View congestion control problem as a 2-dimensional system: \((Q(t), \lambda(t))\) evolve according to

\[
Q(t + 1) = [Q(t) + \lambda(t) - \gamma(t)]^+
\]

\[
\lambda(t + 1) = f(\lambda(t), Q(t), \gamma, Q^*)
\]

\(\rightarrow\) \(f(\cdot)\) is some control law
Working around in \((Q(t), \lambda(t))\)-space:

\[ \rightarrow \text{ phase space} \]
Convergent trajectory:

\[
\rightarrow \text{ asymptotically stable & optimal}
\]
Divergent trajectory:

\[ \rightarrow \text{ unstable} \]
Stable (but not asymptotically so) trajectory:

\[\rightarrow \text{ limit cycle}\]
Which case arises depends on the shape of control law $f(\cdot)$

For example:

- Methods A and C: divergent
- Method B: stable (but not asymptotically) $\rightarrow$ TCP
- Method D: asymptotically stable & optimal $\rightarrow$ “optimal control”

Why does Method D work?

Analysis of 2-dimensional system $(Q(t), \lambda(t))$:

\[
\begin{align*}
Q(t + 1) &= [Q(t) + \lambda(t) - \gamma]^+ \\
\lambda(t + 1) &= f(\lambda(t), Q(t), \gamma, Q^*) \\
&= \lambda(t) + \varepsilon(Q^* - Q(t)) - \beta(\lambda(t) - \gamma)
\end{align*}
\]
First, convert into continuous form (it is easier to manipulate):

\[
\frac{dQ(t)}{dt} = \lambda(t) - \gamma \\
\frac{d\lambda(t)}{dt} = \varepsilon(Q^* - Q(t)) - \beta(\lambda(t) - \gamma)
\]

Note: at optimal operating point \((Q^*, \gamma)\),

\[
\frac{dQ(t)}{dt} = 0 \quad \text{and} \quad \frac{d\lambda(t)}{dt} = 0
\]

\[\rightarrow (Q^*, \gamma) \text{ is rest point or equilibrium}\]

Express in vector form:

\[
\frac{d}{dt} \begin{pmatrix} Q(t) \\ \lambda(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\varepsilon & -\beta \end{pmatrix} \begin{pmatrix} Q(t) \\ \lambda(t) \end{pmatrix} + \begin{pmatrix} -\gamma \\ \varepsilon Q^* + \beta \gamma \end{pmatrix}
\]

\[\rightarrow \text{ second-order linear differential equation}\]
Fact: stability depends on eigenvalues of
\[
\begin{pmatrix}
0 & 1 \\
-\varepsilon & -\beta
\end{pmatrix}
\]

In particular, system is asymptotically stable if the real parts of the eigenvalues are strictly negative.

Eigenvalues are obtained from characteristic equation
\[
\det \begin{pmatrix}
-\nu & 1 \\
-\varepsilon & -\beta - \nu
\end{pmatrix} = \nu(\beta + \nu) + \varepsilon
\]
\[
= \nu^2 + \beta \nu + \varepsilon = 0
\]

This yields:
\[
\nu = \frac{-\beta}{2} \pm \frac{\sqrt{\beta^2 - 4\varepsilon}}{2}
\]

If \( \varepsilon > 0 \), then \( \text{Re}(\nu) < 0 \). If, in addition, \( 0 < \varepsilon < \beta/2 \) then the eigenvalues are real.
Check: without the $-\beta(\lambda(t) - \gamma)$ term in the control law, the characteristic equation becomes

$$\nu^2 + \varepsilon = 0$$

Thus, $\nu = \pm \sqrt{\varepsilon}$ and the real part of $\nu$ is zero, violating the asymptotic stability condition.

$\rightarrow$ mathematical requirement

$\rightarrow$ intuitively: damping effect
TCP congestion control

Recall:

\[ \text{EffectiveWindow} = \text{MaxWindow} - (\text{LastByteSent} - \text{LastByteAcked}) \]

where

\[ \text{MaxWindow} = \min\{\text{AdvertisedWindow}, \text{CongestionWindow}\} \]

Key question: how to set \text{CongestionWindow} which, in turn, affects ARQ’s sending rate?

\[ \rightarrow \text{linear increase/exponential decrease} \]

\[ \rightarrow \text{AIMD} \]
TCP congestion control components:

(i) Congestion avoidance

\[ \rightarrow \text{ linear increase/exponential decrease} \]

\[ \rightarrow \text{ additive increase/exponential decrease (AIMD)} \]

As in Method B, increase CongestionWindow linearly, but decrease exponentially

Upon receiving ACK:

\[ \text{CongestionWindow} \leftarrow \text{CongestionWindow} + 1 \]

Upon timeout:

\[ \text{CongestionWindow} \leftarrow \text{CongestionWindow} / 2 \]

An implementation detail:

\[ \rightarrow \text{ is it correct?} \]
“Linear increase” time diagram:

→ actually results in exponential increase
What we want:

→ increase by 1 every window
Thus, linear increase update:

\[
\text{CongestionWindow} \leftarrow \text{CongestionWindow} + \left( \frac{1}{\text{CongestionWindow}} \right)
\]

Since unit of TCP segment is MSS:

\[
\text{CongestionWindow} \leftarrow \text{CongestionWindow} + \text{MSS} \cdot \left( \frac{\text{MSS}}{\text{CongestionWindow}} \right)
\]

Lastly, upon timeout and exponential backoff, remember halved \text{CongestionWindow} value in \text{SlowStartThreshold}:

\[
\text{SlowStartThreshold} \leftarrow \frac{\text{CongestionWindow}}{2}
\]
(ii) Slow Start

