

Performance of Networked Systems



Lecture 4: Performance and Traffic Management in IP Networks

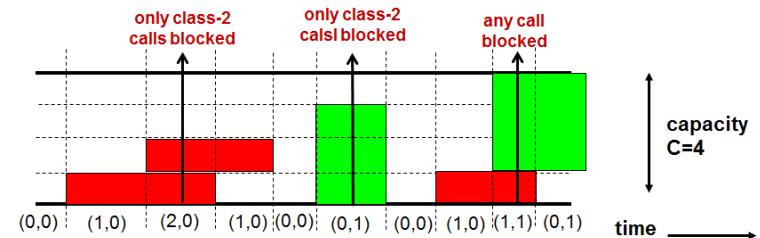
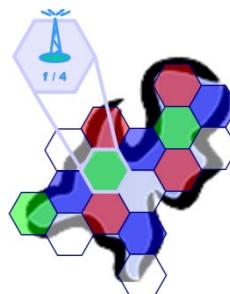
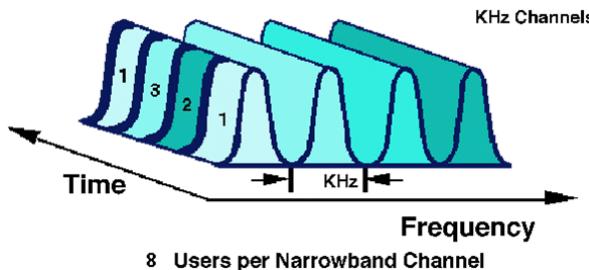
Overview of today's lecture

1. Wrap up of last lecture
2. Processor Sharing models for elastic traffic (left over from last)
3. Traffic characteristics at different time scales
4. Classification of applications: error and delay tolerance
5. Traffic Management mechanisms for IP networks

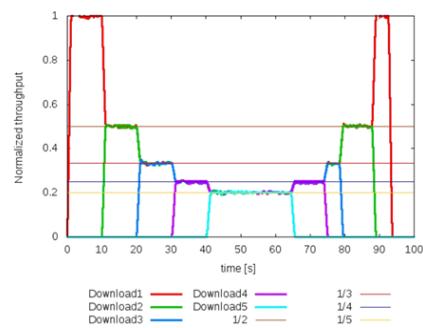


Wrap Up of Last Lecture

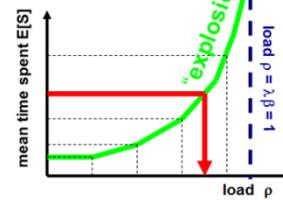
Streaming



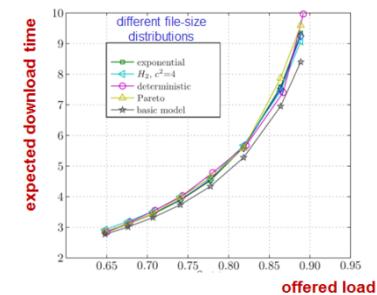
Elastic



basic model (theory)

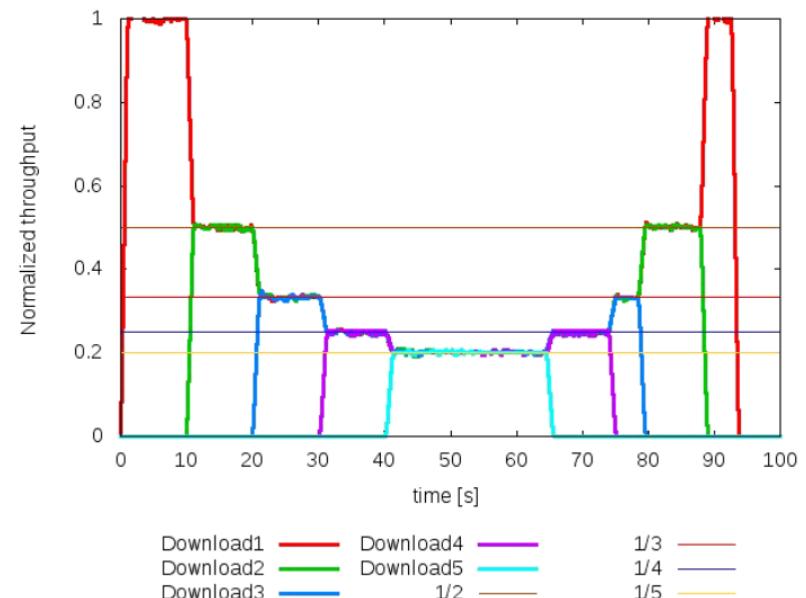
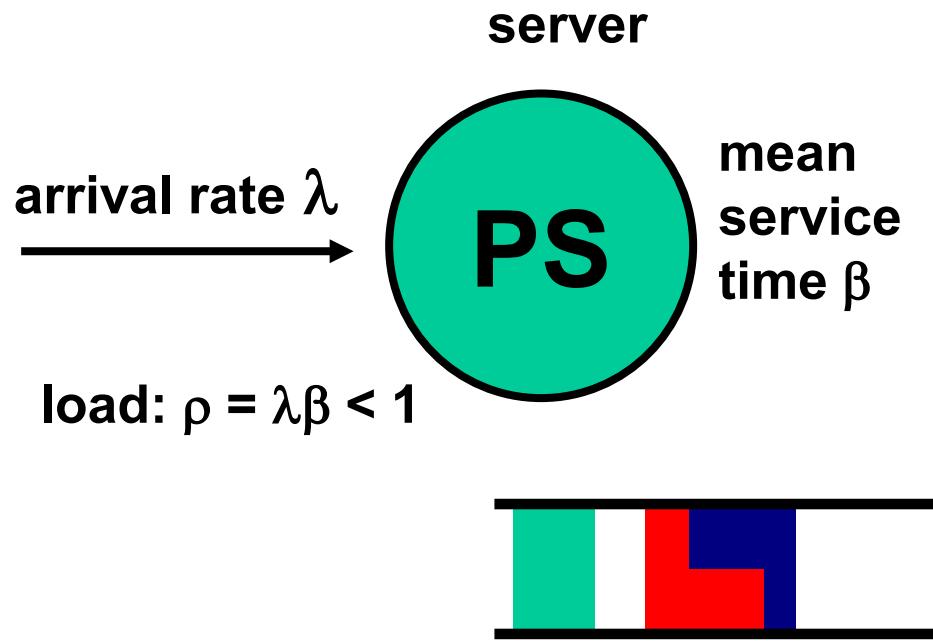


practice (lab setup)



1. Streaming versus elastic traffic
2. Erlang blocking model (“Erlang-B”)
3. Multi-rate models and product-form solution
4. Kaufman-Roberts recursion
5. Elastic traffic: Processor Sharing models (today)

Processor Sharing Models



Processor Sharing (PS): If k customers in the system, then each of them gets processing speed $1/k$ ("fair sharing")

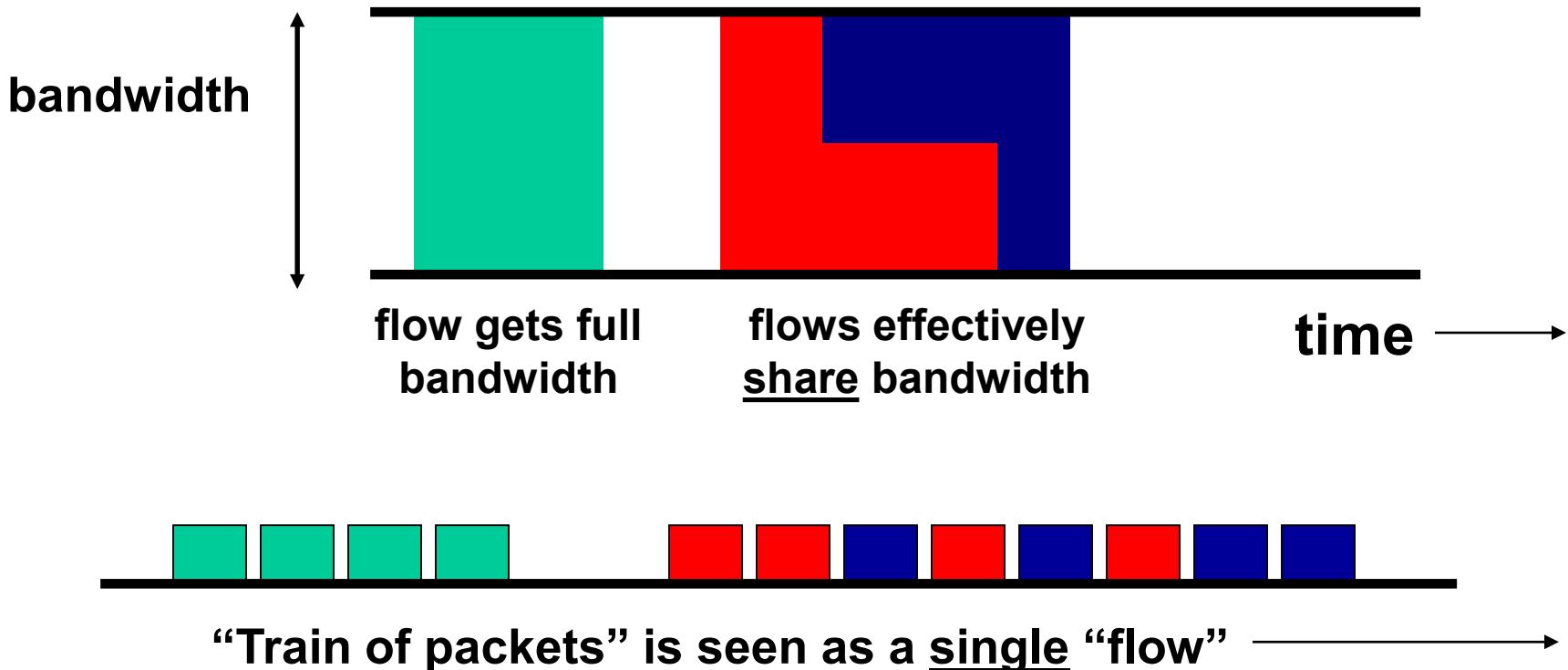
This model is often called the "**M/M/1**" PS model

- 1st "M" means "Markovian" (Poisson), 2nd "M" exponential service times, and 1 means 1 server

Key performance metric: mean sojourn time $E[S]$ (models the transfer time of a flow)



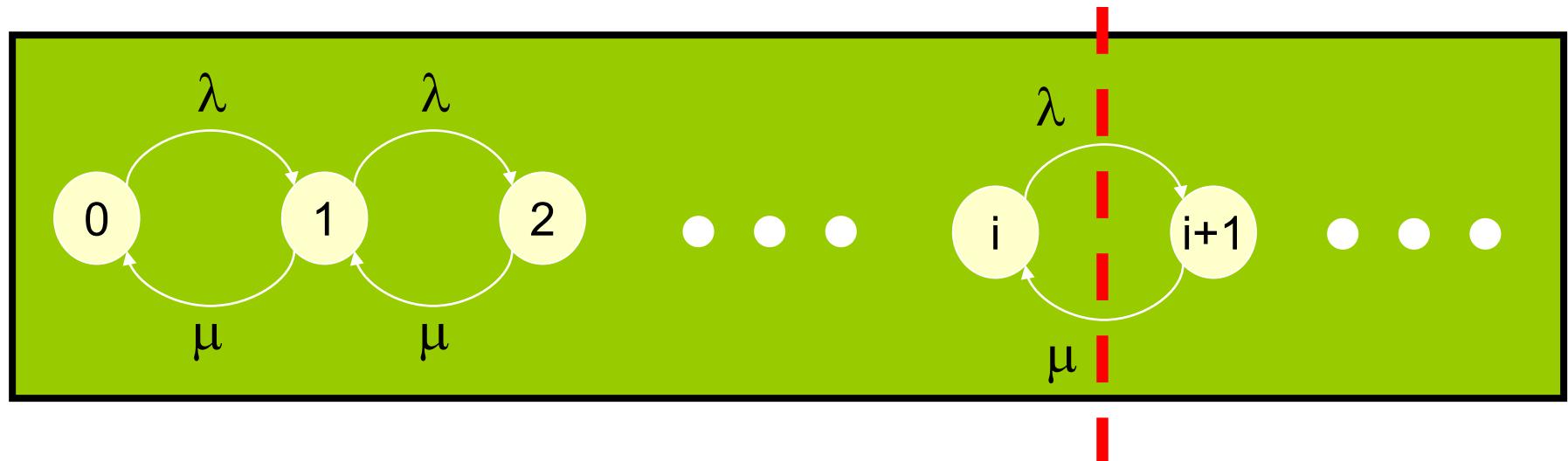
Applications of PS Models



Applications:

1. Multitasking: jobs fairly share CPU power
2. Fair bandwidth sharing among different users

Analysis of the M/M/1 PS Model



N := number of customers in system

$$\pi_j := \Pr\{N = j\} (j = 0, 1, \dots)$$

Balancing arguments:

$$\lambda\pi_i = \mu\pi_{i+1} (i = 0, 1, 2, \dots)$$

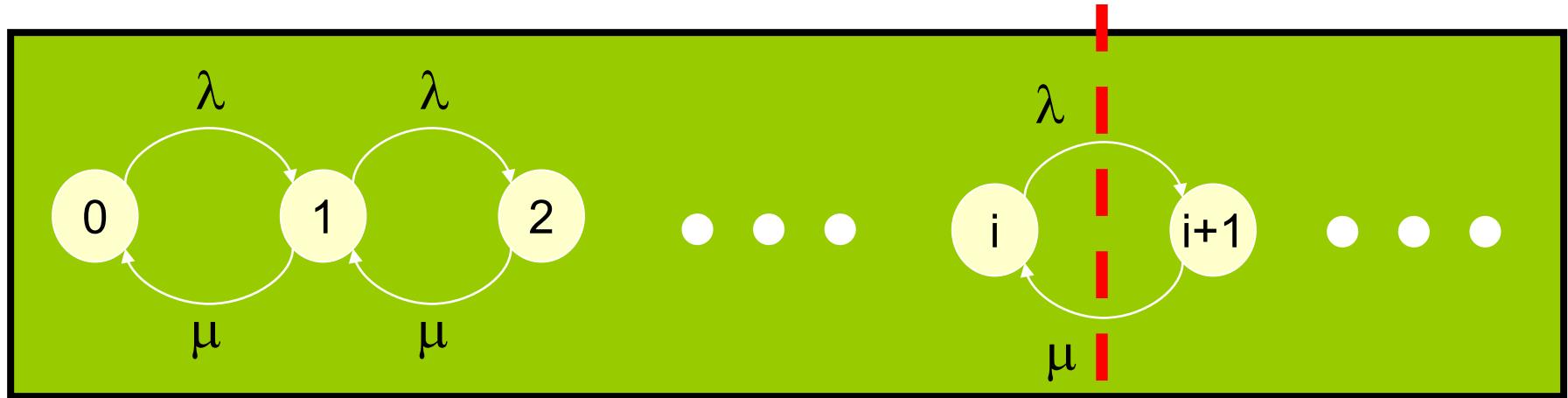
Normalization:

$$\pi_0 + \pi_1 + \dots = 1$$

Note that the state space is unlimited (as opposed to Erlang-B model), because there is no maximum to the number of jobs

