CS 356: Computer Network Architectures

Lecture: Internet Quality of Service
[PD] Chapter 6.5

Xiaowei Yang
xwy@cs.duke.edu
# Overview

- Network Resource Allocation
- Congestion Avoidance

- Why QoS?
  - Architectural considerations

- Approaches to QoS
  - Fine-grained: Integrated services
    - RSVP
  - Coarse-grained:
    - Differentiated services
    - Next lecture

Fine-grained: guarantees for each individual flow
Internet Quality of Service
Motivation

• Internet currently provides one single class of “best-effort” service
  – No assurance about delivery

• Many existing applications are elastic
  – Tolerate delays and losses
  – Can adapt to congestion

• “Real-time” applications may be inelastic
Inelastic Applications

• Continuous media applications
  – Lower and upper limit on acceptable performance
  – Below which video and audio are not intelligible
  – Internet telephones, teleconferencing with high delay (200 - 300ms) impair human interactions

• Hard real-time applications
  – Require hard limits on performance
  – E.g., industrial control applications
    • Internet surgery
Design question #1: Why a New Service Model?

- What is the **basic objective** of network design?
  - Maximize total bandwidth? Minimize latency?
    Maximize ISP’s revenues?
  - **the designer’s choice**: Maximize social welfare: the total **utility** given to users (why not profit?)

- What does utility vs. bandwidth look like?
  - Must be non-decreasing function
  - Shape depends on application
Utility Curve Shapes

- Elastic
- Delay-adaptive
- Hard real-time

- Stay to the right and you are fine for all curves
Playback Applications

- Sample signal → packetize → transmit → buffer → playback
  - Fits most multimedia applications

- Performance concern:
  - Jitter: variation in end-to-end delay
    - Delay = fixed + variable = (propagation + packetization) + queuing

- Solution:
  - Playback point – delay introduced by buffer to hide network jitter
Characteristics of Playback Applications

- In general lower delay is preferable

- Doesn’t matter when packet arrives as long as it is before playback point

- Network guarantees (e.g., bound on jitter) would make it easier to set playback point

- Applications can tolerate some loss
Move playback point around based on observed network delay

Gamble that network conditions will be the same as in the past

Are prepared to deal with errors in their estimate

Will have an earlier playback point than rigid applications

How to adapt:
Applications Variations

Really only two classes of applications
1) Intolerant and rigid
2) Tolerant and adaptive

Other combinations make little sense
3) Intolerant and adaptive
   - Cannot adapt without interruption
4) Tolerant and rigid
   - Missed opportunity to improve delay
Design question 2: How to maximize 
\[ V = \sum U(s_i) \]

- Choice #1: add more pipes

- Choice #2: fix the bandwidth but offer different services
  - Q: can differentiated services improve V?
If all users’ utility functions are elastic

Does equal allocation of bandwidth maximize total utility?

- $\sum s_i = B$
- $\text{Max } \sum U(s_i)$
Design question: is Admission Control needed?

- If $U(\text{bandwidth})$ is concave $\rightarrow$ elastic applications
  - Incremental utility is decreasing with increasing bandwidth
    - $U(x) = \log(x^p)$
    - $V = n \log(B/n) \rightarrow \log B^n n^{1-p}$
  - Is always advantageous to have more flows with lower bandwidth
    - No need of admission control;
This is why the Internet works! And fairness makes sense
Utility Curves – Inelastic traffic

Does equal allocation of bandwidth maximize total utility?
Is Admission Control needed?

• If \( U \) is convex \( \rightarrow \) inelastic applications
  – \( U(\text{number of flows}) \) is no longer monotonically increasing
  – Need admission control to maximize total utility

• **Admission control** \( \rightarrow \) deciding when the addition of new people would result in reduction of utility
  – Basically avoids overload
Incentives

- Who should be given what service?
  - Users have incentives to cheat
  - Pricing seems to be a reasonable choice
  - But usage-based charging may not be well received by users
Over provisioning

- Pros: simple
- Cons
  - Not cost effective
  - Bursty traffic leads to a high peak/average ratio
    - E.g., normal users versus leading edge users
  - It might be easier to block heavy users
Comments

- End-to-end QoS has not happened
- Why?
- Can you think of any mechanism to make it happen?
Approaches to QoS

• Fine-grained:
  – Integrated services
    • RSVP

• Coarse-grained:
  – Differentiated services

Fine-grained: guarantees for each individual flow
Components of Integrated Services

1. **Service classes**
   - What does the network promise?

2. **Service interface**
   - How does the application describe what it wants?

3. **Establishing the guarantee**
   - How is the promise communicated to/from the network
   - How is admission of new applications controlled?

4. **Packet scheduling**
   - How does the network meet promises?
1. Service classes

What kind of promises/services should network offer?

