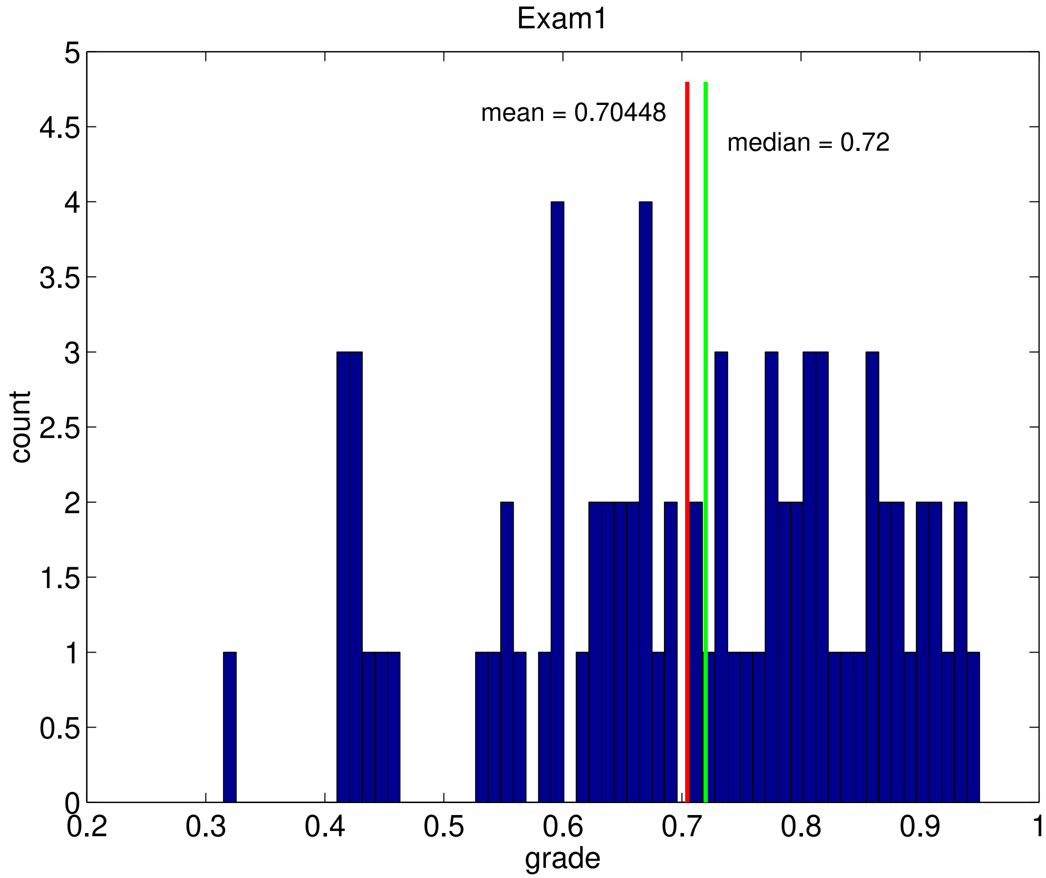
Announcements/Reminders

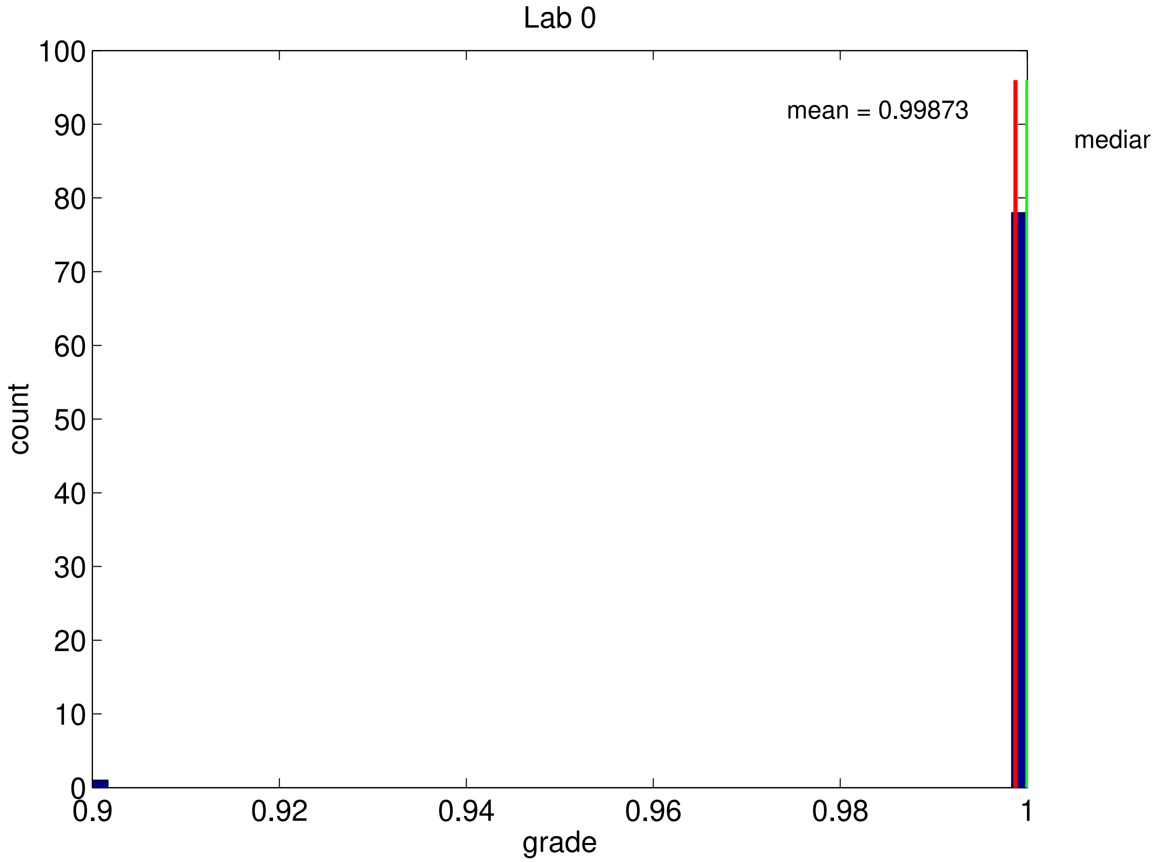
- Exams are available upon request
- Lab 3 proposal comments are back (look in ~/lab3/GRADE.txt file)
- Lab 2 due tonight
- HW 3 due Friday
- Wednesday is the last day to drop (Check your EdLab email for possible recommendations in this regard)

