PCC Vivace: Online-Learning Congestion Control

Mo Dong[1], **Tong Meng**[1], Doron Zarchy[2], Engin Arslan[3], Yossi Gilad[4],
P. Brighten Godfrey[1], Michael Schapira[2]

[1] UIUC, [2] Hebrew University of Jerusalem, [3] University of Nevada, Reno, [4] MIT

Internet Congestion Control

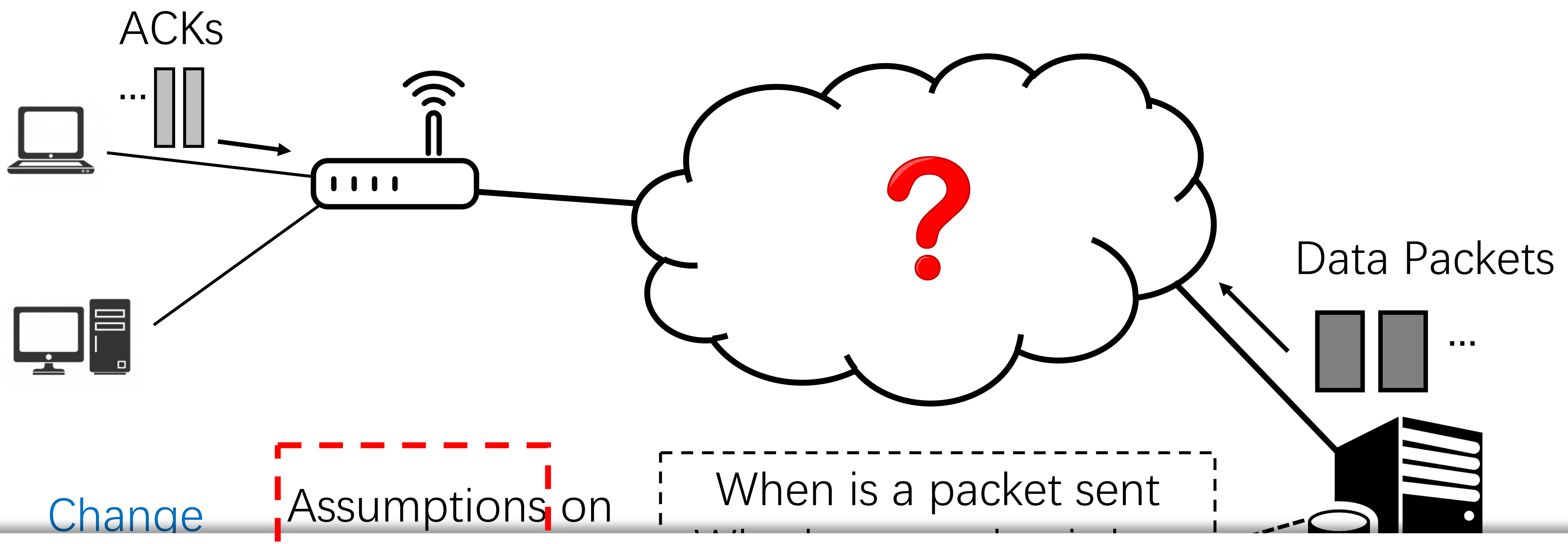
[Winstein et a
- Offline-op
- Generated

[Dong et al. N
- Utility fram
- Online lea

[Cardwell et al. Queue 2016]
- Bottleneck bandwidth probing
- Minimum RTT probing

