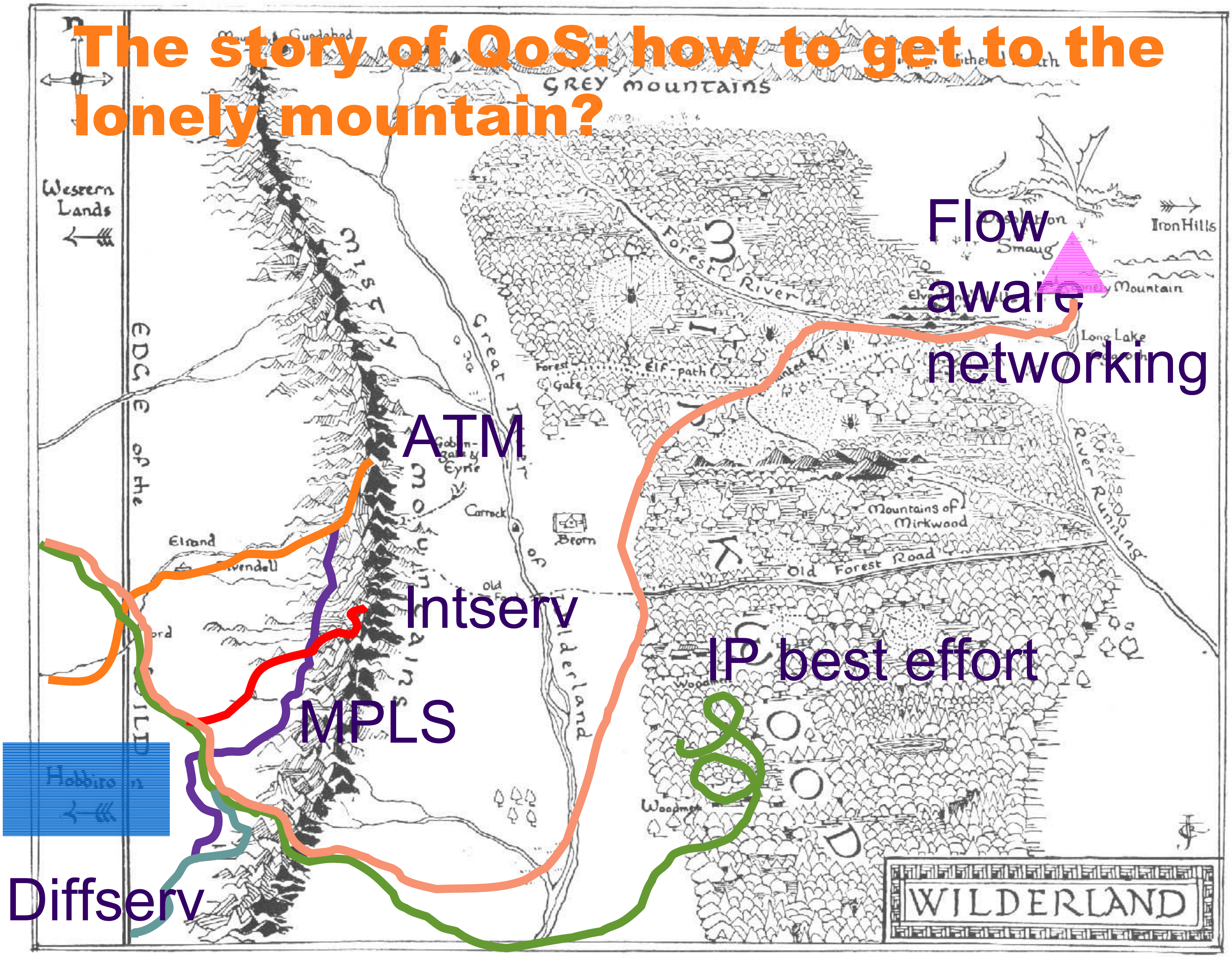
Networking 2004
Athens
11 May 2004

france telecom
R&D

From ATM to IP and back again: the label switched path to the converged Internet, or another blind alley?

Jim Roberts
France Telecom R&D

The story of QoS: how to get to the lonely mountain?



Flow aware networking

ATM

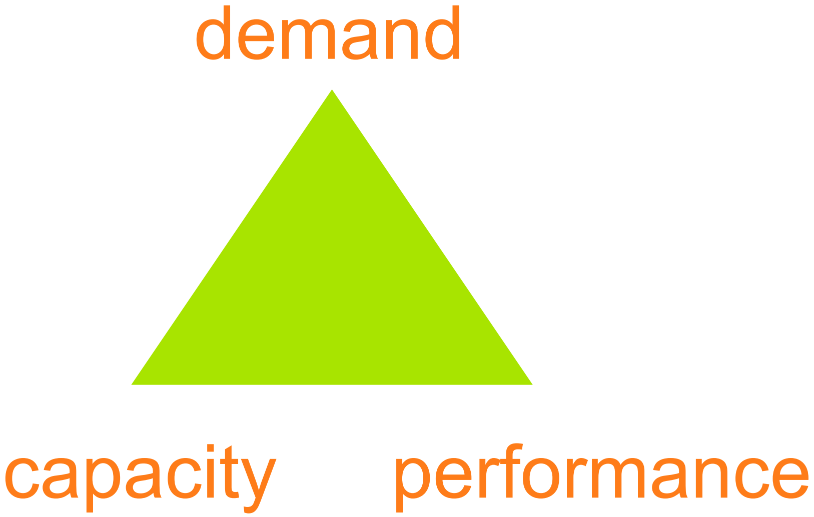
Intserv

MPLS

IP best effort

Diffserv

The traffic - performance relation



demand

capacity

performance

Understanding the traffic-performance relation: the key to QoS

- essential for sizing
  - how much capacity to satisfy demand
- essential for network design
  - how to share network capacity

demand

- volume
- charactéristics

capacity

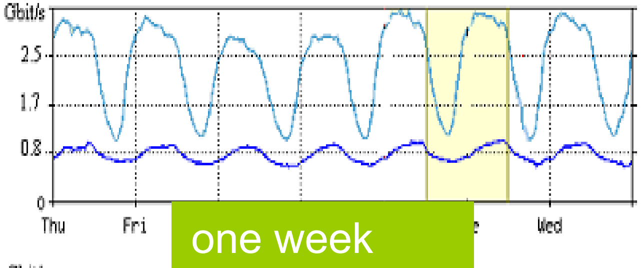
- bandwidth
- how to share it

performance

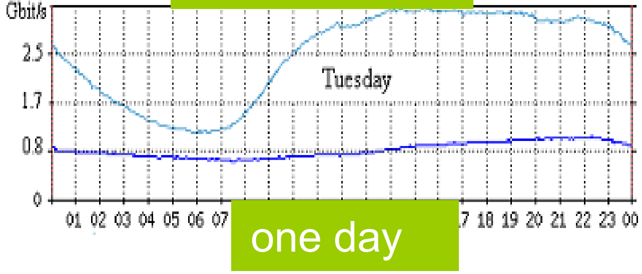
- response time
- packet delays, loss

Modelling IP traffic

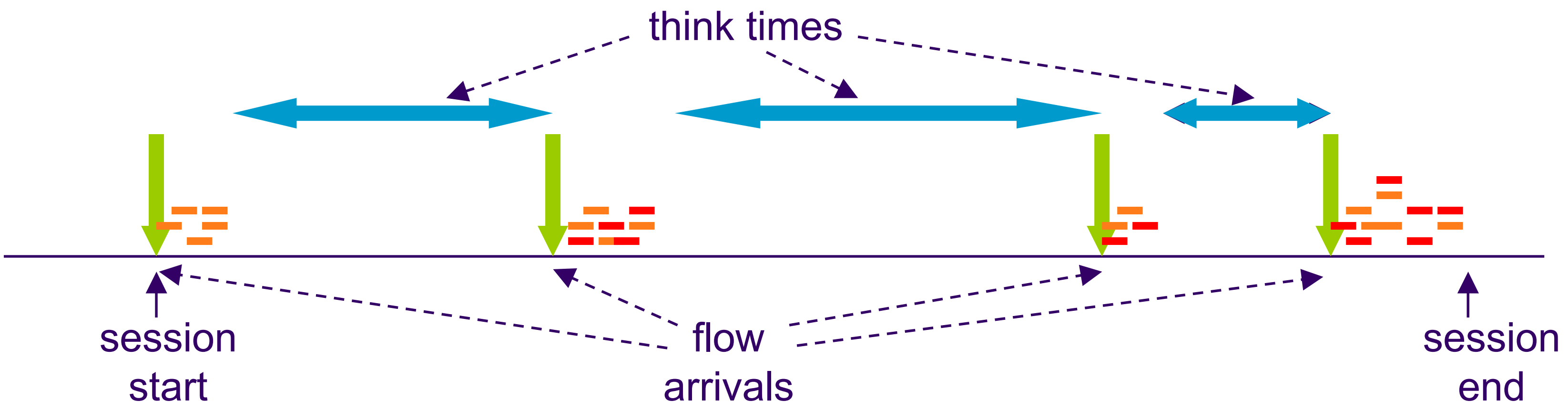
- a stationary process...
  - in the busy period
- demand (bit/sec) = arrival rate x mean size...
  - ... of sessions, flows or packets
- sessions arrive as Poisson process
  - and generate a series of flows and think times