assuming exponential service times with mean $\beta=1/\mu$

Analysis of the M/M/1 PS Model



N := number of customers in system

$$\pi_j = \left(1 - \frac{\lambda}{\mu}\right) \left(\frac{\lambda}{\mu}\right)^j = (1 - \rho) \rho^j \quad (j = 0, 1, \dots)$$

$$\rho := \lambda \beta = \frac{\lambda}{\mu} \quad \text{load}$$

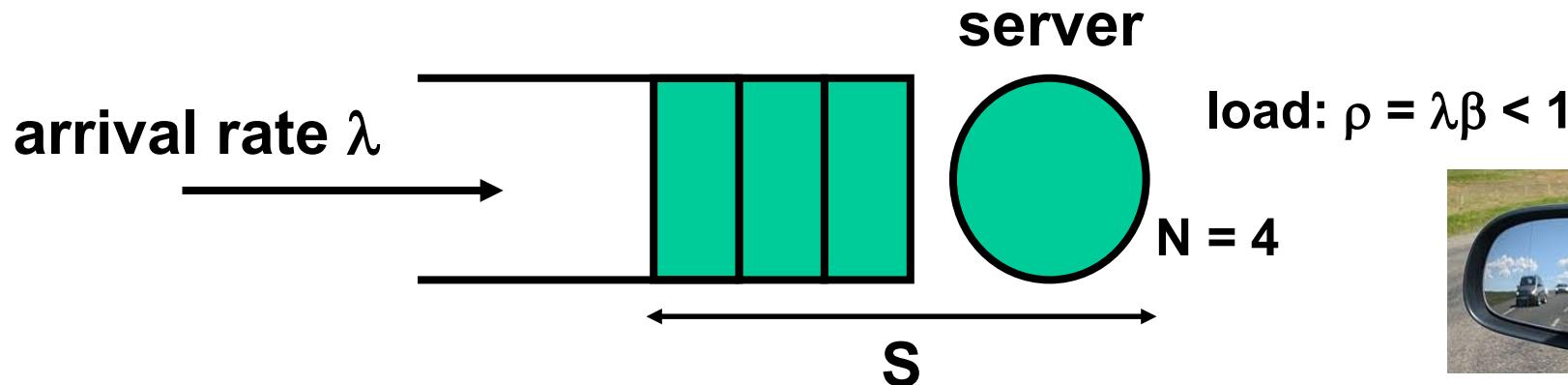
Expected number of customers in system

$$E[N] = \sum_{j=0}^{\infty} j \Pr\{N = j\} = (1 - \rho) \sum_{j=0}^{\infty} j \rho^j = \frac{\rho}{1 - \rho}$$

Expected sojourn time $E[S] = \frac{E[N]}{\lambda} = \frac{\beta}{1 - \rho}$ **(using “Little”)**



Little's Formula



$S :=$ sojourn time: total time job is in the system is (including service)

$N :=$ total number of jobs in the system (including service)

Little's Formula: $E[N] = \lambda E[S]$

Intuition: “On average, the customers in the system $E[N]$ are the ones that arrived during the last $E[S]$ time units”

Interpretation via cost structure:

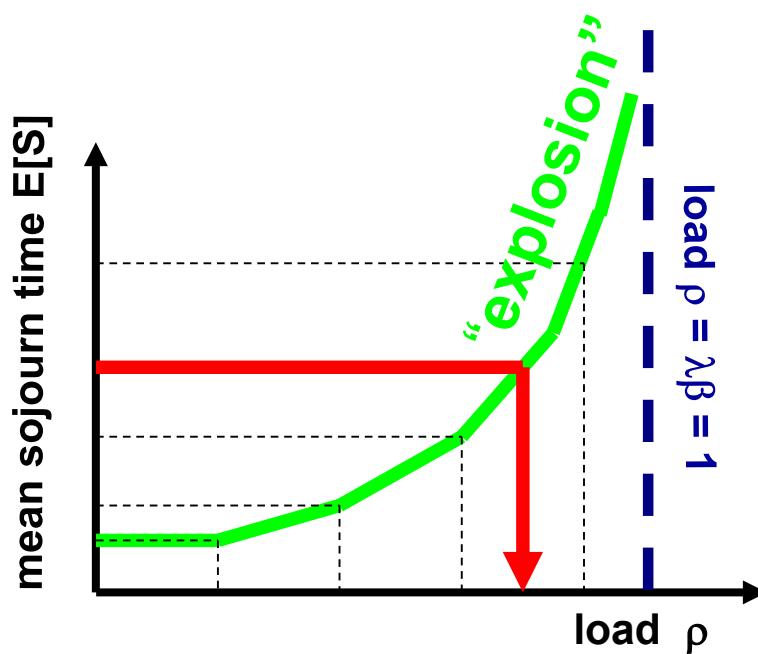
Each customer pays 1 euro per minute in system, earned in two ways

1: customers pay continuously in time, then total $E[N]$ per minute

2: pay upon departure, $E[S]$ per customer, in total $\lambda E[S]$ per minute

Little's formula relates number of jobs to “waiting times”

Analysis of the M/G/1 PS Model



Here, G means “general” service times (so includes also non-exponential service times)

ρ	slow-down factor
50%	2
75%	4
90%	10
95%	20

Expected sojourn time: $E[S] = \frac{\beta}{1 - \rho}$

Insensitivity: result also holds for non-exponential service-time distributions

Interpretation: sharing the capacity with other customers leads to a slow-down factor $1/(1 - \rho)$



Processing Sharing Model

- Unconditional -

Assumptions:

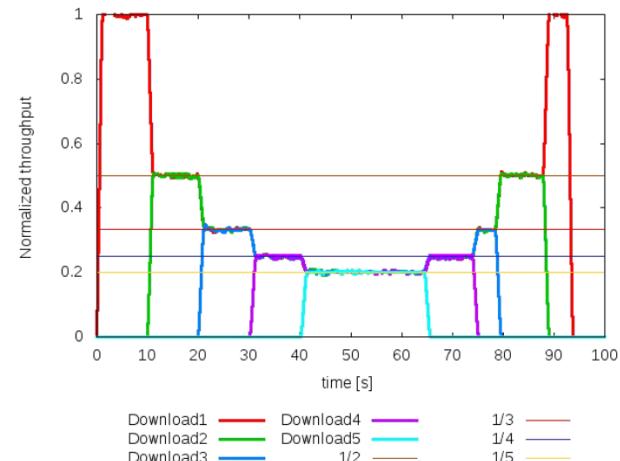
- Network bandwidth = 100 Mbit per second
- Average file size = 5 MByte = 40 Mbit
- File transfer request rate = 2 files per second

Intermediate calculations:

- Arrival rate $\lambda = 2$ (files per second)
- Mean processing time per file $\beta = (40 \text{ Mbit}) / (100 \text{ Mbit/s}) = 0.4 \text{ seconds}$
- Utilization $\rho = 0.8 = 80\%$

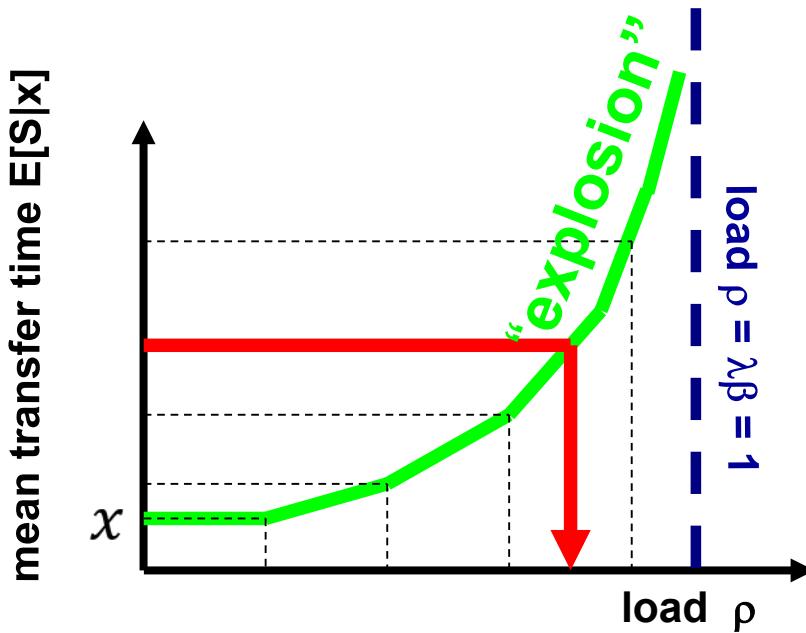
Mean file transfer time:

$$E[S] = \frac{\beta}{1 - \rho} = \frac{0.4}{1 - 0.8} = 2 \text{ seconds}$$





Conditional Sojourn Times



Expected sojourn time of job of size x : $E[S|x] = \frac{x}{1 - \rho}$

Insensitivity: result also holds for non-exponential service-time distributions

Interpretation: sharing the capacity with other customers leads to a **slow-down factor $1/(1 - \rho)$**

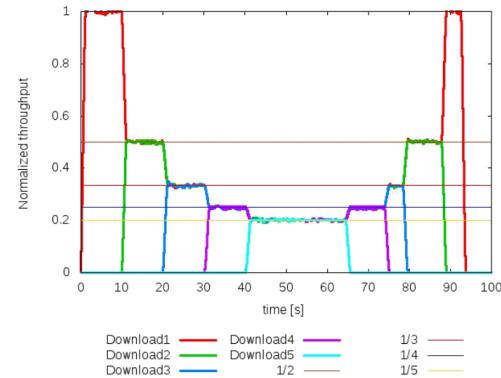


Processing Sharing Model

- conditional transfer time -

Assumptions:

- Network bandwidth = 100 Mbit per second
- Give file size = 5 MByte = 40 Mbit
- File transfer request rate = 2 files per second



Intermediate calculations:

- Arrival rate $\lambda = 2$ (files per second)
- Processing time for *given* file of 5MByte = $(40 \text{ Mbit}) / (100 \text{ Mbit/s}) = 0.4 \text{ seconds}$
- Utilization $\rho = 0.8 = 80\%$

Mean conditional transfer time (of job of size 5MBytes)

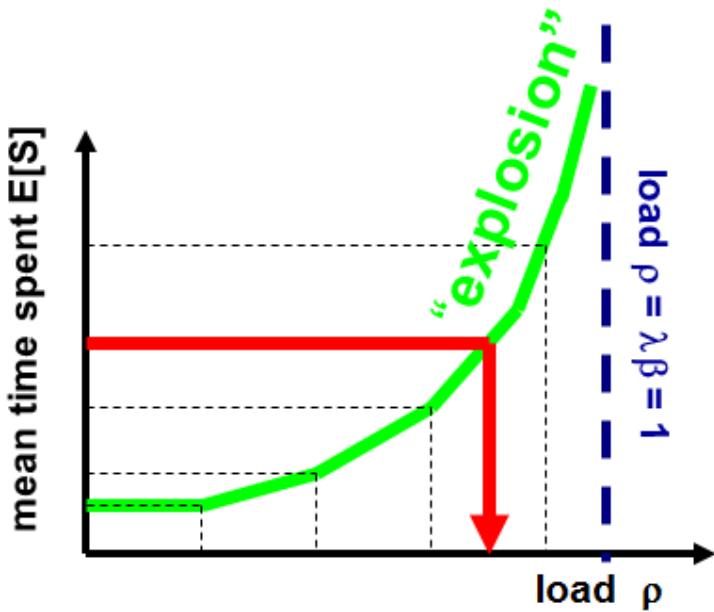
$$E[S|x] = \frac{x}{1-\rho} = \frac{0.4 \text{ (seconds)}}{1-\rho} = \frac{0.4}{1-0.8} = 2 \text{ seconds}$$

5MByte file size
translated in seconds

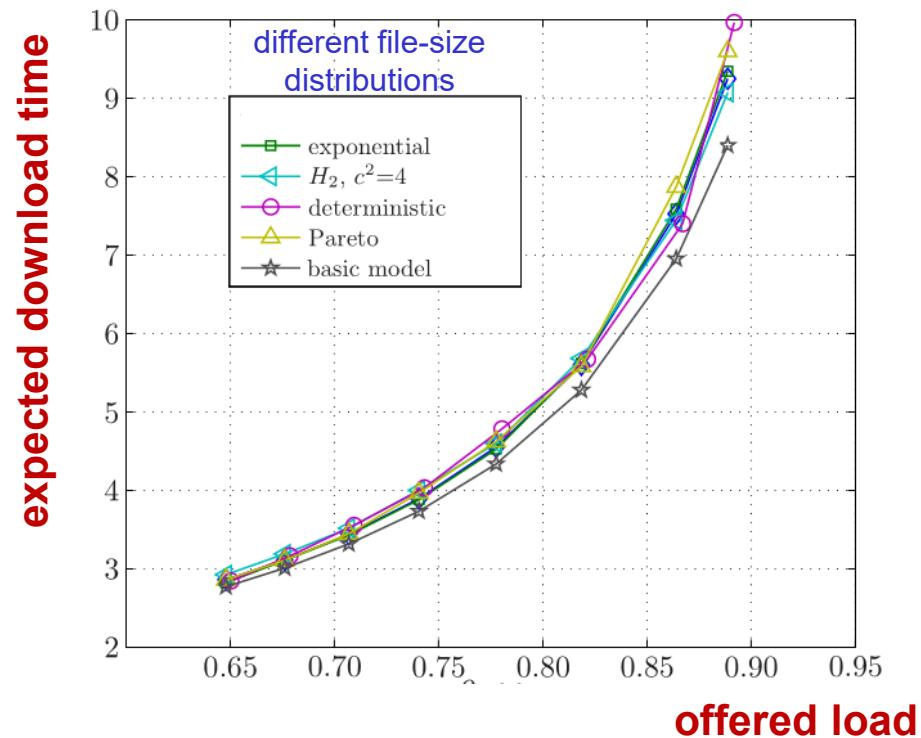


Model Validation

basic model (theory)



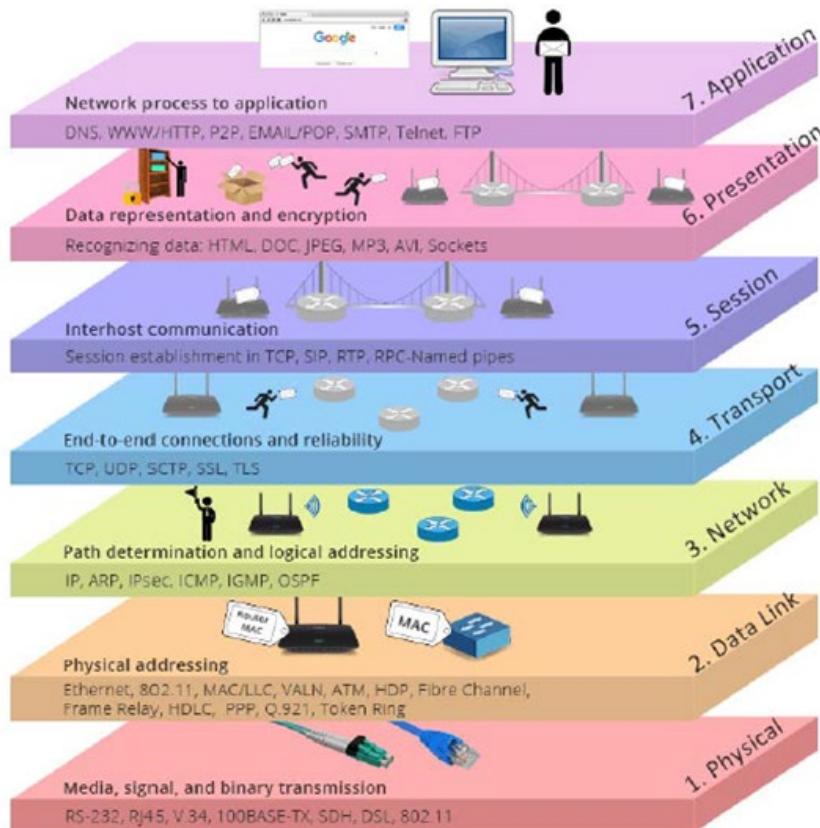
practice (lab setup)



