TCP congestion control

Recall:

$$\texttt{EffectiveWindow} = \texttt{MaxWindow} - (\texttt{LastByteSent} - \texttt{LastByteAcked})$$

where

$$\texttt{MaxWindow} = \min\{\texttt{AdvertisedWindow}, \texttt{CongestionWindow}\}$$

Key question: how to set `CongestionWindow` which, in turn, affects ARQ's sending rate?

$\longrightarrow$ linear increase/exponential decrease

$\longrightarrow$ AIMD

TCP congestion control components:

(i) Congestion avoidance

$\longrightarrow$ linear increase/exponential decrease

$\longrightarrow$ additive increase/exponential decrease (AIMD)

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

Upon receiving ACK:

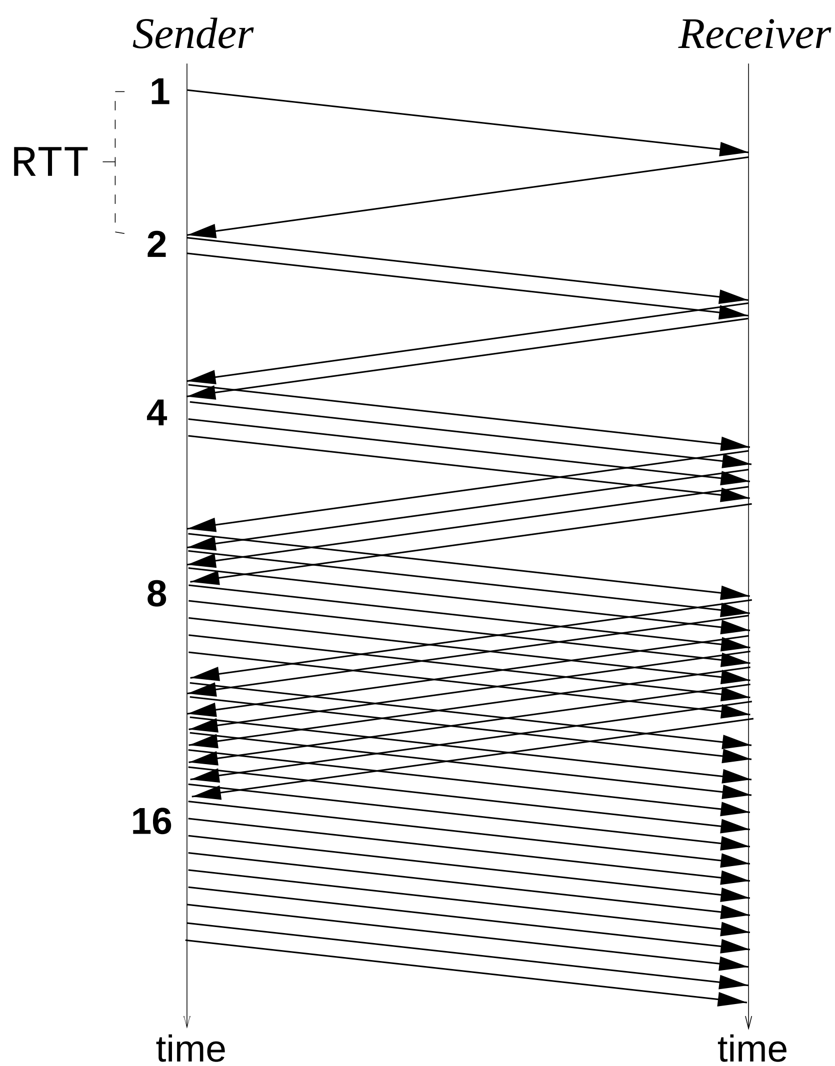
$$\texttt{CongestionWindow} \leftarrow \texttt{CongestionWindow} + 1$$

