TCP Congestion Control: AIMD

The last part of TCP that we're going to study is congestion control.

We'll cover the following

- Requirements of a Congestion Control Algorithm
- TCP Congestion Control Algorithms
  - Additive Increase Multiplicative Decrease
- Quick Quiz!

We’ve already looked at what congestion control is in a previous lesson. Let’s get into some TCP-specific congestion control algorithms now!

But first:

Requirements of a Congestion Control Algorithm

1. The congestion control scheme **must avoid congestion**. In practice, this means that the algorithm should ensure that the bandwidth allocated to a certain host at any given time does not exceed the bandwidth of the bottleneck link.

2. The congestion control scheme **must be efficient**. The congestion control scheme should ensure that the available bandwidth is efficiently used. Namely that the bandwidth allocated to a certain host at any given time doesn’t fall too far below the bandwidth of the bottleneck link.

3. The congestion control scheme **should be fair**. Most congestion schemes aim at achieving max-min fairness, which we had a lesson dedicated to.

TCP Congestion Control Algorithms

To implement most TCP congestion control algorithms, a TCP host must be able to control its **transmission rate**. In order to do so, it can constrain its

sending window. Recall that after transmitting data equal to the window size,

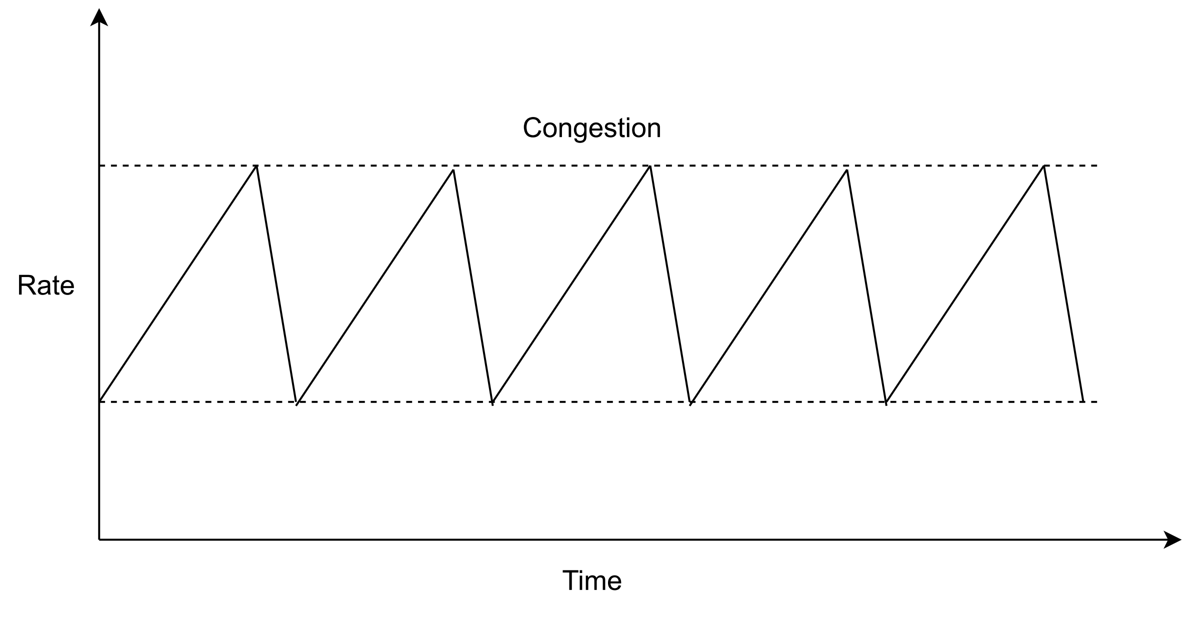
the sender must pause and wait at least one RTT for the *ACK* before it can transmit more data. Thus the maximum data rate is: $\frac{window}{rtt}$ where $window$ is the maximum between the host's sending window and the window advertised by the receiver.

Hence, TCP's congestion control scheme can restrict sending windows based on a **congestion window**. The current value of the congestion window is stored in the TCB of each TCP connection. The value of the window that can be used by the sender is $min(congestion\ window,\ receiving\ window,\ sending\ window)$.

Additive Increase Multiplicative Decrease

The **Additive Increase Multiplicative Decrease** algorithm *decreases* the transmission rate of a host when congestion has been detected in the network. Congestion is detected when acknowledgments for packets do not arrive before the transmission timer times out. Furthermore, it *increases* their transmission rate when the network is *not* congested. Hence, the rate allocated to each host fluctuates with time, depending on the feedback received from the network.

The figure below illustrates the evolution of the transmission rates allocated to a host in a network.



How the transmission rate of a host changes with AIMD. Notice the 'saw tooth' pattern.

The **Additive Increase** part of the TCP congestion control increments the congestion window by *MSS* bytes every round-trip time. In the TCP literature, this phase is often called the *congestion avoidance* phase. Once congestion is detected, the **Multiplicative Decrease** part of the TCP congestion control reacts by multiplying the current value of the congestion window with a number greater than 0 and less than 1.

However, since the increase in the window is so slow, the TCP connection may have to wait for many round-trip times before being able to efficiently use the available bandwidth. To avoid this, the TCP congestion control scheme includes the **slow-start** algorithm.

Quick Quiz!

1 Which of the following is not a requirement of a congestion control algorithm?

- ○ A) Must avoid congestion
- ○ B) Must be fair
- ○ C) Must be efficient
- ○ D) Must be accurate

COMPLETED 0% 1 of 2

Let's have a look at the slow start algorithm in the next lesson.