Exam Grades

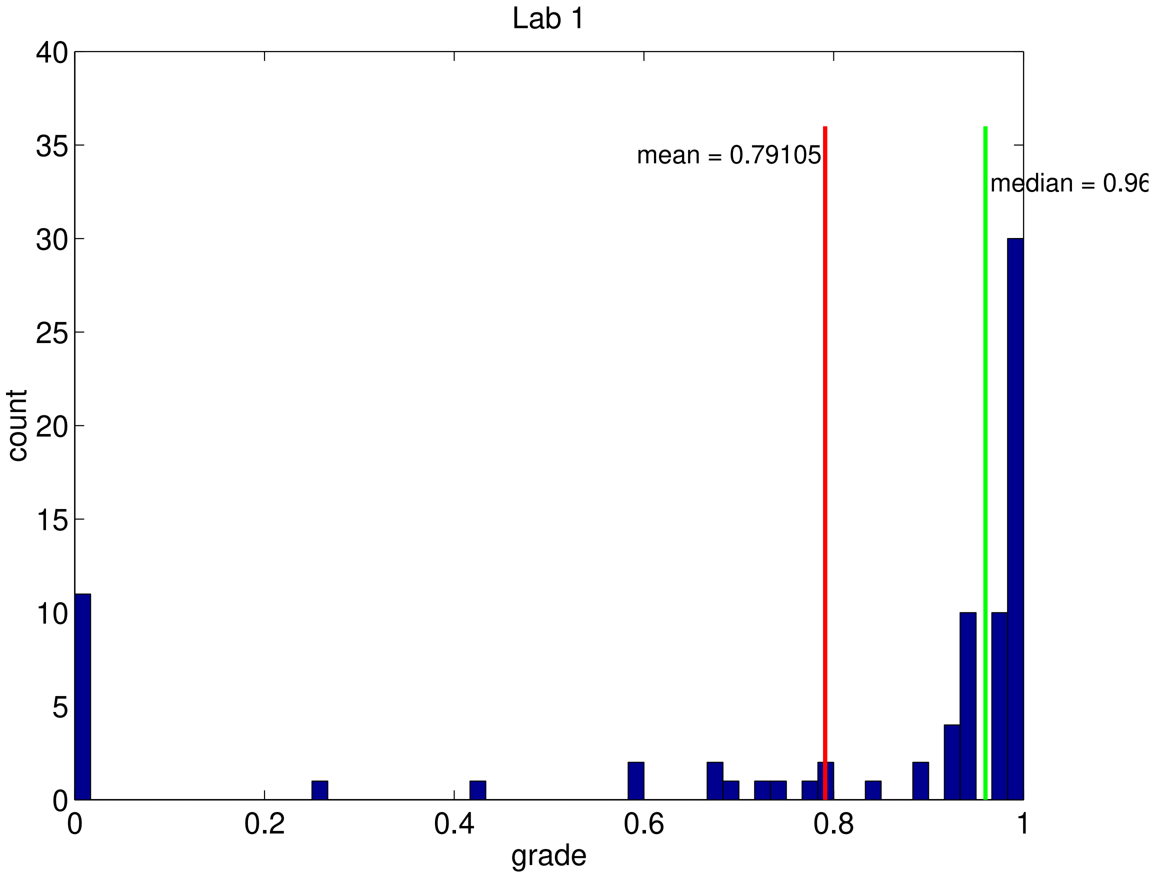


Standard deviation = 0.155

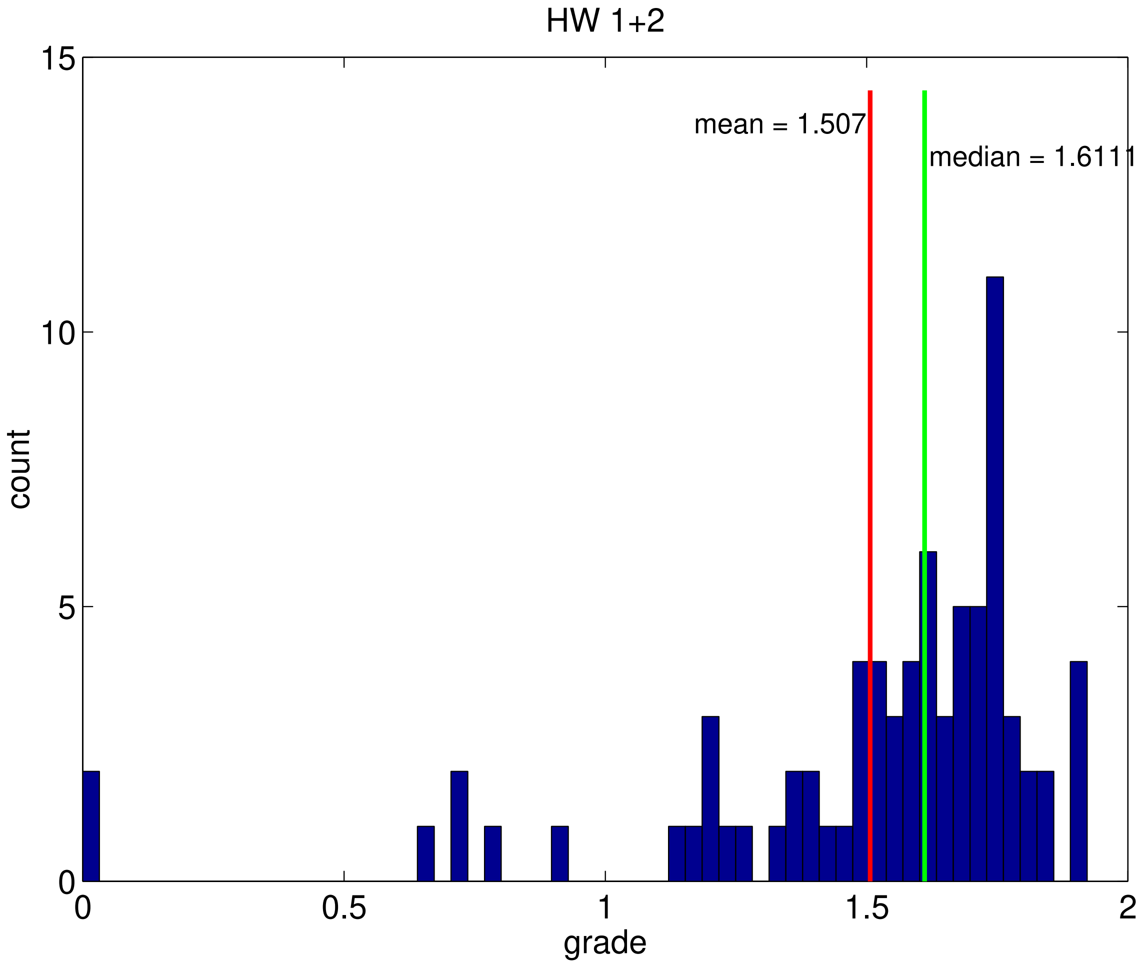
Lab 0 Grades


Lab 0
100
90
80
70
60
50
40
30
20
10
0
count
0.9
0.92
0.94
0.96
0.98
1
grade
mean = 0.99873
mediar