- Lab results match very well with theoretical PS-model
- Transfer time indeed (fairly) insensitive to the file size distribution



Course Structure



HTTP, FTP, MM applications

next week

Transport Control

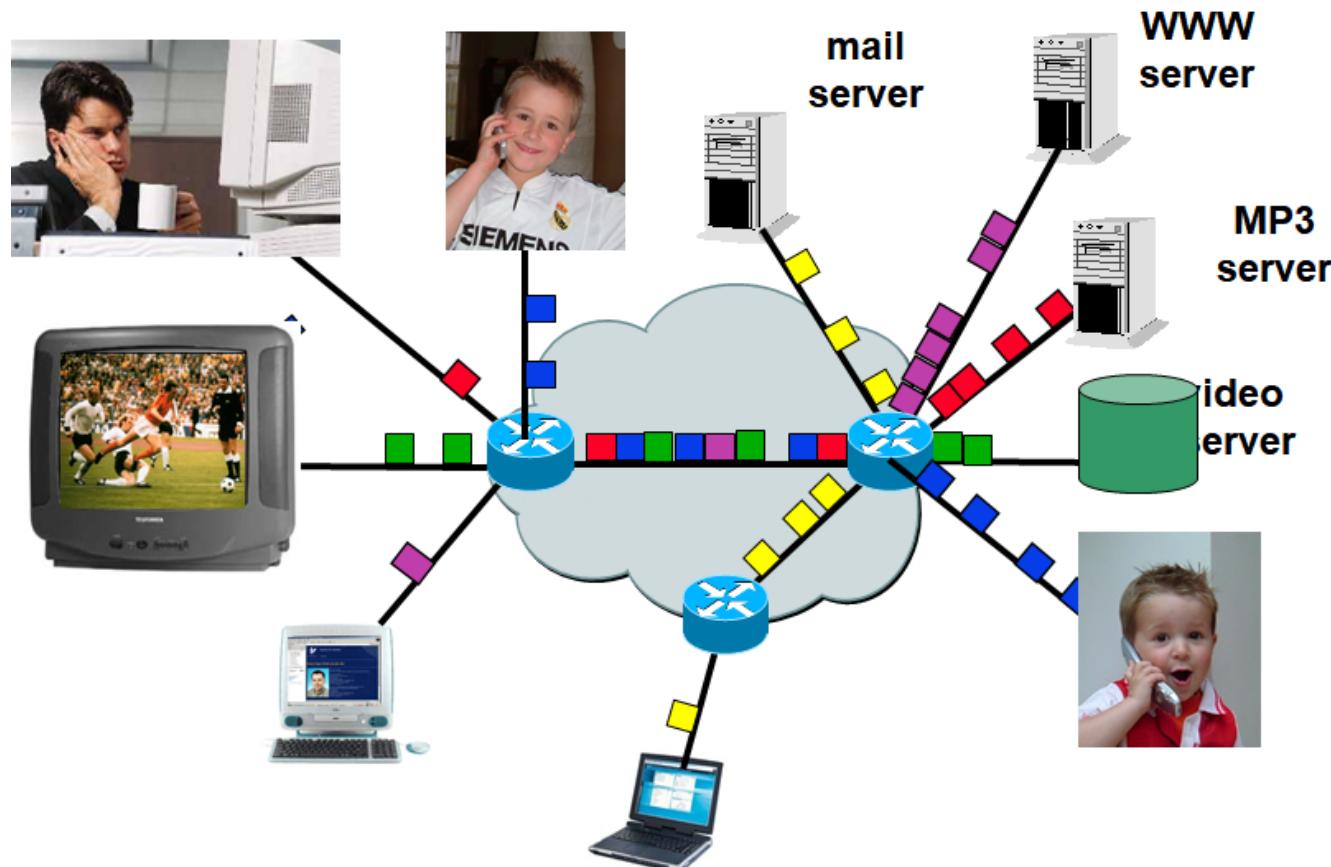
Traffic Management today

Medium Access in two weeks

Idea: understand both technology and theory



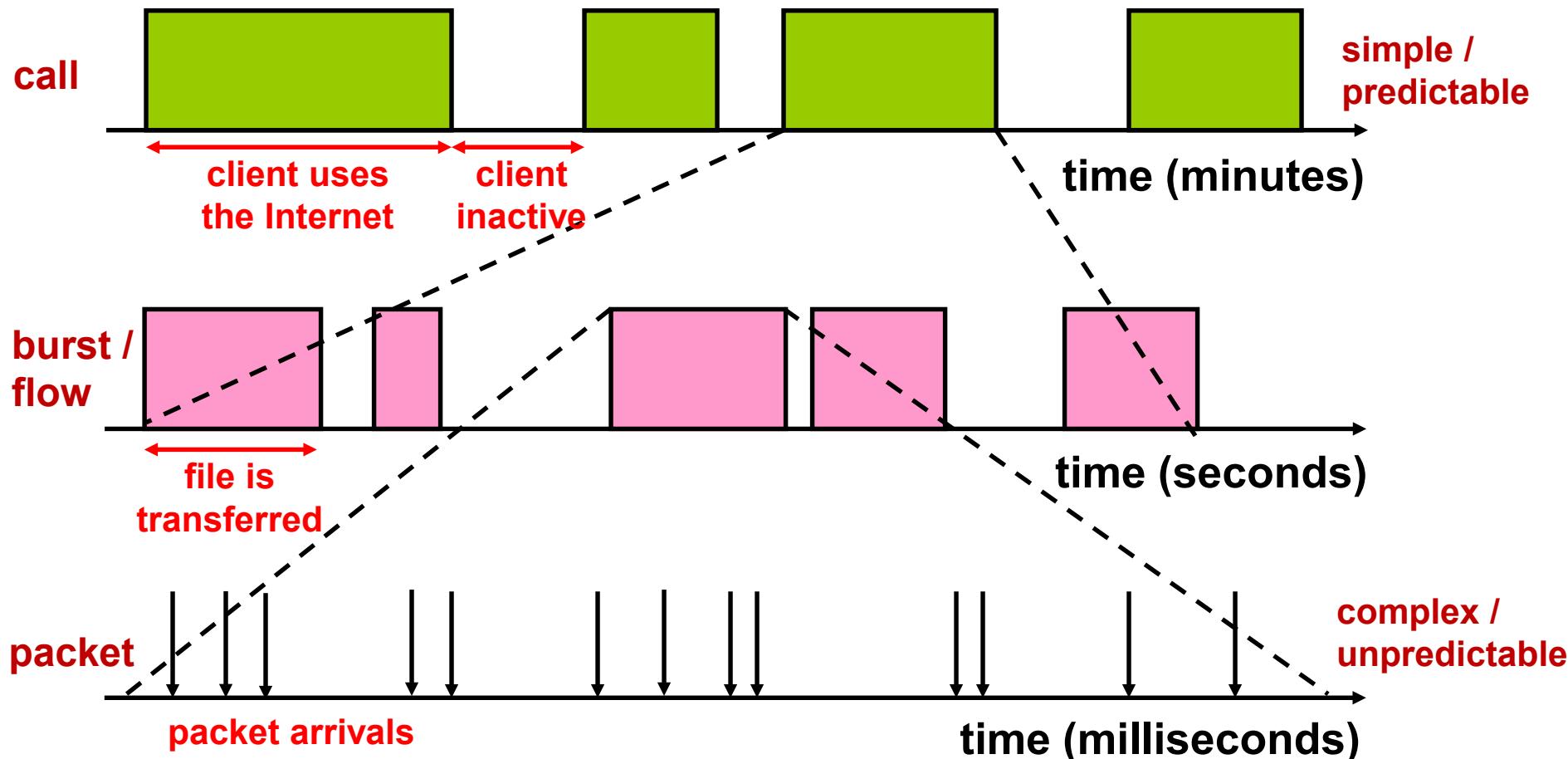
Heterogeneity in IP networks



Different applications have different:

- QoS requirements
- Traffic characteristics

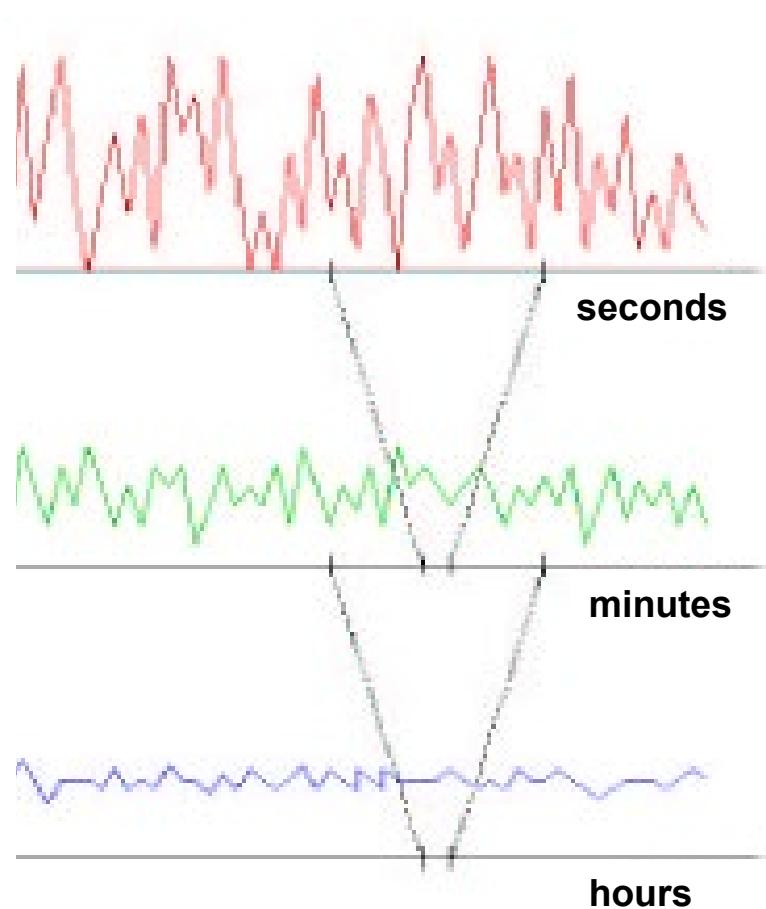
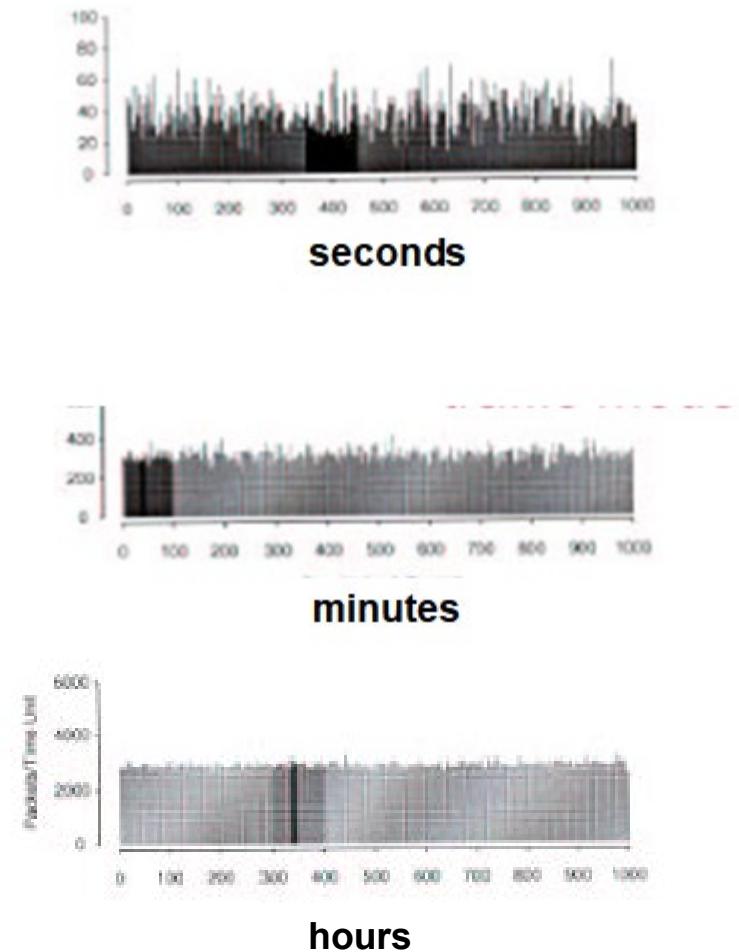
Time Scales – an Internet User



