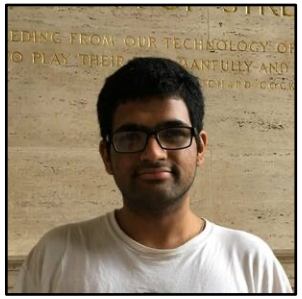
Towards provably performant congestion control

Anup Agarwal
anupa@cmu.edu



Venkat Arun

Devdeep Ray

Ruben Martins

Srinivasan Seshan

Carnegie Mellon

THE UNIVERSITY OF TEXAS AT AUSTIN

Congestion control algorithms (CCAs) break all the time. We want performance guarantees!

With 6 MSS (9KB) buffer CUBIC can only reach 2% capacity.

[PCC, NSDI 15]

On Brazil-California, PCC has 20x larger delay than lowest and on Stanford-California, only 52% of best throughput.

[Pantheon, ATC 18]

Under severe ACK aggregation, BBR leaves the bottleneck idle for potentially long periods.

BBR-dev RFC, 2018

CCAC finds examples where Copa gets arbitrarily low utilization.

[CCAC, SIGCOMM 21]

“Current methods to develop delay-bounding CCAs cannot always avoid starvation”

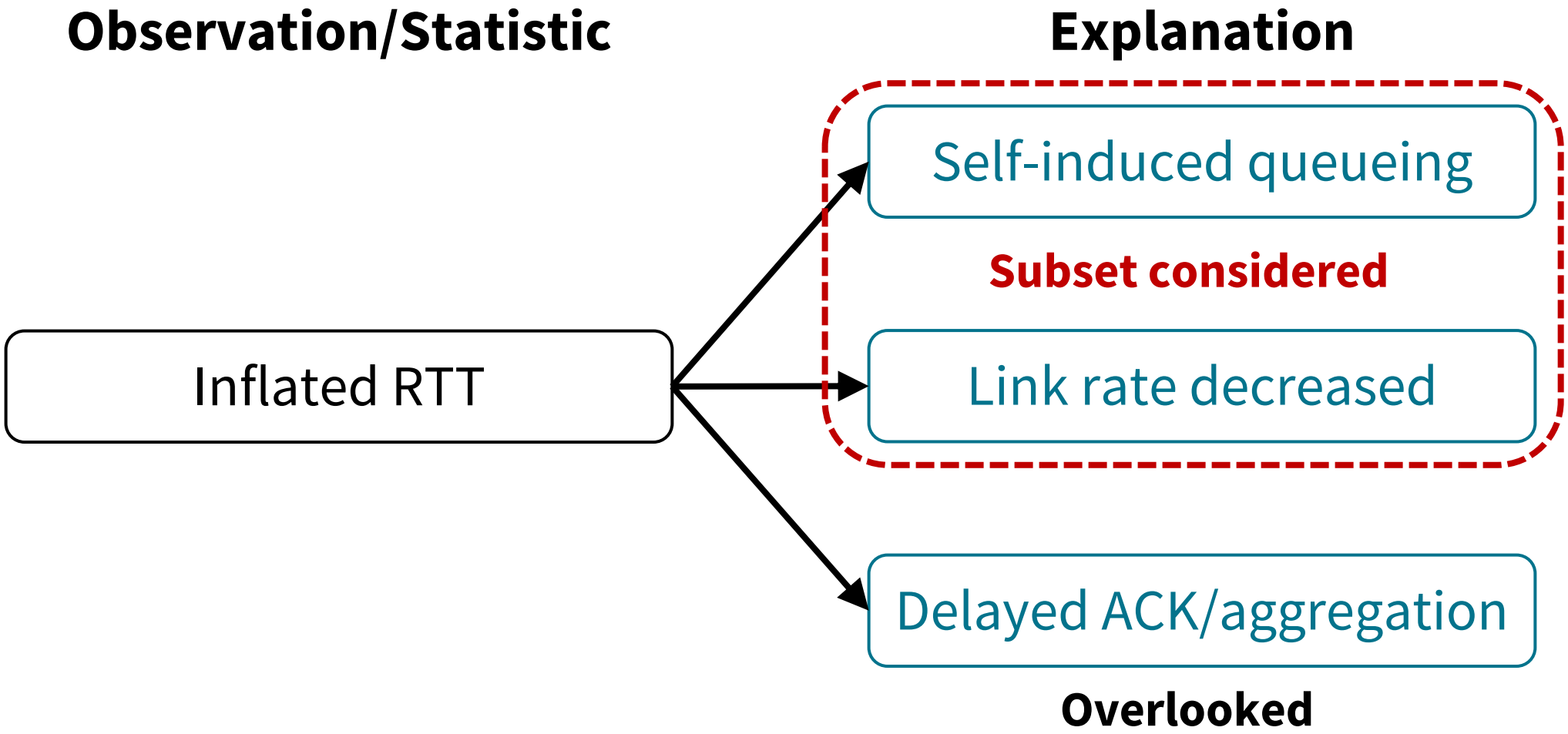
[Arun et al, SIGCOMM 22]

Traditional CCAs use ad-hoc statistics leading to implicit assumptions

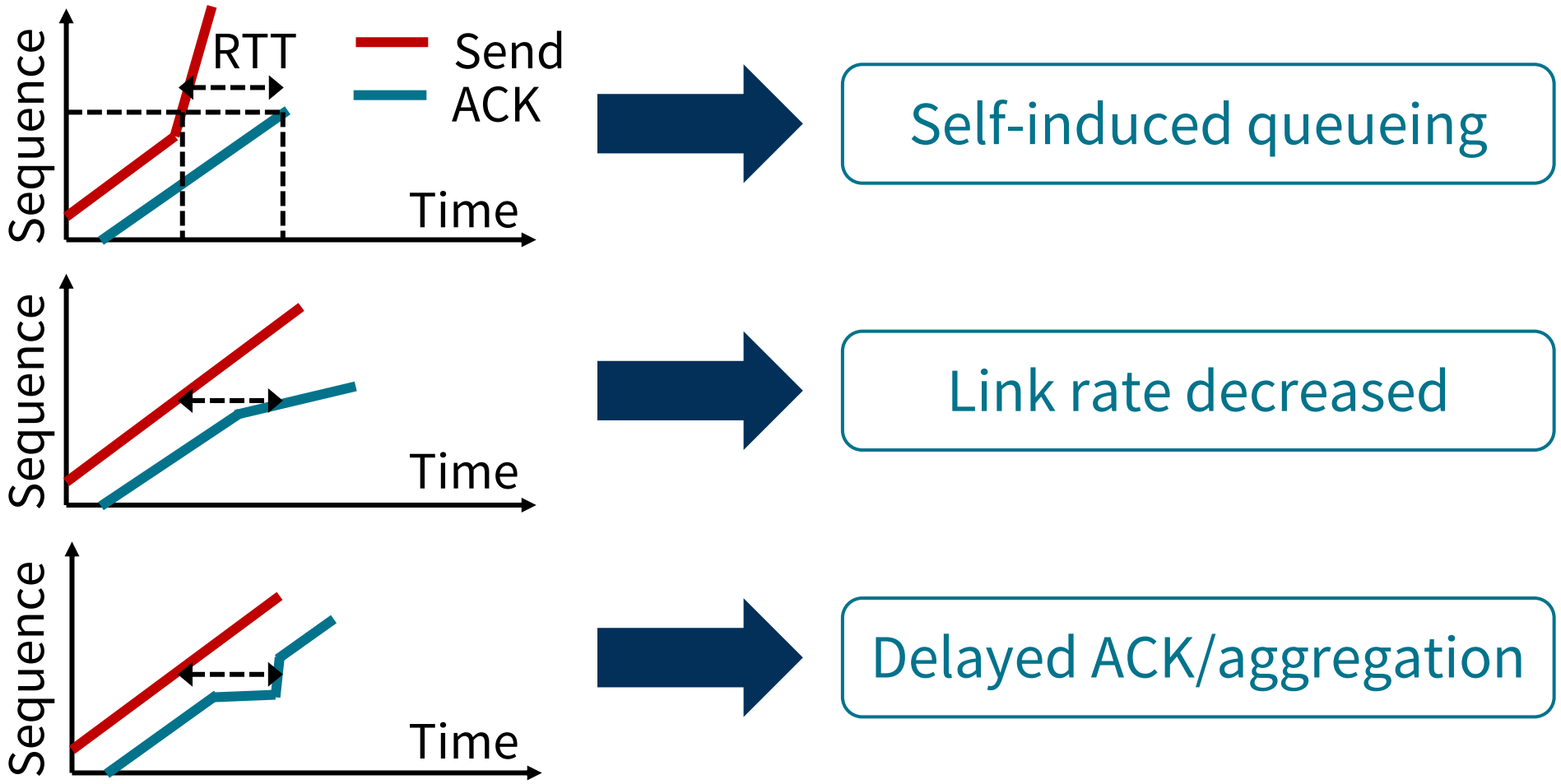
Ad-hoc statistics

- $ssthresh, W_{max}$
- $loss\ happened?$
- $RTT_{standing}, \frac{dRTT}{dt}$
- $\max ACK\ rate, RTT_{min}$

