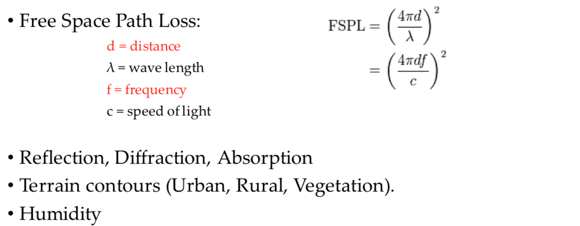
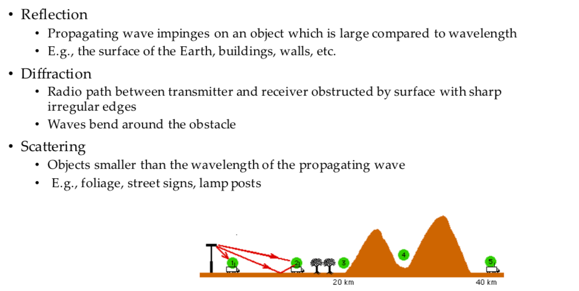
**Lecture 11: Introduction to Wireless**

**Supported data rate fluctuates a lot** because radio waves propagate. 1) Decreasing signal strength 2) waves lose energy due to absorption or scattering 3) multi path fading, reflections from multiple objects

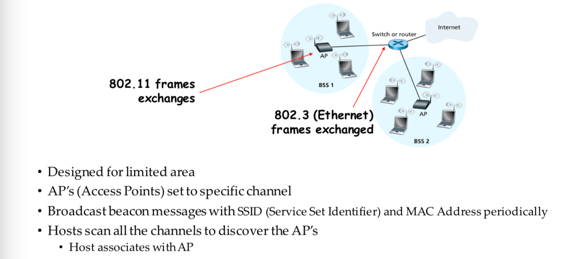


Signals bounce off surface and interfere with one another (self-interference). Mobility creates fluctuations of signal. Power degrades by 1/d^2. Signals blocked by structures. Multipath effects, rapid changes in signal strength over small area or time interval.



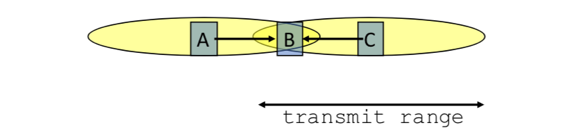
**There is broadcast interference**. Your signal is noise to others. We have to tolerate external interference but we can avoid internal interference. Anybody in proximity can hear and interfere, cannot receive while transmitting (our own transmission deafens receiver), signals sent by sender don’t always end up intact.

**Lower (Signal/Noise) = Higher Bit Error Rate** increasing signal strength not helpful as it increases interference and everyone increases power. Higher SNR = higher bitrate. In face of loss should we decrease/increase bitrate? If free space loss or multi-path fading lower bitrate, if external interference short burst high bitrates. Freq = cycles / sec, wavelength=length of cycle. Wireless spectrum allocated to license holders. Chunks get auctioned for billions of dollars. Wireless standards (cellular, 802.11 WiFi, 802.15 BT, sensor networks). **Antennas –** higher gain antenna is a directional antenna, used to increase signal strength, omni-directional antenna radiates equal radio power in all directions perpendicular to an axis. **WIFI:**

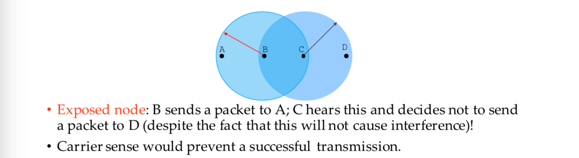


Collision detection(R) – where do collisions occur, Carrier sense(T) – senders can listen before sending.

