Congestion control, generalized

Ankit Singla
Learning goals

• TCP is not good enough
• How do we fix it?!
• Does packet loss really mean congestion?
Congestion

Problem: what rate to send at?
The right rate is important

Rate $> \text{available capacity} \ldots$

Packet loss and delay
Unfair capacity distribution

Rate $< \text{available capacity} \ldots$

Inefficient use of the network
Probing the capacity
Probing the capacity
Feedback control loop

1. Send packets
2. Wait
3. ACKs return in an RTT (or not)
4. Adjust sending rate

The network as a black box
Loss-reactive congestion control

All packets ACK-ed …

Increase rate!

Some packets presumed lost …

Decrease rate!
TCP (extremely briefly)

- **Additive increase**
- **Slow-start**
- **Multiplicative decrease**

Sending rate vs. Transmission number graph.
Problems with TCP

- Congestion window
- Transmission number
- Additive increase
- Slow-start
- Sending rate
- Multiplicative decrease

Graph showing sending rate against transmission number with markers for slow-start and additve increase followed by a multiplicative decrease.
Problems with TCP

- Additive increase
- Slow-start
- Reacting to losses
- Multiplicative decrease

Sending rate vs. Transmission number graph.
Long queues ⇒ queueing delays
**Generic (whatever)-TCP strategy**

<table>
<thead>
<tr>
<th>Low-level event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reno</td>
<td>1 pkt loss</td>
</tr>
<tr>
<td>Scalable</td>
<td>ACK</td>
</tr>
<tr>
<td>CUBIC</td>
<td>Time pass 1ms</td>
</tr>
<tr>
<td>FAST</td>
<td>RTT increase x%</td>
</tr>
<tr>
<td>HTCP</td>
<td>100 ACK</td>
</tr>
</tbody>
</table>

**Hard-wired mappings: low-level events to control actions**
Generic (whatever)-TCP strategy
The event-action mappings encode a model of the network
Problem: this model is often wrong!

<table>
<thead>
<tr>
<th>Event</th>
<th>Model</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet loss</td>
<td>I’m causing congestion</td>
<td>↓↓↓↓</td>
</tr>
<tr>
<td>Shallow buffer overflow</td>
<td></td>
<td>↓</td>
</tr>
<tr>
<td>Congestion from other flows</td>
<td></td>
<td>↔</td>
</tr>
<tr>
<td>Loss is random</td>
<td></td>
<td>↑</td>
</tr>
</tbody>
</table>

...
Generalizing congestion control?

TCP ex Machina: Computer-Generated Congestion Control
Keith Weinstein and Hari Balakrishnan

PCC: Re-architecting Congestion Control for Consistent High Performance
Mo Dong*, Qingxi Li*, Doron Zarchy**, P. Brighten Godfrey*, and Michael Schapira**

BBR: Congestion-Based Congestion Control
BY NEAL CARDWELL, YUCHUNG CHENG, C. STEPHEN GUNN, SOHEIL HASSAS YEGANEH, AND VAN JACOBSON
TCP and its variants have suffered from surprisingly poor performance for decades. We argue the TCP family has little hope to a fundamental architectural deficiency: hardwiring Performance-oriented Congestion Control (PCC), a new congestion control architecture in which each sender continuously observes the connection between its senders and receivers to consistently adopt actions that result in high performance. We prove that PCC converges to a stable and fair equilibrium. Across many real-world and challenging environments, PCC shows consistent and often 10% performance improvement, with better fairness and stability compared to TCP and its variants.

**What is the right rate to send?**
What is the right rate to send?

rate $r$ → utility $u$

$U = f(tpt, \text{loss rate, latency, etc.})$

e.g. $U = tpt \times (1 - \text{loss rate})$
What is the right rate to send?

rate $r_1$ —> utility $u_1$

$U = f(tpt, \text{loss rate, latency, etc.})$

e.g. $U = tpt \times (1 - \text{loss rate})$

No matter how complex the network,
rate $r$ —> utility $u$
PCC: control based on evidence

move to $r_1$

move to $r_2$

$u_1 > u_2$?
Performance-oriented congestion control

Observe real performance

Control based on empirical evidence

move to $r_1$

move to $r_2$

$u_1 > u_2$?
Performance oriented control

$$\text{observed utility}$$

$$r(1-2\epsilon - \epsilon) \quad r \quad r(1+r\epsilon + 2\epsilon)$$

$$\text{randomized controlled trials}$$

$$\epsilon_{\text{min}} = 0.01$$
PCC: control based on evidence

This flow causing congestion

98 Mbps
102 Mbps

u1 > u2?

move to 98 Mbps

u1
u2
PCC: control based on evidence

random loss

98 Mbps
102 Mbps

$u_1 > u_2$?

move to 102 Mbps
Where is the congestion control?

Selfishly maximizing utility $\Rightarrow$ non-cooperative game

Does PCC converge to a fair Nash equilibrium?
Congestion control is in game theory

Some utility functions converge to a fair, efficient NE

\[ u_i(x) = T_i - x_i \cdot L_i \]

throughput

Cut off loss rate at 5%

Loss Rate

observed loss rate
Dynamic behavior and fairness

TCP uses AIMD for asymptotic fairness

Moving away from convergence
Dynamic behavior and fairness

PCC does not need AIMD because it looks at real performance

“Game Theory Force”

High utility
Dynamic behavior and fairness
Dynamic behavior and fairness
Deployment

No hardwired support, packet header, protocol change needed

Where to deploy?

• CDN backbone, inter-data center, dedicated scientific network
• In the wild?
TCP friendliness

PCC’s default utility function is not TCP friendly
PCC vs TCP vs Take a Flight: 100 GB

- Utah, U.S. → Berlin, Germany
- Illinois, US → Waseda, Japan
- Georgia, US → Stockholm, Sweden
- Georgia, US → Ljubljana, Slovenia
- Missouri, US → Rennes, France
- Massachusetts US → Seoul, Korea

Comparison of PCC, TCP CUBIC, and Take a Flight:

- Utah, U.S. to Berlin, Germany: 1:18:27:30
- Illinois, US to Waseda, Japan: 3:12:55:00
- Georgia, US to Ljubljana, Slovenia: 7:1:50:00
Higher performance

**InterDC**

*4X*

**Satellite Networks**

*17X*

**Lossy Networks**

*10X*

**Shallow Network Buffer**

*15X*

**Similar to ICTCP**

**IntraDC Incast**

*5X in median*

**Global Commercial Internet**

*Close to Optimal**

**Rapidly Changing Networks**

*5X in median*
Different utility functions

Same rate control algorithm +

Different utility function =

Flexibility