CS 250: FOUNDATIONS OF COMPUTER SYSTEMS [NETWORKING]

A transmission tale that does not bite its tail
What is it that you send, pray tell?
If not ones and zeros
'tis but a myth, that we shall dispel
Why, what's sent are signals

Powered by modulation
To disambiguate ones from zeros
Alongside duplexity
So data may flow this way or the other

SHRIDEEP PALLICKARA
Computer Science
Colorado State University

1

Frequently asked questions from the previous class survey

- A recursive method is 10-levels deep, and has 2 local variables
  - How many stack frames are allocated? How many times are the variables allocated?

2

Topics covered in today's lecture

- The socket abstraction
- Data encoding formats
- Switched networks
- Multiplexing

3

4

Java provides a `ServerSocket` to enable writing servers

- `ServerSocket` runs on the server
  - **Listens** for *incoming* network connections on a particular **port** on the host that it runs on
- When a client socket on a remote host attempts to connect to that server port
  1. Server **wakes** up
  2. *Negotiates* a **connection** between the client and server
  3. **Opens** a regular `Socket` between the two hosts

Some more about the two types of sockets

- `ServerSocket`s **wait** for connections
- Client `Socket`s **initiate** connections
- Once the `ServerSocket` has set up the connection?
  - **Data always travels over the regular `Socket`**

Using the `ServerSocket`

- Created on a particular **port** using the `ServerSocket(port)` constructor
- Listen for communications on that port using `accept()`
  - **Blocks** *until* a client attempts to make connection
  - Returns a `Socket` object that **connects** the client to the server
- Use the `Socket`'s `getInputStream()` and `getOutputStream()` to communicate

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT NETWORKING L17.7

7

Creating the `ServerSocket`

- ```
  ServerSocket serverSocket =
        new ServerSocket(5000);
  ```
  - Tries to create a server socket on port 5000
- ```
  ServerSocket serverSocket =
        new ServerSocket(5000, 100);
  ```
  - Can hold up to 100 incoming connections
- ```
  ServerSocket serverSocket =
     new ServerSocket(5000, 100,
       InetAddress.getHostByName
           ("address2.cs.colostate.edu"));
  ```
  - On a **multi-homed** host, specify the network-address over which connections should be accepted

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT NETWORKING L17.8

8
  ```

Accepting network connections

```
ServerSocket serverSocket =
     new ServerSocket(portNum);
while(true) {
   Socket socket = serverSocket.accept();
   ...
}
```


Closing the client and server sockets

- Closing a `ServerSocket` **frees** a port on the host that it runs on
- Closing a `Socket` **breaks** the connection between the local and remote hosts

We exchange byte streams over the socket

- The `java.io` package contains the `DataInputStream` and `DataOutputStream` that lets you do this elegantly

