Congestion Control

Hui Chen [a]

[a]CUNY Brooklyn College

November 16, 2023

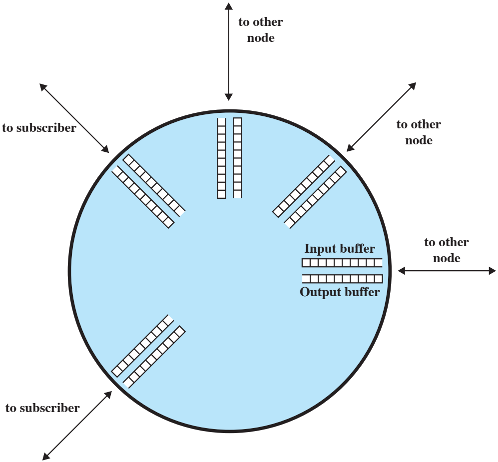
Outline

Congestion

- Congestion occurs when the number of packets being transmitted through a packeted switched network begins to approach the packet-handling capability of the network
- What happens when the network is congested?

Outline

Input and Output Queues at a Forwarding Node[1]

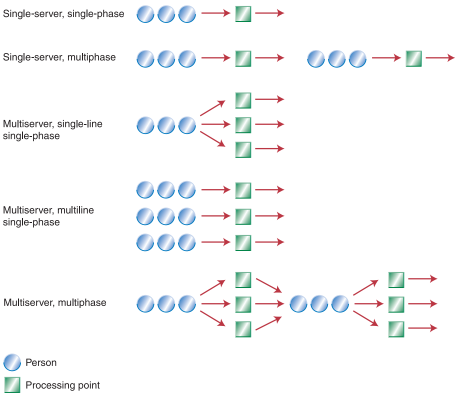


[1]William Stallings. *Data and Computer Communications*. 10th. USA: Prentice Hall Press, 2013. ISBN: 0133506487.

Queueing Perspective

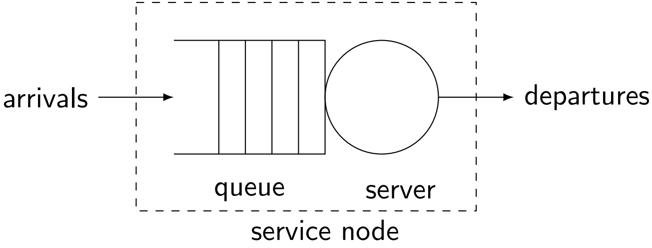
Model a packet switched network as a network of *queues* (i.e., service nodes with queues)

Service Nodes with Queues



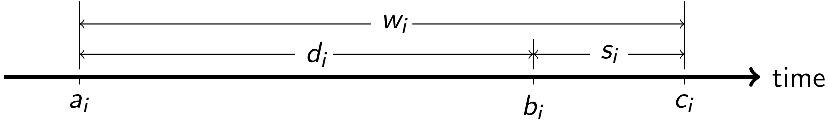
From: "Dear Mona, Which Is The Fastest Check-Out Lane At The Grocery Store?" by Mona Chalabi, originally appeared in Operations Management, 5th Edition by "R. Dan Reid, Nada R. Sanders", 2013

Single-Server Queue (SSQ)



Specifying Single-Server Queue (SSQ)[2]

- Arrival time: a_i
- Delay in queue (queuing delay): d_i
- Time that service begins: $b_i = a_i + d_i$
- Service time: s_i
- Wait in the node (total delay): $w_i = d_i + s_i$
- Departure (completion) time: $c_i = a_i + w_i$



[2] Lawrence M Leemis and Stephen Keith Park. *Discrete-event simulation: A first course*. Pearson Prentice Hall Upper Saddle River, NJ, 2006.

