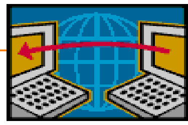


Multiple Access and Spanning Tree

EE122 Fall 2012

Scott Shenker

http://inst.eecs.berkeley.edu/~ee122/

Materials with thanks to Jennifer Rexford, Ion Stoica, Vern Paxson and other colleagues at Princeton and UC Berkeley

1

Give it up for Gautam!

- *"Thanks for being so unbelievably generous with your time and energy on this project! We all really appreciated your almost instantaneous responses on Piazza, your extra office hours, your endless amount of patience with answering the same questions over and over again, and especially all your help today up until the deadline. You are amazing! "*

2

…And to Jamie!

- *"Also, shout-out to Jamie for being so generous with his test-cases yet again, and being so awesome about working out the problems that arose."*

3

Today is Panda's Birthday!

4

Good News!

- **Only half of you will flunk this course!**
- Participation count now at roughly half the class
- Simple question:
  - ***What the hell are the other half of you thinking?***
- The facts:
  - Not enough time for all of you to ask questions in class
  - Cannot just pop your head in OH and have that count.
  - So start participating now….

5

Upcoming lectures

- Congestion Control
- Advanced Topics in Congestion Control
- Wireless (Yahel Ben-David)
- Multicast/QoS/ReverseTR (Scott and Colin)
- Security
- SDN I
- SDN II
- Alternate Architectures
- Summing up

6

Clarification from last time

- Routing tables in “Routing Along DAGs” are per-destination
- Each link has a direction for each destination
  - The direction of the link needed to reach A may be different than the direction of the link needed to reach B
- This is no different than distance vector routing
  - DV has a distance for each destination
- RAD has a vector of directions for each link

Some History

- Ethernet was invented as a broadcast technology
  - Each packet received by all attached hosts
- Easy to set up, cheap to build
  - But hosts had to share channel (multiple access)
- Current Ethernets are “switched”
  - No sharing
- But need spanning tree to route on switches
  - Everyone hates spanning tree, trying to eliminate it

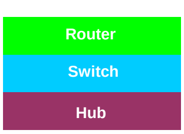
Today

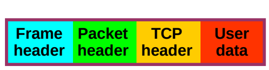
- Study two algorithms that are dying out
  - But both important conceptually!
- Spanning Tree (endangered algorithms list)
  - Still used, but alternatives being developed
- Multiple Access in wired media (extinct)
  - Not used at all, but useful background for wireless

Routing in Switched Ethernets

Shuttling Data at Different Layers

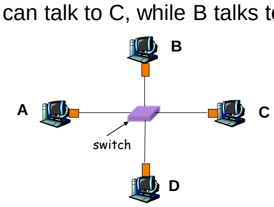
- Different devices switch different things
  - Physical layer: electrical signals or bits (hubs)
  - Link layer: frames (switches)
  - Network layer: packets (routers)





Switches Enable Concurrent Comm.

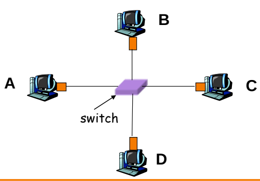
- Host A can talk to C, while B talks to D



- Completely avoids collisions (if hosts directly attached)
  - No need for all material we discuss later in lecture
  - Change in nature of multiple access, but same framing
    - ***Key to the success of ethernet!***

Self Learning

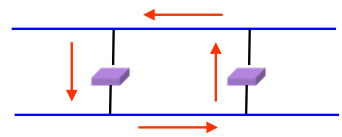
- Maps destination MAC to outgoing interface
- Construct switch table automatically
- Floods when does not have entry in table



13

Flooding Can Lead to Loops

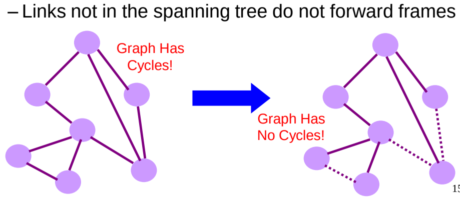
- Flooding can lead to forwarding loops
  - E.g., if the network contains a cycle of switches
  - "Broadcast storm"