Initialize/reset $\textit{CongestionWindow}$ to 1

Perform exponential increase

$$\textit{CongestionWindow} \leftarrow \textit{CongestionWindow} + \textit{MSS}$$

• until timeout, if at start of connection
  → rapidly probe for available bandwidth

• until $\textit{CongestionWindow}$ reaches $\textit{SlowStartThreshold}$, if following Congestion Avoidance phase
  → rapidly climb to safe level

→ “slow” start is misnomer

→ exponential is superfast

→ slow compared to sending all at once
Basic dynamics: procedural

CongestionWindow evolution:
(iii) Exponential timer backoff in Karn/Partridge’s scheme

\[ \text{TimeOut} \leftarrow 2 \times \text{TimeOut} \quad \text{if retransmit} \]

(iv) Fast Retransmit

Upon receiving three duplicate ACKs:

- Transmit next expected segment
  \[ \rightarrow \text{segment indicated by ACK value} \]
- Perform exponential backoff and commence Slow Start
  \[ \rightarrow \text{three duplicate ACKs: likely segment is lost} \]
  \[ \rightarrow \text{react before timeout occurs} \]

TCP Tahoe: features (i)-(iv)
(v) Fast Recovery

Upon Fast Retransmit:

- Skip Slow Start and commence Congestion Avoidance
  → dup ACKs: likely spurious loss
- Insert “inflationary” phase just before Congestion Avoidance

Inflationary phase:

- \( \text{SlowStartThreshold} \leftarrow \text{CongestionWindow} / 2 \)
- \( \text{CongestionWindow} \leftarrow \text{SlowStartThreshold} + 3 \)
- On each additional duplicate ACK, increment \( \text{CongestionWindow} \)
- On first non-dup ACK, commence Congestion Avoidance with
  \( \text{CongestionWindow} \leftarrow \text{SlowStartThreshold} \)
TCP Reno: features (i)-(v)

→ pre-dominant form

Many more versions of TCP:

→ NewReno w/ SACK, w/o SACK, Vegas, etc.
→ wireless, ECN, multiple time scale
→ mixed verdict; pros/cons
Given sawtooth behavior of TCP’s linear increase/exponential backoff:

Why use exponential backoff and not Method D?

- For multimedia streaming (e.g., pseudo real-time), AIMD (Method B) is not appropriate
  → use Method D

- For unimodal case—throughput decreases when system load is excessive—story is more complicated
  → asymmetry in control law may be needed for stability
Congestion control for unimodal curve:

\[ \gamma \]

\[ Q \]

\[ Q^* = 100, \ \varepsilon = 0.1, \ \beta = 0.5 \]

- Case 1: monotone
  \[ \gamma = 10 \text{ for all } Q \]

- Case 2: unimodal
  \[ \gamma = 10 \text{ for } Q \leq Q^* \text{, but } \gamma = 10 - 2(Q^* - Q) \text{ for } Q > Q^* \] with minimum throughput \( \gamma \geq 1 \)
Load-throughput curve:

\[ Q \gamma \]

\[ Q^* \]

\[ \gamma \text{ max} \]

\[ \gamma \text{ min} \]

Compare time evolution of Method for Case 1 and 2

\[ \rightarrow \text{ from (asymptotically) stable to unstable} \]

\[ \rightarrow \text{ worst case: congestion collapse} \]
Monotone load-throughput curve:
Unimodal load-throughput curve:
Intuition: impact of unimodal load-throughput curve

\[ \Delta Q = \lambda \Delta \]

\[ \Delta \lambda = \gamma \]

\[ \Delta Q \]

\[ \Delta \lambda \]

\[ \lambda(t) \]

\[ Q(t) \]

\[ \gamma \]

\[ Q^* \]

\[ \rightarrow \text{ divergence problem} \]

\[ \rightarrow \text{ backward pull not strong enough} \]
What is needed: extra downward boost when $Q(t) > Q^*$

$\rightarrow$ must catch up with decreasing $\gamma(Q)$

$\rightarrow$ danger zone: $Q(t) > Q^*$
Thus, in danger zone $Q(t) > Q^*$:

- Both $\lambda(t) \downarrow$ and $\gamma(Q) \downarrow$
- If $\lambda(t)$ doesn’t decrease fast enough (relative to $\gamma(Q)$)
  \[ \rightarrow \lambda(t) - \gamma(Q) > 0 \]
  \[ \rightarrow Q(t) \text{ keeps on increasing} \]
  \[ \rightarrow \text{further aggravates situation} \]
  \[ \rightarrow \text{one-way street to congestion collapse} \]

Role of exponential decrease:

\[ \rightarrow \text{exponential is superfast} \]

In general, need not be exponential

\[ \rightarrow \text{depends on shape of } \gamma(Q) \]
\[ \rightarrow \text{e.g., Method D with different parameters} \]
$\beta$ is increased from 0.5 to 0.63:
System behavior for unimodal load-throughput curve:

\[
\frac{dQ(t)}{dt} = \lambda(t) - \gamma(Q) \\
\frac{d\lambda(t)}{dt} = \varepsilon(Q^* - Q(t)) - \beta(\lambda(t) - \gamma(Q))
\]

\[\rightarrow\] nonlinear differential equation due to \(\gamma(Q)\)

Stability analysis using local method:

\[\rightarrow\] stable manifold theorem

\[\rightarrow\] relates stability of nonlinear system to linearized one

\[\rightarrow\] to do or not to do?

Result: Method D is stable if

\[-\beta \frac{d\gamma(Q)}{dQ} > 1\]