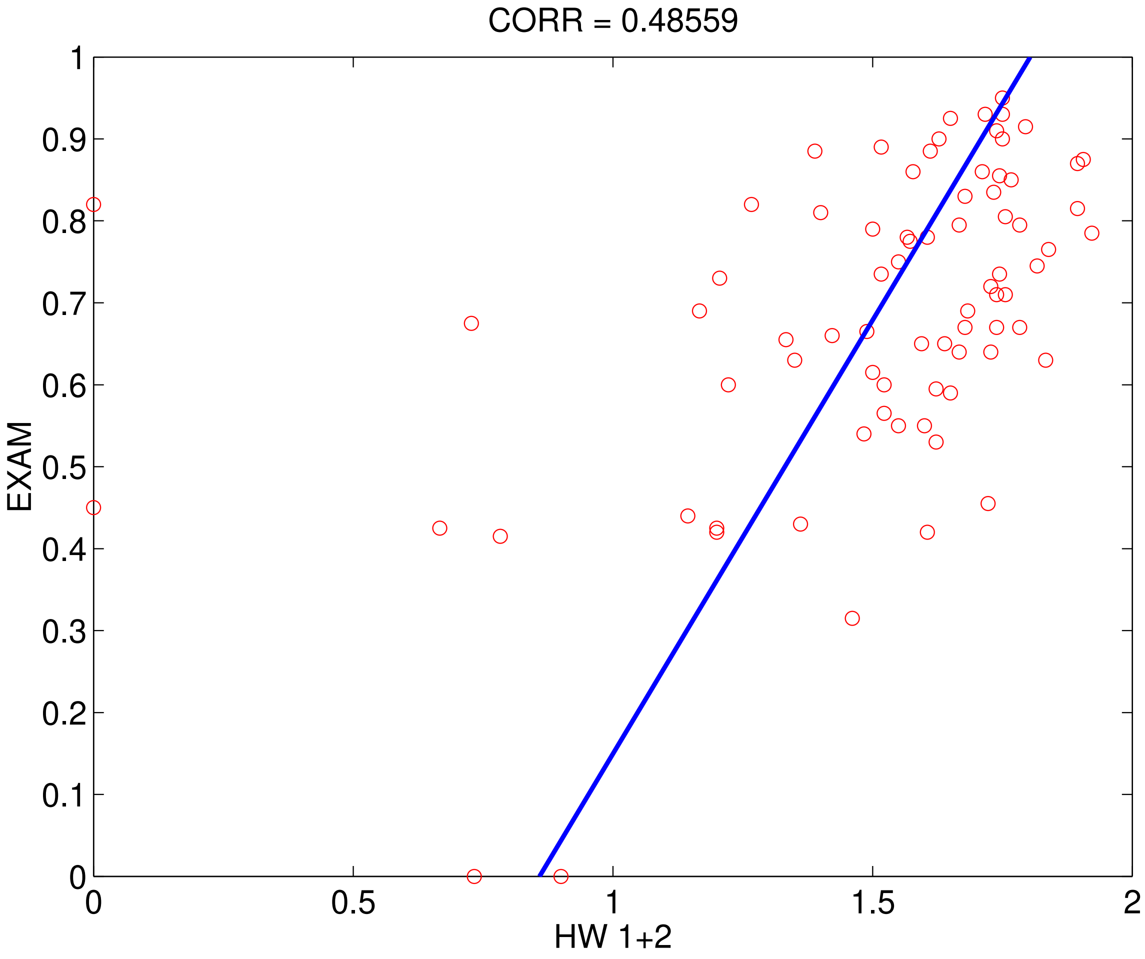
Lab 1 Grades


Lab 1
mean = 0.79105
median = 0.96
count
grade
0
5
10
15
20
25
30
35
40
0
0.2
0.4
0.6
0.8
1