one week



one day



think times

session start

flow arrivals

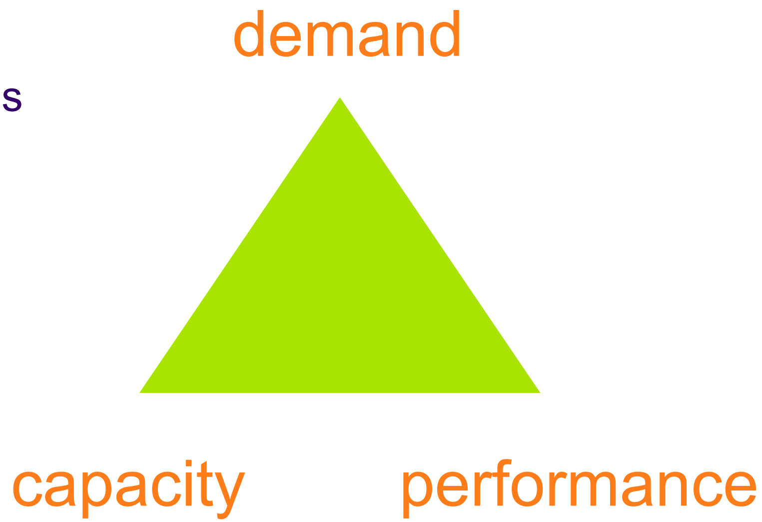
session end

A robust traffic classification

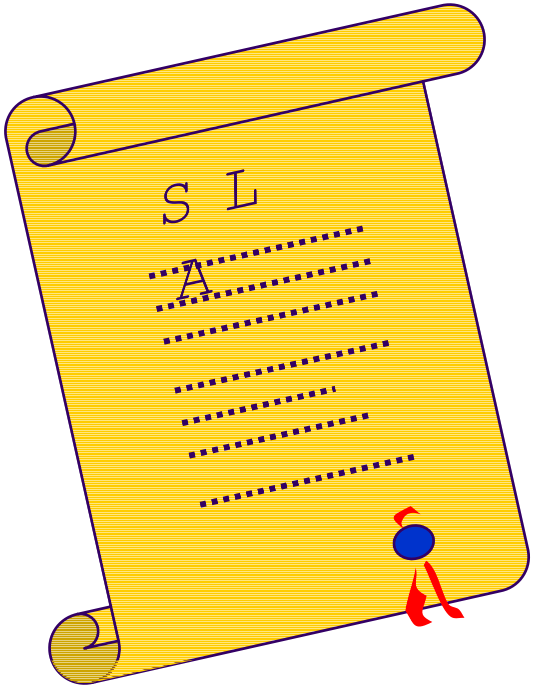
- streaming flows
  - real time voice and video applications (and gaming...)
  - signal conservation: negligible delay and loss
- elastic flows
  - document transfers
  - throughput conservation: negligible rate reduction
- currently, 90% of IP traffic is elastic
  - (except in Korea?)

Results on the traffic-performance relation

- traffic theory for streaming traffic
  - buffered or bufferless statistical multiplexing
  - admission control
  - packet and flow level performance
- traffic theory for elastic traffic
  - statistical bandwidth sharing
  - admission control
  - response times and blocking probabilities
- the basis for sound engineering



The failure of the traffic contract



QoS and the "traffic contrat"

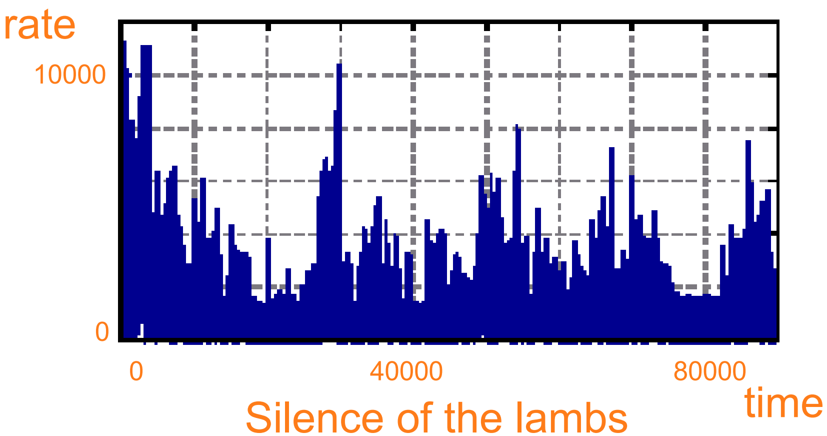
- a contract in three stages:
  - the user specifies its traffic and performance requirements
  - the network applies admission control
  - if admitted, the user's traffic is policed, or resources are explicitly allocated in router queues
- a widely used notion in Intserv, Diffserv, MPLS TE...
  - ... as well as ATM, Frame Relay
  - for microflows, tunnels, aggregates
- but what traffic descriptor for variable rate traffic?
  - it must be "understandable, useful, verifiable" (cf. ITU Rec I.371)
  - NB. the leaky bucket is *verifiable* but neither *understandable* nor *useful*

Trafic descriptors for variable rate flows?

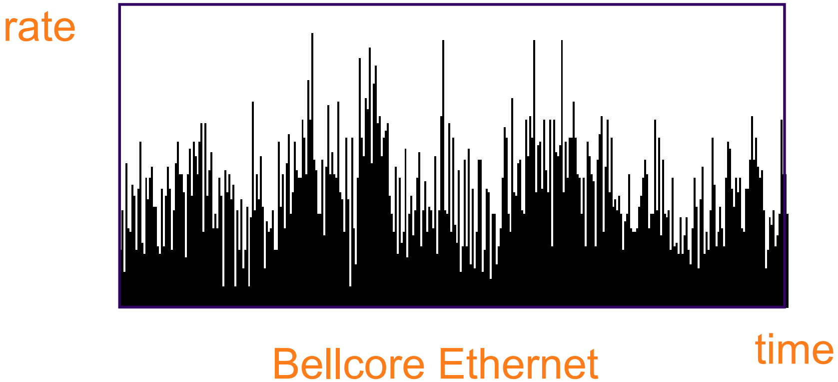
- streaming flows
  - e.g., an MPEG 4 video
  - "self-similar" variations

- aggregates of elastic flows
  - e.g., LAN traffic
  - "self-similar" variations

- *a priori* characterization is impossible
  - e.g., by a leaky bucket
  - ⇒ rate overestimation



Silence of the lambs



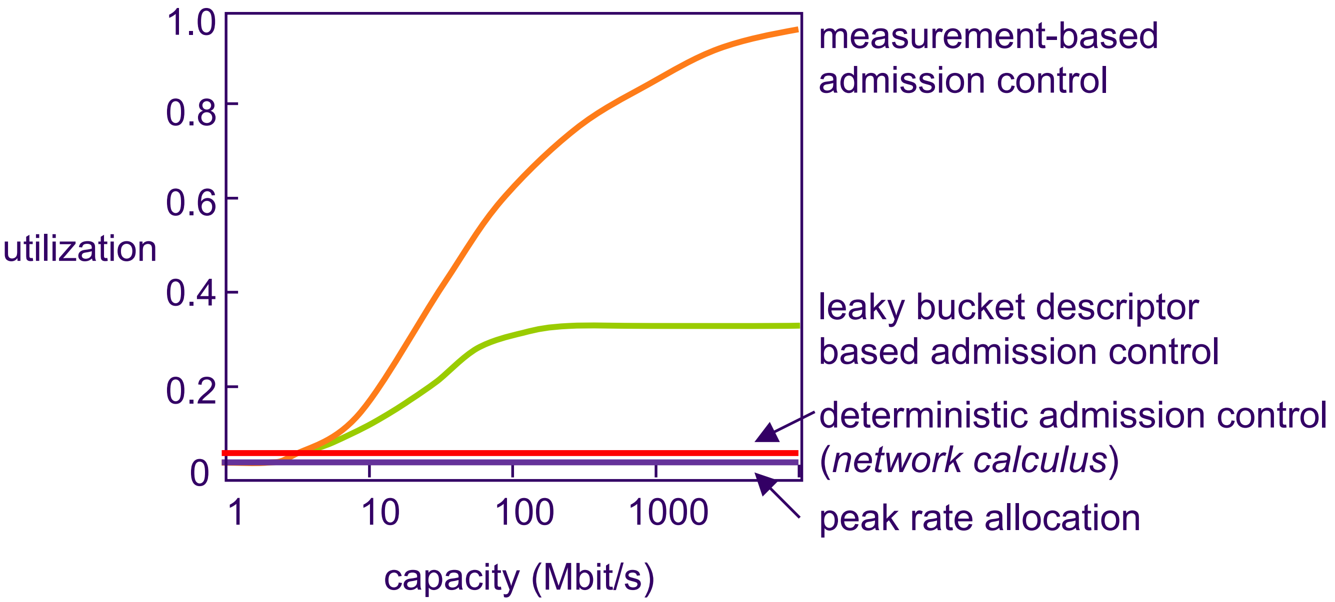
Bellcore Ethernet

QoS and the "traffic contrat"

- a contract in three stages:
  - the user specifies its traffic and performance requirements
  - the network applies admission control
  - if admitted, the user's traffic is policed, or resources are explicitly allocated in router queues
- a widely used notion in Intserv, Diffserv, MPLS TE...
  - ... as well as ATM, Frame Relay
  - for microflows, tunnels, aggegates
- but what traffic descriptor for variable rate traffic?
  - it must be "understandable, useful, verifiable" (cf. ITU Rec I.371)
  - NB. the leaky bucket is *verifiable* but neither *understandable* nor *useful*
- and how to perform admission control?
  - only admit a new demand if performance requirements satisfied
  - using a traffic descriptor ... or by traffic measurement?

