CompSci514/ECE558: Computer Networks

Lecture 22: Review
Xiaowei Yang

xwy@cs.duke.edu
http://www.cs.duke.edu/~xwy
Roadmap

• Summarize what we have learned in this semester
  – Design principles of computer networks
  – Congestion control
  – Routing
  – Datacenter networking: topology and congestion control
  – SDN, NFV, Programmable Routers, RDMA, Network measurement, DDoS, and DHT
Architectural questions tend to dominate CS networking research
Decomposition of Function

Definition and placement of function
  – What to do, and where to do it

The “division of labor”
  – Between the host, network, and management systems
  – Across multiple concurrent protocols and mechanisms
Lecture 3: The Design Philosophy of the DARPA Internet Protocols

Xiaowei Yang
xwy@cs.duke.edu
What is the paper about?

• Where to place functions in a distributed computer system
  – End point, networks, or a joint venture?

• Authors’ arguments:

“The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)”
End-to-End Argument

• Extremely influential

• “…functions placed at the lower levels may be redundant or of little value when compared to the cost of providing them at the lower level…”

• “…sometimes an incomplete version of the function provided by the communication system (lower levels) may be useful as a performance enhancement…”
Example: Reliable File Transfer

- Solution 1: make each step reliable, and then concatenate them
  - Uneconomical if each step has small error probability
Example: Reliable File Transfer

- Solution 2: end-to-end check and retry
  - Correct and complete
Example: Reliable File Transfer

• An intermediate solution: the communication system provides internally, a guarantee of reliable data transmission, e.g., a hop-by-hop reliable protocol
  – Only reducing end-to-end retries
  – No effect on correctness
Question: should lower layer play a part in obtaining reliability?

- Answer: it depends
  - Example: extremely lossy link
    - One in a hundred packets will be corrupted
    - 1K packet size, 1M file size
    - Probability of no end-to-end retry: \((1-1/100)^{1000}\) 
      \(\approx 4.3e-5\)
The Design Philosophy of the DARPA Internet Protocols

- Inter-networking: an IP layer
  - Alternative: A unified approach
    - Can’t connect existing networks
    - Inflexible

- Packet switching vs circuit switching
  - Applications suitable for packet switching
  - Existing networks were packet switching

- Gateways
  - Chosen from ARPANET
  - Store and forward
  - Question: can we interconnect without gateways?
Secondary goals

- In order of importance
  1. Survivable of network failures
  2. Multiple services
  3. Varieties of networks
  4. Distributed management
  5. Cost effective
  6. Easy attachment
  7. Resource accountable

- How will the order differ in a commercial environment?
Design Goals of Congestion Control

- Congestion avoidance: making the system operate around the knee to obtain low latency and high throughput

- Congestion control: making the system operate left to the cliff to avoid congestion collapse
Key insight: packet conservation principle and self-clocking

• When pipe is full, the speed of ACK returns equals to the speed new packets should be injected into the network
Solution: Dynamic window sizing

• Sending speed: SWS / RTT

• Adjusting SWS based on available bandwidth

• The sender has two internal parameters:
  – Congestion Window (cwnd)
  – Slow-start threshold Value (ssthresh)

• SWS is set to the minimum of (cwnd, receiver advertised win)
Two Modes of Congestion Control

1. Probing for the available bandwidth
   - slow start \((cwnd < ssthresh)\)

2. Avoid overloading the network
   - congestion avoidance \((cwnd >= ssthresh)\)
Slow Start

- Initial value: \textbf{Set $cwnd = 1 \text{ MSS}$}
  - Modern TCP implementation may set initial $cwnd$ to a much larger value

- When receiving an ACK, $cwnd += 1 \text{ MSS}$
Congestion Avoidance

- If $cwnd \geq ssthresh$ then each time an ACK is received, increment $cwnd$ as follows:
  - $cwnd +\text{MSS} \times (\text{MSS} / cwnd)$ (cwnd measured in bytes)

- So $cwnd$ is increased by one MSS only if all $cwnd$/MSS segments have been acknowledged.
Example of Slow Start/Congestion Avoidance

Assume $ssthresh = 8 \text{ MSS}$
The Sawtooth behavior of TCP

- For every ACK received
  - $Cwnd += 1/cwnd$

- For every packet lost
  - $Cwnd /= 2$
Why does it work? [Chiu-Jain]

- A feedback control system
- The network uses feedback $y$ to adjust users’ load $\sum x_i$
Goals of Congestion Avoidance

- **Efficiency**: the closeness of the total load on the resource to its knee

- **Fairness**:
  - When all $x_i$'s
  - When all $x_i$'s are zero but $x_j = 1$, $F(x) = 1/n$

\[
F(x) = \frac{(\sum x_i)^2}{n(\sum x_i^2)}.
\]