pla
FAST
H-TCP
PCC
...
BBR

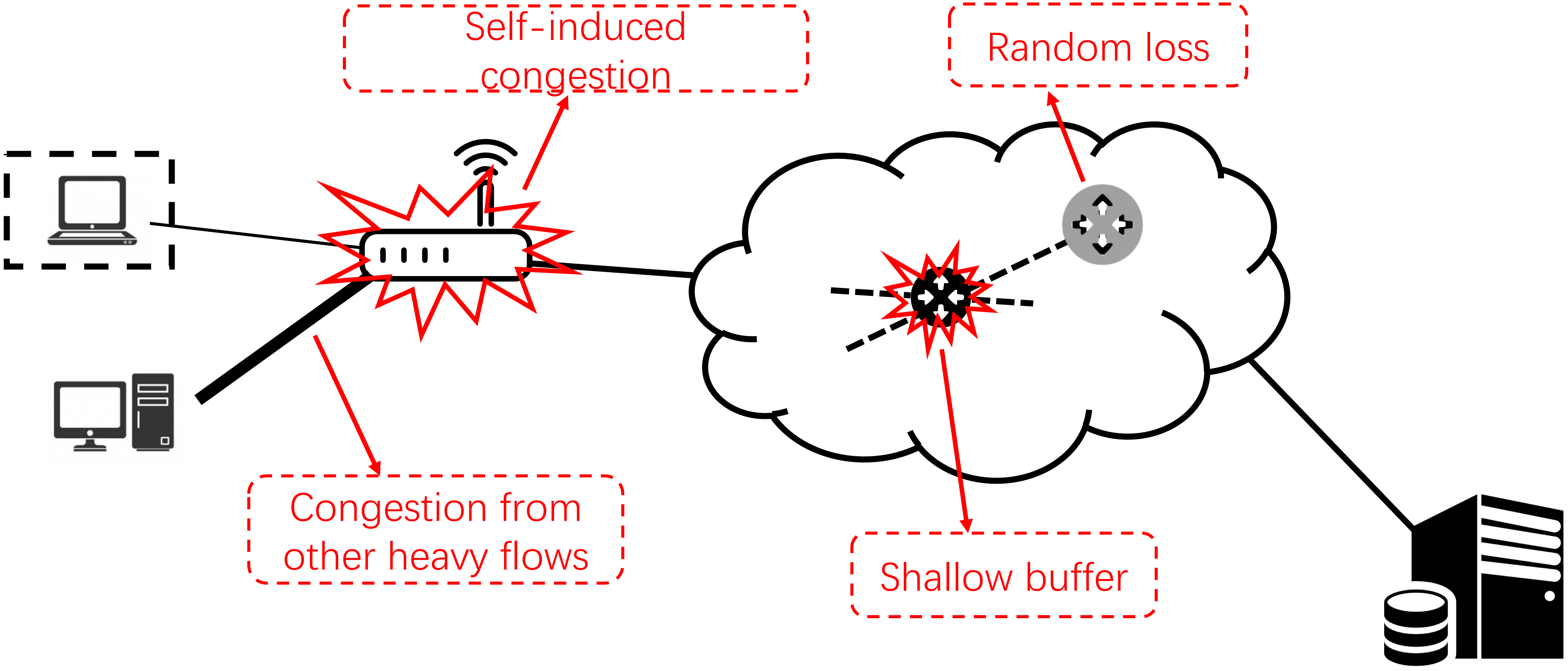
Internet Congestion Control



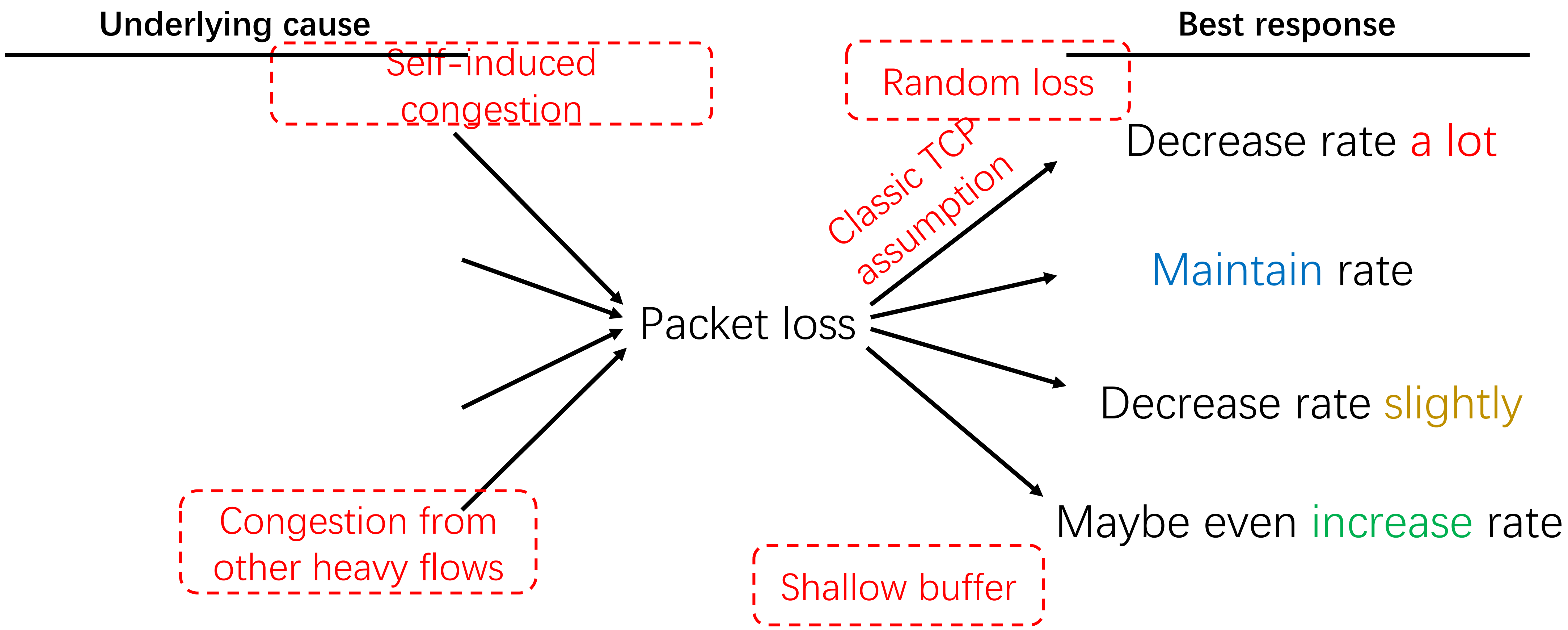
Change Assumptions on

Packet loss / RTT increment indicates congestion

Internet Congestion Control

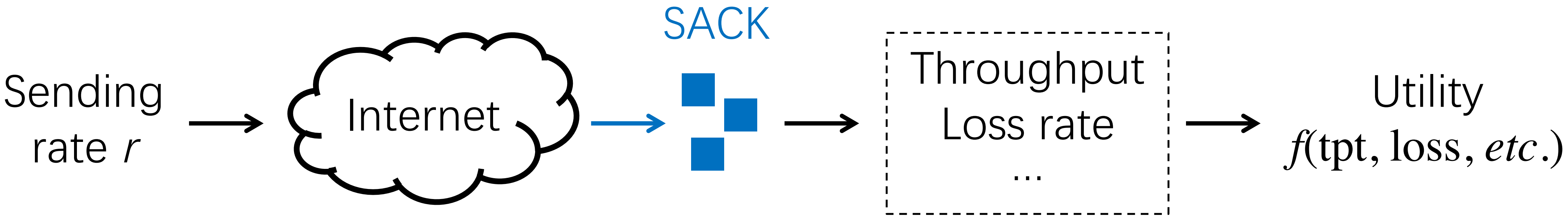


Strong Assumptions Cause Problem

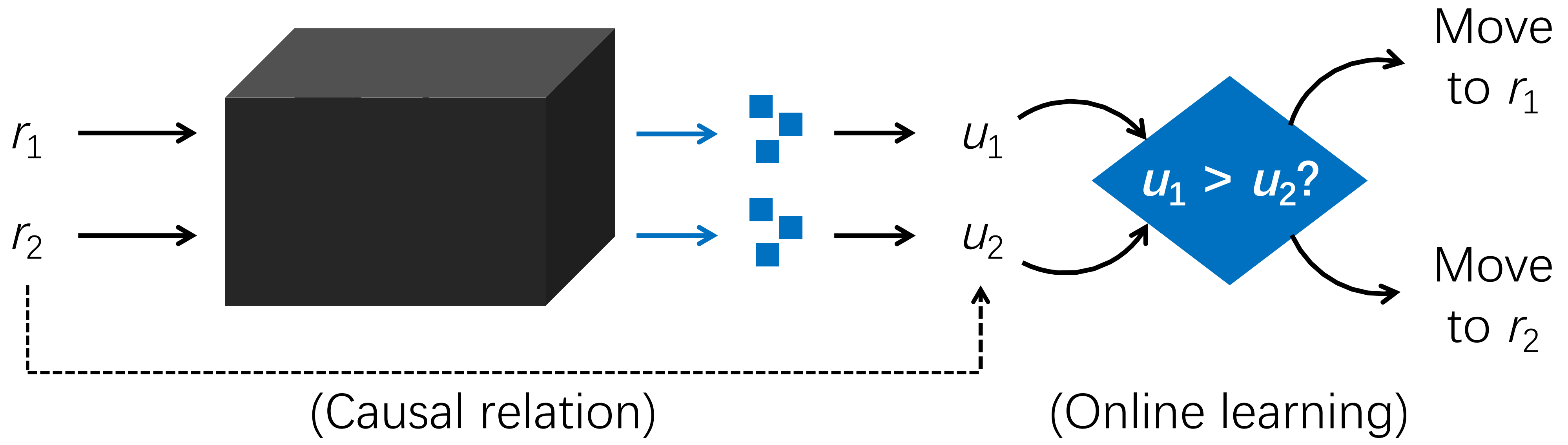


Abstract assumption cannot capture Internet complexity

PCC Utility Framework

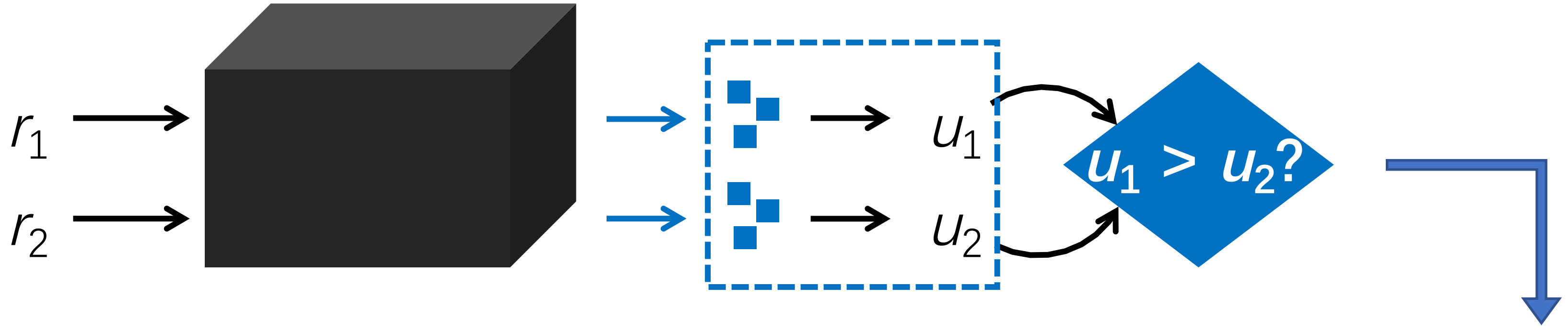


PCC Rate Control



Sender selfishly maximizes its own utility
(*online learning* in *non-cooperative* game)

PCC Design Challenges



utility function + Rate control algorithm

Requirements for consistently high performance:

- Capture application performance objectives
- Guarantee equilibrium with multiple competing senders
- Guarantee reaching equilibrium upon convergence
- Rapidly adapt to network dynamics

PCC Allegro

[Dong et al. NSDI 2015]