Upon timeout:

$$\texttt{CongestionWindow} \leftarrow \texttt{CongestionWindow} \,/\, 2$$

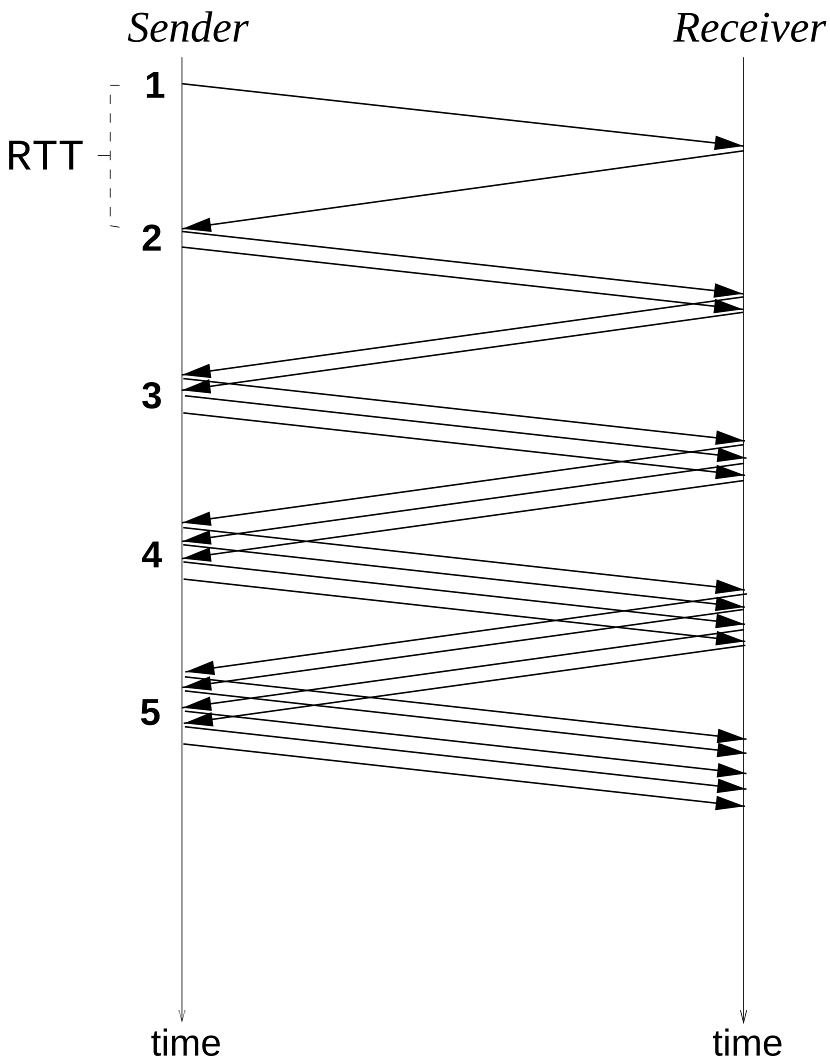
But is it correct...

"Linear increase" time diagram:



⟶ results in exponential increase

What we want:



$\longrightarrow$ increase by 1 every window

Thus, linear increase update:

$$\texttt{CongestionWindow} \leftarrow \texttt{CongestionWindow} + (1\,/\,\texttt{CongestionWindow})$$

Upon timeout and exponential backoff,

$$\texttt{SlowStartThreshold} \leftarrow \texttt{CongestionWindow}\,/\,2$$

(ii) Slow Start

Reset `CongestionWindow` to 1

Perform exponential increase

$$\texttt{CongestionWindow} \leftarrow \texttt{CongestionWindow} + 1$$

- Until timeout at start of connection

  $\rightarrow$ rapidly probe for available bandwidth

- Until `CongestionWindow` hits `SlowStartThreshold` following Congestion Avoidance