14

Solution: Spanning Trees

- Ensure the forwarding topology has no loops
  - Avoid using some of the links when flooding
  - … to prevent loop from forming
- Spanning tree
  - Sub-graph that covers all vertices but *contains no cycles*
  - Links not in the spanning tree do not forward frames



15

You: Design a Spanning Tree Algorithm

- Distributed
- No global information
- Neighbors can exchange information
- Must adapt when failures occur
  - But don't worry about that on first try…
- Take 5 minutes, break into groups, report back

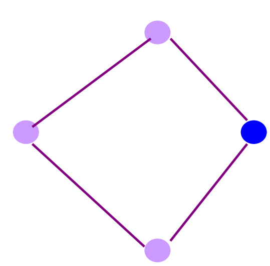
16

What Do We Know?

- Shortest paths to (or from) a node form a tree
  - No shortest path can have a cycle
- But we must limit each node to one outgoing port towards destination
  - Why?
- Because this is not a directed graph!

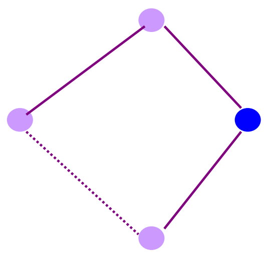
17

Two Shortest Paths Create Cycle!



18

Must only choose one



Algorithm Has Two Aspects

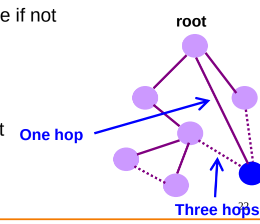
- Pick a root:
  - This will be the destination to which all shortest paths go
  - **Pick the one with the smallest identifier (MAC add.)**
- Compute shortest paths to the root
  - Only keep the links on shortest-paths
  - Break ties in some way, so only keep one shortest path from each node

Breaking Ties

- When there are multiple shortest paths to the root, choose the path that uses the neighbor switch with the lower ID.
- One could use any tiebreaking system, but this is an easy one to remember and implement
- In homeworks and test, remember this.

Constructing a Spanning Tree

- Switches need to elect a root
  - The switch w/ smallest identifier (MAC addr)
- Each switch determines if each interface is on the shortest path from the root
  - Excludes it from the tree if not
- **Messages (Y, d, X)**
  - From node X
  - Proposing Y as the root
  - And the distance is d

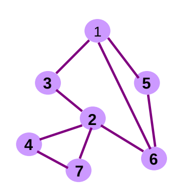


Steps in Spanning Tree Algorithm

- Initially, each switch proposes itself as the root
  - Switch sends a message out every interface
  - … proposing itself as the root with distance 0
  - Example: switch X announces (X, 0, X)
- Switches update their view of the root
  - Upon receiving message (Y, d, Z) from Z, check Y's id
  - If new id smaller, start viewing that switch as root
- Switches compute their distance from the root
  - Add 1 to the distance received from a neighbor
  - Identify interfaces not on shortest path to the root
  - … and exclude them from the spanning tree
- If root or shortest distance to it changed, "flood" updated message (Y, d+1, X)

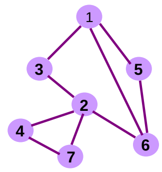
Example From Switch #4's Viewpoint

- Switch #4 thinks it is the root
  - Sends (4, 0, 4) message to 2 and 7
- Then, switch #4 hears from #2
  - Receives (2, 0, 2) message from 2
  - … and thinks that #2 is the root
  - And realizes it is just one hop away
- Then, switch #4 hears from #7
  - Receives (2, 1, 7) from 7
  - And realizes this is a longer path
  - So, prefers its own one-hop path
  - And removes 4-7 link from the tree



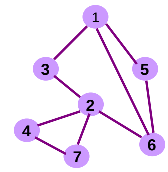
Example From Switch #4's Viewpoint

- Switch #2 hears about switch #1
  - Switch 2 hears (1, 1, 3) from 3
  - Switch 2 starts treating 1 as root
  - And sends (1, 2, 2) to neighbors
- Switch #4 hears from switch #2
  - Switch 4 starts treating 1 as root
  - And sends (1, 3, 4) to neighbors
- Switch #4 hears from switch #7
  - Switch 4 receives (1, 3, 7) from 7
  - And realizes this is a longer path
  - So, prefers its own three-hop path
  - And removes 4-7 link from the tree



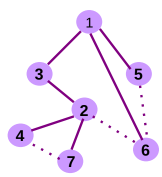
Which links are on spanning tree?