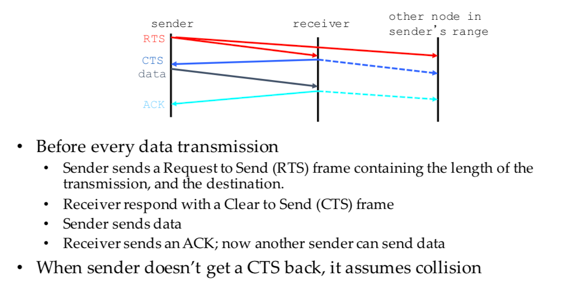
**Hidden terminals**(A, C can’t hear)



**Exposed Terminals**



Collisions are at receiver not sender, does not matter if senders hear someone else. Detect if receiver can hear sender and tell others to be quiet. **Collision Avoidance** try to avoid collisions. Choose random interval and wait that many timeslots before sending. When collision inferred retransmit with binary exponential back off. **CSMA/CA** (carrier sense mult. Access/collision avoidance)



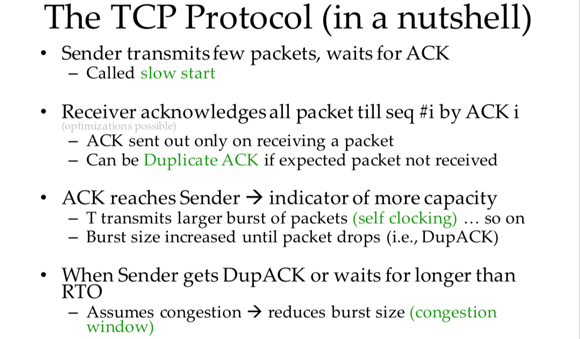
If other nodes hear RTS but not CTS: send, dest for first node is OoR. May cause problems when CTS is lost. When you hear a CTS you keep quiet until you hear ACK.

Frequency spectrum partitioned into several channels, nodes within interference range can just use separate channels

**Lecture 12: More Wireless**

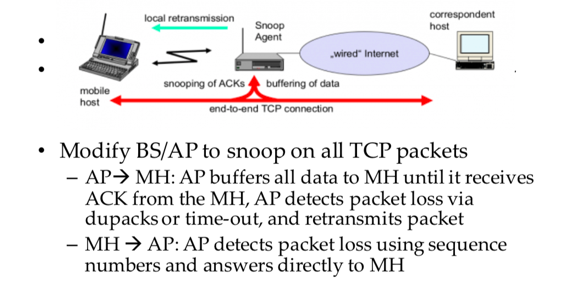
**Multi-hop routing** – each node cannot transmit and receive at the same time. If a node receives RTS simultaneously both could be dropped (collision). TCP uses ACK to indicate delivery = bidirectional traffic

**TCP Congestion control** – How much data to pump into network? No info about network, TCP receiver can give feedback. Flow control and congestion control, what rate to pour?

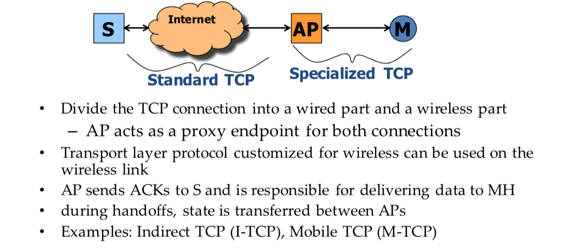


TCP assumes packet loss is due to congestion and regulates through congestion window but wireless loss can be due to many things. TCP is e2e and cannot see the network and cannot classify cause of loss. (TCP) Sender should retransmit a packet lost due to transmission error without congestion control actions. (Network) Transmission error should be hidden from sender.

Mask wireless loss from TCP sender – **Link layer schemes** need link layer modification at both ends but no TCP modification. Assume wireless connection is just one hop and no loss due to congestion. Forward Error Correction – correct small number of errors and hide from TCP. Retransmit a packet at the link layer if errors are detected. To avoid unnecessary fast retransmit link layer should attempt to deliver packets almost in order. **SNOOP TCP aware link layer** require modification to BS/AP (network approach)