Idea: Traffic has different characteristics at different time scales

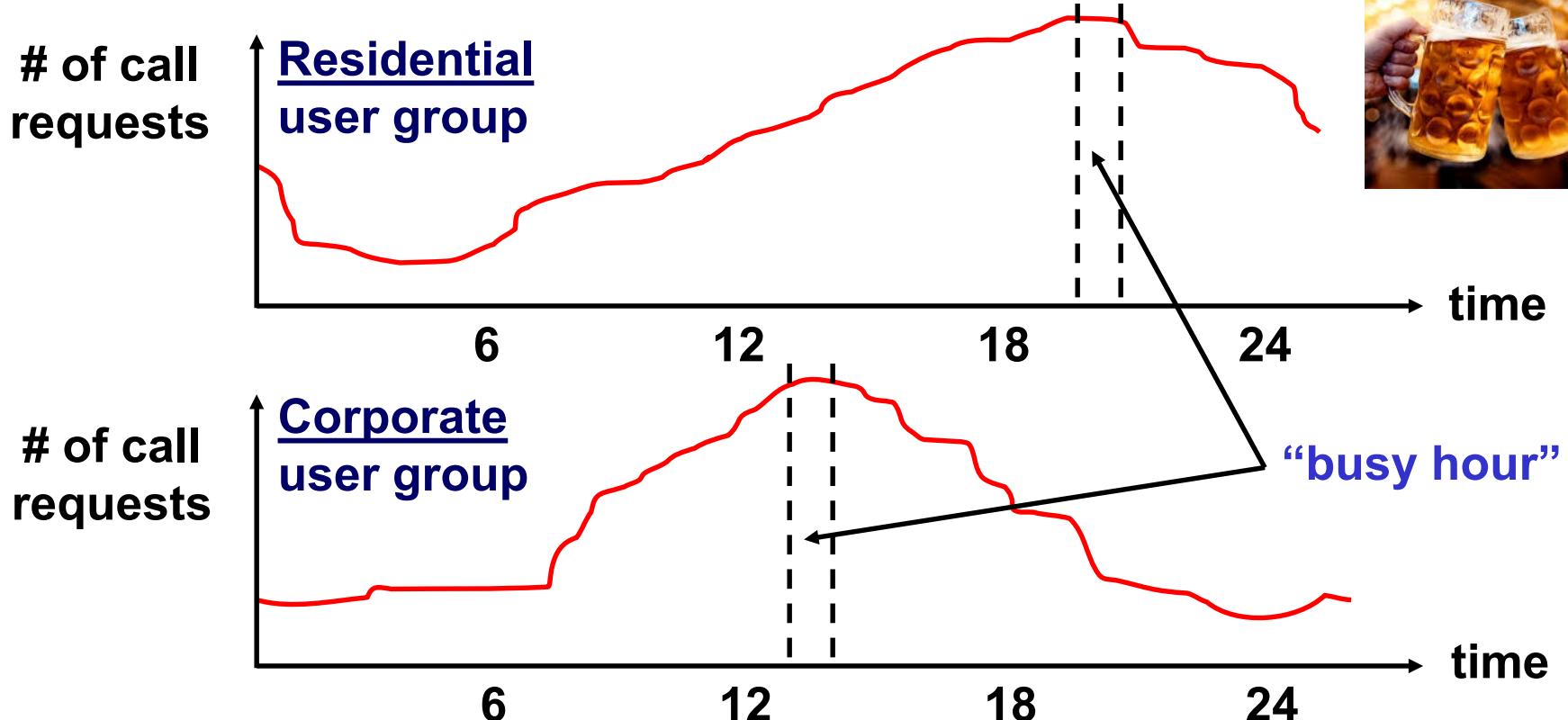


Traffic Aggregation



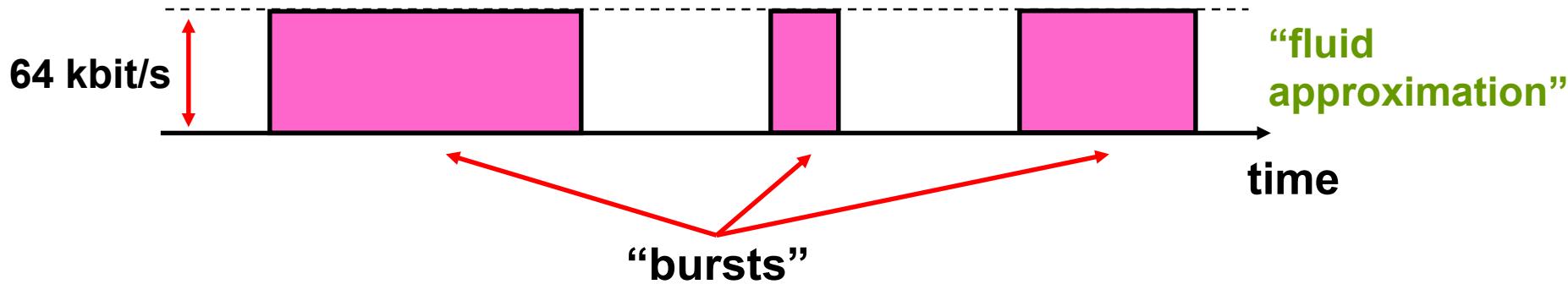
- Traffic over different times scales
- Usually, traffic ‘flattens out’ over longer times scales
- Traffic on shorter time scales usually more ‘bursty’ (peaked)

Call Level Arrival Processes



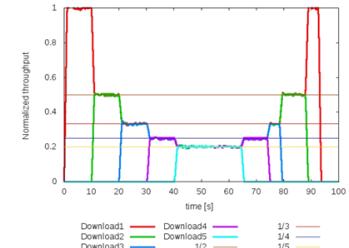
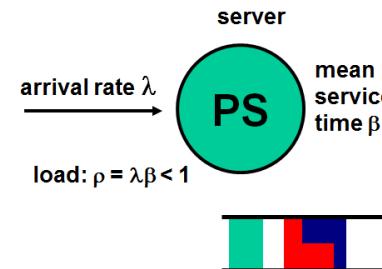
- Poisson arrivals, interarrival times \sim exponential with mean $\lambda(t)$
- Use concept of **“busy hour”** time varying
- Assume $\lambda(t)$ is “locally stationary”, for example per hour: $\lambda = \max_{i=1,\dots,24} \lambda_i$

Burst/Flow Level Models



For voice telephony:

1. **No silence suppression**, constant stream of 64 kbit/s
2. **With silence suppression**
 - during talk, spurts of 64 kbit/s
 - during silences, 0 kbit/s



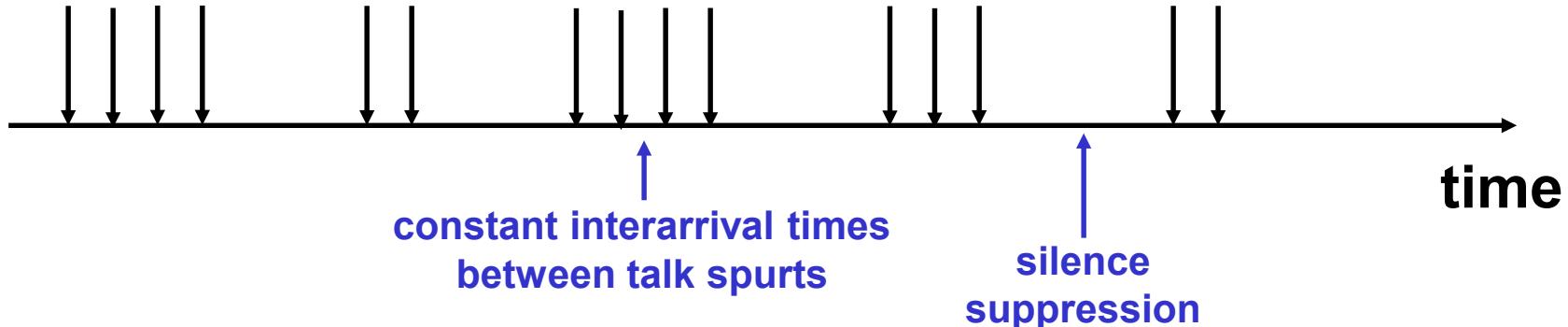
For data transfers:

1. File transfers seen as a single flow ("train of packets")
2. TCP elastic traffic → Processor Sharing (PS) model



Packet Level Models

For voice telephony:

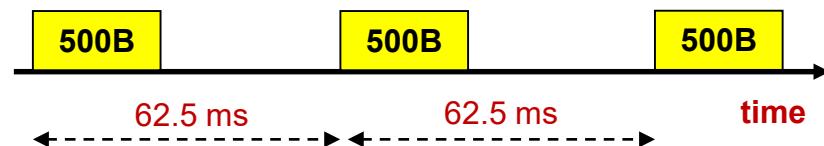


Two types of voice:

1. No silence suppression, 64 kbit/s

Suppose a packet is 500 bytes, then packet interarrival time =

$$\frac{(500 \times 8) \text{ bits}}{64 \times 10^3 \text{ bits/s}} = 0.0625 \text{ s} = 62.5 \text{ ms}$$



2. With silence suppression

Data transfers: Highly complex dynamics due to feedback loops, flow control, overload control, user behavior,...

Tolerance to Error and Delay



↑
error
tolerance



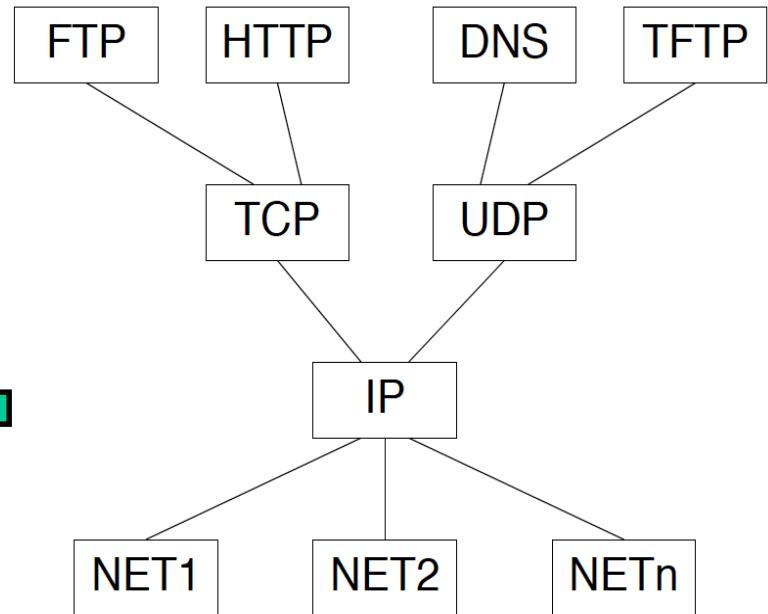
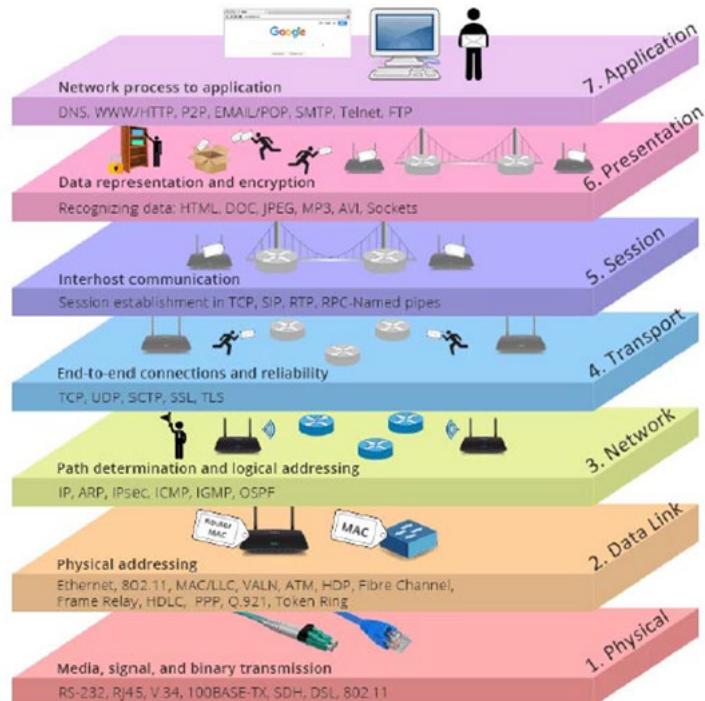
UDP

TCP

- Classification is based upon ITU-T G.1010 (standard)



IP as Generic Protocol that Runs over All Networks



OSI stack

“hour-glass model”

Quality over IP Networks



- **About IP in general**

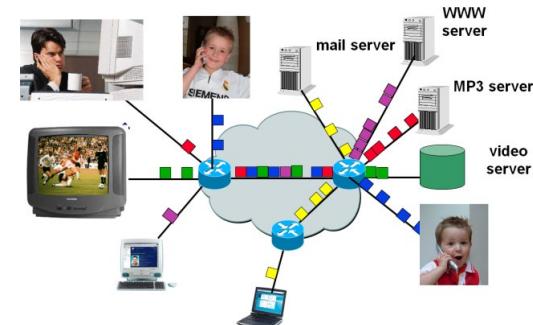
- IP is THE network that enables different networks to “talk” to each other
- IP is the basis for the Internet

- **However...**

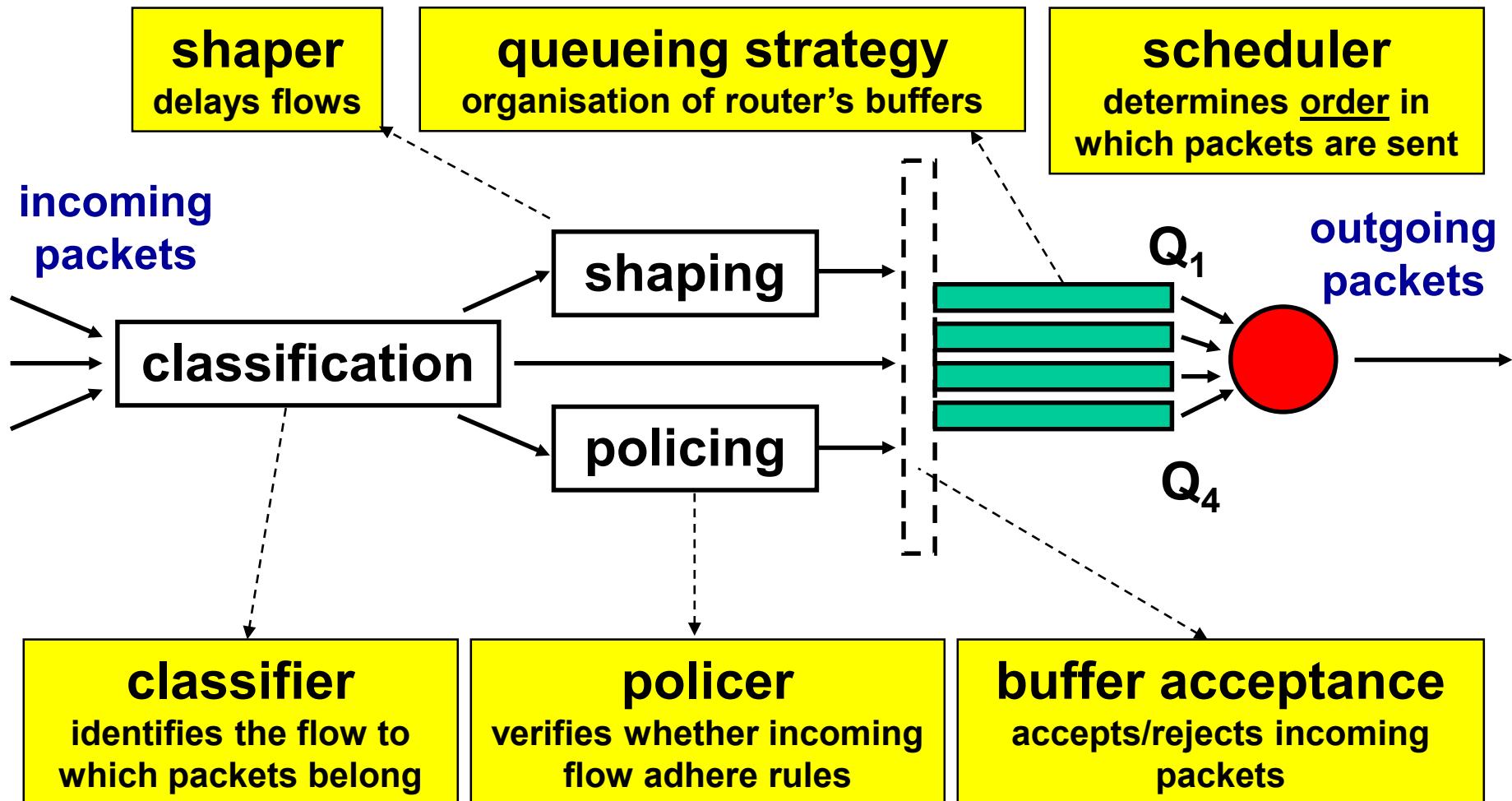
- IP is not “connection-oriented”
- IP does not provide QoS guarantees: “best effort” service

- **How to realize QoS over IP networks?**

- over-provisioning is an option, but may be expensive...
- need to “squeeze the most out of the network”
 - **Traffic Management**...



