Traffic management

An Engineering Approach to Computer Networking

An example

- Executive participating in a worldwide videoconference
- Proceedings are videotaped and stored in an archive
- Edited and placed on a Web site
- Accessed later by others
- During conference
  - Sends email to an assistant
  - Breaks off to answer a voice call

What this requires

- For video
  - *sustained bandwidth of at least 64 kbps*
  - *low loss rate*
- For voice
  - *sustained bandwidth of at least 8 kbps*
  - *low loss rate*
- For interactive communication
  - *low delay (< 100 ms one-way)*
- For playback
  - *low delay jitter*
- For email and archiving
  - *reliable bulk transport*

What if…

- A million executives were simultaneously accessing the network?
  - What *capacity* should each trunk have?
  - How should packets be *routed*? (Can we spread load over alternate paths?)
  - How can different traffic types get different *services* from the network?
  - How should each endpoint *regulate* its load?
  - How should we *price* the network?

- These types of questions lie at the heart of network design and operation, and form the basis for **traffic management.**

Traffic management

- Set of policies and mechanisms that allow a network to *efficiently* satisfy a *diverse* range of service requests
- Tension is between diversity and efficiency
- Traffic management is necessary for providing *Quality of Service (QoS)*
  - Subsumes congestion control (congestion == loss of efficiency)

Why is it important?

- One of the most challenging open problems in networking
- Commercially important
  - AOL ‘burnout’
  - Perceived reliability (necessary for infrastructure)
  - Capacity sizing directly affects the bottom line
- At the heart of the next generation of data networks
- Traffic management = Connectivity + Quality of Service

Outline

- Economic principles
- Traffic classes
- Time scales
- Mechanisms
- Some open problems

Basics: utility function

- Users are assumed to have a *utility function* that maps from a given quality of service to a level of satisfaction, or utility
  - Utility functions are private information
  - Cannot compare utility functions between users
- *Rational* users take actions that maximize their utility
- Can determine utility function by observing preferences

Example

- Let $u = S - a\,t$
  - u = utility from file transfer
  - S = satisfaction when transfer infinitely fast
  - t = transfer time
  - a = rate at which satisfaction decreases with time
- As transfer time increases, utility decreases
- If $t > S/a$, user is worse off! (reflects time wasted)
- Assumes linear decrease in utility
- S and a can be experimentally determined

Social welfare

- Suppose network manager knew the utility function of every user
- *Social Welfare* is maximized when some combination of the utility functions (such as sum) is maximized
- An economy (network) is *efficient* when increasing the utility of one user must necessarily decrease the utility of another
- An economy (network) is *envy-free* if no user would trade places with another (better performance also costs more)
- Goal: maximize social welfare
  - subject to efficiency, envy-freeness, and making a profit

Example

- Assume
  - Single switch, each user imposes load 0.4
  - A's utility: $4 - d$
  - B's utility : $8 - 2d$
  - Same delay to both users
- Conservation law
  - $0.4d + 0.4d = C \Rightarrow d = 1.25\ C \Rightarrow$ sum of utilities $= 12-3.75\ C$
- If B's delay reduced to $0.5C$, then A's delay = $2C$
  - Sum of utilities $= 12 - 3C$
- Increase in social welfare need not benefit everyone
  - A loses utility, but may pay less for service

Some economic principles

- A single network that provides heterogeneous QoS is better than separate networks for each QoS
  - unused capacity is available to others
- Lowering delay of delay-sensitive traffic increased welfare
  - can increase welfare by matching service menu to user requirements
  - BUT need to know what users want (signaling)
- For typical utility functions, welfare increases more than linearly with increase in capacity
  - individual users see smaller overall fluctuations
  - can increase welfare by increasing capacity

Principles applied

- A single wire that carries both voice and data is more efficient than separate wires for voice and data
  - ADSL
  - IP Phone
- Moving from a 20% loaded10 Mbps Ethernet to a 20% loaded 100 Mbps Ethernet will still improve social welfare
  - increase capacity whenever possible
- Better to give 5% of the traffic lower delay than all traffic low delay
  - should somehow mark and isolate low-delay traffic

The two camps

- Can increase welfare either by
  - matching services to user requirements *or*
  - increasing capacity blindly
- Which is cheaper?
  - no one is really sure!
  - small and smart vs. big and dumb
- It seems that smarter ought to be better
  - otherwise, to get low delays for some traffic, we need to give *all traffic* low delay, even if it doesn't need it
- But, perhaps, we can use the money spent on traffic management to increase capacity
- We will study traffic management, assuming that it matters!

