Performance-oriented Congestion Control

Mo Dong, Qingxi Li, Doron Zarchy*, P. Brighten Godfrey and Michael Schapira*

University of Illinois at Urbana Champaign
*Hebrew University of Jerusalem
Qingxi Li  Brighten Godfrey
University of Illinois at Urbana-Champaign

Doron Zarchy and Michael Schapira
Hebrew University of Jerusalem
TCP
Est. 1988

High BDP
- BIC
- H-TCP
- Compound
- CUBIC
- FAST TCP

Wireless
- Westwood
- Vegas
- Veno

Satellite
- Hybla
- STAR

Inter-DC
- Illinois
- SABUL

Intra-DC
- ICTCP
- DCTCP
TCP
Est. 1988

High BDP
- BIC
- H-TCP
- Compound
- CUBIC
- FAST TCP

Wireless
- Westwood
- Vegas
- Veno

Satellite
- Hybla
- STAR

Inter-DC
- Illinois
- SABUL

Intra-DC
- ICTCP
- DCTCP

...
Who can be not happy with TCP?
Who can be not happy with TCP?
<table>
<thead>
<tr>
<th>High BDP</th>
<th>Wireless</th>
<th>Satellite</th>
<th>Inter-DC</th>
<th>Intra-DC</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIC</td>
<td>Westwood</td>
<td>Hybla</td>
<td>Illinois</td>
<td>ICTCP</td>
</tr>
<tr>
<td>H-TCP</td>
<td>Vegas</td>
<td>STAR</td>
<td>SABUL</td>
<td>DCTCP</td>
</tr>
<tr>
<td>Compound</td>
<td>Veno</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUBIC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAST TCP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TCP
Est. 1988

High BDP
- BIC
- H-TCP
- Compound
- CUBIC
- FAST TCP

Wireless
- Westwood
- Vegas
- Veno

Satellite
- Hybla
- STAR

Inter-DC
- Illinois
- SABUL

Intra-DC
- ICTCP
- DCTCP

Point Solutions
<table>
<thead>
<tr>
<th>High BDP</th>
<th>Wireless</th>
<th>Satellite</th>
<th>Inter-DC</th>
<th>Intra-DC</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIC, H-TCP, Compound, CUBIC, FAST TCP</td>
<td>Westwood (Vegas, Veno)</td>
<td>Hybla (STAR)</td>
<td>Illinois (SABUL)</td>
<td>ICTCP (DCTCP)</td>
</tr>
<tr>
<td>10X</td>
<td>10X</td>
<td>17X</td>
<td>4X</td>
<td></td>
</tr>
</tbody>
</table>

Unstable, RTT Unfair, Bufferbloat, Crash on Changing Networks, .......

Point Solutions

Performance

Far from Optimal
TCP fails to achieve consistent high performance
Why is it so hard?
<table>
<thead>
<tr>
<th>TCP Variant</th>
<th>Event/Condition</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reno</td>
<td>1 pkt loss</td>
<td>cwnd/2</td>
</tr>
<tr>
<td>Scalable</td>
<td>ACK</td>
<td>cwnd+1</td>
</tr>
<tr>
<td>CUBIC</td>
<td>Time pass 1ms</td>
<td>cwnd+f(t,cwn,rtt)</td>
</tr>
<tr>
<td>FAST</td>
<td>RTT increase x%</td>
<td>Reduce cwnd to f(x)%</td>
</tr>
<tr>
<td>HTCP</td>
<td>100 ACK</td>
<td>cwnd+f(cwnd)/cwnd</td>
</tr>
<tr>
<td>Protocol</td>
<td>Event</td>
<td>Action</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Reno</td>
<td>1 pkt loss</td>
<td>cwnd/2</td>
</tr>
<tr>
<td>Scalable</td>
<td>ACK</td>
<td>cwnd+1</td>
</tr>
<tr>
<td>CUBIC</td>
<td>Time pass 1ms</td>
<td>cwnd+f(t,cwn,rtt)</td>
</tr>
<tr>
<td>FAST</td>
<td>RTT increase x%</td>
<td>Reduce cwnd to f(x)%</td>
</tr>
<tr>
<td>HTCP</td>
<td>100 ACK</td>
<td>cwnd+f(cwnd)/cwnd</td>
</tr>
</tbody>
</table>
# Hardwired Mapping