Issue 1. [False negatives] Ad-hoc statistics overlook network behaviors

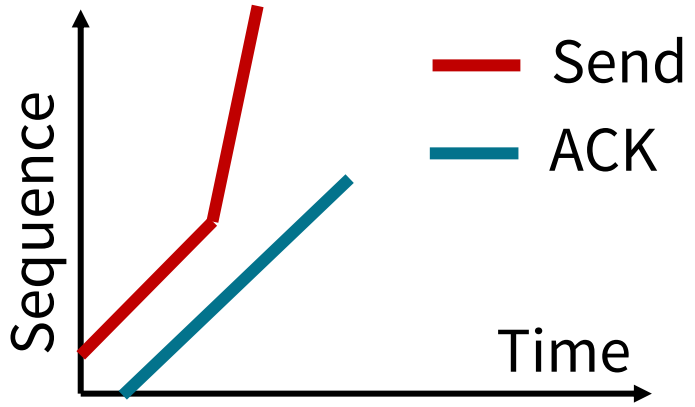


Issue 2: [False positives] Ad-hoc statistics fail to disambiguate explanations even when possible

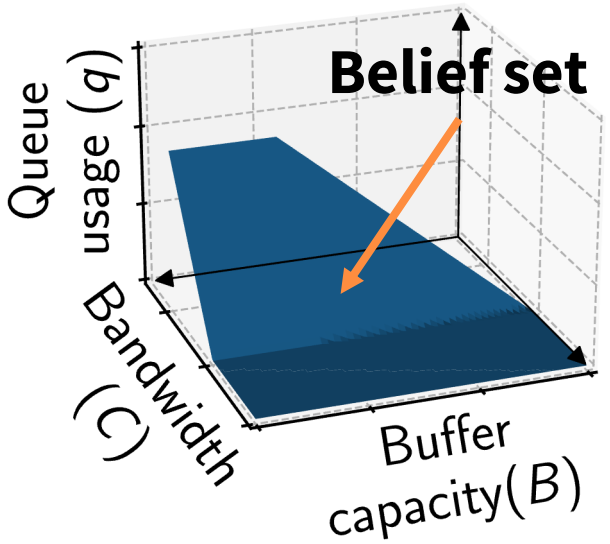


Belief set (Beliefs)

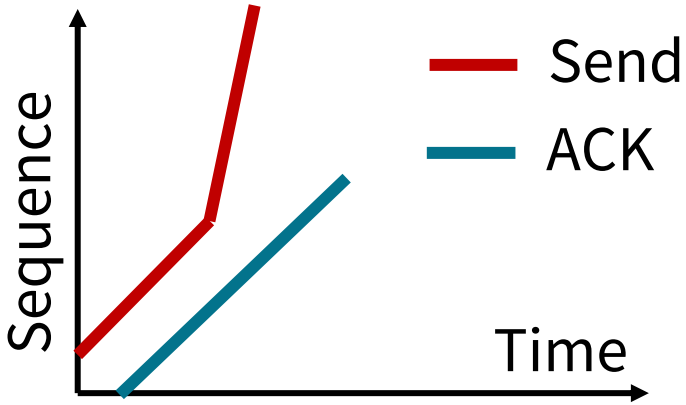
Set of latent parameter combinations that can explain the observation timeseries



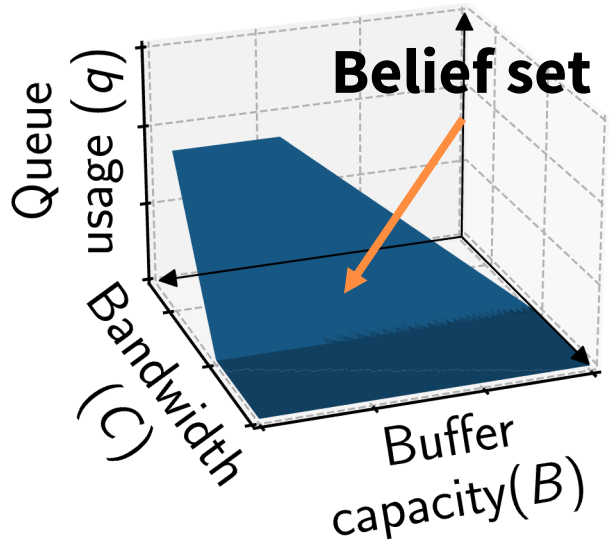
≈



Example point in belief set (parameter combination):
(C=10Mbps, B=100KB, q=10KB)



≈



+ prop. delay, inflight bytes …

Belief set vs. Ad-hoc stats

Set of latent parameter combinations that can explain the observation timeseries

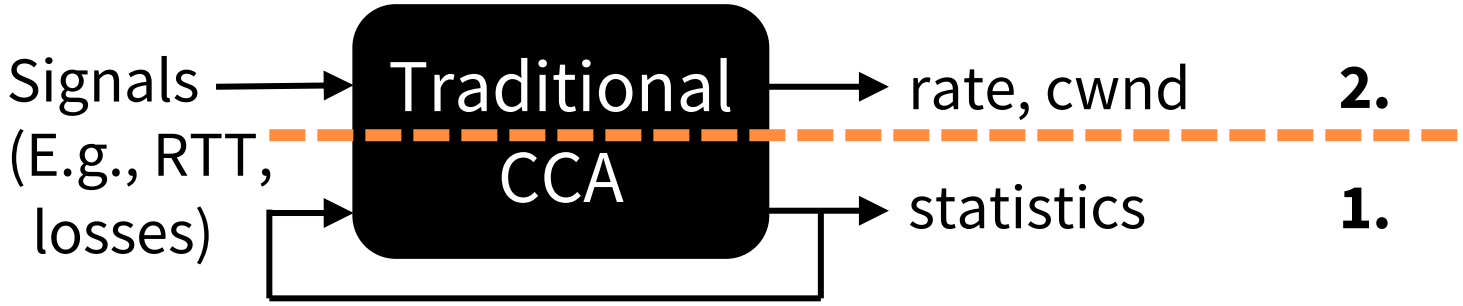
1. Provably equivalent to timeseries

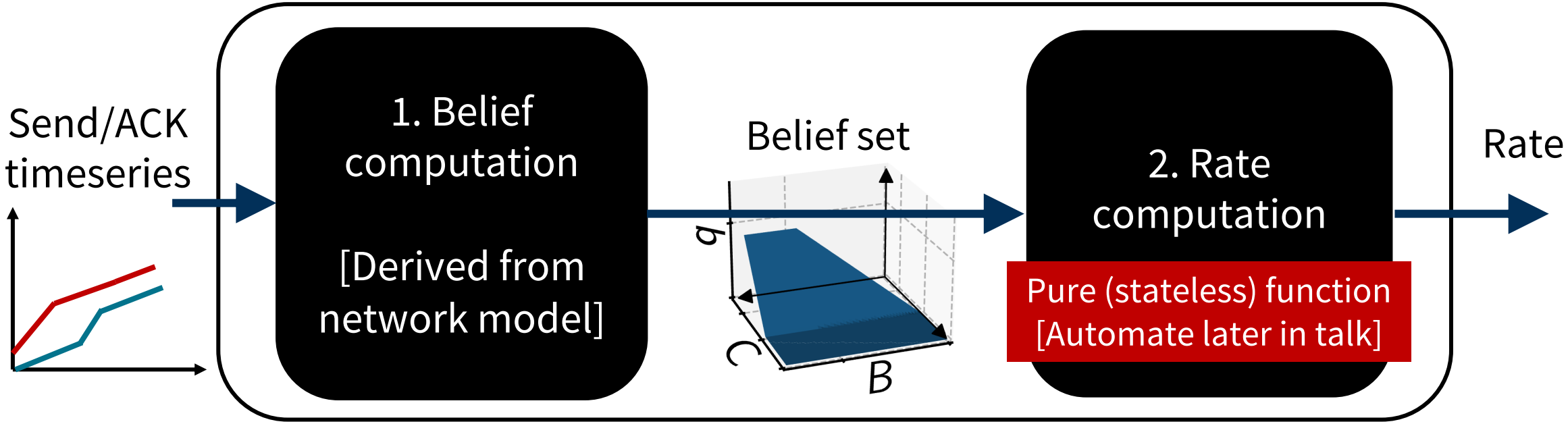
Theorem: If **any CCA** (i.e., "Timeseries → rate") can ensure objective then so can "Belief set → rate"

