Wednesday, June 7, 2017

Introduction to Networking

Packet Switching

The History of Networks and History of the Internet

- history of how messages were communicated in past
- concepts
- Telephone - end of 19 century, Bell
- Licklider - Intergalactic Network
  - Research program at DARPA (Defense Advanced Research Projects Agency)
- ARPANET
  - single, closed, proprietary network, 1969

What is packet switching?

- predecesor
  - circuit switching
    - used in telephone network (wired)
    - history: telephone -> dedicated wire -> switching board operator -> telephone
    - Very phone call has its own dedicated 64 kilobits per second circuit (no sharing)
    - Characteristics
      - call has its private, isolated, guaranteed data rate from end-to-end
      - a call has 3 phases
        - Establish circuit from end-to-end = dialing
        - Communicate
        - Close circuit = tear down
      - Originally, a circuit was an end-to-end physical wire
      - Nowadays, a circuit is like a virtual private wire
    - Problems with it when we use it with Internet

- Inefficient - burst X silence
- Divergent rates for different application
- state management - every switch along the way maintain state

- packet switching
  - no dedicated circuit to carry data
  - Network: host, links, packet switches
    - Different times of packet switch
      - routers, ethernet switches
      - they have buffers
        - switch has to hold one packet before it holds the other
  - Packets are routed individually, by looking up address in router's local table
  - all packets share the full capacity of a link
  - the routers maintain no per-communication state

- Why does internet use packet switching?
  - efficient use of links
    - Originally: links are expensive, bursty flows can share same links
  - Resilience to failure of links and routers
    - one router breaks -> different path
  - Internet was originally designed as an interconnection of the existing networks

Terminology, End to End Delay and Queueing Delay

- Propagation delay
  - the time it takes a single bit to travel over a link at propagation speed c
  - Determined by how long link is
  - c is close to speed of light
  - it doesn't matter if link is running at 1kb/s or 1 Gb/s
- Packetization Delay

- The time from when the first to the last bit of a packet is transmitted
- r - how fast we can put bits on the link, data rate
    - link is 1 kb/s -> we can put 1000 new bits onto the link every second
    - p=64*8=512
- End-to-end delay
    - time from when we send the first bit on the first link until last bit of the packet arrives at the destination
- how to compute it end to end delay
    - switch s1 is going to wait until last packet arrives
    - cut though switching
        - Switch send packet forward as soon as it gets header
        - Switches generally don't do it
- Sharing link
    - several packet arrive to switch -> wait in the router's queue (packet buffer)
    - link is congested
    - if there is no packet buffer we have to drop packet every time two packets arrive at the same time
- end to end delay is unpredictable
    - Ping 168.32.56.34
        - Measure trip time

Playback Buffers

- real time applications care about queueing delay
    - Application can't be sure they will have a voice or video sample in time to deliver it to user
        - Playback buffer
- first example
    - Axis

    - Horizontal axis is the delay
    - Vertical axis tells how many bits are buffered right now in network
    - Horizontal (received by laptop - playback = how long time packet have been buffered)
  - Biggest component in delay
    - Propagation and packatization delay
      - Fixed
  - lower bound - Overall delay can't be less than propagation and packatization delay
  - Upper bound - buffers have maximum size
    - not useful, can be large
  - Right hand size is non decreasing
- we didn't wait long enough until we started to play video
- Buffer goes empty
  - no bytes to decode = deficit
  - rebuffering - freezing the screen, waiting for bytes to accumulate

Simple Deterministic Queue Model

- simple model of a router queue
  - Q(t) - at time t queue has Q packets
  - A(t) - A packets arrived to router up until t
  - D(t) - D packets departed up until time t
  - R - deterministic and fixed rate of r
- Graph
  - Green line
    - arrival rate of bytes on the incoming link
      - bytes in packets so there are “teeth” in graph
    - p - length of packet in bytes

- Yellow line
  - Departure process
  - Works at rate R
- we can tell value of Q = (A - D)
- Delay through queue (horizontal)

- small packets can reduce end to end delay
  - when we send whole message
    - it has to be transferred over first link before it can start on the second link
  - When we send packets we can send other packets parallel
- Example
  - Rate R - all links -> outgoing link would be overwhelmed -> we will be dropping them at a rate of N
  - Benefit the statistical multiplexing gain
    - Benefit that we are getting from summing up arriving rates
    - ration of the rate that we need in to prevent packet loss
    - 2 definitions - one consider buffer and second one doesn't
- Statistical multiplexing
  - Arriving rates A and B, draining rate C