Traffic models

- To align services, need to have some idea of how users or aggregates of users behave = traffic model
  - e.g. how long a user uses a modem
  - e.g. average size of a file transfer
- Models change with network usage
- We can only guess about the future
- Two types of models
  - measurements
  - educated guesses

Telephone traffic models

- How are calls placed?
  - call arrival model
  - studies show that time between calls is drawn from an exponential distribution
  - call arrival process is therefore *Poisson*
  - memoryless: the fact that a certain amount of time has passed since the last call gives no information of time to next call
- How long are calls held?
  - usually modeled as exponential
  - however, measurement studies show it to be *heavy tailed*
  - means that a significant number of calls last a very long time

Internet traffic modeling

- A few apps account for most of the traffic
  - www
  - skype
  - ssh
- A common approach is to model apps (this ignores distribution of destination!)
  - time between app invocations
  - connection duration
  - # bytes transferred
  - packet interarrival distribution
- Little consensus on models
- But two important features

Internet traffic models: features

- LAN connections differ from WAN connections
  - Higher bandwidth (more bytes/call)
  - longer holding times
- Many parameters are heavy-tailed
  - examples
    - # bytes in call
    - call duration
  - means that a *few* calls are responsible for most of the traffic
  - these calls must be well-managed
  - also means that *even aggregates with many calls not be smooth*
  - can have long bursts
- New models appear all the time, to account for rapidly changing traffic mix

Outline

- Economic principles
- Traffic classes
- Time scales
- Mechanisms
- Some open problems

Traffic classes

- Networks should match offered service to source requirements (corresponds to utility functions)
- Example: telnet requires low bandwidth and low delay
  - utility increases with decrease in delay
  - network should provide a low-delay service
  - or, telnet belongs to the low-delay *traffic class*
- Traffic classes encompass both *user requirements* and *network service offerings*

Traffic classes - details

- A basic division: guaranteed service and best effort
  - like flying with reservation or standby
- Guaranteed-service (GS)
  - utility is zero unless app gets a minimum level of service quality
    - bandwidth, delay, loss
  - open-loop flow control with admission control
  - e.g. telephony, remote sensing, interactive multiplayer games
- Best-effort (BE)
  - send and pray
  - closed-loop flow control
  - e.g. email, net news

GS vs. BE (cont.)

- Degree of synchrony
  - time scale at which peer endpoints interact
  - GS are typically *synchronous* or *interactive*
    - interact on the timescale of a round trip time
    - e.g. telephone conversation or telnet
  - BE are typically *asynchronous* or *non-interactive*
    - interact on longer time scales
    - e.g. Email
- Sensitivity to time and delay
  - GS apps are *real-time*
    - performance depends on wall clock
  - BE apps are typically indifferent to real time
    - automatically scale back during overload

Traffic subclasses (roadmap)

- ATM Forum
  - based on sensitivity to bandwidth
  - GS
    - CBR, VBR
  - BE
    - ABR, UBR
- IETF
  - based on sensitivity to delay
  - GS
    - intolerant
    - tolerant
  - BE
    - interactive burst
    - interactive bulk
    - asynchronous bulk

ATM Forum GS subclasses

- Constant Bit Rate (CBR)
  - constant, cell-smooth traffic
  - mean and peak rate are the same
  - e.g. telephone call evenly sampled and uncompressed
  - constant bandwidth, variable quality
- Variable Bit Rate (VBR)
  - long term average with occasional bursts
  - try to minimize delay
  - can tolerate loss and higher delays than CBR
  - e.g. compressed video or audio with constant quality, variable bandwidth

ATM Forum BE subclasses

- Available Bit Rate (ABR)
  - users get whatever is available
  - zero loss if network signals (in RM cells) are obeyed
  - no guarantee on delay or bandwidth
- Unspecified Bit Rate (UBR)
  - like ABR, but no feedback
  - no guarantee on loss
  - presumably cheaper

IETF GS subclasses

- Tolerant GS
  - nominal mean delay, but can tolerate “occasional” variation
  - not specified what this means exactly
  - uses *controlled-load* service
    - book uses older terminology (predictive)
  - even at “high loads”, admission control assures a source that its service “does not suffer”
  - it really is this imprecise!
- Intolerant GS
  - need a worst case delay bound
  - equivalent to CBR+VBR in ATM Forum model

IETF BE subclasses

- Interactive burst
  - bounded asynchronous service, where bound is qualitative, but pretty tight
    - e.g. paging, messaging, email
- Interactive bulk
  - bulk, but a human is waiting for the result
  - e.g. FTP
- Asynchronous bulk
  - junk traffic
  - e.g netnews

Some points to ponder

- The only thing out there is CBR and asynchronous bulk!
- These are application requirements. There are also organizational requirements (link sharing)
- Users needs QoS for other things too!
    - billing
    - privacy
    - reliability and availability

Outline

- Economic principles
- Traffic classes
- Time scales
- Mechanisms
- Some open problems

Time scales

- Some actions are taken once per call
  - tell network about traffic characterization and request resources
  - in ATM networks, finding a path from source to destination
- Other actions are taken during the call, every few round trip times
  - feedback flow control
- Still others are taken very rapidly,during the data transfer
  - scheduling
  - policing and regulation
- Traffic management mechanisms must deal with a range of traffic classes at a range of time scales

Summary of mechanisms at each time scale

- Less than one round-trip-time (cell-level)
  - Scheduling and buffer management
  - Regulation and policing
  - Policy routing (datagram networks)
- One or more round-trip-times (burst-level)
  - Feedback flow control
  - Retransmission
  - Renegotiation

Summary (cont.)