Loss-based utility function

$$u_i(x_i) = T_i \cdot Sigmoid_\alpha(L_i - 0.05) - x_i \cdot L_i$$

$$T_i = x_i(1 - L_i)$$

$$Sigmoid_\alpha(y) = \frac{1}{1+e^{\alpha y}}$$

↓

No latency-awareness

Can cause bufferbloat

+

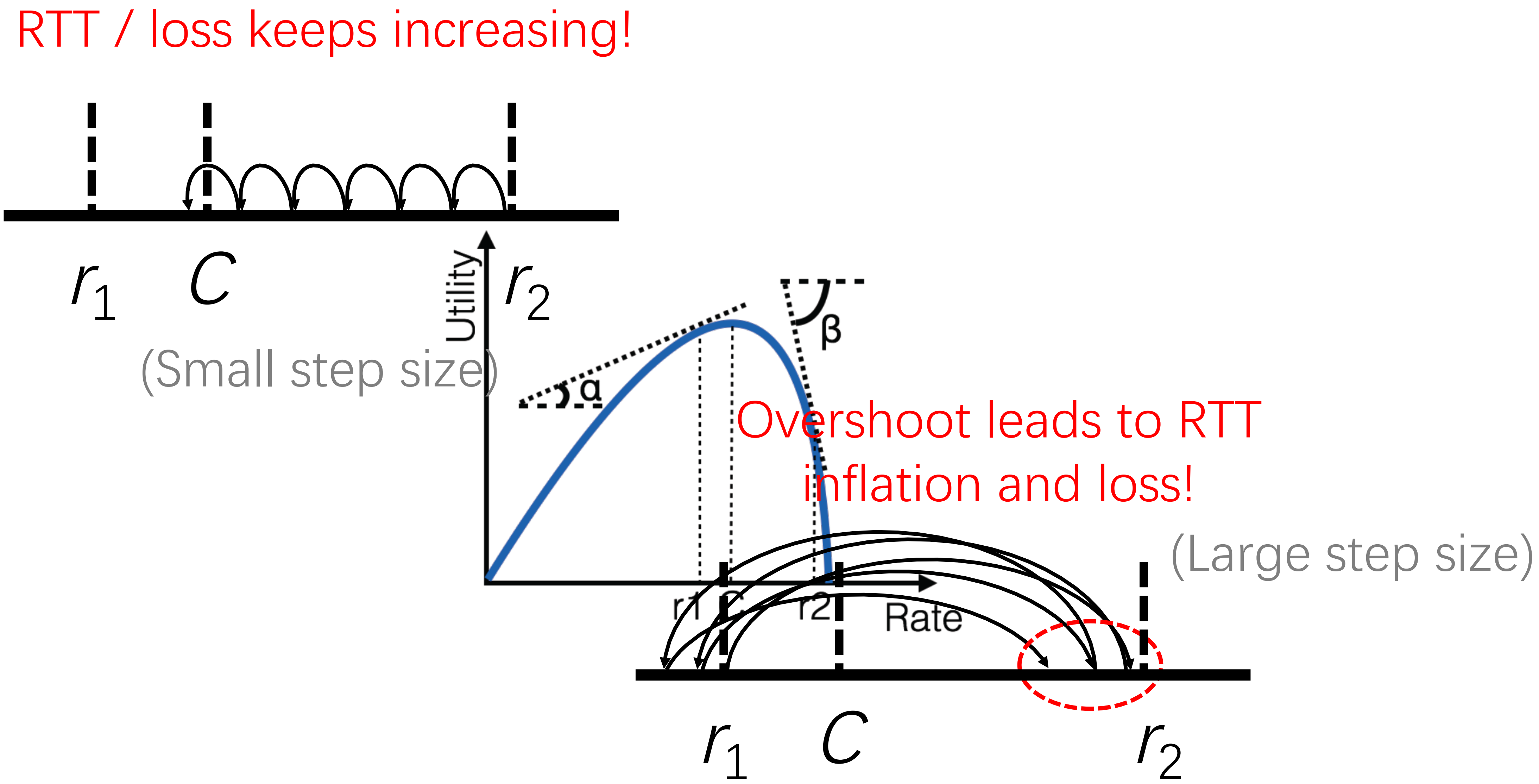
Heuristic rate control

Fixed rate change step size

↓

Slow convergence

Slow reaction to network changes


RTT / loss keeps increasing!
r_1
C
r_2
(Small step size)
Utility
α
β
Overshoot leads to RTT inflation and loss!
r1
C
r2
Rate
(Large step size)
r_1
C
r_2

PCC Vivace

- Leveraging powerful tools from online learning theory

New utility function framework

- Latency-awareness
- Strictly concave ⇒ Equilibrium guarantee
- Flexibility among senders

New control algorithm

- Gradient-ascent ⇒ Convergence speed/stability
- Deals with measurement noise

PCC Vivace

Strictly concave utility function → Strict socially concave game; Unique convergence equilibrium

$$u\left(x_i, \frac{d(RTT_i)}{dT}, L_i\right) = x_i^t - b x_i \frac{d(RTT_i)}{dT} - c x_i \times L_i$$

