Modeling BBR's Interactions With Loss-Based Congestion Control

Ranysha Ware
Carnegie Mellon University

Matthew K. Mukerjee
Nefeli Networks

Justine Sherry
Carnegie Mellon University

Srinivasan Seshan
Carnegie Mellon University

Is BBRv1 fair to legacy congestion control algorithms?

Prior work has tried to answer the fairness question with **measurement.**

Prior work has tried to answer the fairness question with measurement.

BBR is fair to Cubic in deep-buffered networks.

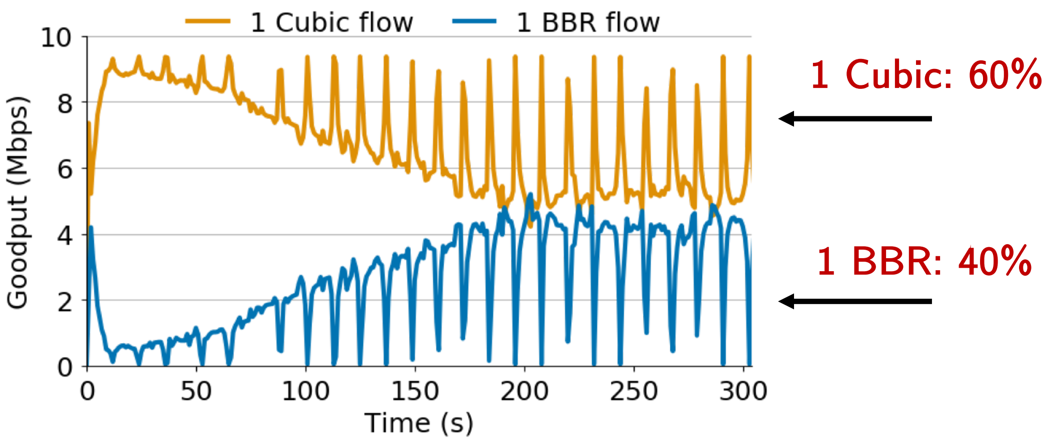


Figure: 1 BBR vs. 1 Cubic.
(10 Mbps network, 32 BDP queue)

Reference: N. Cardwell, et.al. 2016. BBR: Congestion control. In Presentation at IETF97
.

Prior work has tried to answer the fairness question with **measurement.**

BBR is fair to Cubic in deep-buffered networks.

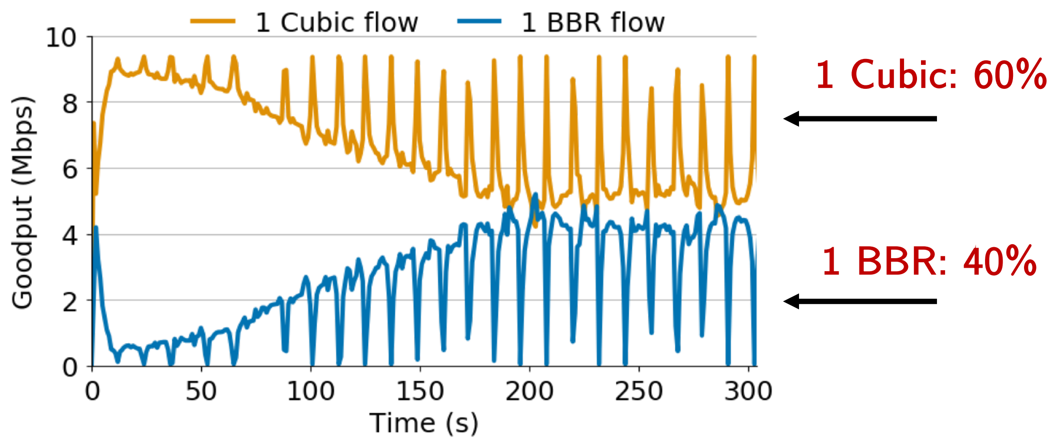


Figure: 1 BBR vs. 1 Cubic.
(10 Mbps network, 32 BDP queue)

BBR is unfair to Cubic in deep-buffered networks.

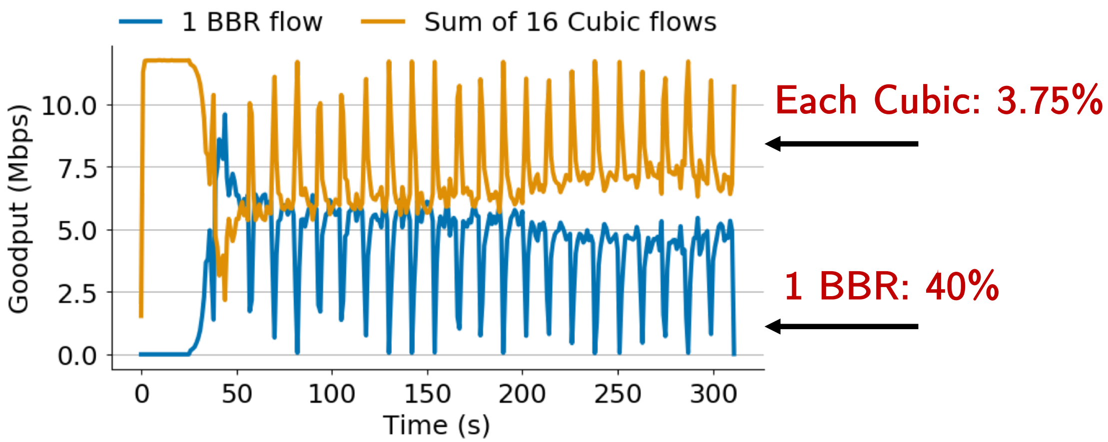


Figure: 1 BBR vs. 16 Cubic.
(10 Mbps network, 32 BDP queue)

Reference: Ware et. al. The Battle for Bandwidth: Fairness and Heterogenous Congestion Control. NSDI 2018.

Prior work does not explain why we see certain behavior.

BBR is <u>fair to Cubic</u> in deep-buffered networks.

BBR is <u>unfair to Cubic</u> in deep-buffered networks.

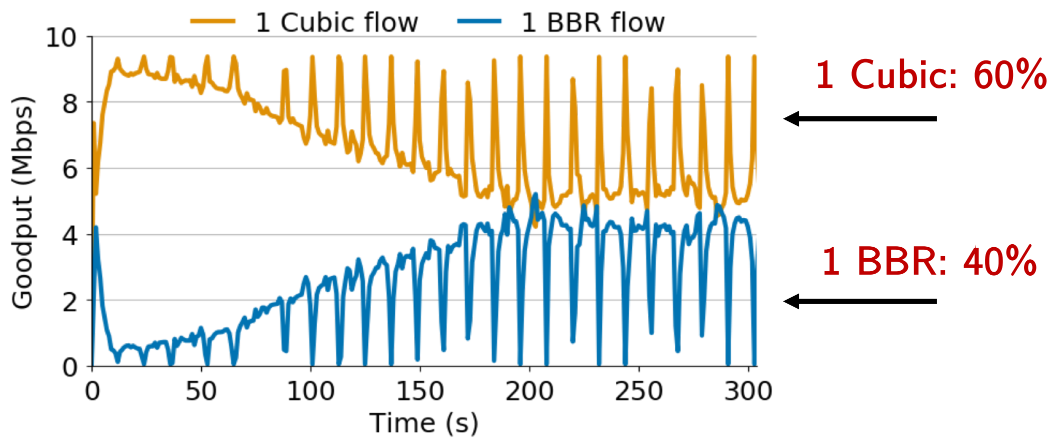


Figure: 1 BBR vs. 1 Cubic.
(10 Mbps network, 32 BDP queue)

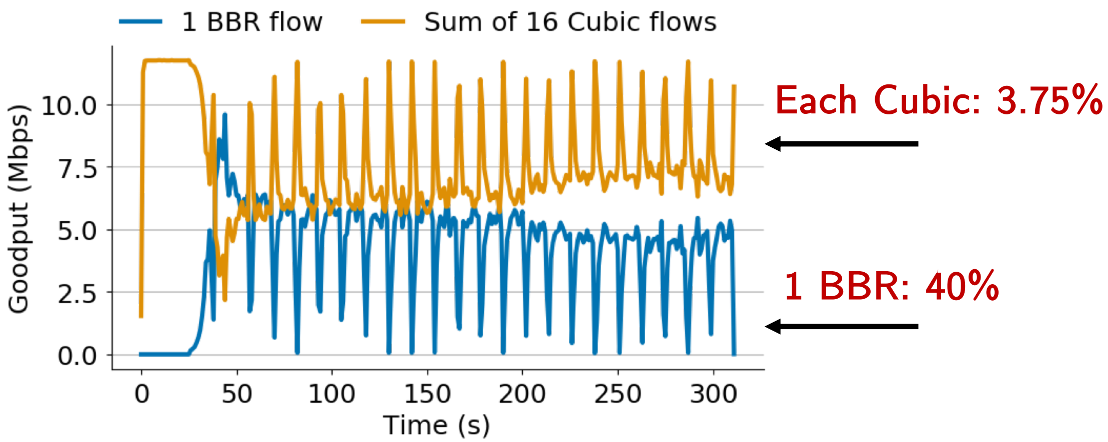


Figure: 1 BBR vs. 16 Cubic.
(10 Mbps network, 32 BDP queue)

How can we explain these results?

We can use modeling to understand an algorithm's behavior.

Mathis equation for TCP Reno's throughput

$$BW < \left(\frac{MSS}{RTT}\right) \frac{1}{\sqrt{p}}$$

Padhye equation for TCP Reno's throughput

$$B(p) \approx \min\left(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}\right)$$

Can we build a model to understand BBR's interactions with loss-based algorithms?

Mathis equation for TCP Reno's throughput

$$BW < \left(\frac{MSS}{RTT}\right)\frac{1}{\sqrt{p}}$$

Padhye equation for TCP Reno's throughput

$$B(p) \approx \min\left(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}\right)$$

Our equation for BBR's throughput

Can we build a model?

Can we build a model to understand BBR's interactions with loss-based algorithms? Yes!

Mathis equation for TCP Reno's throughput

$$BW < \left(\frac{MSS}{RTT}\right)\frac{1}{\sqrt{p}}$$

Padhye equation for TCP Reno's throughput

$$B(p) \approx \min\left(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}\right)$$

Our equation for BBR's throughput

$$BBR_{frac} = \left(1 - \frac{1}{2} + \frac{1}{2X} + \frac{4N}{q}\right) \times \left(1 - \left(\frac{q}{c} + .2 + l\right) \times \frac{1}{10}\right)$$

Our model shows BBR's throughput does not depend on the number of competing loss-based flows.

Mathis equation for TCP Reno's throughput

$$BW < \left(\frac{MSS}{RTT}\right)\frac{1}{\sqrt{p}}$$

Padhye equation for TCP Reno's throughput

$$B(p) \approx \min\left(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p(1+32p^2)}\right)$$

Our equation for BBR's throughput

$$BBR_{frac} = \left(1 - \frac{1}{2} + \frac{1}{2X} + \frac{4N}{q}\right) \times \left(1 - \left(\frac{q}{c} + .2 + l\right) \times \frac{1}{10}\right)$$

None of these variables depend on the number of loss-based flows!

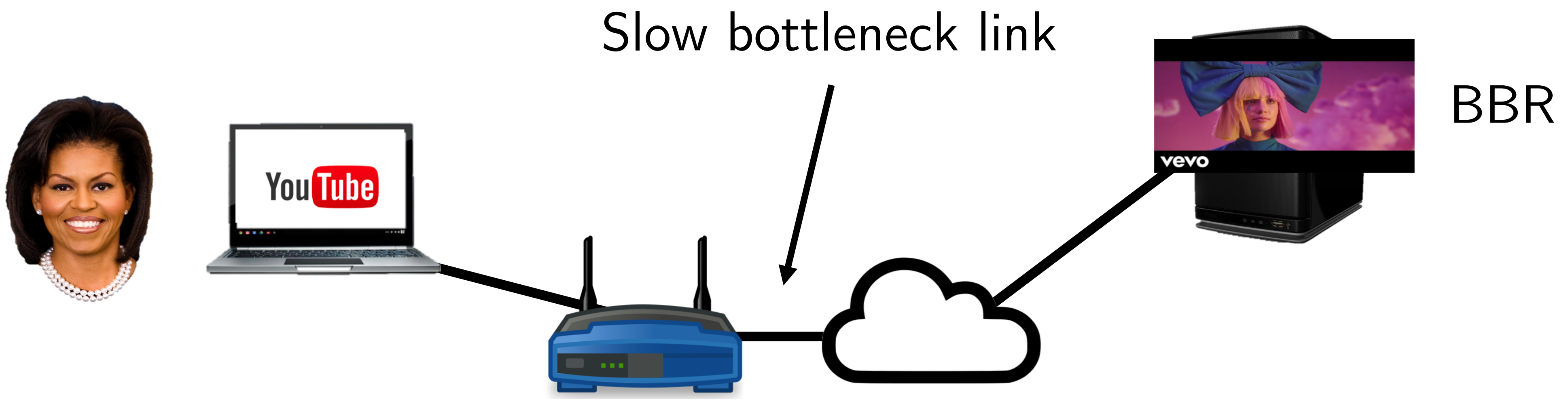
BBR is a **rate-based algorithm**.
How does BBR figure out sending rate?

BBR is a **rate-based algorithm**.
How does BBR figure out sending rate? **ProbeBW**

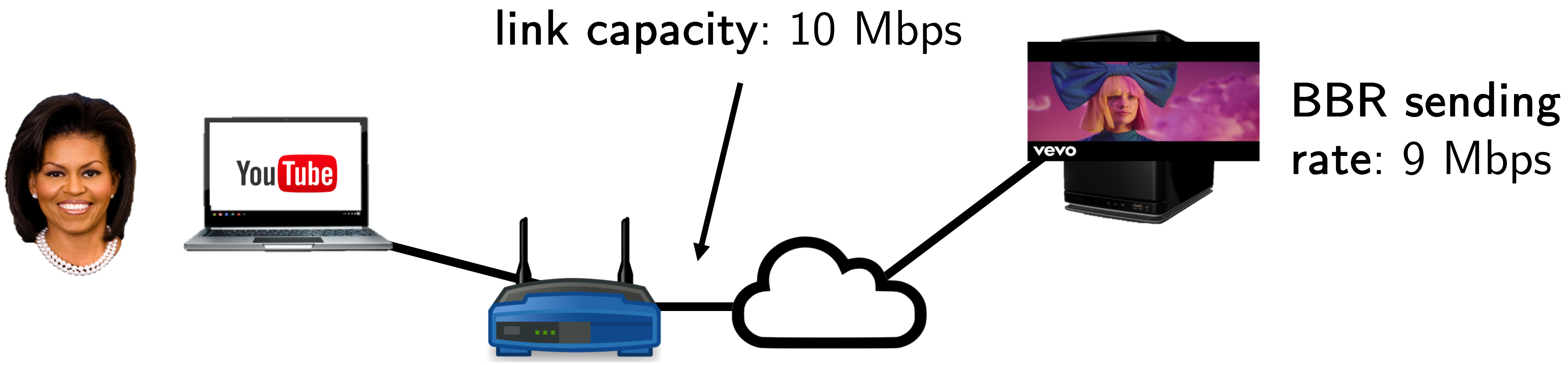
- Send at rate r - 6 RTTs
- Sent at rate 1.25r. - 1 RTT
- Reduce to new sending rate (Drain) - 1 RTT

BBR is a **rate-based algorithm**.