- Session (call-level)
  - Signaling
  - Admission control
  - Service pricing
  - Routing (connection-oriented networks)
- Day
  - Peak load pricing
- Weeks or months
  - Capacity planning

Outline

- Economic principles
- Traffic classes
- Mechanisms at each time scale
  - Faster than one RTT
    - scheduling and buffer management
    - regulation and policing
    - policy routing
  - One RTT
  - Session
  - Day
  - Weeks to months
- Some open problems

Renegotiation

Renegotiation

- An option for guaranteed-service traffic
- Static descriptors don't make sense for many real traffic sources
  - interactive video
- Multiple-time-scale traffic
  - burst size B that lasts for time T
  - for zero loss, descriptors (P,0), (A, B)
    - P = peak rate, A = average
  - T large => serving even slightly below P leads to large buffering requirements
  - one-shot descriptor is inadequate

Renegotiation (cont.)

- Renegotiation matches service rate to traffic
- Renegotiating service rate about once every ten seconds is sufficient to reduce bandwidth requirement nearly to average rate
  - works well in conjunction with optimal smoothing
- Fast buffer reservation is similar
  - each burst of data preceded by a reservation
- Renegotiation is not free
  - signaling overhead
  - call admission ?
    - perhaps measurement-based admission control

RCBR

- Extreme viewpoint
- All traffic sent as CBR
- Renegotiate CBR rate if necessary
- No need for complicated scheduling!
- Buffers at edge of network
  - much cheaper
- Easy to price
- Open questions
  - when to renegotiate?
  - how much to ask for?
  - admission control
  - what to do on renegotiation failure

Outline

- Economic principles
- Traffic classes
- Mechanisms at each time scale
  - Faster than one RTT
  - One RTT
  - Session
    - Signaling
    - Admission control
  - Day
  - Weeks to months
- Some open problems

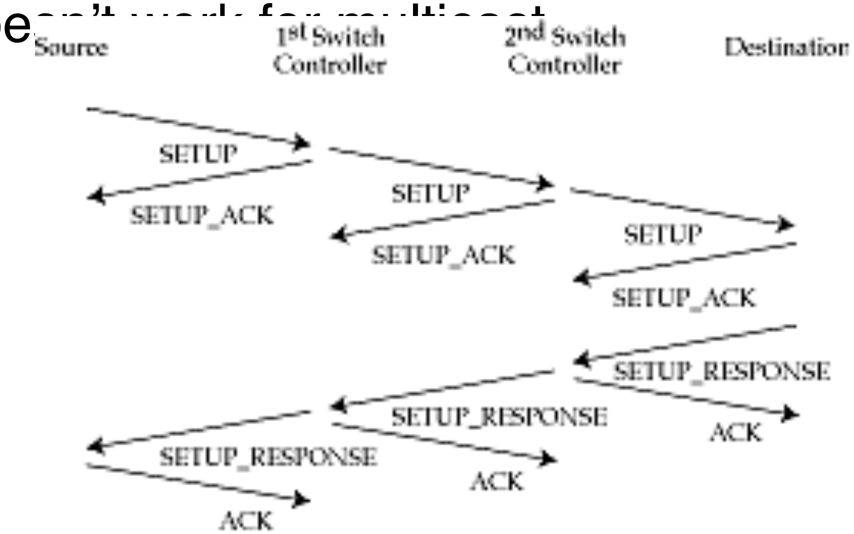
Signaling

Signaling

- How a source tells the network its utility function
- Two parts
  - how to carry the message (transport)
  - how to interpret it (semantics)
- Useful to separate these mechanisms

Signaling semantics

- Classic scheme: sender initiated
- SETUP, SETUP_ACK, SETUP_RESPONSE
- Admission control
- Tentative resource reservation and confirmation
- Simplex and duplex setup
- Doesn't work for multicast



Resource translation

- Application asks for end-to-end quality
- How to translate to per-hop requirements?
  - E.g. end-to-delay bound of 100 ms
  - What should be bound at each hop?
- Two-pass
  - forward: maximize (denial!)
  - reverse: relax
  - open problem!

