Lecture 11 — April 24, 2002

Lecturer: Rahul Hariharan *Scribe: Ben Leong*

11.1 Congestion Control

The effects of congestion can be seen in Figure 11.1. As shown in the figure, the throughput of a connection increases as the offered load increases until a point when the queues in the system start to fill up and the increase in throughput levels off. When the offered load becomes too large and the buffers are totally filled up, packets begin to get dropped by the system and throughput starts to decrease.

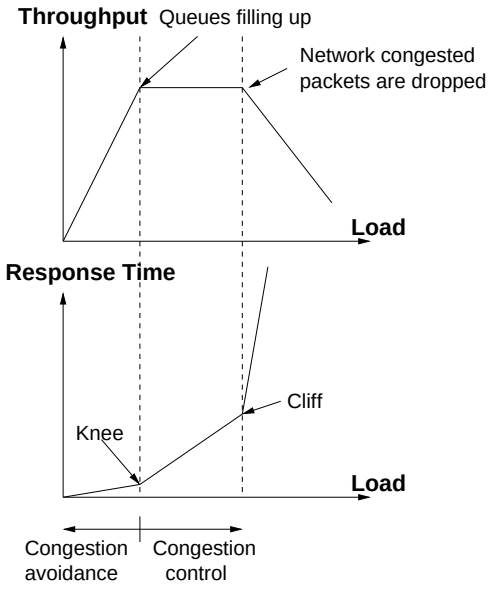

Figure 11.1. Effects of Congestion

The effect of congestion on response time is also intuitively clear. When the system is not congestion, the response time is short. As congestion increases and packets get queued, the response time gets longer. Finally, when packets start to get dropped, the response time starts to shoot up sharply. The three regions of operation described are divided by two points, known as the *knee* and *cliff* respectively. Algorithms which seek to operate the system at the knee are known as congestion avoidance algorithms, while those that operate between the knee and the cliff are known as congestion control algorithms.

11.2 Basic Model

We will model the network is a simple queueing system consisting of n flows and flow $i, 1 \leq i \leq n$, has throughput, $x_i(t)$. X_{goal} is the total capacity of the network.

In order to quantify the "goodness" of a congestion control algorithm, we shall introduce the following metrics:

- Efficiency
- Fairness
- Distributedness
- Convergence

11.2.1 Efficiency

For an algorithm to be efficient, the total throughput should be close to the total capacity of the network, i.e.

$$\sum_n x_i(t) \approx X_{goal}$$

11.2.2 Fairness

There are many measures of fairness that have been proposed. The general idea is that flows belonging to the same class should have appoximately equal shares of the available bandwidth. The measure of fairness that we shall adopt for our discussions is

$$F(t) = \frac{[\sum_n x_i(t)]^2}{n \sum_n x_i^2(t)}$$

Clearly, $\frac{1}{n} \leq F \leq 1$.

11.2.3 Distributedness

The distributedness criterion is simply that all the flows must be able to make independent decision without resort to some centralized coordinator.

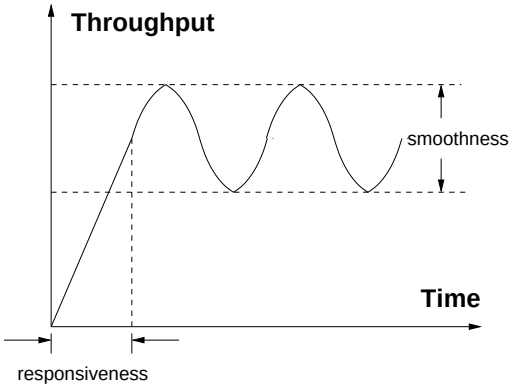

Figure 11.2. Convergence.

11.2.4 Convergence

Convergence measures how long it takes for the system to start from any point and converge to the ideal state. As shown in Figure 11.2, we can measure the responsiveness and the smoothness. Responsiveness measures how long it takes for the system to reach the ideal state, while smoothness measures the magnitude of the oscillations about the ideal state even after the system reaches a dynamic equilibrium.

