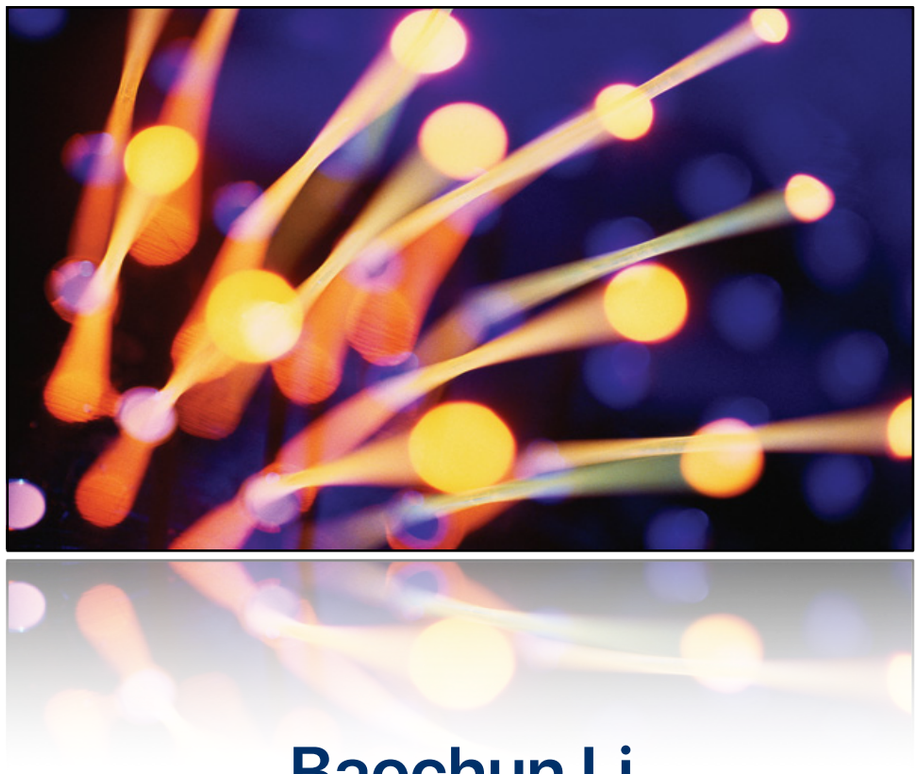
"Fairness"