- Take a few minutes, work this out
- 3-1?
- 5-1?
- 6-1?
- 2-6?
- 2-3?



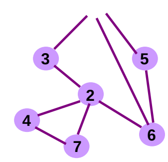
Links on spanning tree

- 3-1
- 5-1
- 6-1
- 2-3
- 4-2
- 7-2



Now which ones are on the spanning tree?

- 2 is new root
- 3-2
- 6-2
- 4-2
- 7-2
- 5-6



Robust Spanning Tree Algorithm

- Algorithm must react to failures
  - Failure of the root node
    - Need to elect a new root, with the next lowest identifier
  - Failure of other switches and links
    - Need to recompute the spanning tree
- Root switch continues sending messages
  - Periodically reannouncing itself as the root (1, 0, 1)
  - Other switches continue forwarding messages
- Detecting failures through timeout (soft state)
  - If no word from root, *time out and claim to be the root!*

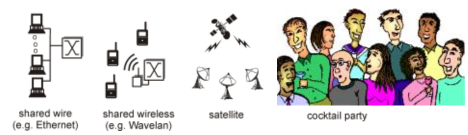
Why do people hate spanning tree?

- Delay in reestablishing spanning tree
  - Network is "down" until spanning tree rebuilt
  - Work on rapid spanning tree algorithms…
    - And multiple spanning trees
- Much of the network bandwidth goes unused
  - Forwarding is only over the spanning tree
  - *Why did you bother with all those other links?*

Broadcast vs Point-to-Point

Point-to-Point vs. Broadcast Media

- Point-to-point: dedicated pairwise communication
  - Long-distance fiber link
  - Point-to-point link between Ethernet switch and host
- Broadcast: shared wire or medium
  - Traditional Ethernet
  - 802.11 wireless LAN



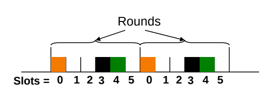
Multiple Access Algorithm

- Single shared broadcast channel
  - Must avoid having multiple nodes speaking at once
  - Otherwise, collisions lead to garbled data
  - Need distributed algorithm for sharing the channel
  - Algorithm determines which node can transmit
- Classes of techniques
  - Channel partitioning: divide channel into pieces
  - Taking turns: scheme for trading off who gets to transmit
  - Random access: allow collisions, and then recover

Channel Partitioning: TDMA

TDMA: Time Division Multiple Access

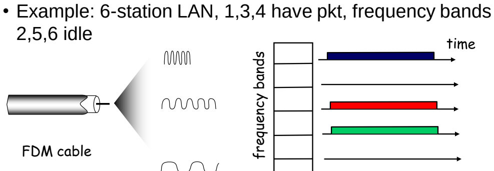
- Access to channel in "rounds"
  - Each station gets fixed length slot in each round
- Time-slot length is packet transmission time
  - *Unused slots go idle*
- Example: 6-station LAN with slots 0, 3, and 4



Channel Partitioning: FDMA

FDMA: Frequency Division Multiple Access

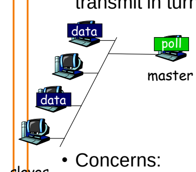
- Channel spectrum divided into frequency bands
- Each station assigned fixed frequency band
- Unused transmission time in frequency bands go idle
- Example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



"Taking Turns" MAC protocols

Polling

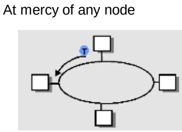
- Master node "invites" slave nodes to transmit in turn



- Concerns:
  - Polling overhead
  - Latency
  - Single point of failure (master)

Token passing

- Control token passed from one node to next sequentially
- Node must have token to send
- Concerns:
  - Token overhead
  - Latency
  - At mercy of any node