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reno</td>
<td>1 pkt loss</td>
<td>cwnd/2</td>
</tr>
<tr>
<td>Scalable</td>
<td>ACK</td>
<td>cwnd+1</td>
</tr>
<tr>
<td>CUBIC</td>
<td>Time pass 1ms</td>
<td>cwnd+f(t,cwn,rtt)</td>
</tr>
<tr>
<td>FAST</td>
<td>RTT increase x%</td>
<td>Reduce cwnd to f(x)%</td>
</tr>
<tr>
<td>HTCP</td>
<td>100 ACK</td>
<td>cwnd+f(cwnd)/cwnd</td>
</tr>
</tbody>
</table>
## Hardwired Mapping

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Event</td>
<td>Action</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Why?**

- Assumed Network

---

**Action 3**

- Network

---

**Action 4**
Event | Action
--- | ---
1 | 2
3 | 4
5 | 6

Why?

Assumed Network

Real Network

Action 3

Action 4

Performance
Event | Action
---|---

Why?

Assumed Network ➞ Real Network

Action 3 ➞ Performance ➞ Action 4

Real Network ≠ Performance
Flow $f$ sends at R

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Flow \( f \) sends at \( R \)

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td></td>
</tr>
</tbody>
</table>
Flow f sends at R

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td></td>
</tr>
</tbody>
</table>
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td></td>
</tr>
</tbody>
</table>

Network

$f$ causes most congestion
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec $R$ a lot</td>
</tr>
</tbody>
</table>

Network

$f$ causes most congestion
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec $R$ a lot</td>
</tr>
</tbody>
</table>

Network

shallow buffer overflow
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Decrease a lot</td>
</tr>
</tbody>
</table>

shallow buffer overflow
Flow f sends at R

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec R a little</td>
</tr>
</tbody>
</table>

Network

shallow buffer overflow
Flow \( f \) sends at \( R \)

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec ( R ) a lot (crossed out)</td>
</tr>
<tr>
<td></td>
<td>Dec ( R ) a little</td>
</tr>
</tbody>
</table>

Network

other high rate flow causing congestion
Flow \( f \) sends at \( R \)

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec ( R ) a lot</td>
</tr>
<tr>
<td></td>
<td>Dec ( R ) a little</td>
</tr>
</tbody>
</table>

other high rate flow causing congestion

Network
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec $R$ a lot</td>
</tr>
<tr>
<td></td>
<td>Dec $R$ a little</td>
</tr>
<tr>
<td></td>
<td>Maintain $R$</td>
</tr>
</tbody>
</table>

other high rate flow causing congestion

Network
Flow f sends at R

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec R a lot</td>
</tr>
<tr>
<td></td>
<td>Dec R a little</td>
</tr>
<tr>
<td></td>
<td>Maintain R</td>
</tr>
</tbody>
</table>

loss is random
Flow $f$ sends at $R$  

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td><em>Dec $R$ a lot</em></td>
</tr>
<tr>
<td></td>
<td><em>Dec $R$ a little</em></td>
</tr>
<tr>
<td></td>
<td><em>Maintain $R$</em></td>
</tr>
</tbody>
</table>

Network

loss is random
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dec R a lot</td>
</tr>
<tr>
<td></td>
<td>Dec R a little</td>
</tr>
<tr>
<td></td>
<td>Maintain R</td>
</tr>
<tr>
<td></td>
<td>Increase R</td>
</tr>
</tbody>
</table>

loss is random
### Flow f sends at R

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec R a lot</td>
</tr>
<tr>
<td></td>
<td>Dec R a little</td>
</tr>
<tr>
<td></td>
<td>Maintain R</td>
</tr>
<tr>
<td></td>
<td>Increase R</td>
</tr>
</tbody>
</table>

Network

loss is random
Flow $f$ sends at $R$

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Loss</td>
<td>Dec $R$ a lot</td>
</tr>
<tr>
<td></td>
<td>Dec $R$ a little</td>
</tr>
<tr>
<td></td>
<td>Maintain $R$</td>
</tr>
<tr>
<td></td>
<td>Increase $R$</td>
</tr>
</tbody>
</table>
Flow f sends at R