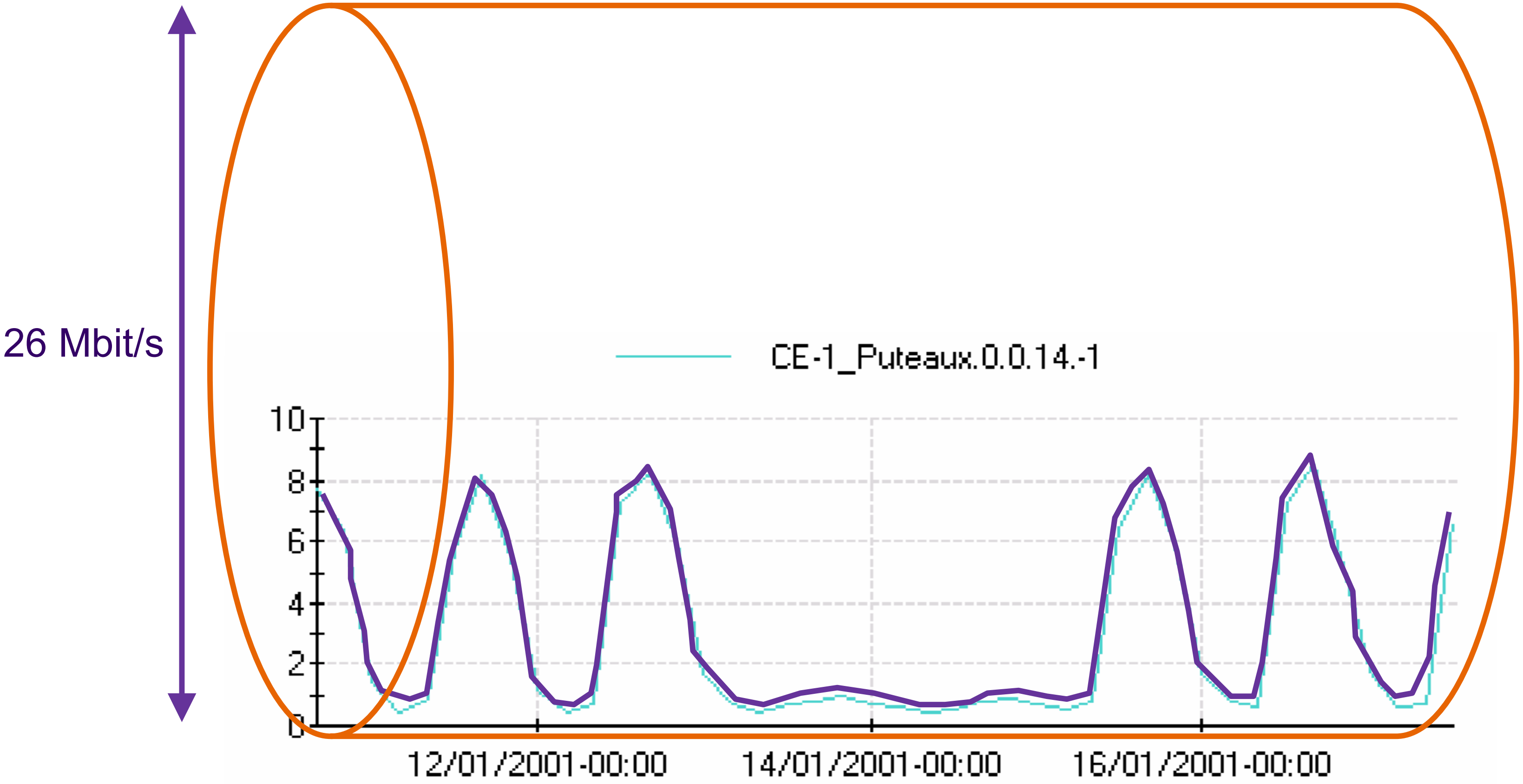
Admission control: a case study

- for flows of peak rate 1,5 Mbit/s and mean rate 50 Kbit/s...
  - on/off sources, exponential bursts and silences
  - performance required: delay < 50 ms
- ... policed by a leaky bucket of rate 150 Kbit/s
  - for a low probability of non-conformance (10^{-6})



Over-provisioning or under-provisioning?

- traffic measured on a VBR ATM trunk with sustainable rate 26 Mb/s
- over-booking is necessary, but by what factor? what QoS guarantees?



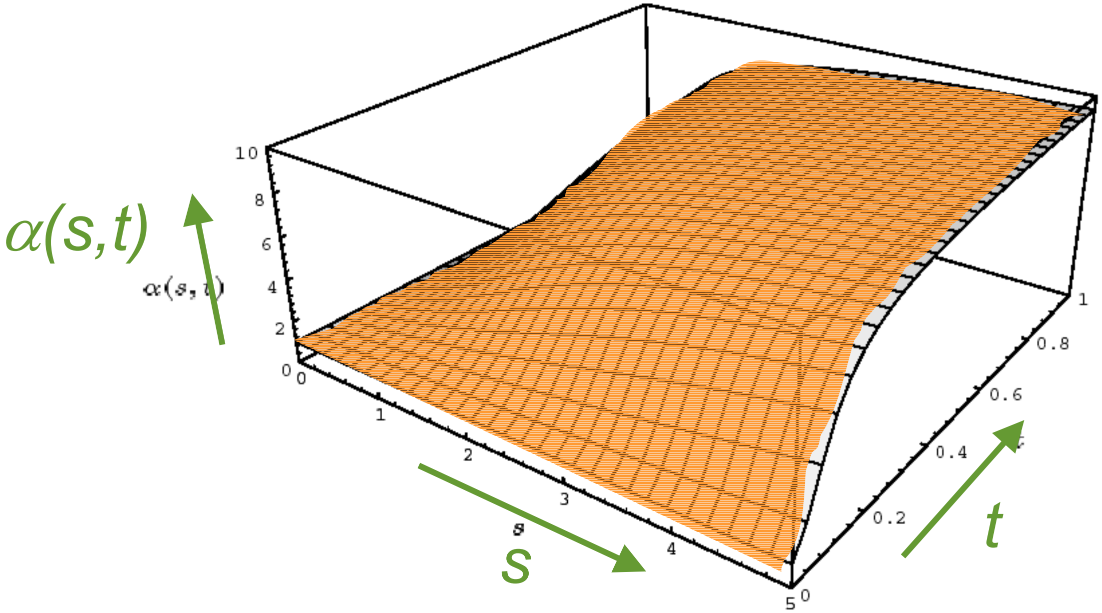
Current prospects for QoS

- rely on over-provisioning
    - over-provision for reliability, no need for QoS mechanisms
    - but what is over-provisioning? how much extra?
- MPLS traffic engineering
    - create "traffic trunks" (virtual circuits with capacity attributes)
    - "*For the purpose of bandwidth allocation, a single canonical value of bandwidth requirements can be computed from a traffic trunk's traffic parameters. Techniques for performing these computations are well known. One example of this is the theory of effective bandwidth*" (RFC 2702).

Effective bandwidth (Kelly 1996)

- ➔ effective bandwidth is a **function:** $\alpha(s,t) = \frac{1}{st} \log E[\exp\{sA(0,t)\}]$
  - $A(0,t)$ = traffic arriving in $(0,t)$
- ➔ it is **not** a canonical value