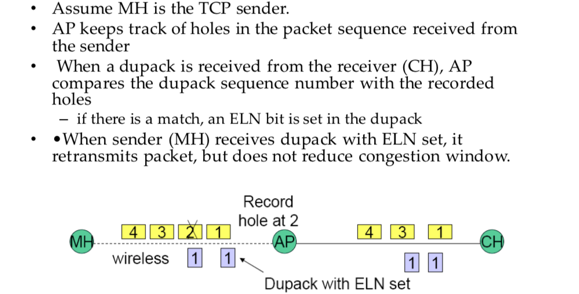
Hide loss from sender, modify BS, preserve end to end semantics. Does not work with encrypted TCP head or asymmetric routes. **Split connection approach**



No changes to fixed network, transmission errors do not propagate to fixed network, custom protocol for wireless. Bad: no end to end semantics, AP needs large buffer space, AP must maintain per-TCP connection state

End to end approach, modify TCP so sender responds differently to non-congestion loss.

**Explicit notification**, wireless node determines packets are lost due to errors and send explicit notification, sender retransmits and does not reduce CW.

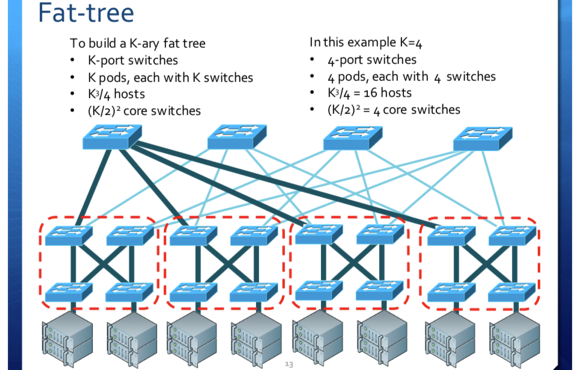


**Selective Acknowledgement**, TCP SACK allows multiple packet losses to be indicated by a single acknowledgement. Can list up to 3 blocks of data that have been received.

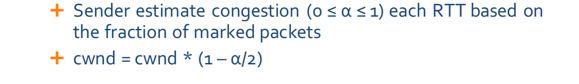
**Lecture 13: Data Center Networks**

Problems: slow, wiring, expensive, hard to manage

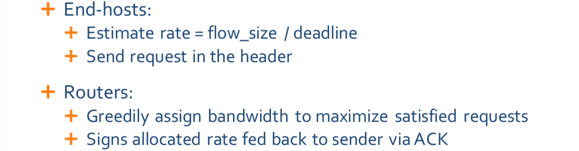
**Oversubscription,** bandwidth gets scarce as you move up tree, limits scalability



Full bisection bandwidth, low cost commodity hardware, redundancy. Bad: custom routing, wiring is a nightmare. 3K^3 / 4: Wiring is complex and costly, difficult to change, traffic demands unpredictable. **Sporadic congestion losses caused by traffic hotspots.** Need flexible interconnects to add bandwidth on demand (Wireless, Optical). **Wireless** create links on the fly, 60 Ghz beamforming (multi Gbps data rate, small interference footprint). Connect racks by reflecting signal off ceiling, no more link blockage and small interference. Challenges: minimize interference, antenna rotation delay. **Optical links** mechanically adjust mirrors in milliseconds. Wavelength division multiplexing, single port carries multiple streams concurrently. **Helios,** packet switch network for bursty traffic and optical circuits for stable traffic. **Transport protocols,** queue buildup problem (long TCP flows congest the network, ramp up and don’t top until loss and oscillate around max utilization, short flows cannot compete). **Partition/Aggregate pattern,** problem: incast (all workers answer at same time->packet loss), buffer pressure (cheap switches share buffer memory so thin flow congested by fat flow). **DCTCP** alter TCP. Scale window in proportion to congestion, use explicit congestion notification ECN



No scheduling, oblivious to deadline. **Clean slate D^3,** ask for bandwidth required to meet deadline