2. No false positives/negatives

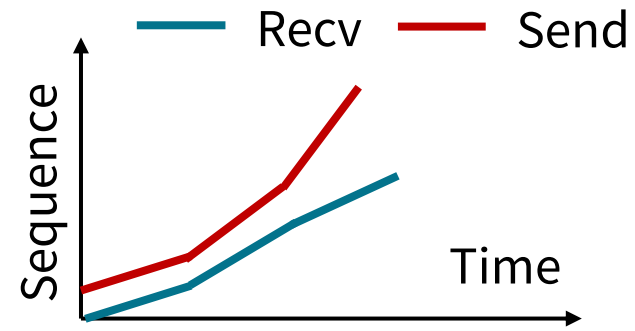
3. Mechanically derived from a network model (explicit assumptions)

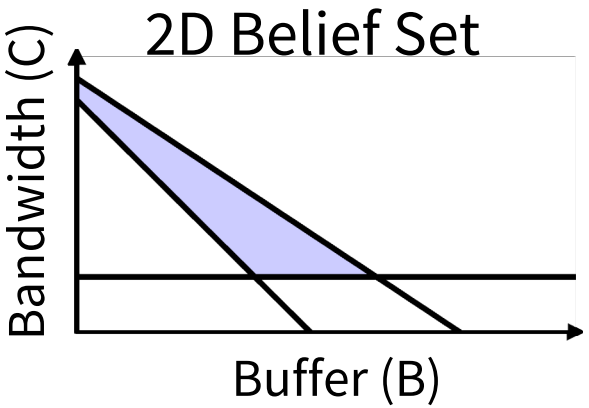
Benefit 1: Belief set simplifies CCA design



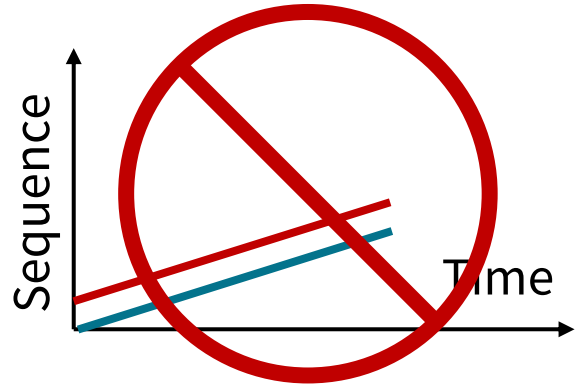


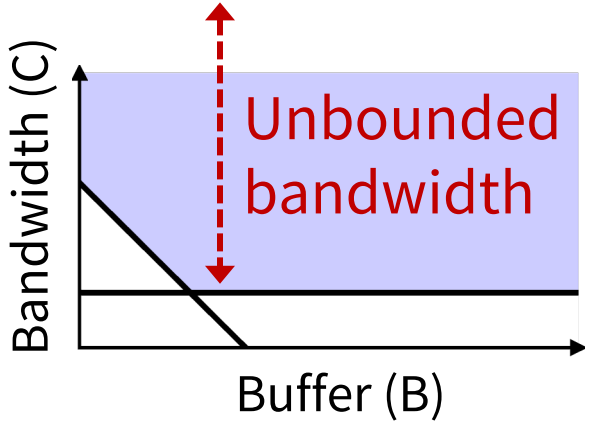
Benefit 2: Formally reason about tradeoffs [Necessary to shrink the belief set]





Not shrinking → Bad CCA



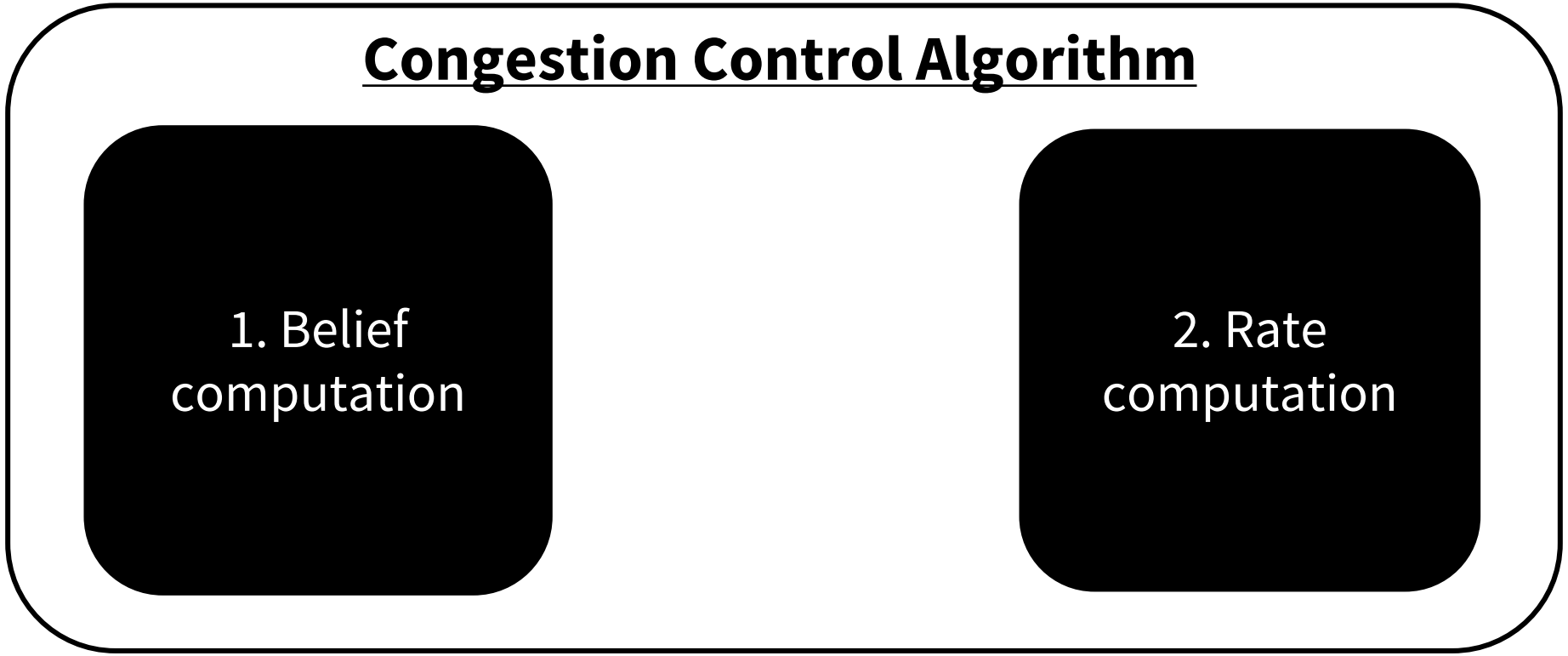


Proof strategy

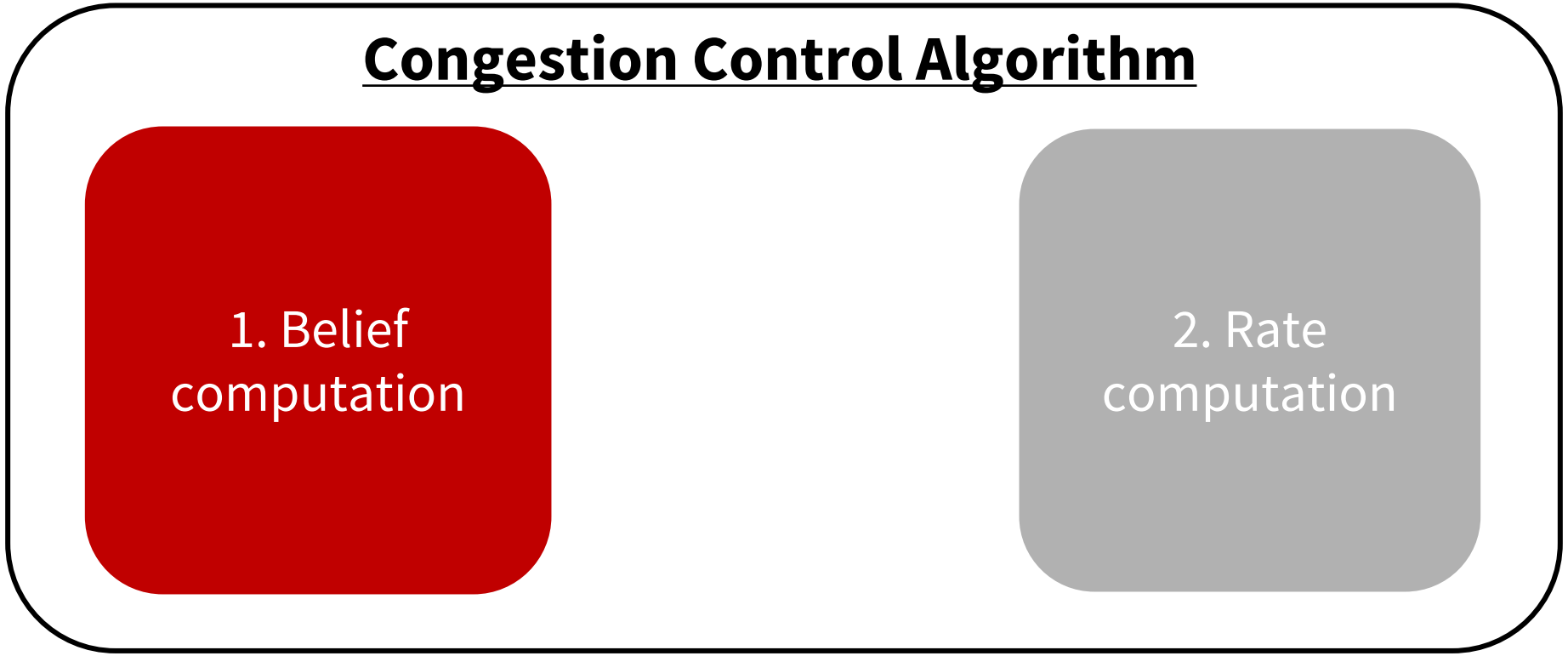
Only way to shrink beliefs is to violate objectives