- **Distributedness**
  - A centralized scheme requires complete knowledge of the state of the system

- **Convergence**
  - The system approach the goal state from any starting state
Metrics to measure convergence

- Responsiveness
- Smoothness

Fig. 3. Responsiveness and smoothness.
Model the system as a linear control system

\[ x_i(t + 1) = \begin{cases} 
  a_I + b_I x_i(t) & \text{if } y(t) = 0 \Rightarrow \text{Increase,} \\
  a_D + b_D x_i(t) & \text{if } y(t) = 1 \Rightarrow \text{Decrease.} 
\end{cases} \]

- Four sample types of controls
- AIAD, AIMD, MIAD, MIMD
Phase plane
TCP congestion control is AIMD

- Problems:
  - Each source has to probe for its bandwidth
  - Congestion occurs first before TCP backs off
  - Unfair: long RTT flows obtain smaller bandwidth shares
Macroscopic behavior of TCP

• Throughput is inversely proportional to RTT:
  \[
  \frac{\sqrt{1.5 \cdot MSS}}{RTT \cdot \sqrt{p}}
  \]

• In a steady state, total packets sent in one sawtooth cycle:
  \[ S = w + (w+1) + \ldots (w+w) = \frac{3}{2} w^2 \]

• the maximum window size is determined by the loss rate
  \[ \frac{1}{S} = p_1 \]
  \[ w = \frac{1}{\sqrt{1.5p}} \]

• The length of one cycle: \( w \times RTT \)
• Average throughput: \( \frac{3}{2} w \times MSS / RTT \)
Explicit Congestion Notification

• Use a Congestion Experience (CE) bit to signal congestion, instead of a packet drop
• Why is ECN better than a packet drop?
• AQM is used for packet marking
Other Congestion Control Algorithms

- XCP
- VCP
- BBR
- Cubic
Design Space for resource allocation

- Router-based vs. Host-based
- Reservation-based vs. Feedback-based
- Window-based vs. Rate-based
Fair Queuing Motivation

- End-to-end congestion control + FIFO queue (or AQM) has limitations
  - What if sources mis-behave?

- Approach 2:
  - Fair Queuing: a queuing algorithm that aims to “fairly” allocate buffer, bandwidth, latency among competing users
Outline

• What is fair?
• Weighted Fair Queuing
• Other FQ variants
One definition: Max-min fairness

• Many fair queuing algorithms aim to achieve this definition of fairness

• Informally
  – Allocate user with “small” demand what it wants, evenly divide unused resources to “big” users

• Formally
  – 1. No user receives more than its request
  – 2. No other allocation satisfies 1 and has a higher minimum allocation
    • Users that have higher requests and share the same bottleneck link have equal shares
  – Remove the minimal user and reduce the total resource accordingly, 2 recursively holds
Max-min example

1. Increase all flows’ rates equally, until some users’ requests are satisfied or some links are saturated

2. Remove those users and reduce the resources and repeat step 1

• Assume sources 1..n, with resource demands X1..Xn in an ascending order

• Assume channel capacity C.
  – Give C/n to X1; if this is more than X1 wants, divide excess (C/n - X1) to other sources: each gets C/n + (C/n - X1)/(n-1)
  – If this is larger than what X2 wants, repeat process
Design of weighted fair queuing

- Resources managed by a queuing algorithm
  - Bandwidth: Which packets get transmitted
  - Promptness: When do packets get transmitted
  - Buffer: Which packets are discarded
  - Examples: FIFO
    - The order of arrival determines all three quantities

- Goals:
  - Max-min fair
  - Work conserving: link’s not idle if there is work to do
  - Isolate misbehaving sources
  - Has some control over promptness
    - E.g., lower delay for sources using less than their full share of bandwidth
    - Continuity
      - On Average does not depend discontinuously on a packet’s time of arrival
      - Not blocked if no packet arrives
Design goals

- Max-min fair
- Work conserving: link’s not idle if there is work to do
- Isolate misbehaving sources
- Has some control over promptness
  - E.g., lower delay for sources using less than their full share of bandwidth
  - Continuity
    - On Average does not depend discontinuously on a packet’s time of arrival
    - Not blocked if no packet arrives
Implementing max-min Fairness

• Generalized processor sharing
  – Fluid fairness
  – Bitwise round robin among all queues
• WFQ:
  – Emulate this reference system in a packetized system
  – Challenges: bits are bundled into packets. Simple round robin scheduling does not emulate bit-by-bit round robin
Emulating Bit-by-Bit round robin

- Define a virtual clock: the round number \( R(t) \) as the number of rounds made in a bit-by-bit round-robin service discipline up to time \( t \).