- ```
  DataInputStream din =
    new DataInputStream(socket.getInputStream());
  ```

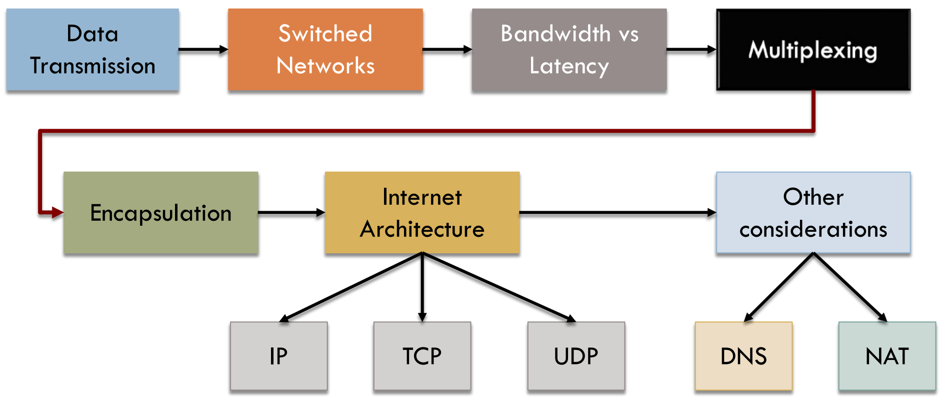
- ```
  DataOutputStream dout =
  new DataOutputStream(socket.getOutputStream());
  ```

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT NETWORKING L17.11

11

12

Communications & Networking:
Topics that we will cover

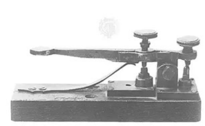


13

Some historical examples

Optical Telegraph

Invented By the Chappe Brothers in France
Circa: **1791**



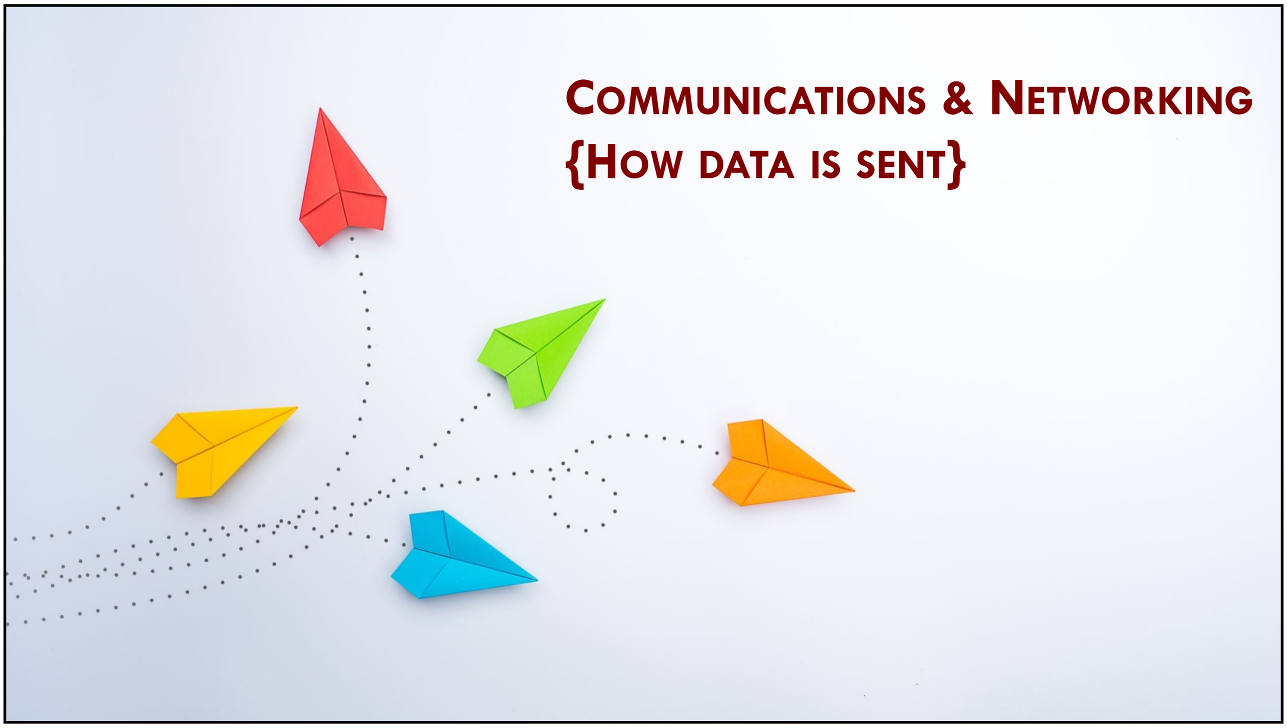
Key Type of an electrical Telegraph

1835: Morse Code invented by American Professor, Samuel Morse

1837: Two practical electric telegraphs appeared at almost the same time: British inventors William Cooke and Charles Wheatstone.

1848: The Associated Press was formed to pool Telegraph expenses

14



15

How is the data sent?

- Are we sending **1**'s and **0**'s?
- Whatever the physical medium, we use **signals**
  - Electromagnetic waves traveling at the speed of light
    - Speed of light is different in different mediums

COLORADO STATE UNIVERSITY | Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT | NETWORKING | L17.16

16

Components of encoding binary data in a signal

- Modulation
- Duplexity

Encoding binary data: Modulation

- Objective is to send a ***pair*** of **distinguishable** signals
- Vary frequency, amplitude, or phase of the signal to transmit information
  - E.g., vary the power (amplitude) of signal
  - $x(t) = A \sin(2\pi ft + \theta)$

Encoding binary data: Duplexity

- *How many* bit streams can be encoded on a link at a time?
  - If it is one: nodes must share access to link
- Can data flow in both **directions** at the same time?
  - Yes ➔ full-duplex
  - No ➔ half-duplex

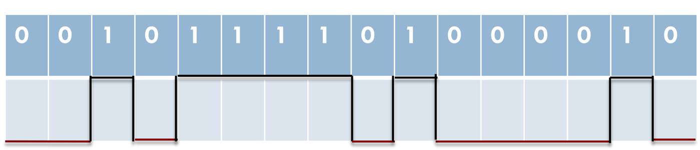
For our purposes, let's ignore details of modulation

- Assume we are working with two signals
  - High and low
- In practice:
  - Different voltages on a copper-based link
  - Different power-levels on an optical link

Let's do the obvious thing

- Map **1** to a high signal
- Map **0** to a low signal

Non-return to zero (NRZ)



Problems with NRZ because of consecutive 1's and 0's: BASELINE WANDER