Fundamental tradeoff in objectives

Talk outline

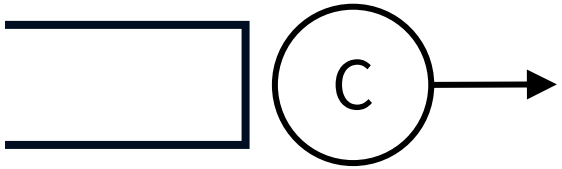


Talk outline



Background on network models [Mathematically describe packet service]

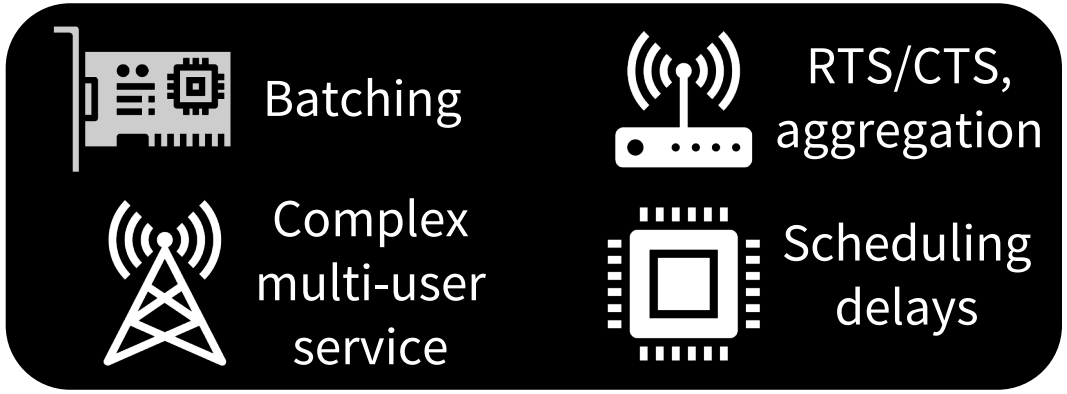
Idealistic model



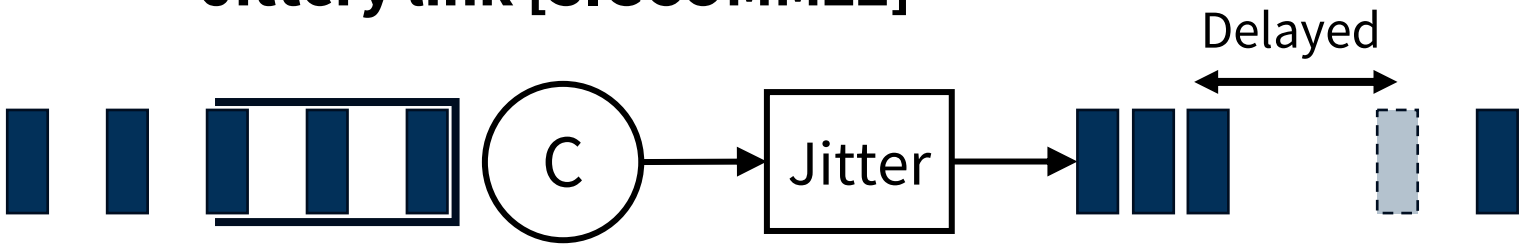
Simple deterministic or stochastic processes

Does not express

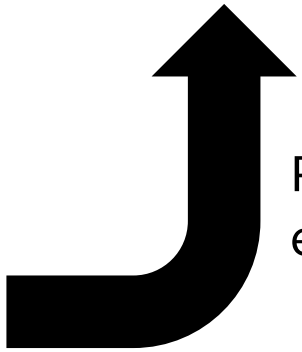
Behaviors in real networks



Jittery link [SIGCOMM22]

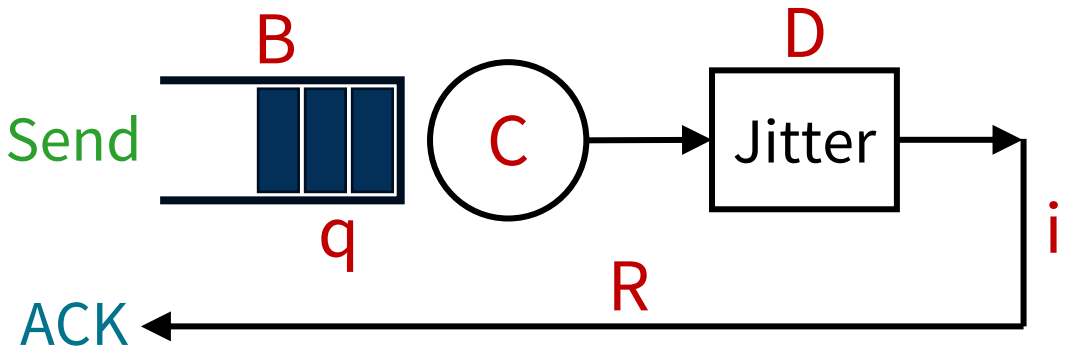


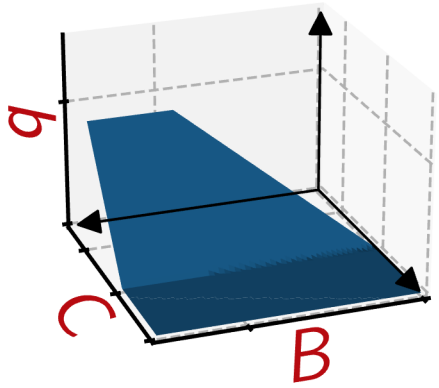
Arbitrarily delay packets up to D seconds



Provably express

Network model ➔ Belief computation





+ R, D, i

Non-observables [Latents]
Observables [Send/ACK seq]

Mathematical constraints

Describe possible ACK seq
given latents and send seq

$$A[T] - A[0] \leq C * (T + D)$$

Re-interpret constraints

Describe possible latents
given ACK seq and send seq

$$C \geq \frac{A[T] - A[0]}{T + D}$$

Talk outline

Congestion Control Algorithm

2. Rate computation

Stone image generated using https://deepai.org/

Program synthesis: Belief-to-rate program

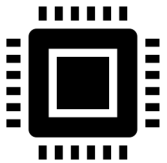
Encode into logical formulas [CCAC, SIGCOMM21]

Query specification

Network model

Objectives

- Utilization $\geq 80\%$
- Queuing delay $\leq D$
- Loss $\leq$ 1 pkt / RTT



SMT Solvers

Func(beliefs) → rate

if $q_{max} > MSS$:

rate $= C_{min} - q_{max}/R_{min}$

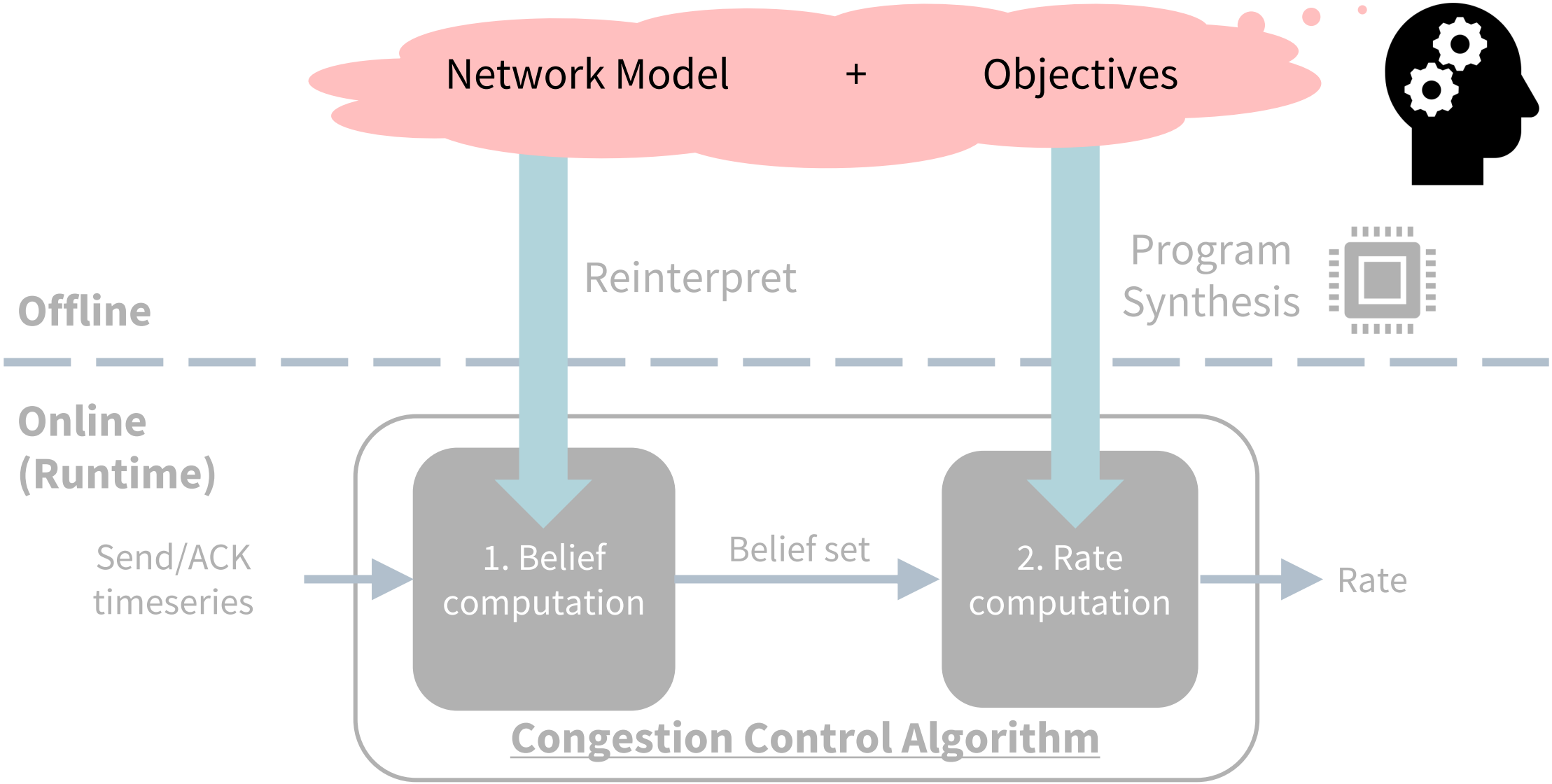
else:

rate $= 2C_{min} + MSS/R_{min}$

return rate

"Find **Func(beliefs) → rate** that ensures **objectives** on all scenarios described by **network model**"

Putting it all together



Results

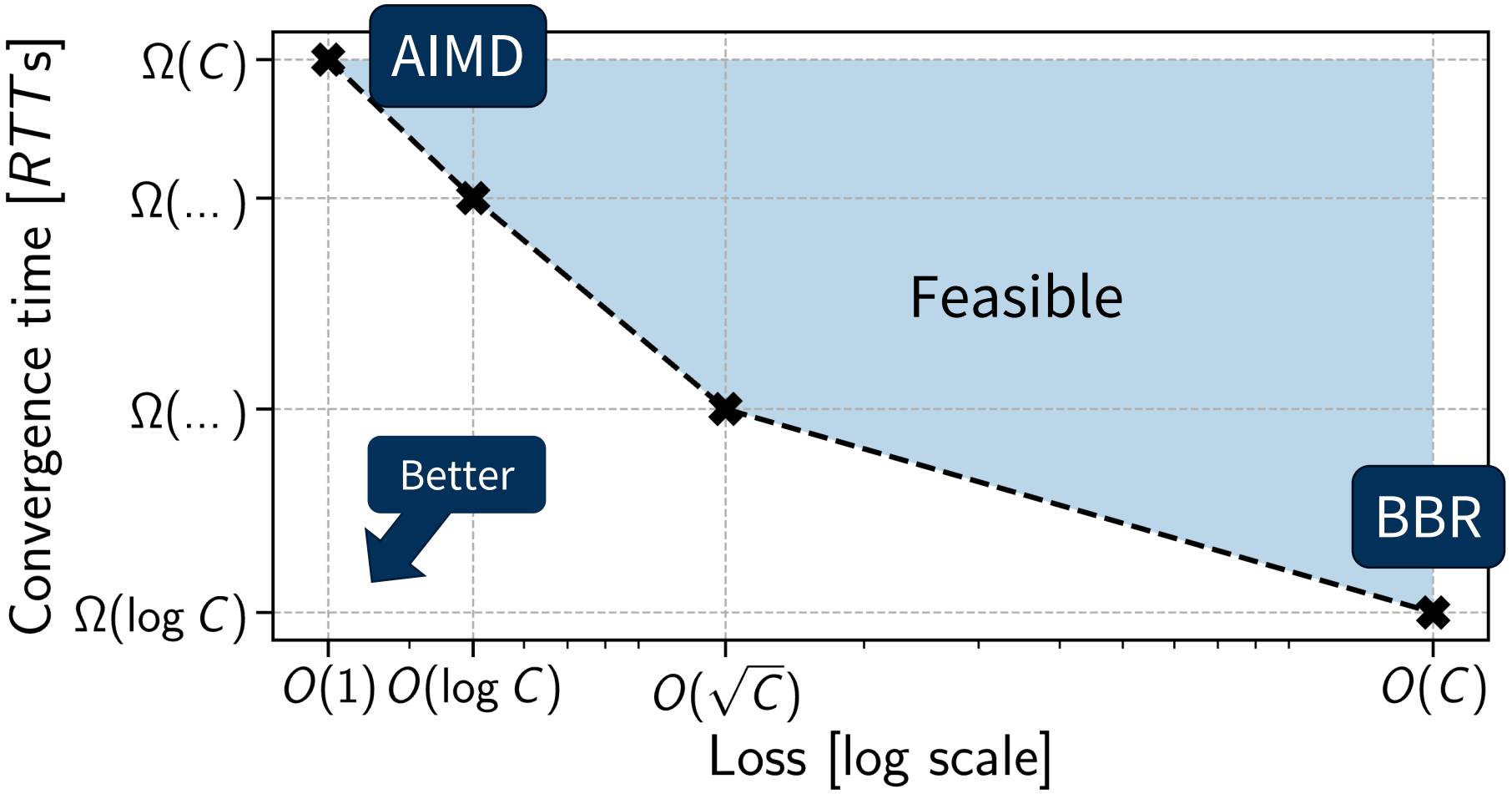
1. Fundamental loss-convergence time tradeoff
2. Synthesized CCAs
3. Proofs about performance
4. Empirical evaluation

Results

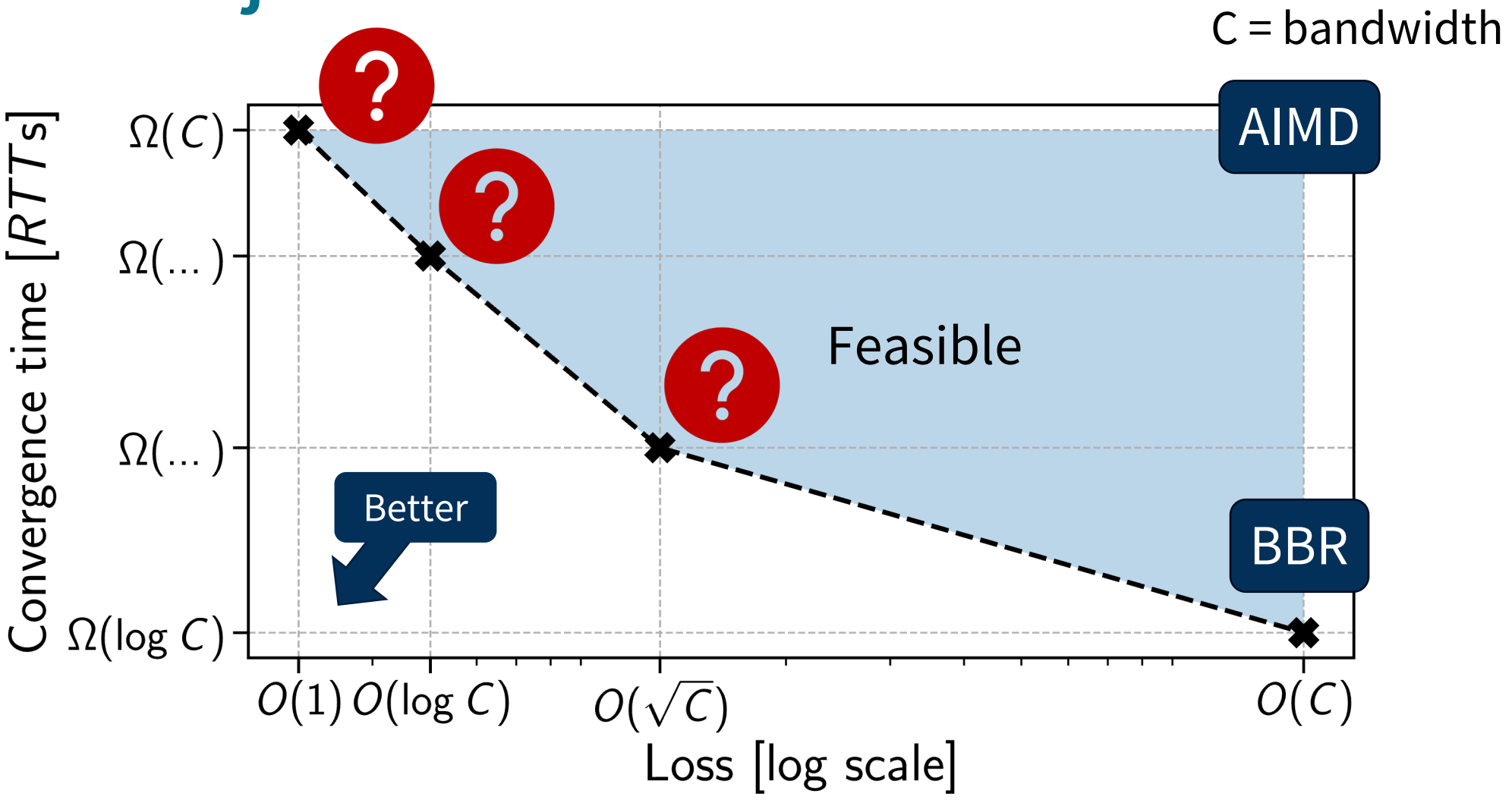
1. **Fundamental loss-convergence time tradeoff**
2. **Synthesized CCAs**
3. Proofs about performance
4. Empirical evaluation