- A packet with size \( P \) whose first bit serviced at round \( R(t_0) \) will finish at round:
  - \( R(t) = R(t_0) + P \)

- Schedule which packet gets serviced based on the finish round number.
Compute finish times

- Arrival time of packet $i$ from flow $\alpha$: $t_i^\alpha$
- Packet size: $P_i^\alpha$
- $S_i^\alpha$ be the round number when the packet starts service
- $F_i^\alpha$ be the finish round number
- $F_i^\alpha = S_i^\alpha + P_i^\alpha$
- $S_i^\alpha = \text{Max} \ (F_{i-1}^\alpha, R(t_i^\alpha))$
Compute $R(t)$ can be complicated

• Single flow: clock ticks when a bit is transmitted. For packet $i$:
  - Round number $\leq$ Arrival time $A_i$
  - $F_i = S_i + P_i = \max(F_{i-1}, A_i) + P_i$

• Multiple flows: clock ticks when a bit from all active flows is transmitted
  - When the number of active flows vary, clock ticks at different speed: $\frac{\partial R}{\partial t} = \frac{1}{N_{ac}(t)}$
An example

- Two flows, unit link speed 1 bit per second
Weighted Fair Queuing

Different queues get different weights
  – Take $w_i$ amount of bits from a queue in each round
  – $F_i = S_i + P_i / w_i$
What is routing?

End-hosts

Routers
The Internet: Zooming In

- **ASes**: Independently owned & operated commercial entities

![Diagram showing BGP connections between Comcast, Abilene, AT&T, Cogent, Duke, and Autonomous Systems (ASes)]
ASes (or domains)

- Autonomously administered
- Economically motivated
- All must cooperate to ensure reachability
- Routing between: BGP
- Routing inside: Up to the AS
  - OSPF, E-IGRP, ISIS (You may have heard of RIP; almost nobody uses it)
- Inside an AS: Independent policies about nearly everything.
Transit ASes vs Stub ASes

All ASes are not equal
AS relationships

• Very complex economic landscape.

• Simplifying a bit:
  – Transit: “I pay you to carry my packets to everywhere” (provider-customer)
  – Peering: “For free, I carry your packets to my customers only.” (peer-peer)

• Technical definition of tier-1 ISP: In the “default-free” zone. No transit.
  – Note that other “tiers” are marketing, but convenient. “Tier 3” may connect to tier-1.
Tier 1 ISP
Default free, Has information on every prefix

Tier 2: Regional/National

Tier 2

Tier 2

Tier 2

Tier 3: Local

Tier 3 (local)
Who pays whom?

• Transit: Customer pays the provider
  – Who is who? Usually, the one who can “live without” the other. AT&T does not need Duke, but Duke needs some ISP.

• What if both need each other? Free Peering.
  – Instead of sending packets over $$ transit, set up a direct connection and exchange traffic for free!
• Tier 1s must all peer with each other by definition
  – Tier 1s form a full mesh Internet core
• Peering *can* give:
  – Better performance
  – Lower cost
  – More “efficient” routing (keeps packets local)
• But negotiating can be very tricky!
Terms

- **Route**: a network prefix plus path attributes
- **Customer/provider/peer routes**: route advertisements heard from customers/providers/peers.
- **Transit service**: If A advertises a route to B, it implies that A will forward packets coming from B to any destination in the advertised prefix.
BGP version 4

• Design goals:
  – Scalability as more networks connect
  – Policy: ASes should be able to enforce business/routing policies
    • Result: Flexible attribute structure, filtering
  – Cooperation under competition:
    • ASes should have great autonomy for routing and internal architecture
    • But BGP should provide global reachability
BGP

Autonomous Systems (ASes)

Route Advertisement

Traffic

Session (over TCP)

BGP peers
• BGP messages
  – OPEN
  – UPDATE
    • Announcements
      – Dest  Next-hop  AS Path … other attributes …
      – 128.2.0.0/16  196.7.106.245  2905 701 1239 5050 9
    • Withdrawals
  – KEEPALIVE
    • Keepalive timer / hold timer

• Key thing: The Next Hop attribute
Path Vector

• ASPATH Attribute
  – Records what ASes a route went through
  – Loop avoidance: Immediately discard
  – Short path heuristics

• Like distance vector, but fixes the count-to-infinity problem
An example of BGP advertisement