<table>
<thead>
<tr>
<th>Event</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet</td>
<td>Decrease R a lot</td>
</tr>
<tr>
<td></td>
<td>Maintain R</td>
</tr>
<tr>
<td></td>
<td>Increase R</td>
</tr>
</tbody>
</table>

No event-control mapping optimal for all network scenarios
What is the right rate to send?
What is the right rate to send?
What is the right rate to send?
What is the right rate to send?
What is the right rate to send?
What is the right rate to send?
What is the right rate to send?
What is the right rate to send?

rate $r$ \quad \rightarrow \quad $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?

No matter how complex the network, rate $r \rightarrow$ utility $u$

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

e.g. $U = tpt \times (1 - \text{loss rate})$
move to $r_1$
move to $r_1$

move to $r_2$

$u_1 > u_2$?
Performance-oriented Congestion Control

move to $r_1$

move to $r_2$

$u_1 > u_2$?
Performance-oriented Congestion Control

Observe real performance
Performance-oriented Congestion Control

Observe real performance
Control based on empirical evidence
Performance-oriented Congestion Control

Observe real performance

Control based on empirical evidence

yields Consistent high performance

move to $r_1$

move to $r_2$

$u_1 > u_2$?
This flow causing congestion

98 Mbps
102 Mbps

move to 98 Mbps

u_1 > u_2?

move to 102 Mbps

random loss

98 Mbps
102 Mbps

u_1 > u_2?
Consistent High Performance

4X InterDC

17X Satellite Networks

10X Lossy Networks

Solves RTT Unfairness

15X Shallow Network Buffer

Similar to ICTCP IntraDC Incast

Close to Optimal Rapidly Changing Networks

5X in median Global Commercial Internet
Software Components

Sender

Performance Oriented Control Module

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics

(tpt., loss rate, RTT)

Network

Data

SACK

Receiver

Sent Packet Log

(tpt., loss rate, RTT)
Software Components

Sender

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Receiver

Network

Data

SACK

Sender

Software Components

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Data

SACK

Sender

Software Components

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Data

SACK

Sender

Software Components

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Data

SACK

Sender

Software Components

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Data

SACK

Sender

Software Components

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Data

SACK

Sender

Software Components

Performance Oriented Control Module

(Sending Rate, Utility)

Utility Function

Sending Rate Control

Sending Module

Sent Packet Log

Monitor Module

Performance Metrics (tpt., loss rate, RTT)

Data

SACK

Sender
Performance Oriented Control

observed utility

rate
Performance Oriented Control

observed utility

? 

rate

$r$
Performance Oriented Control

\[ \text{observed utility} \]

\[ r(1 - \varepsilon) \quad r \]

rate
Performance Oriented Control

\[ r(1 - \varepsilon) \quad r \quad r(1 + \varepsilon) \]
Performance Oriented Control

\[ r(1 - \epsilon) \quad r \quad r(1 + \epsilon) \]

randomized controlled trials

observed utility

rate
Performance Oriented Control

\[ r(1 - \varepsilon) \quad r \quad r(1 + \varepsilon) \]

randomized controlled trials
Performance Oriented Control

observed utility

randomized controlled trials

$\text{rate}$

$\frac{r(1 - \varepsilon)}{r}$

$\frac{r(1 + \varepsilon)}{r}$

$\frac{r(1 - \varepsilon)}{r}$
Performance Oriented Control

observed utility

rate

$\text{randomized controlled trials}$

$r(1 - \varepsilon)$  $r$  $r(1 + \varepsilon)$
Performance Oriented Control

\[ r(1 - 2\varepsilon) \quad r \quad r(1 + 2\varepsilon) \]

observed utility

\[ \varepsilon_{\text{min}} = 0.01 \]

randomized controlled trials
Where is Congestion Control?

$\langle r_1, r_2 \rangle$
Where is Congestion Control?

Selfishly maximizing utility
Where is Congestion Control?

Selfishly maximizing utility => non-cooperative game
Where is Congestion Control?

