Computer Science 162 Lecture Notes

Gautam Wilkins Armen Khodaverdian

April 13, 2009

1 Announcements

Midterm 2 will be in 306 Soda Hall on Wednesday 4/15/09 at 7:00 PM. There will be a regular class that day as well, but the material covered will **not** be on Midterm 2 but will be on Midterm 3. The midterm is closed book, closed notes.

2 Lecture Material: Networks, Communication Protocols, and Distributed Systems

2.1 History

This subject came about in the 1960s, where users wanted to be able to send data between machines without physically transferring storage media. Today, there a very large number of small machines that need to share information.

There are getting to be more and more connected areas around the world. We want to be able to tie all areas together with networks, but in order to do so we need *protocols*. Protocols can be thought of as similar to languages; everyone speaks at least one language, and we need a common language in order to properly communicate.

The only thing that stops the world from looking like a single integrated computer system is performance. The speed of light is a fundamental limitation on the speed of data transfer, and thus, when sending messages, distance matters. Over long distances there is a performance lag, though this lag has been getting progressively shorter due to such things as the introduction of cables with Gigabit per second transfer rates and switches and circuits that operate on nanosecond timeframes. Thus, even when messages (such as email) are sent through dozens of routers and numerous cables, it still seems virtually instantaneous.

One of the first networks was the Advanced Research Projects Agency Network (ARPANET), which was developed by ARPA, a United States Government Agency. It was developed in the late 1960s and connected together a number of time sharing systems belonging to Defense Department contractors in order to facilitate communication. It was designed with resiliency and defense considerations in mind, and thus contained a good amount of redundancy. There were two basic pieces of hardware, routers and terminals. There were a fairly limited number of defense contractors with access to this network, and others found it desirable, which prompted the construction of other, similar systems. The fundamental protocols of ARPANET are very similar to the protocols employed by the current internet as we know it, which is rather impressive given how long ago ARPANET was developed.

The following are some other early networks worthy of mention:

- USNET: Required explicit routing for all transers. Cheap and easy to use.
- CSNET: A network supported by the National Science Foundation. It was designed to be a clone of ARPANET for university use.

- BITNET: A network that connected IBM mainframes. It was employed by a number of physics laboratories.
- VNET IBM: An internal network used by IBM. It was designed with security as a principal concern.
- DECNET: A network and series of protocols created by Digital Equipment Corporation (DEC) and used internally as well as sold as a product.

In the 1980s a large number of commercial networks started up. All of these separate networks were eventually linked and thus called the "internetwork" or internet.

2.1.1 Questions

1. *Did IBM's personal network have better security than the Defense Department's network?* Yes, it did. You shouldn't necessarily assume that government systems are always more secure than privately operated ones. Government systems have and continue to be exploited.[1]

2.2 Network Topologies

There are essentially two types of networks: fully connected (not really feasible for large networks) or partially connected networks where messages are sent through nodes. The simplest kinds of network topologies are the ring and the star. The star focuses on connecting everything to a central node, while the ring creates a loop around all the nodes. Such networks often rely on multiaccess bus/broadcast, such as the traditional Ethernet, which was just a single cable with links for each user.

2.3 Network Performance

When we consider network performance, there are many parameters to consider:

- Latency: the time to get 1 bit from end to end (determined by distance and speed of transfer medium).
- Setup Latency: time to transfer first bit from end to end.
- Transmission Latency: time to transfer subsequent bits from end to end (almost always shorter than setup latency).
- Bandwidth: how many bits/sec are transferred from end to end. If we wanted to transfer a very large amount of data it may make more sense to package hard drives and ship them to their destination.
- Cost: how much will it cost to transfer the data.

[1] *Annecdote:* Livermore laboratory has high security and requires an ID badge in order to enter. A guard physically examines the ID before allowing the owner in. About 20 years ago, someone pasted a picture of an ape over his ID badge and went in and out for a week without anyone noticing. When he finally reported this lapse to security, he was fired.

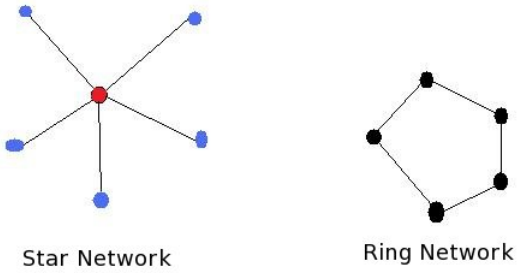

Figure 1: Topologies of Ring and Star Networks

2.4 Network Protocols

It is very easy to invent a network, but everyone must agree on its protocols in order to ensure reliable communication. For example, if you went to Irkutsk (in Siberia), the odds of people speaking English are probably low. If you cannot speak Russian then you will not be able to communicate with anyone. When creating networks a lot of time is spent trying to come to an agreement on protocols. Recommended reading is section 15.6 in the textbook to see specific network protocols.