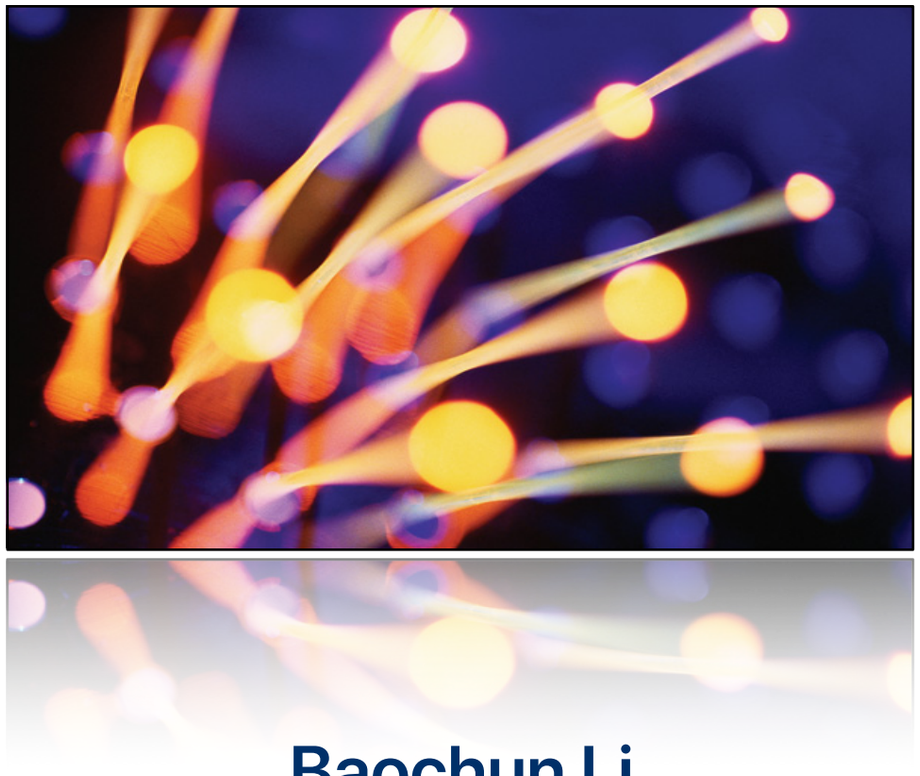
Baochun Li

Department of Electrical and Computer Engineering

University of Toronto

Episode 5. Scheduling and Traffic Management

Part 1



Baochun Li
Department of Electrical and Computer Engineering
University of Toronto

Keshav Chapter 9.1, 9.2, 9.3, 9.4, 9.5.1, 13.3.4

Outline for the entire episode

What is scheduling?

Why do we need it?

Requirements of a scheduling discipline

Fundamental choices

Scheduling best effort connections

Scheduling guaranteed-service connections

Scheduling

Sharing resources always results in contention

A scheduling discipline resolves contention —

who's next?

Key to fairly sharing resources and providing performance guarantees

thus important for Quality of Service provisioning

Example of a web server

Consider a set of queries made to a web server

- Each query represents a service request contending for shared resources — the web server
- Assume the server can serve only one request at a time
- Requests that arrive when a server is busy must wait

A busy server holds an incoming request in a service queue

- The request will eventually be selected for service
- The queuing delay of a request is defined as the time between its arrival and eventual service
- The server allocates different mean queuing delays to different requests by its choice of service order

The scheduling discipline of a server makes the decision

Scheduling disciplines

A scheduling discipline is important for —

Fairly sharing network resources

Provide performance-critical applications, such as telephony and interactive multi-party games, with Quality of Service guarantees

It has two orthogonal components —

It decides the order in which requests are serviced

It manages the service queue of requests awaiting service

If packets cannot be served and the service queue overflows, which one should be dropped?

Intuition: allocate different loss rates to different users

Where should we use scheduling?

Usually studied in the literature for the output queue of a switch — in the network layer

In the web server example — in the application layer

In general — any layer dealing with a multiplexed resource must deal with scheduling

Example: Job scheduling in datacenter clusters

But only in packet-switched networks — circuit-switched networks do not need scheduling!

Outline

What is scheduling?

Why do we need it?

Requirements of a scheduling discipline

Fundamental choices

Scheduling best effort connections

Scheduling guaranteed-service connections

Two types of applications

We expect two types of applications —

Best-effort applications (adaptive, non-real time, bulk)

e.g. email, some types of file transfer

The performance requirements are **elastic: they can adapt to the resources available**

Guaranteed-service applications (non-adaptive, real time)

e.g. packet voice, interactive video, stock quotes

Voice: human ergonomic constraints require the round-trip delay to be smaller than 150 ms

Requires the network to **reserve resources** on their behalf

Why do we need a scheduling discipline?

The performance received by a connection from a source to a destination depends on the scheduling discipline at each switch along the path!

A switch queues requests, represented by packets ready for transmission, in a per-link output queue

At each output queue, a scheduling discipline is used to choose which ready packet to transmit next

What can a scheduling discipline do?

Allocate different mean delays to different connections

by a choice of service order

Allocate different bandwidths to different connections

by serving at least a certain number of packets from a connection

Allocate different loss rates

by giving more or fewer buffers

Determine how fair the network is

Even for best effort applications

The Conservation Law

First-Come-First-Served (FCFS) is simple, but cannot differentiate among connections

But though a more complex scheduling discipline can achieve such differentiation, the sum of the mean queuing delays received by the set of connections — weighted by their share of the link's load — is independent of the scheduling discipline!