Selfishly maximizing utility
=> non-cooperative game

Do we converge to a fair Nash equilibrium?
A class of utility functions converge to a fair and efficient Nash Equilibrium
A class of utility functions converge to a fair and efficient Nash Equilibrium

\[ u_i(x) = T_i - x_i L_i \]
A class of utility functions converge to a fair and efficient Nash Equilibrium

\[ u_i(x) = T_i \]

throughput

\[ -x_i * L_i \]

sending rate

observed loss rate
A class of utility functions converge to a fair and efficient Nash Equilibrium

\[ u_i(x) = T_i \times \text{sigmoid}(L_i - 0.05) - x_i \times L_i \]

Cut off loss rate at 5%
TCP Dynamics
TCP Dynamics

AIMD as a “hack” to asymptotic fairness
TCP Dynamics

AIMD as a “hack” to asymptotic fairness
TCP Dynamics

AIMD as a “hack” to asymptotic fairness
TCP Dynamics

AIMD as a “hack” to asymptotic fairness
TCP Dynamics

AIMD as a “hack” to asymptotic fairness

Moving away from convergence
PCC Dynamics

Graph showing the variable $C$ over time $t$, with two horizontal lines at $C = 24$. The graph is static with no trend or change over time.
PCC Dynamics

PCC does not need AIMD because it looks at real performance
PCC Dynamics

PCC does not need AIMD because it looks at real performance
PCC Dynamics

PCC does not need AIMD because it looks at real performance
PCC Dynamics

PCC does not need AIMD because it looks at real performance

“Game Theory Force”

High Utility
Convergence
Convergence

500s Interval
2000s/each

100Mbps, 30ms
Deployment

• No hardwired support, packet header, protocol change needed

• Where to deploy
  • CDN backbone, Inter-data center, dedicated scientific nw
  • “In the Wild”? 
TCP Friendliness
TCP Friendliness

- Is PCC TCP-friendly?
TCP Friendliness

- Is PCC TCP-friendly?
TCP Friendliness
TCP Friendliness

- PCC's default utility function is not TCP Friendly
TCP Friendliness

• PCC’s default utility function is not TCP Friendly

But not *that* bad
TCP Friendliness

- PCC's default utility function is not TCP Friendly

But not that bad
TCP Friendliness

- PCC's default utility function is not TCP Friendly

But not that bad
TCP Friendliness

- PCC’s default utility function is not TCP Friendly
- Different utility functions can be a solution
TCP Friendliness

• PCC’s default utility function is not TCP Friendly

• Different utility functions can be a solution

TCP vs TCP

TCP vs PCC
TCP Friendliness

• PCC’s default utility function is not TCP Friendly

• Different utility functions can be a solution

TCP vs TCP vs PCC
TCP Friendliness

• PCC’s default utility function is not TCP Friendly

• Different utility functions can be a solution

<table>
<thead>
<tr>
<th>β = 10</th>
<th>10Mbit/s</th>
<th>50Mbit/s</th>
<th>90Mbit/s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30ms</td>
<td>60ms</td>
<td>90ms</td>
</tr>
<tr>
<td>0.94</td>
<td>0.75</td>
<td>0.67</td>
<td></td>
</tr>
<tr>
<td>0.74</td>
<td>0.73</td>
<td>0.81</td>
<td></td>
</tr>
<tr>
<td>0.89</td>
<td>0.91</td>
<td>1.01</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>β = 100</th>
<th>10Mbit/s</th>
<th>50Mbit/s</th>
<th>90Mbit/s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30ms</td>
<td>60ms</td>
<td>90ms</td>
</tr>
<tr>
<td>0.71</td>
<td>0.58</td>
<td>0.63</td>
<td></td>
</tr>
<tr>
<td>0.56</td>
<td>0.58</td>
<td>0.54</td>
<td></td>
</tr>
<tr>
<td>0.63</td>
<td>0.62</td>
<td>0.88</td>
<td></td>
</tr>
</tbody>
</table>
Consistent High Performance
Consistent High Performance

Satellite Networks
Consistent High Performance

Satellite Networks

17X vs Hybla
Consistent High Performance

Satellite Networks

Lossy Networks

17X vs Hybla

Throughput (Mbps)

