An experimental study of the learnability of congestion control

Anirudh Sivaraman, Keith Winstein, Pratiksha Thaker, Hari Balakrishnan

MIT CSAIL

May 22, 2014

Designing congestion-control protocols today

- Formulate a mental model of the target network and application workload
- Decide on the protocol's goal
- Design a protocol to achieve this goal on the target network
- Model and goal can either be implicit or explicit

But, the model is always wrong ...

- Bufferbloat when queues are incorrectly sized
- Diminished fairness and unpredictability in small-packet regimes
- Incast in datacenters
- Lost throughput under stochastic loss

Our work

- Can we formalize this design process?
- Quantify consequences of model mismatch

Approach

- Specify a *training scenario* for training.
  - Topology
  - Locations of senders and receiver
  - Application workload
  - Buffer size and queuing discipline
- Specify an *objective function*
- Synthesize a protocol using an automated protocol-synthesis tool, Remy [**?**]
- Evaluate on a *testing scenario* inside ns-2
- Difference between training and testing scenario represents model imperfection

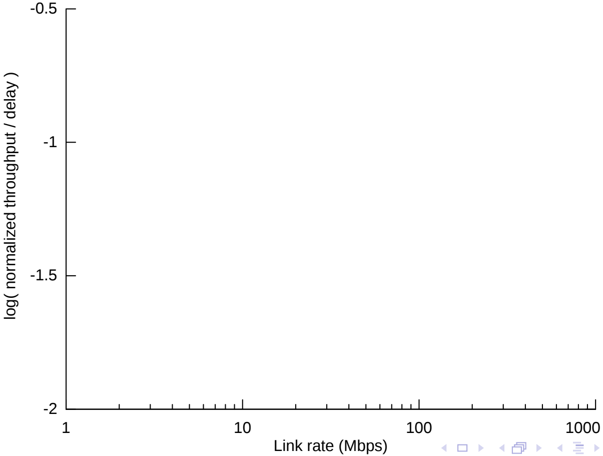
Automated protocol synthesis

- Find best protocol for an imperfect network model.
- Problem is NEXP-complete.
- Rely on Remy to produce Tractable Attempts at Optimal (TAO) congestion-control protocols.

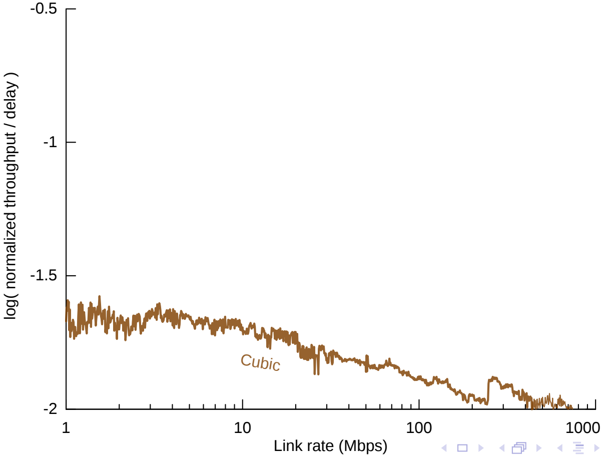
Tractable Attempts at Optimal

- Just how suboptimal are these TAO protocols?
-
-

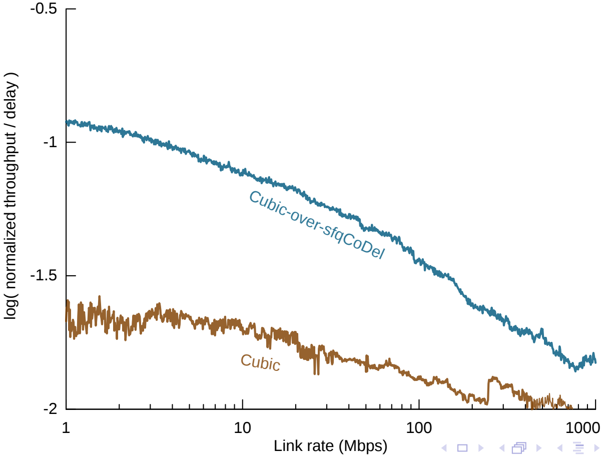
The cost of generality, or forwards-compatibility