  $\rightarrow$ rapidly climb to safe level

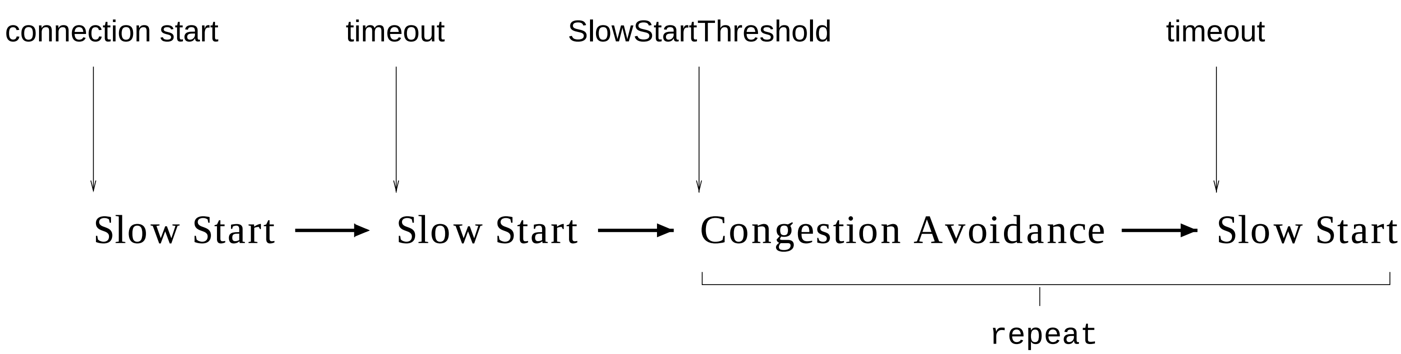
$\longrightarrow$ "slow" is a misnomer

$\longrightarrow$ exponential increase is super-fast

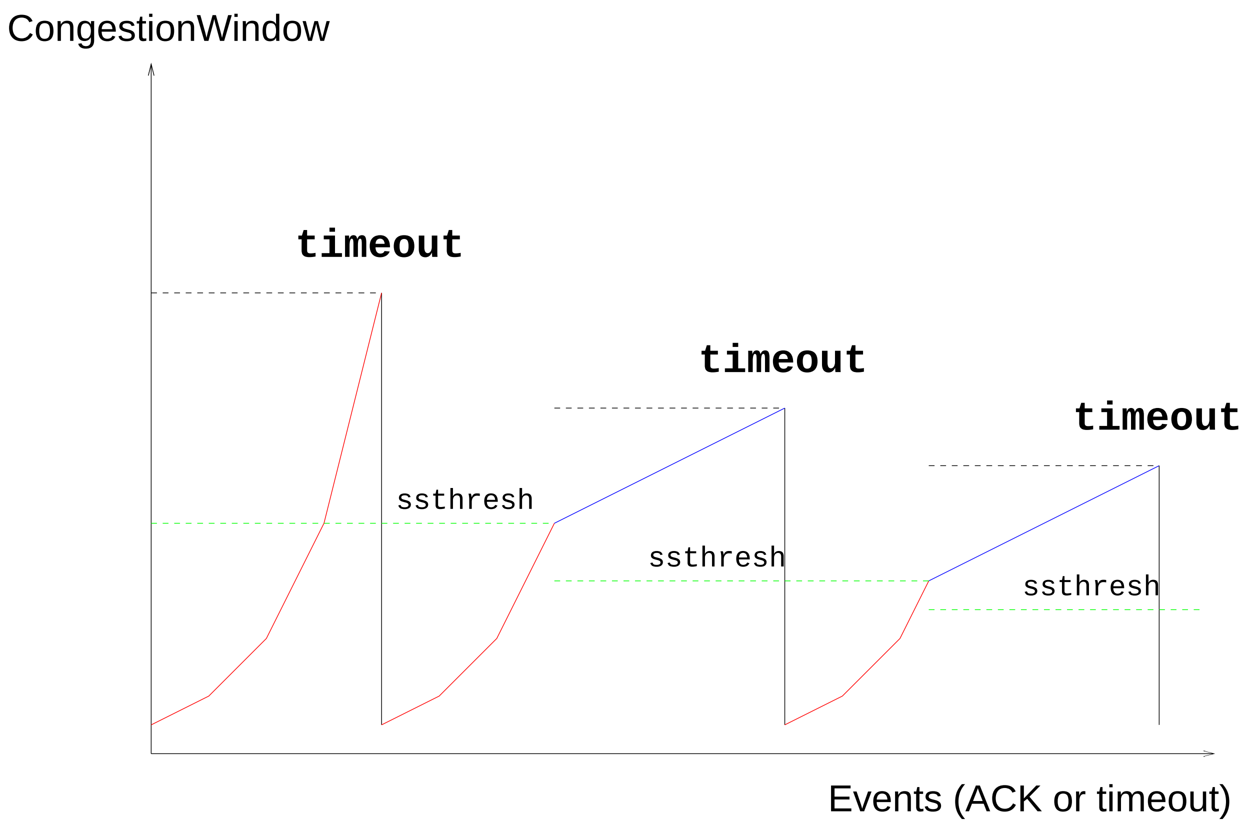
Basic dynamics:

⟶ after connection set-up

⟶ before connection tear-down



`CongestionWindow` evolution:



(iii) Exponential timer backoff

$$\texttt{TimeOut} \leftarrow 2 \cdot \texttt{TimeOut} \qquad \text{if retransmit}$$

(iv) Fast Retransmit

Upon receiving three duplicate ACKs:

- Transmit next expected segment

  → segment indicated by ACK value

- Perform exponential backoff and commence Slow Start

⟶ three duplicate ACKs: likely segment is lost

⟶ react before timeout occurs

TCP Tahoe: features (i)-(iv)

(v) Fast Recovery

Upon Fast Retransmit:

- Skip Slow Start and commence Congestion Avoidance

  $\rightarrow$ dup ACKs: likely spurious loss

- Insert "inflationary" phase just before Congestion Avoidance

TCP Reno: features (i)-(v)

$\longrightarrow$ pre-dominant form

Many more versions of TCP:

$\longrightarrow$ NewReno w/ SACK, w/o SACK, Vegas, etc.

$\longrightarrow$ wireless, ECN, multiple time scale

$\longrightarrow$ 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

  $\longrightarrow$ use Method D

- For unimodal case—throughput decreases when system load is excessive—story is more complicated

  $\longrightarrow$ asymmetry in control law needed for stability

Congestion Control: Selfishness, Stability and Optimality

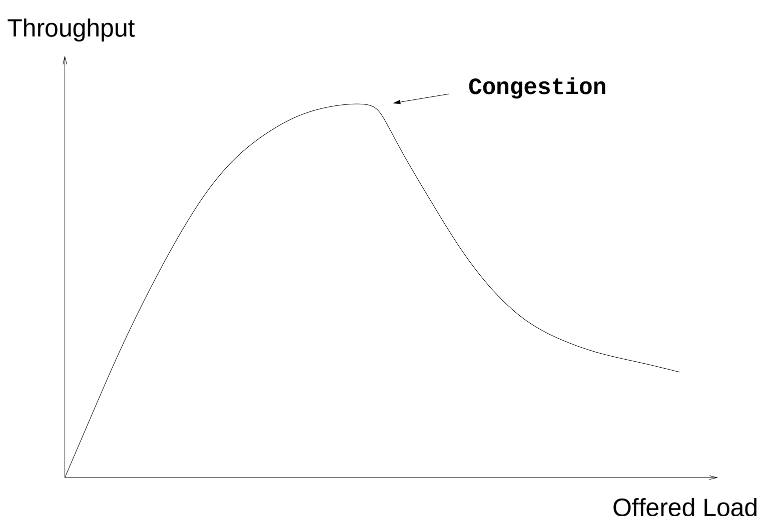
$\longrightarrow$ to be, or not to be, selfish . . .

$\longrightarrow$ noncooperative game theory

$\longrightarrow$ John von Neumann, John Nash, . . .

Congestion and "tragedy of commons":

$\longrightarrow$ Garrett Hardin, '68



- if everyone acts selfishly, no one wins

  $\rightarrow$ in fact, everyone loses

- can this be prevented?

Two-party congestion control setting:

⟶ Prisoner's Dilemma game

⟶ both cooperate (stay silent): 1 year each

⟶ both selfish (rat on the other): 5 years each

⟶ one cooperative/one selfish: 9 vs. 0 years

When cast as congestion control game:

Alice \ *Bob*	C	N
C	5, 5	1, 9
N	9, 1	3, 3

⟶ (a, b): throughput (Mbps) achieved by Alice/Bob

⟶ what may happen?