$$0 < t < 1, b \geq 0, c > 0$$

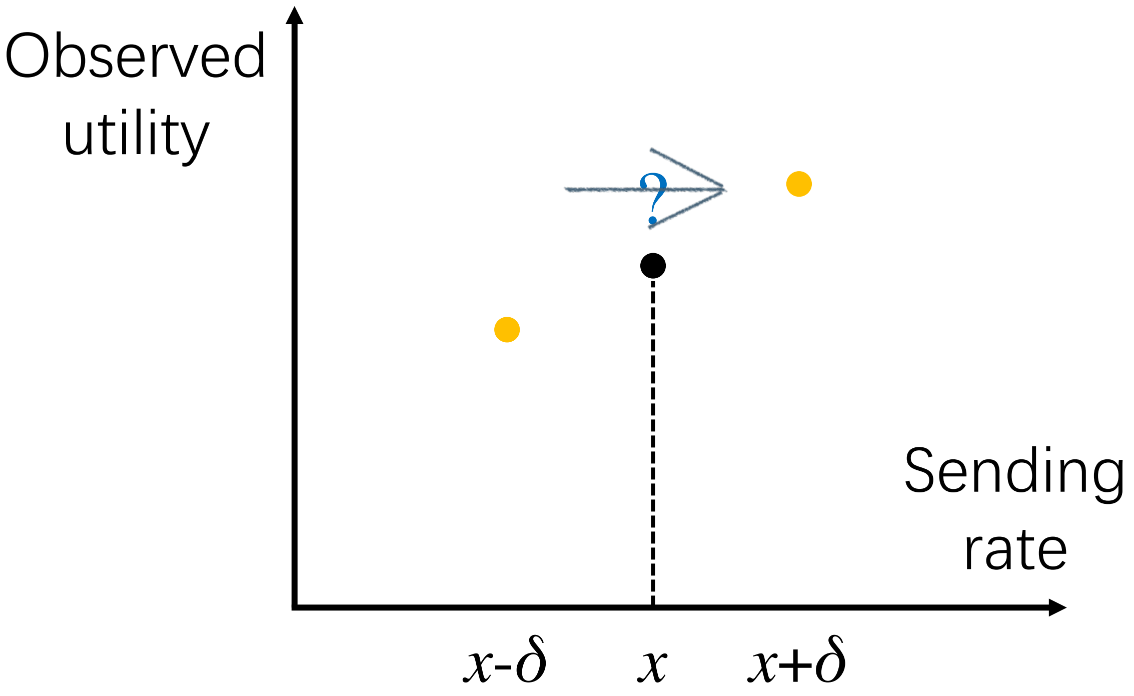
Tolerate p-random-loss if

$$c = \frac{tC^{t-1}}{p}$$

No latency inflation upon convergence if

$$b \geq tn^{2-t}C^{t-1}$$

PCC Vivace

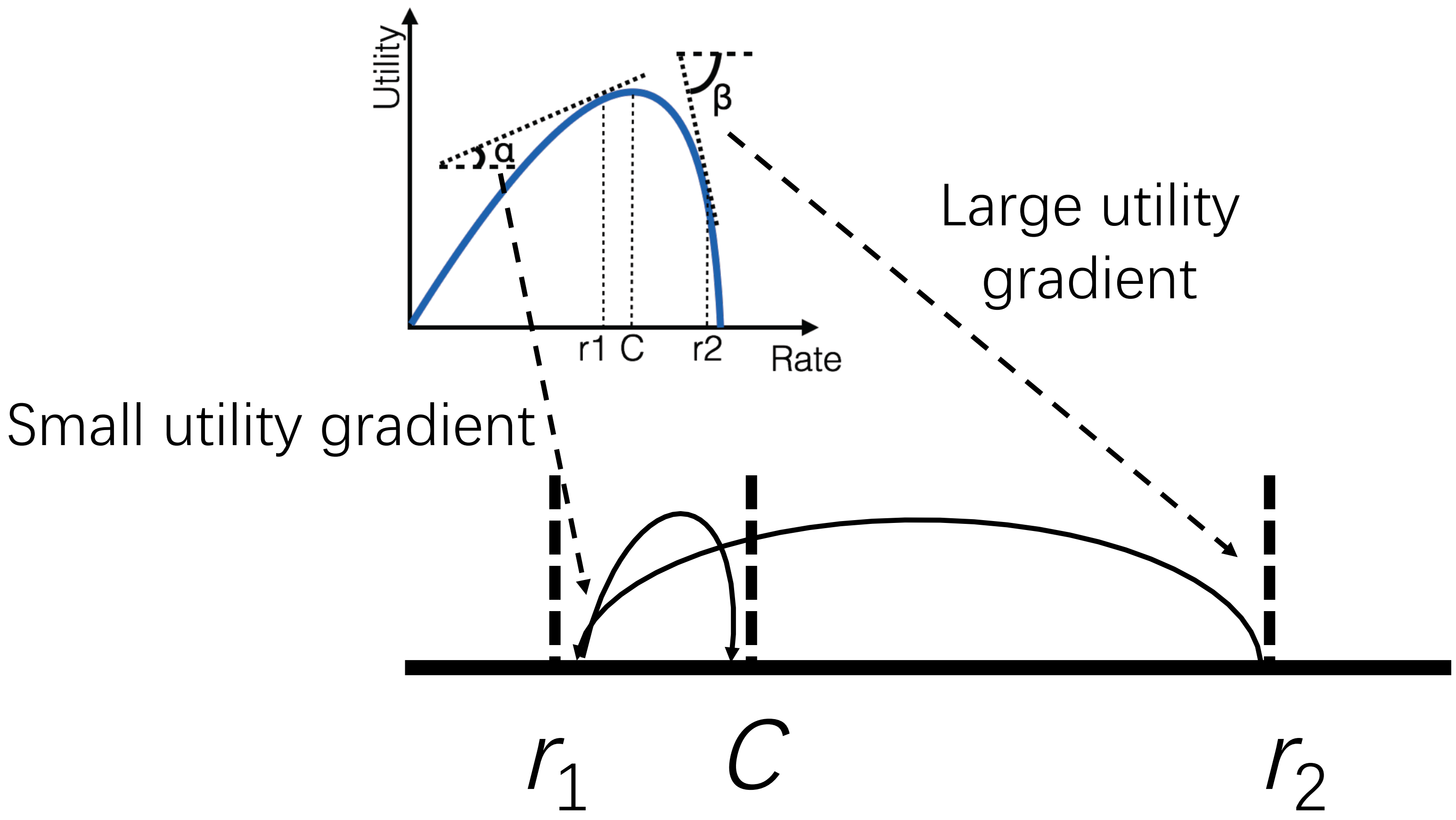


Gradient-ascent rate control

$$\frac{u(x+\delta)-u(x-\delta)}{2\delta} \rightarrow \alpha \cdot \Delta x$$

Techniques to deal with measurement noise:

- Linear regression
- RTT gradient low-pass filter
- Double check



Utility
α
β
r1 C
r2
Rate
Large utility gradient
Small utility gradient
r_1
C
r_2

PCC Vivace

Strictly concave utility function

+

Gradient-ascent rate control

$$u\left(x_i, \frac{d(RTT_i)}{dT}, L_i\right) = x_i^t - b x_i \frac{d(RTT_i)}{dT} - c x_i \times L_i$$

$$0 < t < 1, b \geq 0, c > 0$$

$$\frac{u(x+\delta) - u(x-\delta)}{2\delta} \to \alpha \cdot \Delta x$$

↓

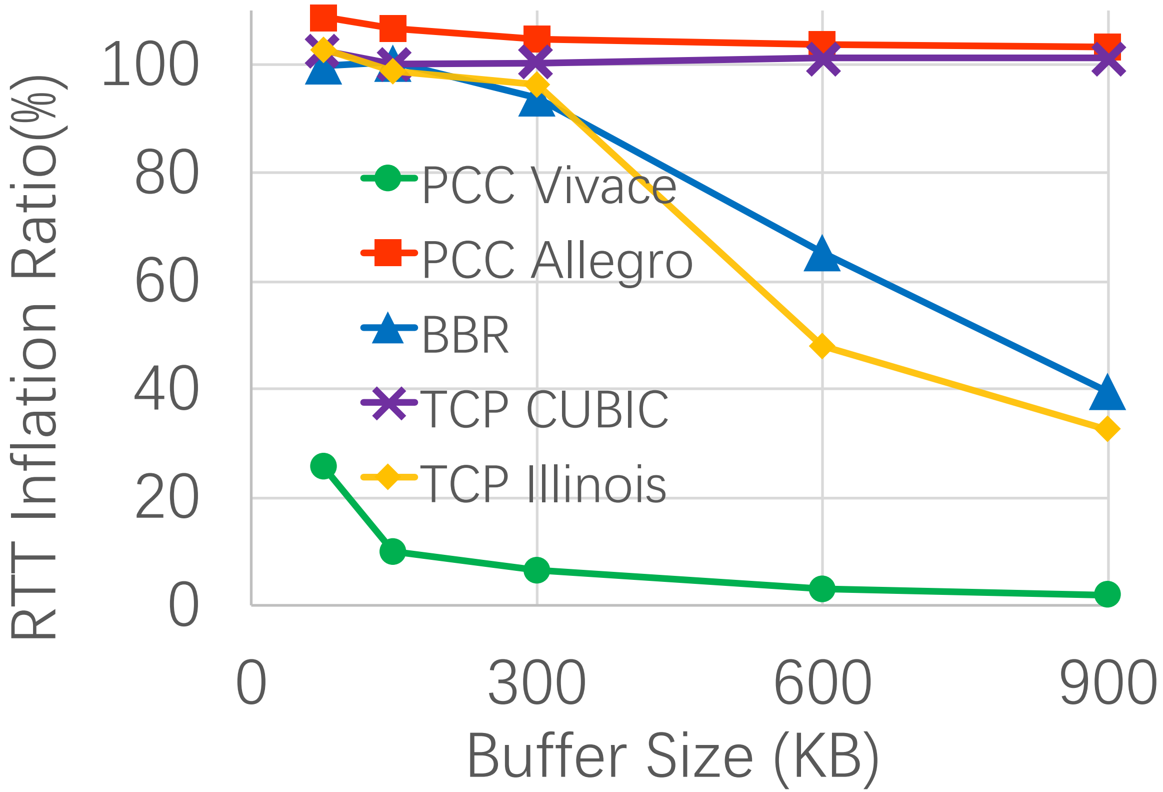
"No-regret" guarantee:
A Powerful lens for analysis

Evaluation

- Implementation
  - UDT-based user-space implementation
  - Emulab experiments, Amazon EC2 experiments
  - User-space PCC proxy for video streaming
- Protocols for comparison
  - TCP variants (TCP CUBIC, TCP Illinois, TCP Vegas, *etc.*)
  - BBR
  - PCC Allegro
  - PCC Vivace

Vivace Utility Performance

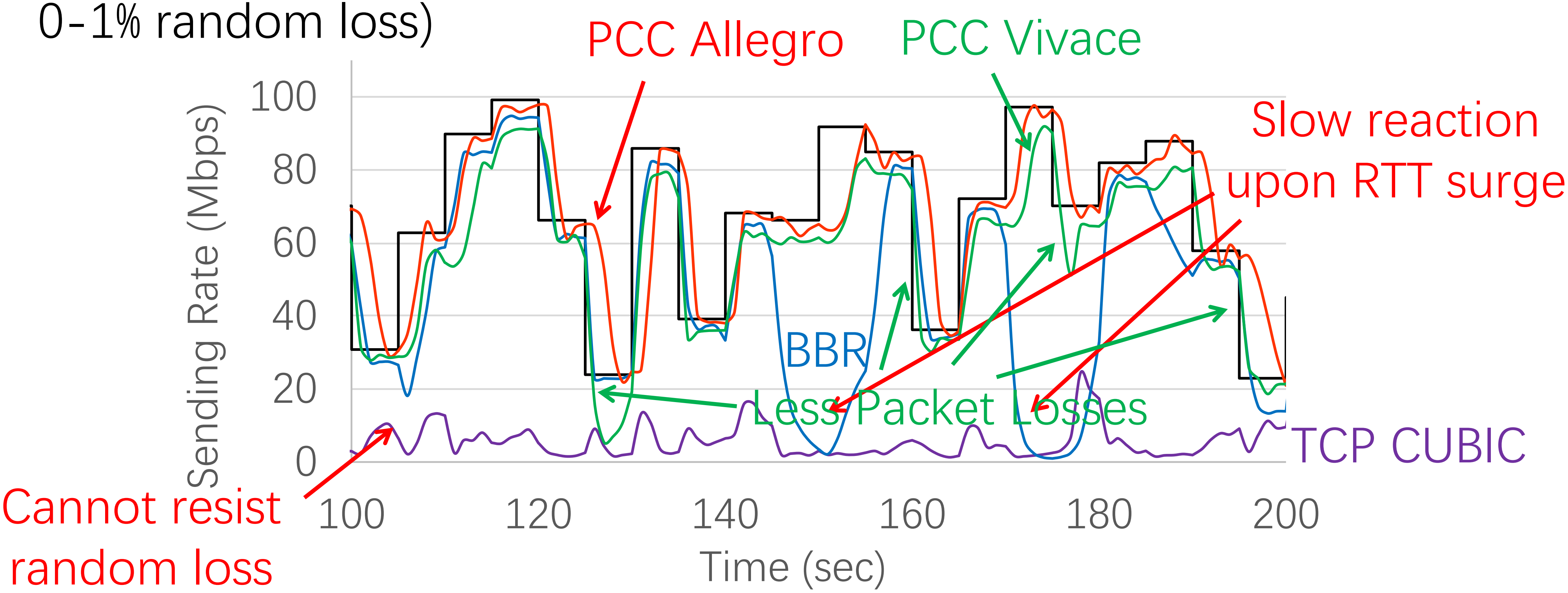
- Latency awareness (100Mbps, 30ms RTT Emulab bottleneck link)