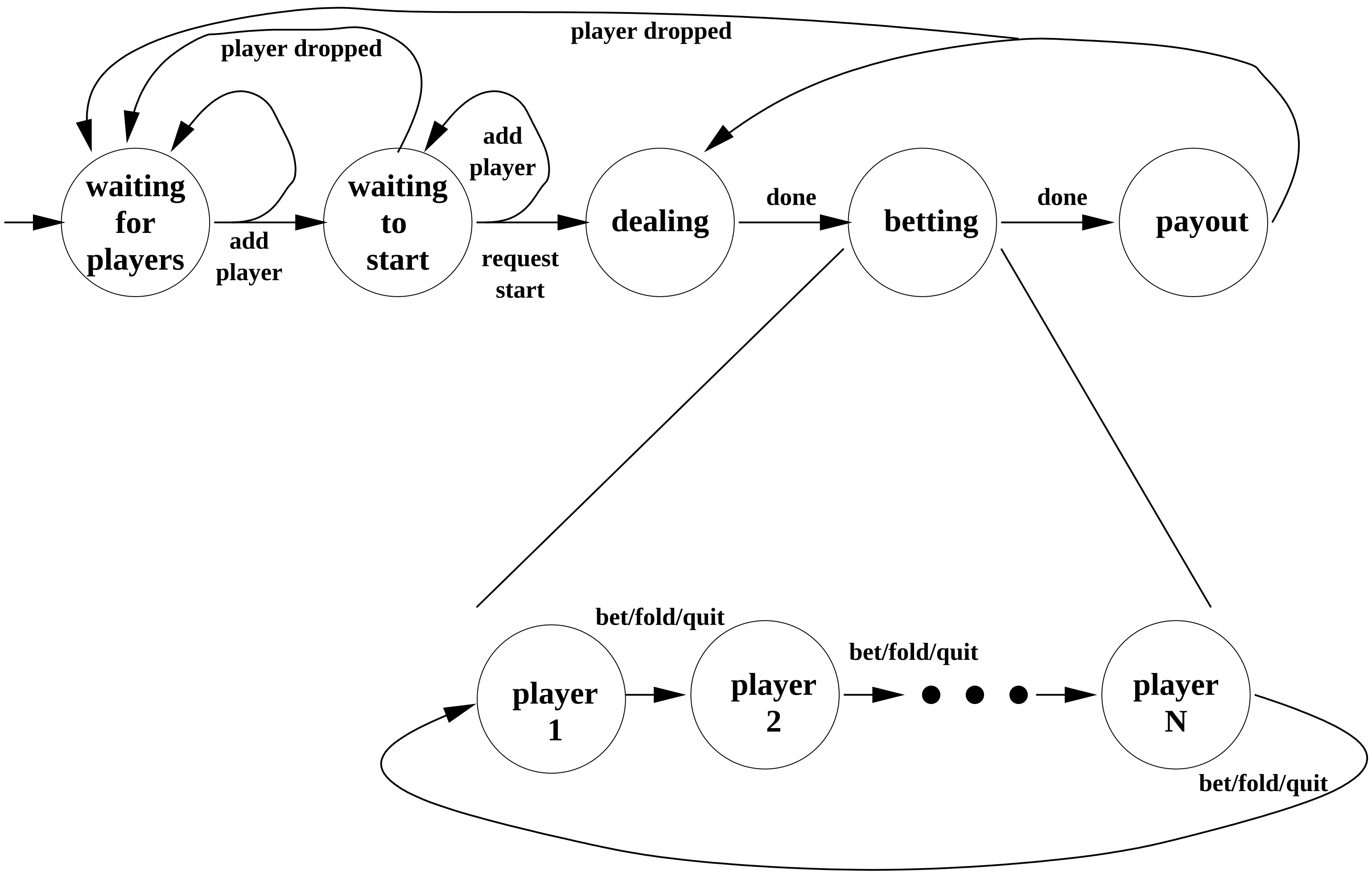
Homework Grades



Homework Grades Predict Exam Grade



Distributed Programming with Events: The Poker Example

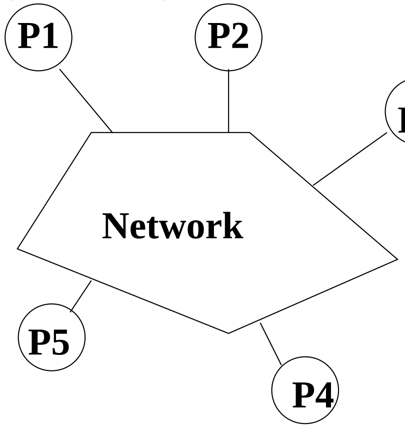


Lab 3 Hints

- RMI calls can return arbitrary objects (just as any method call can). However, if the object is not of a primitive class, the class must implement the 'Serializable' interface (just declare this in the class definition and you are all set).
- In many cases, your server will have to communicate information back to the clients. There are at least two ways to do this:
  - Have the clients check continuously for new messages (see the ReturnMessage example that we give for our poker player).
  - Make the main client object also extend the UnicastRemoteObject (but you should not have to bind the client object to an RMI server).
- Reminder: all work must be your own (the only exception is the example code that we provide in the class).

Distributed Systems

- **Distributed system:** a set of physically separate processors connected by one or more communication links



- Nearly all systems today are distributed in some way.
  - Email, file servers, network printers, remote backup, world wide web

Parallel versus Distributed Systems

- **Tightly-coupled systems:** "parallel processing"
  - Processors share clock, memory, and run one OS
  - Frequent communication
- **Loosely-coupled systems:** "distributed computing"
  - Each processor has its own memory
  - Each processor runs an independent OS
  - Communication should be less frequent

Advantages of Distributed Systems

- **Resource sharing:**
  - Resources need not be replicated at each processor (for example, shared files)
  - Expensive (scarce) resources can be shared (for example, printers)
  - Each processor can present the same environment to the user (for example, by keeping files on a file server)

- **Computational speedup:**
  - n processors potentially gives you n times the computational power
  - Problems must be decomposable into subproblems
  - Coordination and communication between cooperating processes (synchronization, exchange of results) is needed.

Advantages of Distributed Systems

- **Reliability:**
  - Replication of resources yields fault tolerance.
  - For example, if one node crashes, the user can work on another.
  - Performance will degrade, but system remains operational.
  - However, if some component of the system is centralized, a single point of failure may result
  - **Example:** If an Edlab workstation crashes, you can use another workstation. If the file server crashes, none of the workstations are useful.
- **Communication:**
  - Users/processes on different systems can communicate.
  - For example, mail, transaction processing systems like airlines, and banks, WWW.