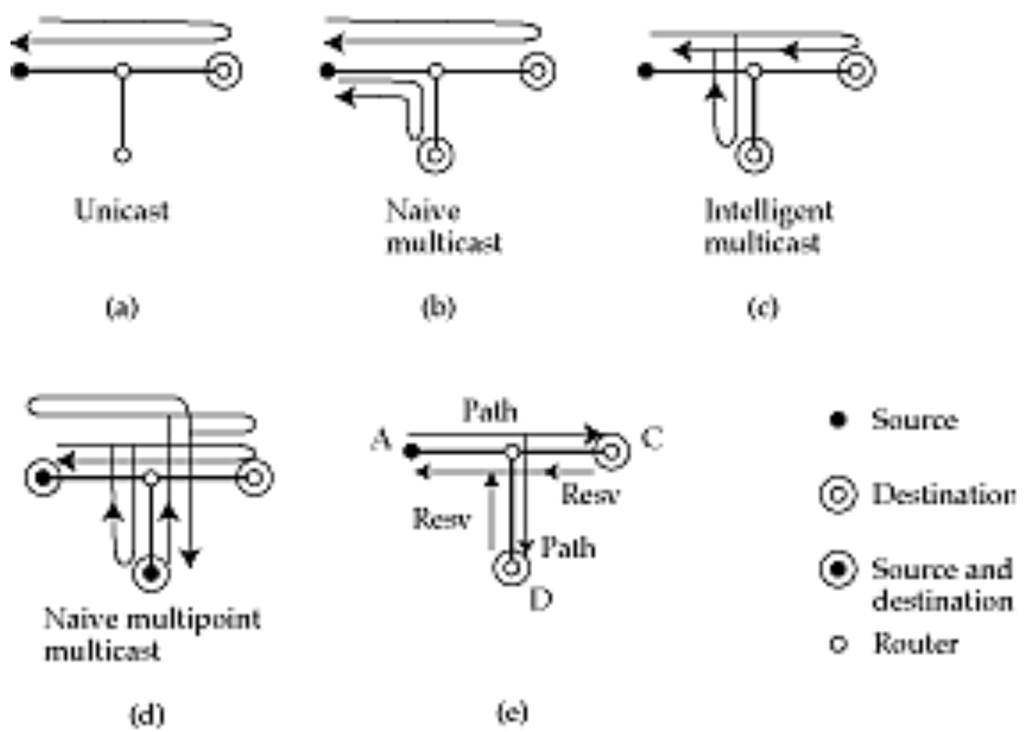
Signaling: transport

- Telephone network uses Signaling System 7 (SS7)
  - Carried on Common Channel Interoffice Signaling (CCIS) network
  - CCIS is a datagram network
  - SS7 protocol stack is loosely modeled on ISO (but predates it)
- Signaling in ATM networks uses Q.2931 standard
  - part of User Network Interface (UNI)
  - complex
  - layered over SSCOP (a reliable transport protocol) and AAL5

Internet signaling transport: RSVP

- Main motivation is to efficiently support multipoint multicast with resource reservations
- Progression
  - Unicast
  - Naïve multicast
  - Intelligent multicast
  - Naïve multipoint multicast
  - RSVP

RSVP motivation



Multicast reservation styles

- Naïve multicast (source initiated)
  - source contacts each receiver in turn
  - wasted signaling messages
- Intelligent multicast (merge replies)
  - two messages per link of spanning tree
  - source needs to know all receivers
  - and the rate they can absorb
  - doesn't scale
- Naïve multipoint multicast
  - two messages per source per link
  - can't share resources among multicast groups

RSVP

- Receiver initiated
- Reservation state per group, instead of per connection
- PATH and RESV messages
- PATH sets up next hop towards source(s)
- RESV makes reservation
- Travel as far back up as necessary
  - how does receiver know of success?

Filters

- Allow receivers to separate reservations
- Fixed filter
  - receive from eactly one source
- Dynamic filter
  - dynamically choose which source is allowed to use reservation

Soft state

- State in switch controllers (routers) is periodically refreshed
- On a link failure, automatically find another route
- Transient!
- But, probably better than with ATM

Why is signaling hard ?

- Complex services
- Feature interaction
  - call screening + call forwarding
- Tradeoff between performance and reliability
- Extensibility and maintainability

Outline

- Economic principles
- Traffic classes
- Mechanisms at each time scale
  - Faster than one RTT
  - One RTT
  - Session
    - Signaling
    - Admission control
  - Day
  - Weeks to months
- Some open problems