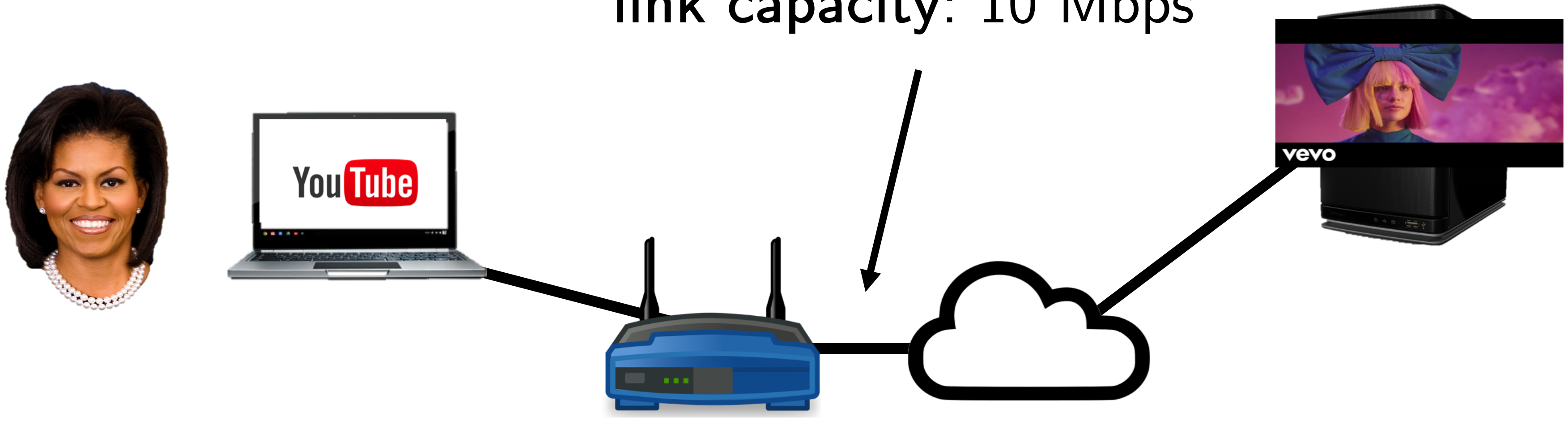
How does BBR figure out sending rate? **ProbeBW**



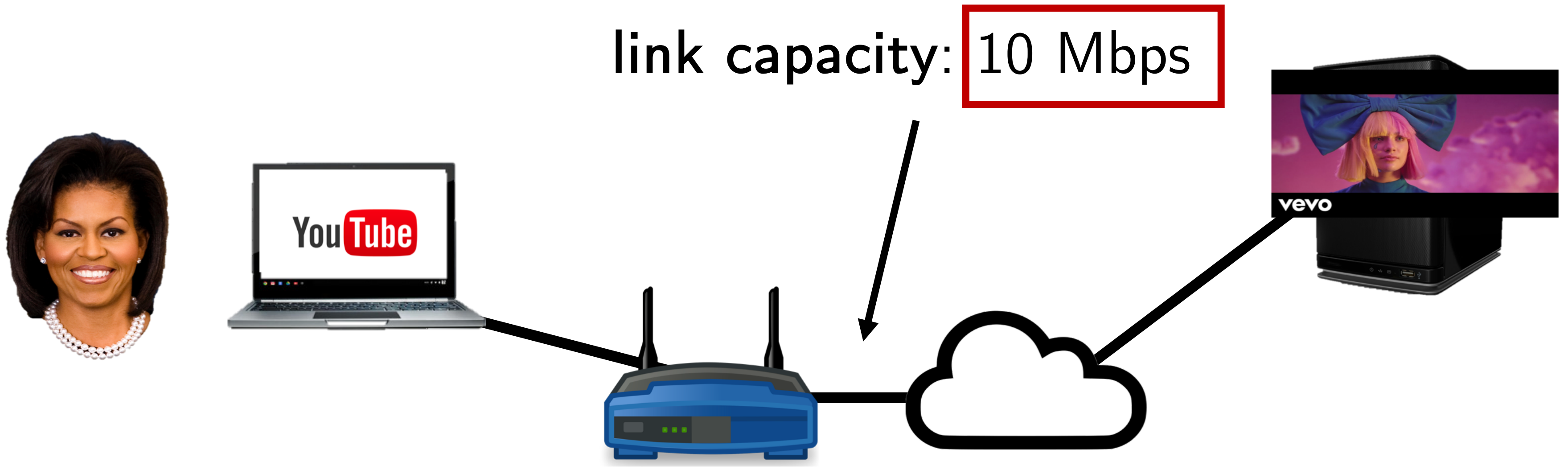
BBR bandwidth estimate is the largest throughput it has seen over an 8 RTT window.



During ProbeBW, BBR increases its sending rate by 25% to see if it can get more throughput.



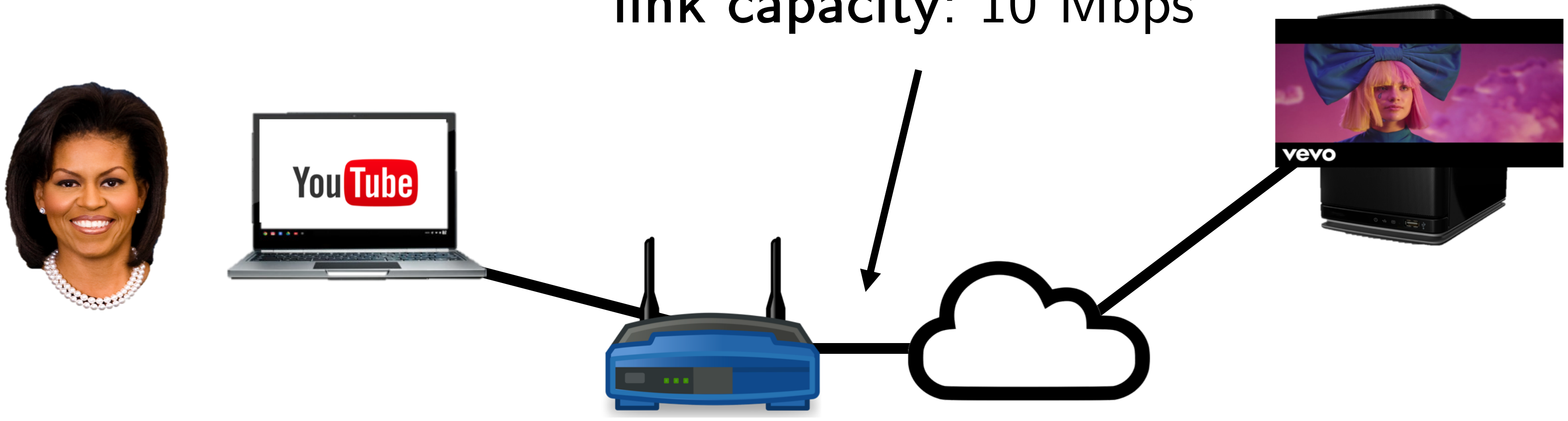
During ProbeBW, BBR increases its sending rate by 25% to see if it can get more throughput.



BBR sending rate: ~~9 Mbps~~
1.25 x 9 Mbps
= 11.25 Mbps

During ProbeBW, BBR increases its sending rate by 25% to see if it can get more throughput.

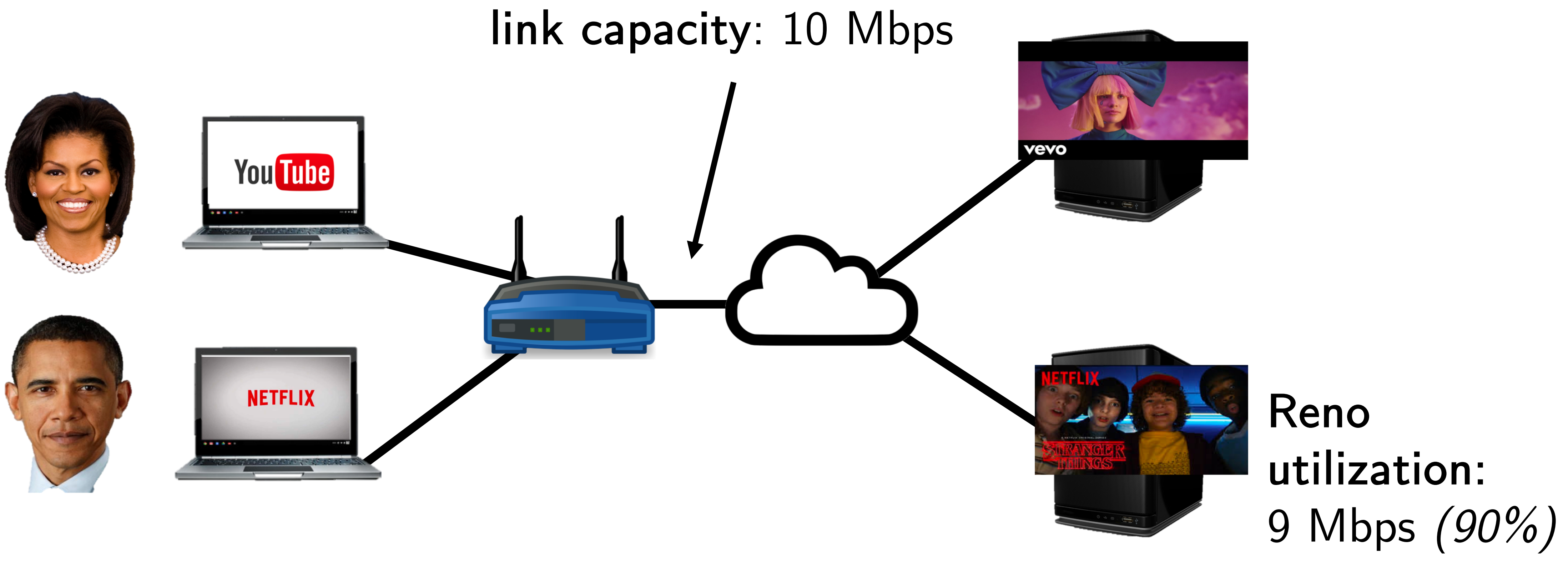
link capacity: 10 Mbps



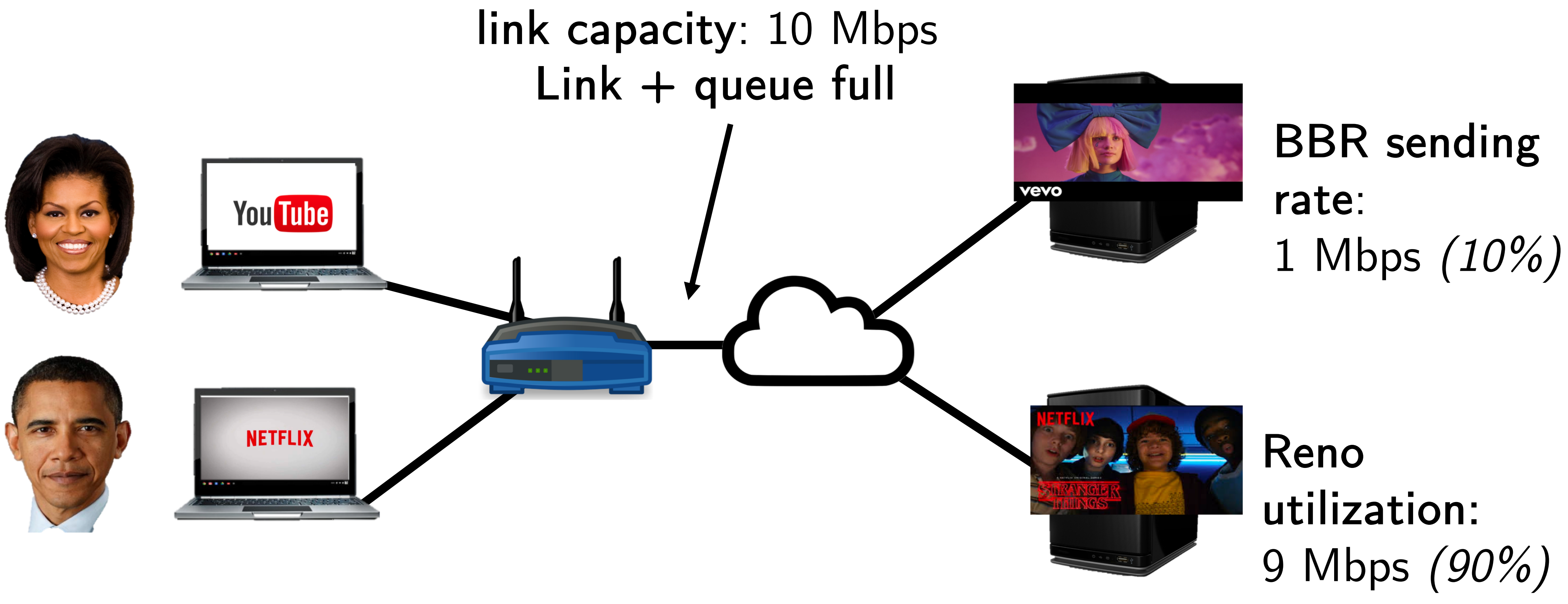
BBR sending rate: ~~9 Mbps~~
~~1.25 x 9 Mbps~~
10 Mbps

Send at this new rate for 6 RTTs

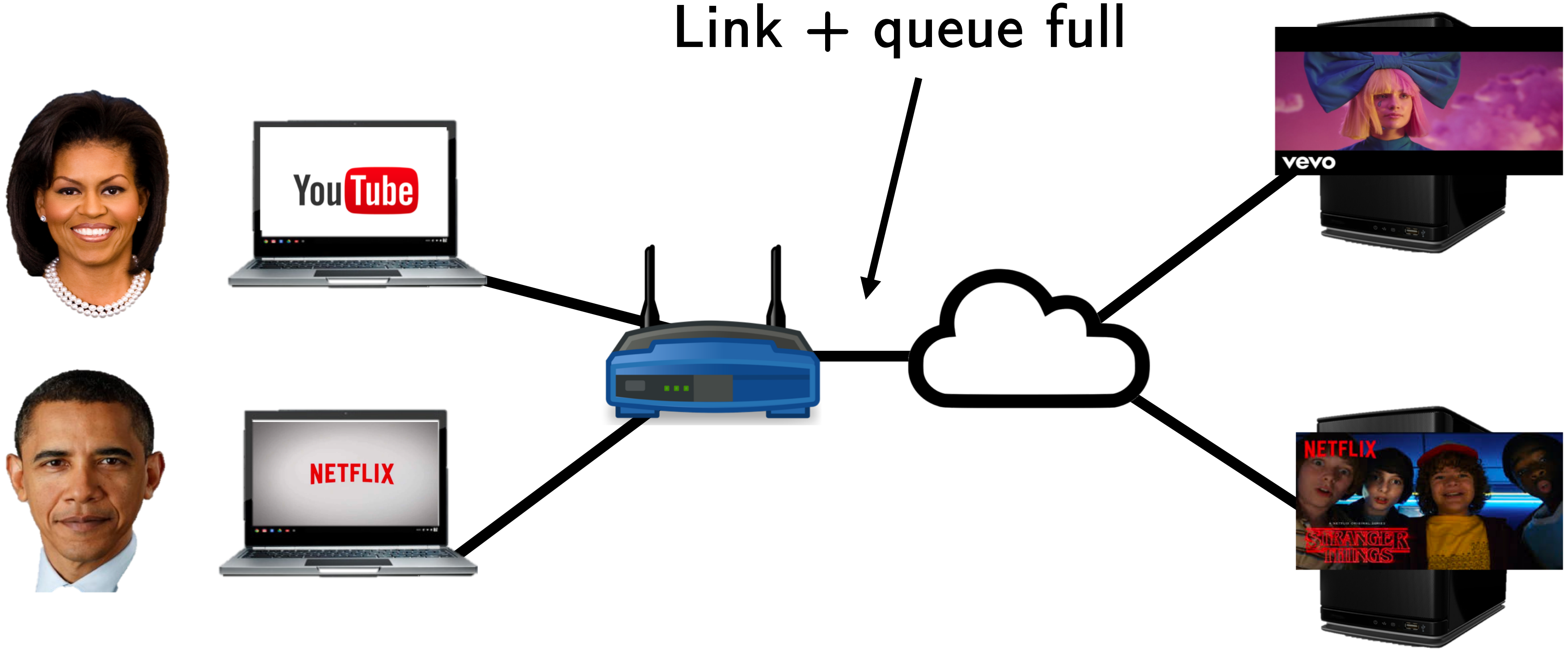
What happens during ProbeBW when competing with Reno or Cubic?