Distributed Operating Systems

- Manages distributed resources
- Looks to users like a centralized OS
  - Access remote resources as they do local ones
  - But, operate on multiple, independent processors
- Provides transparency
  - Location
  - Data/process migration
  - Concurrency control
  - Replication
  - Parallelism

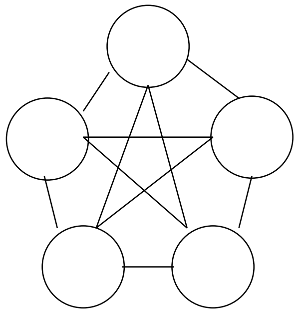
Distributed Systems

- What do we need to consider when building these systems?
  - Communication and networks
  - Security
  - Reliability
  - Performance and scalability
  - Programming models

Networks

- Networks are usually concerned with providing efficient, correct, and robust message passing between two separate nodes.
- **Local Area Network (LAN)** usually connects nodes in a single building and needs to be fast and reliable (for example, Ethernet).
  - **Media:** twisted-pair, coaxial cable, Cat5, fiber optics
  - **Typical bandwidth:** 10-100 Mb/s
- **Wide Area Network (WAN)** connects nodes across the state, country, or planet.
  - WANs are typically slower and less reliable than LAN (for example, Internet).
  - **Media:** telephone lines (T1 service), microwave links, satellite channels
  - **Typical bandwidth:** 1.544 Mb/s (T1), 45 Mb/s (T3), 1Gb (Internet-2)

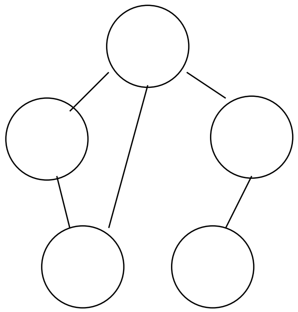
Point-to-Point Network Topologies



Fully Connected

- **Fully connected**: all nodes connected to all other nodes
  - Each message takes only a single "hop", i.e., goes directly to the destination without going through any other node
  - Failure of any one node does not affect communication between other nodes
  - Expensive, especially with lots of nodes, not practical for WANs

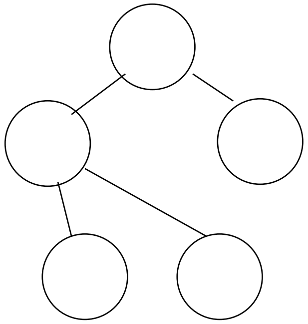
Point-to-Point Network Topologies



Partially Connected

- **Partially connected:** links between some, but not all nodes
  - Less expensive, but less tolerant to failures. A single failure can partition the network.
  - Sending a message to a node may have to go through several other nodes ⇒need routing algorithms.
  - WANs typically use this structure.

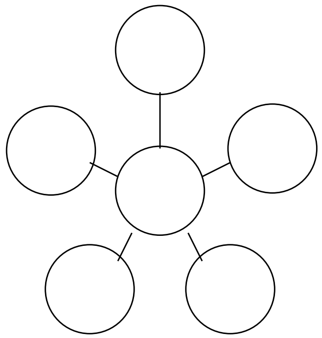
Point-to-Point Networks Topologies



Tree Structured

- **Tree structure:** network hierarchy
  - All messages between direct descendants are fast, but messages between "cousins" must go up to a common ancestor and then back down.
  - Some corporate networks use this topology, since it matches a hierarchical world view...
  - Not tolerant of failures. If any interior node fails, the network is partitioned.

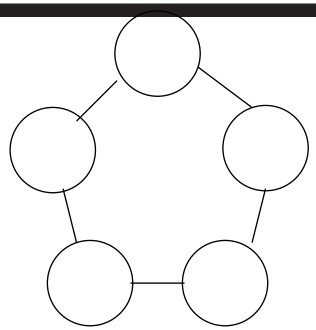
Point-to-Point Networks Topologies



Star

- **Star:** - all nodes connect to a single centralized node
  - The central site is generally dedicated to network traffic.
  - Each message takes only two hops.
  - If one piece of hardware fails, it can disconnect the entire network.
  - Inexpensive, and sometimes used for LANs

Ring Networks Topologies

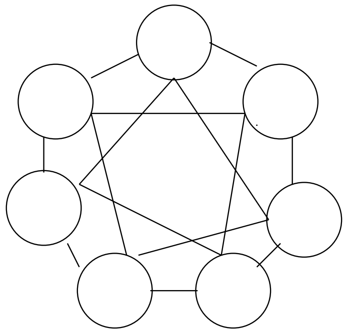


Ring

- **One directional ring** - nodes can only send in one direction.
  - Given n nodes, message may need to go $n - 1$ hops.
  - Inexpensive, but one failure partitions the network.

- **Bi-directional ring** - nodes can send in either direction.
  - With n nodes, a message needs to go at at most $n/2$ hops.
  - Inexpensive, tolerates a single failure by increasing message hops. Two failures partition the network.