Admission control

Admission control

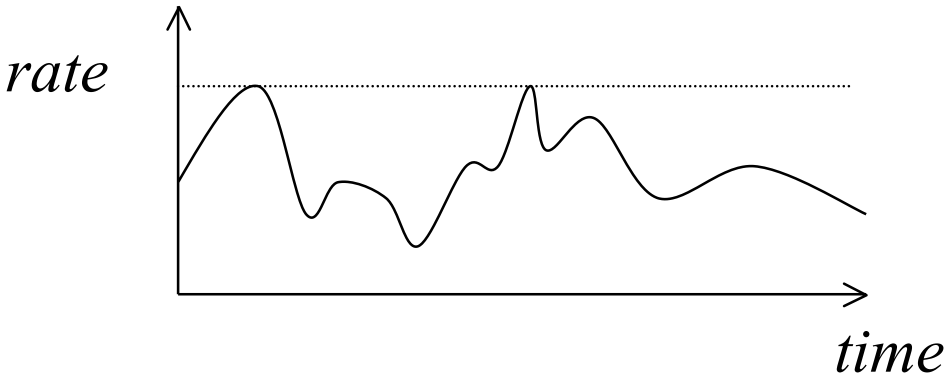
- Can a call be admitted?
- CBR admission control
  - simple
  - on failure: try again, reroute, or hold
- Best-effort admission control
  - trivial
  - if minimum bandwidth needed, use CBR test

VBR admission control

- VBR
  - peak rate differs from average rate = *burstiness*
  - if we reserve bandwidth at the peak rate, wastes bandwidth
  - if we reserve at the average rate, may drop packets during peak
  - key decision: how much to overbook
- Four known approaches
  - peak rate admission control
  - worst-case admission control
  - admission control with statistical guarantees
  - measurement-based admission control

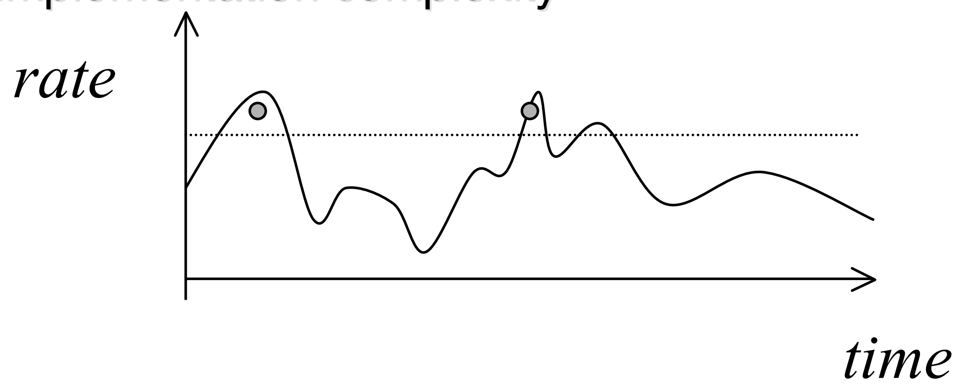
1. Peak-rate admission control

- Reserve at a connection's peak rate
- Pros
  - simple (can use FIFO scheduling)
  - connections get zero (fluid) delay and zero loss
  - works well for a small number of sources
- Cons
  - wastes bandwidth
  - peak rate may increase because of scheduling jitter



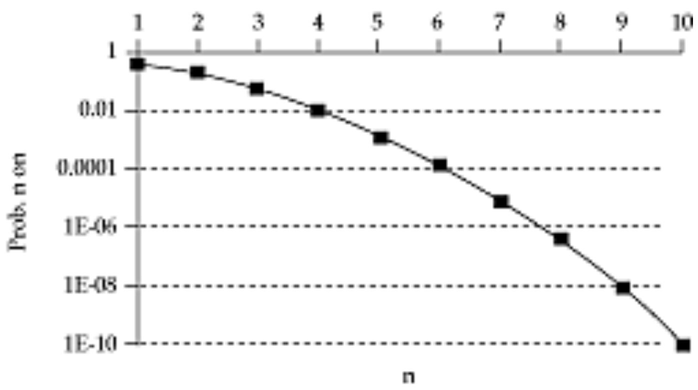
2. Worst-case admission control

- Characterize source by ‘average’ rate and burst size (LBAP)
- Use WFQ or rate-controlled discipline to reserve bandwidth at average rate
- Pros
  - may use less bandwidth than with peak rate
  - can get an end-to-end delay guarantee
- Cons
  - for low delay bound, need to reserve at more than peak rate!
  - implementation complexity



3. Admission with statistical guarantees

- Key insight is that as # calls increases, probability that multiple sources send a burst decreases
  - sum of connection rates is increasingly smooth
- With enough sources, traffic from each source can be assumed to arrive at its average rate
- Put in enough buffers to make probability of loss low



3. Admission with statistical guarantees (contd.)

- Assume that traffic from a source is sent to a buffer of size *B* which is drained at a constant rate *e*
- If source sends a burst, its delay goes up
- If the burst is too large, bits are lost
- *Equivalent bandwidth* of the source is the rate at which we need to drain this buffer so that the probability of loss is less than *l* and the delay in leaving the buffer is less than *d*
- If many sources share a buffer, the equivalent bandwidth of each source decreases (why?)
- Equivalent bandwidth of an ensemble of connections is the sum of their equivalent bandwidths

3. Admission with statistical guarantees (contd.)

- When a source arrives, use its performance requirements and current network state to assign it an equivalent bandwidth
- Admission control: sum of equivalent bandwidths at the link should be less than link capacity
- Pros
  - can trade off a small loss probability for a large decrease in bandwidth reservation
  - mathematical treatment possible
  - can obtain delay bounds
- Cons
  - assumes uncorrelated sources
  - hairy mathematics

4. Measurement-based admission

- For traffic that cannot describe itself
  - also renegotiated traffic
- *Measure* ‘real’ average load
- Users tell peak
- If peak + average < capacity, admit
- Over time, new call becomes part of average
- Problems:
  - assumes that past behavior is indicative of the future
  - how long to measure?
  - when to forget about the past?