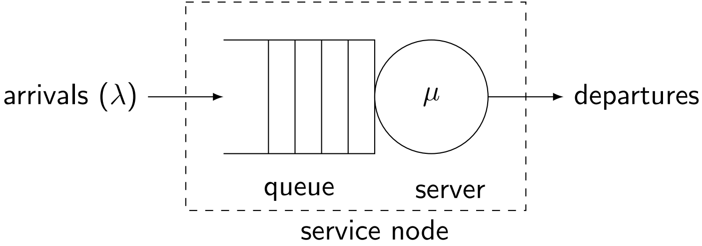
Computational Model for SSQ

Algorithm 1: Computing delays of SSQ

```
Function ssq()
    c_0 ← 0.0
    i ← 0
    while more jobs to process do
        i ← i + 1
        a_i ← GetArrival()
        if a_i < c_{i-1} then
            d_i ← c_{i-1} − a_i
        else
            d_i ← 0.0
        s_i ← GetService()
        c_i ← a_i + d_i + s_i
    n ← i
    return c,d
```

M/M/1 Queue

Arrivals follow a Poisson process and service times have an exponential distribution (both are Markovian or memoryless). Because arrivals follow a Poisson process, inter-arrival times also have an exponential distribution.



Computational Model for M/M/1 Queue: `Exponential`

Algorithm 2: Drawing from an Exponential distribution

Input : m, the mean, $m = \frac{1}{\mu}$
Output: t, a sample
Function `Exponential`(m)
  $t \leftarrow m \ln(1. - \texttt{Uniform}(0., 1.))$
  return t

From: https://www.explainxkcd.com/wiki/index.php/1374:_Urn

Computational Model for M/M/1 Queue: `GetArrival`

Algorithm 3: Computing arrival times

Input : λ, the rate of arrival
Input : a_p, the previous arrival time
Output: a, the arrival time
Function `GetArrival`(a_p, λ)
  $t \leftarrow$ `Exponential`$(1/\lambda)$
  $a \leftarrow a_p + t$
  return a

Computational Model for M/M/1 Queue: `GetService`

Algorithm 4: Computing service times

Input : m, the mean of the service times, $m = \frac{1}{\mu}$

Output: s, the service time

Function `GetService`(m)

```
    s ← Exponential(m)
    return s
```

Some Job (Packet)-Averaged Statistics of SSQ (1 of 2)

- Average inter-arrival time (e.g., per packet)

$$\bar{r} = \frac{1}{n}\sum_{i=1}^{n} r_i = \frac{a_n}{n} \tag{1}$$

For M/M/1 queue,

$$\bar{r} = \frac{1}{\lambda} \tag{2}$$

- Average service time

$$\bar{s} = \frac{1}{n}\sum_{i=1}^{n} s_i \tag{3}$$

For M/M/1 queue,

$$\bar{s} = \frac{1}{\mu} \tag{4}$$

Some Job (Packet)-averaged Statistics of SSQ (2 of 2)

- Average delay

$$\bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_i \tag{5}$$

- Average wait

$$\bar{w} = \frac{1}{n} \sum_{i=1}^{n} w_i \tag{6}$$

- Since $w_i = d_i + s_i$,

$$\bar{w} = \frac{1}{n} \sum_{i=1}^{n} d_i + s_i = \frac{1}{n} \sum_{i=1}^{n} d_i + \frac{1}{n} \sum_{i=1}^{n} s_i = \bar{d} + \bar{s} \tag{7}$$

Traffic Intensity (Normalized Load)

Traffic intensity is the ratio of average service time to average inter-arrival time

$$\rho = \frac{\bar{s}}{\bar{r}} \tag{8}$$

For M/M/1 queue,

$$\rho = \frac{\frac{1}{\mu}}{\frac{1}{\lambda}} = \frac{\lambda}{\mu} \tag{9}$$

Time-Averaged Statistics: Server Utilization

- Server utilization (the time-averaged packets served, i.e., normalized throughput)

$$\bar{x} = \frac{\sum_{i=1}^{n} s_i}{c_n} = \frac{\frac{1}{n}\sum_{i=1}^{n} s_i}{\frac{1}{n}c_n} = \frac{n}{c_n}\bar{s} \tag{10}$$