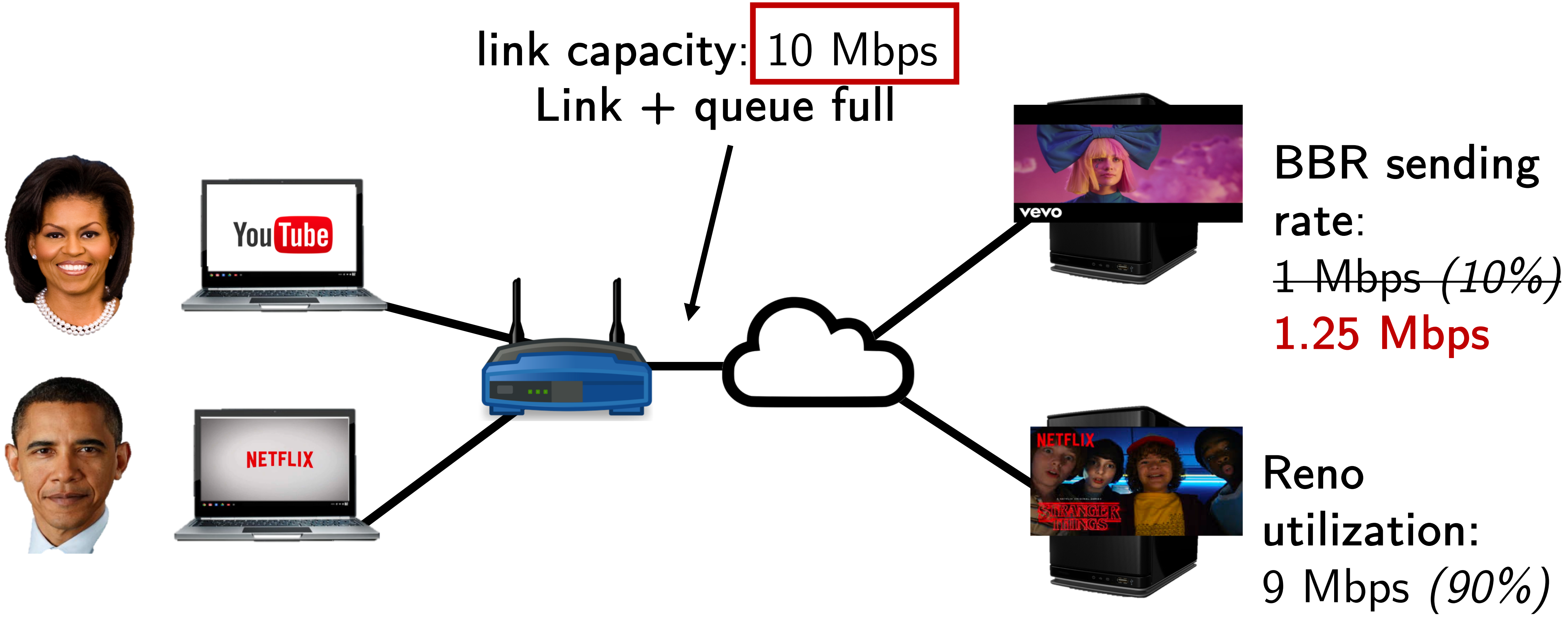
What happens during ProbeBW when competing with Reno or Cubic?



During ProbeBW, BBR will cause packet loss.



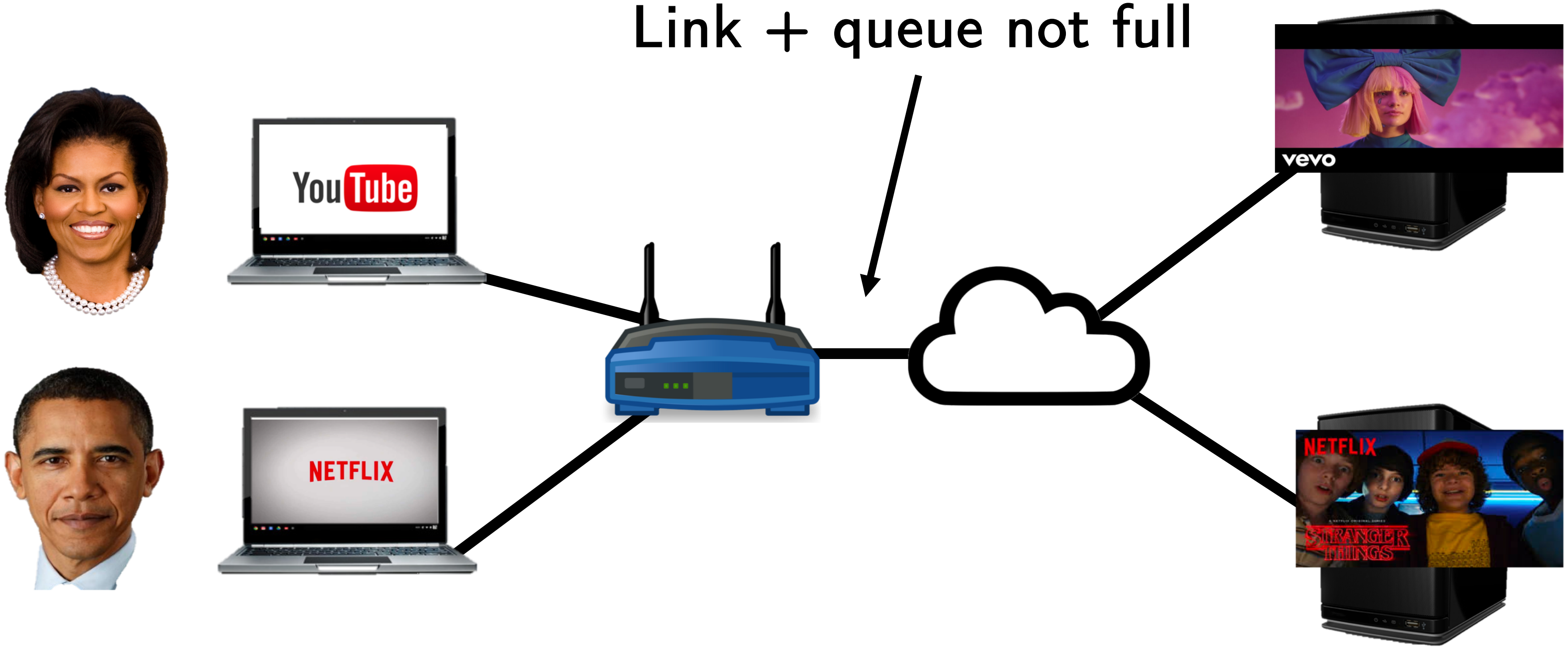
During ProbeBW, BBR will cause packet loss.



During ProbeBW, BBR will cause packet loss.

link capacity: 10 Mbps

Link + queue not full



BBR sending rate:

~~1 Mbps *(10%)*~~

~~1.25 Mbps~~

1.21 Mbps(12%)

Reno utilization:

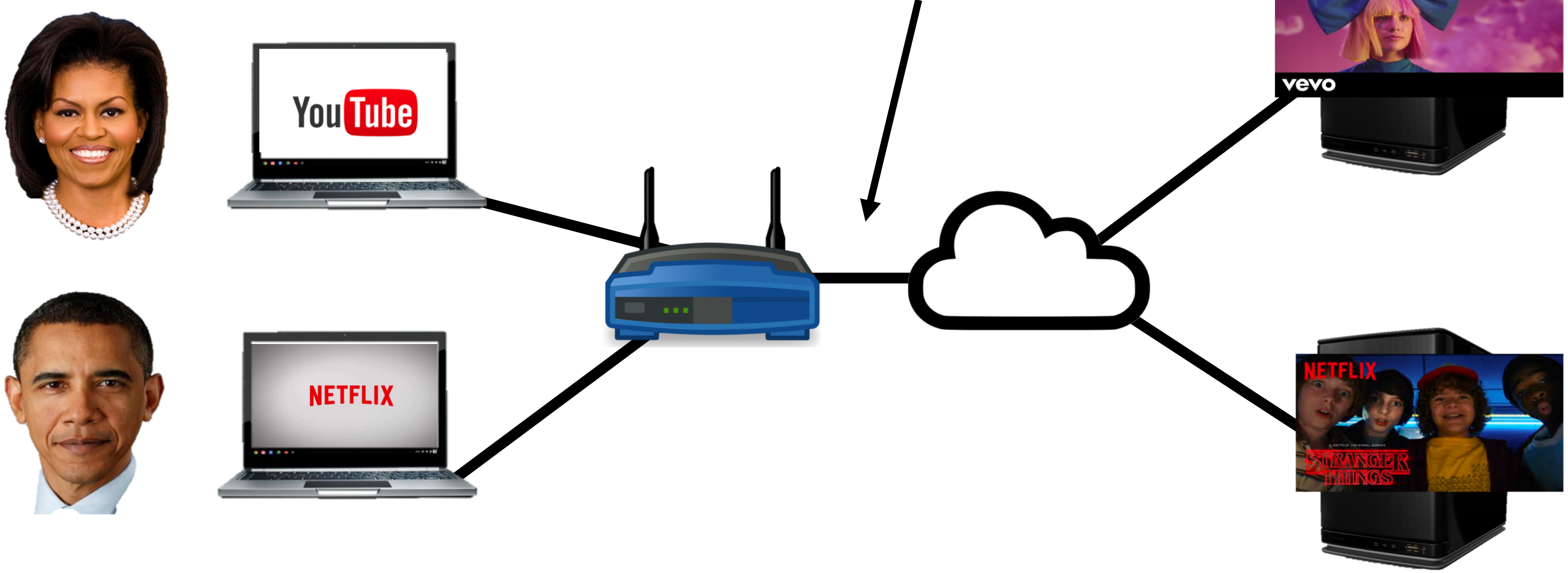
~~9 Mbps *(90%)*~~

4.5 Mbps (45%)

BBR will **increase** its steady-state sending rate while loss-based flows will back off.

link capacity: 10 Mbps

Link + queue not full



BBR sending rate:

~~1 Mbps *(10%)*~~

~~1.25 Mbps~~

1.21 Mbps(12%)

Reno utilization:

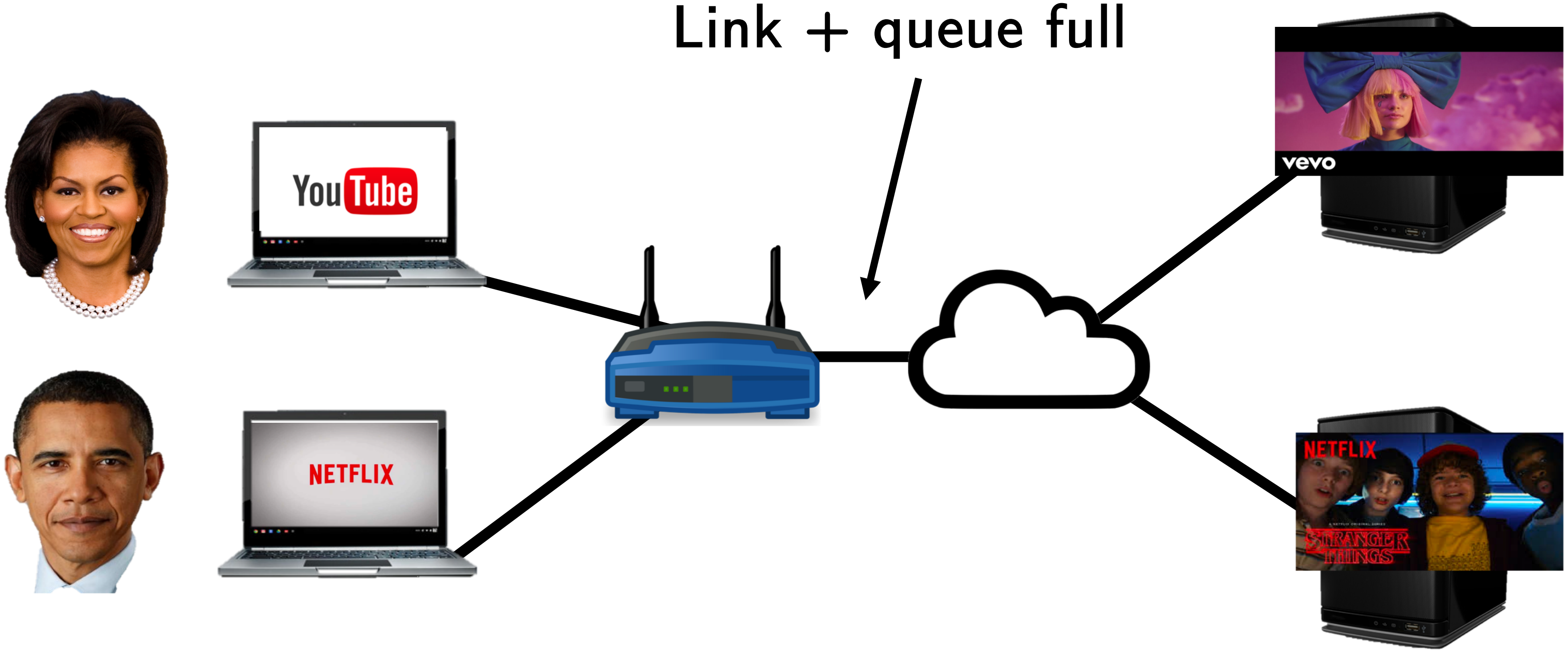
~~9 Mbps *(90%)*~~

4.5 Mbps (45%)

BBR's new rate is 1.21 Mbps

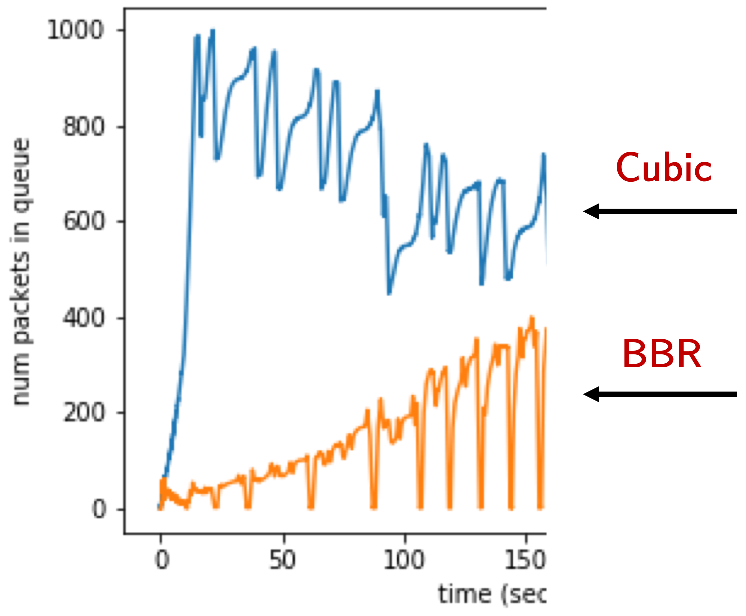
Cubic and Reno cannot return to their former throughput.

link capacity: 10 Mbps
Link + queue full

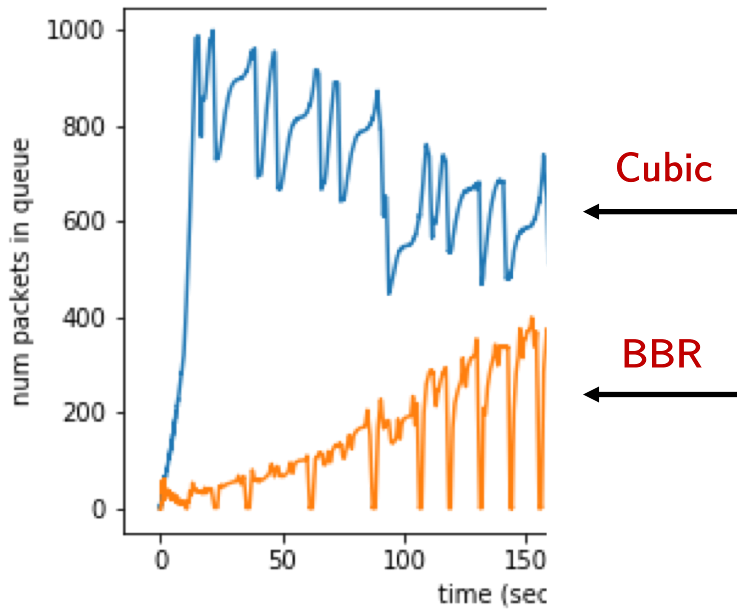


BBR sending rate:
~~1 Mbps *(10%)*~~
~~1.25 Mbps~~
1.21 Mbps(12%)

