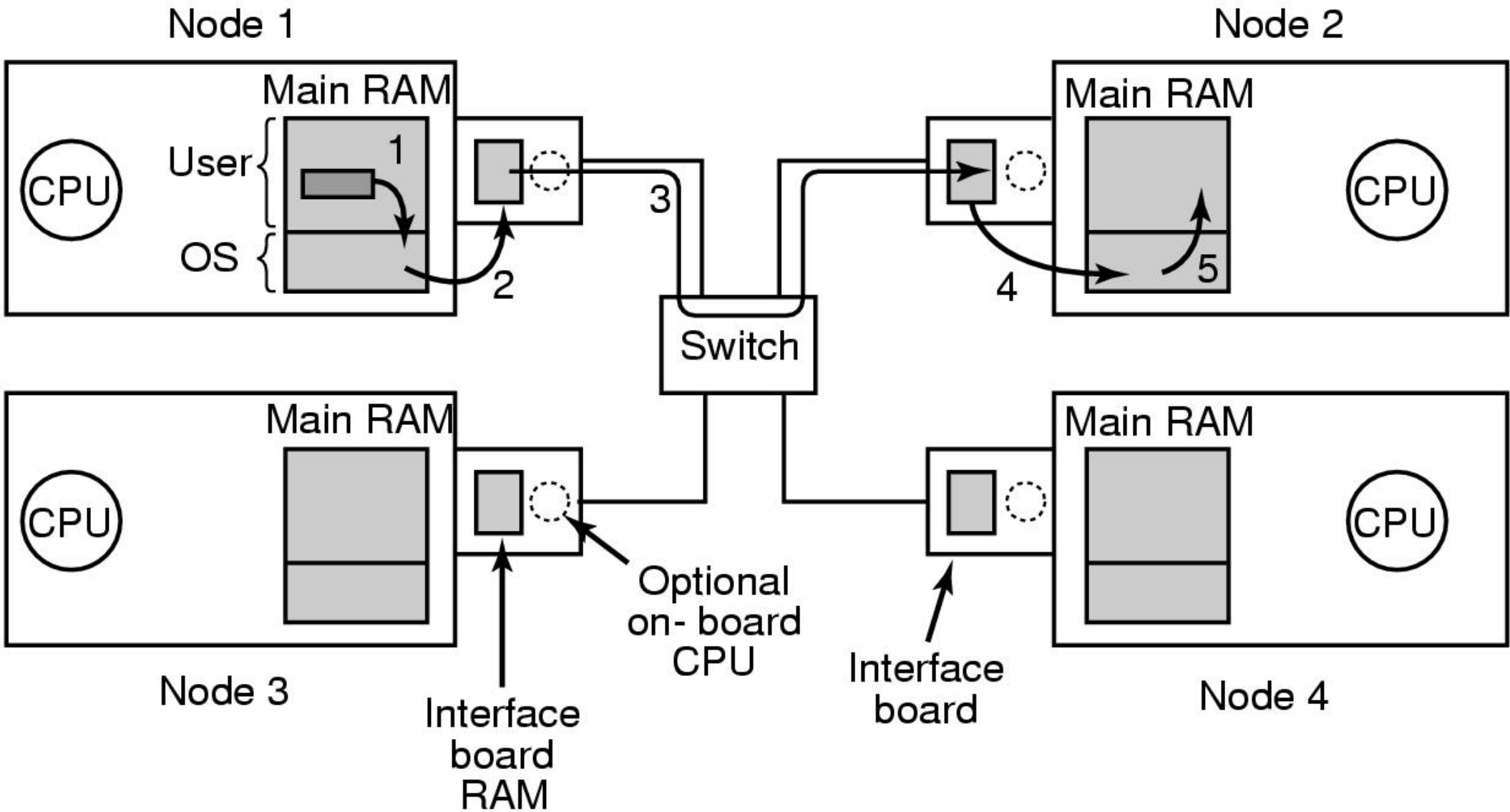
# Routing

- Packet switching
  - Copying in the intermediate destinations
- Circuit switching
  - Dedicated path (setup cost)
- Wormhole Routing

# Goals of the talk

- Bypassing OS layers
  - Avoid expensive system calls
  - Avoid copies
- Fast Communication
  - Between IB and CPU