- When the node is saturated (i.e., the server is always busy), $c_n/n = \bar{s}$,i.e.,

$$\bar{x} = 1 \tag{11}$$

- When the node is not saturated (i.e., sometimes idle) but at the steady state ($n \to \infty$),

$$\lim_{n\to\infty} \frac{c_n}{n} = \lim_{n\to\infty} \frac{a_{n-1} + r_n}{n} = \lim_{n\to\infty} \frac{a_{n-1}}{n} + \lim_{n\to\infty} \frac{r_n}{n} = \frac{1}{\lambda} \tag{12}$$

$$\bar{x} = \frac{n}{c_n}\bar{s} = \frac{\lambda}{\mu} = \rho \tag{13}$$

Time-Average Statistics: Waiting Time and Delay

- Time-averaged delay in queue

$$\bar{q} = \frac{\sum_{i=1}^{n} d_i}{c_n} = \frac{\frac{1}{n}\sum_{i=1}^{n} d_i}{\frac{1}{n} c_n} = \frac{n}{c_n}\bar{d} \tag{14}$$

- Time-averaged waiting time

$$\bar{l} = \frac{\sum_{i=1}^{n} w_i}{c_n} = \frac{\frac{1}{n}\sum_{i=1}^{n} w_i}{\frac{1}{n} c_n} = \frac{n}{c_n}\bar{w} \tag{15}$$

- Since $w_i = d_i + s_i$,

$$\bar{l} = \bar{q} + \bar{x} \tag{16}$$

Time-Average Statistics: Unsaturated Node

When the node isn't saturated ($\rho < 1$) but at the steady state ($n \to \infty$),

$$\lim_{n\to\infty} \frac{c_n}{n} = \frac{1}{\lambda} \tag{17}$$

- Time-averaged delay in queue

$$\bar{q} = \lim_{n\to\infty} \frac{n}{c_n}\bar{d} = \lambda\bar{d} \tag{18}$$

- Time-averaged waiting time

$$\bar{l} = \lim_{n\to\infty} \frac{n}{c_n}\bar{w} = \lambda\bar{w} \tag{19}$$

- Since $w_i = d_i + s_i$,

$$\bar{w} = \bar{d} + \bar{s} = \bar{d} + \frac{1}{mu} \tag{20}$$

- Also,

$$\bar{l} = \bar{q} + \bar{x} = \bar{q} + \rho \tag{21}$$

Time-Average Statistics: Unsaturated Node

Rewrite these four equations in the following matrix form,

$$\begin{bmatrix} \mu & 0 & -\mu & 0 \\ 0 & 1 & 0 & -1 \\ \lambda & -1 & 0 & 0 \\ 0 & 0 & \lambda & -1 \end{bmatrix} \begin{bmatrix} \bar{w} \\ \bar{l} \\ \bar{d} \\ \bar{q} \end{bmatrix} = \begin{bmatrix} 1 \\ \rho \\ 0 \\ 0 \end{bmatrix} \tag{22}$$

- Can we solve it (λ and μ are knowns)? However, there are only 3 independent rows.
- Cannot, but we only need to determine one of the four statistics.