Bottleneck Buffersize (KB)

PCC

TCP Hybla

TCP Illinois

TCP CUBIC

TCP New Reno

Loss Rate

Throughput (Mbps)

PCC

TCP Illinois

TCP CUBIC

31
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

Shallow
Network Buffer
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer

IntraDC
Incast
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer

Similar to ICTCP
IntraDC
Incast
Consistent High Performance

**17X vs Hybla**
Satellite Networks

**10X vs Illinois**
Lossy Networks

**15X less buffer**
Shallow Network Buffer

**Similar to ICTCP**

IntraDC
Incast

Rapidly Changing Networks

---

Goodput (Mbps)
Number of Senders

PCC data=256KB
PCC data=128KB
PCC data=64KB
TCP data=256KB
TCP data=128KB
TCP data=64KB

---

IntraDC
Incast

Rapidly Changing Networks

---

Optimal

PCC
TCP Illinois
TCP CUBIC

---

Satellite Networks

Lossy Networks

Shallow Network Buffer

Rapidly Changing Networks
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer

Similar to ICTCP
IntraDC
Incast

Close to Optimal
Rapidly Changing Networks

IntraDC
Incast
Consistent High Performance

17X vs Hybla Satellite Networks
10X vs Illinois Lossy Networks

15X less buffer Shallow Network Buffer

Similar to ICTCP IntraDC Incast
Close to Optimal Rapidly Changing Networks

RTT Unfairness

IntraDC
Incast

IntraDC
Incast

SendingRate (Mbps)
Time (s)

Goodput (Mbps)
Number of Senders

Goodput (Mbps)
Number of Senders

Goodput (Mbps)
Number of Senders

IntraDC
Incast

IntraDC
Incast

IntraDC
Incast

IntraDC
Incast

IntraDC
Incast

IntraDC
Incast
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer

Similar to ICTCP
IntraDC
Incast

Close to Optimal
Rapidly Changing Networks

Solves
RTT Unfairness

IntraDC
Incast

Goodput (Mbps)

Number of Senders

SendingRate(Mbps)

Time(s)

Relative throughput of long-RTT flow

RTT of Long-RTT flow (ms)
Consistent High Performance

Satellite Networks
- 17X vs Hybla
- 10X vs Illinois

Lossy Networks
- Similar to ICTCP
- IntraDC
- Incast

Rapidly Changing Networks
- Close to Optimal
- Rapidly Changing Networks

Network Buffer
- 15X less buffer
- Shallow Network Buffer

IntraDC
- Incast

Table 1: PCC significantly outperforms TCP in inter-data center environments. RTT is in msec; throughput in Mbps.

<table>
<thead>
<tr>
<th>Transmission Pair</th>
<th>RTT</th>
<th>PCC</th>
<th>SABUL</th>
<th>CUBIC</th>
<th>Illinois</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPO → NYSERNet</td>
<td>129</td>
<td>326</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPO → Missouri</td>
<td>90.7</td>
<td>90.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPO → Illinois</td>
<td>86.6</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NYSERNet → Missouri</td>
<td>47.4</td>
<td>816</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wisconsin → Missouri</td>
<td>9.03</td>
<td>801</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPO → Wisc.</td>
<td>38.9</td>
<td>783</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NYSERNet → Wisc.</td>
<td>38.3</td>
<td>791</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Missouri → Wisc.</td>
<td>20.9</td>
<td>807</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NYSERNet → Illinois</td>
<td>36.1</td>
<td>808</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCC data=256KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCC data=128KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCC data=64KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP data=256KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP data=128KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCP data=64KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Solves
- RTT Unfairness
Consistent High Performance

**17X vs Hybla Satellite Networks**

**10X vs Illinois Lossy Networks**

**15X less buffer Shallow Network Buffer**

**Similar to ICTCP IntraDC Incast**

**Close to Optimal Rapidly Changing Networks**

**Solves RTT Unfairness**

**4X vs Illinois 1.23X vs UDT InterDC**

Table 1: PCC significantly outperforms TCP in inter-data center environments. RTT is in m; throughputs in Mbps.