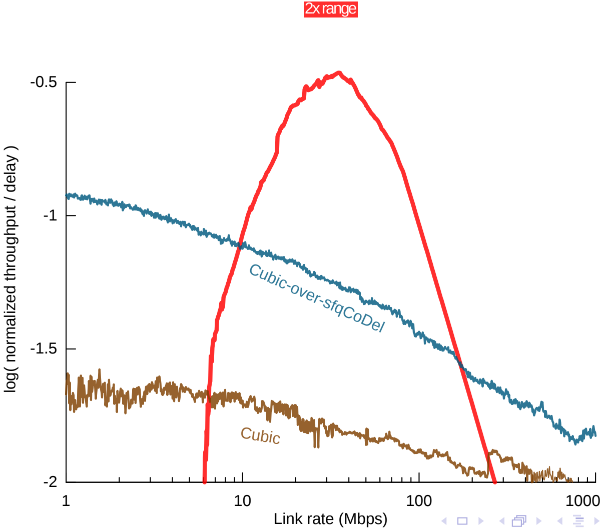
The cost of generality, or forwards-compatibility



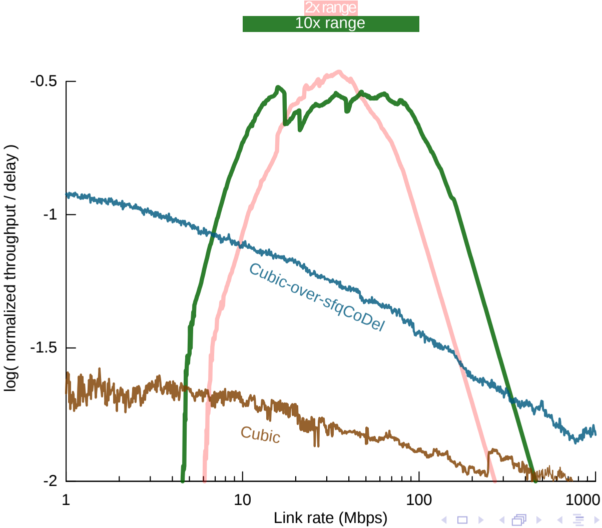
The cost of generality, or forwards-compatibility



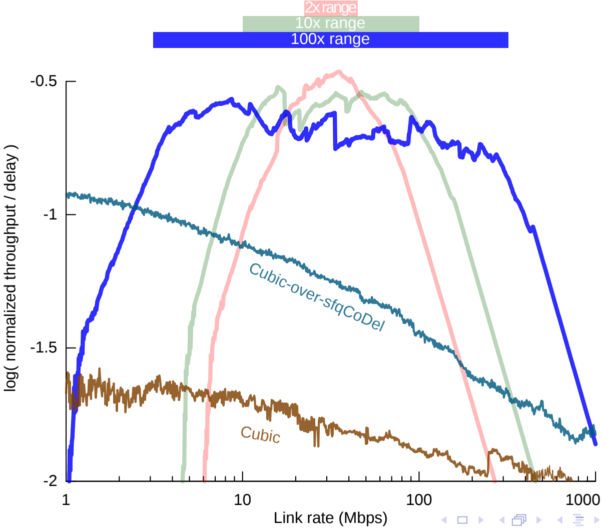
The cost of generality, or forwards-compatibility



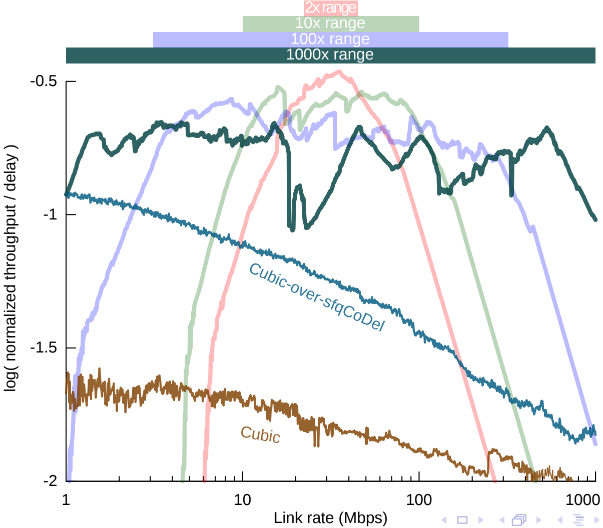
The cost of generality, or forwards-compatibility