Loss-convergence tradeoff due to jitter and shallow buffers

C = bandwidth



Loss-convergence tradeoff due to jitter and shallow buffers



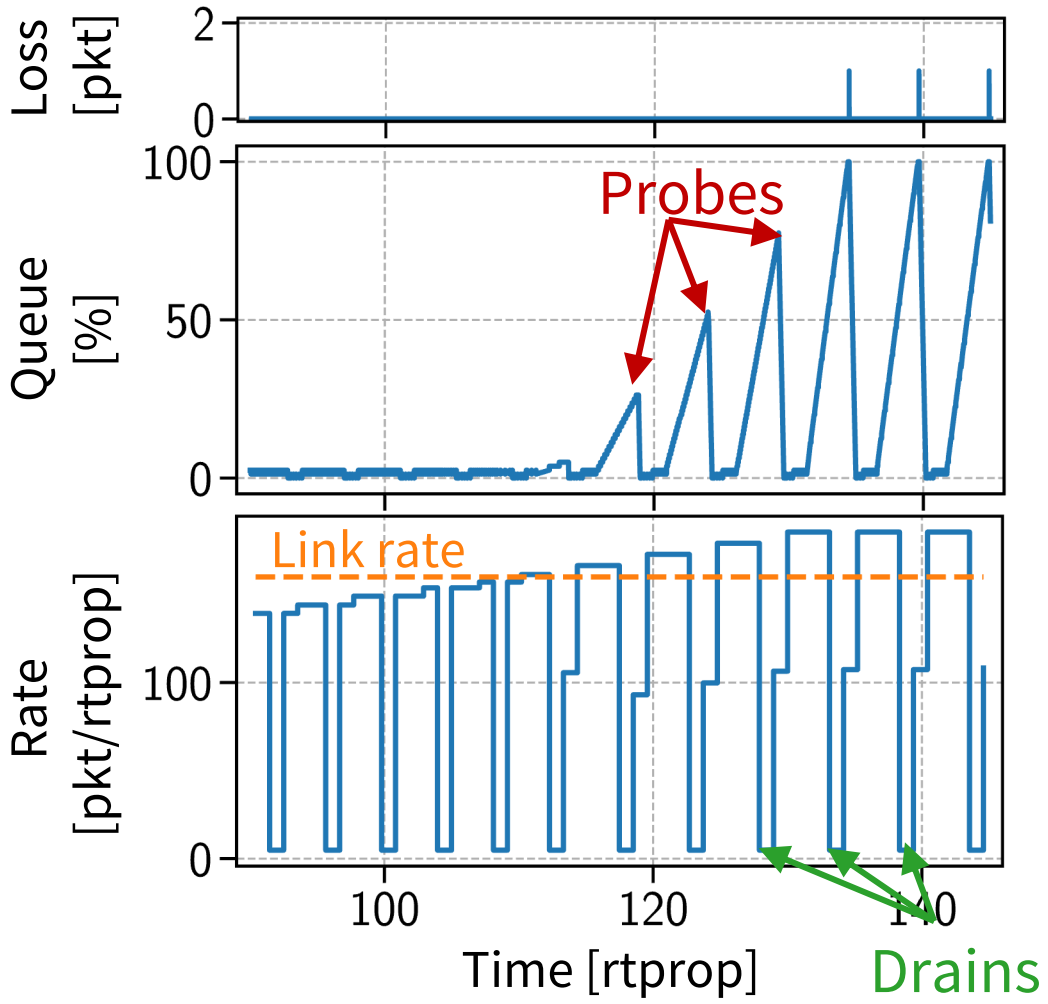
Synthesized CCAs on the Pareto frontier

- Coordinated rates & duration for drains & probes

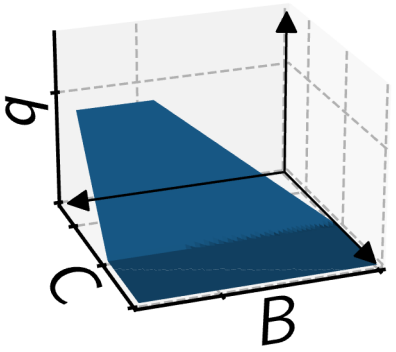
Surprising behavior:

- Draining not only to infer propagation delay (rtprop),
- But is a key part of probing

See paper for details & how the belief set evolves



Key takeaways



Belief set

- Summarizes possible network states, given observations
- Enables structured/systematic CCA **design** and formal **analysis**

Design

- Automatically synthesize competitive CCAs
- Derived from spec (Network model + Objectives), no hand tuning