< 2ms inflation in all cases
90% smaller than BBR under 2BDP

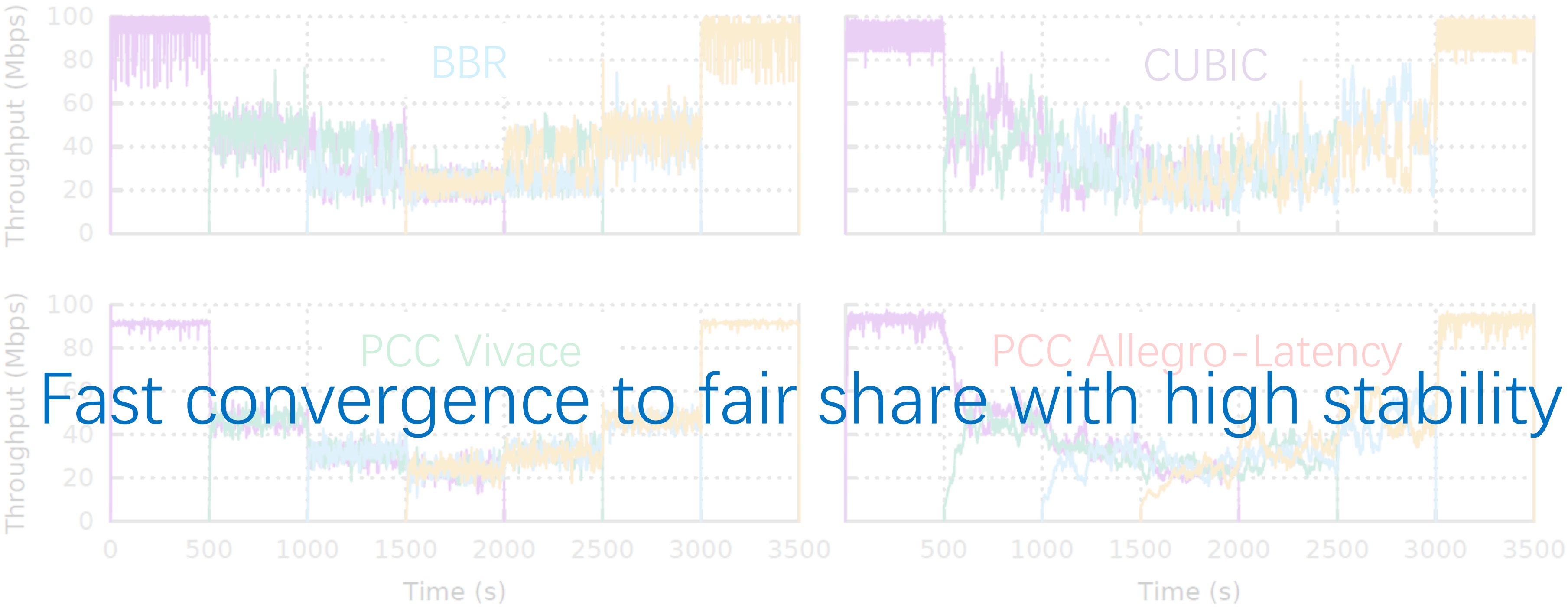
Vivace Rate Control Performance

- Rapid reaction to network changes (10-100Mbps, 10-100ms RTT, 0-1% random loss)



Vivace upon Convergence

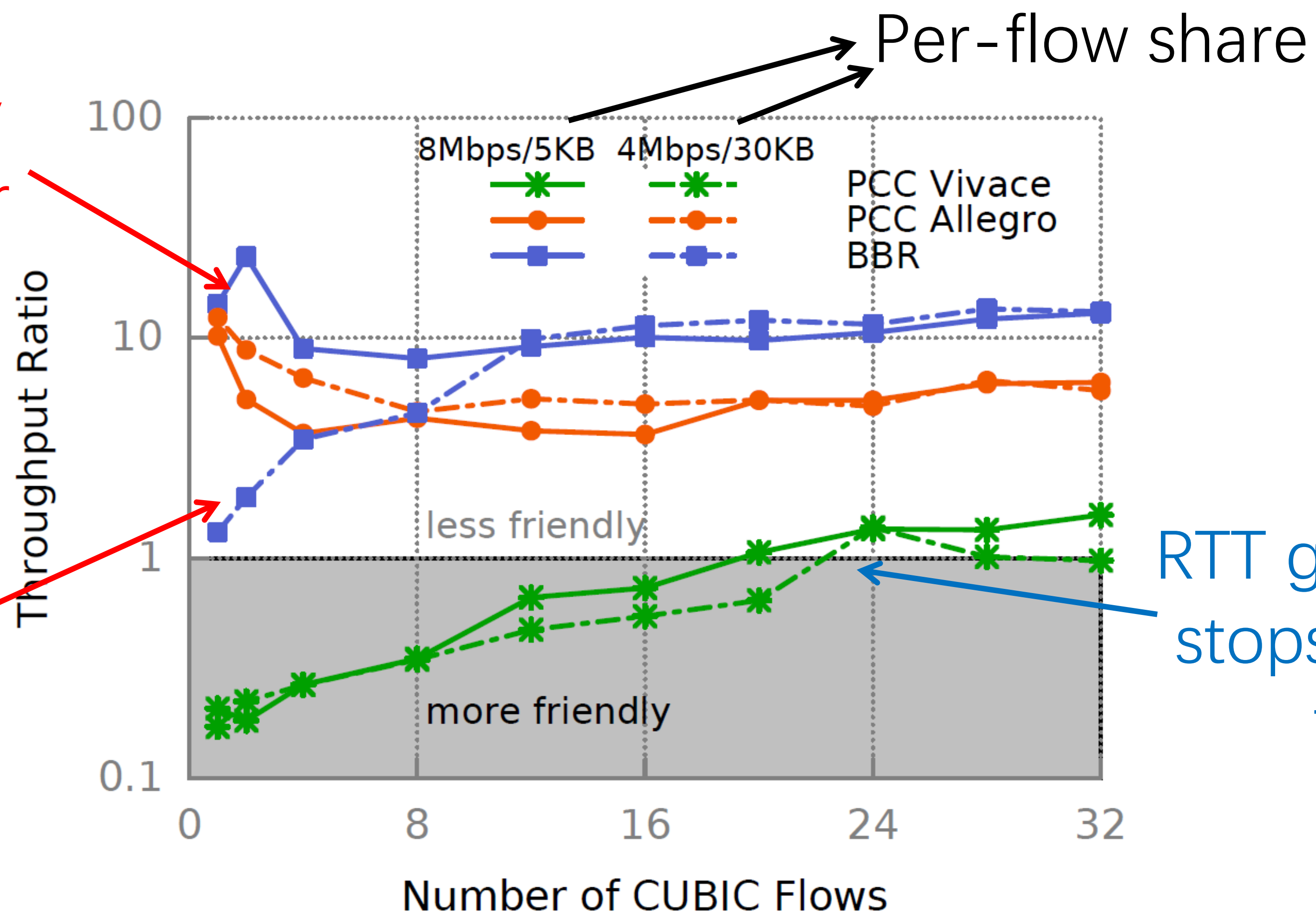
- Fair equilibrium (100Mbps, 30ms RTT, 75KB buffer)