Notes on Effective Bandwidths

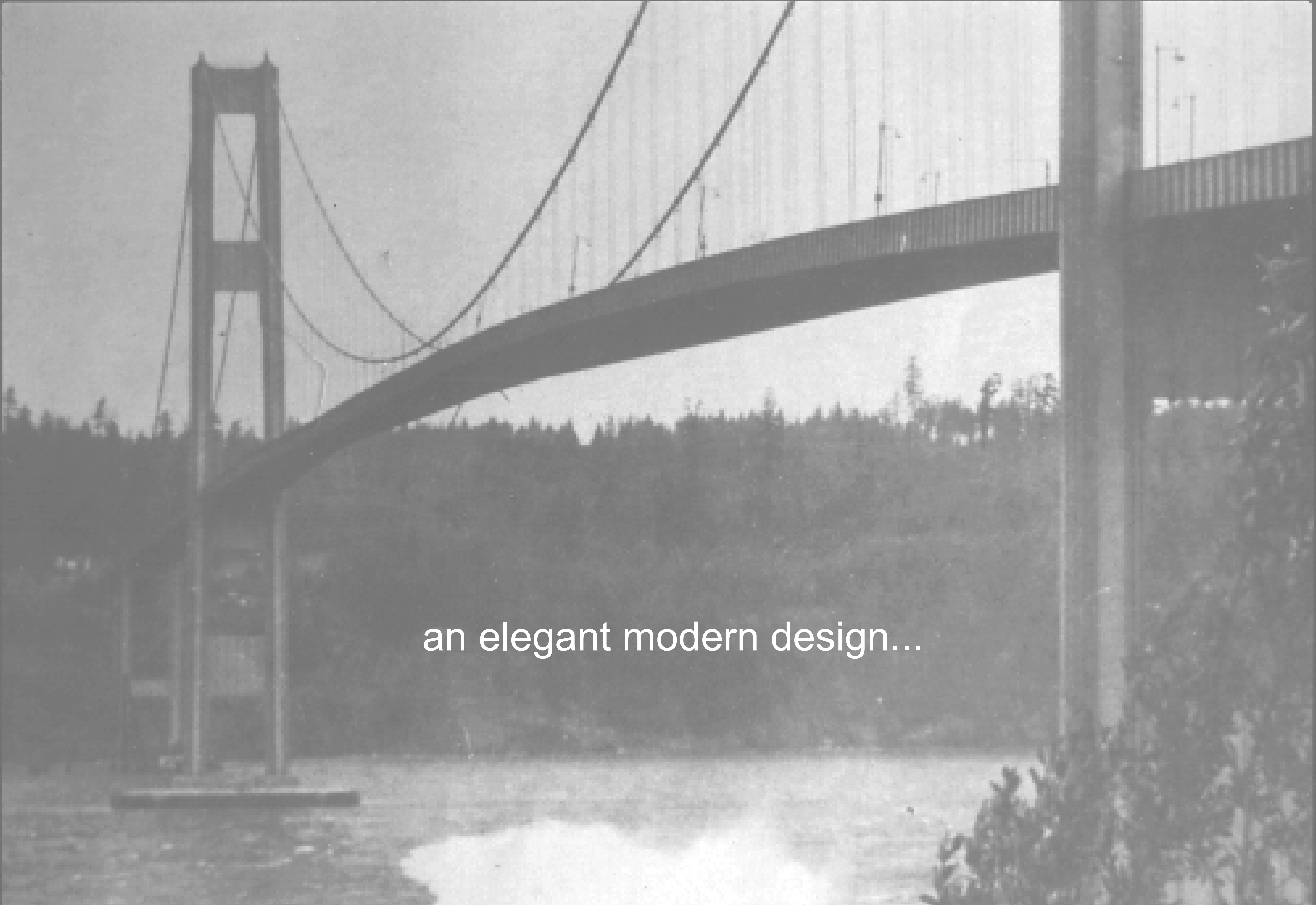


Effective bandwidth of an on-off fluid source, with param

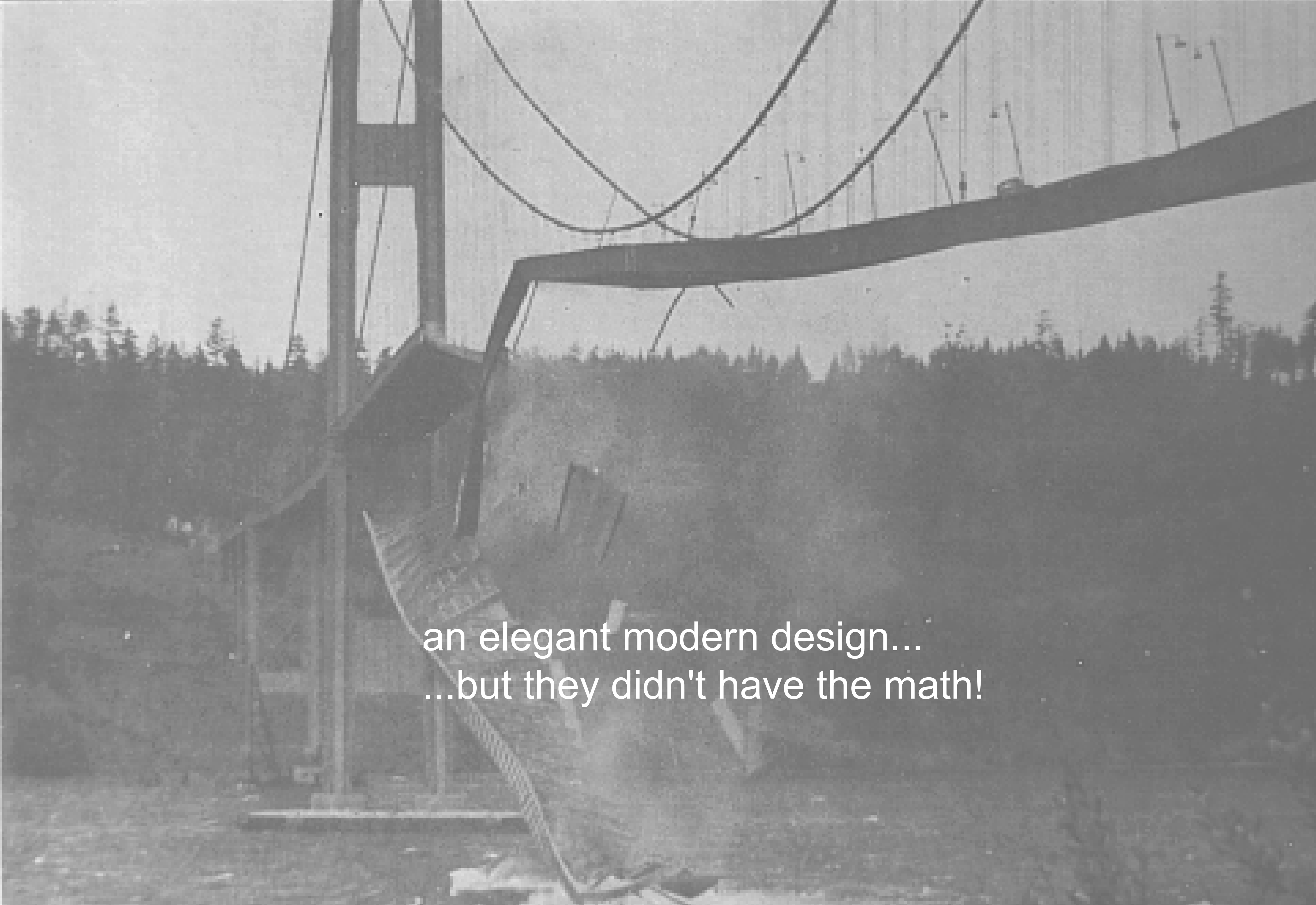
Current prospects for QoS

- rely on over-provisioning
  - over-provision for reliability, no need for QoS mechanisms
  - but what is over-provisioning? how much extra?
- MPLS traffic engineering
  - create "traffic trunks" (virtual circuits with capacity attributes)
  - "*For the purpose of bandwidth allocation, a single canonical value of bandwidth requirements can be computed from a traffic trunk's traffic parameters. Techniques for performing these computations are well known. One example of this is the theory of effective bandwidth*" (RFC 2702).
- Diffserv and traffic engineering
  - "*we don't have the math, so let's not bother*" (Diffserv list)
  - "*merely use different under- and over-provisioning ratios per class*"
- a metaphor...

over-provisioning?



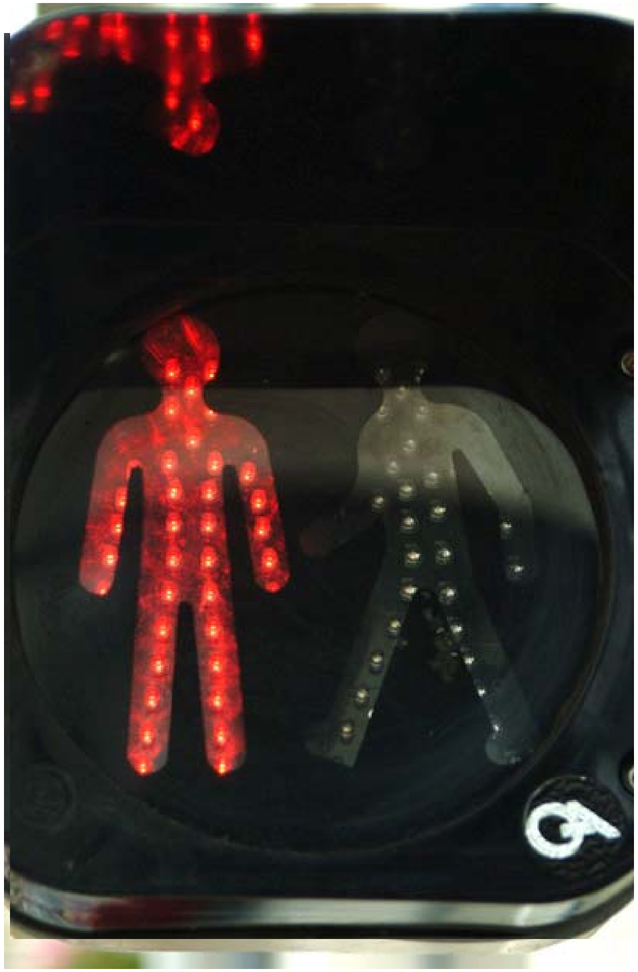
an elegant modern design...



an elegant modern design...
...but they didn't have the math!

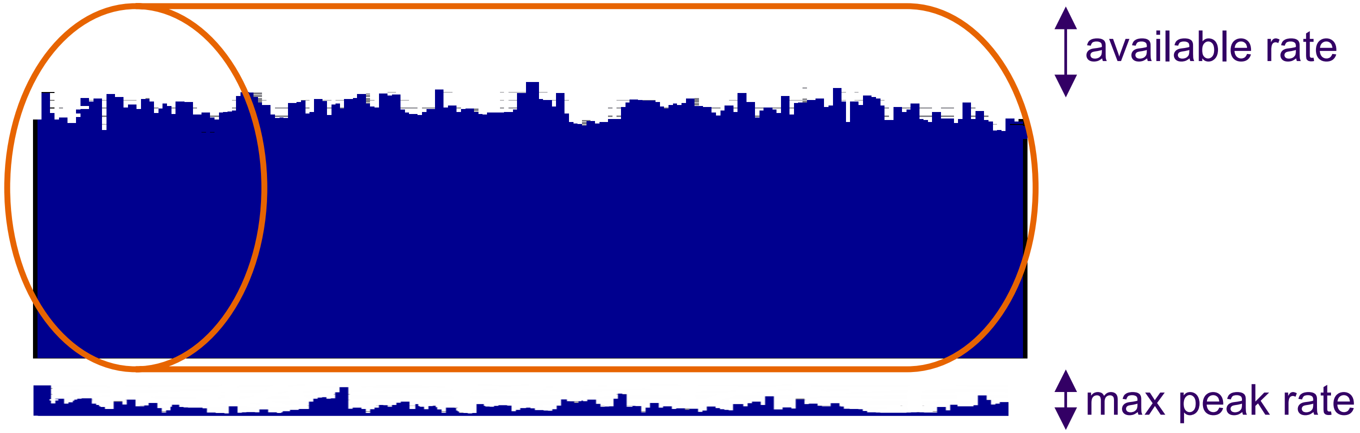
Flow-aware networking

Admission control: a necessary insurance



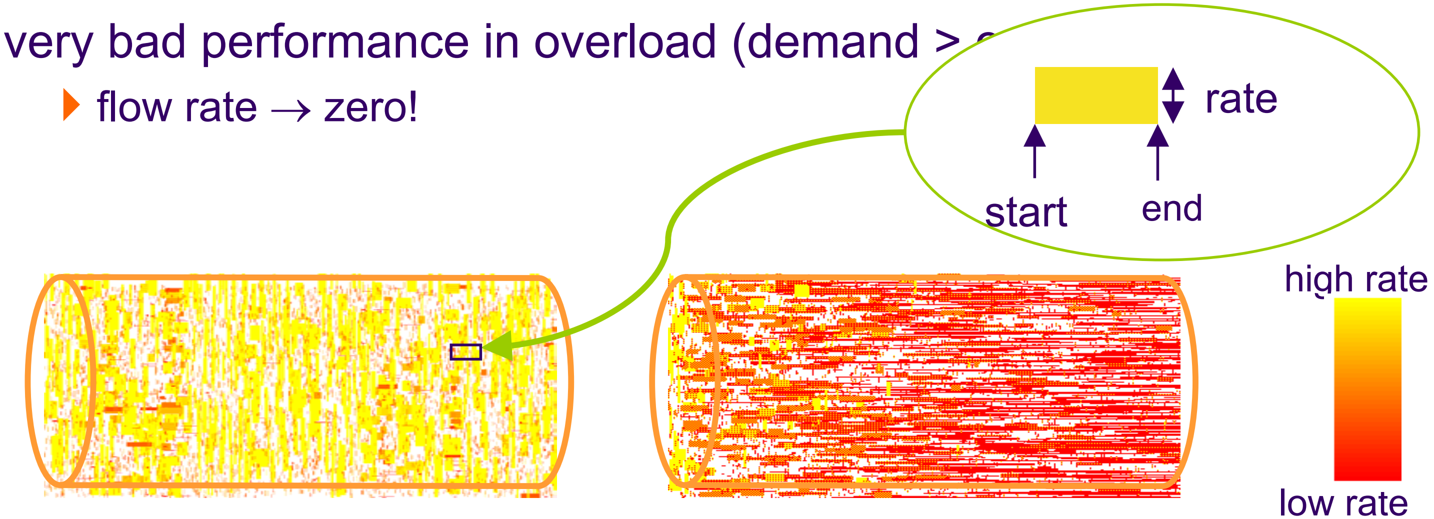
Implicit measurement-based admission control