<table>
<thead>
<tr>
<th>Source 1</th>
<th>Source 2</th>
<th>InterDC 1</th>
<th>InterDC 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>329</td>
<td>326</td>
<td>329</td>
<td>326</td>
</tr>
<tr>
<td>354</td>
<td>366</td>
<td>354</td>
<td>366</td>
</tr>
<tr>
<td>279</td>
<td>297</td>
<td>279</td>
<td>297</td>
</tr>
<tr>
<td>47.4</td>
<td>48.6</td>
<td>47.4</td>
<td>48.6</td>
</tr>
<tr>
<td>20.9</td>
<td>21.5</td>
<td>20.9</td>
<td>21.5</td>
</tr>
<tr>
<td>36.1</td>
<td>37.2</td>
<td>36.1</td>
<td>37.2</td>
</tr>
</tbody>
</table>
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer

Similar to ICTCP
IntraDC

Close to Optimal
Rapidly Changing Networks

Solves
RTT Unfairness

4X vs Illinois
Incast

1.23X vs UDT

Global Commercial Internet
Consistent High Performance

17X vs Hybla
Satellite Networks

10X vs Illinois
Lossy Networks

15X less buffer
Shallow Network Buffer

Similar to ICTCP
IntraDC
Incast

Close to Optimal
Rapidly Changing Networks

Solves
RTT Unfairness

4X vs Illinois
1.23X vs UDT
InterDC

4X median vs CUBIC
1.48X vs UDT + 4X less loss
Global Commercial Internet

Table 1: PCC significantly outperforms TCP in inter-data center environments. RTT is in ms; throughputs in Mbps.

<table>
<thead>
<tr>
<th>Source</th>
<th>PCC</th>
<th>TCP</th>
<th>Throughput (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCC data=256KB</td>
<td>129</td>
<td>326</td>
<td></td>
</tr>
<tr>
<td>PCC data=128KB</td>
<td>80.7</td>
<td>90.1</td>
<td></td>
</tr>
<tr>
<td>PCC data=64KB</td>
<td>47.4</td>
<td>816</td>
<td></td>
</tr>
<tr>
<td>TCP data=256KB</td>
<td>700</td>
<td>547</td>
<td></td>
</tr>
<tr>
<td>TCP data=128KB</td>
<td>38.0</td>
<td>487</td>
<td></td>
</tr>
<tr>
<td>TCP data=64KB</td>
<td>20.9</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>PCC</td>
<td>36.1</td>
<td>674</td>
<td></td>
</tr>
<tr>
<td>TCP</td>
<td>0.1</td>
<td>141</td>
<td></td>
</tr>
</tbody>
</table>
Consistent High Performance

Satellite Network

WINDS System

![Throughput vs. Bottleneck buffer size graph](image)

- **Link capacity**: 17X improvement
- **Throughput (Mbps)**
- **Bottleneck buffer size (KB)**
Consistent High Performance

Rapidly Changing Networks

BW: 10-100Mbps; RTT: 10-100ms; Loss Rate: 0-1%
Change every 5 seconds
Consistent High Performance

Rapidly Changing Networks

BW: 10-100Mbps; RTT: 10-100ms; Loss Rate: 0-1%
Change every 5 seconds

Sending Rate (Mbps) vs. Time (s)

- Optimal
- TCP Illinois
- TCP CUBIC
Consistent High Performance

Rapidly Changing Networks

BW: 10-100Mbps; RTT: 10-100ms; Loss Rate: 0-1%
Change every 5 seconds
Global Commercial Internet
Global Commercial Internet
PCC vs TCP vs Take a Flight

100G Data
PCC vs TCP vs Take a Flight

100G Data

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

0:00:00  1:18:27:30  3:12:55:00  5:7:22:30  7:1:50:00

PCC
TCP CUBIC
Take a Flight
PCC vs TCP vs Take a Flight

100G Data

- 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

0:00:00 → 1:18:27:30 → 3:12:55:00 → 5:7:22:30 → 7:1:50:00

- PCC
- TCP CUBIC
- Take a Flight
PCC vs TCP vs Take a Flight

100G Data

- 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

Time:
- 0:0:00:00
- 1:18:27:30
- 3:12:55:00
- 5:7:22:30
- 7:1:50:00
PCC vs TCP vs Take a Flight