Depends on the characteristics of the applications that will use the network....
Service classes

- **Guaranteed** service
  - For *intolerant and rigid* applications
  - Fixed guarantee, network meets commitment as long as clients send at match traffic agreement

- **Controlled load** service
  - For *tolerant and adaptive* applications
  - Emulate lightly loaded networks

- **Datagram/best effort service**
  - Networks do not introduce loss or delay unnecessarily

Two components

If conditions do not change, commit to current service

If conditions change, take steps to deliver consistent performance (help apps minimize playback delay)

Implicit assumption – network does not change much over time
Components of Integrated Services

1. Type of commitment
   What does the network promise?

2. Service interface
   How does the application describe what it wants?

3. Establishing the guarantee
   How is the promise communicated to/from the network
   How is admission of new applications controlled?

4. Packet scheduling
   How does the network meet promises?
Service interfaces

- Flowspecs
  - TSpec: a flow’s traffic characteristics
    - Difficult: bandwidth varies
  - RSpec: the service requested from the network
    - Service dependent
      - E.g. controlled load
A Token Bucket Filter

Tokens enter bucket at rate $r$

Operation:
- If bucket fills, tokens are discarded
- Sending a packet of size $P$ uses $P$ tokens
- If bucket has $P$ tokens, packet sent at max rate, else must wait for tokens to accumulate
Token Bucket Operations

Tokens → Overflow

Tokens ↓ Packet

Enough tokens → packet goes through, tokens removed

Tokens ↓ Packet

Not enough tokens → wait for tokens to accumulate
Token Bucket Characteristics

• In the long run, rate is limited to r
• In the short run, a burst of size b can be sent
• Amount of traffic entering at interval T is bounded by:
  – Traffic = b + r*T
• Information useful to admission algorithm
Token Bucket Specs

Flow A: $r = 1$ MBps, $B=1$ byte
Flow B: $r = 1$ MBps, $B=1$ MB
TSpec

- TokenBucketRate
- TokenBucketSize
- PeakRate
- MinimumPolicedUnit
- MaximumPacketSize
Service Interfaces: RSpec

• Guaranteed Traffic
  – TokenRate and DelayVariation
  – Or DelayVariation and Latency

• Controlled load
  – Type of service
Components of Integrated Services

1. Type of commitment
   What does the network promise?

2. Service interface
   How does the application describe what it wants?

3. Establishing the guarantee
   How is the promise communicated to/from the network
   How is admission of new applications controlled?

4. Packet scheduling
   How does the network meet promises?
Different receivers have different capabilities and want different QOS
Changes in group membership should not be expensive
Reservations should be aggregate – i.e. each receiver in group should not have to reserve
Should be able to switch allocated resource to different senders
RSVP Service Model

- Make reservations for simplex data streams
- Receiver decides whether to make reservation
- Control msgs in IP datagrams (proto #46)
- PATH/RESV sent periodically to refresh soft state

One pass:
Failed requests return error messages - receiver must try again
No e2e ack for success
PATH Messages

• PATH messages carry sender’s Tspec
  — Token bucket parameters

• Routers note the direction PATH messages arrived and set up reverse path to sender

• Receivers send RESV messages that follow reverse path and setup reservations

• If reservation cannot be made, user gets an error
RESV Messages

- Forwarded via reverse path of PATH

- A receiver sends RESV messages
  - TSpec from the sender
  - Rspec
Admission control

- Router performs admission control and reserves resources
  - If request rejected, send error message to receiver
  - Guaranteed service: a yes/no based on available bandwidth
  - Controlled load: heuristics
    - If delay has not exceeded the bound last time after admitting a similar flow, let it in
Soft State to Adapt to Routing Changes

• Problems: Routing protocol makes routing changes

• Solution:
  – PATH and RESV messages sent periodically
  – Non-refreshed state times out automatically

• Ex: a link fails. How is a new reservation established?
Merging multicast reservations

A requests a delay < 100ms
B requests a delay < 200ms

Sender 1
Sender 2
PATH
RESV (merged)
Receiver A
Receiver B
Components of Integrated Services

1. Type of commitment
   What does the network promise?

2. Service interface
   How does the application describe what it wants?

3. Establishing the guarantee
   How is the promise communicated to/from the network
   How is admission of new applications controlled?

4. Packet scheduling
   How does the network meet promises?
Packet classification and scheduling

1. Map a packet to a service class
   - (src addr, dst addr, proto, src port, dst port)

2. Use scheduling algorithms to provide the service
   - An implementation issue
Scheduling for Guaranteed Traffic

- Use WFQ at the routers
  - Q: will DRR work?

- Each flow is assigned to its individual queue

- Parekh’s bound for worst case queuing delay = b/r
  - b = bucket depth
  - r = rate of arrival
Controlled Load Service

Goals:
- Isolation
  - Isolates well-behaved from misbehaving sources
- Sharing
  - Mixing of different sources in a way beneficial to all

Possible Mechanisms:
- WFQ
  - Aggregate multiple flows into one WFQ
Unified Scheduling

- Scheduling: use WFQ in routers

Each guaranteed flow gets its own queue

All controlled load service flows and best effort aggregates in single separate queue

Controlled load classes

Worst case delay for classes separated by order of magnitude

When high priority needs extra bandwidth – steals it from lower class

Best effort traffic acts as lowest priority class
Scalability

• A lot of requests and state!