11.3 Conditions for Convergence to Efficiency and Fairness

Next, we shall attempt to derive the necessary conditions for an algorithm to converge to a fair and efficient outcome[1]. We will restrict ourselves to the class of linear controls. In our model, the network will provide us with binary feedback, $y(t)$, which is interpreted by users as follows:

$$y(t) = \begin{cases} 0 & \rightarrow \quad \text{Increase load} \\ 1 & \rightarrow \quad \text{Decrease load} \end{cases}$$

In response to feedback, the offered load at time $t+1$, $x_i(t+1)$ is determined as a function of $x_i(t)$ and $y(t)$ as follows:

$$x_i(t+1) = \begin{cases} a_I + b_I x_i(t) & \text{if} \quad y(t) = 0 \\ a_D + b_D x_i(t) & \text{if} \quad y(t) = 1 \end{cases}$$

In a simple 2-user case, the resource allocation can be represented as a point in a 2-dimensional space as shown in Figure 11.3. In this figure, the horizontal axis represents the allocation to user 1 and the vertical axis represents allocation to user 2. All allocations for which $x_1 + x_2 = X_{goal}$ are efficient allocations. This corresponds to the straight line marked "efficiency line." All allocations for which $x_1 = x_2$ are fair allocations. This corresponds to the straight line marked "Fairness = 1." The two lines intersect at the the point $(\frac{X_{goal}}{2}, \frac{X_{goal}}{2})$, which is the optimal point. The goal of a control scheme is to bring the system to this point.

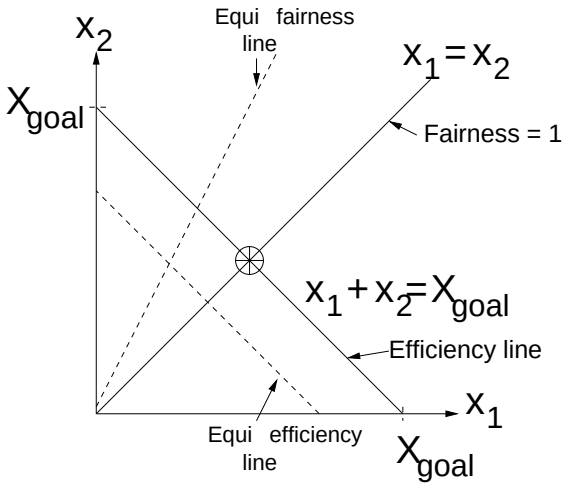

Figure 11.3. Vector Representation for a two-user system

11.3.1 Efficiency

In order for the system to satify the efficiency criterion, the following conditions must hold:

$$\sum_{i=1}^{n} x_i(t+1) > \sum_{i=1}^{n} x_i(t) \quad \text{if} \quad y(t) = 0$$

$$\sum_{i=1}^{n} x_i(t+1) < \sum_{i=1}^{n} x_i(t) \quad \text{if} \quad y(t) = 1$$

11.3.2 Fairness

In order for the system to converge to fairness, the following condition must be satisfied:

$$F(t+1) > F(t)$$

$$\frac{[\sum_n x_i(t+1)]^2}{n\sum_n x_i^2(t+1)} > \frac{[\sum_n x_i(t)]^2}{n\sum_n x_i^2(t)}$$

With some algebra, we can re-write $F(t+1)$ as follows:

$$F(t+1) = F(t) + (1 - F(t))\left(1 - \frac{[\sum_n x_i(t)]^2}{\sum_n (c x_i^2(t))}\right), \text{ where } c = \frac{a}{b}$$

We note that fairness improves if $c > 0$ and fairness remains constant if $c = 0$. Hence, either

$$\frac{a_I}{b_I} \geq 0 \quad \& \quad \frac{a_D}{b_D} > 0$$

or

$$\frac{a_I}{b_I} > 0 \quad \& \quad \frac{a_D}{b_D} \geq 0$$

In other words, a_I and b_I must be of the same sign and a_D and b_D must also be of the same sign. In addition, at most one of a_I and b_I can be zero and at most one of a_I and a_D can be zero.