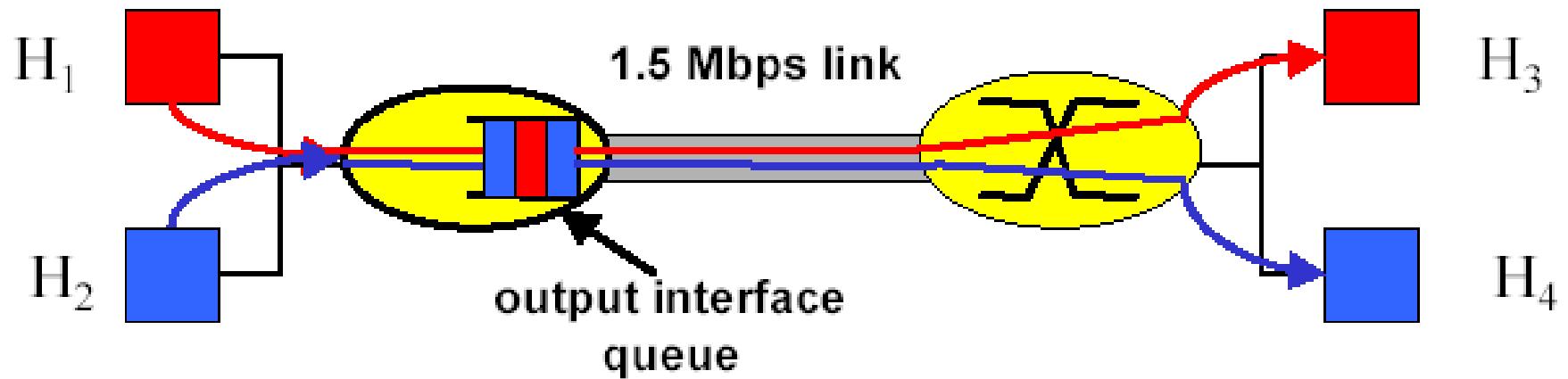
QoS Functionalities of Routers



Idea: Routers can implement all kinds of QoS mechanisms



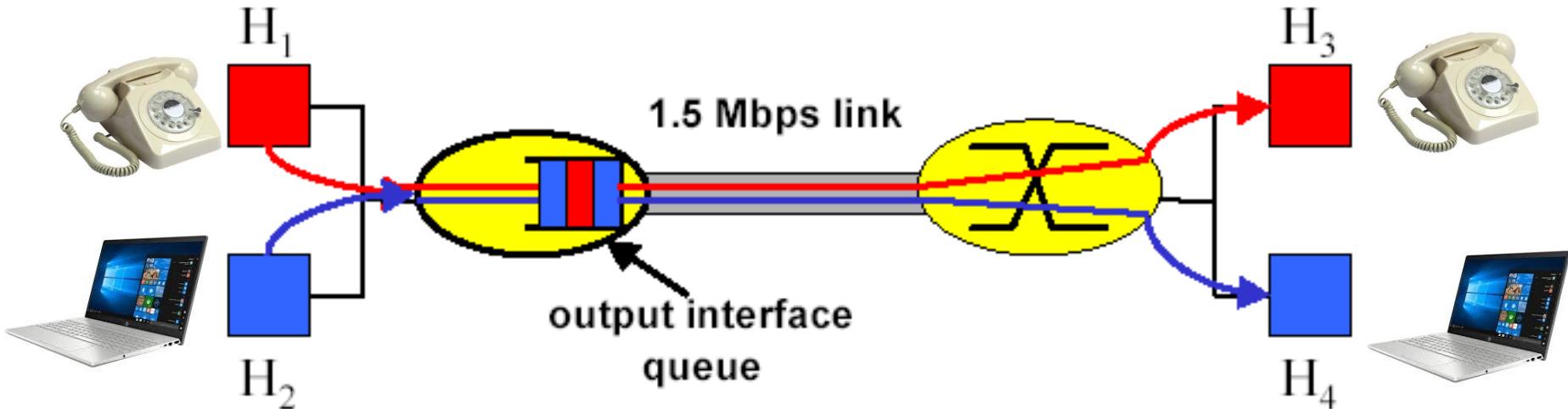
“Dumbbell” Topology



- Simple model for sharing and congestion topologies
- **Shared link** connecting the two sites is bottleneck



Packet Marking



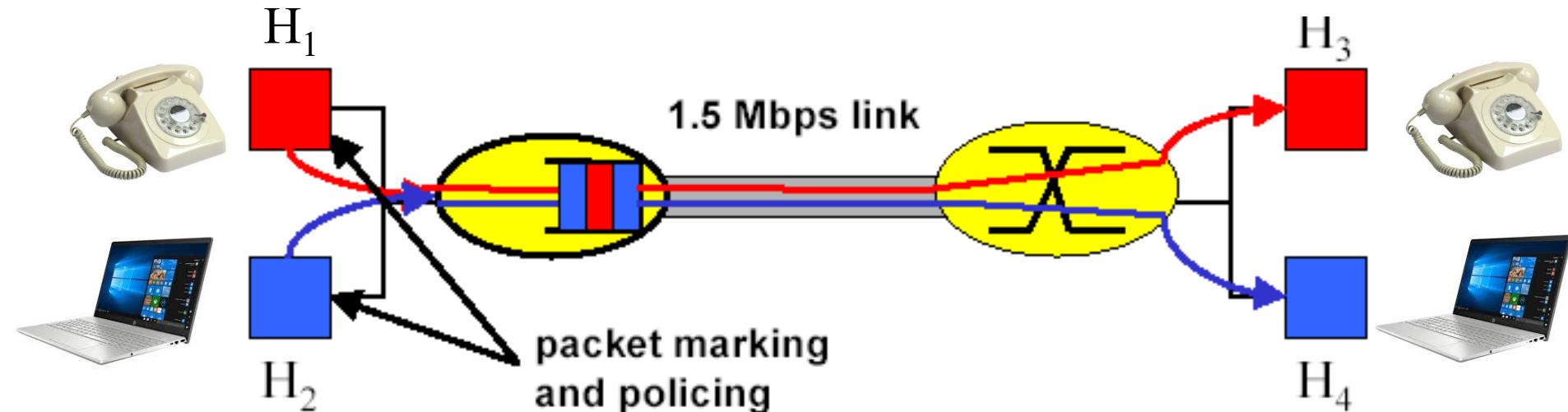
Marking of packets is needed for router to distinguish between different classes (“coloring of packets”)

Router policies are needed to treat packets accordingly

- for example: priorities of different classes (real-time traffic may get priority over non-real-time traffic)



Traffic Policing

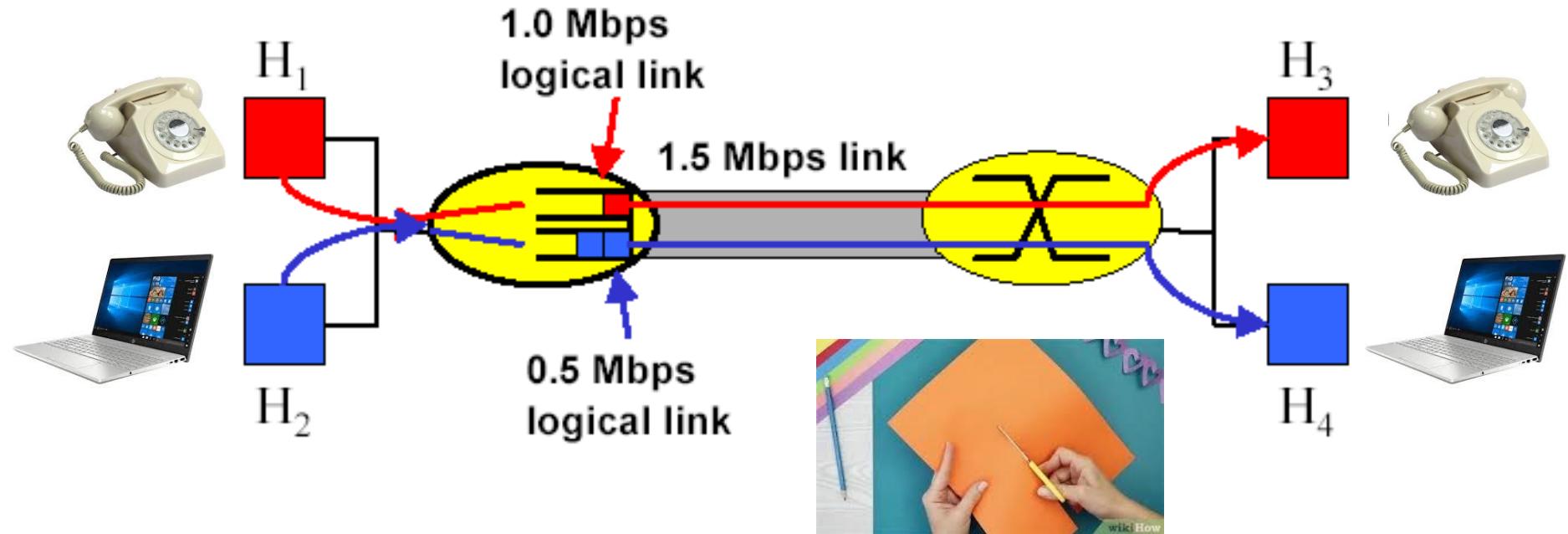


- **Problem:** Applications may **misbehave** by sending more packets than assumed or agreed upon in SLA
- **Protection** mechanism needed for one class from other classes
- Requires policing mechanisms to **filter traffic streams** to ensure that source does not violate agreement on bandwidth usage





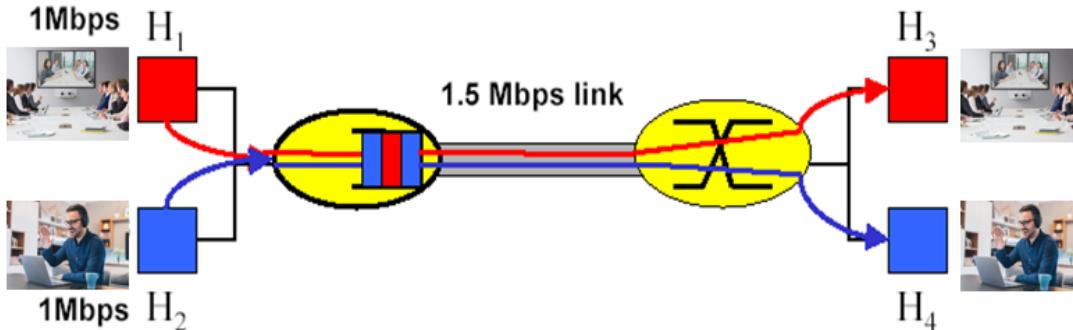
Bandwidth Partitioning



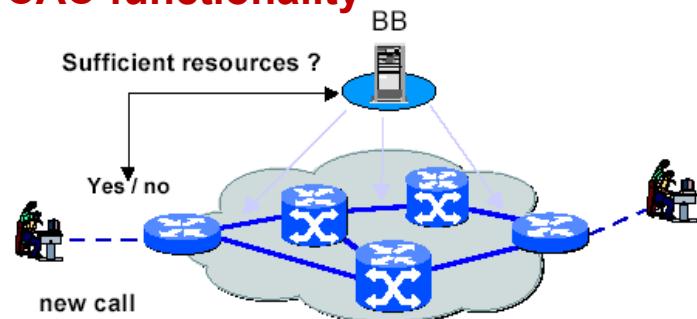
- Alternative to marking and policing: allocate **fixed proportion of bandwidth** to each application flow (“dedicated bandwidth”)
- Generally leads to **inefficient use** of bandwidth
- While providing isolation, it is desirable to **use resources as efficiently** as possible



Admission Control



CAC functionality



Problem: One can not support traffic beyond link capacity



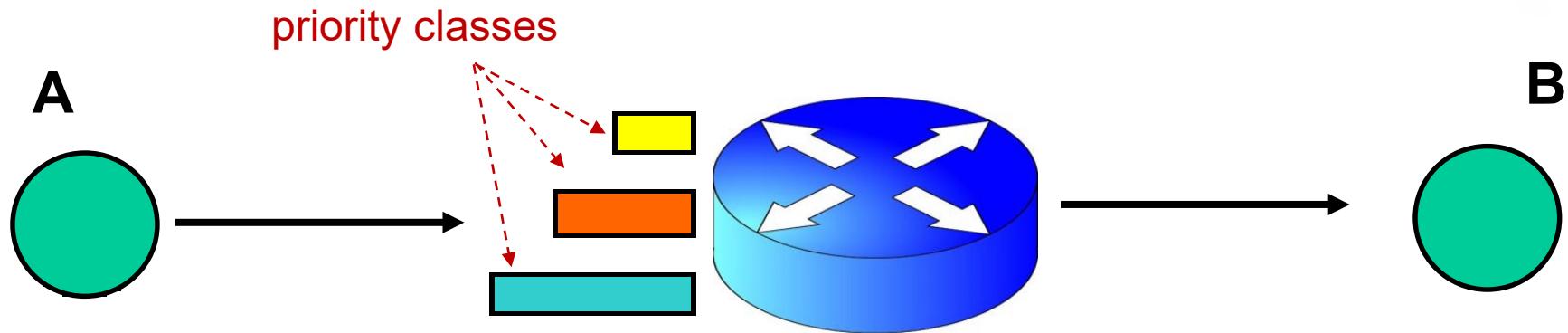
Solution: “Call” (of “Connection”) Admission Control (CAC)

Application flow declares its needs, the **network may reject** call if it cannot satisfy needs

If the network accepts too many “calls”, then QoS agreements with other customers may be violated (leading to fines)



Packet Classification



Differentiated Services (“DiffServ”)

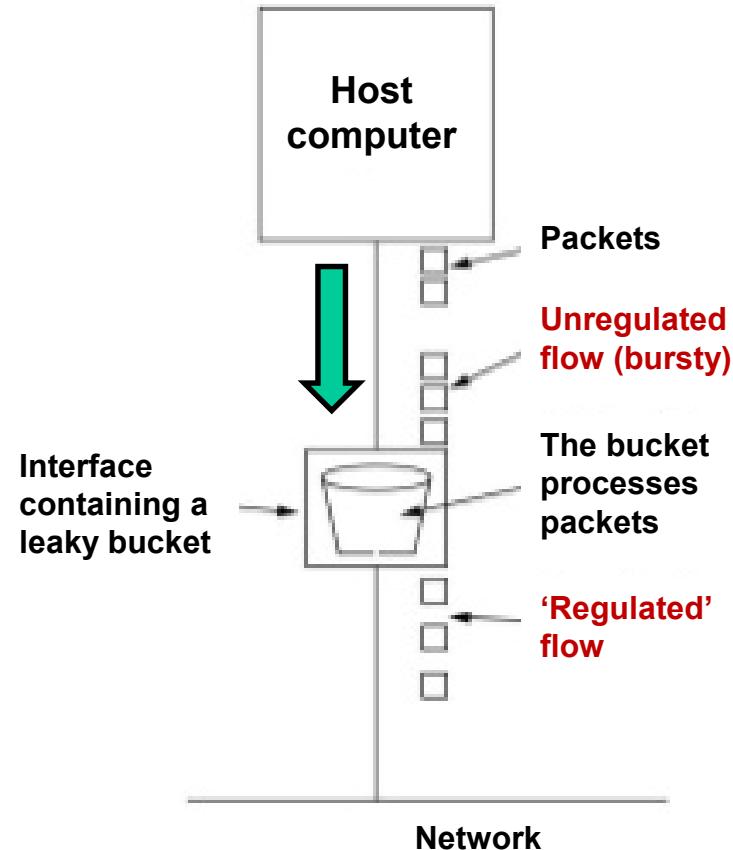
- Definition of small number of priority classes
 - For example: low-latency (voice, streaming), best-effort (Web, file transfer)
- DiffServ-aware routers implement “per-hop behavior” packet forwarding
- Benefit: scalability no issue
- Drawback: performance only relative, no absolute QoS guarantees