Fast convergence to fair share with high stability

TCP Friendliness

BBR not friendly with small buffer



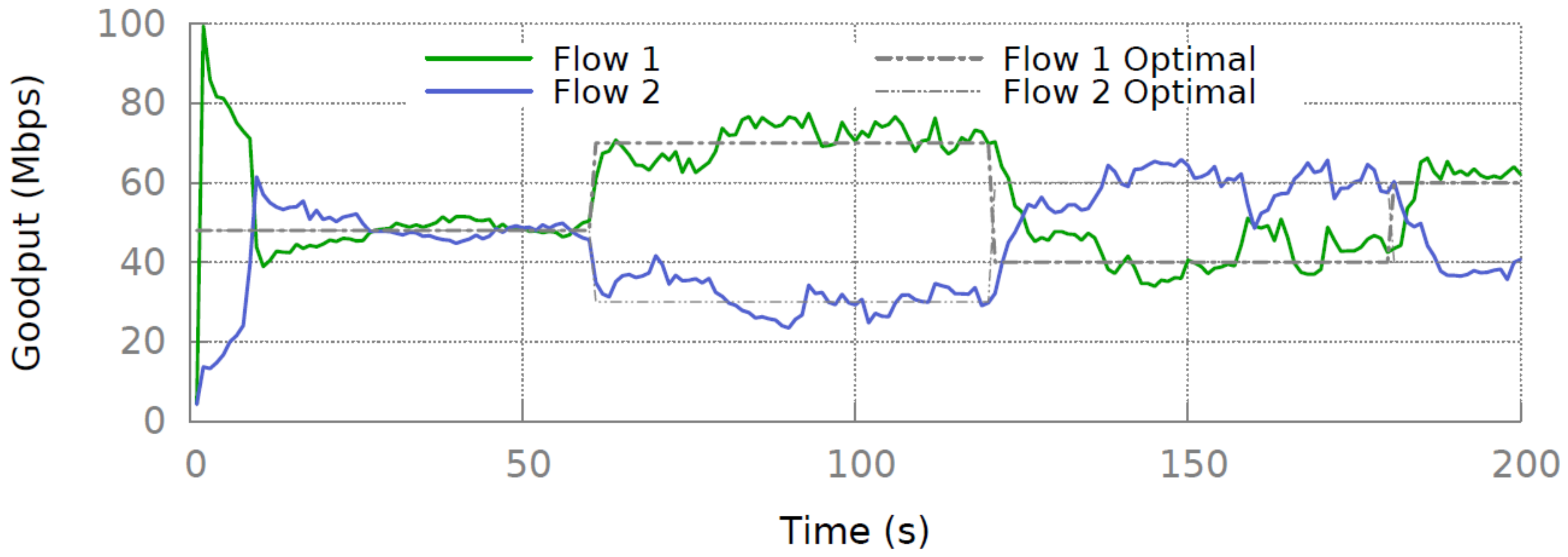
BBR keeps grabbing 50% bandwidth

RTT gradient → 0, stops being over friendly

Insights from Learning-Theoretic Tools

- Flexible equilibrium state with heterogeneous senders

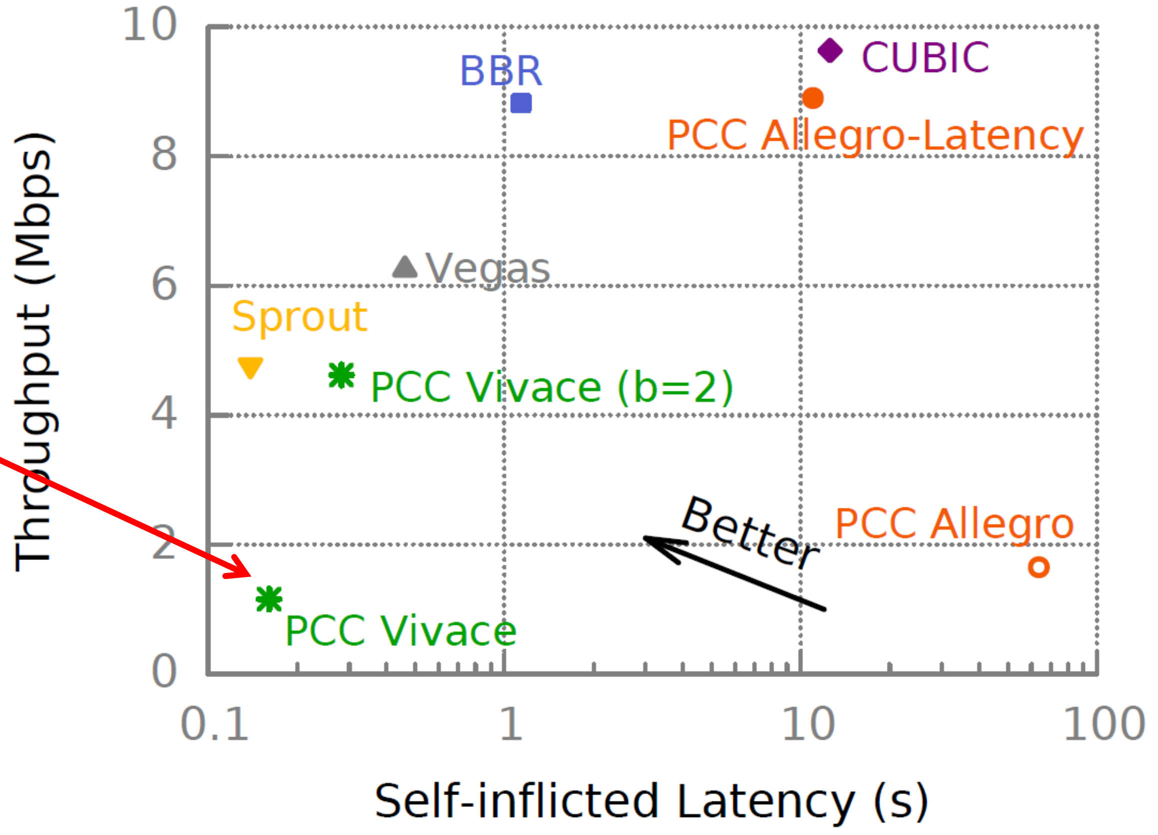
$$u(x_i, L_i) = x_i - c_i x_i \left(\frac{1}{1 - L_i} - 1 \right) \qquad c_i = \frac{C}{x_i^*}$$



Limitation in Extremely Dynamic Networks

- LTE (Mahimahi emulator [Netravali et al. ATC 2015])

Low Throughput



https://www.youtube.com/watch?v=Y3IzuCdwdUo&t=27s
(Demo comparing PCC with UDP and TCP video streaming)