The cost of generality, or forwards-compatibility



The cost of generality, or forwards-compatibility

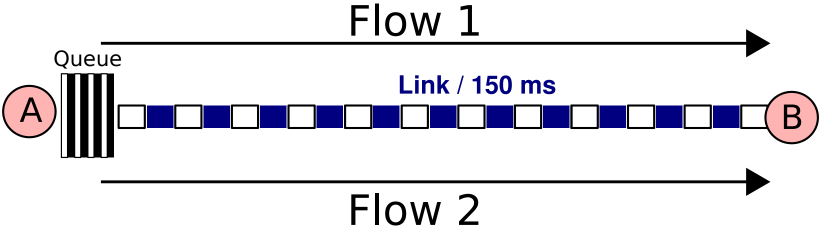


Workload imperfections

RTT imperfections

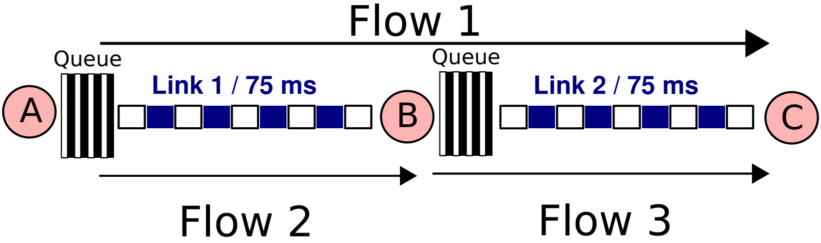
When the model is wrong about the topology