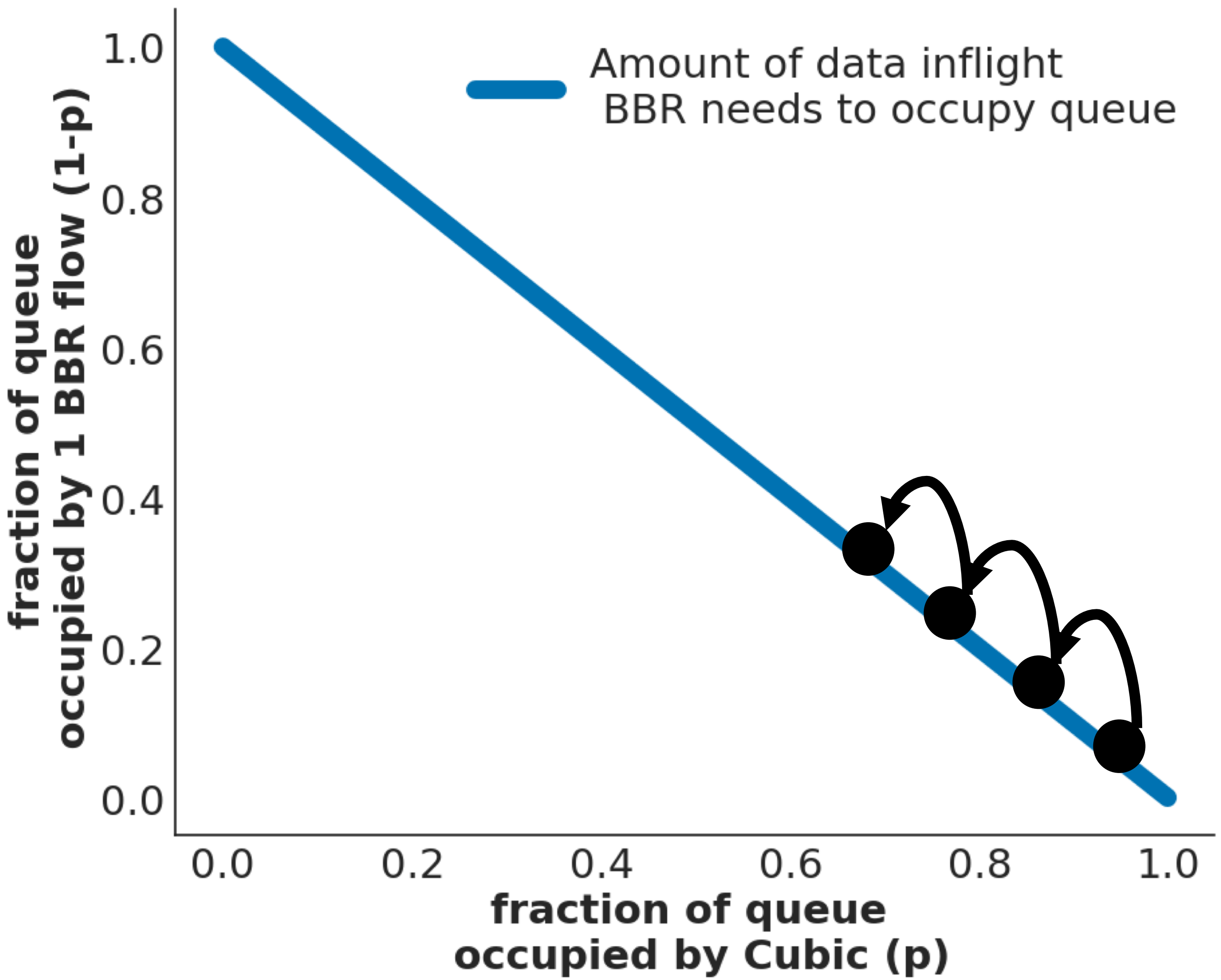
Reno utilization:
~~9 Mbps *(90%)*~~
~~4.5 Mbps (45%)~~
8.79 Mbps (88%)

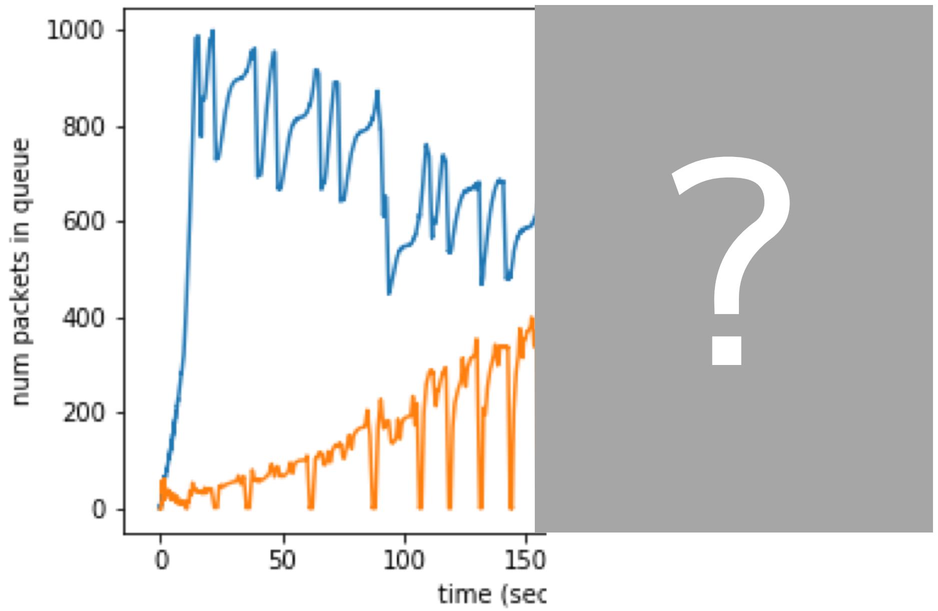


During ProbeBW, BBR will put more packets into the queue and will update its BW estimate.

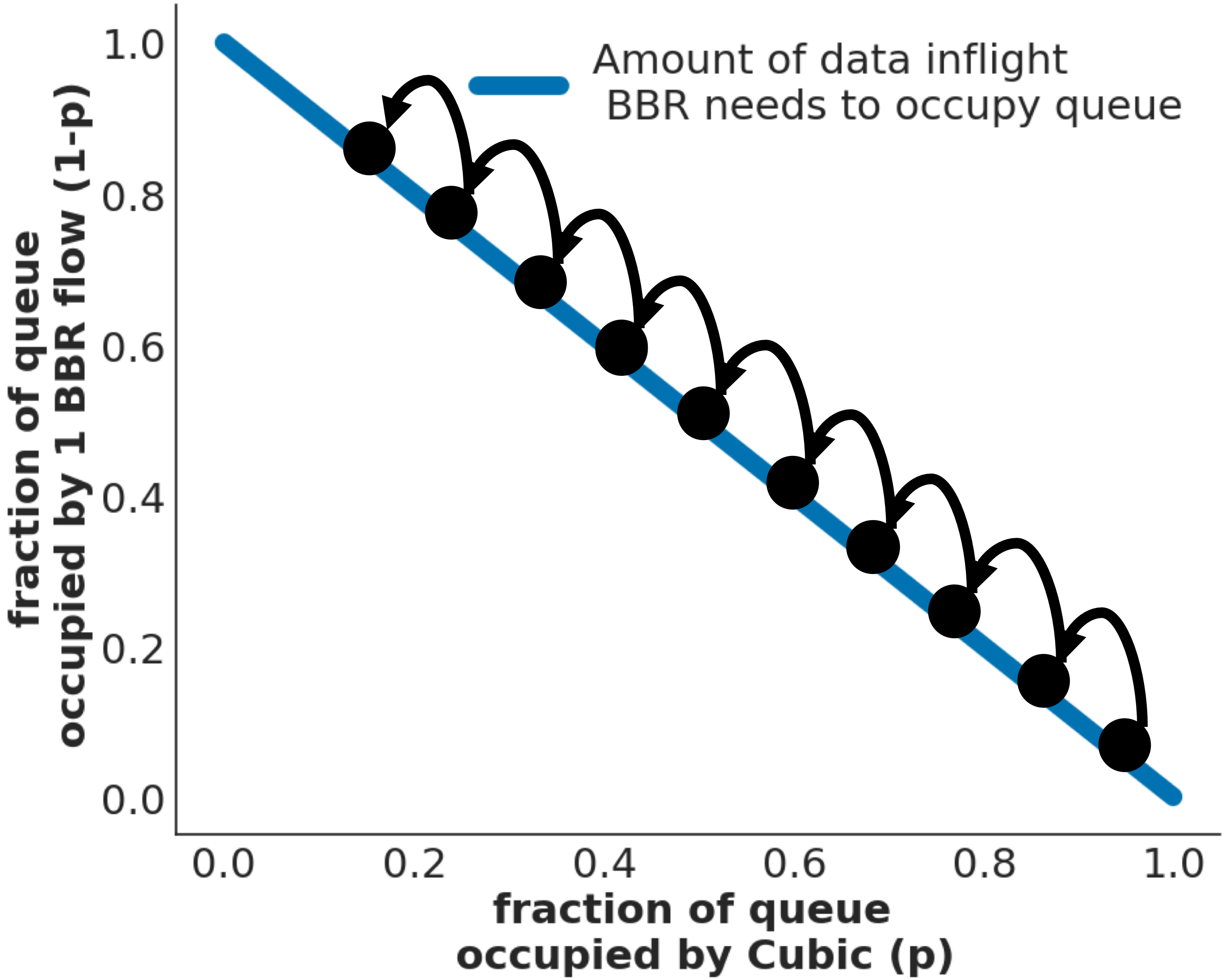


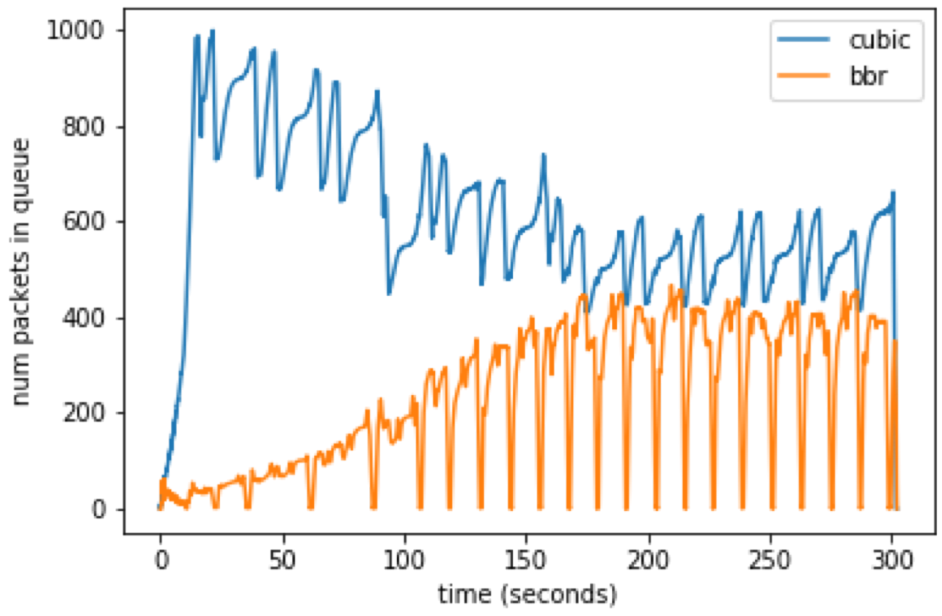
During ProbeBW, BBR will put more packets into the queue and will update its BW estimate.



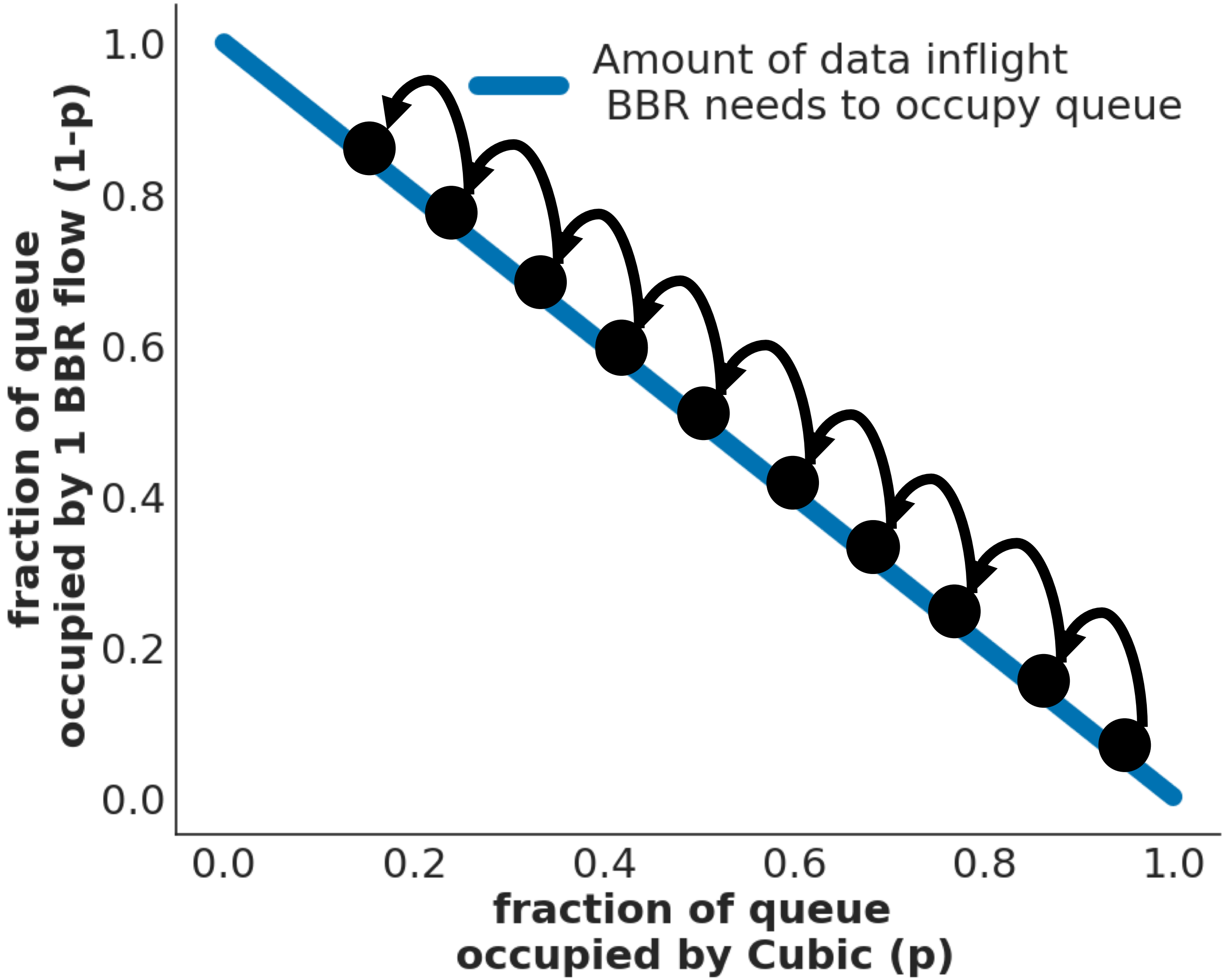


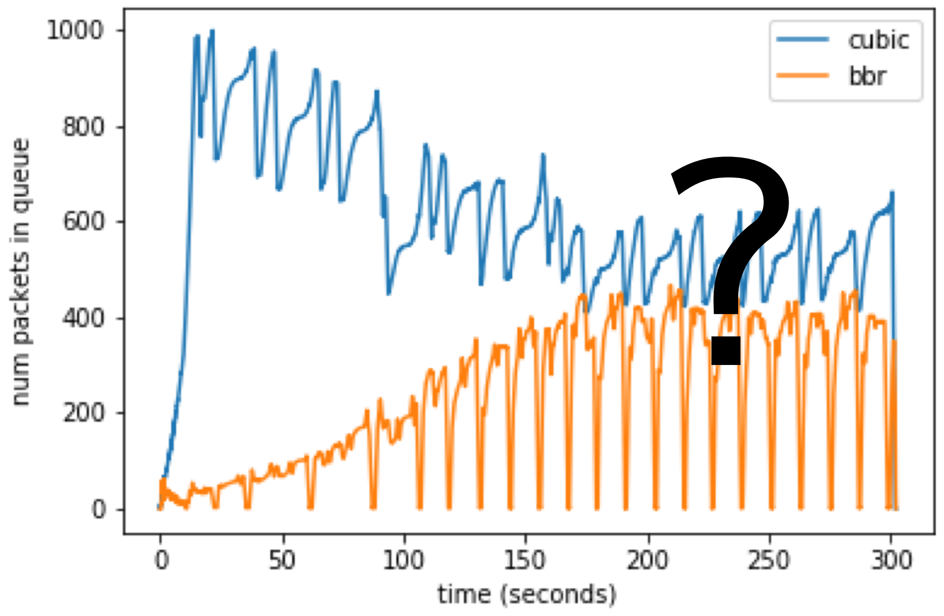
Shouldn't BBR just keep going into ProbeBW, putting more and more packets into the queue?



