Overview

- What is QoS?
- Queuing discipline and scheduling
- Traffic Enforcement
- Integrated services
What is QoS?

- Current Internet supports best effort packet delivery
  - Sufficient for most applications – "elastic" applications
  - But some applications require or can benefit from a “higher” level of service
- “Higher” quality of service can mean that bounds are provided for one or more performance parameters.
  - Bandwidth: fast data transfers, video
  - Delay, jitter: telephony
  - Packet loss, bit error rate: update services
- QoS can also mean that a user gets “better” treatment (than other users)
  - But no guarantees are given

Why Should we Consider QoS?

- What is the basic objective of network design?
  - Maximize total bandwidth? Minimize latency?
  - Maximize user satisfaction – the total utility given to users
  - Maximize profit?
- What does utility vs. bandwidth look like?
  - Shape depends on application
  - Must be non-decreasing function
Performance versus Satisfaction

User Satisfaction

Unacceptable

No longer Matters

Service Level

Utility Curve Shapes

Elastic

Hard real-time

Delay- or Rate-adaptive

Does equal allocation of bandwidth maximize total utility?

Stay to the right and you are fine for all curves
Utility curve – Elastic traffic

Does equal allocation of bandwidth maximize total utility?

Utility Curves – Inelastic traffic

Does equal allocation of bandwidth maximize total utility?
Inelastic Applications

- Continuous media applications
  - **Lower and upper limit** on acceptable performance.
  - BW below which video and audio are not intelligible
  - Internet telephones, teleconferencing with high delay (200 - 300ms) impair human interaction
  - Sometimes called “tolerant real-time” since they can adapt to the performance of the network

- Hard real-time applications
  - Require **hard limits on performance**
  - E.g. control applications

Quality of Service versus Fairness

- Traditional definition of fairness: treat all users equally.
  - E.g., share bandwidth on bottleneck link equally
- QoS: treat users differently.
  - For example, some users get a bandwidth guarantee, while others have to use best effort service
- The two are not in conflict
  - All else being equal, users are treated equally
  - Unequal treatment is based on policies, price:
    - Administrative policies: rank or position
    - Economics: extra payment for preferential treatment
QoS Analogy: Surface Mail

- The defaults if “first class mail”.
  - Usually gets there within a few days
  - Sufficient for most letters
- Many “guaranteed” mail delivery services: next day, 2-day delivery, next day am, ….
  - Provide faster and more predictable service at a higher cost
  - Providers differentiate their services: target specific markets with specific requirements and budgets
- Why don’t we do the same thing in networks?

How to Provide QoS?

- Admission control limits number of flows
  - You cannot provide guarantees if there are too many flows sharing the same set of resources (bandwidth)
  - For example, telephone networks - busy tone
  - This implies that your request for service can be rejected
- Traffic enforcement limits how much traffic flows can inject based on predefined limits.
  - Make sure user respects the traffic contract
  - Data outside of contract can be dropped (before entering the network!) or can be sent at a lower priority
- Scheduling support in the routers guarantee that users get their share of the bandwidth.
  - Again based on pre-negotiated bounds
What is a flow?

- Defines the granularity of QoS and fairness
  - TCP flow
  - Traffic to or from a device, user, or network
  - Bigger aggregates for traffic engineering purposes
- Flows are defined using a packet classifier
  - Classifier uses a set of fields in the packet header to generate a flow ID
  - Example: (src IP, dest IP, src port, dest port, protocol)
  - Or: (src prefix, dest prfix)
Admission Control

• If $U(\text{bandwidth})$ is concave
  \rightarrow \text{elastic applications}
  • Incremental utility is decreasing with increasing bandwidth
  • Is always advantageous to have more flows with lower bandwidth
    • No need of admission control;
      This is why the Internet works!
• Not so for delay-adaptive and real-time applications

Admission Control

• If $U$ is convex \rightarrow \text{inelastic applications}
  • $U(\text{number of flows})$ is no longer monotonically increasing
  • Need admission control to maximize total utility
• Admission control \rightarrow deciding when adding more people would reduce overall utility
  • E.g., bandwidth or latency guarantees
  • Basically avoids overload
Overview