Some Results of M/M/1 (Steady State, Unsaturated)

With some efforts (we skip the derivation here, or via simulation),

$$\bar{r} = \frac{1}{\lambda} \tag{23}$$

$$\bar{s} = \frac{1}{\mu} \tag{24}$$

$$\bar{x} = \frac{\lambda}{\mu} \tag{25}$$

$$\bar{l} = \frac{\rho}{1-\rho} \tag{26}$$

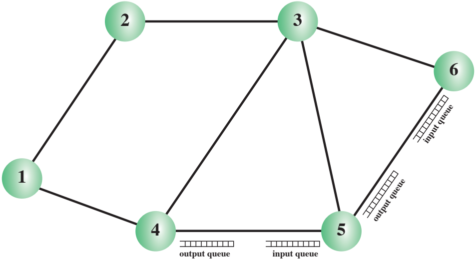
$$\bar{w} = \frac{1}{\mu - \lambda} \tag{27}$$

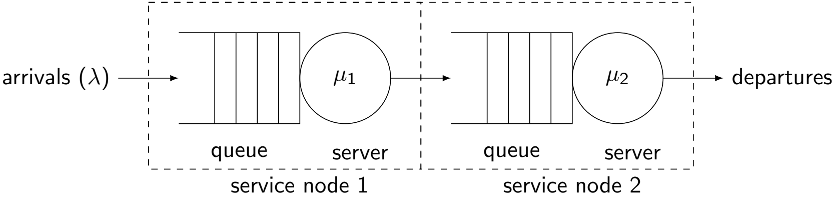
$$\bar{d} = \frac{q}{\lambda} = \frac{\rho}{\mu - \lambda} \tag{28}$$

$$\bar{q} = \bar{l} - \rho = \frac{\rho^2}{1-\rho} \tag{29}$$

Outline

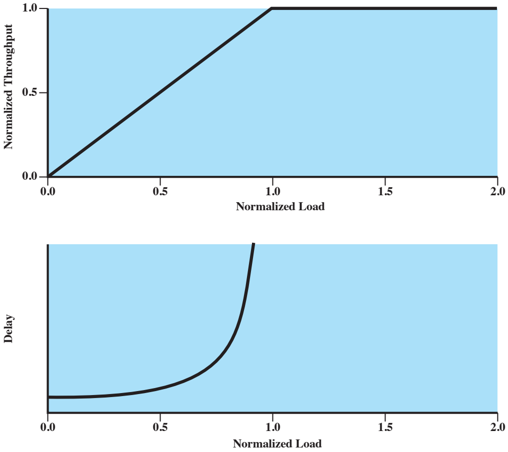
Interaction of Queues in a Data Network[3]





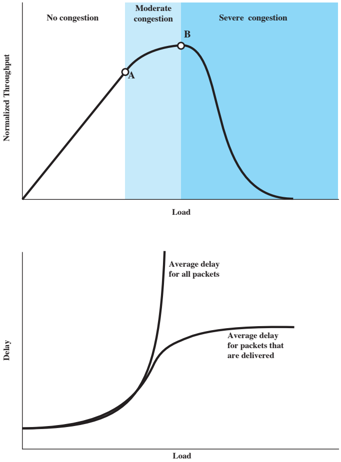
[3] William Stallings. *Data and Computer Communications*. 10th. USA: Prentice Hall Press, 2013. ISBN: 0133506487.

Ideal Network Utilization[4]



[4] William Stallings. *Data and Computer Communications*. 10th. USA: Prentice Hall Press, 2013. ISBN: 0133506487.

Effects of Congestion[5]



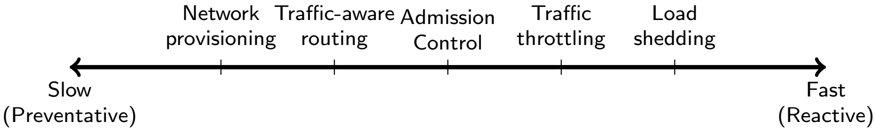
[5] William Stallings. *Data and Computer Communications*. 10th. USA: Prentice Hall Press, 2013. ISBN: 0133506487.

Control Collapse and Need for Congestion Control

- Goodput. The rate at which useful packets are delivered by the network.
- Congestion collapse. A prolonged period during which goodput dropped precipitously (i.e., by more than a factor of 100) due to congestion in the network.
  - Starting in 1986, the growing popularity of the early Internet led to the first occurrence of congestion collapse.
  - Research on congestion control followed.