⟶ what do "rational" (w.r.t. selfishness) players do?

Outcome of game with cooperative players?

$\longrightarrow$ configuration (C,C) with payoff (5,5)

$\longrightarrow$ system optimal: $5 + 5 = 10$ (sum of payoffs)

$\longrightarrow$ note: (1,9) and (9,1) are also system optimal

$\longrightarrow$ also Pareto optimal

Def. (Pareto optimality): A system state or configuration is Pareto optimal if total system payoff cannot be improved without sacrificing one (or more) player's payoff.

$\longrightarrow$ improvement requires "sacrificial lamb"

$\longrightarrow$ welfare notion of overall goodness

$\longrightarrow$ note: system optimal $\Rightarrow$ Pareto optimal (trivial)

$\longrightarrow$ (5,5), (1,9), (9,1): Pareto optimal

$\longrightarrow$ (3,3): not Pareto optimal

Outcome of game with noncooperative (i.e., selfish) players?

$\longrightarrow$ (N,N) with payoff (3,3)

$\longrightarrow$ notion of stability: Nash equilibrium

Def. (Nash equilibrium): A configuration is a Nash equilibrium (NE) if no selfish player has an incentive to unilaterally change his/her action.

$\longrightarrow$ (N,N) with payoff (3,3) is NE

$\longrightarrow$ Alice, alone, changing N to C: (N,N) $\mapsto$ (C,N)

$\longrightarrow$ (C,N) has payoff (1,9): bad for Alice

$\longrightarrow$ Nash equilibrium is a rest point

$\longrightarrow$ i.e., stable fixed-point (equilibrium)

$\longrightarrow$ idea: due to John Nash

$\longrightarrow$ key contribution: dynamics under selfishness

Is congestion control game NE (3,3) system optimal?

- $\longrightarrow$ no: (1,9) and (9,1): total payoff 10 (vs. 6)
- $\longrightarrow$ in fact: system optimal (5,5) is better for both
- $\longrightarrow$ in general, NE need not be system optimal
- $\longrightarrow$ also NE need not be Pareto optimal

Puts a damper on Adam Smith's postulate:

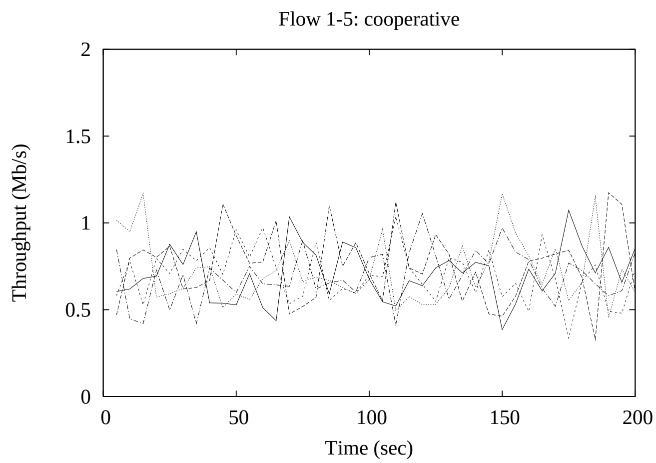
- $\longrightarrow$ wise, efficient "invisible hand" (i.e., "market")
- $\longrightarrow$ economy of selfish users self-organizes efficiently
- $\longrightarrow$ rarely true: Achilles' heel of "pure" capitalism

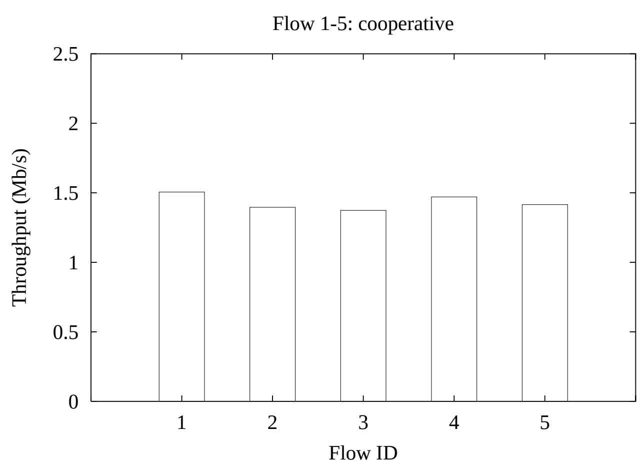
Karl Marx & communism?