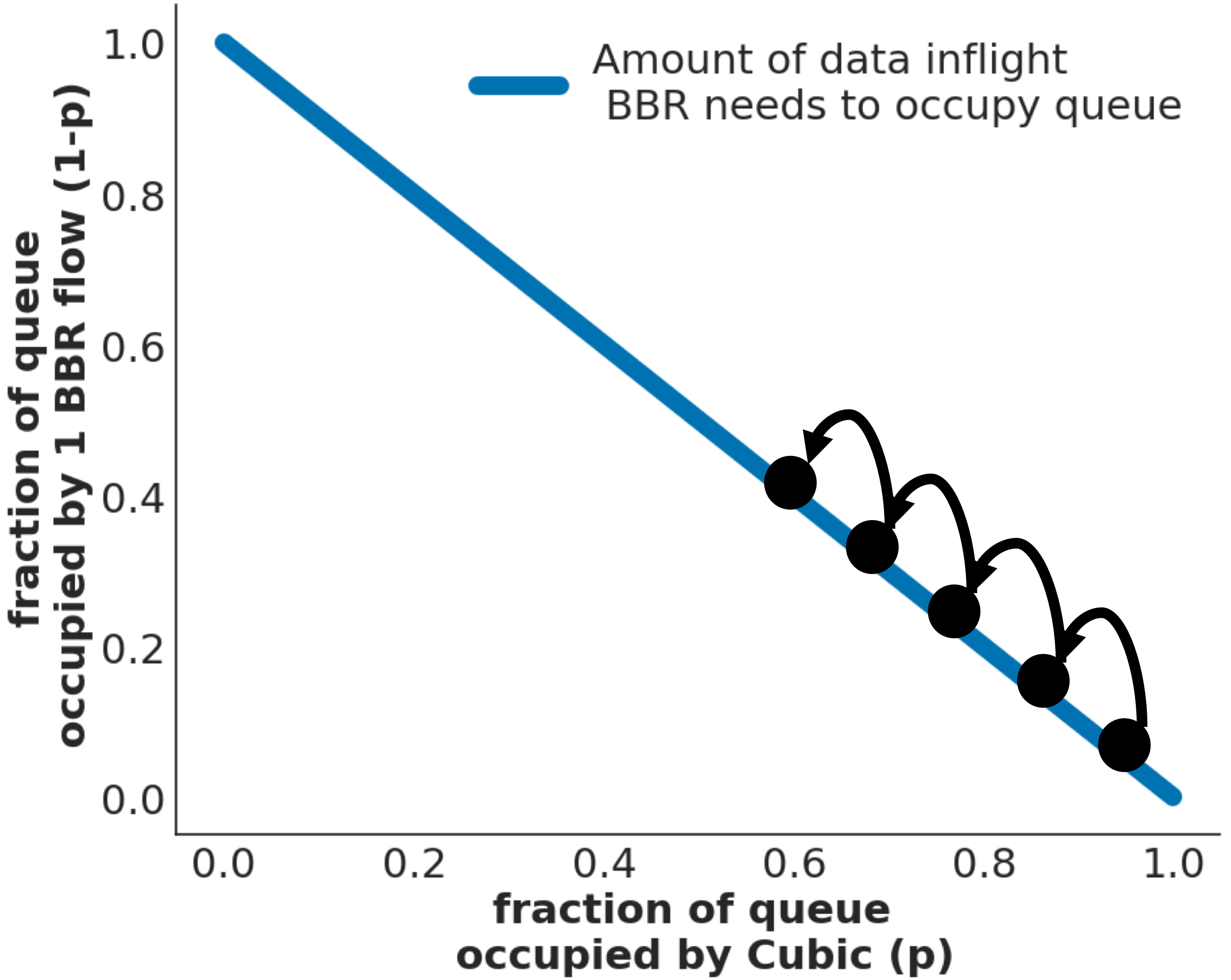


Why doesn't BBR keep putting more packets into the queue?





Why doesn't BBR keep putting more packets into the queue?



What is stopping ProbeBW from consuming the whole link?

One sentence in the BBR paper revealed the answer.

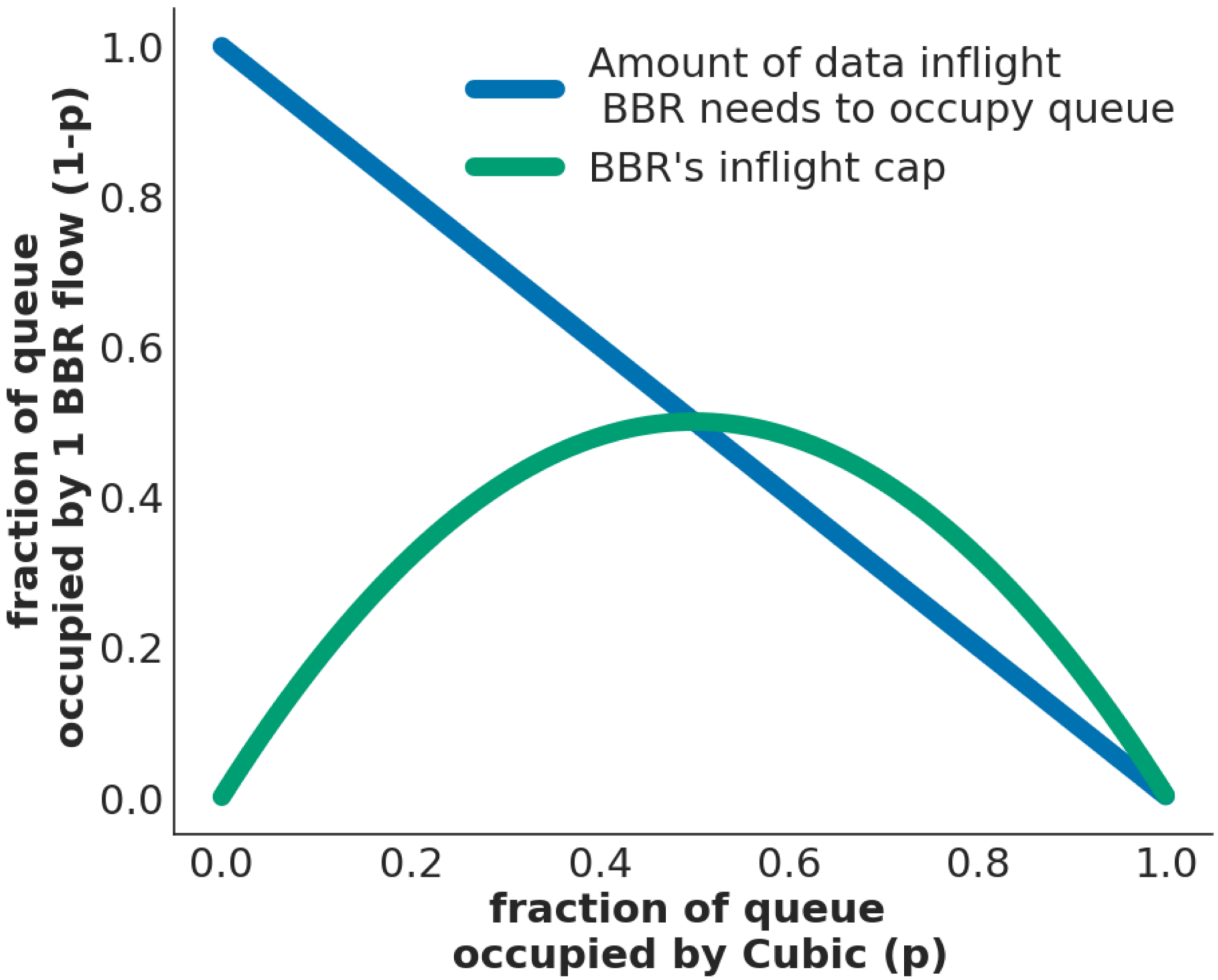
> DELAYED AND STRETCHED ACKS
>
> Cellular, Wi-Fi, and cable broadband networks often delay and aggregate ACKs.[1] When inflight is limited to one BDP, this results in throughput-reducing stalls. Raising ProbeBW's cwnd_gain to two allowed BBR to continue sending smoothly at the estimated delivery rate, even when ACKs are delayed by up to one RTT. This largely avoids stalls.

A safety mechanism dictates BBR's link fraction under competition.

Key Insight:
Under competition,
BBR is not rate-limited,
it is <u>window-limited</u>
due to the in-flight cap.

We need to model the **in-flight cap**.

We need to model the in-flight cap.



Assume that we have 1 BBR flow vs. 1 Cubic flow in a **deep-buffered network**.

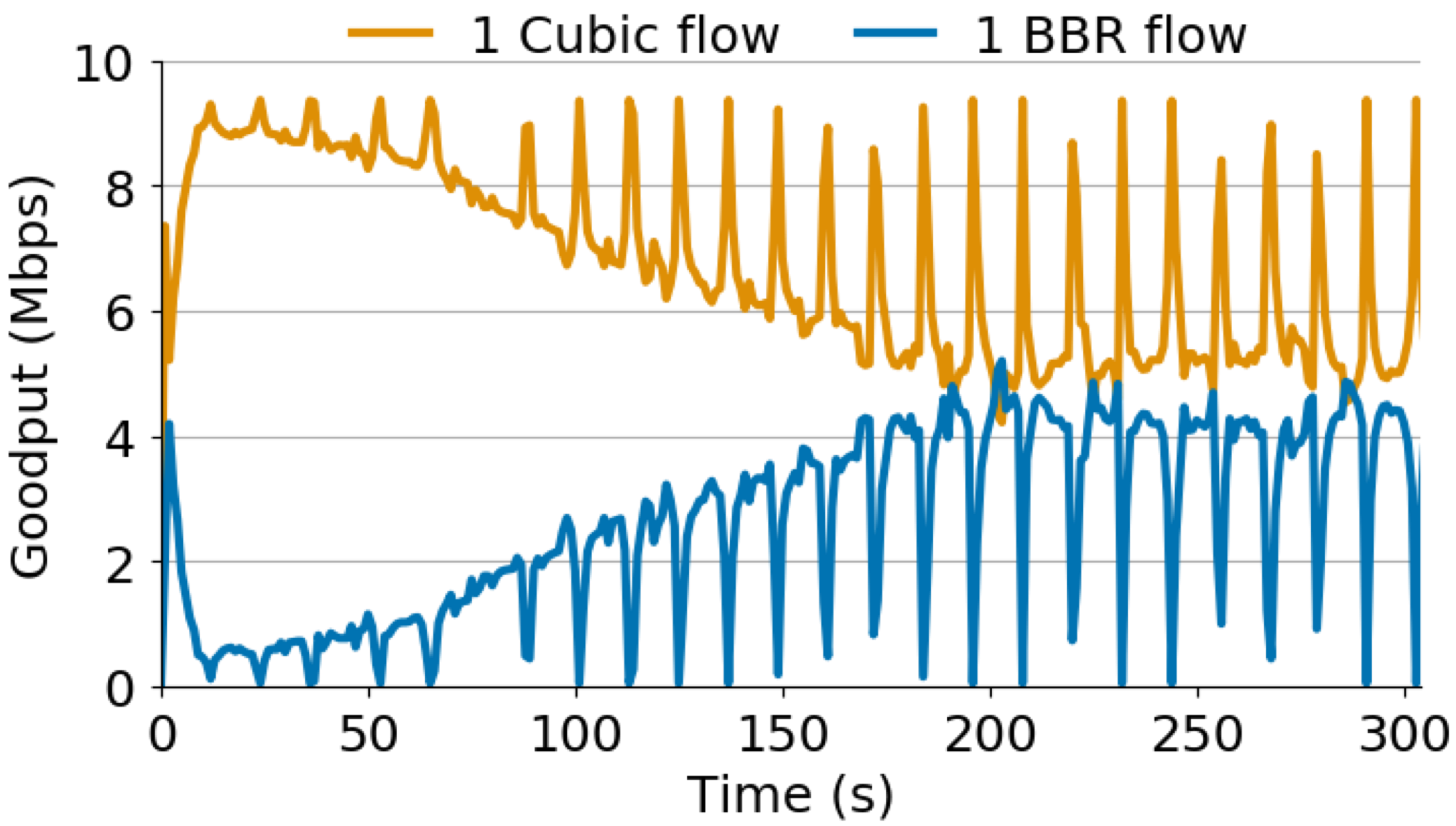
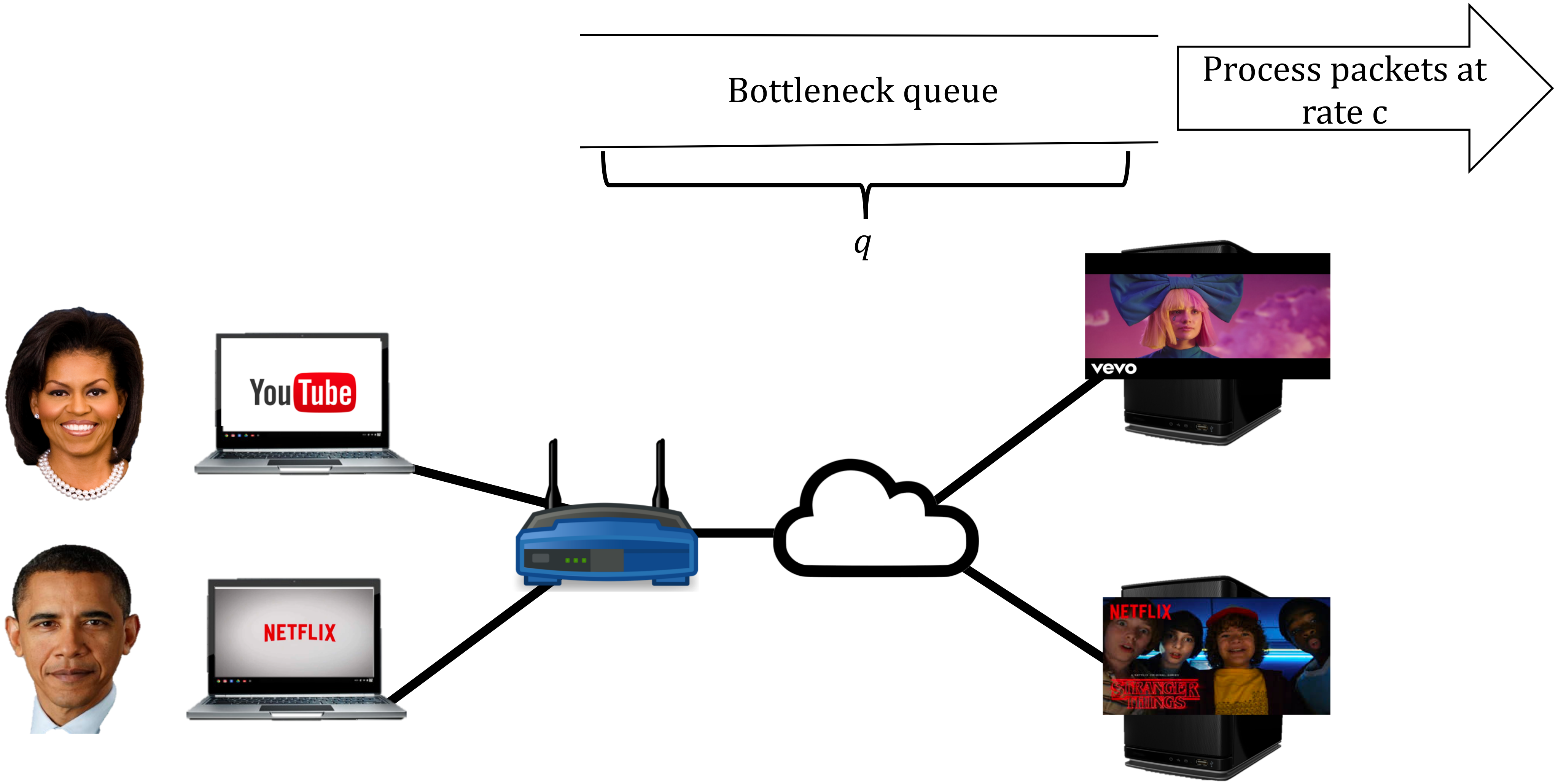


Figure: 1 BBR vs. 1 Cubic. (10 Mbps network, 32 BDP queue)