11.3.3 Distributedness

In order for the system to satify the distributedness criterion, the following conditions must hold:

$$\begin{cases} x_i(t+1) > x_i(t) \ \forall \ i \ \text{ if } \ y(t) = 0 \\ x_i(t+1) \leq x_i(t) \ \forall \ i \ \text{ if } \ y(t) = 1 \end{cases}$$

This implies:

$$\begin{cases} a_I + (b_I - 1)x_i(t) > 0 \ \text{ if } \ y(t) = 0 \\ a_D + (b_D - 1)x_i(t) \leq 0 \ \text{ if } \ y(t) = 1 \end{cases}$$

The first condition implies that $a_1 > 0$ and $b_1 \geq 1$ while the second condition implies that $a_D \leq 0$ and $b_I < 1$. Coupling these results with the results in the preceding Section, we obtain:

$$\begin{cases} a_D = 0 \\ a_I > 0 \\ 0 \leq b_D < 1 \\ b_I \geq 1 \end{cases}$$

Finally, we observe that fairness is an increasing function of $c_I = \frac{a_I}{b_I}$ and $c_D = \frac{a_D}{b_D} = 0$. Hence to maximize c_I, we set $b_I = 1$.

11.3.4 Convergence

From

$$x_i(t+1) = a_I + b_I x_i(t)$$

Next, we sum across all nodes to obtain:

$$X(t+1) = na_I + b_I X(t)$$

from which, we derive

$$\begin{aligned} \frac{dX}{dt} &= na_I + (b_I - 1)X(t) \\ \Rightarrow \int_0^t dt &= \int_{X_0}^{X_{goal}} \frac{dX}{na_I + (b_I - 1)X}, \ \text{if } b \neq 1 \\ t &= \frac{1}{b_I - 1} \log \left[\frac{na_I + (b_I - 1)X_{goal}}{na_I + (b_I - 1)X_0} \right], \ \text{if } b \neq 1 \end{aligned}$$

If $b_I = 1$, then the solution is $t = \frac{X_{goal} - X_0}{na_I}$. This expression gives us a bound on the responsiveness of the algorithem, i.e. the time it takes to reach X_{goal}.

11.3.5 Concluding Remarks

We therefore conclude that the following parameters are optimal:

$$\begin{cases} a_D = 0 \\ a_I > 0 \\ 0 \leq b_D < 1 \\ b_I = 1 \end{cases}$$

This results in the following linear controls:

$$x_i(t+1) = \begin{cases} a_I + x_i(t) & \text{if} \quad y(t) = 0 \\ b_D x_i(t) & \text{if} \quad y(t) = 1 \end{cases}$$

This is also known as the Additive Increase Multiplicative Decrease (AIMD) algorithm.

We conclude by noting that in our model above, we have not taken into account some factors. In particular,

- We have assumed that all senders are synchronized.
- We ignored the possibility of delayed feedback.
- We have ignored the presence of the other flows (> 2) in the network.
- The feedback available has been assumed to be a simple binary feedback. We may want to ask if we can do better if more information is available.

11.4 Application - Transmission Control Protocol (TCP)

The Transmission Control Protocol[2, 3, 4, 5] (TCP) is a self-clocking, sliding window protocol , i.e. it keeps track of outstanding data packets in flight as a window, that provides handling for both timeouts and retransmissions. There are also other rate-based protocols that instead control the rate at which data is sent by the sender. The window size determines the number of bytes of data that can be sent before an acknowledgement from the receiver is necessary.