Outline

General Approaches



- Network provisioning, e.g., building network with sufficient bandwidth, upgrading routers and links.
- Traffic-aware routing, e.g., changing routes to shift traffic away from heavily used paths (by changing the shortest path weights), splitting traffic across multiple paths.
- Admission control, e.g., decreasing the load by refusing new connection.
- Traffic throttling, i.e.,telling the senders to throttle back their transmissions and slow down.
- Load shedding, i.e., discarding packets that it cannot deliver.

Traffic-aware Routing

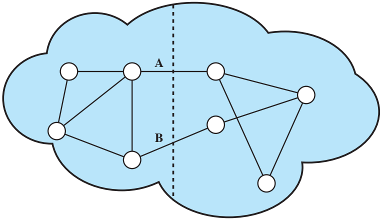


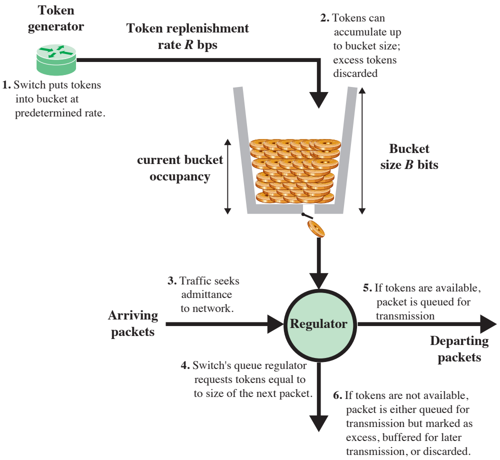
Figure 19.7 Packet-Switching Network Subject to Oscillations

- Taking traffic load into consideration when computing link costs
  - Example. Let the link cost to be a function of the (fixed) link bandwidth, propagation delay, measured load or average queuing delay.
    - Problem. Oscillation, leading to erratic routing and many potential problems.
- Solution. Multipath routing. Traffic engineering (adjustments are made outside the routing protocol by slowly changing its inputs.)

Admission Control

- To decrease load by refusing connections, but which connection to refuse?
- Need to be able to characterize a potential incoming traffic
  - Data rate isn't sufficient, e.g., Web traffic (bursty) vs. streaming video (continuous)
  - A common traffic descriptor is the leaky bucket or token bucket
  - Based on the descriptor, determine whether a congestion would happen if admitted, and then admit or refuse.

Traffic Shapping and Policying via Token Bucket[6]



[6] William Stallings. *Data and Computer Communications*. 10th. USA: Prentice Hall Press, 2013. ISBN: 0133506487.

Traffic Throttling

- Based on "periodical" feedback, ask the senders to throttle back their transmissions and slow down.
- Signals?
  - Averages of utilization do not directly account for the burstiness of most network traffic; while the queueing delay inside routers directly captures any congestion experienced by packets.
  - Measuring queuing delay via an Exponentially Weighted Moving Average (EWMA).

$$d_{k+1} = \alpha d_k + (1 - \alpha)s \tag{30}$$

- Feedback mechanisms?

Feedback Mechanisms

Examples of feedback mechanisms.

- Choke packets. Tell the sender directly by sending to it a packet containing informatino about the congested packet.
- Explicit Congestion Notification (ECN).
  1. Tag any packet it forwards (by setting a bit in the packet's header) to signal that it is experiencing congestion.
  2. The destination will then echo any marks back to the sender as an ECN in its next reply packet.
- Hop-by-Hop Backpressure.
  - High speeds network or over long distances, e.g., how many packets a host in San Francisco may have transmitted before a choke packet originated in New York reach it?
  - "Choke" at every router along the path.
  - Provide quick relief at the point of congestion, at the price of using up more buffers upstream, i.e., applying a backpressure