- If the scheduler is work-conserving (it is idle only if its queue is empty)
- Called the Conservation Law
- It implies that a scheduling discipline can only reallocate the delays

Outline

What is scheduling?

Why do we need it?

Requirements of a scheduling discipline

Fundamental choices

Scheduling best effort connections

Scheduling guaranteed-service connections

Requirements of a scheduling discipline

An ideal scheduling discipline should —

be easy to implement (both guaranteed-service and best-effort applications)

be fair (for best-effort applications)

provide performance bounds (for guaranteed-service applications)

allow easy admission control decisions — decide whether a new flow can be allowed (for guaranteed-service applications)

Requirement 1. Ease of Implementation

Scheduling discipline has to make a decision once every few microseconds!

Should be implementable in a few instructions or hardware

for hardware: critical constraint is the memory required to maintain scheduling state (e.g., pointers to packet queues)

Work per packet should scale less than linearly with the number of active connections

O(N) is not good enough!

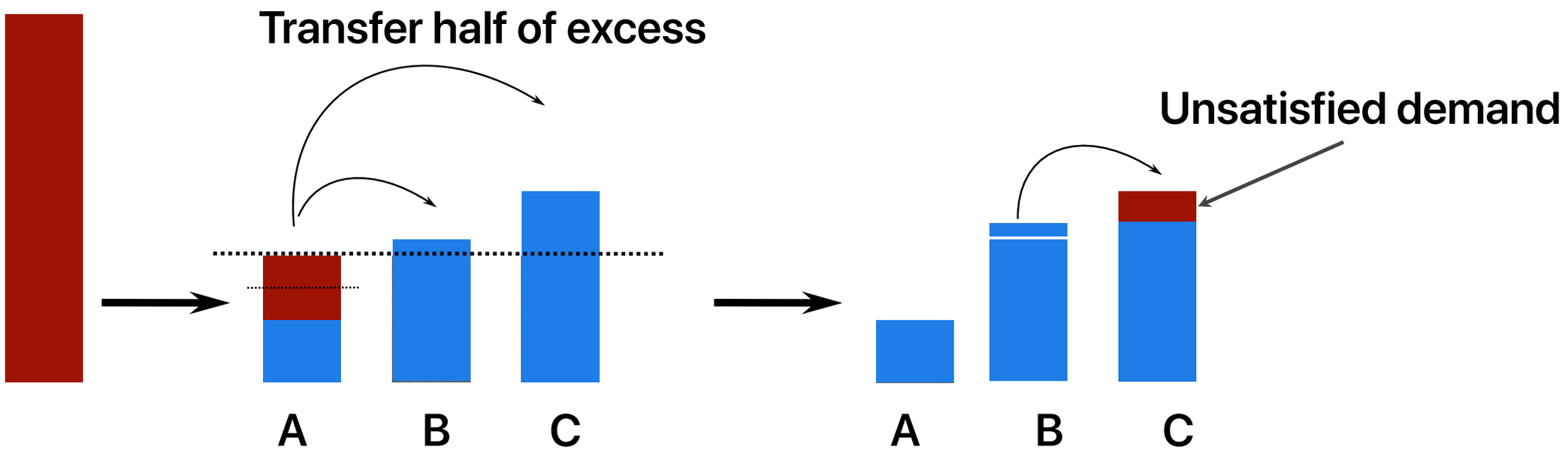
Requirement 2. Fairness

Scheduling discipline allocates a resource

An allocation is fair if it satisfies max-min fairness

each connection gets no more than what it wants

the excess, if any, is equally shared



Requirement 2. Fairness (continued)

Fairness is intuitively a good idea

But it also automatically provides protection

- traffic hogs cannot overrun others
- automatically builds firewalls around heavy users

Fairness is a global objective, but a scheduling discipline takes only local resource allocation decisions

- To translate from a local decision to a global one, each connection should limit its resource usage to the smallest locally fair allocation along its path

Dynamics → delay to regain global fairness → global fairness may never be achieved

Requirement 3. Performance Bounds

What is it?