In its *Congestion Avoidance* mode, TCP uses the AIMD algorithm to adapt to congestion. The successful transmission of a packet is indicated by an Acknowlegement (ACK) from the receiver, while the loss of a packet is detected by time-out. TCP attempts to estimate the roundtrip time (RTT) and the variance of the RTT in order to determine an appropriate timeout threshold. Where w is the window size, the window size is increased by one (approximately) for every RTT:

$$\begin{cases} w \leftarrow w + \frac{1}{w} & \text{if} \quad \text{ACK is received} \\ w \leftarrow \frac{w}{2} & \text{if} \quad \text{transmission timeout} \end{cases}$$

During the initial *Slow Start* phase, the window is adjusted more aggressively to allow TCP to saturate the capacity of the link as soon as possible. In particular,the window size is doubled every RTT:

$$w \leftarrow w + 1$$

Based on the above algorithm, the behavior of the TCP window in steady state is of a shape given in Figure 11.4. From the figure, we find that the throughput of TCP is given by $\frac{3}{4}\frac{w^2}{R_{TT}}$.

Suppose we assume a perfect model where only one packet is lost every time the congestion window is cut in half. Then, the packet loss probability, p, is given by:

$$p = \frac{1}{\frac{w^2}{8} + \frac{w^2}{4}} = \frac{8}{3w^2}$$

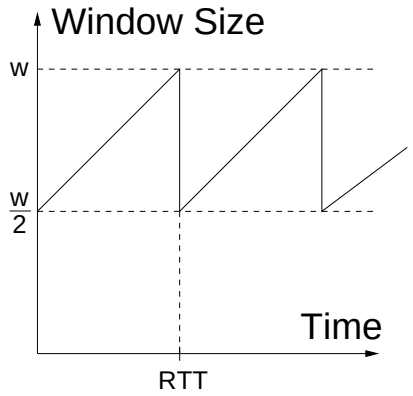

Figure 11.4. TCP in Steady State.

This implies $w = \sqrt{\frac{8}{3p}}$ and allows us to express the throughput in terms of the packet loss probability:

$$\text{Throughput} = \frac{3}{4R_{TT}}\sqrt{\frac{8}{3p}} = \frac{1}{R_{TT}}\sqrt{\frac{3}{2p}}$$

Again, we need to recognize that this is a very simplistic model because packet drops are usually not independent and also, this model ignores the effects of dynamic behavior due to changes in the number of flows.

11.5 Application - eXplicit Control Protocol (XCP)

The eXplicit Control Protocol[6] (XCP) is a novel reliable protocol that attempts to separate efficiency from fairness explicit feedback solve bandwidth inefficiency by having routers add additional congestion information to the router packets. Based on the spare bandwidth available, the protocol attempts to alleviate the unfairness caused by large delays by compensating flows for long RTTs. The routers deal only with aggregate traffic flow.

Bibliography

[1] D. Chiu and R. Jain. Analysis of the increase/decrease algorithms for congestion avoidance in computer networks. *Journal of Computer Networks and ISDN*, 17(1):1–14, June 1989.

[2] W. R. Stevens. *TCP/IP Illustrated, VOLUME 1; The Protocols*. Addison Wesley, Reading, 1994.

[3] J. Postel. Transmission control protocol, RFC 793, September 1981.

[4] V. Jacobson, R. Braden, and D. Borman. TCP extensions for high performance, RFC 1323, May 1992.

[5] M. Allman, V. Paxson, and W. Richard Stevens. TCP congestion control, RFC 2581, April 1999.

[6] Dina Katabi, Mark Handley, and Charlie Rohrs. Internet congestion control for future high bandwidth-delay product environments. In *Proc. ACM SIGCOMM '2002 conference on Applications, technologies, architectures, and protocols for computer communication.*, 2002.

[7] J. Mahdavi and S. Floyd. TCP-friendly unicast rate-based flow control, January 1997.