Queueing Model Properties

- Network is set of queues interconnected by some links
  - Process of packet arrival is complicated
    - so we think of it as random event
- Queueing Theory
  - study of random arrival processes
- Circled arrow
  - you can't have a negative queue occupancy

- Bustiness increases delay
- Determinism minimizes delay
  - Random arrivals wait longer on average than simple periodic arrivals
- Little's result
  - Lambda - arrival rate
  - L - average number of queues in system (+currently being serviced)
  - D - average delay of packets that arrived until they completed service
  - Valid if there are no packets that are lost or dropped
- Poisson process
  - models aggregation of many independent random events
  - packet that arrives are not Poisson
- M/M/1 Queue
  - M - Markovian arrival process (Poisson process), exponential
  - it assumes a nice simple Poisson arrival

Packet Switching 1

- Generic packet switch
  - 3 stages of packet switch
    - look up the address
      - Destination address -> where to go next
      - egress link = the port that it is going to
    - Update header
      - decrement TTL, update checksum
    - Queue the packet
      - if congestion -> buffer
- Ethernet switch
  - Example of packet switch, dealing with ethernet frames

- 4 operations
  - examine header of each arriving frame
  - Forward packet if Ethernet DA is in the forwarding table
  - Broadcast the frame to all ports if the Ethernet DA is not in the forwarding table
    - Except the one the frame arrived
  - entries in the table are learned by examining the Ethernet SA of arriving packets

- Internet router
  - Process IP addresses instead
  - 7 operations
    - Checks if the Ethernet DA of the arriving frame belongs to the router, accept the frame
      - Everything else is dropped
    - examine the IP version number and length of the datagram
    - Decrement TTL and update IP header checksum
    - check to see if TTL=0
    - if the IP DA is in the forwarding table, forward to the correct egress port(s0 for the next hop
    - find the Ethernet DA for the next hop router
    - create a new Ethernet frame and send it
- Lookup address
  - how is address looked up in forwarding table
  - Ethernet switches
    - It stores addresses in the hash table (maybe 2-way hash)
    - look in the exact match in the hash table
    - that is because there can be many entries
  - IP switches

- we don't look up exact match
- we look up on what's called a longest prefix match
- find match -> it gets address of next router -> find equivalent Ethernet address
- Longest prefix match
  - binary trees
    - we use binary trees to find longest prefix match
  - Ternary content addressable memory TCAM
    - brute force
    - 1 - bit matters, 0 - bit doesn't matter
  - Generic lookup

Packet Switching 2

- Output switch
  - Worst case
    - N ports
    - Running rate R
    - R switch reading rate
    - So: N*R writing rate to queue
    - So: memory run an aggregate a total rate of up to (N+1)*R
- Input switch
  - Packets are held at the input side of the switch
  - so its speed is reduced from (N+1)*R to 2R
  - are more scalable
    - Head of line blocking
      - Problem
        - unused outputs
      - virtual output queues

- each input maintains a separate queue for each output

- Output queued packet switch
  - the average delay that a packet would experience as a function of the load
- main properties of output queued switches
  - Work conserving
    - output line is never idle when there is a packet in the system waiting to go to it
  - Throughput is maximized
    - you cannot have a higher throughput than keeping all the lines busy whenever there's packet available for them
    - we lose 58% of the performance of the system as a consequence of the head of line blocking

Rate Guarantees

- FIFO queue
  - First in first out
  - free for all
  - Whoever sends the most packets will receive the highest usage of this output link
  - Encourages bad behavior
- Strict priorities
  - decision based on bits in the header
  - in IP header: type of service field ToS
  - it is always going to take packets from the high priority if they are there
    - it will only serve the low priority if there is nothing in the high priority queue
  - use it when there is small amount of high priority traffic
- Weighted priorities
  - Twice as many opportunities to send

- each flow has guaranteed service rate
  - Scheduling packets in order of their bit-by-bit finishing times

Delay Guarantees

- weighted for queueing
  - delay through router will be bounded by the size of the buffer divided by R1
- How we can control the delay of packets
  - the rate at which a queue is served
  - the size of each queue
- packets arrives at such a rate that it overflows the buffer
- if we know the size of a queue and the rate at which it is served than we can bound the delay through it
- we can pick the size of the queue and WFQ (weighted fair queueing) lets us pick the rate at which it is served
- Therefore we just need a way to prevent packets being dropped along the way
  - For this we can use leaky bucket regulator
- we can therefore bound the end to end delay