None of these are the "Internet way"…

- Why not?
- What's wrong with
  - TDMA
  - FDMA
  - Polling
  - Token passing
- Turn to random access
  - Optimize for the common case (no collision)
  - Don't avoid collisions, just recover from them….
  - **Sound familiar?**

37

Random Access MAC Protocols

38

Random Access MAC Protocols

- When node has packet to send
  - Transmit at full channel data rate
  - No *a priori* coordination among nodes
- Two or more transmitting nodes ⇒ collision
  - Data lost
- Random access MAC protocol specifies:
  - How to detect collisions
  - How to recover from collisions
- Examples
  - ALOHA and Slotted ALOHA
  - CSMA, CSMA/CD, CSMA/CA (wireless, covered later)

39

Key Ideas of Random Access

- Carrier sense
  - *Listen before speaking, and don't interrupt*
  - Checking if someone else is already sending data
  - … and waiting till the other node is done
- Collision detection
  - *If someone else starts talking at the same time, stop*
    - ***But make sure everyone knows there was a collision!***
  - Realizing when two nodes are transmitting at once
  - …by detecting that the data on the wire is garbled
- Randomness
  - *Don't start talking again right away*
  - Waiting for a random time before trying again

40

Where it all Started: AlohaNet

- Norm Abramson left Stanford in 1970
- ***So he could surf!***
- Set up first data communication system for Hawaiian islands
- Hub at U. Hawaii, Oahu
- Had two radio channels:
  - Random access:
    - Sites sending data
  - Broadcast:
    - Hub rebroadcasting data

41

Aloha Signaling

- Two channels: random access, broadcast
- Sites send packets to hub (random)
  - If received, hub sends ACK (random)
  - If not received (due to collision), site resends
- Hub sends packets to all sites (broadcast)
  - Sites can receive even if they are also sending
- Questions:
  - When do you resend? Resend with probability p
  - How does this perform? Need a clean model….

42

Slotted ALOHA

Assumptions

- All frames same size
- Time divided into equal slots (time to transmit a frame)
- Nodes are synchronized
- Nodes begin to transmit frames only at start of slots
- If multiple nodes transmit, nodes detect collision

Operation

- When node gets fresh data, transmits in next slot
- No collision: success!
- Collision: node retransmits with probability **p** until success

43

Slot-by-Slot Example

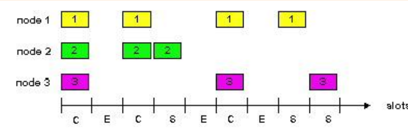
44

Efficiency of Slotted Aloha

- Suppose N stations have packets to send
  - Each transmits in slot with probability *p*
- Probability of successful transmission:
  - by a particular node i: $S_i = p\,(1-p)^{(N-1)}$
  - by any of N nodes: $S = N\,p\,(1-p)^{(N-1)}$
- What value of p maximizes prob. of success:
  - For fixed p, S ➔ 0 as N increases
  - But if p = 1/N, then S ➔ 1/e = 0.37 as N increases
- Max efficiency is only slightly greater than 1/3!

45

Pros and Cons of Slotted Aloha



Pros

- Single active node can continuously transmit at full rate of channel
- Highly decentralized: only need slot synchronization
- Simple

Cons

- Wasted slots:
  - Idle
  - Collisions
- Collisions consume entire slot
- Clock synchronization

46

Improving on Slotted Aloha

- Fewer wasted slots
  - *Need to decrease collisions and empty slots*
- Don't waste full slots on collisions
  - *Need to decrease time to detect collisions*
- Avoid need for synchronization
  - *Synchronization is hard to achieve*
  - ***And Aloha performance drops if you don't have slots***

47

CSMA (Carrier Sense Multiple Access)

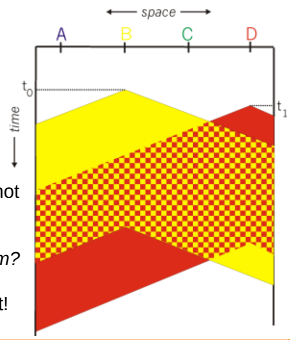
- CSMA: listen before transmit
  - If channel sensed idle: transmit entire frame
  - If channel sensed busy, defer transmission
- Human analogy: don't interrupt others!
- Does this eliminate all collisions?
  - No, because of nonzero propagation delay

48

CSMA Collisions

Propagation delay: two nodes may not hear each other's before sending.