- a minimal traffic descriptor
  - an upper bound on flow peak rate
- real time estimation of available bandwidth
  - e.g., using method of Grossglauser and Tse (2003)
- only admit a new flow if available rate > R_s (max peak rate)
  - same blocking rate for all rate classes
  - no need to signal rate requirement
- *implicit* admission control
  - "on the fly" flow identification, flow reject by packet discard



Performance of elastic flows

- ➔ assume perfectly fair sharing
  - an imperfectly realized objective of TCP...
  - ... but a simple processor sharing model
- ➔ excellent performance in normal load (utilization < 90%)
  - flow rate ≈ min {peak rate, capacity – demand}
  - the peak rate (e.g., access rate) is limiting in general
- ➔ very bad performance in overload (demand >
  - flow rate → zero!

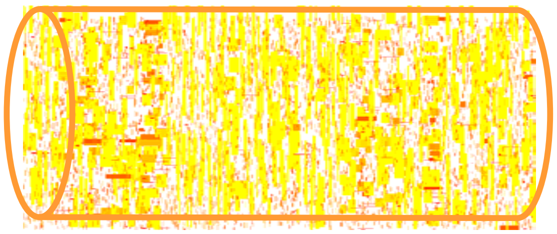


transparency
(demand < capacity)

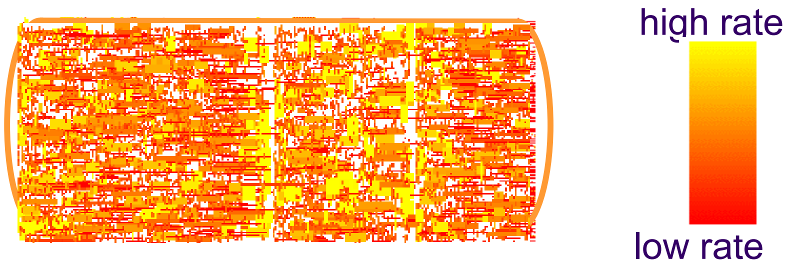
congestion
(demand > capacity)

Measurement-based admission control (just in case...)

- to avoid quality degradation in overload...
- ... pro-actively reject new flows in case of congestion
- requires *implicit* admission control for reactivity
  - continuous real time estimation of realized rate
  - ... reject new flow if this rate < R_e...
  - ... by discarding its packets
- this is easy to perform!
  - choose a threshold R_e of around 1% of link capacity



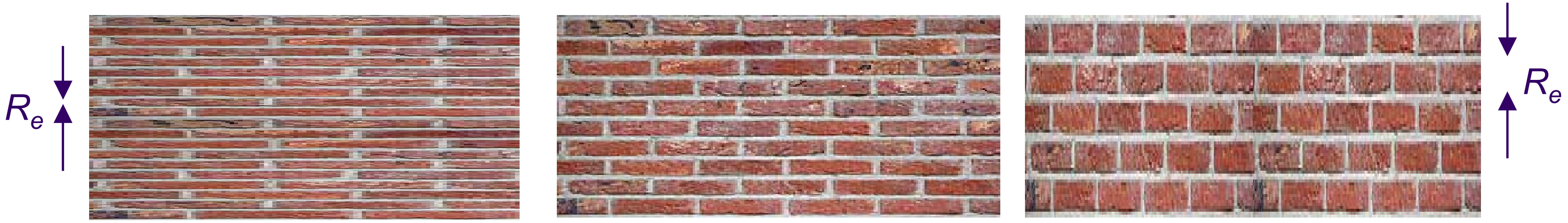
transparency
(demand < capacity)



admission control
(demand > capacity)

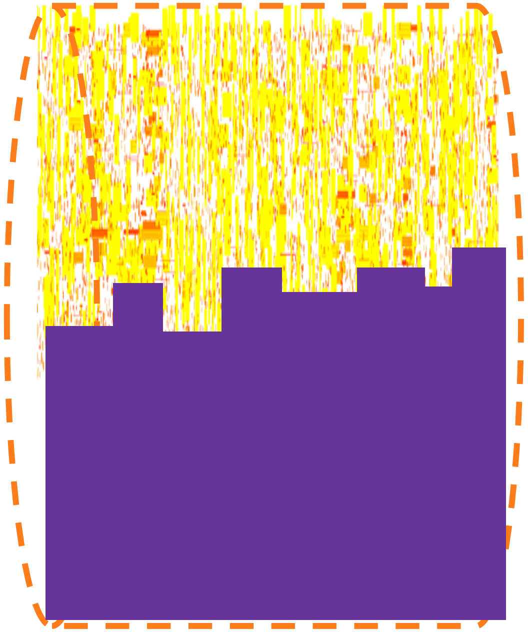
Choosing the thresholds

- streaming flows, R_s
    - application peak rates $\Rightarrow$ lower bound (2 – 5 Mbit/s ?)
    - efficiency (scale economies) $\Rightarrow$ upper bound (~C/100)
- elastic flows, R_e
    - minimum throughput $\Rightarrow$ lower bound (0.1 – 1 Mbit/s ?)
    - low blocking at normal load $\Rightarrow$ upper bound (~C/100)
- a common admission condition
    - for most links, $R_s < R_e \approx C/100$



poor throughput better best

Flow aware networking – 1G



- distinguish streaming and elastic flows
- give priority to packets of streaming flows
  - elastic flows share the residual capacity
- apply implicit admission control to all flows
  - identify flows "on the fly"
  - reject new flows (if necessary) by packet discard
- advantages ☺
  - simple (compared to QoS architectures)
  - cost-effective, controlled performance,...
  - ... and many others!
- disadvantages ☹
  - it is necessary to police the peak rate of streaming flows
  - relies on user cooperation in implementing end to end controls

Flow aware networking – 2G

- avoid explicitly distinguishing streaming and elastic flows
  - user-network interface of the best effort Internet
  - i.e., no policing, limited authentification, simple accounting,...
- provide performance guarantees:
  - streaming quality for peak rates < R_s
  - elastic flow throughput > R_e (if possible)

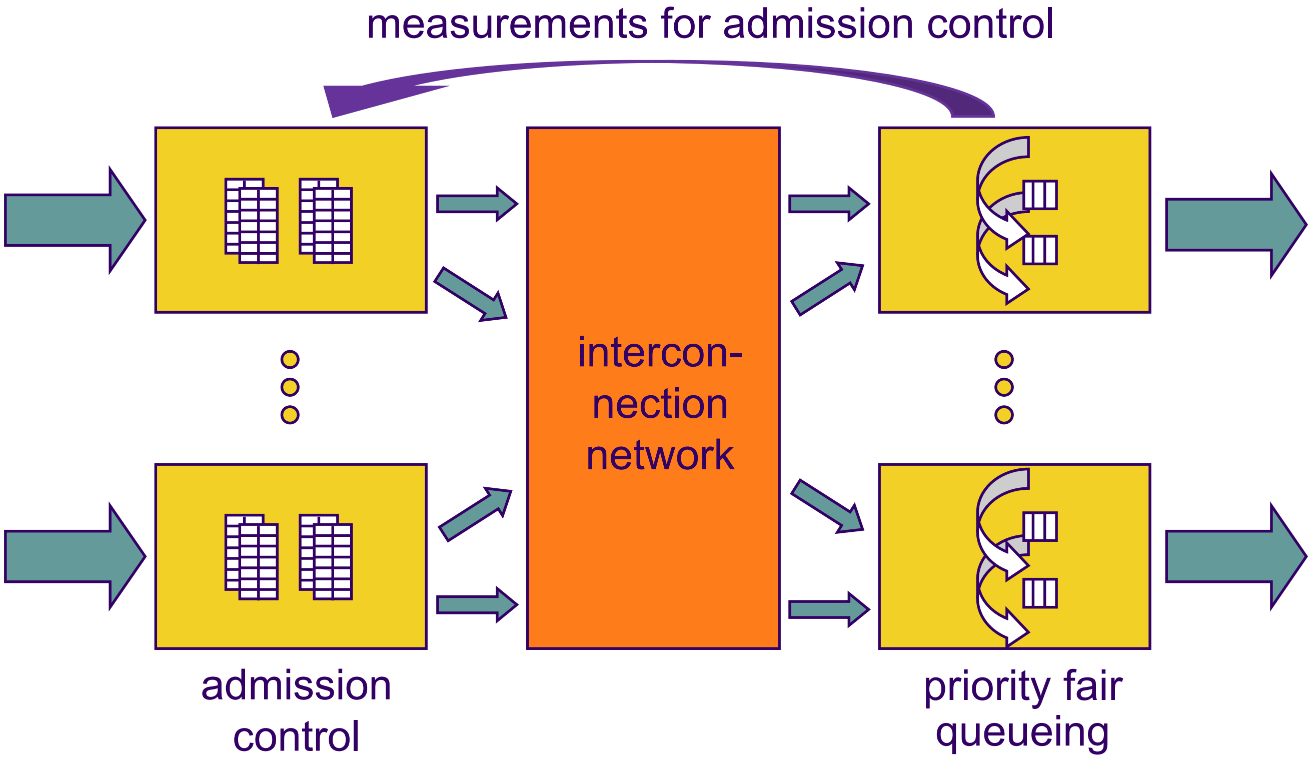
- by joint use of admission control and fair queueing

in a *Cross-protect* router!