# Communication Sequence



# Network Interface

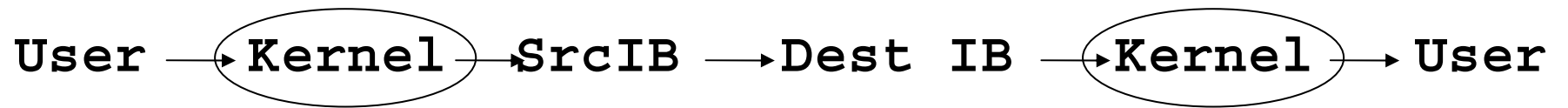
- A dedicated RAM
  - To maintain a continuous flow
- DMA channels
  - High speed data transfer
- Coprocessor
  - Perform some the the CPU's work
  - Synchronize with CPU
  - Operate a lower speed

# Low-Level Communication Software

- Key Performance Issue
  - Packet Copying

**User → Kernel → SrcIB → Dest IB → Kernel → User**

# Map IB to User Space



# Issues

- How to share among multiple processors?
  - Race conditions
- How to protect kernel?
  - Two interface boards (kernel traffic, user traffic)

# Node - IB Communication

- DMA from CPU RAM to IB
- User – initiated DMA
  - System call to translate virtual to physical address
- Kernel might replace a page while transfer
  - Pin the memory pages for each packet
    - Expensive system call

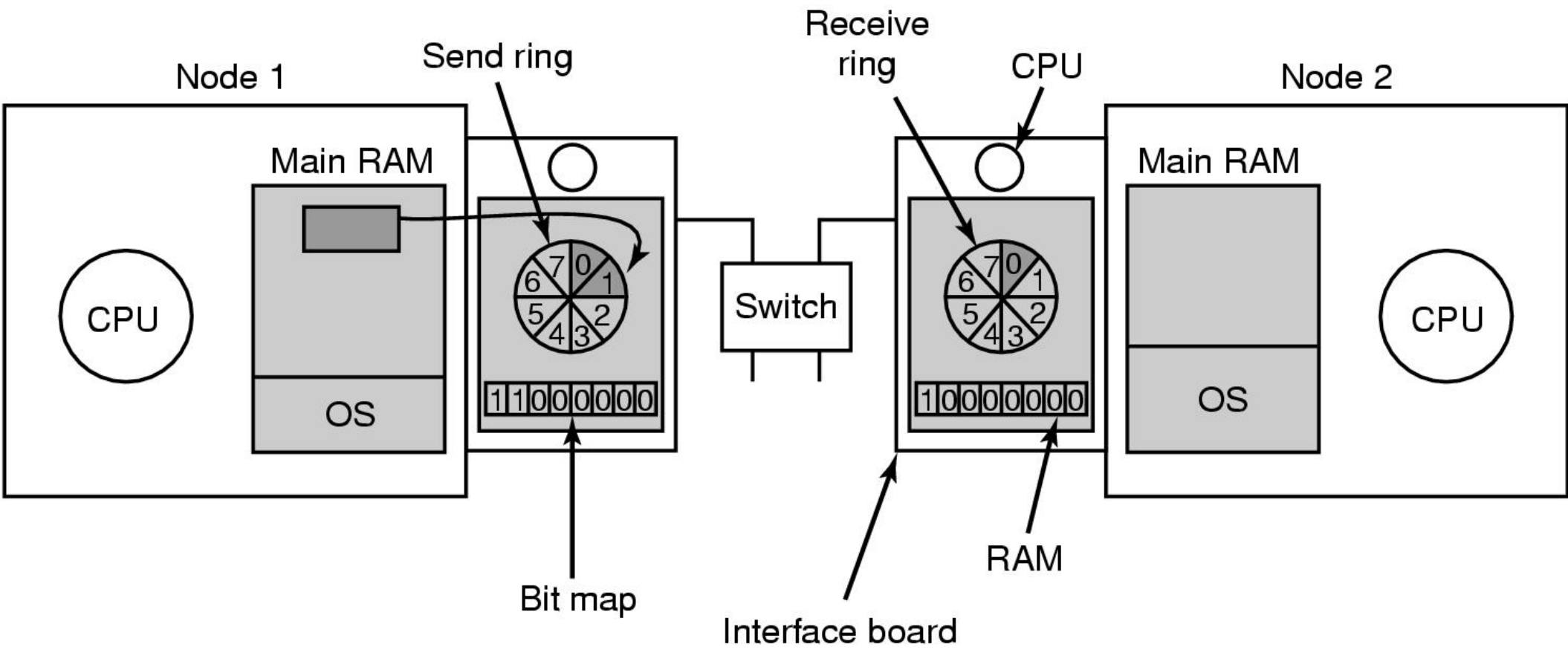
# Pinning

- Pin one page at startup
  - Copy outgoing packets to the page
  - Copy overhead (back to the original problem)
- Large packets
  - DMA with pinning
- Small packets
  - Programmed I/O