Would slots hurt or help?



CSMA reduces but does not eliminate collisions

Biggest remaining problem?

Collisions still take full slot! How do you fix that?

CSMA/CD (Collision Detection)

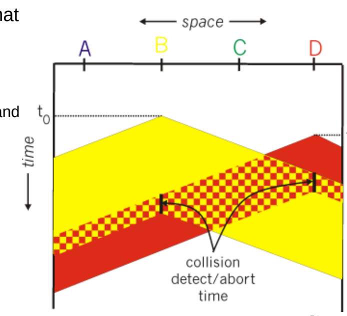
- CSMA/CD: carrier sensing, deferral as in CSMA
  - **Collisions detected within short time**
  - Colliding transmissions aborted, reducing wastage
- Collision detection easy in wired LANs:
  - Compare transmitted, received signals
- Collision detection difficult in wireless LANs:
  - Reception shut off while transmitting (well, perhaps not)
  - Not perfect broadcast (limited range) so collisions local
  - Leads to use of *collision avoidance* instead
    - o ***Will discuss in wireless lecture***

CSMA/CD Collision Detection

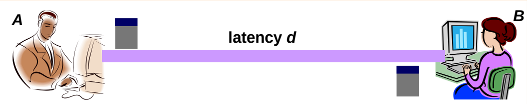
B and D can tell that collision occurred.

Note: for this to work, need restrictions on minimum frame size and maximum distance.

Why?

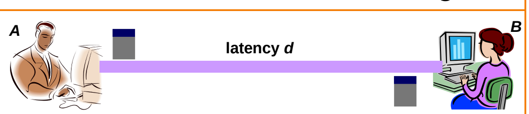


Limits on CSMA/CD Network Length



- Latency depends on physical length of link
  - Time to propagate a packet from one end to the other
- Suppose *A* sends a packet at time ***t***
  - And *B* sees an idle line at a time just before ***t+d***
  - … so *B* happily starts transmitting a packet
- *B* detects a collision, and sends jamming signal
  - But *A* can't see collision until ***t+2d***

Limits on CSMA/CD Network Length



- *A* needs to wait for time ***2d*** to detect collision
  - So, *A* should keep transmitting during this period
  - … and keep an eye out for a possible collision
- Imposes restrictions. E.g., for 10 Mbps Ethernet:
  - Maximum length of the wire: 2,500 meters
  - Minimum length of a frame: 512 bits (64 bytes)
    - o 512 bits = 51.2 µsec (at 10 Mbit/sec)
    - o For light in vacuum, 51.2 µsec ≈ 15,000 meters vs. 5,000 meters "round trip" to wait for collision
  - What about 10Gbps Ethernet?

Performance of CSMA/CD

- Time wasted in collisions
  - Proportional to distance d
- Time spend transmitting a packet
  - Packet length p divided by bandwidth b
- Rough estimate for efficiency (K some constant)

$$E \sim \frac{\frac{p}{b}}{\frac{p}{b} + Kd}$$

- Note:
  - For large packets, small distances, E ~ 1
  - As bandwidth increases, E decreases
  - That is why high-speed LANs are all switched

Ethernet Multiple Access

First widely deployed multiple access

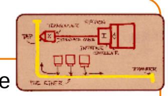
Benefits of Ethernet

- Easy to administer and maintain
- Inexpensive
- Increasingly higher speed
- Evolvable!

Evolution of Ethernet

- Changed everything except the frame format
  - From single coaxial cable to hub-based star
  - From shared media to switches
  - From electrical signaling to optical
- Lesson #1
  - The right interface can accommodate many changes
  - Implementation is hidden behind interface
- Lesson #2
  - Really hard to displace the dominant technology
  - Slight performance improvements are not enough

Ethernet: CSMA/CD Protocol



- **Carrier sense**: wait for link to be idle
- **Collision detection**: listen while transmitting
  - No collision: transmission is complete
  - Collision: abort transmission & send **jam** signal
- **Random access**: binary exponential back-off
  - After collision, wait a random time before trying again
  - After m^{th} collision, choose K randomly from {0, …, 2^m-1}
  - … and wait for K*512 bit times before trying again
    - Using min packet size as "slot"
    - **If transmission occurring when ready to send, wait until end of transmission (CSMA)**

Binary Exponential Backoff (BEB)

- Think of time as divided in slots
- After each collision, pick a slot randomly within next 2^m slots
  - Where m is the number of collisions since last successful transmission
- Questions:
  - Why backoff?
  - Why random?
  - Why 2^m?
  - Why not listen while waiting?