The *Cross-protect* router

The Cross-protect mechanisms

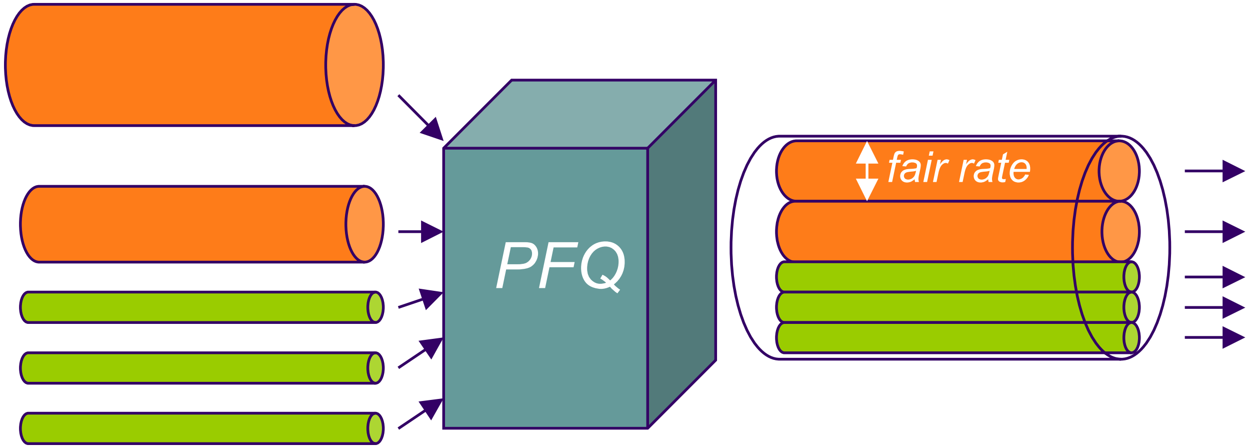
- admission control ensures scalability of fair queueing
- fair queueing provides measurements for admission control
- "priority fair queueing" protects streaming flows and ensures fairness



Priority fair queueing

- self-clocked fair queueing for max-min fair sharing
  - per-flow rate ≤ *fair rate*
- priority to packets of rate < *fair rate*
- admission control to ensure *fair rate* > threshold

- assured fairness for elastic flows
- low delay and loss for streaming flows



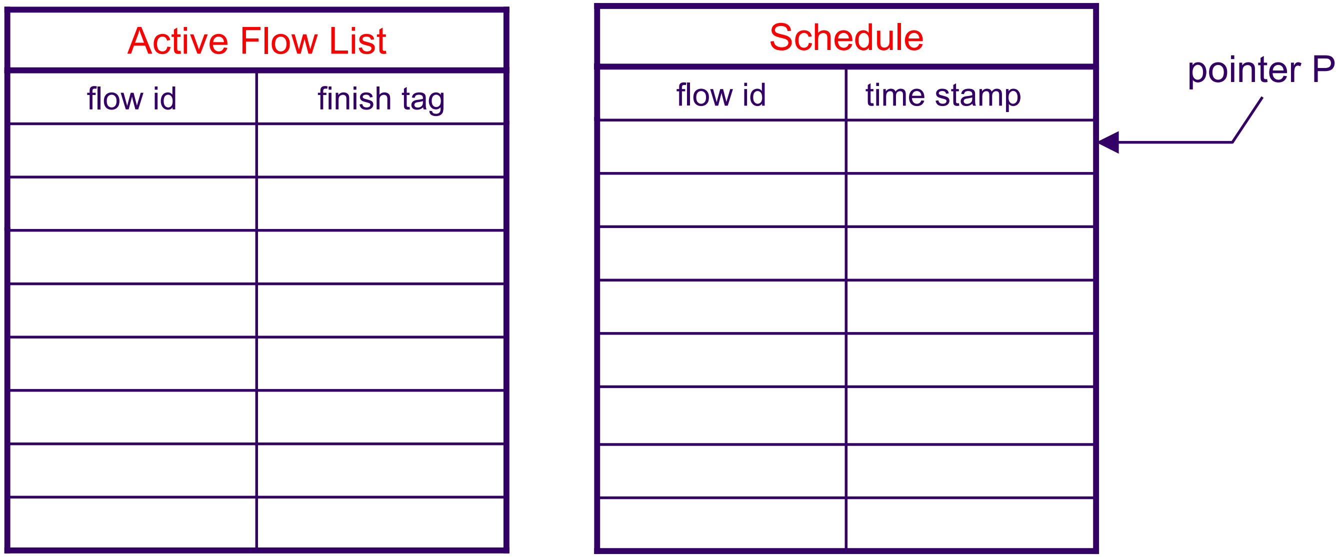
PFQ algorithm (assume constant size packets)

- on packet arrival
  - if (flow id ∈ flow list)
    - write (id, finish tag) to schedule
    - finish tag += 1
  - else
    - write (id, virtual time) to schedule
    - at position P+1
    - finish tag = virtual time +1
  - update active flow list: (id, finish tag)

- on packet departure
  - virtual time = time stamp of first packet
  - for all flows in active flow list
    - if (virtual time ≥ finish tag) remove

virtual time = time stamp of packet at scheduler head

P indicates position of last priority packet



Active Flow List

flow id	finish tag

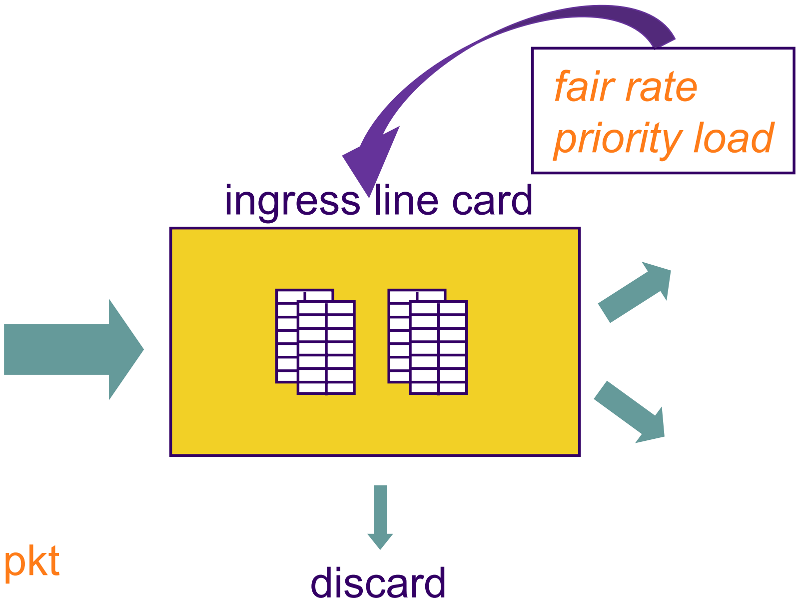
Schedule

flow id	time stamp

pointer P

Implicit admission control

- maintain protected flow lists
  - {flow ID, time of last packet}
  - multiple lists for scalability
- on a packet arrival:
  - read packet ID (on the fly)
  - if flow ID ∈ flow list forward packet
  - else (i.e., new flow)
    - if link congested discard packet
    - else add to list of protected flows, forward pkt
- based on soft state
  - if no packets in time out interval remove flow from list
- admission conditions from PFQ scheduler
  - *fair rate* > threshold 1
  - *priority load* < threshold 2



fair rate
priority load

ingress line card

discard

PFQ algorithm provides congestion indicators

- fair rate
  - bandwidth of a hypothetical permanent flow
- priority load
  - load due to priority packets

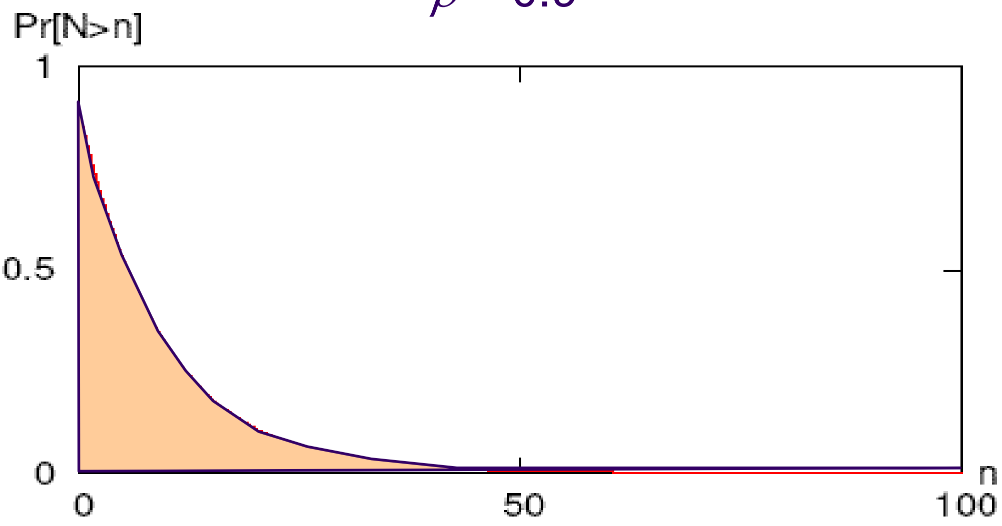
Scalability

- per-flow implicit measurement-based admission control
  - see Caspian Networks: 2 million flows/sec, 6 million active flows on an OC192 (10 Gbit/s) !
  - can certainly do better, or as well but more cheaply
- priority fair queueing
  - complexity depends on number of flows with one or more queued packets
  - this number is bounded (with high probability) by admission control...
  - ...to 100s, not 100 000s...
  - ... and does not depend on link size!