# Coprocessor

- Some IB have their own CPUs
  - eg. Myrinet
- Operate at lower frequency
  - CPU  $\longleftrightarrow$  Co-Processor bottleneck
  - Optimized Programs for Co-Processor
  - Synchronization (ring buffer)

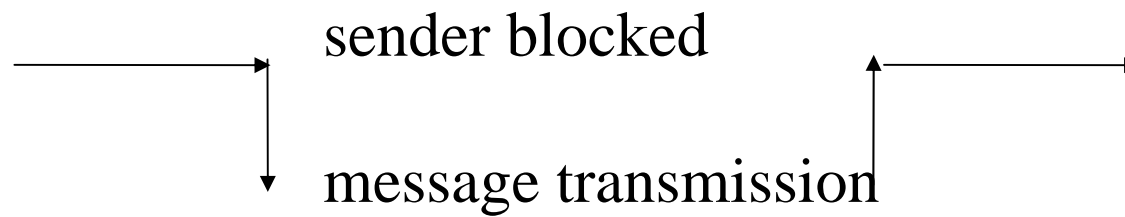
# Ring Buffer



# User Level Communication Software

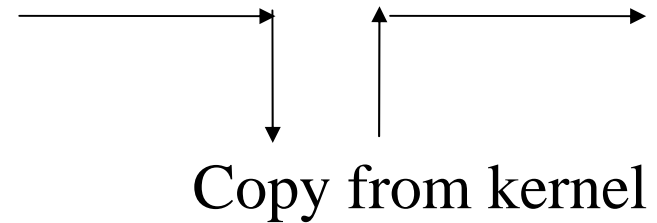
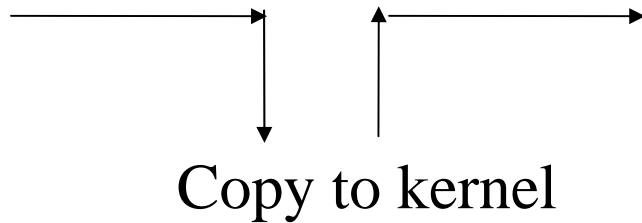
- Send / Receive
  - Blocking Send / Receive
  - Non Blocking Send / Receive
  - Non Blocking Send / Receive with Interrupts

# Blocked Send/Recv



No Overlap of Computation & Communication

# Non Blocking Send/Recv



Good overlap (useful in neighbor communication)

Copying is done to avoid overwriting send buffer

Copying is expensive

Alternative : Interrupt, making programming difficult 18

# User Level Communication Software

- Send / Receive
  - Blocking Send / Receive
    - Idle CPU
  - Non Blocking Send / Receive
    - Copy overhead
  - Non Blocking Send / Receive with Interrupts
    - Difficulty in programming

**Non Blocking Send / Blocking Receive**

# Active Messages

- Extend the idea of non-blocking with interrupts
- Message contains the interrupt handler address
  - Handler runs in the user space at the destination
- Handler - short and finish to completion
  - Recover the message
  - Inject it into the ongoing computation