Behavior of BEB Under Light Load

Look at collisions between two nodes

- First collision: pick one of the next two slots
  - Chance of success after first collision: 50%
  - Average delay 1.5 slots
- Second collision: pick one of the next four slots
  - Chance of success after second collision: 75%
  - Average delay 2.5 slots
- In general: after m^{th} collision
  - Chance of success: $1-2^{-m}$
  - Average delay (in slots): ½ + $2^{(m-1)}$

BEB: Theory vs Reality

In theory, there is no difference between theory and practice. But, in practice, there is.

61

BEB Reality

- Performs well (far from optimal, but no one cares)
  - *Large packets are ~23 times as large as minimal slot*
- Is mostly irrelevant
  - *Almost all current ethernets are **switched***

62

BEB Theory

- A very interesting algorithm
- Stability for finite N only proved in 1985
  - Ethernet can handle nonzero traffic load without collapse
    - Greenberg et al. (AT&T)
- All backoff algorithms unstable for infinite N (1985)
  - Poisson model: infinite user pool, total demand is finite
    - David Aldous (UCB Statistics)
- Not of practical interest, but gives important insight
  - Multiple access should be in your "bag of tricks"

63

Question

- Two hosts, each with infinite packets to send
- What happens under BEB?
- Throughput high or low?
- Bandwidth shared equally or not?

64

The BEB Game Show!

- Starring two enthusiastic volunteers

65

MAC "Channel Capture" in BEB

- Finite chance that first one to have a successful transmission will never relinquish the channel
  - The other host will *never* send a packet
- Therefore, asymptotically channel is fully utilized and completely allocated to one host

66

Example

- Two hosts, each with infinite packets to send
  - Slot 1: collision
  - Slot 2: each resends with prob ½
    - Assume host A sends, host B does not
  - Slot 3: A and B both send (collision)
  - Slot 4: A sends with probability ½, B with prob. ¼
    - Assume A sends, B does not
  - Slot 5: A definitely sends, B sends with prob. ¼
    - Assume collision
  - Slot 6: A sends with probability ½, B with prob. 1/8
- Conclusion: if A gets through first, the prob. of B sending successfully halves with each collision

67

Another Question

- Hosts now have large but finite # packets to send
- What happens under BEB?
- Throughput high or low?

68

Answer

- Efficiency less than one, no matter how many packets
- Time you wait for loser to start is proportional to time winner was sending….

69

Different Backoff Functions

- Exponential: backoff ~ a^i
  - Channel capture?
  - Efficiency?
- Superlinear polynomial: backoff ~ i^p p>1
  - Channel capture?
  - Efficiency?
- Sublinear polynomial: backoff ~ i^p p≤1
  - Channel capture?
  - Efficiency?

70

Different Backoff Functions

- Exponential: backoff ~ a^i
  - Channel capture *(loser might not send until winner idle)*
  - Efficiency less than 1 *(time wasted waiting for loser to start)*
- Superlinear polynomial: backoff ~ i^p p>1
  - Channel capture
  - Efficiency is 1 (for any finite # of hosts N)
- Sublinear polynomial: backoff ~ i^p p≤1
  - No channel capture *(loser not shut out)*
  - Efficiency is less than 1 (and goes to zero for large N)
    - *Time wasted resolving collisions*

71

Why Do I Care?

- Why do you like music?
- It makes me happy….
- But also, until this work was done, no one knew about capture, or what properties of the backoff enabled it.
- You don't understand something until you've ***played*** with it. Just getting it to work isn't enough.

72

That's All for Today!

- Next week, congestion control

73