- Receiver keeps **average** of the signal seen so far
- Average is used to *distinguish* between low and high
- Lots of consecutive **1**/**0**'s will make it difficult to detect a significant change

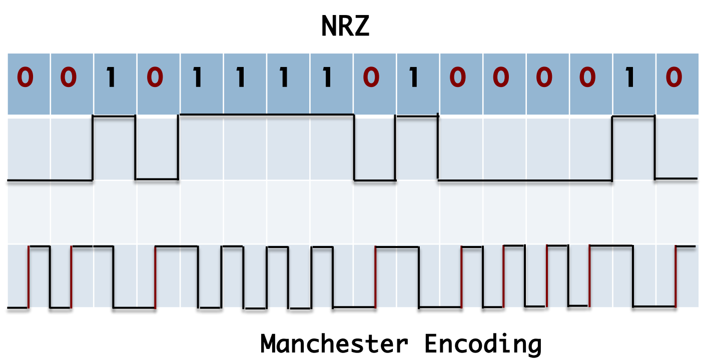
Problems with NRZ because of consecutive 1's and 0's: CLOCK RECOVERY

- Every clock cycle, sender transmits and the receiver receives
- Sender and receiver's clocks must be perfectly **synchronized**
  - Otherwise, it is not possible to decode the signal

Manchester encoding

- **0** is a low-to-high transition
- **1** is a high-to-low transition

Manchester encoding and NRZ



Some more about Manchester encoding

- Doubles the rate at which signal **transitions** are made on the link
  - Receiver has ½ the time to *detect* each pulse
- Rate of signal changes: baud rate
- Bit rate is ½ the baud rate
  - Encoding is considered 50% efficient

NRZI (Non return to zero inverted)

- Make a transition from current signal to encode a 1
  - **Stay** at current signal to encode a 0
- Solves the problem of consecutive 1's
  - But does nothing for consecutive 0's

4B/5B encoding

- Attempts to address inefficiencies in Manchester encoding
  - Without suffering from problems due to extended high/low signals
- The crux here is to insert **extra** bits into bitstream
  - Breakup long sequences of 1s or 0s
  - 4 bits of actual data encoded in a 5-bit code
  - 5-bit codes are carefully selected
    - No more than 1 leading 0 & no more than 2 trailing 0s

4B/5B encoding

4B	5B
0000	11110
0001	01001
0010	10100
0011	10101
0100	01010
0110	01110
0111	01111
1000	10010
1001	10011
1010	10110
1011	10111
1100	11010
1101	11011
1110	11100
1111	11101

4B/5B: Rules for the conversion of 4-bit codes to 5-bit codes

- Objective is to ensure that in each translation there is:
  - No more than one leading 0
  - No more than two trailing 0's
  - When sent back-to-back
    - No pair of 5-bit codes results in more than 3 consecutive 0's being transmitted
- 5-bit codes are transmitted using NRZI
  - This is why they are so concerned with consecutive 0's

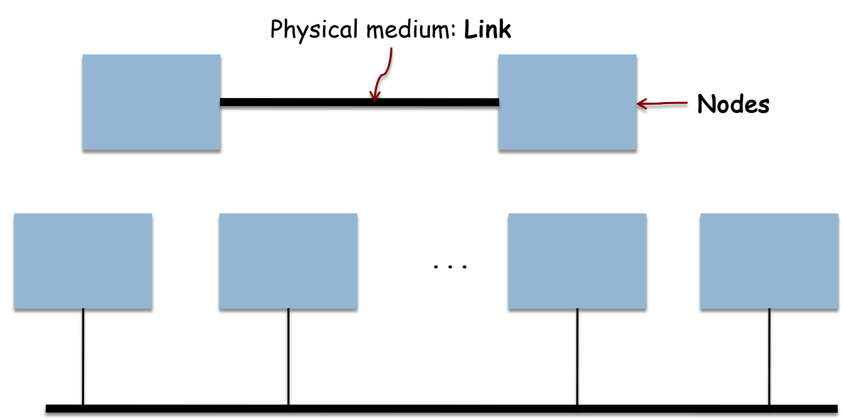
31

32

Expectations that we have of a network

- Application **programmer**
  - Error-free and timely delivery of messages