Protocols all have layers. The lowest layer is electrical and handles the physical transmission of data. The upper layers consist of software. Each layer has an interface to the layers directly above and below it and in end to end communication, each layer of the sender talks to the corresponding layer of the receiver (i.e. electrical layer talks to electrical layer, application layer talks to application layer).

ISO Protocols have the following layers, listed in order from lowest to highest:

- Physical: Lowest layer. Handles voltages, delay, current, and error correction at the byte level.
- Data Link: Get packets between two connected components. Includes low level error correction.
- Network: Get packets from source to destination by routing through other machines.
- Transport: Low level access to the network. Keeps packets in order, controls transmission rate, generates physical address, takes care of retransmission of lost packets.

- Session: Process to process protocols.
- Presentation: Handles differences between sites and formats.
- Application: User interface to network.

The physical, data link, network link, and transport layers are all well defined, but the session, presentation, and application layers are more "fuzzy." They are not strictly defined the way the lower levels are.

A lot of original computer science technology was developed in the U.S., but networking has been heavily done in Europe. With a lot of borders, different currencies, etc. European nations have a great deal of experience in setting standards and protocols.

2.4.1 Questions

1. *For the transport layer, aren't there specific protocols, like UDP?* Yes. The transport layer is an abstraction, and protocols like TCP/IP and UDP are specific instances of the transport layer.

2.5 Network Types

There are two main types of networks, Local Area Networks (LAN) and Wide Area Networks (WAN). LAN systems are in the same geographic area, whereas WAN systems are geographically distributed. WANs are interconnected, made up of LANs that are broadcast networks (every node communicates directly with every other).

The simplest scheme for broadcast networks is pure broadcast, where all messages are sent to every node in the network (much like radio broadcasts). This is the same as the original version of Ethernet, where "ether" refers to the shared communication medium between the nodes (generally a cable).

An example of such a network is ALOHAnet, a satellite based network that was implemented in Hawaii. The protocol for ALOHA is relatively simple. Each device sends a transmission to the satellite, which receives the message and then retransmits it to all other devices. The sending device listens for its own message, and if it comes back properly, the it has been successfully transmitted. If the message comes back garbled, that means another device was transmitting a message at the same time (this is known as a collision). If a collision occurred then the sending device must retransmit its message. This system, however, is inherently unstable. As more senders send more messages, the probability of collisions increases. As collisions occur, there becomes a backlog of messages that must be resent, which further increases the number of messages being sent. The end result is that for large levels of traffic, nothing can be transmitted. This problem can be easily analyzed and the maximum utilization of such a system is around 18% of the theoretical bandwidth.

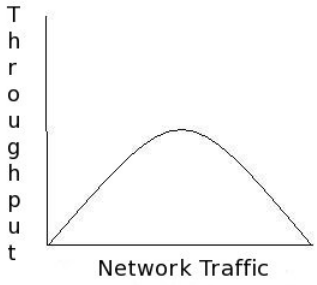

Figure 2: Plot demonstrating failure of AlohaNet under high traffic

A better protocol is SlottedALOHA. Since any overlap between transmitted messages results in both being garbled SlottedALOHA makes it so that two messages either completely overlap or do not overlap at all. To ensure this SlottedALOHA employs a clock. Now instead of simply transmitting at any time, transmissions are only allowed between clock ticks. So a message transmission starts on a clock tick and must end before the next clock tick. Thus, if two messages collide, they will both have been sent in the same slot. A mathematical analysis on the SlottedALOHA shows that the maximum attainable bandwidth is 36% of the theoretical maximum.

2.5.1 Questions

1. *Do all senders use the same clock?* Yes, the simplest solution is to have the satellite broadcast the clock signal to all the senders.

2. *Do we waste time at the edges of the clock? Do all messages have to be the same legnth?* The messages are not strictly required to fill the entire slot, but each transmission can take no longer to transmit than the interval between clock ticks. The simplest way to deal with long messages is to break them up into packets and send them one at a time.

3. *How do senders know if the satellite is being overloaded?* You can track load by checking to see if your message was retransmitted properly. If it comes back garbled then a collision occurred. If a large percentage of your transmissions collide then the satellite is probably overloaded.

2.6 Ethernet

If the communication medium does not involve satellite transmission, then better methods can be used. In the original Ethernet protocol all users shared a single cable (maximum length of 4000 ft. for collision detection considerations). Users connect to the cable through a pin stuck in the center connecter.

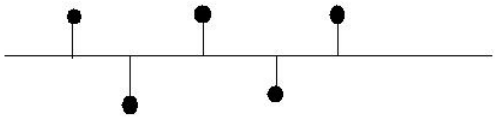

Figure 3: Ethernet cable with multiple connecting nodes

Ethernet employs Carrier Sense, where each sender monitors the channel before sending a transmission. If there is another message being transmitted, then the sender waits before sending its message. Carrier Sense alone, however, is not enough to ensure that collisions will not happen. If two senders simultaneously send a message then a collision will occur even with Carrier Sense. To deal with this, Ethernet employs Carrier Sense Multiple Access with Collision Detection (CSMA/CD). With Collision Detection a sender also monitors the channel while sending a message. If it detects another message being transmitted at the same time, it immediately stops transmitting its message (to decrease wasted bandwidth) and then waits a random amount of time before retransmitting. As the number of collisions increases the amount of time a sender waits before retransmitting increases to help relieve network congestion. Why would you want to wait for a random amount of time instead of a for a fixed amount?