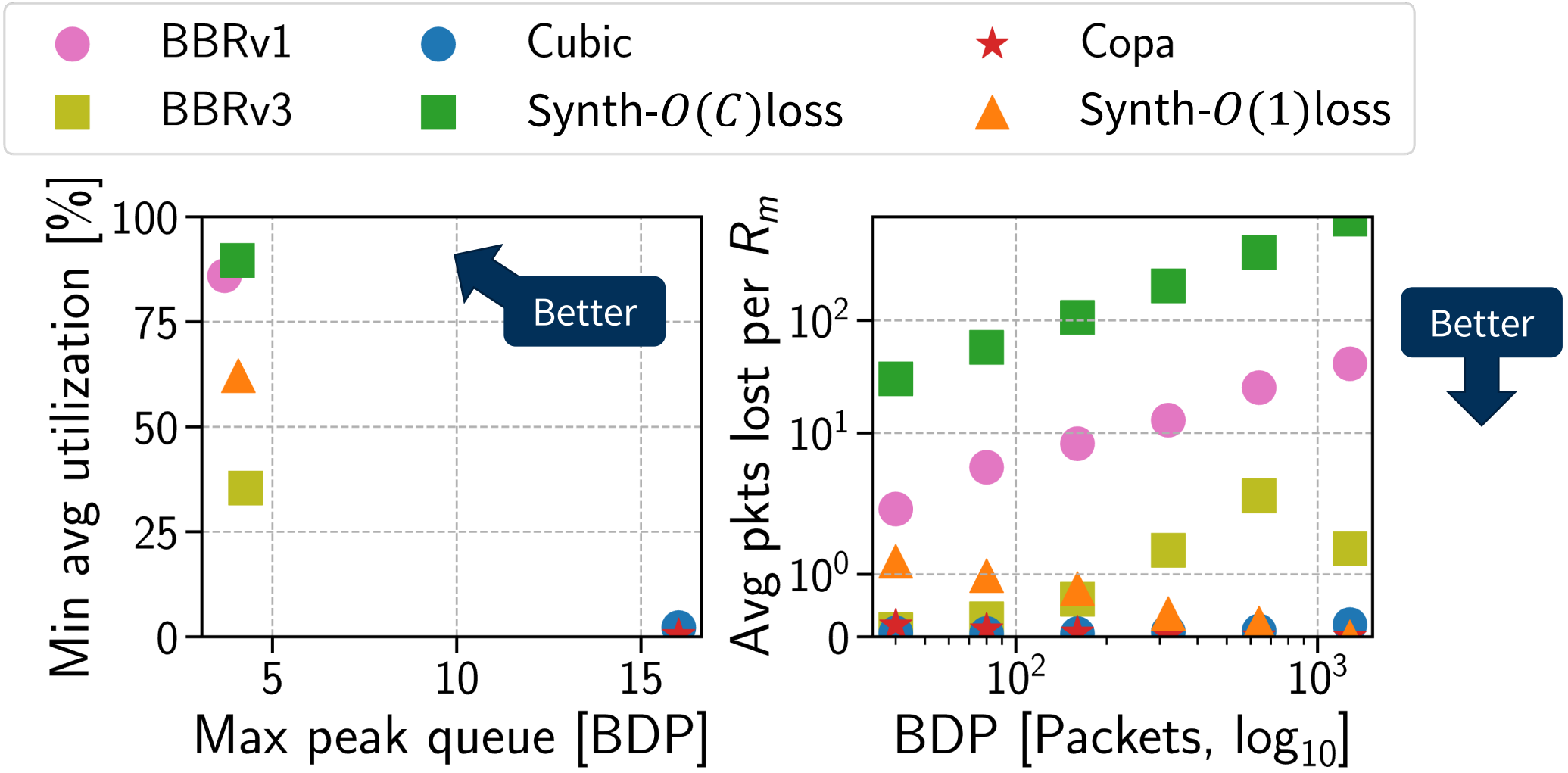
Analysis

- Prove performance guarantees
- Discover & prove impossibility results

Contact:
anupa@cmu.edu

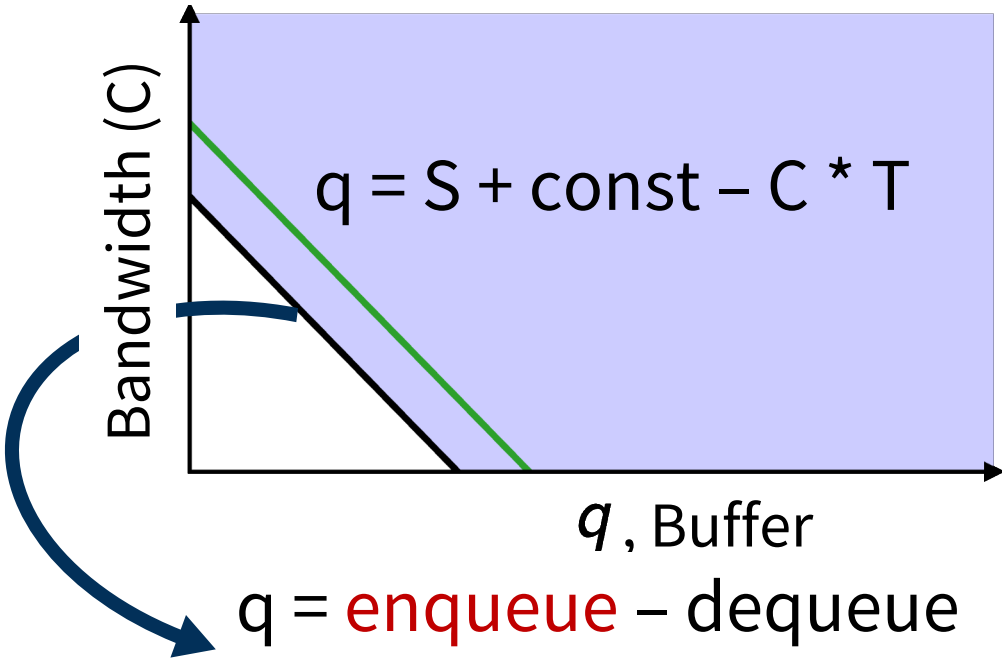
Backup slides

Synthesized CCAs empirical evaluation



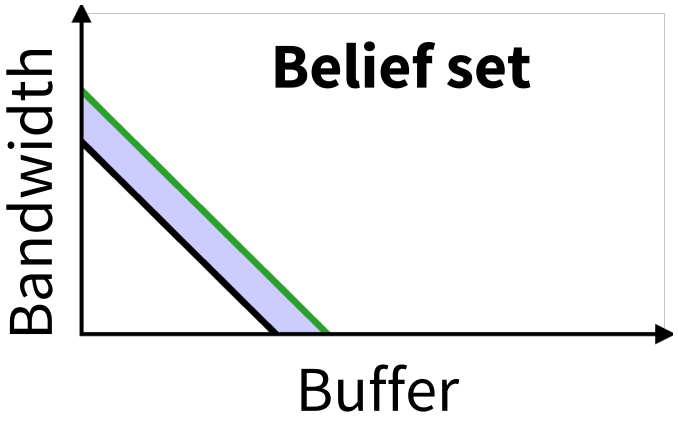
Interesting draining + probing mechanism to meet the Pareto frontier

Observation: No loss on sending S bytes in T time

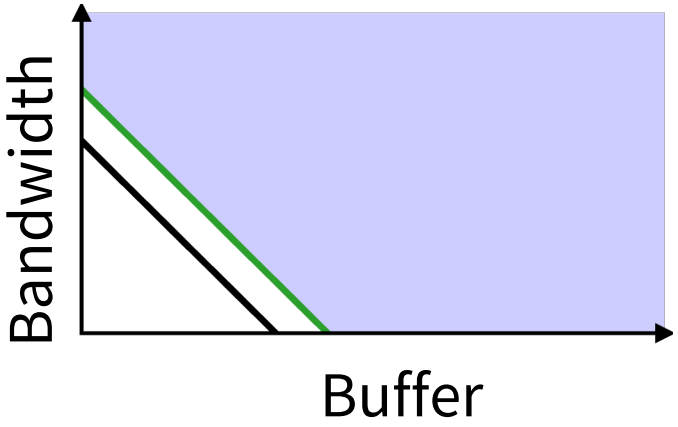


q = enqueue – dequeue
= S – C * T

Loss observed



No Loss



Handling changing network parameters

Parameters change within the belief set. Nothing to do

Parameters change to values outside the belief set

- Belief set becomes empty
  - Recompute using recent history
- Does not become empty
  - Speculatively recompute beliefs using recent history

Multi-flow co-existence and Fairness

- Currently, we do not explicitly model multi-flow scenarios
- CCAs guarantee O(D) delay ➔ agg. utilization guarantee
- Empirically some CCAs fair (JFI 0.94), others unfair (JFI 0.58)

Potential solutions for fairness

Existing fairness mechanisms (jitter-free links)

Reno	$cwnd \propto 1/\sqrt{loss\_rate}$	**Timely**	None (unfair)
Copa, Vegas, FAST	$cwnd \propto 1/delay$	**Swift**	$cwnd \propto 1/delay^2$
BBR (rate limited)	Inc-Dec (de-sync probes)	**DCTCP**	$cwnd \propto 1/\sqrt{fraction\ ECN}$
BBR (cwnd limited)	???		

- [Contract] Explore the space of common (implicit) signal contracts
- [Inc-Dec] sub-linear increases, super-linear decreases

- For the above 2, does one imply the other always?
- What is the domain of contracts? Utility (objectives) vs. contract?
  - Inverse sqrt, Inverse sqr, Inverse. What works best and when?
- Currently contract is after-thought (derived). Should we design contracts first?

CCAs tailored for network model & objective combinations

Network model	Environment	Objectives		CCA
		Losses	Convergence time	
CCAC/CBR Delay	Infinite or large buffer	0	$O(log\ C)$	cc_qdel
CCAC/CBR Delay	Short or arbitrary buffer	$O(C)$	$O(log\ C)$	cc_probe_qdel
CCAC	Short or arbitrary buffer	$o(C)$	Any	Open problem
CBR Delay	Short or arbitrary buffer	$O(1)$	$O(C)$	cc_probe_slow
CBR Delay	Short or arbitrary buffer	$O(1)$	$o(C)$	Proved impossible

No existing CCA can guarantee even 1% utilization on these networks

Utilization, delay objectives included in all queries

Designing network models

- Techniques: Data driven, manually, or program synthesis
- Space of network models or language/grammar describing models.
- Q1. What model best fits observed data?
- Q2. What are the minimum assumptions to still build good CCAs?
- Q3. What model captures the behaviors that individual network elements exhibit?