PFQ scalability: case 1) all flows are backlogged

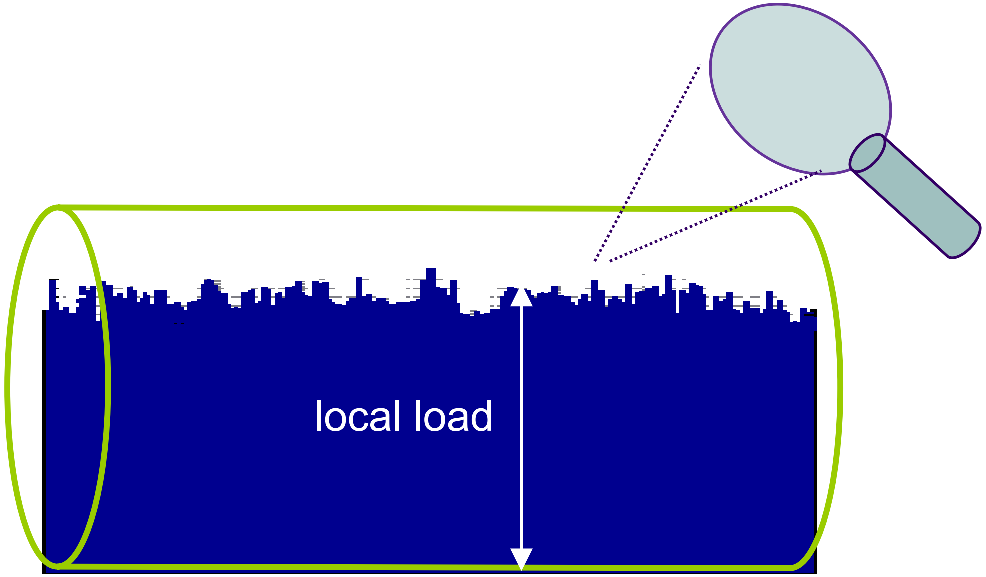
- given fair sharing, number of flows is population of a Processor Sharing queue
  - Pr [N>n] ~ $\rho^{(n+1)}$ (for Poisson session model)
  - e.g., for $\rho = 0.9$, Pr [N>100] ≈ 10^{-4}
- apply admission control to ensure fair rate ≥ 0.01 C
  - i.e., number of flows N ≤100, *always*

$\rho = 0.9$



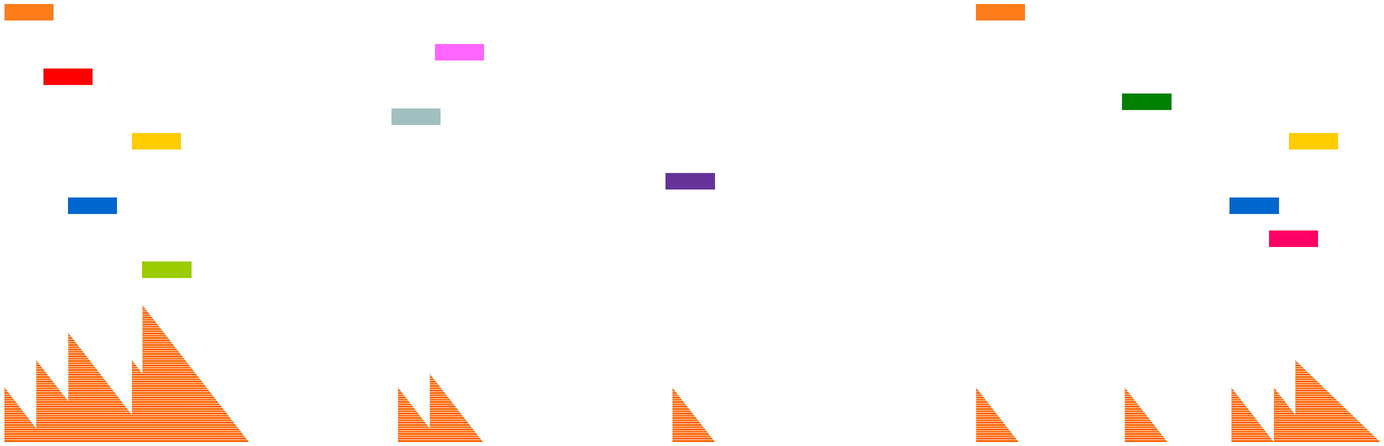
PFQ scalability: case 2) no backlogged flows

- occurs when C is very large (C >> flow peak rate)
- assume:
  - a large number of independent flows
  - constant packet size
  - local load < 0.9 (by admission control)



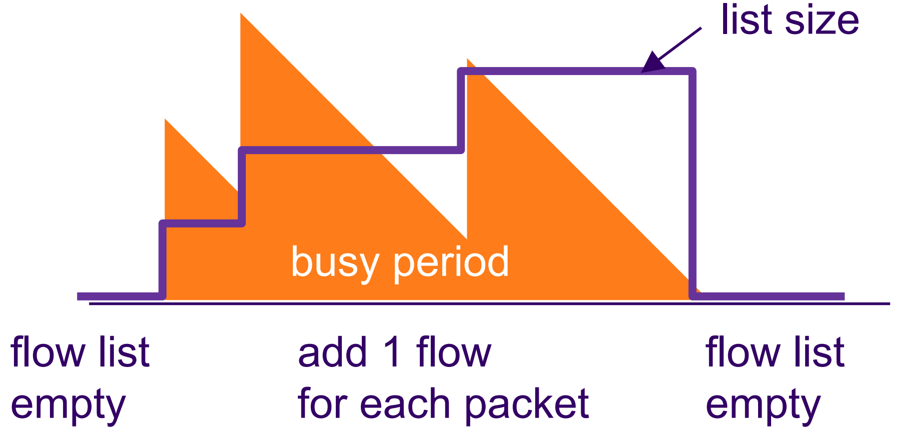
PFQ scalability:
case 2) no backlogged flows

- occurs when C is very large (C >> flow peak rate)
- assume:
  - a large number of independent flows
  - constant packet size
  - local load < 0.9 (by admission control)
- flows list size behaves locally like M/D/1 busy period duration
  - e.g., for local load = 0.9, Pr[N<140] = 0.99



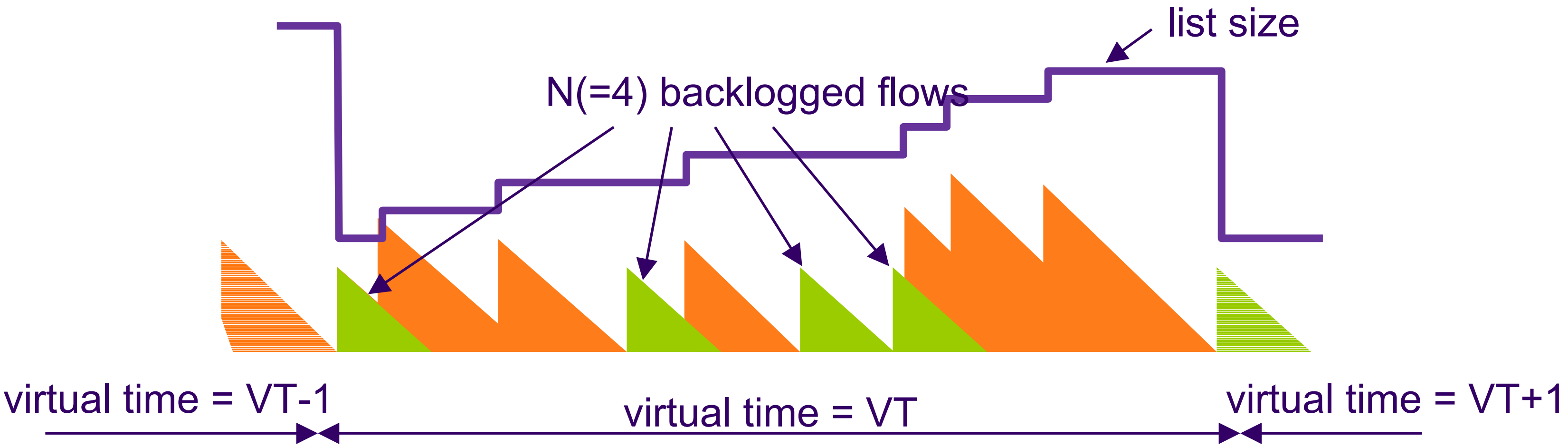
PFQ scalability: case 2) no backlogged flows

- occurs when C is very large (C >> flow peak rate)
- assume:
  - a large number of independent flows
  - constant packet size
  - local load < 0.9 (by admission control)
- flows list size behaves locally like M/D/1 busy period duration
  - e.g., for local load = 0.9, Pr[N<140] = 0.99