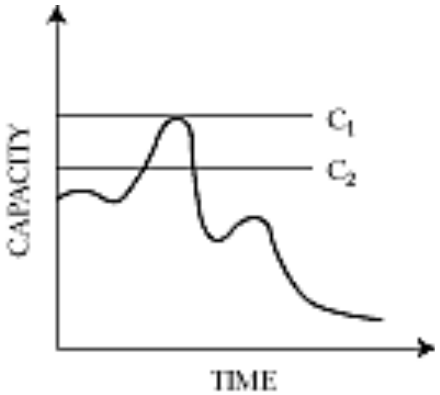
Outline

- Economic principles
- Traffic classes
- Mechanisms at each time scale
  - Faster than one RTT
  - One RTT
  - Session
  - Day
  - Weeks to months
- Some open problems

Peak load pricing

Problems with cyclic demand

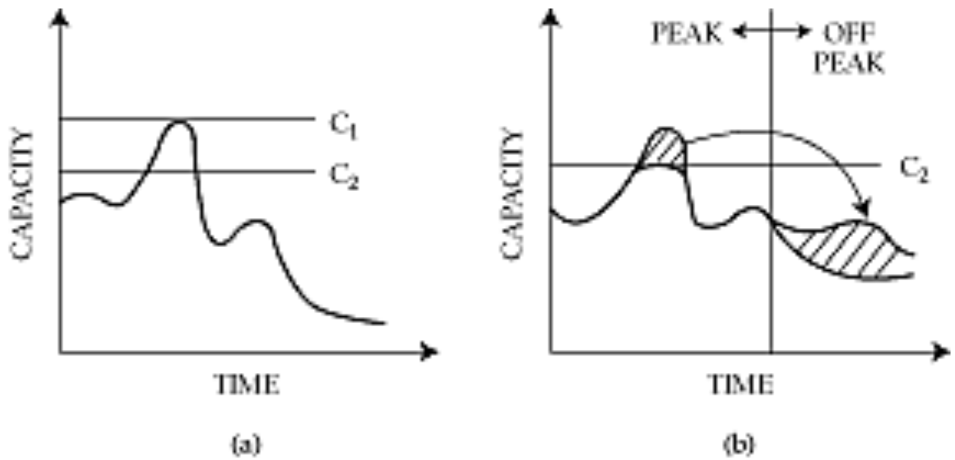
- Service providers want to
  - avoid overload
  - use all available capacity
- Hard to do both with cyclic demand
  - if capacity C1, then waste capacity
  - if capacity C2, overloaded part of the time



(a)

Peak load pricing

- Traffic shows strong daily peaks => cyclic demand
- Can shift demand to off-peak times using pricing
- Charge more during peak hours
  - price is a *signal* to consumers about network preferences
  - helps both the network provider and the user



Example

- Suppose
  - network capacity = C
  - peak demand = 100, off peak demand = 10
  - user's utility = -total price - overload
  - network's utility = revenue - idleness
- Price = 1 per unit during peak and off peak times
  - revenue = 100 + 10 = 110
  - user's utility = -110 -(100-C)
  - network's utility = 110 - (C - off peak load)
  - e.g if C = 100, user's utility = -110, network's utility = 20
  - if C = 60, user's utility = -150, network's utility = 60
  - increase in user's utility comes as the cost of network's utility

Example (contd.)

- Peak price = 1, off-peak price = 0.2
- Suppose this decreases peak load to 60, and off peak load increases to 50
- Revenue = 60*1 + 50*0.2 = 70
  - lower than before
- But peak is 60, so set C = 60
- User’s utility = -70 (greater than before)
- Network’s utility = 60 (same as before)
- Thus, with peak-load pricing, user’s utility increases at no cost to network
- Network can gain some increase in utility while still increasing user’s utility

Lessons

- Pricing can control user's behavior
- Careful pricing helps both users and network operators
- Pricing is a *signal* of network's preferences
- Rational users help the system by helping themselves

Outline

- Economic principles
- Traffic classes
- Mechanisms at each time scale
  - Faster than one RTT
  - One RTT
  - Session
  - Day
  - Weeks to months
- Some open problems