PCC In Action

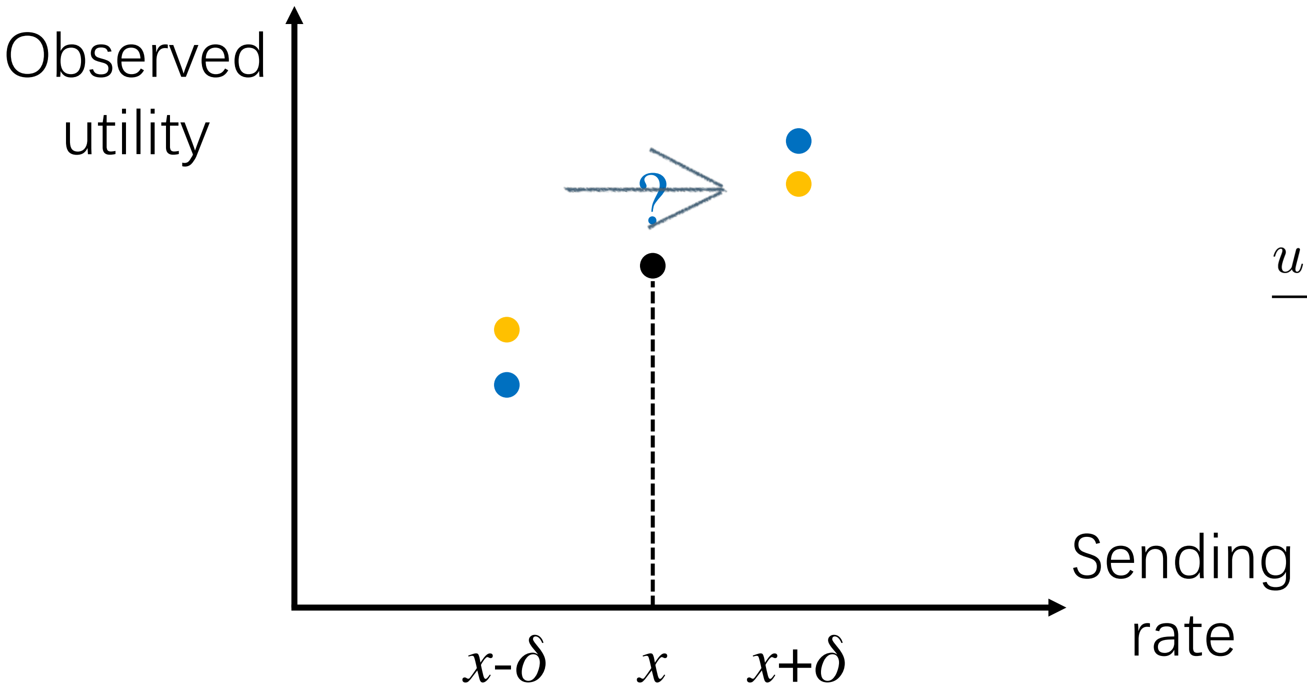
- Open source release on GitHub (https://github.com/pccproject)
  - UDP implementation used in experiments presented here
  - QUIC implementation with Google
  - Pantheon implementation for test purpose
  - Kernel implementation in the works
- VACC variant of PCC by HUAWEI and vodafone
  - Kernel implementation with optimizations for video over LTE
  - Ongoing research project with successful field tests

Conclusion

- PCC Vivace: Leveraging no-regret learning for congestion control
  - Consistent high performance as PCC Allegro
  - Latency awareness, mitigated bufferbloat (latency inflation, congestion loss)
  - Provably fair, yet also flexible equilibrium convergence
  - Fast and stable convergence, even with changing network conditions
  - Improved TCP friendliness, safer to deploy
- Thanks for generous project support

Thanks!

PCC Vivace



Gradient-ascent

~~Heuristic~~

rate control

$$\frac{u(x+\delta) - u(x-\delta)}{2\delta} \rightarrow \alpha \cdot \Delta x$$

PCC Vivace

Strictly concave

~~Loss-based~~

utility function

$$u\left(x_i, \frac{d(RTT_i)}{dT}, L_i\right) = x_i^t - b x_i \frac{d(RTT_i)}{dT} - c x_i \times L_i$$

$$0 < t < 1, b \geq 0, c > 0$$

Linear regression

Low pass filter (> 0.01)

+

Gradient-ascent

~~Heuristic~~

rate control

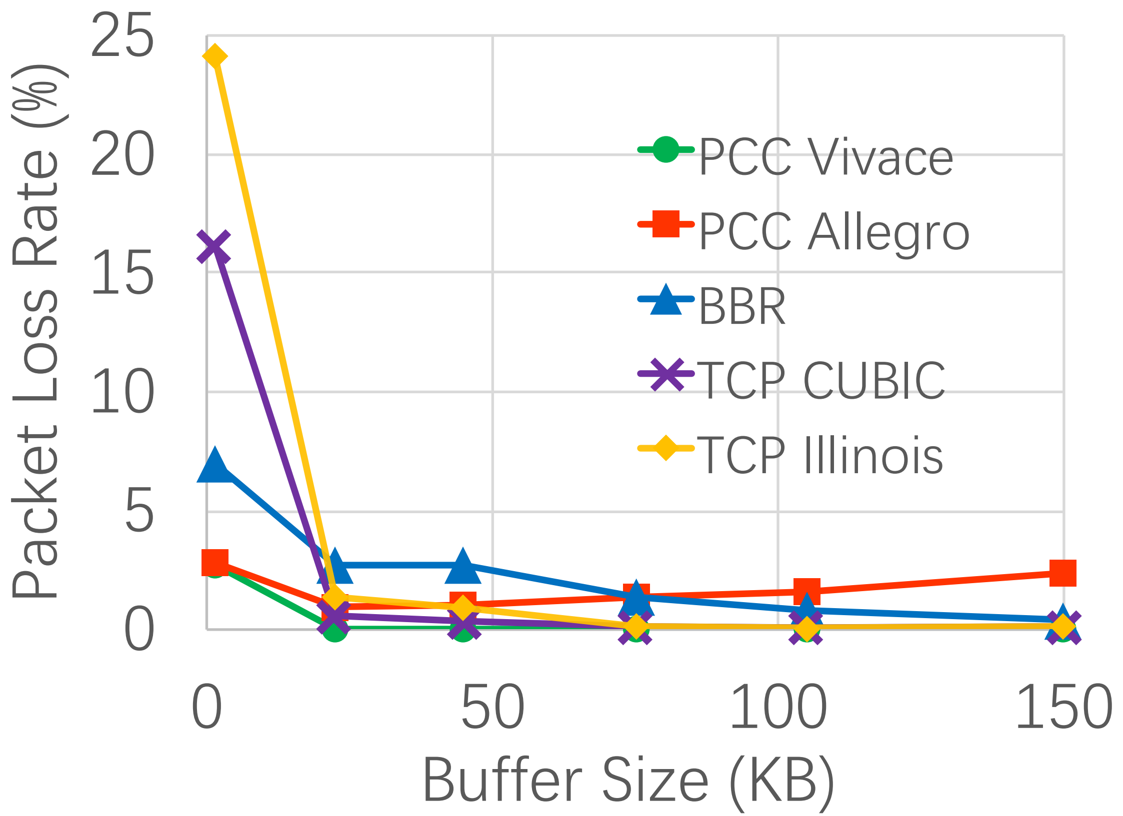
$$\frac{u(x+\delta) - u(x-\delta)}{2\delta} \rightarrow \alpha \cdot \Delta x$$

$L(x+\delta) = 0.01\%$ $L(x\text{-}\delta) = 2.0\%$

Double check

Vivace Utility Performance

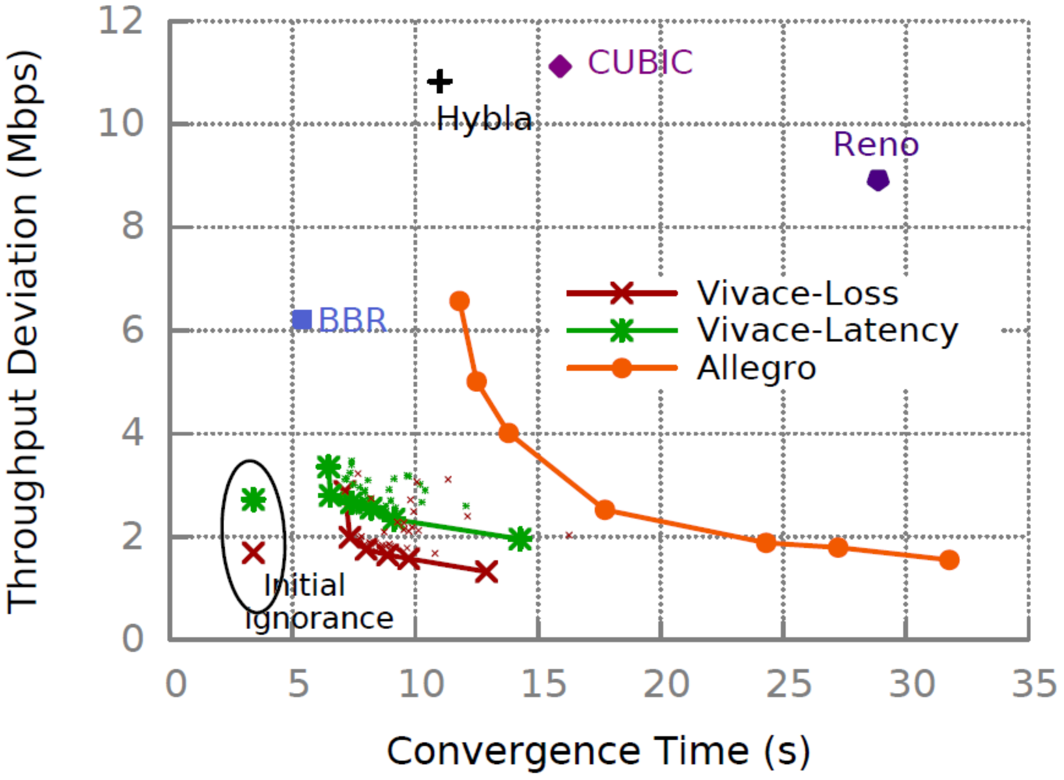
- Latency awareness (100Mbps, 30ms RTT Emulab bottleneck link)