A way to obtain an arbitrarily desired level of service

Restricted by the Conservation Law

First-Come-First-Served (FCFS) will offer a tight lower bound in terms of the sum of delays

A smaller delay in one connection → larger delays elsewhere

Can be expressed deterministically or statistically

"Every packet has a delay of no more than 100 ms."

"The probability that a packet has a delay > 10 ms is < 0.01."

Contract between the user and operator

An operator can guarantee performance bounds for a connection only by reserving some network resources

Either on-the-fly, during the call establishment phase of a connection, or in advance

But the amount of resources reserved depends on the connection's traffic intensity!

Guaranteed-service connections must agree to limit their usage

The relationship between the network operator and the user can be viewed as a legal contract

The user agrees that its traffic will remain within certain bounds

The operator guarantees that the network will meet the connection's performance requirements

Bandwidth

Specified as minimum bandwidth measured over a pre-specified interval

E.g. at least 5 Mbps over intervals of 1 sec

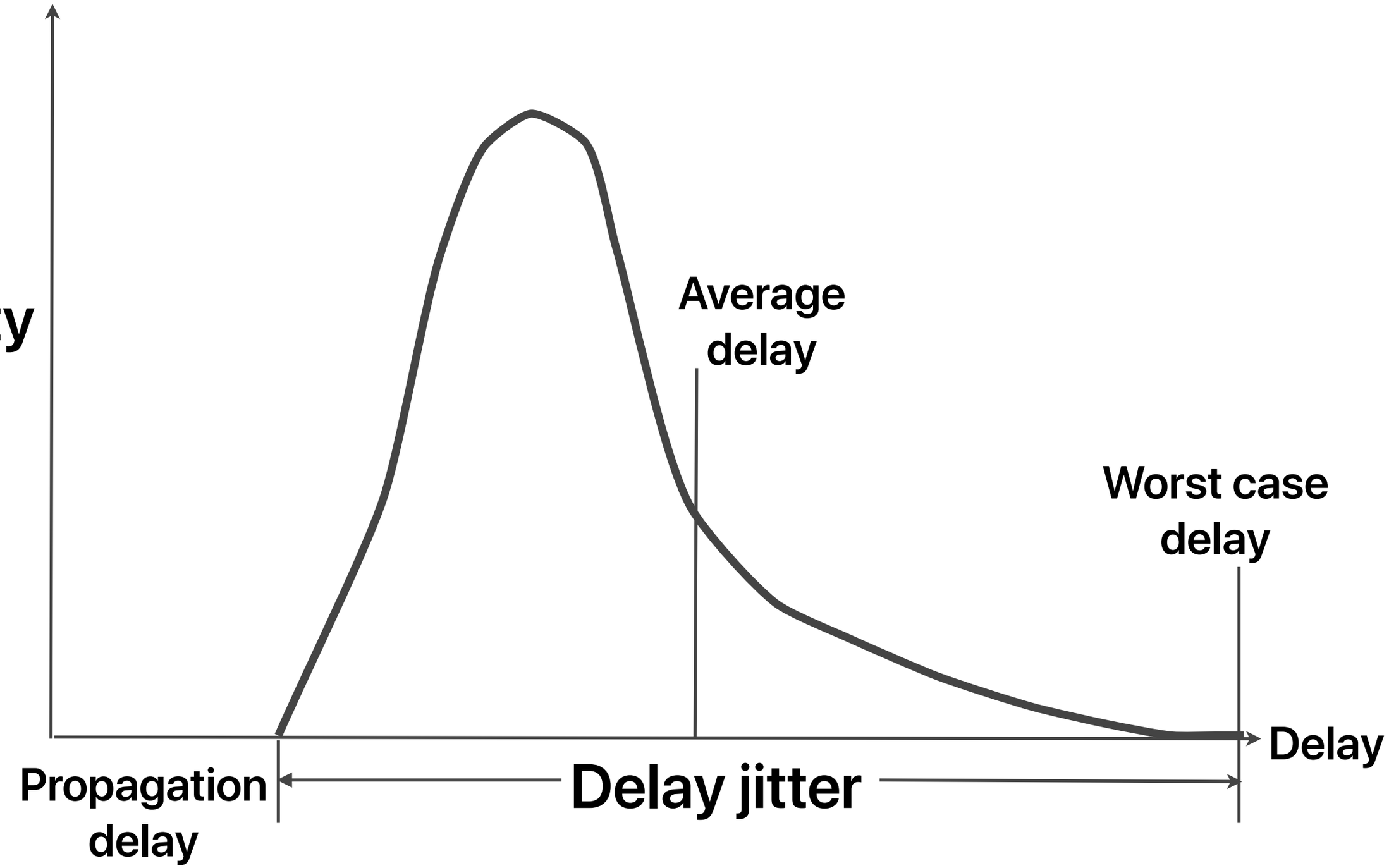
Meaningless without an interval!