- $\longrightarrow$ fantasy & wishful thinking
- $\longrightarrow$ evolution (hereto) has put premium on selfishness
- $\longrightarrow$ vulnerable to selfish elements
- $\longrightarrow$ Marx & Confucius: both more harm than good?

5 regular (cooperative) TCP flows:

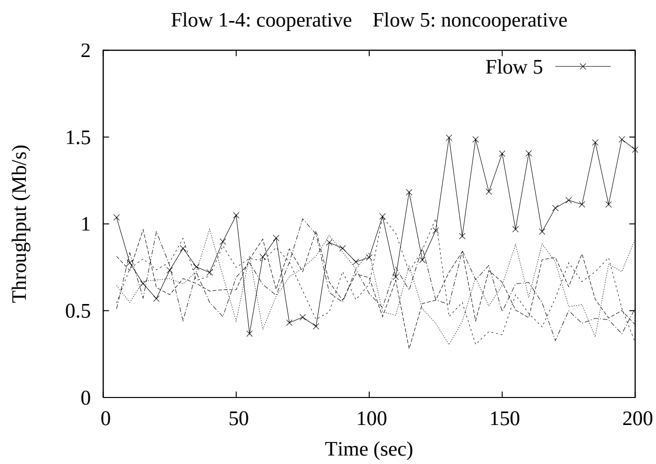
$\longrightarrow$ share 11 Mbps WLAN bottleneck link

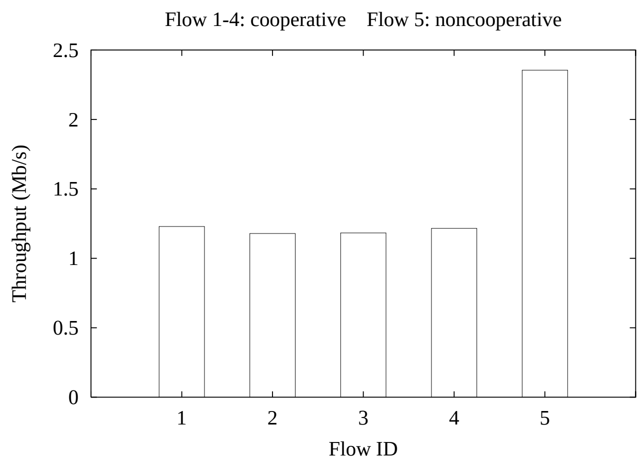




4 regular (cooperative) TCP flows and 1 noncooperative TCP flow:

$\longrightarrow$ same benchmark set-up





Remarks:

- NE, in general, are neither efficient nor fair

  $\longrightarrow$ $\exists$ special cases: strong rules/penalties

- in fact, in general, a Nash equilibrium need not exist

  $\longrightarrow$ system subject to oscillation

  $\longrightarrow$ circular "chain reaction"

- Nash's main result (game theory): finite noncooperative games with mixed strategies—choose action probabilistically—always possess equilibrium

  $\longrightarrow$ vs. pure strategy (more in tune with reality)

  $\longrightarrow$ pure strategy games: hard problem

- congestion pricing

  $\longrightarrow$ penalize those who congest: e.g., usage pricing

  $\longrightarrow$ in the States: flat pricing (dominant)

  $\longrightarrow$ not skimpy like the rest of the world!

- repeated/evolutionary games

  → e.g., iterated Prisoner's Dilemma

  → rob bank/get caught, again and again ...

  → what should the prisoners do then?

  → tit-for-tat, grim trigger: can be optimal

  → most relevant for greedy TCP