- Network **designer**
  - Cost effective design
  - Effective and fair allocation of resources
- Network **provider**
  - Easy to administer and manage
  - Isolate faults and account for usage

A network must provide connectivity among a set of computers



Multiple-access

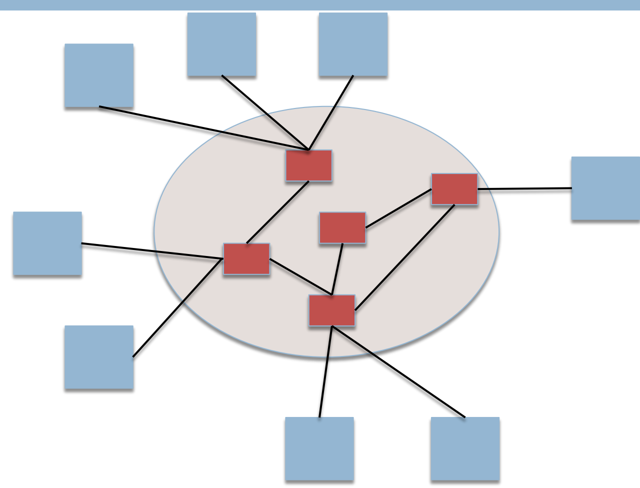
Multiple access links are limited in size

- Geographical **distances** that can be covered
- **Number** of nodes that can be connected

Connectivity between nodes need not imply a direct physical connection. Otherwise …

- Networks would be very **limited** in the number of nodes they could connect
- Number of wires out the back of a node
  - Unmanageable
  - Very expensive

Switched networks: Indirect connectivity among cooperating nodes



37

Switched networks: Indirect connectivity among cooperating nodes

- Nodes with *at least* two links
  - Run software that forwards data on one link out on another
- Types
  - Circuit switched
  - Packet switched

38

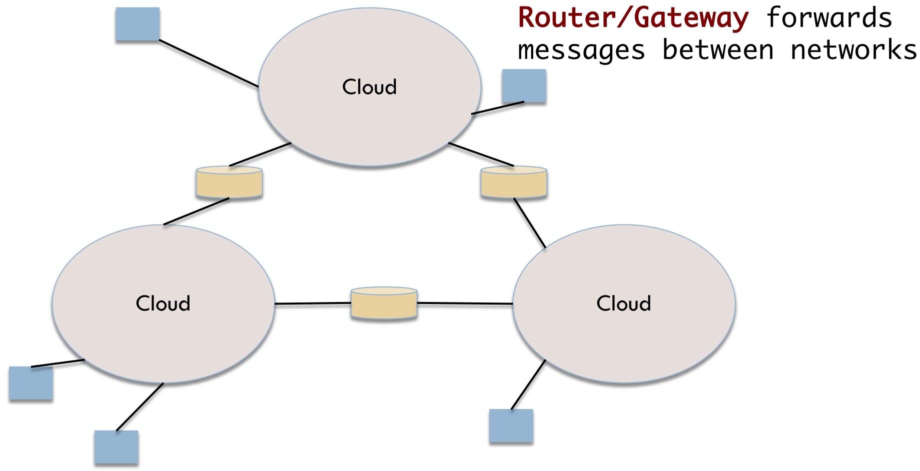
Switched networks: Circuit switched networks

- Establish a **dedicated** circuit
  - Across a set of links
  - No one else can use this till termination
- Allows source to send a stream of bits
  - **Across circuit** to the destination node
- Employed by the telephone system
  - Also known as POTS (Plain Old Telephone System)

Switched networks: Packet switched networks

- Nodes in the network send **discrete** data blocks to each other
  - **Packets**
- Use **store-and-forward**
  1. Receive complete packet over some link
  2. Store packet in internal memory
  3. Forward complete packet to another node
- Used by the **overwhelming majority** of computer networks

Interconnection of networks



Addressing: A node must be able to say which nodes it wishes to communicate with

- Assign an **address** (byte string) to each node
  - Distinguish node from other nodes in the network
- Source specifies address of the destination node
- Switches and routers use address to forward messages *towards* the destination node
  - **Routing**

COST EFFECTIVE RESOURCE SHARING

43

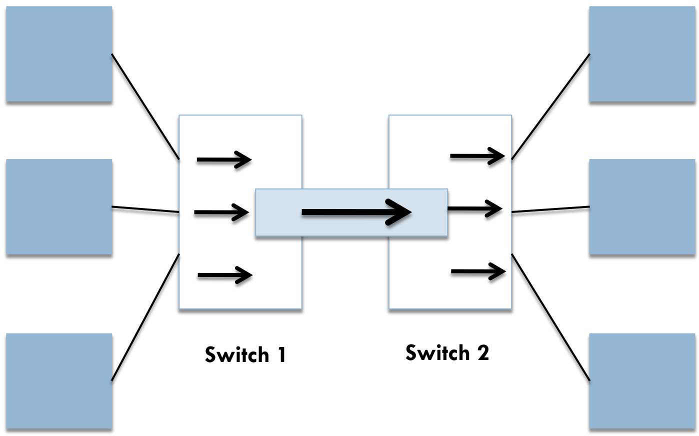
How do all hosts that want to communicate share the network …

- At the same time?