PFQ scalability:
case 3) N (≤100) backlogged flows

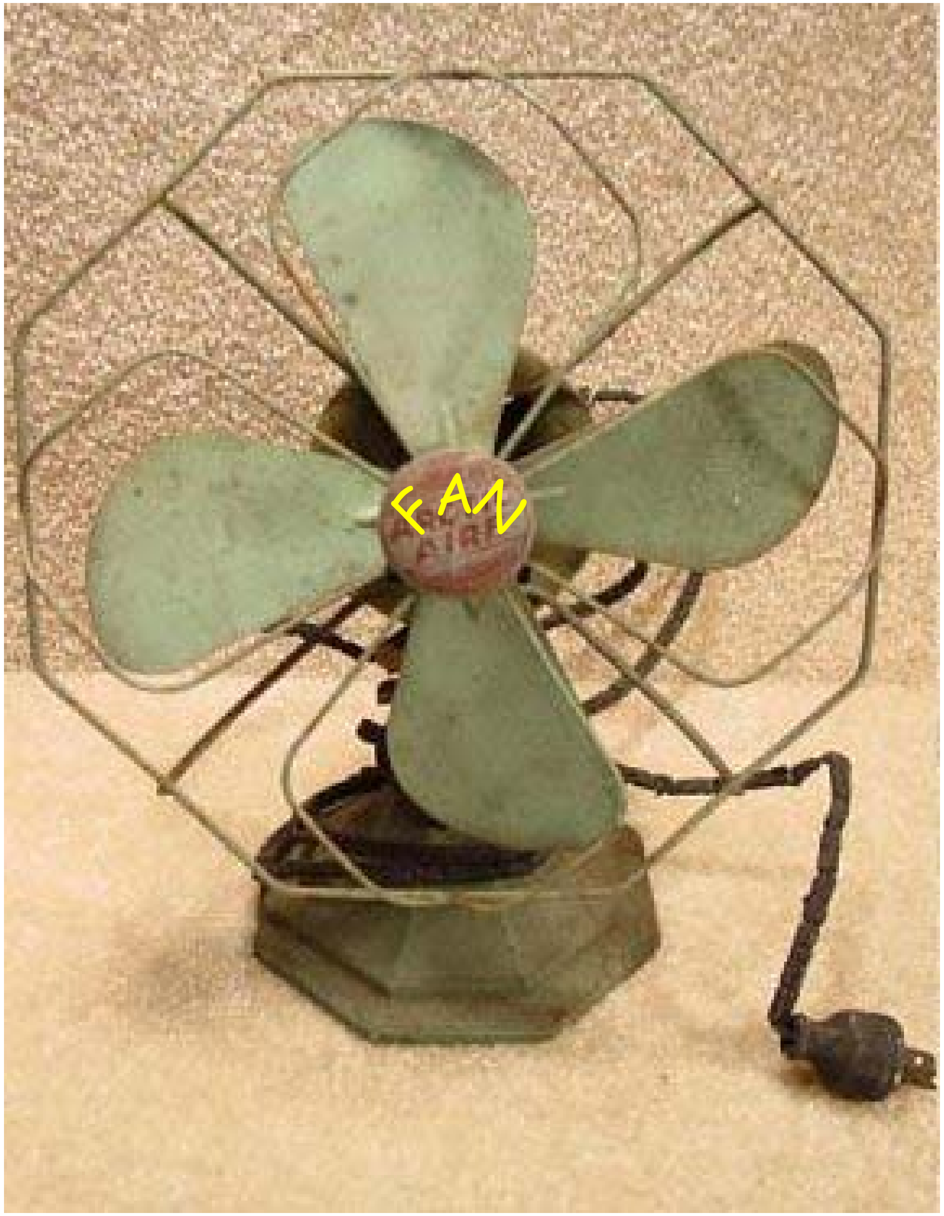
- assume
  - a large number of non-backlogged flows
  - constant size packets
- "cycles" defined by value of virtual time
- number of flows = cycle length ≤ N consecutive M/D/1 busy periods
- assume M/D/1 load ≤ min {0.9, 1 – 0.01 N} (by admission control)
- Pr [list size > 476] < 0.99 in worst case (N=10, load = 0.9)



QoS without classes of service: "under" and "over"

- flows are "over" or "under" the fair rate
  - flows that are under have negligible delay and loss
  - flows that are over have to adjust their rate and expect significant delay
- admission control maintains the fair rate high enough
  - ~1% of link capacity
- "high enough" for a class of streaming applications
  - for interactive and streaming flows...
  - ... and signalling and games and ...
- "high enough" to maintain throughput
  - for elastic flows that have a high peak rate

Using flow-aware networking



User-defined flows

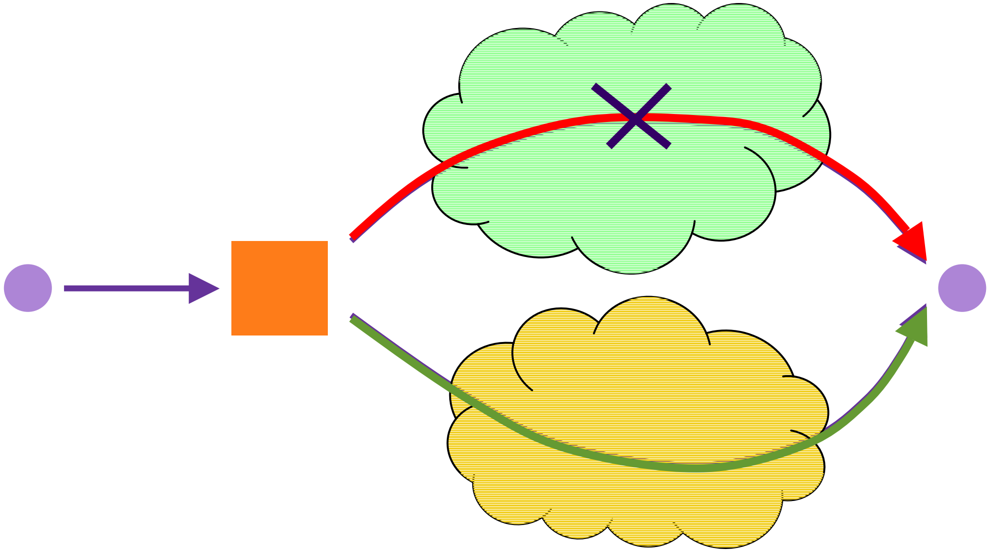
- using the IPv6 flow label
  - an ideal solution
  - need for standards?
- flow identifier in IPv4
  - the 5-tuple?
  - how to deal with tunnels?
- flexible service creation
  - at the edge...
  - ... like the current Internet!

Selective admission control

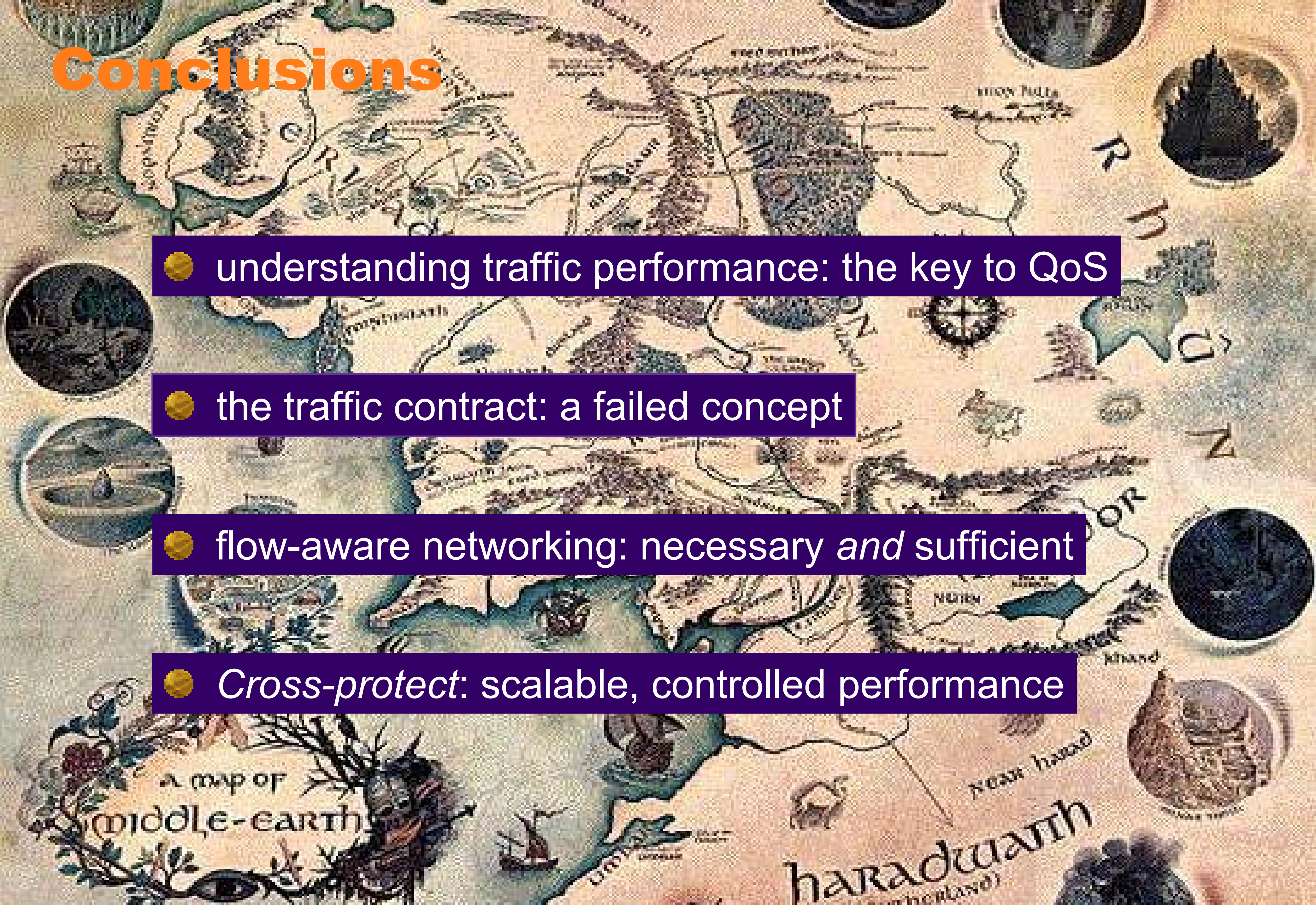
- by applying different admission thresholds
  - for emergency calls
  - for five 9's reliability
  - for routing efficiency
- block ordinary flows congestion attains level 1
  - using measured fair rate and priority load
- only block premium flows if congestion attains level 2
  - a rare event given prior blocking of ordinary traffic
  - cf. "trunk reservation" in circuit switching

Adaptive routing

- using flow label for load balancing
  - #(flow label // IP addresses) ⇒ route choice
- alternative routing
  - on flow blocking, change flow label and retry
- multipath routing
  - applications initiate several flows
  - proceed on best route, or continue on all



Conclusions



- understanding traffic performance: the key to QoS
- the traffic contract: a failed concept
- flow-aware networking: necessary *and* sufficient
- *Cross-protect*: scalable, controlled performance