Average-case vs. Worst-case performance

- Haven’t found strong evidence that improving worst-case hurts average-case. Perhaps we can explicitly optimize both.
- Annotate beliefs with probabilities. Optimize expected utility.
- Specification to include ensemble of network models
  - If network behaves like jittery link then ensure some objectives
  - AND if ideal link then better objectives

Timeline of research progress

- 2021 CCAC [SIGCOMM] – everything is broken
- 2022 CCmatic [HotNets] Ad-hoc CEGIS
  - Utilization, delay
  - Jittery link with infinite buffer, single-flow
- 2024 CCmatic [NSDI] Belief framework
  - **Utilization, delay, losses, convergence time**
  - **Jitter, shallow buffers, single-flow**
  - **Fundamental convergence vs. loss tradeoff**
- 2025?
  - Multi-flow fairness, coexistence?
  - Improve average-case performance? Other network models?
  - Expressivity …

Limitations & ongoing/future work

- Fairness/Coexistence
  - Sublinear increase, linear decrease [Chiu/Jain]
  - Contract from other flows to compute beliefs
  - Encode decisions (modulate rate + Fourier transform)

- Expressivity of templates
  - Compute best rate for history (or beliefs) instead of strategy
  - Game theory & deductive logic

- Performance of CCAs
  - Asymptotically optimal. Improve constant factors.
  - Improve average case performance
  - Other network models?

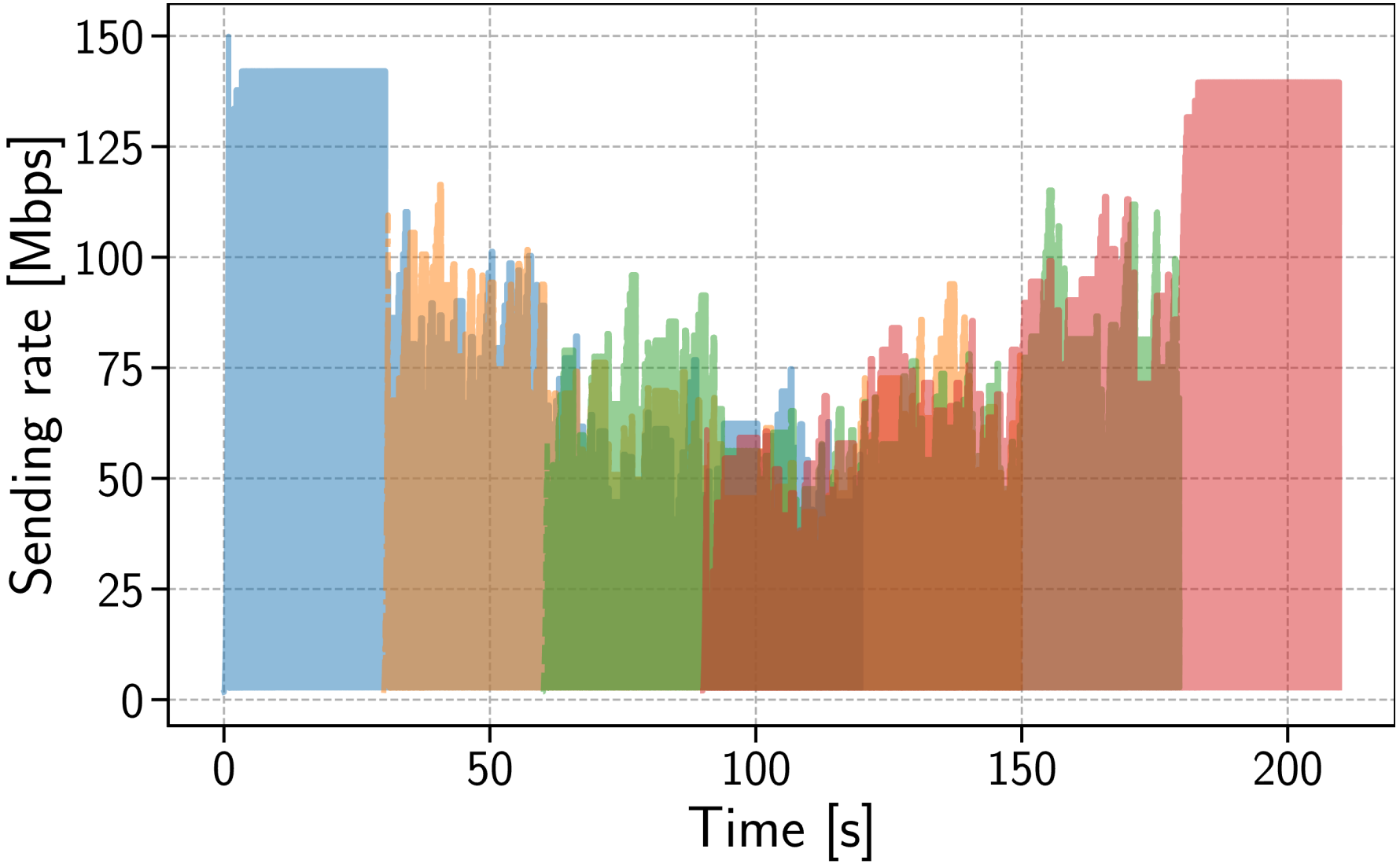
Some other use cases of beliefs

- Make existing CCAs explainable/robust.

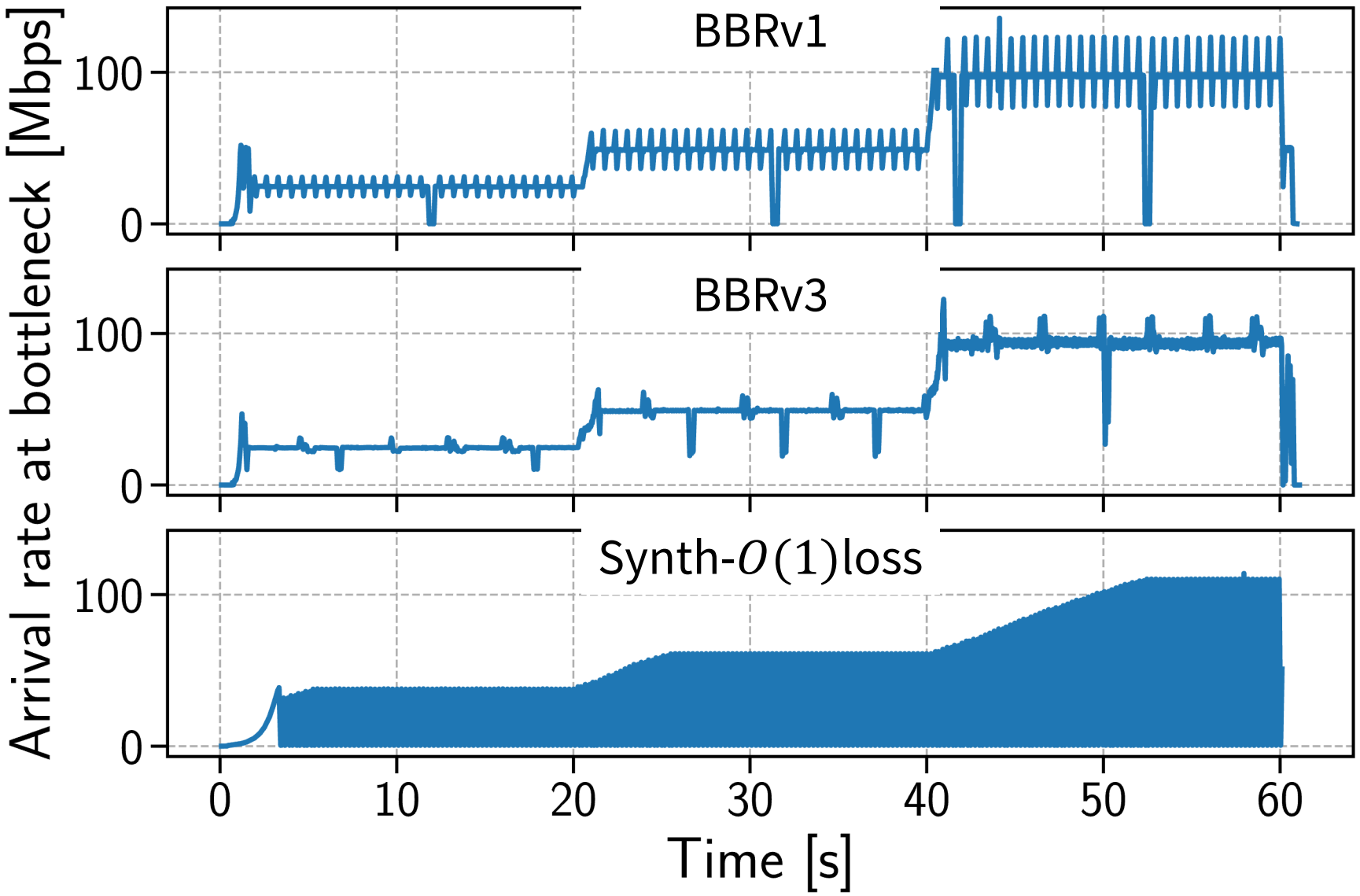
- **Debug CCAs in the wild.**
  If beliefs are very different from CCA's actions, then something is wrong. Trigger telemetry!

- **Characterize the Internet.**
  What fraction of paths exhibit X beliefs vs Y beliefs.

Empirical: Fairness



Empirical: Convergence time (Inc)



Empirical: Convergence time (Dec)