• What is QoS?
• Queuing discipline and scheduling
• Traffic Enforcement
• Integrated services

Queuing Disciplines

• Each router must implement some queuing discipline
• Queuing allocates both bandwidth and buffer space:
  • Bandwidth: which packet to serve (transmit) next
  • Buffer space: which packet to drop next (when required)
• Queuing also affects latency
### Network Queuing Disciplines

- **First-in-first-out (FIFO) + drop-tail**
  - Simplest choice - used widely in the Internet
  - FIFO means all packets treated equally
  - Drop-tail: new packets gets dropped when queue is
  - Important distinction:
    - FIFO: scheduling discipline
    - Drop-tail: drop policy

- **Alternative is to do Active Queue Management**
  - To improve congestion response
  - Support fairness in presence of non-TCP flows
  - To give flows different types of service – QoS

### Alternative Drop Policies

- Avoid lockout and full queue problems
- Random drop and drop front policies
  - Drop random packet or packet that the head of the queue is full and a new packet arrives
  - Solve the lock-out problem but not the full-queues problem
- Random Early Discard (RED) and Explicit Congestion Notification (ECN) slow down receivers before queues are full
  - See TCP lectures
Problems in Achieving fairness

- In the Internet, fairness is only achieved if all flows play by the rules
- In practice: most sources must use TCP or be “TCP friendly”
  - most sources are cooperative
  - most sources implement homogeneous/compatible control law
    - compatible means less aggressive than TCP
- What if sources do not play by the rule?
  - E.g., TCP versus UDP without congestion control

Fairness Goals

- Allocate resources fairly
- Isolate ill-behaved users
  - Router does not send explicit feedback to source
  - Still needs e2e congestion control
- Still achieve statistical multiplexing
  - One flow can fill entire pipe if no contenders
  - Work conserving → scheduler never idles link if it has a packet
What is Fairness?

- At what granularity?
  - Flows, connections, domains?
- What if users have different RTTs/links/etc.
  - Should it share a link fairly or be TCP fair?
- Maximize fairness index?
  - Fairness = \( \frac{(\sum x_i)^2}{n(\sum x_i^2)} \)  \( 0 < \text{fairness} < 1 \)
  - Basically a tough question to answer!
- Good to separate the design of the mechanisms from definition of a policy
  - User = arbitrary granularity
- We will focus on max-min fairness – just an example

Max-min Fairness

- Allocate user with “small” demand what it wants, evenly divide unused resources to “big” users
- Formally:
  - Resources allocated in terms of increasing demand
  - No source gets resource share larger than its demand
  - Sources with unsatisfied demands get equal share of resource
Implementing Max-min Fairness

- Generalized processor sharing
  - Fluid fairness
  - Bitwise round robin among all queues
- Why not simple round robin?
  - Variable packet length → can get more service by sending bigger packets
  - Unfair instantaneous service rate
    - What if arrive just before/after packet departs?
- We will use bit-bit round robin as an example
  - Many other algorithms exist

Bit-by-bit RR Illustration

- Not feasible to interleave bits on real networks
  - FQ simulates bit-by-bit RR
Fair Queuing

- Mapping bit-by-bit schedule onto packet transmission schedule
- Transmit packet sequentially but in bit RR order
  - How do you compute this packet order?
  - Must be efficient and work for any order

Bit-by-bit RR

- Single flow: clock ticks when a bit is transmitted. For packet \( i \):
  - \( P_i \) = length, \( A_i \) = arrival time, \( S_i \) = begin transmit time, \( F_i \) = finish transmit time
  - \( F_i = S_i + P_i = \max(F_{i-1}, A_i) + P_i \)
- Multiple flows: clock ticks when a bit from all active flows is transmitted \( \rightarrow \) round number
  - Can calculate \( F_i \) for each packet if number of flows is known at all times
    - Need to know flow count to calculate clock tick time
Bit-by-bit RR Example