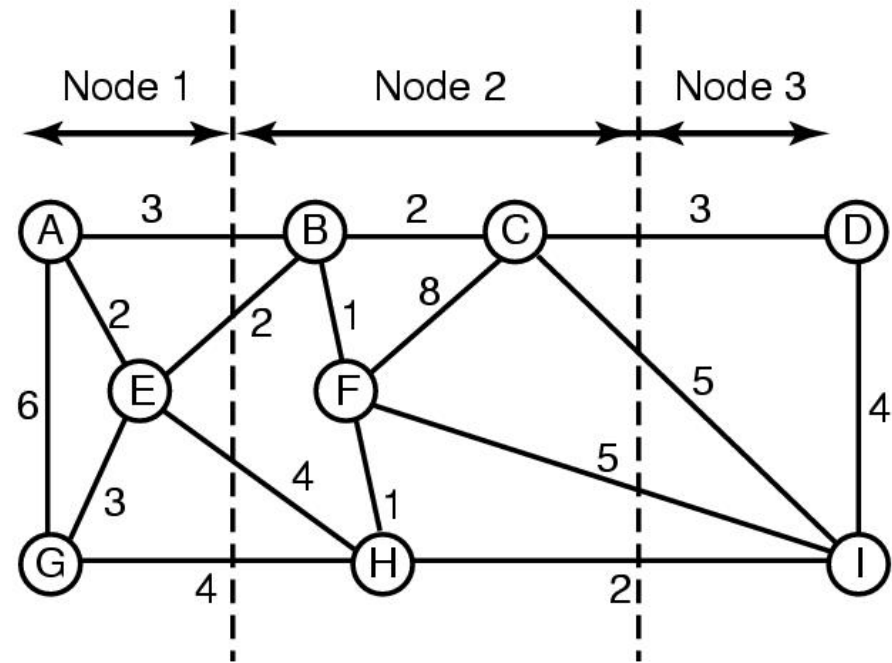
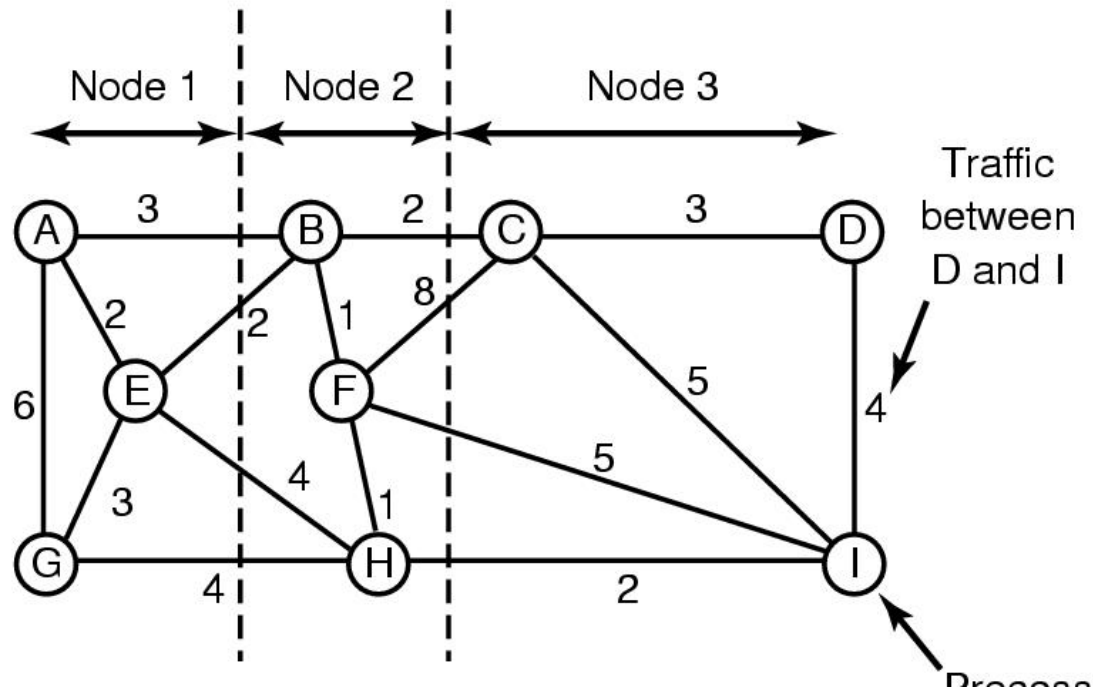
# Load Balancing

- Minimize wasted CPU cycles
- Minimize network traffic

# Load Balancing Algorithms

- Graph Theoretic Algorithms
  - Model process and communication as graph
  - Find a *k-disjoint* sub graphs
  - Map them to physical nodes
    - Minimize the network

# Load Balancing



# Process Migration

An overloaded node sends new process to an underloaded node (sender initiated)

An underloaded node requests new process from an overloaded node (receiver initiated)

# Processor Virtualization

- $V$  – set of virtual processors
- $P$  – set of physical nodes
  - $V \gg P$
- AMPI (Kale et al)

# AMPI Goals

- Adaptive overlap
- Automatic load balancing
- Asynchronous collective operations
- Automatic Checkpointing
- Better Cache performance

# Conclusions

- Message passing is an active area of research
- Key performance factors
  - Optimizing the communication overhead
  - Bypassing OS layers
  - Load balancing
- Smart group communications (not covered)
  - AlltoAll, Alltoone, reduction etc

# References

- High Performance Messaging on Workstations: Illinois Fast Messages for Myrinet, Scott Pakin et al.
- Active Messages: A Mechanism for Integrated Communication and Computation, Thorsten von Eicken et al.
- Adaptive MPI, Choa Huang et al