Assume:

c = btlnk link capacity

q = size of btlnk queue

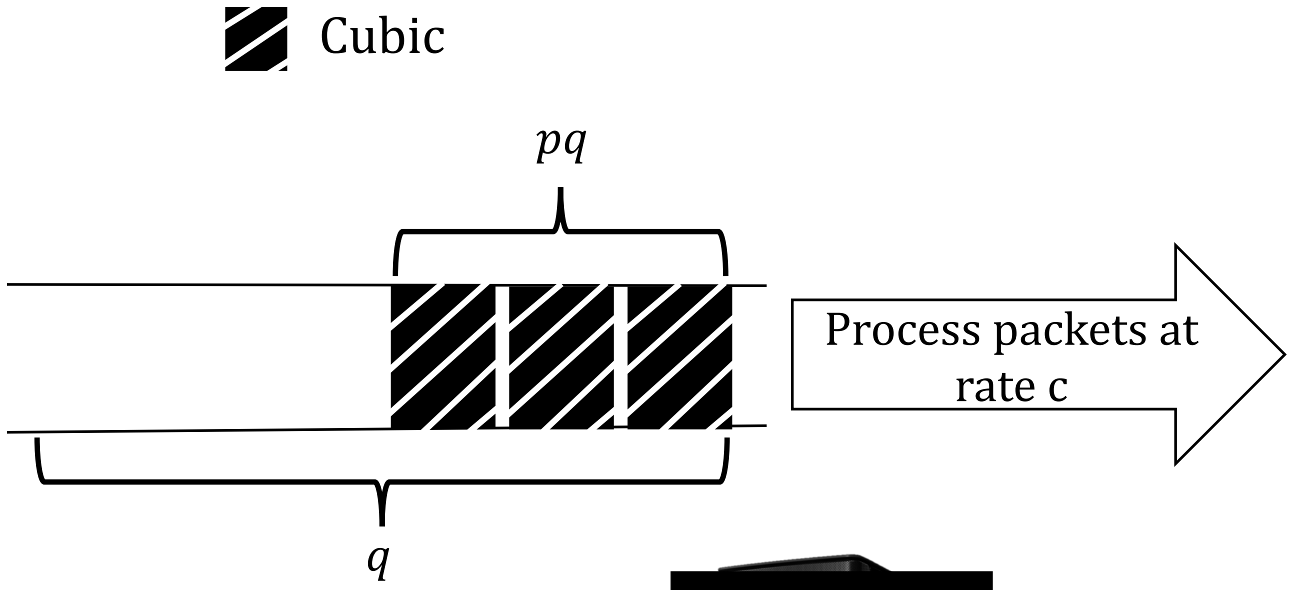


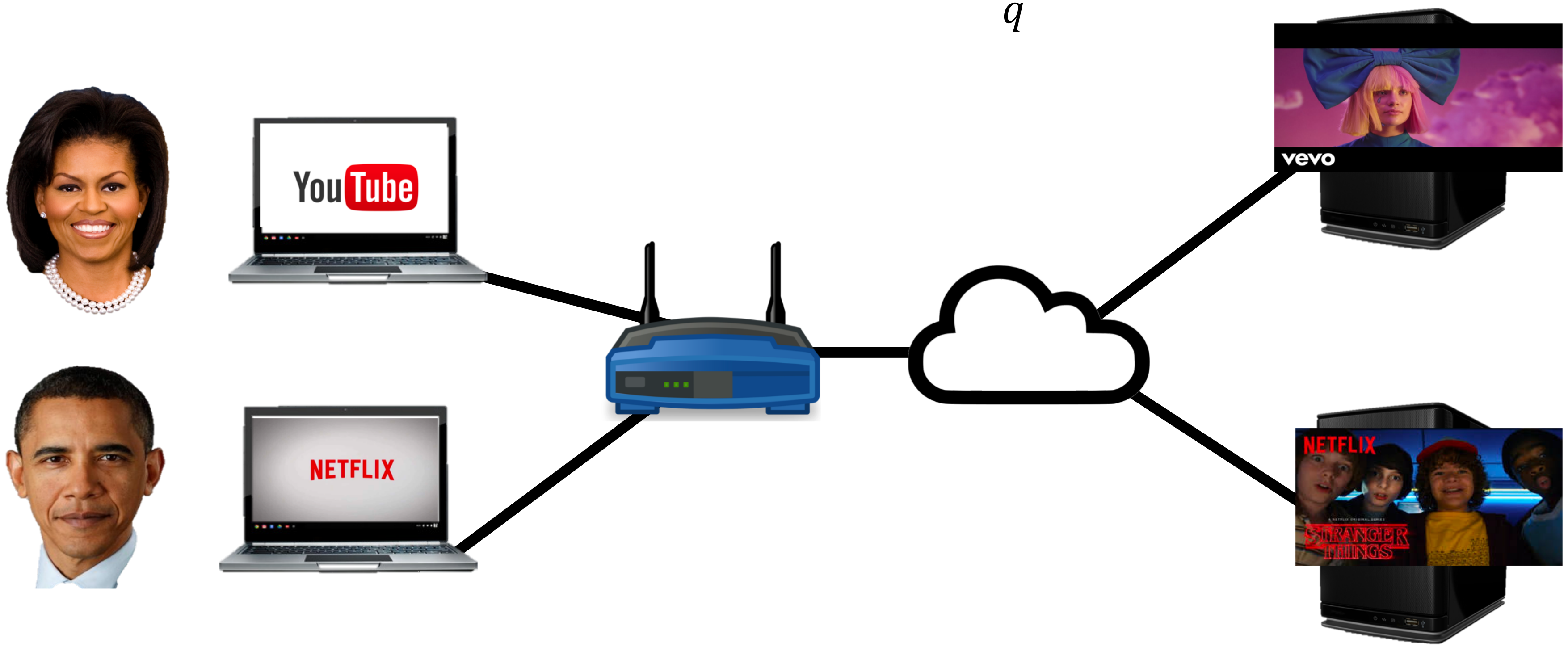
Assume:

c = btlnk link capacity

q = size of btlnk queue

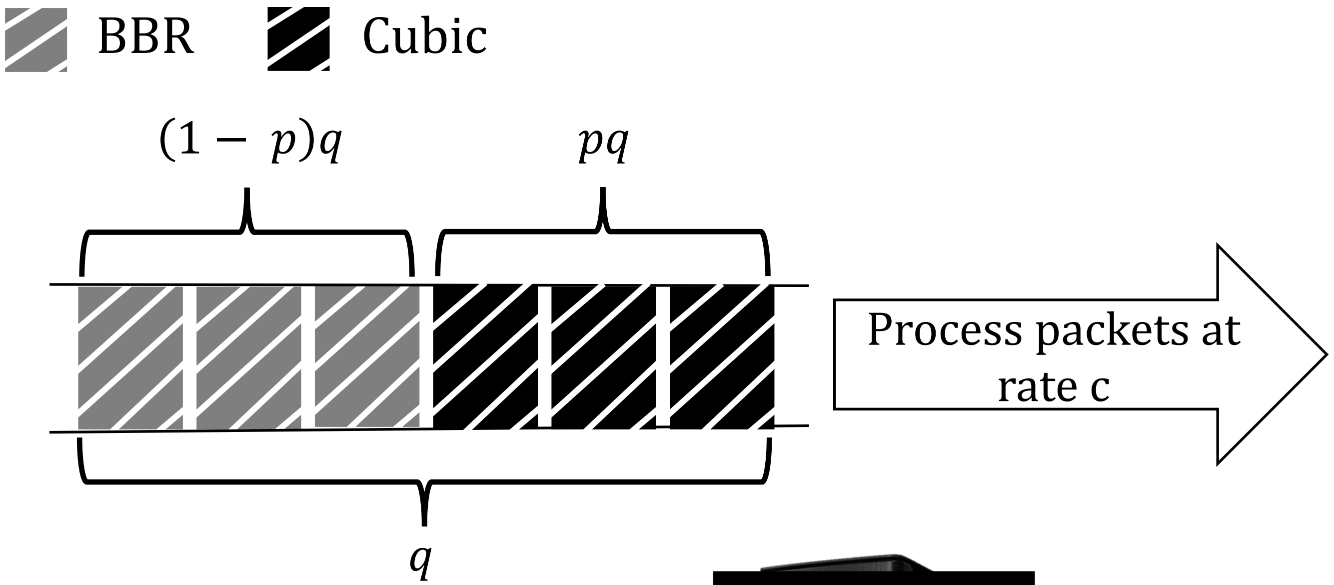
p = fraction of btlnk queue occupied by Cubic

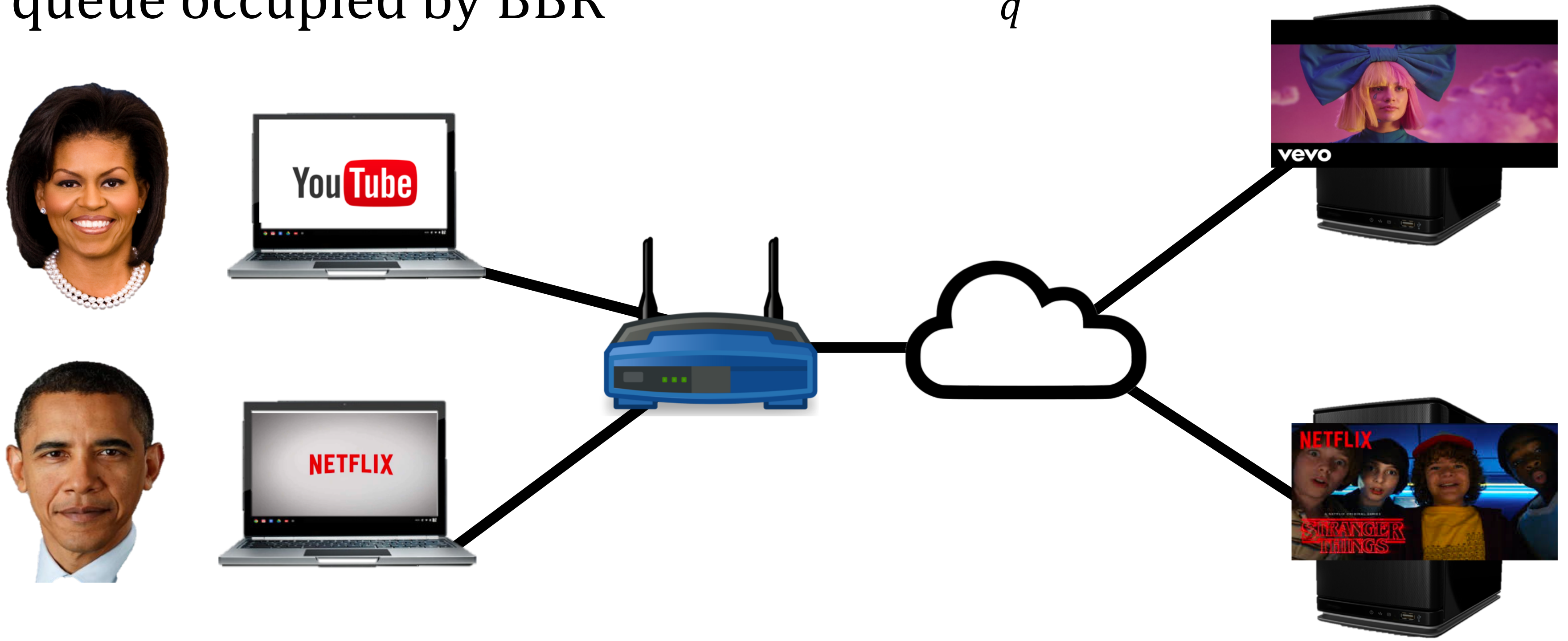




Assume:
c = btlnk link capacity
q = size of btlnk queue
p = fraction of btlnk queue occupied by Cubic

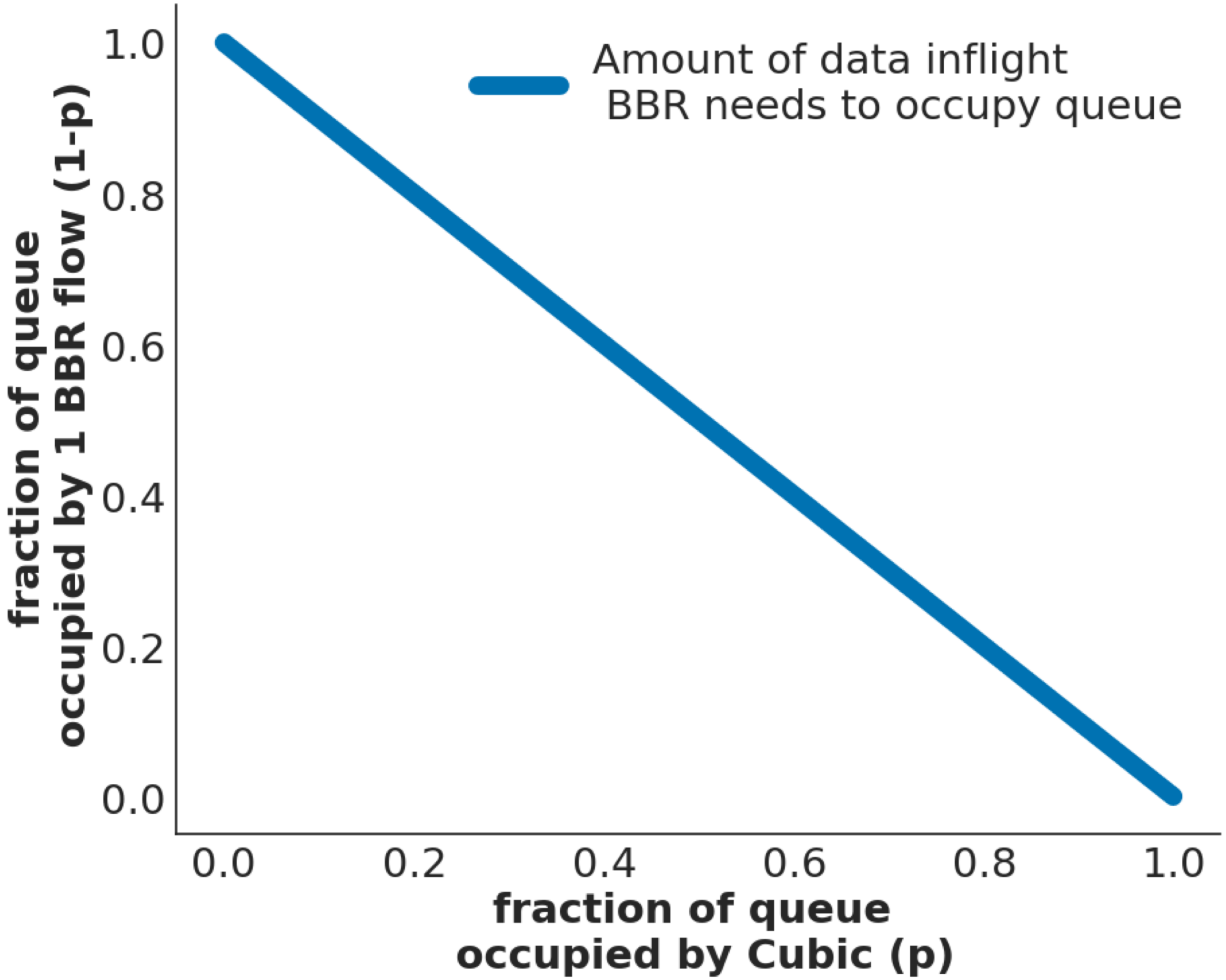
1- p = fraction of btlnk queue occupied by BBR





Assume:
c = btlnk link capacity
q = size of btlnk queue
p = fraction of btlnk queue occupied by Cubic