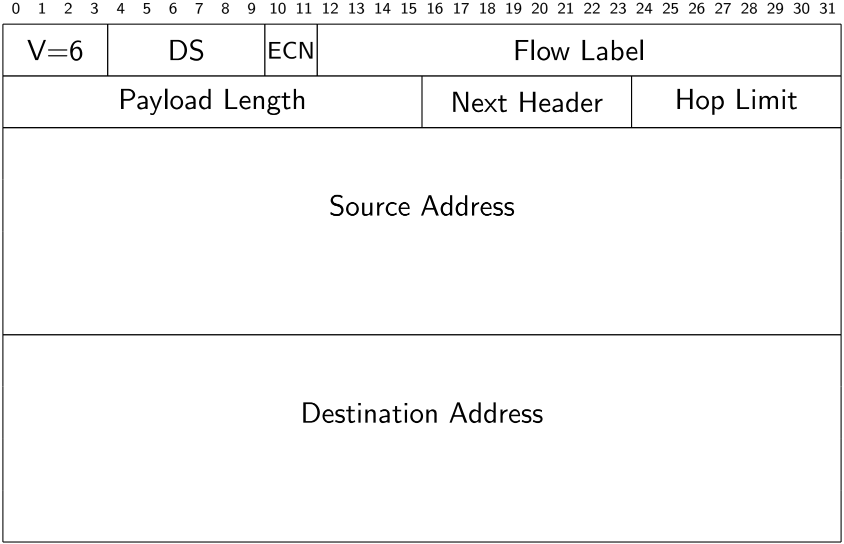
IPv4 Packet Header

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td></tr>
<tr><td colspan="4">V=4</td><td colspan="4">IHL</td><td colspan="6">DS</td><td colspan="2">ECN[7]</td><td colspan="16">Total Length</td></tr>
<tr><td colspan="16">Identification</td><td colspan="3">Flags</td><td colspan="13">Fragment Offset</td></tr>
<tr><td colspan="8">Time To Live</td><td colspan="8">Protocol[8]</td><td colspan="16">Header Checksum</td></tr>
<tr><td colspan="32">Source Address</td></tr>
<tr><td colspan="32">Destination Address</td></tr>
<tr><td colspan="32">Options + Paddings
(32 bits × n, n = 0, 1, 2, . . .)
. . .</td></tr>
</table>

[7] See RFC 3168

[8] https://www.iana.org/assignments/protocol-numbers

IPv6 Packet Header



Load Shedding

Which packets to drop?

- Varying from application to application, e.g.,
  - Old packets are more valuable for a file transfer application
  - New packets are more valuable for a real-time control application
- Two simple policies, dubbed names "Wine" and "Milk" (people prefers fresh milk but old wine)
  - Wine. Drop new packets
  - Milk. Drop old packets
- Some protocols/applications are complex, requiring more intelligent solution.
  - Packets carrying routing information.
  - In MPEG compressed video, I-frames can be independently uncompressed, but not the others.
- Require sends' cooperation, or some incentives to encourage senders' cooperation.

Random Early Detection (RED) for Load Shedding

- Detecting and dealing congestion earlier is a better approach.
- Explicit Consigestion Signal may not be available (Internet hosts may not yet get congestion signals from routers in the form of ECN)
- What's the most reliable indication of congestion that hosts get from the network is packet loss (implicit signal)
  - TCP treats packet loss as the signal of a congestion
- Exploit this to proactively battle congestion, i.e., have routers drop packets early before congestion, e.g., RED.
  - Routers maintain a running average of their queue lengths.
  - When the average queue length on some link exceeds a threshold, drop at random a small fraction of the packets
- ECN or RED. If ECN available, ECN; or, RED.

Router-Centric versus Host-Centric

Both the routers inside the network and the hosts at the edges of the network participate in resource allocation, where should we place the majority of the burden?