- How about **sharing** links?
  - Hosts want to use it at the same time
- **Multiplexing** …
  - ANALOGY: Sharing CPU among multiple processes

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT NETWORKING L17.44

44

Data sent by multiple users can be multiplexed over the physical links



Multiplexing data onto a physical link

- Synchronous time division multiplexing (STDM)
  - Divide time into quanta
  - Assign quanta in round-robin fashion
- Frequency division multiplexing (FDM)
  - Transit data *flows* at different frequencies

Problems with STDM and FDM

- {Problem-1} **Limited** to specific situations
  - Max number of flows is *fixed*
  - Known *ahead* of time
- {Problem-2} If one of the flows does not have data?
  - Its share of the physical link remains **idle**
- In computer communications:
  1. Amount of time a link is idle can be very large
  2. Data flows are fluid

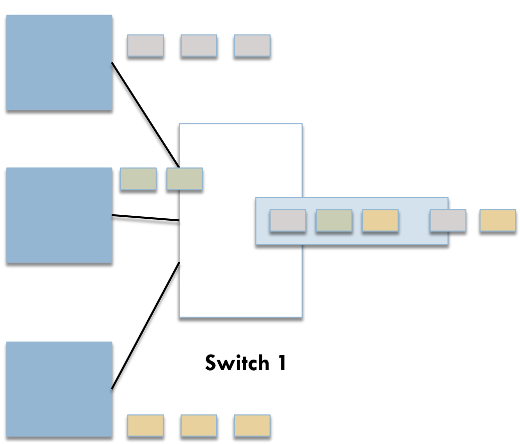
Statistical multiplexing

- Physical link is shared over time
- Data is transmitted from each flow **on demand**
  - Not a predetermined slot
  - When there is only one flow?
    - *No need to wait* for quantum to come around

Limiting transmissions so that other flows can have a turn

- Upper bound on **size** of data block that each flow is allowed to transmit
  - **Packet**
- Larger application messages
  - Fragmented into several packets
  - Receiver reassembles these
- Each flow sends packets over the link
  - Decision made on a **packet-by-packet basis**

Multiplexing packets from multiple sources onto a shared link



Deciding which packet to send over a shared link

- In some cases, decision is made by switches
- Service packets using
  - FIFO
  - Round robin
    - Ensure flows receive a certain **share** of the bandwidth
    - Maximum **threshold** for delays for certain packets
- Networks that allow special treatment of flows
  - Quality of Service

The contents of this slide-set are based on the following references

- *Computer Networks: A Systems Approach. Larry Peterson and Bruce Davie. 4th edition. Morgan Kaufmann. ISBN: 978-0-12-370548-8. [Chapter 1, 2]*
- *Java Network Programming, Third Edition. Elliotte Rusty Harold. O'Reilly. ISBN-10: 0596007213 / 978-0596007218.* [Chapter 7]