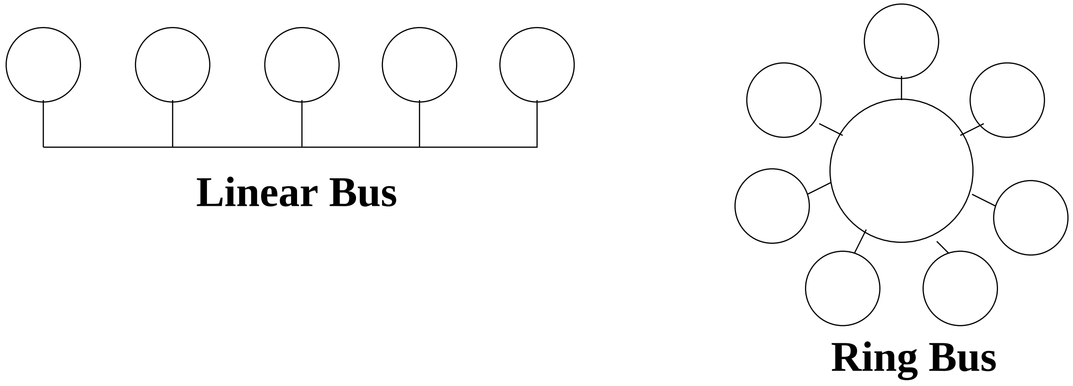
Ring Networks Topologies



Doubly Linked Ring

- **Doubly connected ring** nodes connected to neighbors and one away neighbors
  - A message takes at most $n/4$ hops.
  - More expensive, but more tolerant of failures.

Bus Network Topologies



- **Bus** - nodes connect to a common network
- **Linear bus** - single shared link
  - Nodes connect directly to each other using multiaccess bus technology.
  - Inexpensive (linear in the number of nodes) and tolerant of node failures.
  - Ethernet LAN use this structure.
- **Ring bus** - single shared circular link
  - Same technology and trade-offs as a linear bus.

Communication in Distributed Systems

Naming and Addressing:

- Must be ultimately be able to address a machine and a process
  - Typically locate an address through a Domain Name Server (DNS)
    * Servers are distributed and hierarchical
    * A local server cache is used to speed the lookup process
    * Secondary, back-up servers are used in case of a crash
  - Locate a local address through a broadcast request

Routing Strategies

Sending Messages Through the Network

- Routers: entity in the network responsible for moving messages between physical connections
- Routing table: representation of how a packet destined for a specific IP address should be routed
  - Often static
  - ... But can be dynamic (especially in mobile networks)
- Routing protocol: mechanism by which routing tables may be automatically updated as the network topology changes

Connection Strategies

How are network resources recruited during a dialog between two remotely-located processes?

- Circuit switching
  - Telephone networks
  - Resources required along the entire path (buffers and bandwidth) are reserved for the entire duration of the session
  - Guarantee transmission rate, but a potential waste of resources

Connection Strategies (cont)

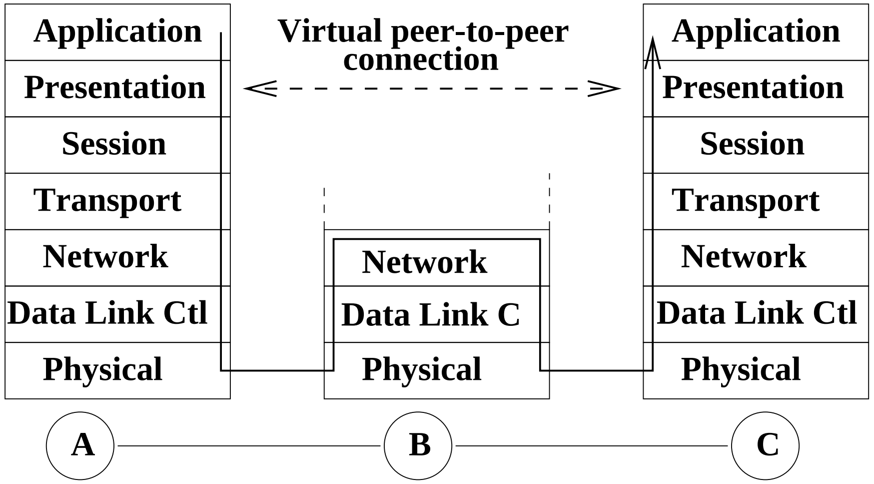
- Packet switching
  - Internet
  - Resources are not reserved, but instead are used on demand
  - Different packets of the same session may take different paths through the network
  - Congestion control, but no guarantees of transmission rate
- Message switching
  - ATM networks
  - An entire message is kept as a single packetized unit