- (Router-centric) Routers make decisions.
  - When packets are forwarded?
  - Which packets are to be dropped
  - How many packets hosts are allowed to send?
- (Host-centric). Hosts make decisions.
  - Observe the network conditions (e.g., how many packets they are successfully getting through the network);
  - Adjust their behavior accordingly.

Reservation-Based versus Feedback-Based

Whether they use reservations or feedback for resource allocation. In a

- (Reservation-based) Some entity (e.g., the end host) asks the network for a certain amount of capacity to be allocated for a flow.
  - If the request cannot be satisfied at some router, then the router rejects the reservation.
  - Routers then allocates enough resources (buffers and/or percentage of the link's bandwidth) to satisfy this request.
- (Feedback-based approach) End hosts begin sending data without first reserving any capacity and make adjusts according to to the feedback received.
  - Explicit feedback versus implicit feedback
  - Explicit feedback, e.g., a congested router sends a "slow down" message to the host)
  - Implicit feedback, e.g., the end host adjusts its sending rate according to the externally observable behavior of the network, such as packet losses.

Credit Based versus Rate Based

Both flow-control and resource allocation mechanisms need a way to express, to the sender, how much data it is allowed to transmit.

- (Window based or Credit based) About how much buffer space the receiver has, and it limits how much data the sender can transmit.
  - For instance, in TCP, the receiver advertises a window to the sender.
- (Rate Based) About how many bits per second the receiver or network is able to absorb.
  - Some applications prefer rate-based approach, e.g., a multimedia applications, which tend to generate data at some average rate and which need at least some minimum throughput to be useful.

Service Model and Resource Allocation

Best-effort service model versus QoS-based service model

- A best-effort service model usually implies that feedback is being used
  - Does not allow users to reserve network capacity.
  - Burdon for congestion control falls to the end hosts (with some assistance from the routers.)
  - In practice, such networks use window-based information.
  - Example. The Internet.
- A QoS-based service model probably implies some form of reservation.
  - Need significant router involvement (e.g.., different queues for different level of resource reservation)
  - Often express express such reservations in terms of rate (windows are only indirectly related to how much bandwidth a user needs from the network.)

Outline

Effective Resource Allocation

Congestion control is essentially a resource allocation problem.

- An efficient allocation of bandwidth across transport entities will use all of the network capacity that is available; however, ...
- Recall the relationship among traffic load, delay, and goodput.
- Kleinrock's power

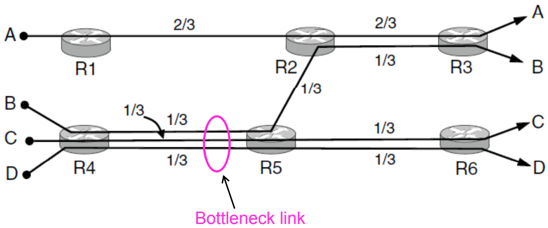
$$\text{power} = \frac{\text{load}}{\text{delay}} \tag{31}$$

Fair Resource Allocation

At least, we should give bandwidth to all flows (without starvation). What exactly is fairness?

Max-min Fairness

- An allocation is max-min fair if the bandwidth given to one flow cannot be increased without decreasing the bandwidth given to another flow with an allocation that is no larger.
- Max-min fairness gives equal shares of bottleneck.
- Example.[9]



[9] Andrew S. Tanenbaum and David Wetherall. *Computer Networks*. 5th ed. Boston: Prentice Hall, 2011. ISBN: 978-0-13-212695-3. URL: https://www.safaribooksonline.com/library/view/computer-networks-fifth/9780133485936/.