- BGP routing table entry for 152.3.0.0/16, version 1009002
- Paths: (36 available, best #10, table default)
- Not advertised to any peer
- Refresh Epoch 1
- 54728 20130 6939 11164 81 13371
  - 140.192.8.16 from 140.192.8.16 (140.192.8.16)
  - Origin IGP, localpref 100, valid, external
    - rx pathid: 0, tx pathid: 0
- Refresh Epoch 1
- 58901 51167 3356 209 81 13371
  - 93.104.209.174 from 93.104.209.174 (93.104.209.174)
  - Origin IGP, localpref 100, valid, external
    - rx pathid: 0, tx pathid: 0
- Refresh Epoch 1
Two Flavors of BGP

• **External BGP (eBGP):** exchanging routes *between* ASes
  – External peers typically directly connected

• **Internal BGP (iBGP):** disseminating routes to external destinations among the routers *within an AS*
  – Internal peers are not
  – Require IGP to find routes
BGP

Autonomous Systems (ASes)

Route Advertisement

Traffic

Session (over TCP)
Enforcing business relationships

- Two mechanisms:
  - Route export filters
    - Control what routes you send to neighbors
  - Route import ranking
    - Controls which route you prefer of those you hear.
Export Policies

• Provider $\rightarrow$ Customer
  – All routes so as to provide transit service

• Customer $\rightarrow$ Provider
  – Only customer routes
  – Why?
  – Only transit for those that pay

• Peer $\rightarrow$ Peer
  – Only customer routes
Import policies

• Same routes heard from providers, customers, and peers, whom to choose?
  – customer > peer > provider
  – Why?
  – Choose the most economic routes!
    • Customer route: charge $$ 😊
    • Peer route: free
    • Provider route: pay $$ ☹️
An annotated AS Graph

Figure 2: An Annotated AS Graph

- Sibling: provide transit services for each other.
  - May belong to the same company
The valley-free property

• Valley-free: After traversing a provider-to-customer or peer-to-peer edge, the AS path cannot traverse a customer-to-provider or peer-to-peer edge
Datacenter Networks

- Topology
- Incast
- DCTCP
Exercise

- 24*10G silicon
- 12-line cards
- 288 port non-blocking switch
Figure 11: Two ways to depopulate the fabric for 50% capacity. Figure 10 also depicts how we started connecting our cluster fabric to the external inter cluster networking. We defer detailed discussion to Section 4.

While Watchtower cluster fabrics were substantially cheaper and of greater scale than anything available for purchase, the absolute cost remained substantial. We used two observations to drive additional cost optimizations. First, there is natural variation in the bandwidth demands of individual clusters. Second, the dominant cost of our fabrics was in the optics and the associated fiber.

Hence, we enabled Watchtower fabrics to support depopulated deployment, where we initially deployed only 50% of the maximum bisection bandwidth. Importantly, as the bandwidth demands of a depop cluster grew, we could fully populate it to 100% bisection in place. Figure 11 shows two high-level options, (A) and (B), to depopulate switches, optics, and fiber, shown in red. (A) achieves 50% capacity by depopulating half of the S2 switches and all fiber and optics touching any depopulated S2 switch. (B) instead depopulates half S3 switches and associated fiber and optics. (A) shows 2x more depopulated elements vs. (B) for the same fabric capacity. (A) requires all spine S3 chassis to be deployed up front even though edge aggregation blocks may be deployed slowly leading to higher initial cost. (B) has a more gradual upfront cost as all spine chassis are not deployed initially. Another advantage of (B) over (A) is that each ToR has twice the burst bandwidth.

In Watchtower and Saturn (Section 3.4) fabrics, we chose option (A) because it maximized cost savings. For Jupiter fabrics (Section 3.5), we moved to option (B) because the upfront cost of deploying the entire spine increased as we moved toward building-size fabrics and the benefits of higher ToR bandwidth became more evident.

3.4 Saturn: Fabric Scaling and 10G Servers

Saturn was the next iteration of our cluster architecture. The principal goals were to respond to continued increases in server bandwidth demands and to further increase maximum cluster scale. Saturn was built from 24x10G merchant silicon building blocks. A 24x10G Pluto ToR Switch and a 12-linecard 288x10G Saturn chassis (including logical topology) built from the same switch chip. Four Saturn chassis housed in two racks cabled with fiber (right).

Figure 12: Components of a Saturn fabric. A 24x10G Pluto ToR Switch and a 12-linecard 288x10G Saturn chassis (including logical topology) built from the same switch chip. Four Saturn chassis housed in two racks cabled with fiber (right).
Summary

• Fundamental design philosophies
• Congestion control
• Routing
• Datacenter networking
• Other modern networking topics
  – SDN, NFV, Programmable Routers, RDMA, Network measurement, DDoS, DHT