Traffic Regulation



Leaky Bucket with water



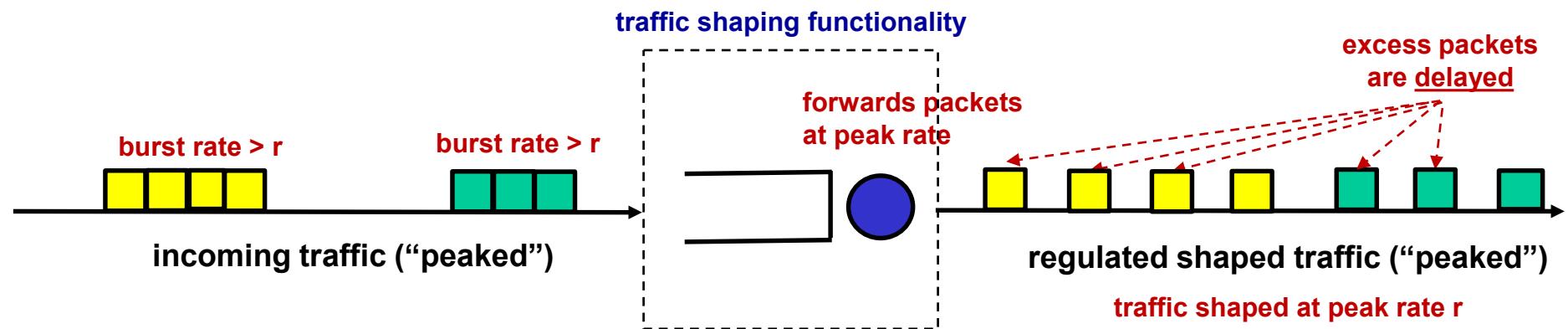
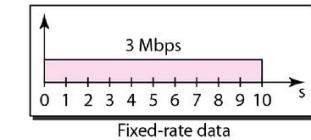
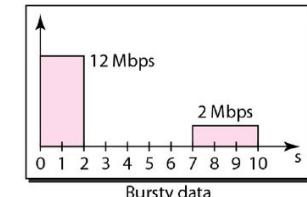
Leaky Bucket with packets

Regulation mechanisms: Traffic shaping and policing

Traffic Shaping



Traffic shaping is aimed at “topping off” peaks (at peak rate r)



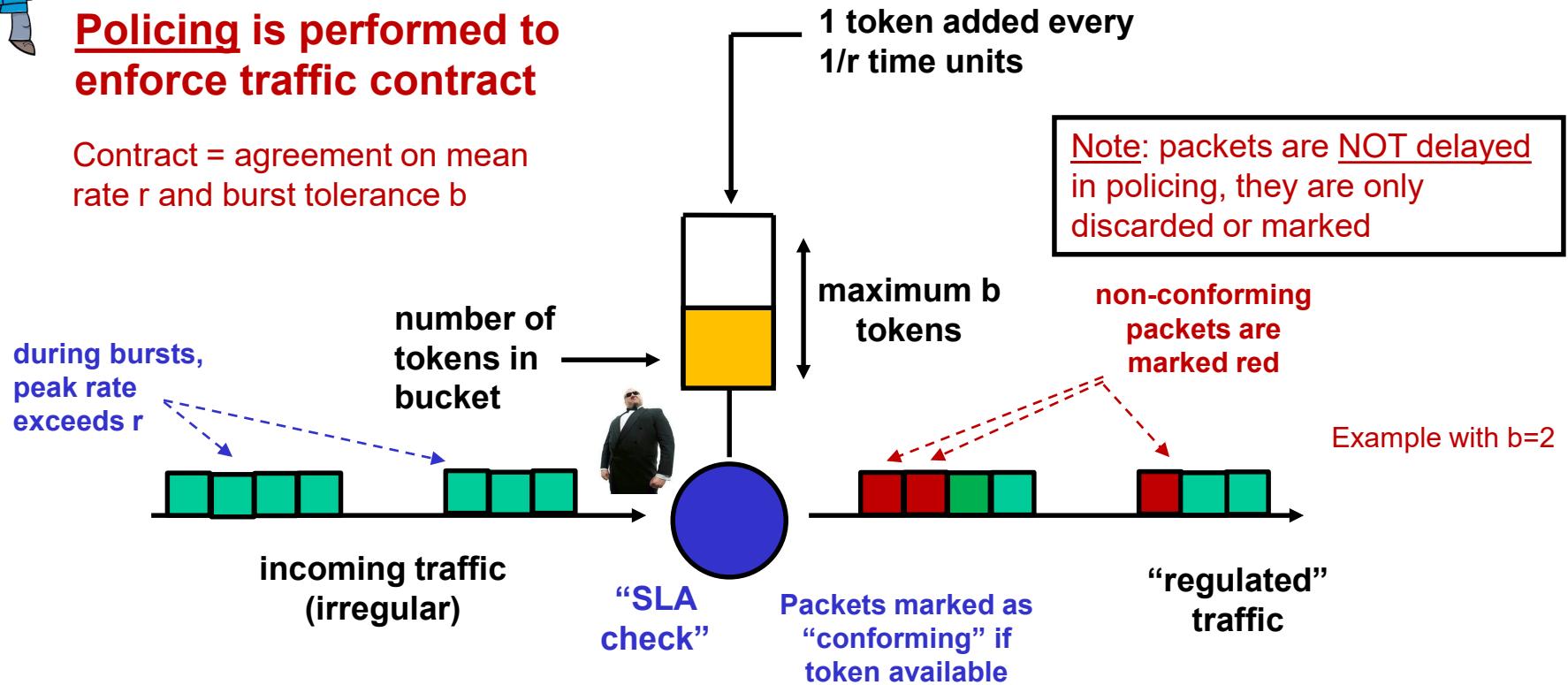
- Shaping functionality is aimed at **filtering bursty traffic before going into the network**
- Packets are **delayed** if no tokens available (during peaks)



Traffic Policing

Policing is performed to enforce traffic contract

Contract = agreement on mean rate r and burst tolerance b

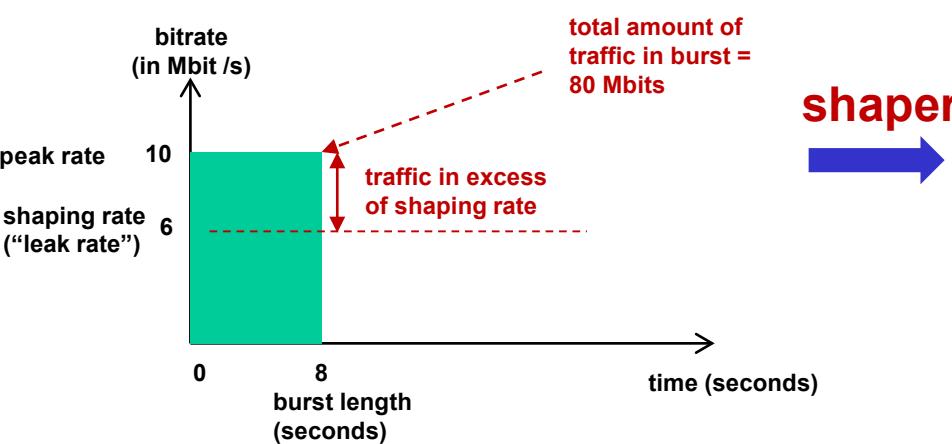
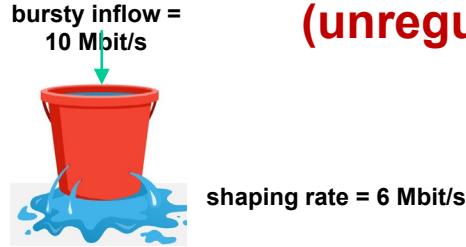


- Leaky Bucket (LB) implementation
 - r is average rate, b = size of token bucket ("burst tolerance")
- Number of tokens represents **amount of credit** of packets that arrive at faster rate than the agreed-upon mean rate r
- During period of T seconds, LB declares at most $b+Tr$ bytes as "conforming"
- Non-conforming packets are **marked** (e.g., discarded, or given low-priority)

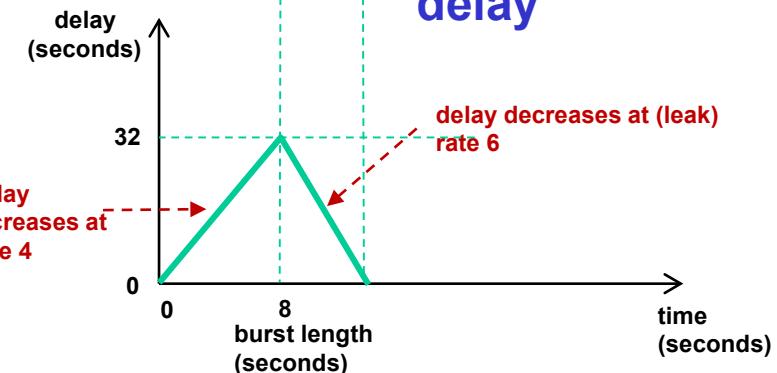
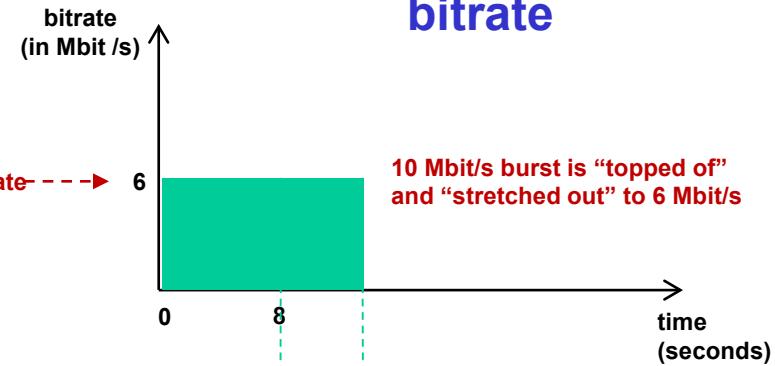


Traffic Shaping: Example for a single traffic burst

Incoming traffic before shaping (unregulated)



Traffic after shaping (regulated)

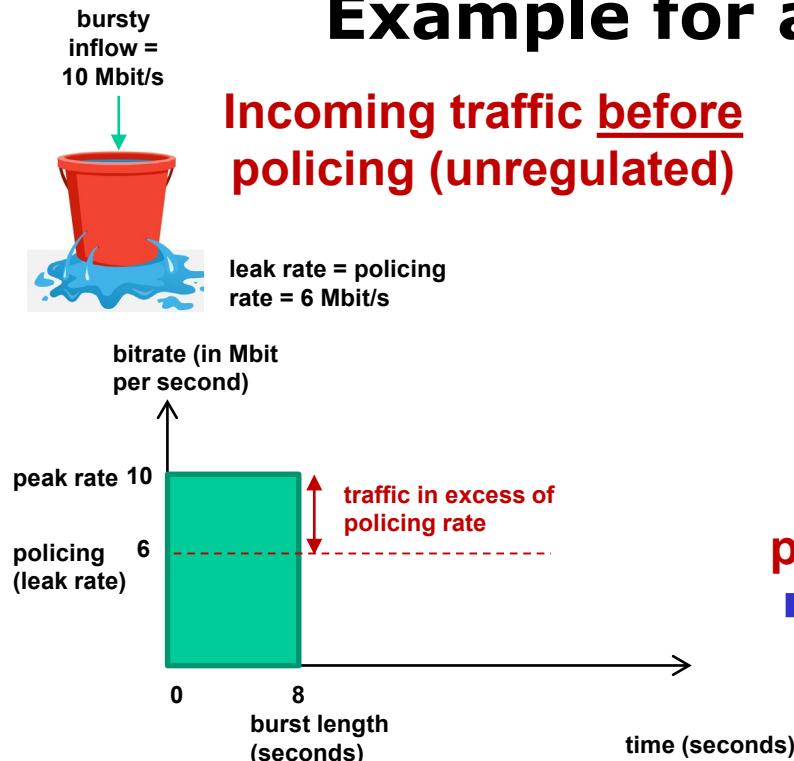


The way shaping works:

- When **burst rate > shaping rate** → shaping buffer level rises (at a rate equal to peak rate *minus* shaping rate)
- The delay then equals buffer level / shaping rate



Traffic Policing: Example for a single traffic burst



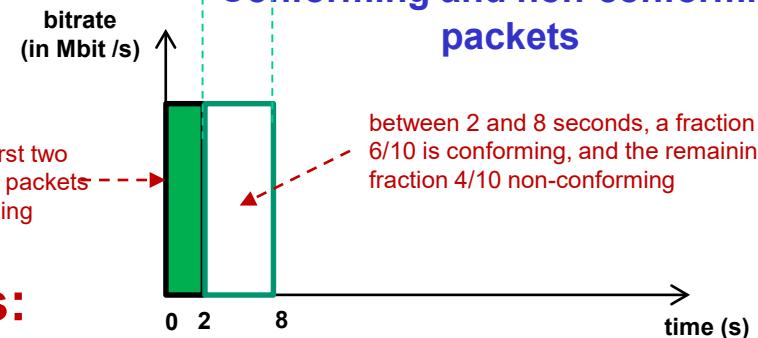
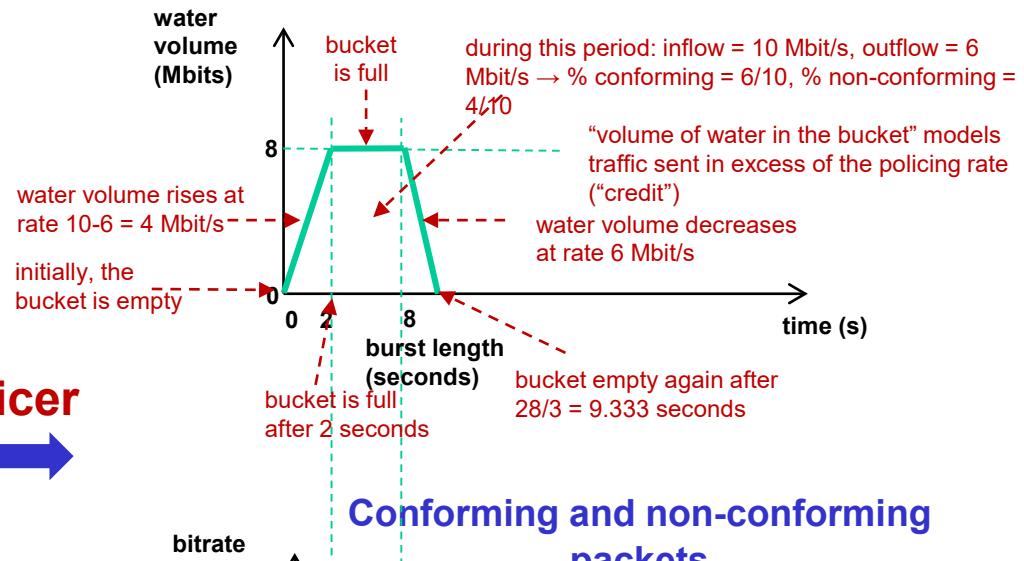
Assumptions for this specific example:

1. Peak rate = 10 Mbit/s, policing rate 6 Mbit/s
2. Burst tolerance = 1 Mbyte = 8 Mbit ("volume of the bucket")

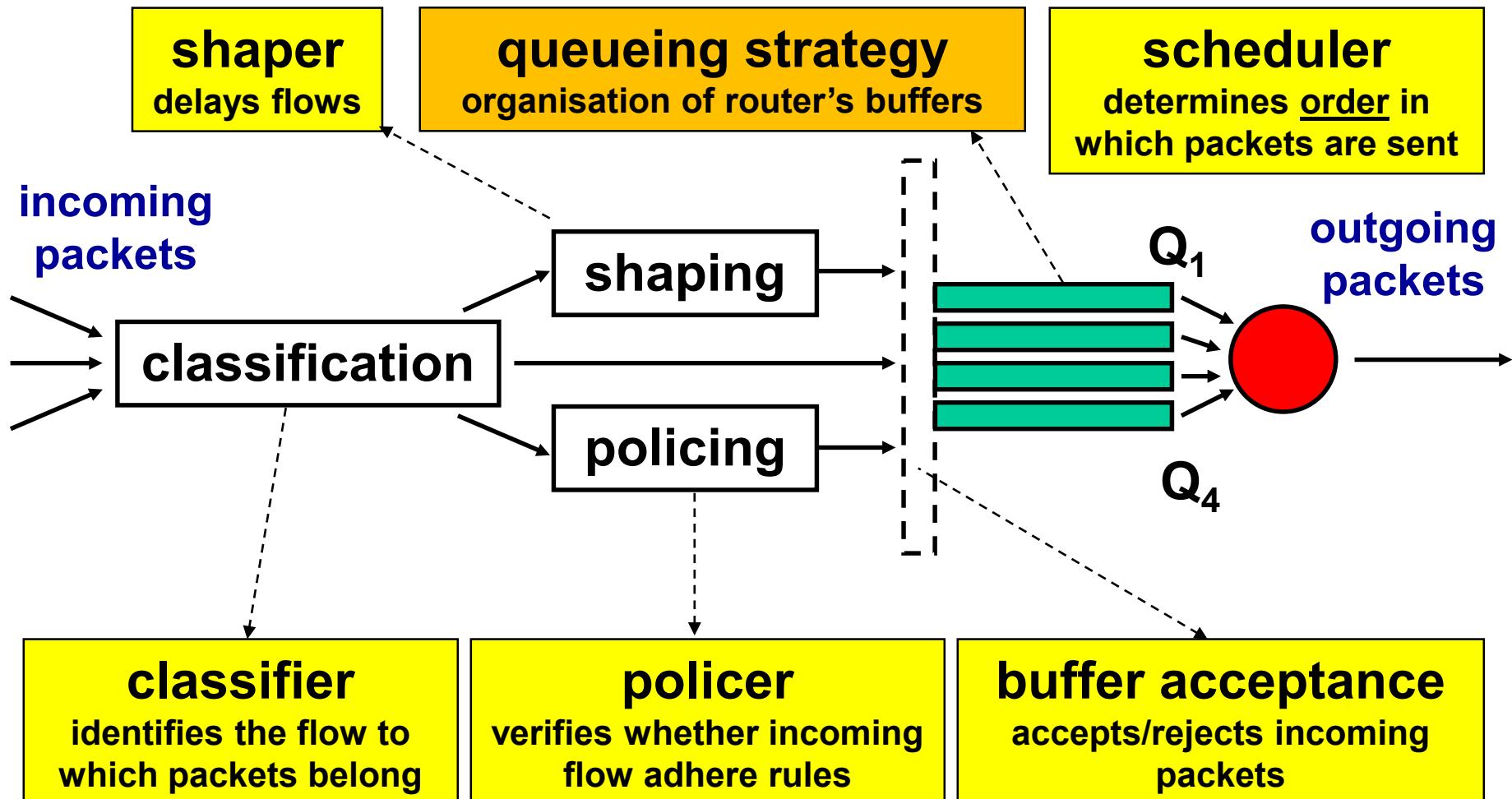
The way leaky bucket policing works:

1. When burst rate > policing rate => water level in the bucket rises
2. When **water volume = burst tolerance** (that is: "the bucket is full") only a fraction "f" of the packets are marked conforming and the other packets as non-conforming
3. Fraction conforming "f" is given by $f = \text{policing rate} / \text{peak rate}$, and fraction non-conforming by $1-f = (\text{peak rate} - \text{policing rate}) / \text{peak rate}$

"Water volume in leaky bucket"



QoS Functionalities of Routers



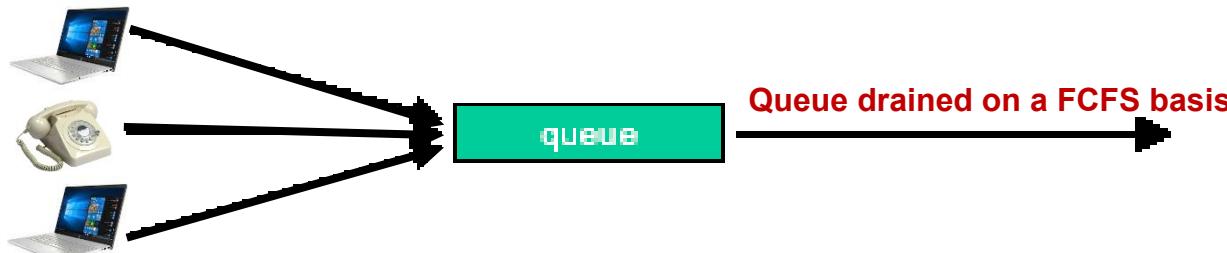
Idea: Routers can implement all kinds of QoS mechanisms



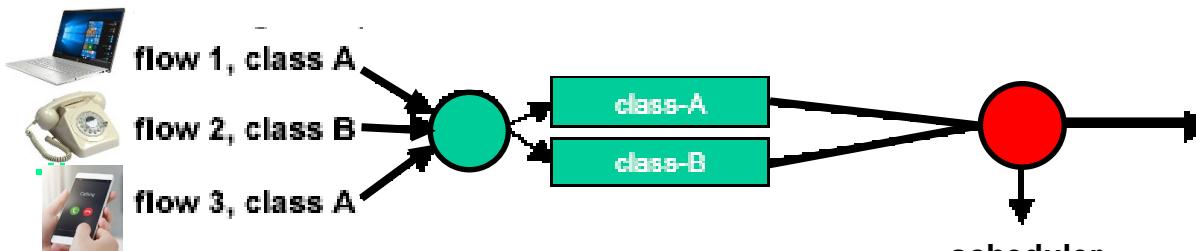
Queue Management

Many options to divide the buffer space in logical queues

1. One logical queue for all flows together

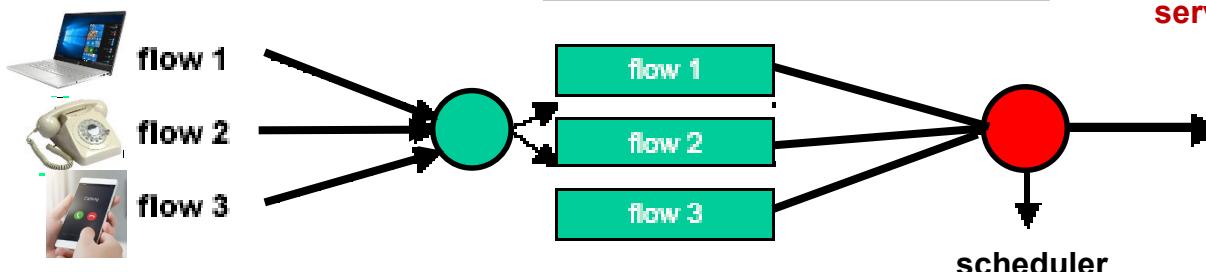


2. One logical queue for one class of flows



3. One logical queue for each individual flow

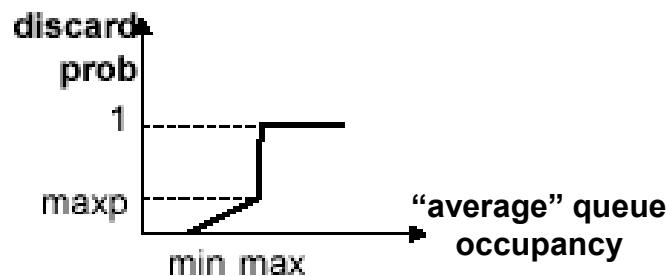
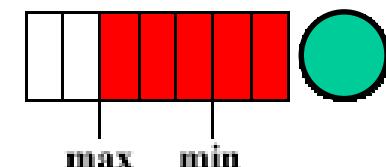
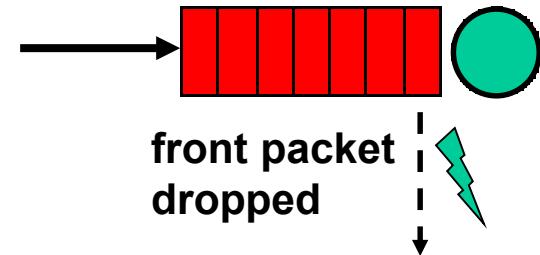
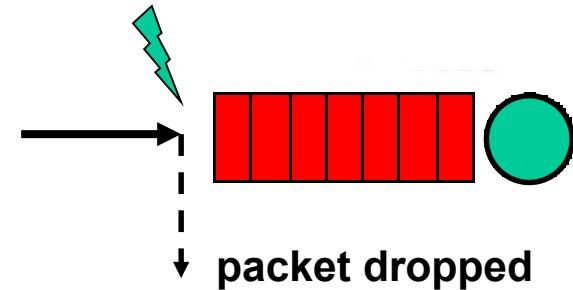
Scheduler determines the order in which the queues are served



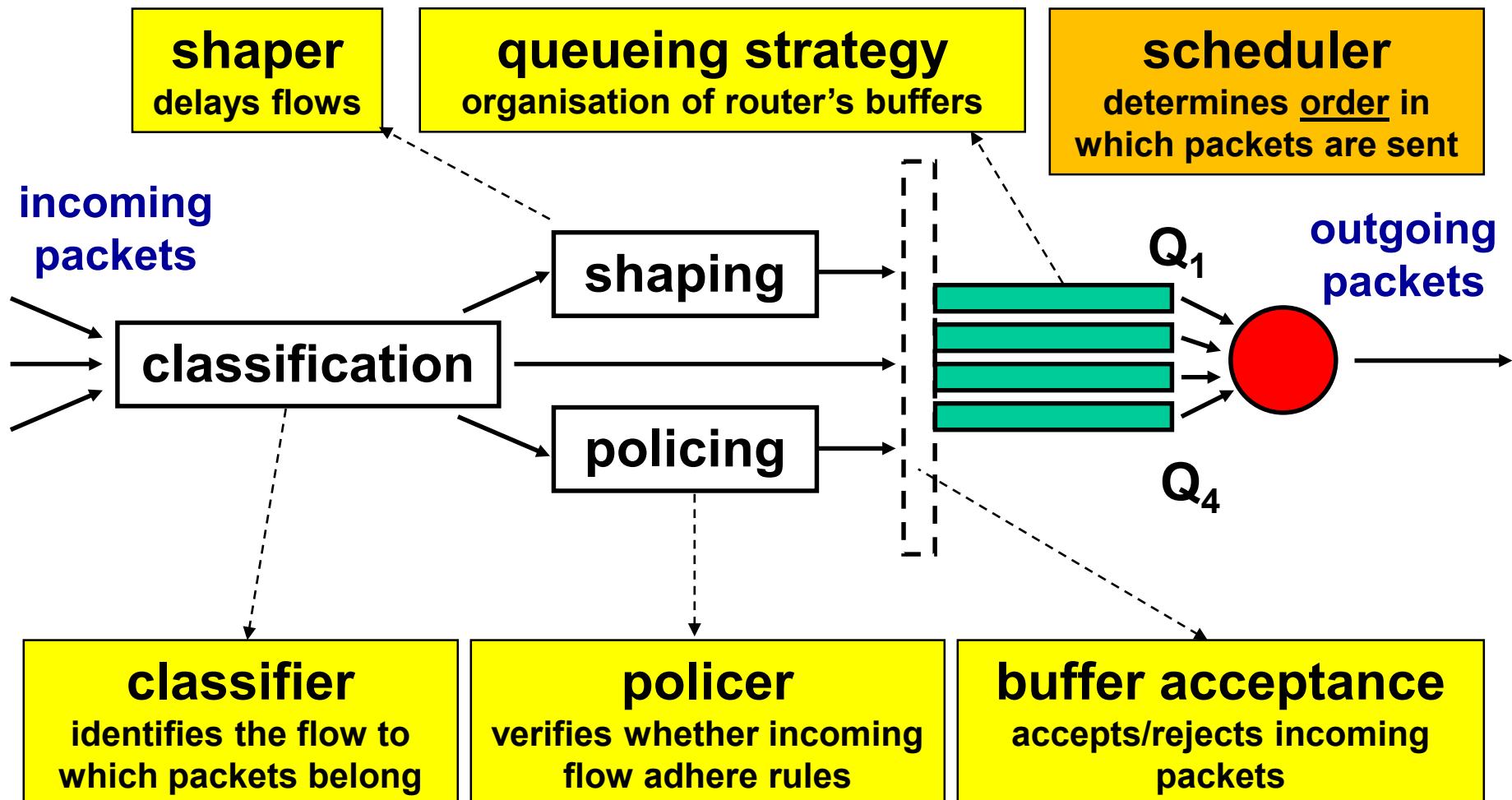


Queue Management

- **Tail drop:** drops incoming packets when buffer is full
- **Front drop:** when buffer is full, drops packet there are in the front
- **Random Early Discard (RED):** when average buffer occupancy above threshold, drop packets at random