Jain's Fairness Index

- Given a set of flow throughputs $(x_1, x_2, \ldots, x_n)$ (measured in consistent units such as bits/second), the following function assigns a fairness index to the flows,

$$f(x_1, x2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2} \tag{32}$$

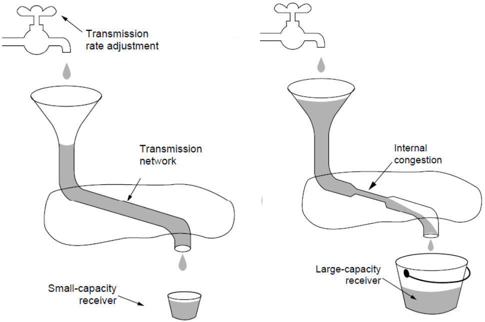
Convergence

How quickly does a congestion control algorithm converge to a fair and efficient allocation of bandwidth.

Controlling Resource Allocation

How do we regulate the sending rates to obtain a desirable bandwidth allocation directly or indirectly.

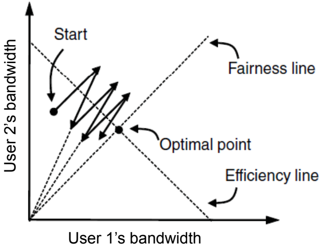
- Flow control. What if there is insufficient buffering at the receiver.
- Congestion control. What if there is insufficient capacity in the network (insufficient buffering at the routers).



Control Law

The way in which the rates are increased or decreased is given.

- Feedback may be explicit or implicit, and may be precise or imprecise.
- Binary congestion feedback and AIMD (Additive Increase Multiplicative Decrease), e.g.,[10]



[10] Andrew S. Tanenbaum and David Wetherall. *Computer Networks*. 5th ed. Boston: Prentice Hall, 2011. ISBN: 978-0-13-212695-3. URL: https://www.safaribooksonline.com/library/view/computer-networks-fifth/9780133485936/.

Outline

TCP Congestion Control

TCP uses AIMD with lossy signal to control congestion

- Implemented as a congestion window (`cwnd`) for the number of segments allowed in the network
- Uses several mechanisms that work together
  - ACK clock. Using congestion window (`cwnd`) to smooth out packet bursts
  - Slow-start. Doubling `cwnd` each RTT to rapidly increase send rate to reach roughly the right level
  - Additive increase. Increasing `cwnd` by 1 packet each RTT to slowly increase send rate to probe at about the right level
  - Fast retransmit. Resending lost packet after 3 duplicate ACKs; sending new packet for each new ACK, to recover from a lost packet without stopping ACK clock

Sending Rate and Congestsion Control Window

Congestion window controls the sending rate

- Rate is cwnd/RTT; window can stop sender quickly
- ACK clock (regular receipt of ACKs) paces traffic and smoothes out sender bursts

RTT Estimation and Retranmission Timer Management

First, need to estimate round-trip time (RTT, denoted as d below)

- Simple average.

$$\bar{d}_{k+1} = \frac{1}{K} \sum_{i=1}^{K+1} d_i = \frac{K}{K+1} \bar{d}_k + \frac{1}{K} d_{k+1} \tag{33}$$

- Exponential average.

$$\bar{d}_{k+1} = \alpha \bar{d}_k + (1 - \alpha) d_{K+1} \tag{34}$$

Then, estimate the retransmission time-out value (T) as,

$$T_{K+1} = \bar{d}_{K+1} + \Delta \tag{35}$$

or,

$$T_{K+1} = \min(U, \max(L, \beta \bar{d}_{K+1})) \tag{36}$$

where U and L are two pre-chosen upper and lower bounds where the time-out value is in, β is a constant.

Improving RTT Estimation and Retransmission Timer

- Jacobson's algorithm
- Karn's algorithm

Window Management

With slow start, we have the following constraint (flow control + congestion control)

$$\text{awnd} = \min(\text{credit}, \text{cwnd}) \tag{37}$$

where awnd is the number of segments outstanding without being acknowledged, credit TCP flow-control credit allocation, and cwnd congestion window.

Slow Start and Fast Recovery

- Slow start followed by additive increase (TCP Tahoe)
- Fast recovery(TCP Reno)