< 0.5% loss with 13.5KB buffer
55% smaller than TCP CUBIC

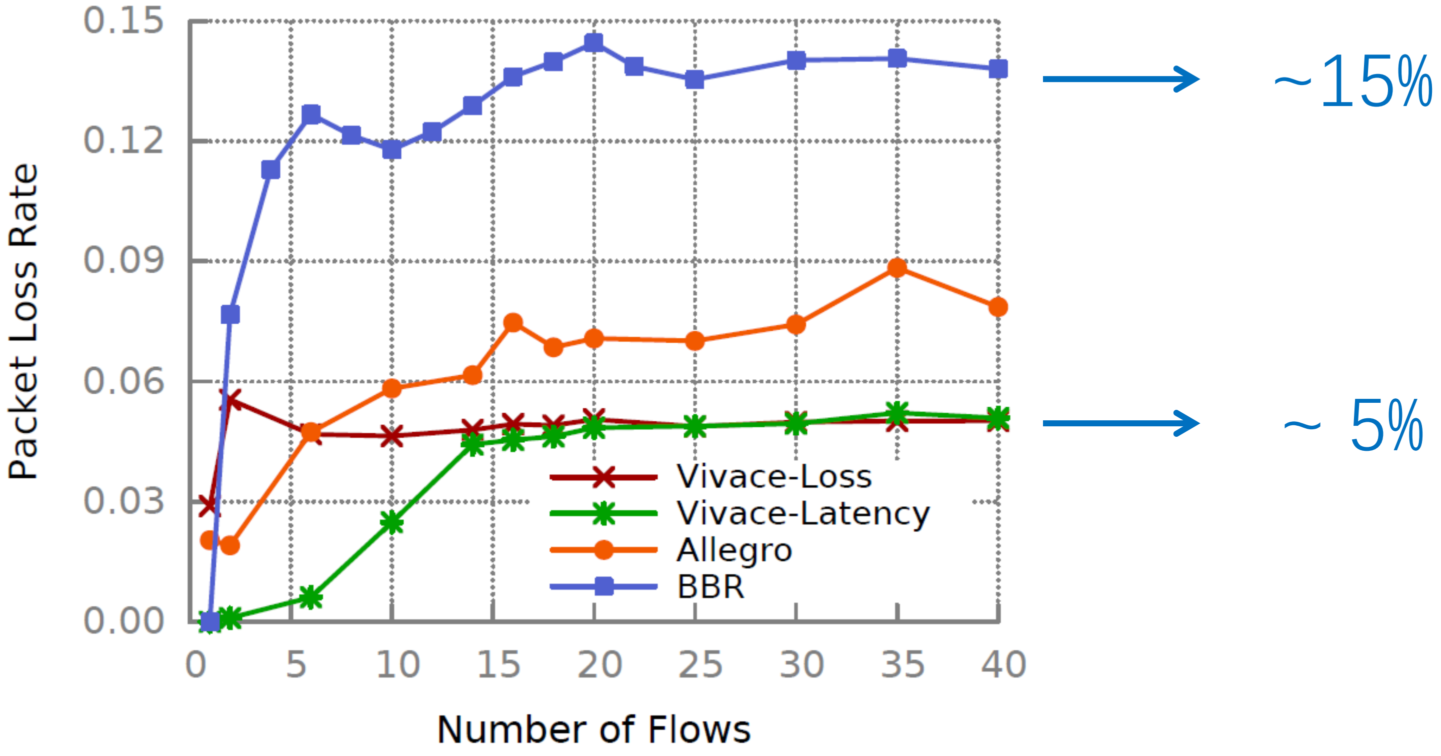
Vivace Rate Control Performance

- Convergence Speed/Stability Tradeoff (100Mbps, 30ms RTT, 75KB buffer)

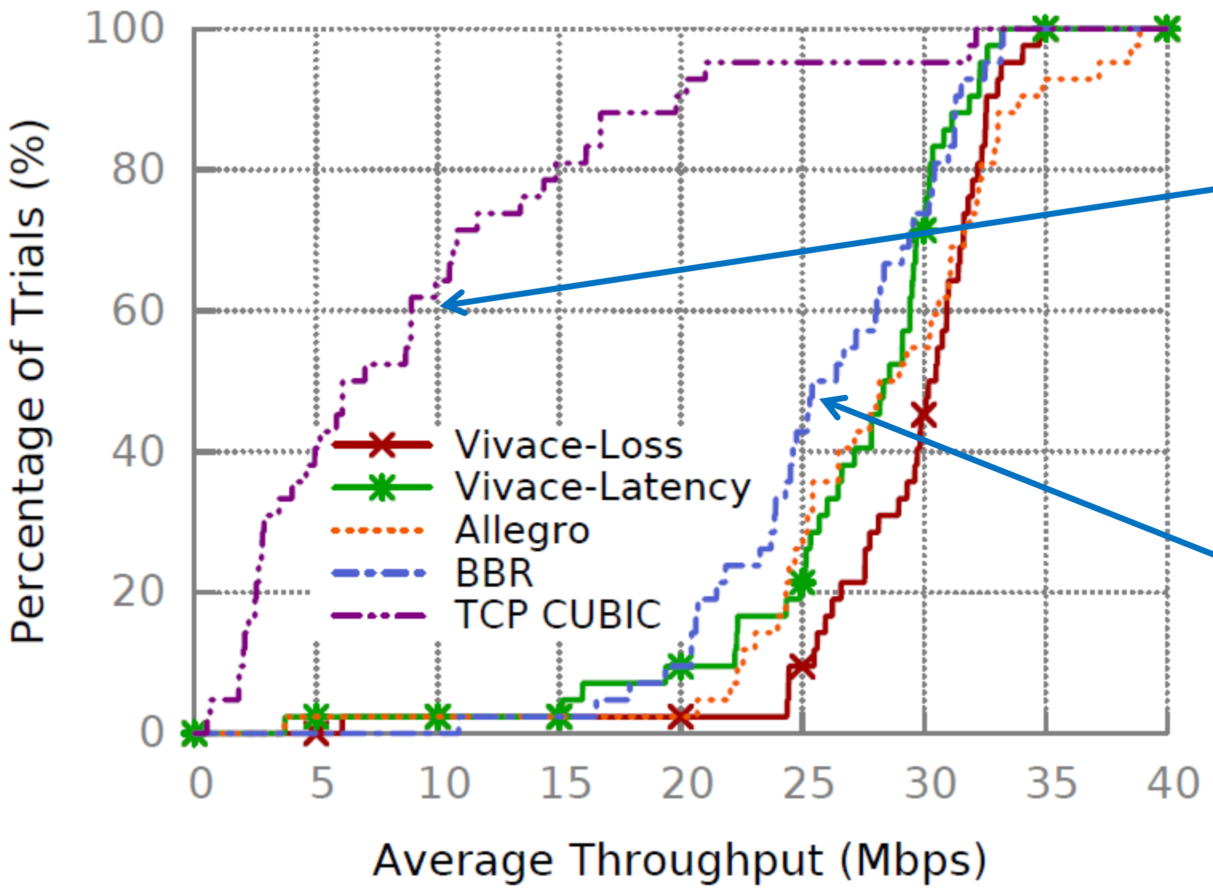


Insights from No-Regret Guarantee

- Random loss tolerance vs. Congestion loss (8Mbps, 25KB per-flow share)



Performance in Real-World



3.7x median gain over CUBIC

11.6% median gain over BBR

Limitation of No-Regret

"Sender's choices of rates are asymptotically (across time) no worse, utility-wise, than sending at what would have been (in hindsight) the best *fixed* rate"

Still make sense in highly dynamic environment?