1- p = fraction of btlnk queue occupied by BBR

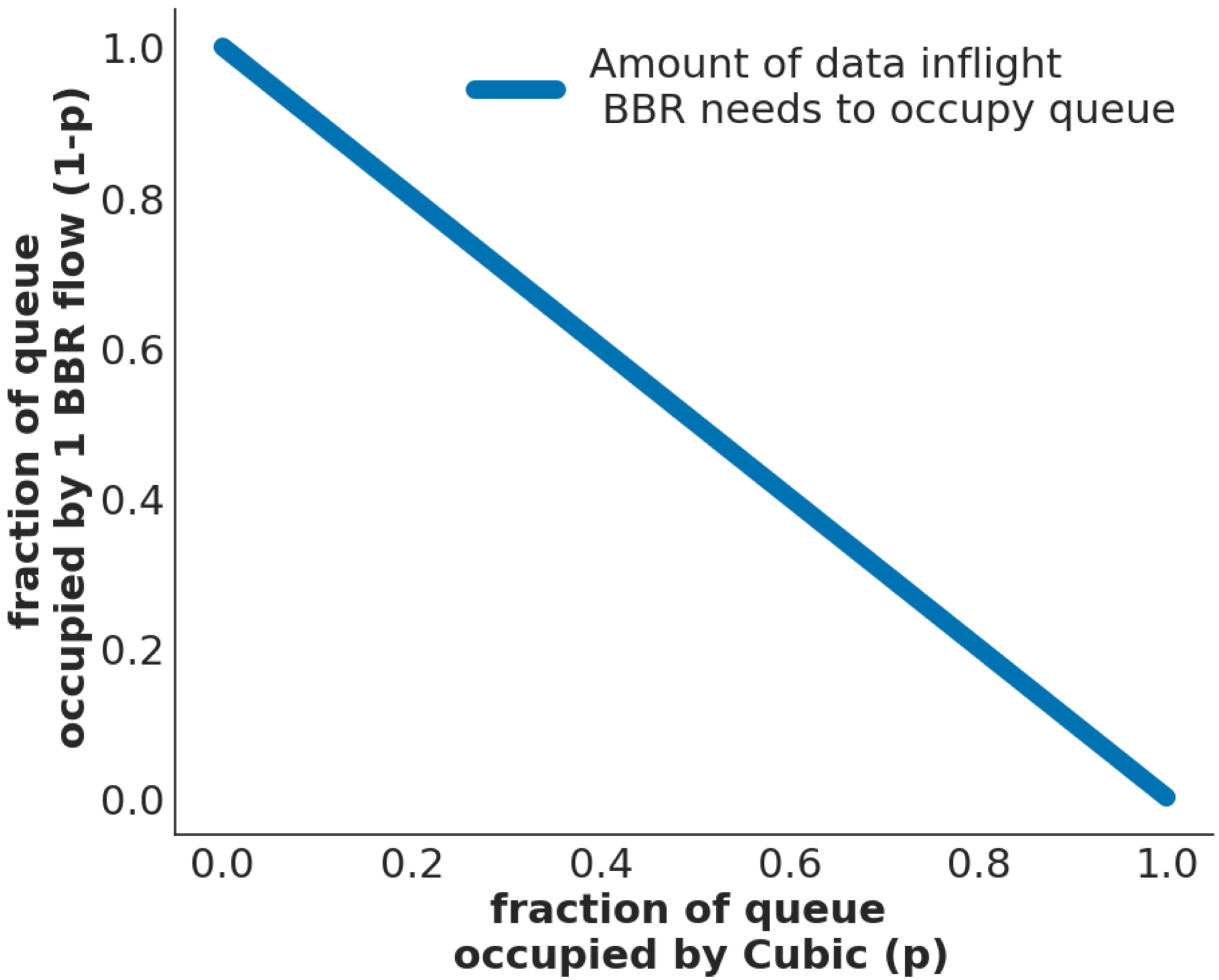


Assume:
c = btlnk link capacity
q = size of btlnk queue
p = fraction of btlnk queue occupied by Cubic

1- p = fraction of btlnk queue occupied by BBR

BBR inflight cap
$= 2 * BW * RTT$

$BW = (1 - \mathrm{p}) * \mathrm{c}$

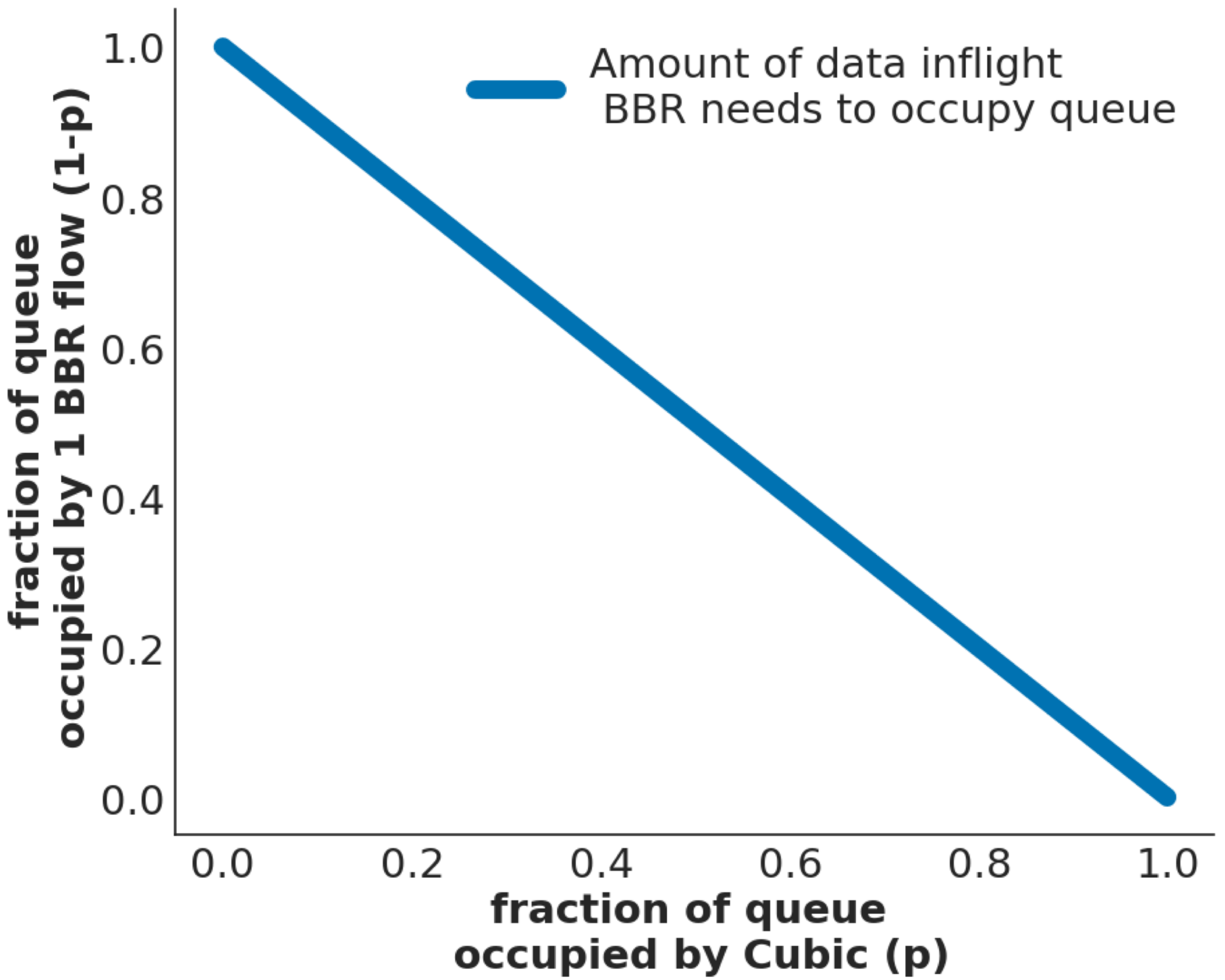


Assume:
c = btlnk link capacity
q = size of btlnk queue
p = fraction of btlnk queue occupied by Cubic

1- p = fraction of btlnk queue occupied by BBR

BBR inflight cap
$= 2 * BW * RTT$

$$BW = (1 - \mathrm{p}) * \mathrm{c}$$
$$RTT = \frac{p * q}{c}$$



Assume:
c = btlnk link capacity
q = size of btlnk queue
p = fraction of btlnk queue occupied by Cubic

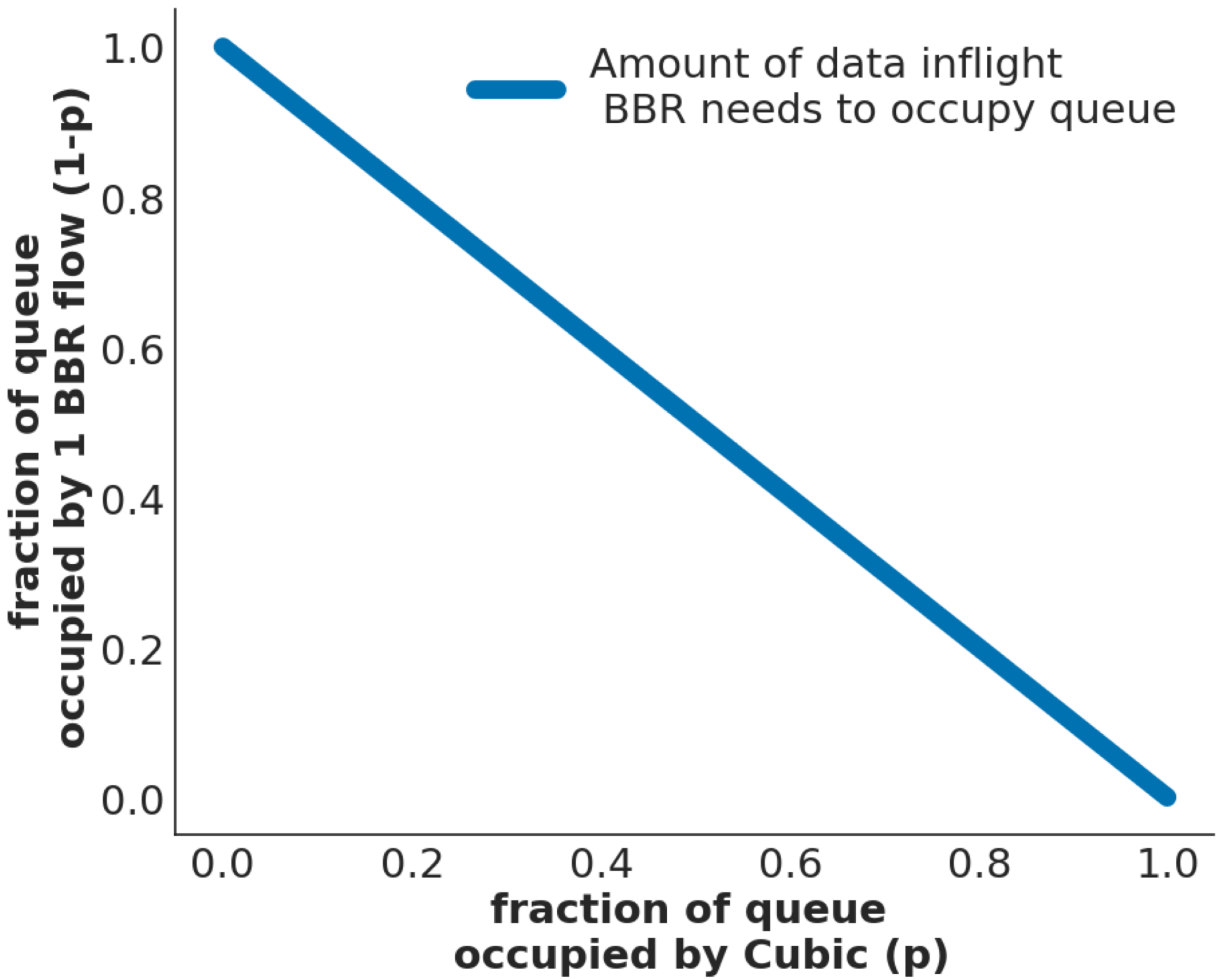
1- p = fraction of btlnk queue occupied by BBR

BBR inflight cap
$= 2 * BW * RTT$

$$BW = (1 - p) * c$$

$$RTT = \frac{p * q}{c}$$

$$\mathbf{= 2 * (p - p^2) * q}$$



Assume:

c = btlnk link capacity
q = size of btlnk queue
p = fraction of btlnk queue occupied by Cubic

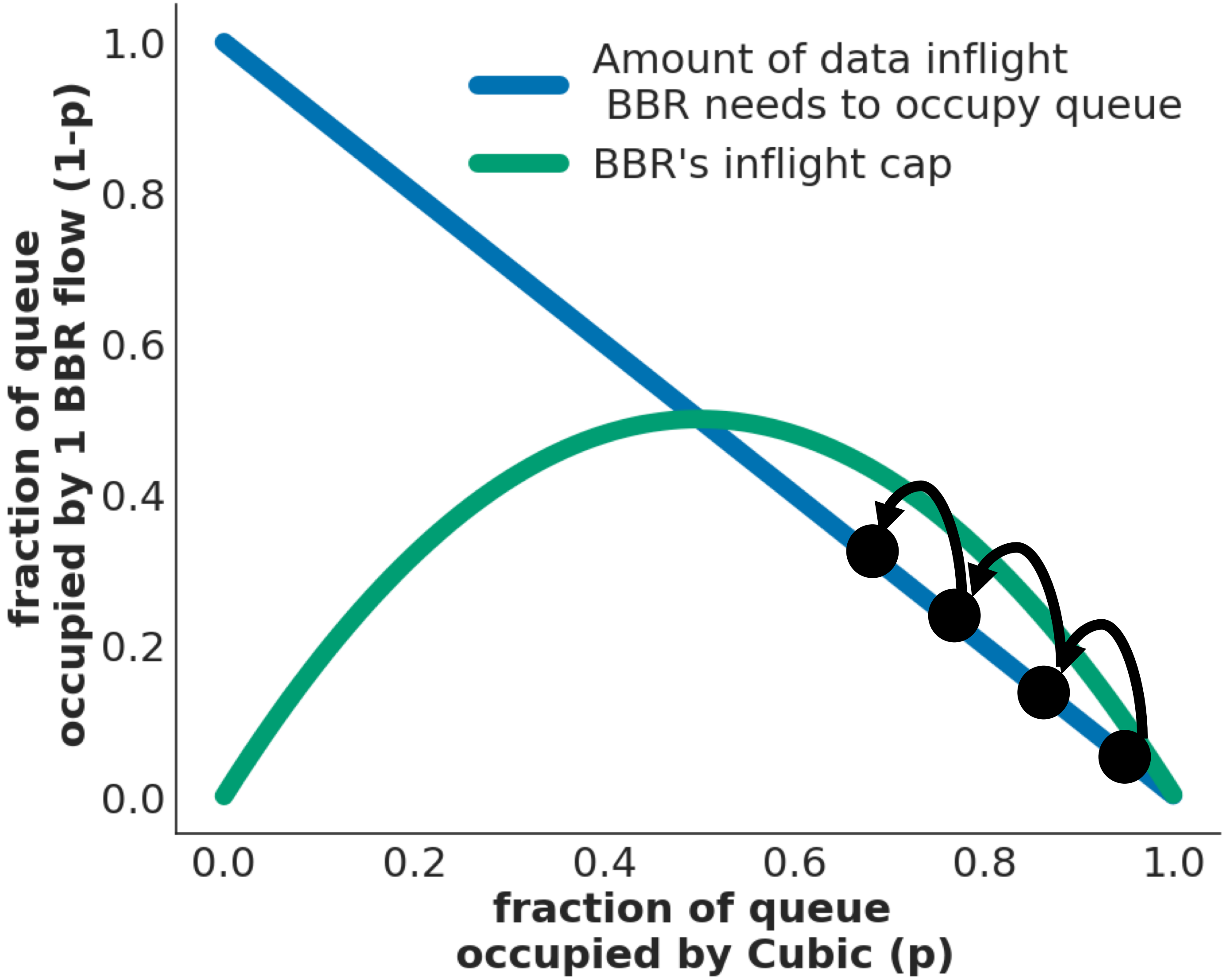
1- p = fraction of btlnk queue occupied by BBR