Capacity planning

Capacity planning

- How to modify network topology, link capacity, and routing to most efficiently use existing resources, or alleviate long-term congestion
- Usually a matter of trial and error
- A more systematic approach:
  - measure network during its busy hour
  - create traffic matrix
  - decide topology
  - assign capacity

1. Measure network during busy hour

- Traffic ebbs and flows during day and during week
- A good rule of thumb is to build for the worst case traffic
- Measure traffic for some period of time, then pick the busiest hour
- Usually add a fudge factor for future growth
- Measure bits sent from each endpoint to each endpoint
  - we are assuming that endpoint remain the same, only the internal network topology is being redesigned

2. Create traffic matrix

- # of bits sent from each source to each destination
- We assume that the pattern predicts future behavior
  - probably a weak assumption
    - what if a web site suddenly becomes popular!
- Traffic over shorter time scales may be far heavier
- Doesn’t work if we are adding a new endpoint
  - can assume that it is similar to an existing endpoint

3. Decide topology

- Topology depends on three considerations
  - *k*-connectivity
    - path should exist between any two points despite single node or link failures
  - geographical considerations
    - some links may be easier to build than others
  - existing capacity

4. Assign capacity

- Assign sufficient capacity to carry busy hour traffic
- Unfortunately, actual path of traffic depends on routing protocols which measure instantaneous load and link status
- So, we cannot directly influence path taken by traffic
- Circular relationship between capacity allocation and routing makes problem worse
  - higher capacity link is more attractive to routing
  - thus carries more traffic
  - thus requires more capacity
  - and so on…
- Easier to assign capacities if routing is *static* and links are always up (as in telephone network)