Can be a bound on average (sustained) rate or peak rate

Peak is measured over a small interval

Delay and delay jitter

Bound on some parameter of the delay distribution curve



Requirement 4. Ease of Admission Control

Admission control needed to provide Quality of Service

Overloaded resource cannot guarantee performance

Choice of scheduling discipline affects ease of admission control algorithm

Outline

What is scheduling?

Why do we need it?

Requirements of a scheduling discipline

Fundamental choices

Scheduling best effort connections

Scheduling guaranteed-service connections

Fundamental choices

Number of priority levels

Work-conserving vs. non-work-conserving

Degree of aggregation

Service order within a priority level or aggregation class

Choice 1. Number of priority levels

Packet is served from a given priority level only if no packets exist at higher levels (multilevel priority with exhaustive service)

Highest level gets lowest delay

At least three are desirable:

- high priority level: urgent messages (e.g. network control)
- medium priority level: guaranteed service traffic
- low priority level: best-effort traffic

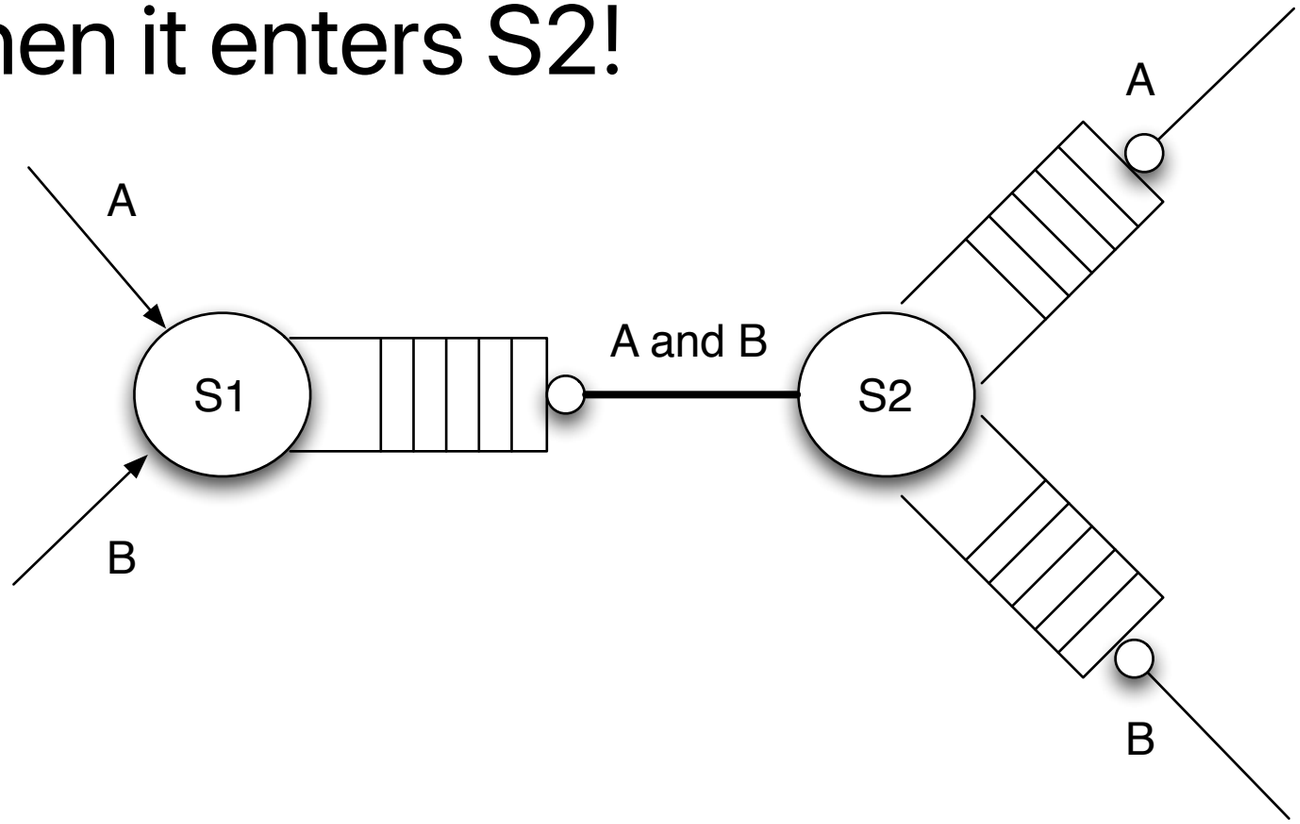
Watch out for starvation!

Choice 2. Work-conserving vs. non-work-conserving

Work-conserving discipline is never idle when packets await service

Why bother with non-work-conserving?

In this "dumbbell" topology, if B is bursty, and S1 is work-conserving, even if A is smooth when it enters S1, it can be bursty when it enters S2!



Choice 2. Work-conserving vs. non-work-

Key conceptual idea: delay a packet till it is eligible

Reduces delay jitter → smaller buffers in the network

How to choose eligibility time?

rate-jitter regulator: bounds maximum outgoing rate