An Example

- 1 UDP (10 Mbps) and 31 TCPs sharing a 10 Mbps line
Throughput of UDP and TCP Flows With FIFO

Example: Throughput of TCP and UDP Flows With Fair Queueing Router
**Fair Queuing Tradeoffs**

- Complex computation
  - Classification into flows may be hard
  - Must keep queues sorted by finish times
  - \( \frac{dR}{dt} \) changes whenever the flow count changes
- Complex state – must keep queue per flow
  - Hard in routers with many flows (e.g., backbone routers)
  - Flow aggregation is a possibility (e.g., do fairness per domain)
- FQ can control congestion by monitoring flows
- Weighted fair queuing can give flows a different fraction of the bandwidth - controlled by a weight \( W_i \)
  - Bandwidth of flow \( i \) is \( \frac{W_i}{\sum W_j} \)

**WFQ Illustration**

Variation: Weighted Fair Queuing (WFQ)
Overview

- What is QoS?
- Queuing discipline and scheduling
- Traffic Enforcement
- Integrated services

Traffic Enforcement: Token Bucket Filter

**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

Tokens enter bucket at \( r \)

Bucket **depth** \( b \): capacity of bucket
Token Bucket Operation

- Tokens
- Packet
- Overflow

Enough tokens → packet goes through, tokens removed
Not enough tokens → wait for tokens to accumulate

Token Bucket Characteristics

- Can characterize flow using a token bucket: smallest parameters for which no packets will be delayed
- On the long run, rate is limited to \( r \)
- On the short run, a burst of size \( b \) can be sent
- Amount of traffic entering at interval \( T \) is bounded by:
  - \( \text{Traffic} = b + r \times T \)
- Information useful to admission algorithm
**Token Bucket**

- **Parameters**
  - $r$ – average rate, i.e., rate at which tokens fill the bucket
  - $b$ – bucket depth
  - $R$ – maximum link capacity or peak rate (optional parameter)
  - A bit is transmitted only when there is an available token

![Diagram of Token Bucket](image)

**Traffic Enforcement: Example**

- $r = 100$ Kbps; $b = 3$ Kb; $R = 500$ Kbps

- **(a)** $T = 0$: 1Kb packet arrives
- **(b)** $T = 2$ms: packet transmitted
  - $b = 3$Kb – 1Kb + 2ms*100Kbps = 2.2Kb
- **(c)** $T = 4$ms: 3Kb packet arrives
- **(d)** $T = 10$ms: packet transmitted
  - $b = 3$Kb
- **(e)** $T = 16$ms: packet transmitted
  - $b = 0.6$Kb
Token Bucket Specs - Example

Flow A: \( r = 1 \text{ MBps}, B=1 \text{ byte} \)
Flow B: \( r = 1 \text{ MBps}, B=1\text{MB} \)

Overview

- What is QoS?
- Queuing discipline and scheduling
- Traffic Enforcement
- Integrated services
Integrated Services Traffic Classes

• **Guaranteed service**
  • For **hard real-time** applications
  • Fixed guarantee, network meets commitment if clients send at agreed-upon rate

• **Predicted service**
  • For **delay-adaptive** applications
  • 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

• **Datagram/best effort service**

Guarantee Proven by Parekh

• Given:
  • Flow \( i \) shaped with token bucket and leaky bucket rate control (depth \( b \) and rate \( r \))
  • Network nodes do WFQ
  • Admissions control limits number of flows

• Cumulative queuing delay \( D_i \) suffered by flow \( i \) has upper bound
  • \( D_i < b/r \), (where \( r \) may be much larger than average rate)
  • Assumes that \( \Sigma r < \) link speed at any router
  • All sources limiting themselves to \( r \) will result in no network queuing

• Basis for the IETF integrated services standard
Lessons

• What type of applications are there? → Elastic, adaptive real-time, and hard real-time.
• Why do we need admission control → to maximize utility
• How do token buckets + WFQ provide QoS guarantees?