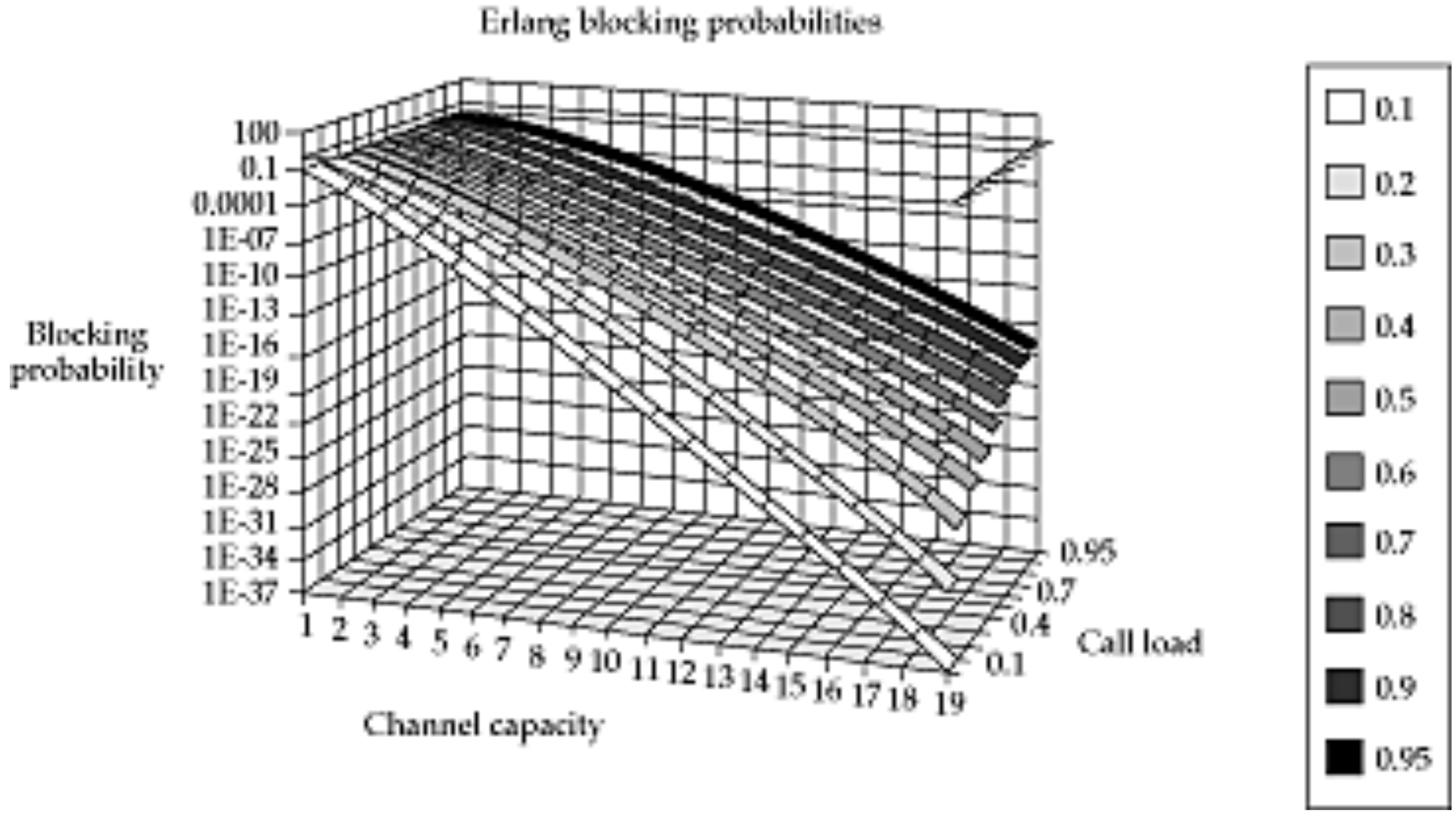
Telephone network capacity planning

- How to size a link so that the call blocking probability is less than a target?
- Solution due to Erlang (1927)
- Assume we know mean # calls on a trunk (in erlangs)
- Mean call arrival rate = l
- Mean call holding time = m
- Then, call load A = lm
- Let trunk capacity = N, infinite # of sources
- Erlang’s formula gives blocking probability
  - e.g. N = 5, A = 3, blocking probability = 0.11
- For a fixed load, as N increases, the call blocking probability decreases exponentially

Recall erlang eqn

$$E(v,C) = \frac{v^c}{C!}\left[\sum_{i=0}^{C}\frac{v^i}{i!}\right]^{-1}$$

Sample Erlang curves



Capacity allocation

- Blocking probability along a path
- Assume traffic on links is independent
- Then, probability is product of probability on each link
- Routing table + traffic matrix tells us load on a link
- Assign capacity to each link given load and target blocking probability
- Or, add a new link and change the routing table

Capacity planning on the Internet

- Trial and error
- Some rules of thumb help
- In 2000, measurements indicate that sustained bandwidth per active user is about 50 Kbps
  - add a fudge factor of 2 to get 100 Kbps
- During busy hour, about 40% of potential users are active
- So, a link of capacity C can support 2.5C/100 Kbps users
- e.g. 100 Mbps backbone could support 2500 users

- Now - PON with 10GigE per 1000 users:)

Capacity planning on the Internet

- About 10% of campus traffic enters the Internet
- A 2500-person campus usually uses a 100Mbps and a 25,000-person campus a 1Gbps
- Why?
  - regional and backbone providers throttle traffic using pricing
  - Restricts higher rate to a few large customers
- Regionals and backbone providers buy the fastest links they can
- Try to get a speedup of 10-30 over individual access links

Problems with capacity planning

- Routing and link capacity interact
- Measurements of traffic matrix
- Survivability

Outline

- Economic principles
- Traffic classes
- Mechanisms at each time scale
- Some open problems

Some open problems

Six open problems

- Resource translation
- Renegotiation
- Measurement-based admission control
- Peak-load pricing
- Capacity planning
- A metaproblem

1. Resource translation

- Application asks for end-to-end quality in terms of bandwidth and delay
- How to translate to resource requirements in the network?
- **Bandwidth is relatively easy, delay is hard**
- One approach is to translate from delay to an equivalent bandwidth
  - can be inefficient if need to use worst case delay bound
  - average-case delay usually requires strong source characterization
- Other approach is to directly obtain per-hop delay bound (for example, with EDD scheduling)
- How to translate from end-to-end to per-hop requirements?
  - Two-pass heuristic

2. Renegotiation

- Static descriptors don't make sense for interactive sources or multiple-time scale traffic
- Renegotiation matches service rate to traffic
- Renegotiation is not free- incurs a signaling overhead
- Open questions
  - when to renegotiate?
  - how much to ask for?
  - admission control?
  - what to do on renegotiation failure?

3. Measurement based admission

- For traffic that cannot describe itself
    - also renegotiated traffic
- Over what time interval to measure average?
- How to describe a source?
- How to account for nonstationary traffic?
- Are there better strategies?

4. Peak load pricing

- How to choose peak and off-peak prices?
- When should peak hour end?
- What does peak time mean in a global network?

5. Capacity planning

- Simultaneously choosing a topology, link capacity, and routing metrics
- But routing and link capacity interact
- What to measure for building traffic matrix?
- How to pick routing weights?
- Heterogeneity?

6. A metaproblem

- Can increase user utility either by
  - service alignment *or*
  - overprovisioning
- Which is cheaper?
  - no one is really sure!
  - small and smart vs. big and dumb
- It seems that smarter ought to be better
  - for example, to get low delays for telnet, we need to give *all traffic* low delay, even if it doesn't need it
- But, perhaps, we can use the money spent on traffic management to increase capacity!
- Do we really need traffic management?

Macroscopic QoS

- Three regimes
  - scarcity - micromanagement
  - medium - generic policies
  - plenty - are we there yet?
- Example: video calls
- Take advantage of law of large numbers
- Learn from the telephone network