delay-jitter regulator: compensates for variable delay at previous hop

Do we need non-work-conserving disciplines?

Can remove delay-jitter at an endpoint instead

but also reduces size of switch buffers

Increases mean delay

not a problem for playback applications

Wastes bandwidth

can serve best-effort packets instead

Always punishes a misbehaving source

can't have it both ways

Bottom line: not too bad, implementation cost may be the biggest problem

Choice 3. Degree of aggregation

More aggregation

less state

cheaper

smaller VLSI

less to advertise

But: less individualization

Solution

aggregate to a **class**, members of a class have same performance requirement

no protection within class

Choice 4. Service order within a priority level or class

In order of arrival (FCFS) or in order of a per-packet service tag

Service tags: can arbitrarily reorder queue

- can achieve allocations that are close to max-min fair
- both protection and delay bounds are also possible
- but need to sort queue, which can be expensive

FCFS

- bandwidth "hogs" win (no protection)
- rewards greedy behaviour (peer-to-peer networks)
- no guarantee on delays

Summary of fundamental choices

Each feature comes at some cost of implementation

For a small LAN switch —

- Traffic loads are likely to be much smaller than the available capacity
- Service queues are small
- Users are cooperative
- A single FCFS scheduler, or a two-priority scheduler, should be sufficient

For a heavily-loaded wide-area public switch —

- Possibly non-cooperative users
- Protection is desirable
- Multiple priority levels, non-work-conserving at some priority levels, aggregation within some priority levels, and non-FCFS service order

Keshav Chapter 9.1, 9.2, 9.3, 9.4, 9.5.1, 13.3.4