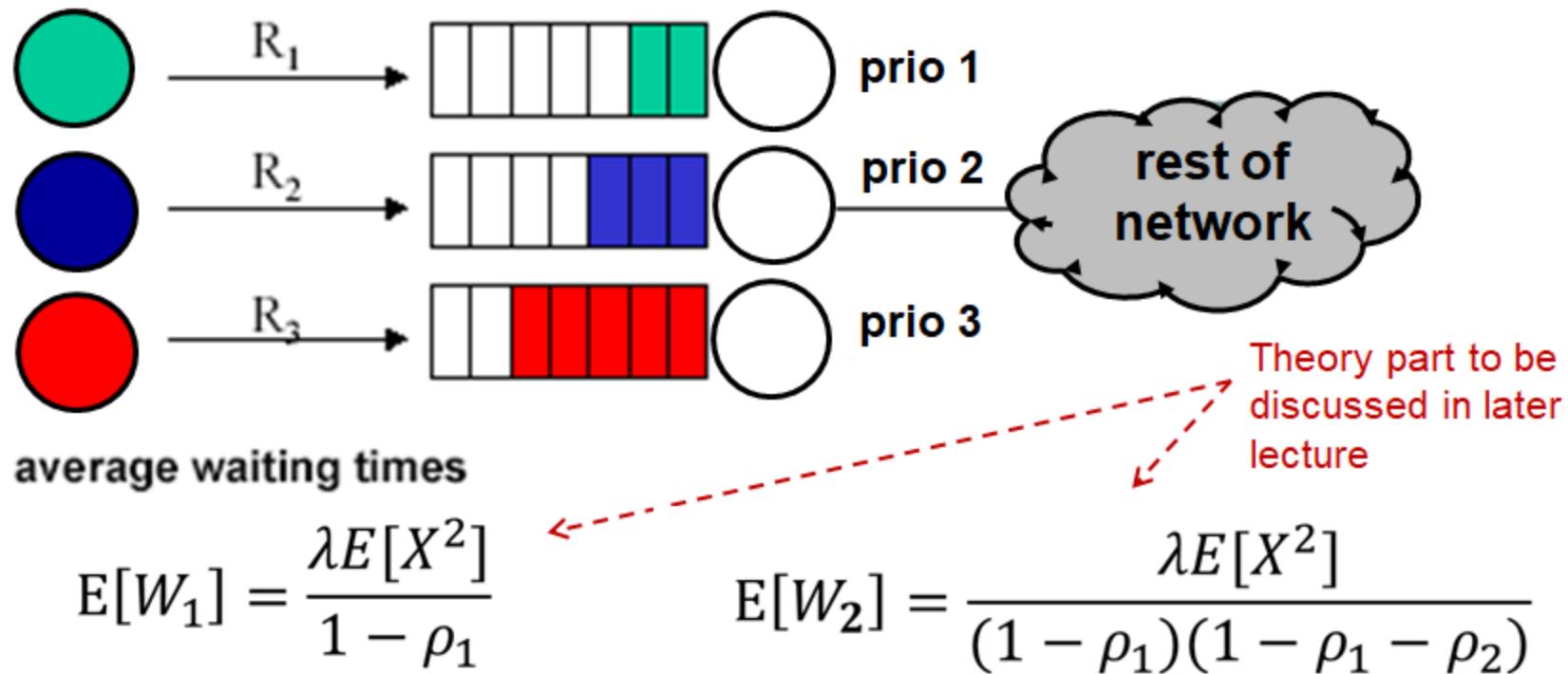
QoS Functionalities of Routers



Idea: Routers can implement all kinds of QoS mechanisms



Priority Scheduling

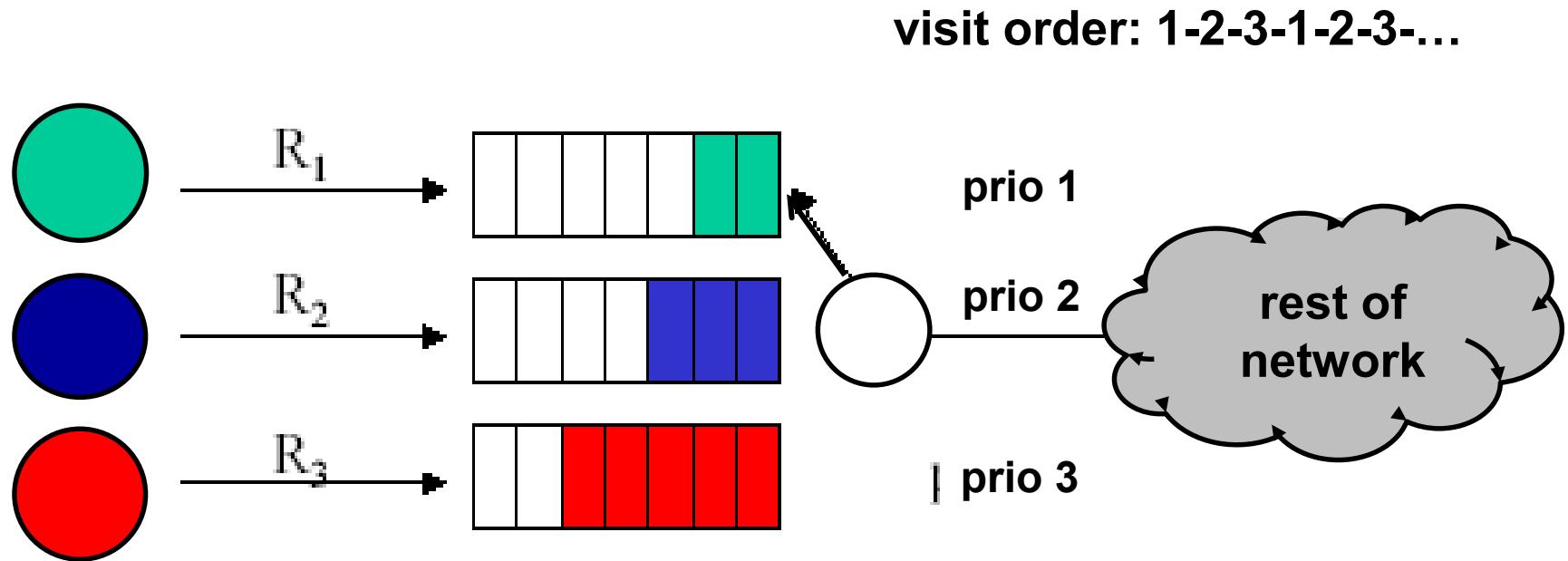


- **Priority scheduling**

- multiple queues, served in order of priority
- green packets high priority, blue medium, red low
- possible starvation for low-priority queues



Round-Robin Scheduling

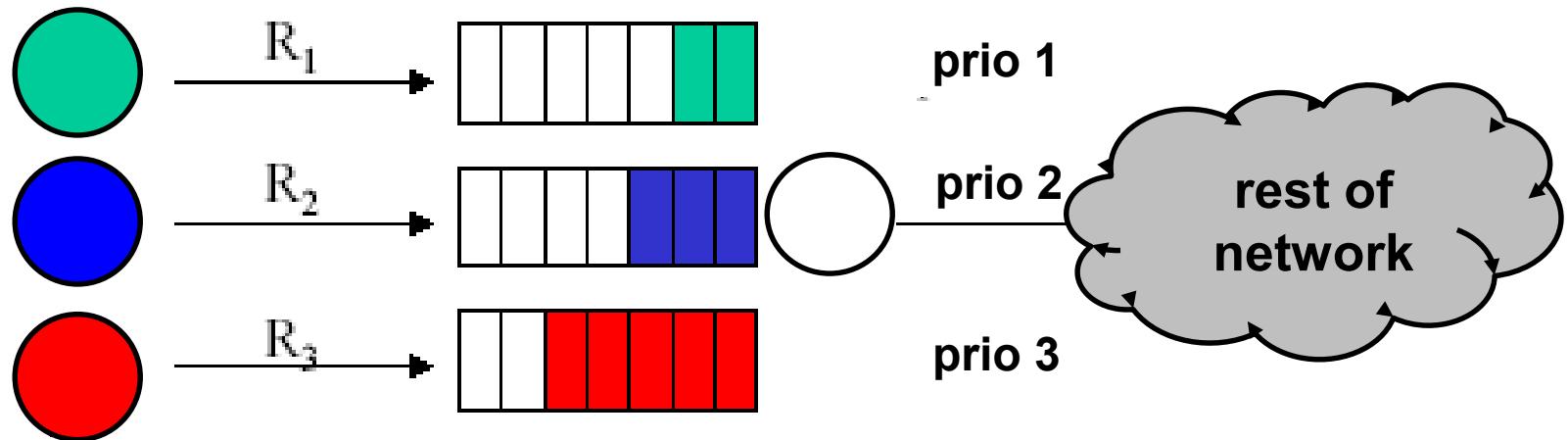


- **Round Robin scheduling**

- multiple queues, served in circular order
- ultimately fair, but may not be good for well-paying high-priority customers



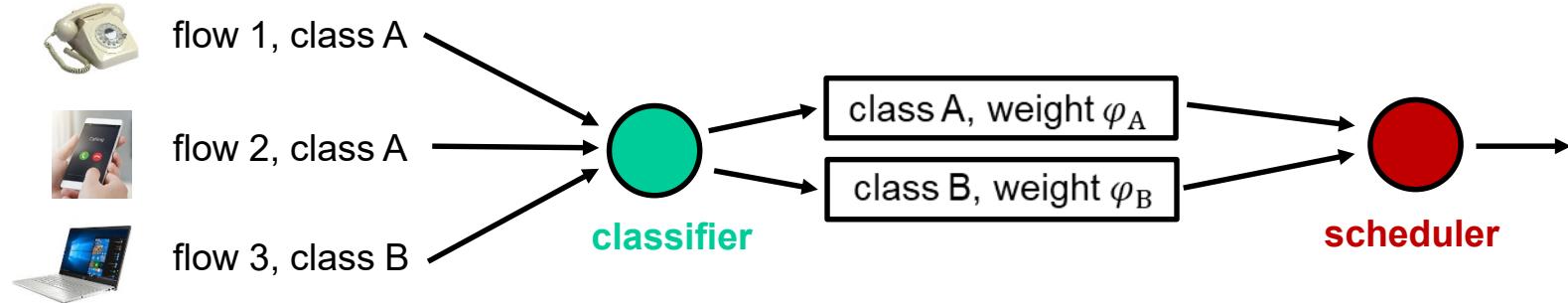
Weighted Fair Queueing



Example visit order: 1-1-1-2-2-3-...

- **Weighted Fair Queueing (WFQ) scheduling**
 - associate “weight” with each class, which determines the relative number of service units in each cycle

Generalized Processor Sharing

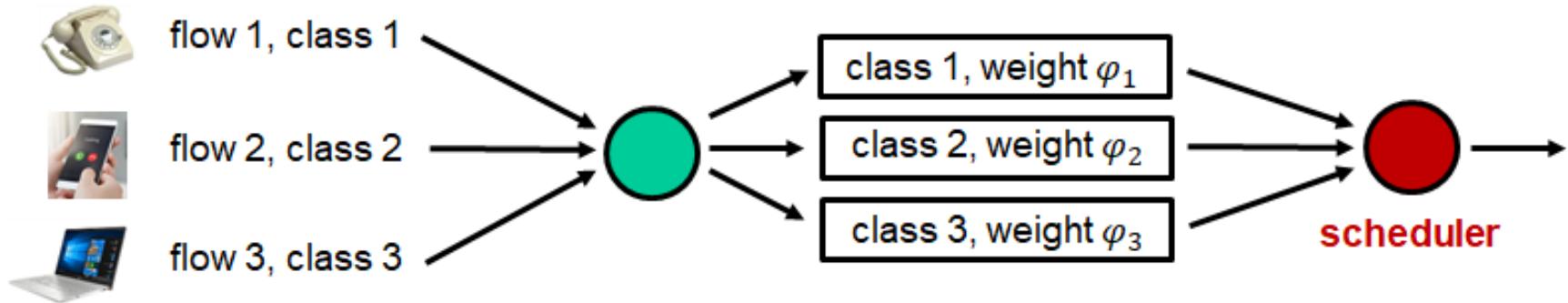


- **Generalized Processor Sharing scheduling**
 - “idealized” work-conserving scheduler
 - traffic flows are approximated as “fluid” flows
 - each flow class k has weight φ_k , and receives at least fraction φ_k of available bandwidth
 - within each class k , bandwidth is equally shared among all the flows in class k (in a Processor Sharing fashion)

Generalized Processor Sharing



Example

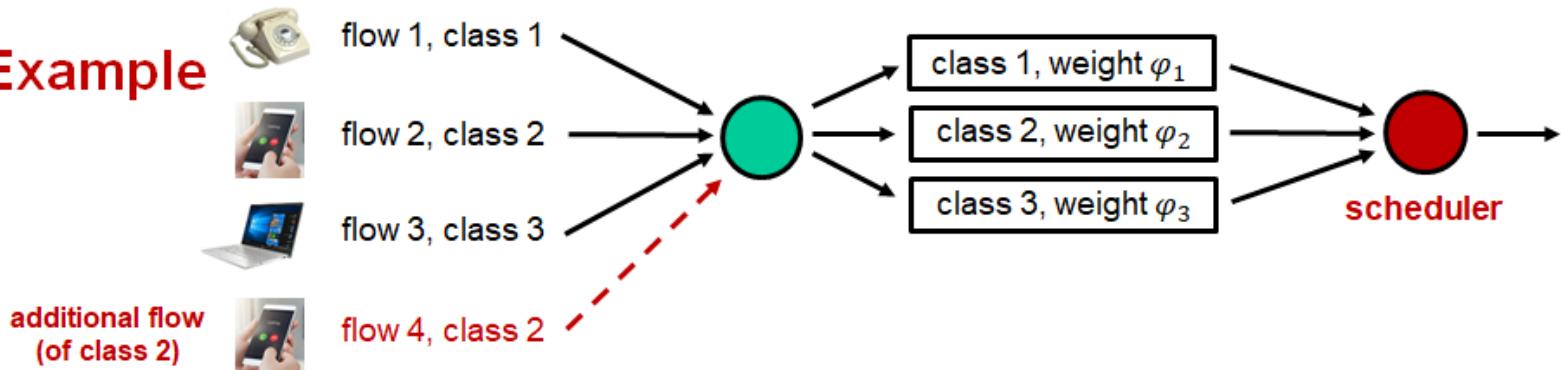


class k	weight φ_k	# of flows in class k	Minimum fraction of BW for <u>each</u> flow in class k	Minimum fraction of BW for class k in total
1	2	1	$2/(1*2+1*3+1*5) = 2/10$	2/10
2	3	1	$3/(1*2+1*3+1*5) = 3/10$	3/10
3	5	1	$5/(1*2+1*3+1*5) = 5/10$	5/10

Generalized Processor Sharing



Example



class k	weight φ_k	# of flows in class k	Minimum fraction of BW for <u>each</u> flow in class k	Minimum fraction of BW for class k in total
1	2	1	$2/(1*2+1*3+1*5) = 2/10$	$2/10$
2	3	1	$3/(1*2+1*3+1*5) = 3/10$	$3/10$
3	5	1	$5/(1*2+1*3+1*5) = 5/10$	$5/10$



class k	weight φ_k	# of flows in class k	Minimum fraction of BW for <u>each</u> flow in class k	Minimum fraction of BW for class k in total
1	2	1	$2/(1*2+2*3+1*5) = 2/13$	$2/13$
2	3	2	$3/(1*2+2*3+1*5) = 3/13$	$6/13$
3	5	1	$5/(1*2+2*3+1*5) = 3/10$	$5/13$

additional flow

Performance of Networked Systems



Lecture 4: Performance and Traffic Management in IP Networks

Overview of today's lecture

1. Processor Sharing models for elastic traffic
2. Traffic characteristics at different time scales
3. Classification of applications: error and delay tolerance
4. Traffic Management mechanisms for IP networks