Principles of Packet-Switched Network Communication

- Data sent into the network is chopped into "packets", the network's basic transmission unit.
- Packets are sent through the network.
- Computers at the switching points control the packet flow.
- **Analogy:** cars/road/police - packets/network/computer
- Shared resources can lead to contention (traffic jams).
- **Analogy:**
  - **Shared node** - Mullins Center
  - **Shared link** - bridge

Communication Protocols

- Protocol: a set of rules for communication that are agreed to by all parties
- Protocol stack : networking software is structured into layers
  - Each layer N, provides a service to layer N+1, by using its own layer N procedures and the interface to the N-1 layer.
  - Example: International Standards Organization/ Open Systems Interconnect (ISO/OSI)



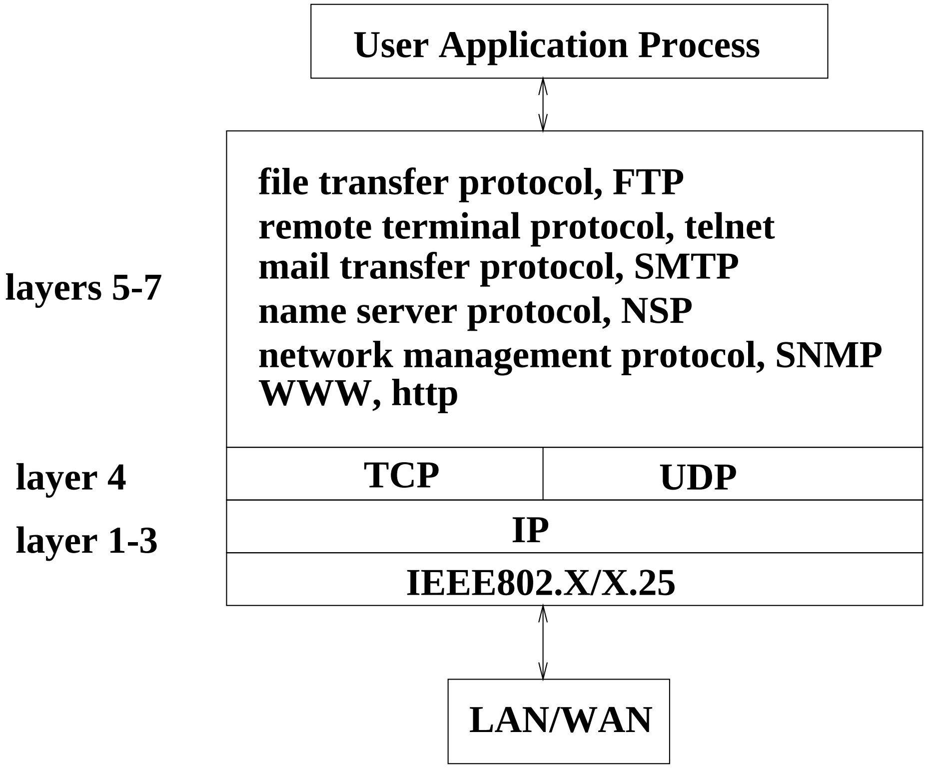
ISO Network Protocol Stack

- **Application layer:** applications that use the net, e.g., mail, netscape, X-services, ftp, telnet, provide a UI
- **Presentation layer:** data format conversion, e.g., big/little endian integer format)
- **Session layer:** implements the communication strategy, such as RPC. Provided by libraries.

ISO Network Protocol Stack (cont)

- **Transport layer:** reliable end-to-end communication between any set of nodes. Provided by OS.
- **Network layer:** routing and congestion control. Usually implemented in OS.
- **Data Link Control layer:** reliable point-to-point communication of packets over an unreliable channel. Sometimes implemented in hardware, sometimes in software (PPP).
- **Physical layer:** electrical/optical signaling across a "wire". Deals with timing issues. Implemented in hardware.

TCP/IP Protocol Stack



TCP/IP Protocol Stack (cont)

- Most Internet sites use TCP/IP - Transmission Control Protocol/Internet Protocol.
  - Has fewer layers than ISO to increase efficiency.
  - Consists of a suite of protocols: UDP, TCP, IP...
  - TCP is a **reliable** protocol – packets are received in the order they are sent
  - UDP (user datagram protocol) an **unreliable** protocol (no guarantee of delivery).

Packet

- Each message is chopped into packets.
  - Each packet contains all the information needed to recreate the original message.
  - For example, packets may arrive out of order and the destination node must be able to put them back into order.
  - Ethernet Packet Contents

bytes		
7	preamble - start of packet	fixed pattern so packet start is recognizable
1	start of frame delimiter	
6	destination address	
6	source address	
2	length of data section	
0-1500	data	
0-46	pad(optional)	packet must be > 63 bytes long
4	frame checksum	

  - The data segment of the packet contains headers for higher protocol layers and actual application data

Summary

- Virtually all computer systems contain distributed components
- Networks hook them together
- Networks make trade-offs between speed, reliability, and expense

Next Time

- Sockets and Remote Procedure Calls: Chapter 15