100G Data

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

0:00:00:00 1:18:27:30 3:12:55:00 5:7:22:30 7:1:50:00

- PCC
- TCP CUBIC
- Take a Flight
Consistent High Performance

Global Commercial Internet

Throughput Improvement Ratio vs Percentage of Trials

- PCC vs TCP CUBIC
Consistent High Performance

Global Commercial Internet

Throughput Improvement Ratio

Percentage of Trials

44%, 10X

PCC vs TCP CUBIC
Consistent High Performance

Global Commercial Internet

Throughput Ratio over TCP CUBIC

Percentage of Trials

PCC_d
Consistent High Performance

Global Commercial Internet

Percentage of Trials vs. Throughput Ratio over TCP CUBIC

Default utility function

PCC_d
Consistent High Performance

Global Commercial Internet

Percentage of Trials

Throughput Ratio over TCP CUBIC

Default utility function

PCC_f
PCC_d
Consistent High Performance

Global Commercial Internet

Throughput Ratio over TCP CUBIC

Percentage of Trials

Default utility function

Friendly Utility Function

PCC_f
PCC_d
Consistent High Performance

Global Commercial Internet

Friendly Utility Function

Default utility function

Throughput Ratio over TCP CUBIC

Percentage of Trials
Consistent High Performance

Global Commercial Internet

- Intra-continent 2.33X median
- Inter-continent 25X median
Consistent High Performance

Global Commercial Internet

![Graph showing performance comparison]

- Green line: PCC vs SABUL
- Blue line: PCC vs PCP
- Red line: PCC vs TCP CUBIC

Percentage of Trials vs Throughput Improvement Ratio
Consistent High Performance

Global Commercial Internet

1.48X Median
4X less loss
Long list of things we have done but don’t have time to talk about

- More stories about the fact that TCP is broken
- Proof of Nash Equilibrium and Convergence
- Concrete Implementation of PCC
  - Performance monitoring
  - Details of learning control algorithm
  - Implementation designs and optimizations
- Performance Evaluation
  - Inter data center networks
  - Small buffer networks
  - Reactiveness and stability tradeoff
  - Jain index fairness
  - Benefit of Randomized Control Trials
  - Details of TCP friendliness evaluation
  - Emulated satellite networks
  - Emulated datacenter networks
  - Cure RTT unfairness
  - Does not fundamentally harm short flow FCT
  - Evaluation in the wild vs non-TCP protocols
- Flexibility by pluggable utility function
Long list of things we have done but don’t have time to talk about

- More stories about the fact that TCP is broken
- Proof of Nash Equilibrium and Convergence
- Concrete Implementation of PCC
  - Performance monitoring
  - Details of learning control algorithm
  - Implementation designs and optimizations
- Performance Evaluation
  - Inter data center networks
  - Small buffer networks
  - Reactiveness and stability tradeoff
  - Jain index fairness
  - Benefit of Randomized Control Trials
  - Details of TCP friendliness evaluation
  - Emulated satellite networks
  - Emulated datacenter networks
  - Cure RTT unfairness
  - Does not fundamentally harm short flow FCT
  - Evaluation in the wild vs non-TCP protocols
- Flexibility by pluggable utility function

11 more!
Same Rate Control Algorithm
Same Rate Control Algorithm +
Same Rate Control Algorithm + Different Utility Function = Flexibility
Limitation and Future Work
Limitation and Future Work

• Understanding of Utility Function and Convergence
• Better Control Algorithm
• Better Scalability
Limitation and Future Work

• Understanding of Utility Function and Convergence
• Better Control Algorithm
• Better Scalability
Related Works
Related Works

• TCP Family: TCP exMachina (Remy)
Related Works

- TCP Family: TCP exMachina (Remy)
- Non-TCP protocols: PCP
Related Works

- TCP Family: TCP exMachina (Remy)
- Non-TCP protocols: PCP
- In-network feedback protocols: DCTCP, XCP
Rate control based on empirically observed performance yields

- Consistent high performance
- Better stability than TCP
- Flexible performance objectives
- Relatively easy and safe to deploy
speedier.net/pcc

github.com/modong/pcc

Find me to see a demo