BBR inflight cap
$= 2 * BW * RTT$

$$BW = (1 - \mathrm{p}) * \mathrm{c}$$

$$RTT = \frac{p * q}{c}$$

$$= \mathbf{2 * (p - p^2) * q}$$



Assume:

c = btlnk link capacity

q = size of btlnk queue

p = fraction of btlnk queue occupied by Cubic

$1 - p$ = fraction of btlnk queue occupied by BBR

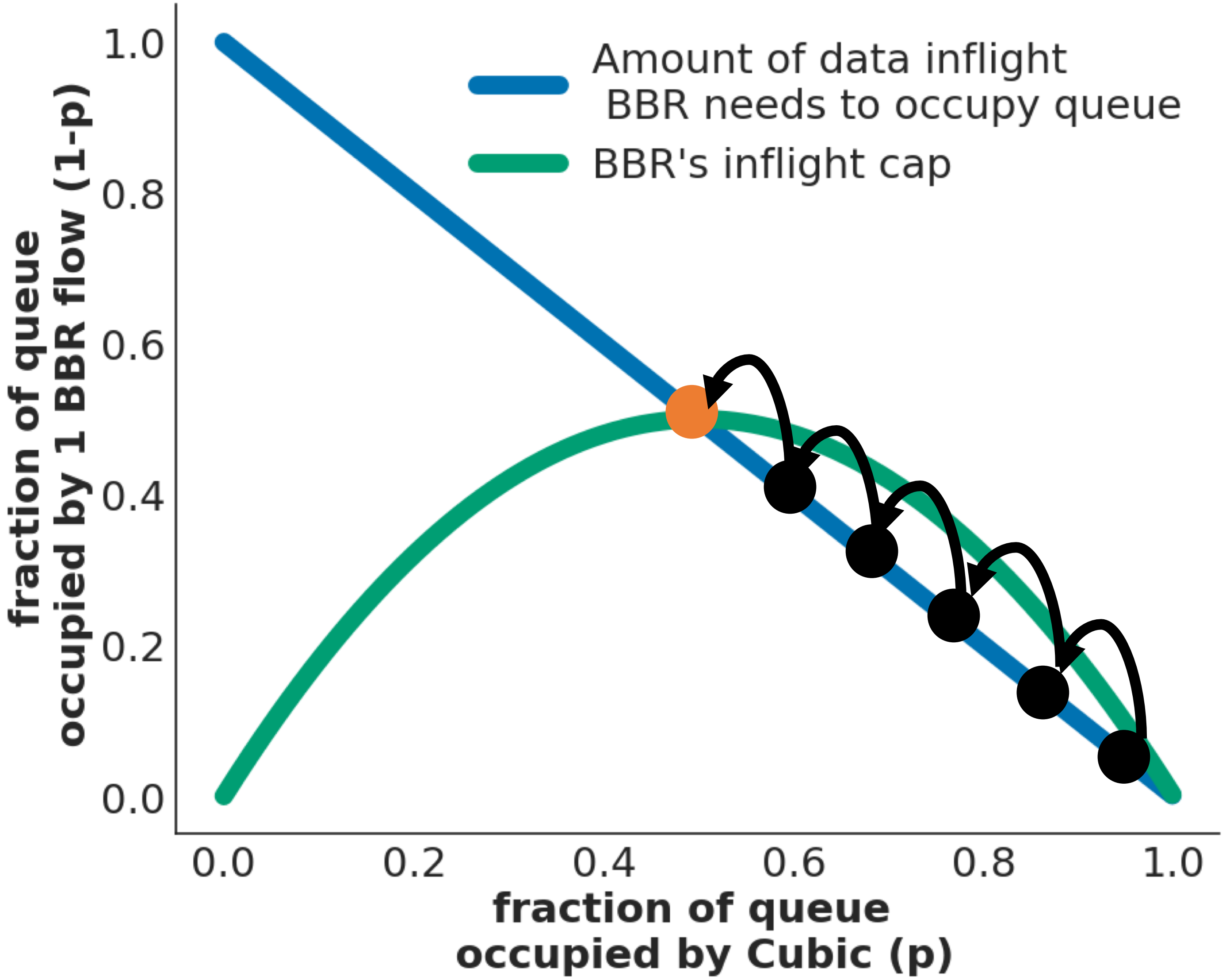
BBR inflight cap

$= 2 * BW * RTT$

$BW = (1 - p) * c$

$RTT = \frac{p * q}{c}$

$= 2 * (p - p^2) * q$



Assume:

c = btlnk link capacity

q = size of btlnk queue

p = fraction of btlnk queue occupied by Cubic

1- p = fraction of btlnk queue occupied by BBR

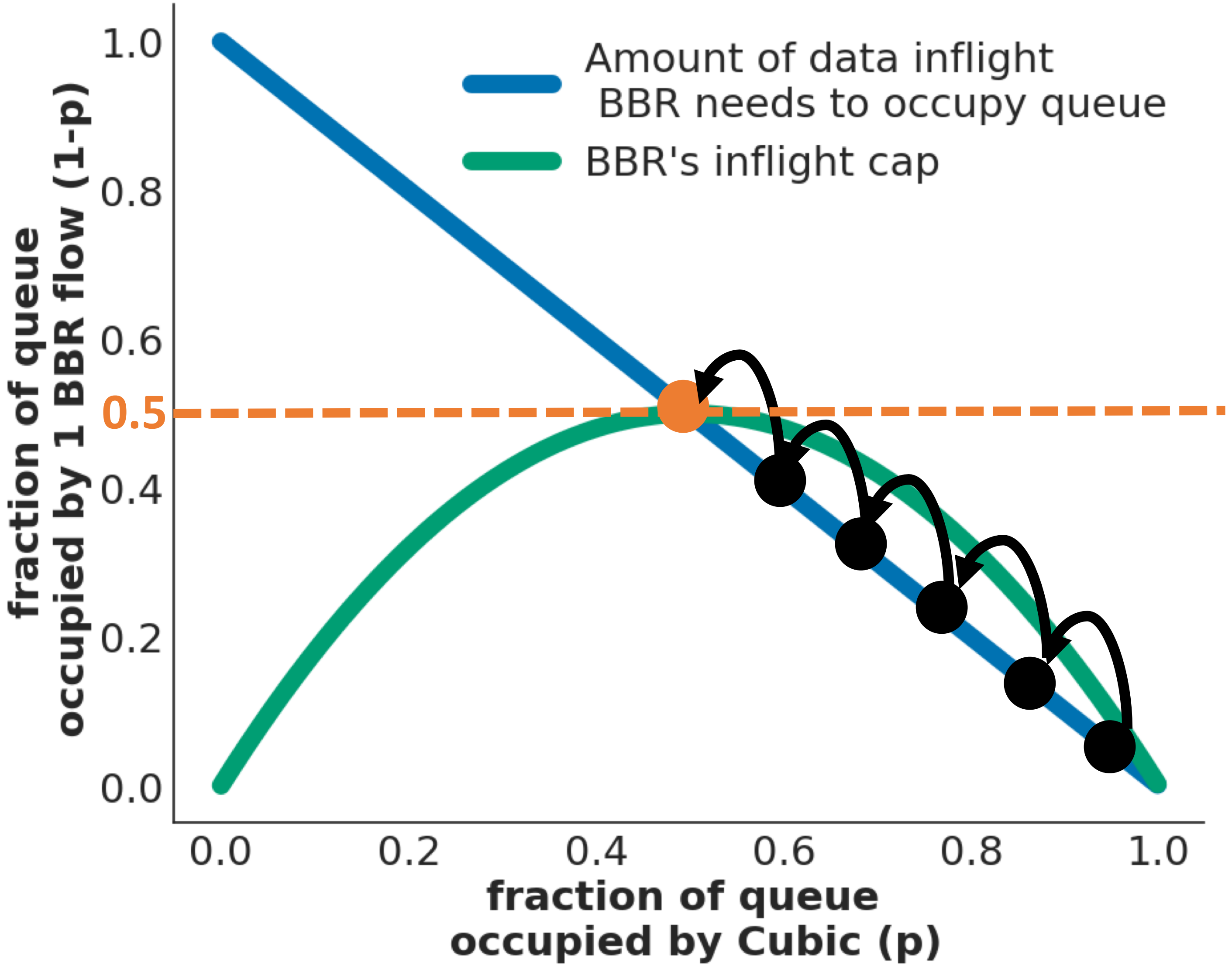
BBR inflight cap

$= 2 * BW * RTT$

$BW = (1 - \text{p}) * \text{c}$

$RTT = \dfrac{p * q}{c}$

$= \mathbf{2 * (p - p^2) * q}$



1 BBR flow gets **up to half the queue/link** with a 2 BDP in-flight cap.

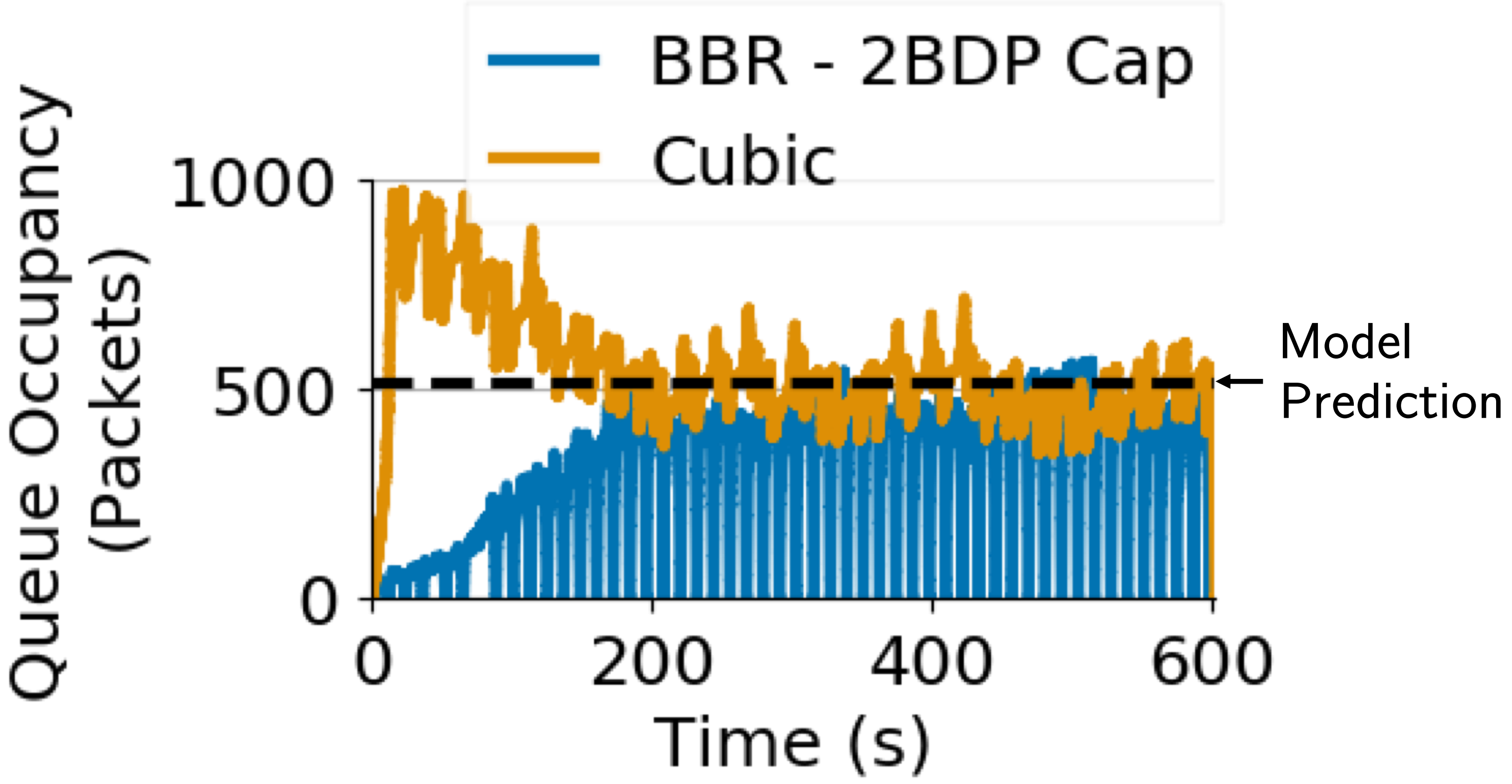


Figure: 1 BBR vs. 1 Cubic (32 BDP queue)

When we **change the in-flight cap**, we see BBR can get more of the queue.

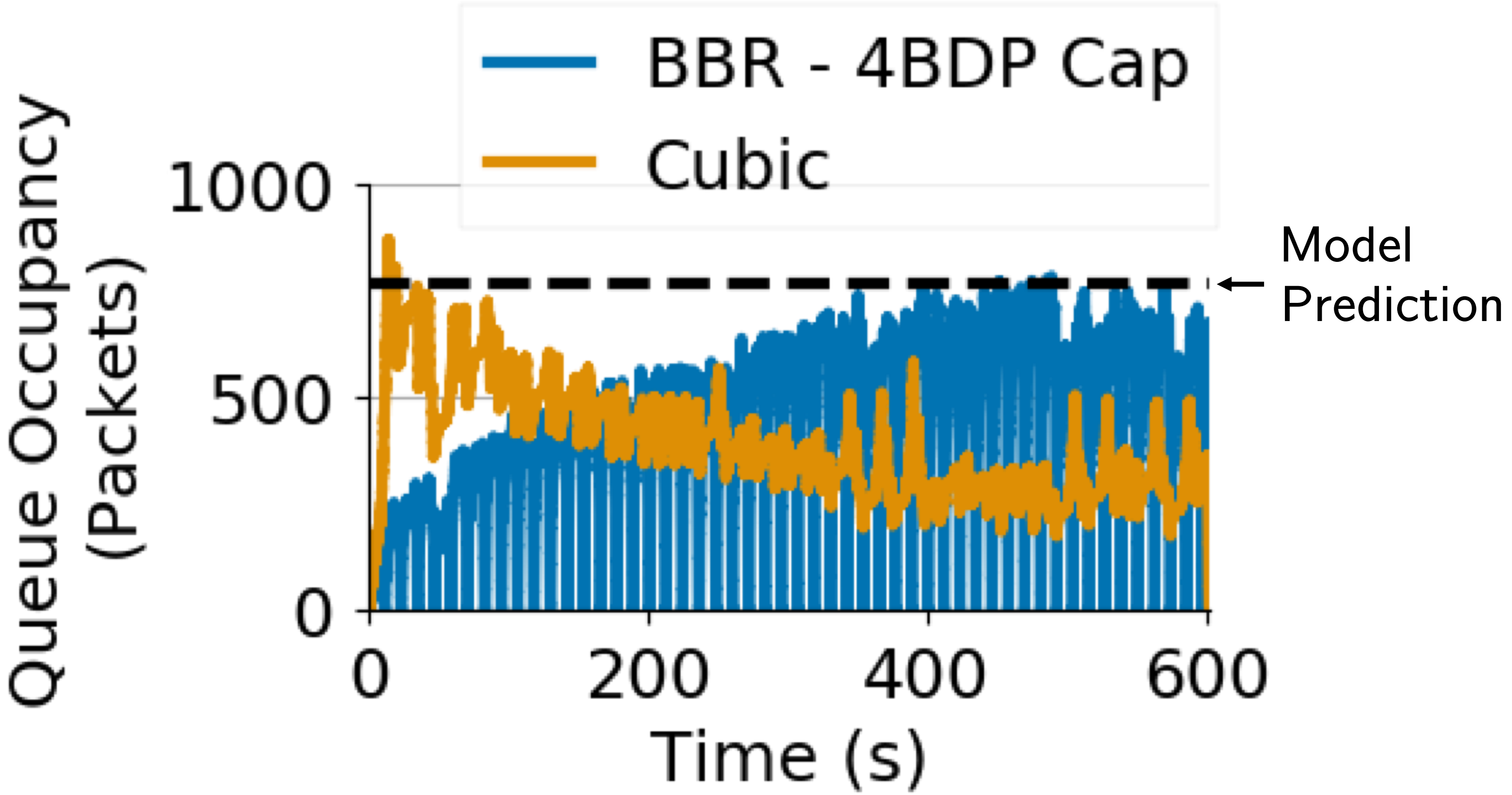


Figure: 1 BBR vs. 1 Cubic (32 BDP queue)

Our paper has a more **robust model** of BBR's in-flight cap.

4 Key Differences From Simple Model:

1. Propagation delay (l)
2. Queue size ($q = Xcl$)
3. # of BBR flows (N)
4. Probing overhead

$$BBR_{frac} = \left(1 - \frac{1}{2} + \frac{1}{2X} + \frac{4N}{q}\right) \times \left(1 - \left(\frac{q}{c} + .2 + l\right) \times \frac{1}{10}\right)$$

Our model predicts BBR's throughput when competing against Cubic flows with a median error of 5% (error is 8% for Reno).

4 Key Differences From Simple Model:

1. Propagation delay (l)
2. Queue size ($q = Xcl$)
3. # of BBR flows (N)
4. Probing overhead

See paper for details!

$$BBR_{frac} = \left(1 - \frac{1}{2} + \frac{1}{2X} + \frac{4N}{q}\right) \times \left(1 - \left(\frac{q}{c} + .2 + l\right) \times \frac{1}{10}\right)$$

Modeling BBR's Interactions With Loss-Based Congestion Control

Takeaways:

When BBR competes with other traffic, it becomes window-limited, sending packets at a rate determined by its in-flight cap.

BBR's in-flight cap does not depend on the number of competing loss-based flows.

Ranysha Ware

rware@cs.cmu.edu
@ranyshaware