• ISPs feel it is not the right service model for them!
• Per-flow reservation/queue

  – OC-48 link 2.5Gbps
  – 64Kbps audio stream
  – \( \rightarrow 39,000 \) flows
  – Reservation and state needs to be stored in memory, and 
    refreshed periodically
  – Classify, police, nd queue each flows
Comments on RSVP

• Not widely deployed as a commercial service
• Used for other purposes
  – Setting up MPLS tunnels etc.
Summary

• Why QoS?
  – Architectural considerations

• Approaches to QoS
  – Fine-grained: Integrated services
    • RSVP
  – Coarse-grained:
    • Differentiated services

• Next lecture:
  – DiffServ
  – Net Neutrality

Fine-grained: guarantees for each individual flow
DiffServ
Motivation of DiffServ

- Analogy:
  - Airline service, first class, coach, various restrictions on coach as a function of payment

- Economics and assurances
  - Pay more, and get better service
  - Best-effort expected to make up bulk of traffic,
  - Revenue from first class important to economic base
  - Not motivated by real-time or maximizing social welfare

(will pay for more plentiful bandwidth overall)
Basic Architecture

• Agreements/service provided within a domain
  – Service Level Agreement (SLA) with ISP

• Edge routers do traffic conditioning
  – Shaping, Policing, and Marking

• Core routers
  – Process packets based on packet marking and defined per hop behavior (PHB)

• More scalable than IntServ
  – No per flow state or signaling

Edge routers: Perform per aggregate shaping and policing
Mark packets with a small number of bits; each bit encoding represents a class or subclass
DiffServ Architecture Example

Duke

Shaping, policing, marking

UNC

Per-hop behavior

AT&T
Per-hop Behaviors (PHBs)

- Define behavior of individual routers rather than end-to-end services; there may be many more services than behaviors
  - No end-to-end guarantee

- Multiple behaviors – need more than one bit in the header

- Six bits from IP TOS field are taken for Diffserv code points (DSCP)
Admitted based on peak rate
Unused premium goes to best effort

Based on expected capacity usage profiles
Traffic unlikely to be dropped if user maintains profile
Out-of-profile traffic marked
Expedited Forwarding PHB

• **Goal**: EF packets are forwarded with minimal delay and loss

• **Mechanisms**:
  – User sends within profile and network commits to delivery with requested profile

  – Rate limiting of EF packets at edges only, using token bucket to shape transmission

  – Priority or Weighted Fair Queuing
A congested DS node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value.

Implemented using RED with In/Out bit.

---

**Assured Forwarding PHB**

- **Goal**: good services for in-profile traffic
- **Mechanisms**:
  - User and network agree to some traffic profile
    - How to define profiles is an open/policy issue
  - Edges mark packets up to allowed rate as “in-profile” or low drop precedence
  - Other packets are marked with one of two higher drop precedence values
  - Random Early Detection in/out queues
DiffServ Architecture Example

Duke

UNC

Shaping, policing, marking

Edge

Core

Per-hop behavior

AT&T
Edge Router Input Functionality

Classify packets based on packet header
Traffic Conditioning

- Packet input → Wait for token → Set EF bit → Packet output
- Packet input → Test if token → No token
- Packet input → Set AF “in” bit → Packet output
Router Output Processing

- Two queues: EF packets on higher priority queue
- Lower priority queue implements RED “In or Out” scheme (RIO)
Router Output Processing

- Two queues: EF packets on higher priority queue
- Lower priority queue implements RED “In or Out” scheme (RIO)

![Diagram of Router Output Processing]

- What DSCP?
- EF
- High-priority Q
- Packets out
- AF
- If "in" set incr in_cnt
- Low-priority Q
- RIO queue management
- If "in" set decr in_cnt
Red with In or Out (RIO)

- Similar to RED, but with two separate probability curves
- Has two classes, “In” and “Out” (of profile)
- “Out” class has lower Min$_{thresh}$, so packets are dropped from this class first
  - Based on queue length of all packets
- As avg queue length increases, “in” packets are also dropped
  - Based on queue length of only “in” packets
RIO Drop Probabilities

- $P(\text{drop})$
- $\text{MaxP}$
- $\text{AvgLen}$

- $\text{Min}_{\text{out}}$
- $\text{Min}_{\text{in}}$
- $\text{Max}_{\text{out}}$
- $\text{Max}_{\text{in}}$
Pre-marking and traffic conditioning

Company A
- Packets in premium flows have bit set
- Internal router
- First hop router
- Edge router
- Unmarked packet flow

ISP
- Premium packet flow restricted to R bytes/sec
- Policing
- Edge router
Edge Router Policing

1. Arriving packet
2. Is packet marked?
   - Yes: Forwarding engine
   - No: EF set
3. EF set
4. Token available?
   - Yes: Clear "in" bit
   - No: Drop packet
5. Clear "in" bit
6. No"
Remarks on QoS

• “Dead” at the Internet scale
• Areas of success
  – Enterprise networks
  – Residential uplinks
  – Datacenter networks
Conclusion

• Multicast
  – Service model
  – Sample routing protocols

• QoS
  – Why do we need it?
    – Integrated Services
    – Differentiated Services
      • Motivated by business models