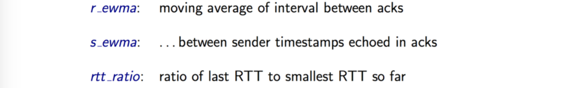


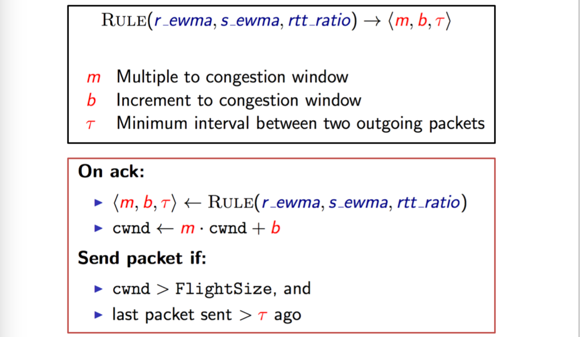
Higher goodput under heavy load, support deadline bound apps. BAD: all or nothing deployment, complexity in switch, applicational level changes

**Lecture 14: Data Driven Networking Design**

Understand real network behaviors from in the wild measurements. Use machine learning to identify rules for network functionalities.

**TCP Remy,** Rule (send packet or do not send packet), objective (maximize proportional fair, minimize avg flow completion time, page load time, tail completion time). Assumes that everyone is running the same algorithm.

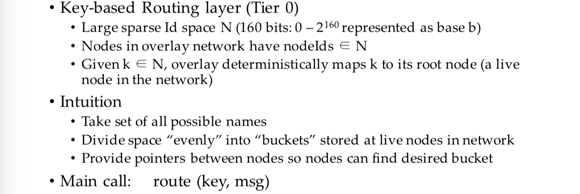


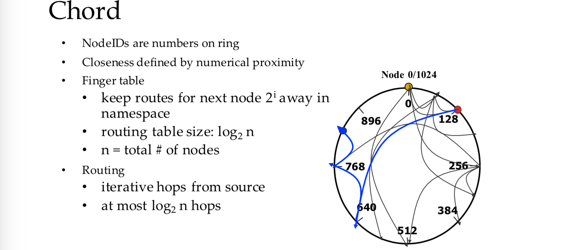


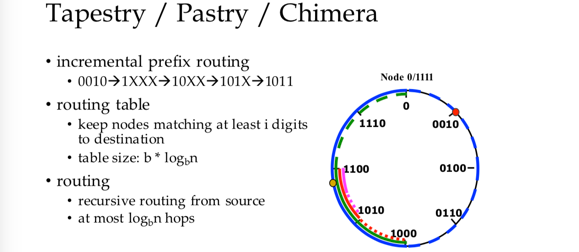
Assumes that designer can model the network explicitly, mismatched set of assumptions? (link speed, delay) can tolerate mismatched link rate assumption but need precision about the number of senders. **Sybils,** a fake account that attempts to create many friendships with honest users. Prior work assumed that sybils form tight knit communities. In reality sybils are hard bc cheap labor can create realistic fakes, sybils also only a small portion of sybils are connected to each other.

**Lecture 15: Decentralized P2P**

Users bring their own resources to the table. Clients = peers = servers. Scalability grows with users, load spread across many peers, peers are distributed. How do peers talk to each other? How do peers know who to talk to? **Unstructured P2P applications,** focus on locating popular objects, hierarchy of super nodes index client file collections, queries sent as controlled flood amongst supernodes. BAD: does not support generic any to any node communication, can only search a small portion of files in system or else will flood network with query traffic, uncommon files are easily lost. **Gnutella,** connect to random set of existing hosts, resolve queries through localized flooding, high bandwidth costs in control msgs, flood of queries took up all available bandwidth. Resilient to random failures but not attacks. Random 30% removed vs top 4% removed. **Hierarchical P2P Networks,** Kazaa, hierarchical flooding helps improve scale, susceptible to poison attacks (Get online and distribute large # of popular files with static.). BitTorrent, seed has a complete copy and forwards different blocks to different users for simultaneous downloads by many users. Need tracker to match leechers with seeds. **Structured P2P,** match all objects to machine names, mapping must be easy, embedded into the network, and consistent. Each machine only needs to know log(n) other nodes and routing from A->B takes at most log(n) steps. Deterministic key-node mapping. Key based routing. Incremental routing towards destination ID.







**Misc.**