The original Ethernet protocol has some problems, namely:

- Reliability: A user can either maliciously or unintentionally jam a network.
- Fairness: A user that constantly sends transmissions will get an unfair share of bandwidth.
- Bandwidth: Original cables limited transmission rate to 10 Mbps.
- Security: Everyone on the network can see all transmissions.

Current Ethernet technology uses switches and hubs to help control collision and deal with signal degradation. Also, there are currently cables that transmit at rates of 100 Mbps and 1Gbps.[2]. Wireless Ethernet works with protocols that are similar to Wired Ethernet, though with some modifications.

2.7 Ring Networks

As mentioned earlier, Ring Networks are a type of broadcast network. All the nodes are connected in a ring, and all messages are passed around the ring in a circular fashion. To avoid collisions this method relies on a token that is passed around to each node on the ring. When a node wishes to send a message it waits until it acquires the token, then keeps it and transmits its message. When the message is retransmitted to the original sender it has passed to all the other nodes as well, so the sender releases the token and passes it on to the next node.

This type of broadcast network is a very poor one in practice. It creates two main central points of failure. First is that if a single node or cable dies, then the whole network goes down as well. The second is the token itself. If the token vanishes or is corrupted then the whole network fails. Also, if a node does not ever pass on the token, then no other node in the network can transmit data. The dilemma becomes even worse if the token is accidentally cloned, resulting in two nodes transmitting at the same time and causing collisions.

In the past IBM used Ring Networks for a number of years, but have long since discontinued their use.

2.8 Linking Two Machines

There are three methods by which we can establish a link between two separate machines:

1. Circuit Switching: This method is similar to the way telephone calls are handled. A physical circuit is created between the sender and receiver on which only their data is allowed to travel.

2. Packet Switching: The message is broken up into packets, sent piece by piece, and reassembled at the end. Packet Switching does not have a set route for all the packets to take, so different packets can travel on different routes. Using Packet Switching creates a virtual circuit (as in VoIP) where the sending and receiving machines have the illusion of a circuit connecting them.

3. Message Switching: Message Switching creates a virtual circuit between two nodes for long enough to send a complete message and then drops the circuit.

[2] As a note regarding higher cable bandwidth, few PC network cards can consistently push data at a rate of 1 Gbps, so often cables with such a transfer rate are not particularly more useful than their 100 Mbps counterparts

When using Packet Switching the packets must be forwarded from machine to machine in order to get from one sender to another, and often times must also be sent between networks. We call machines that transfer packets between networks *gateways*.

2.9 Names, Addresses, and Routes

The name is a symbolic name for something, such as "Google" or "Amazon." The address refers to the location of the object in question, usually consisting of a network, a number of a site on the network, and the ID of a host. The route is the actual path through networks and machines that a packet takes in order to reach its destination.

Sometimes the sender may have to specify the route, but this is inconvenient for the user, as well as potentially slow. In practice, routing is done by machines on the internet.[3] Essentially, every machine knows how to reach the nearest gateway, and the gateway knows how to reach other gateways in order to transmit a packet. Domain Name System (DNS) Servers manage the mappings from the name of a page (i.e. "http://www.google.com") to an IP address, which routers then use to determine the path the packet should take to reach its destination. If the name or IP address is not recognized, then the packet is dropped.

Networks are set up so that multiple routes are possible when trying to reach a destination. This means that if a single network goes down, packets that would normally be routed through it can instead take other routes to reach their destination.

2.10 Communication Problems

There are numerous potential problems that can arise while transmitting packets over a network. Packets can get lost or dropped either due to transmission errors or corruption, routers can become full (called network congestion, which results in packets being dropped), packets might be sent to a machine that is not operational, and so on.

Packets can arrive out of order, and datagram protocols attempt to deliver individual packets, but make no guarantees regarding whether the packets will actually arrive, or whether the packets will arrive in the order that they were sent.

This can be a problem for some applications, since they often want assurances regarding the delivery and order of arrival of the packets they send. The solution to this problem involves the transport layer protocol. First it establishes a connection between the sending and receiving machines using what is known as the Three Way Handshake. The sending machine first transmits a request for a connection to the receiving machine. The receiving machine then sends an acknowledgement (ack) to the sending machine. The sending machine

[3]The mechanisms of this will be discussed in later lectures

then sends an ack back to the sending machine. When the receiving machine receives the ack, the connection has been established. When the receiver gets any subsequent packets from the sender it always sends an ack back, and the sender retransmits any packets that it does not get an ack for. This ensures that all packets will be delivered to the sender.