One bottleneck

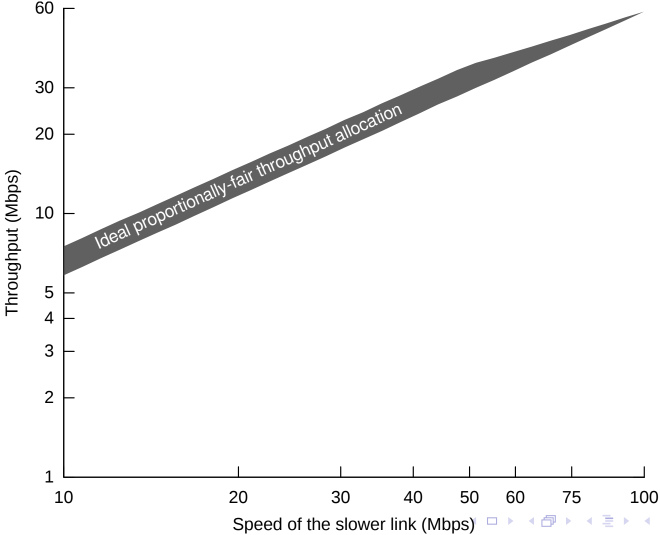


When the model is wrong about the topology

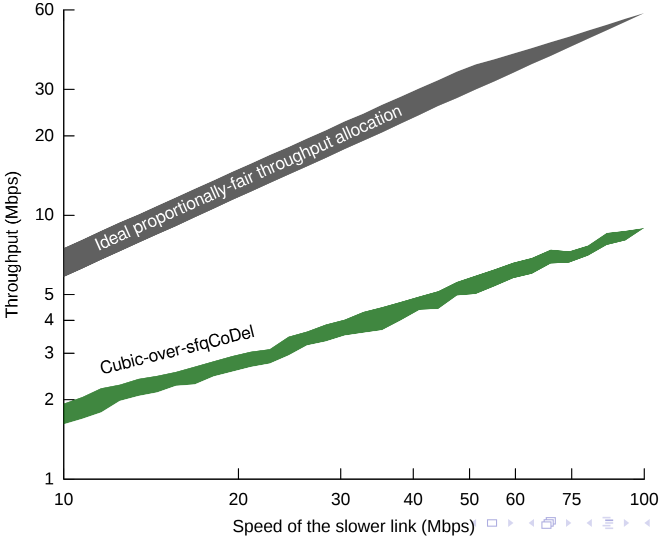
Two bottlenecks



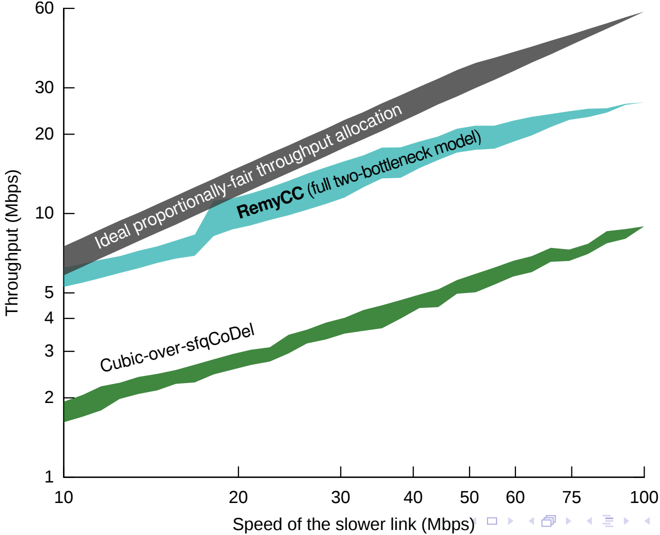
When the model is wrong about the topology



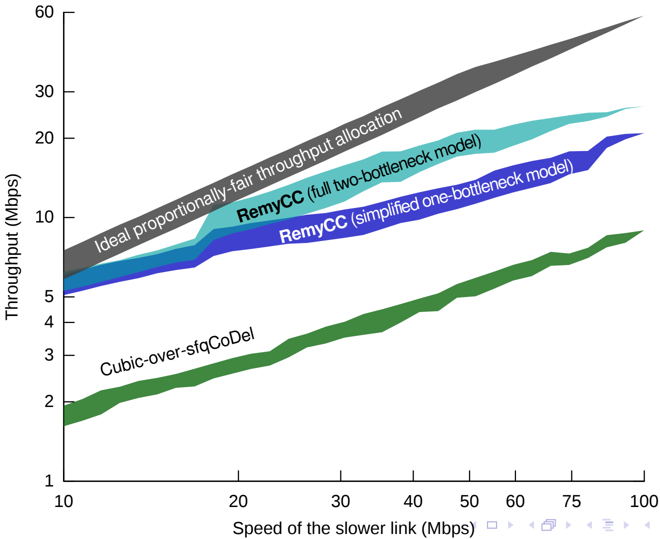
When the model is wrong about the topology



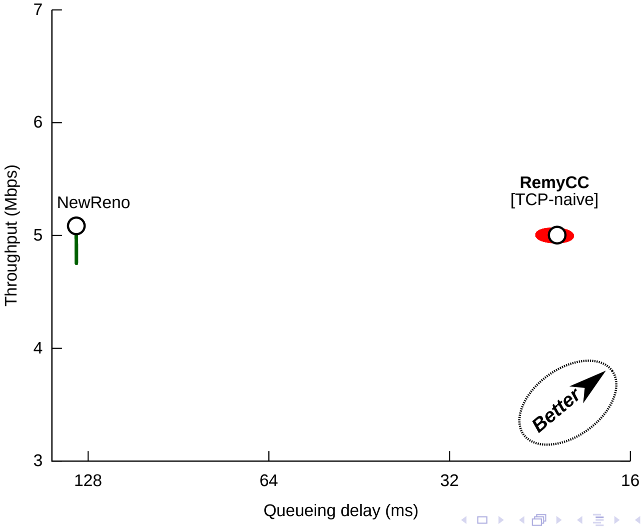
When the model is wrong about the topology



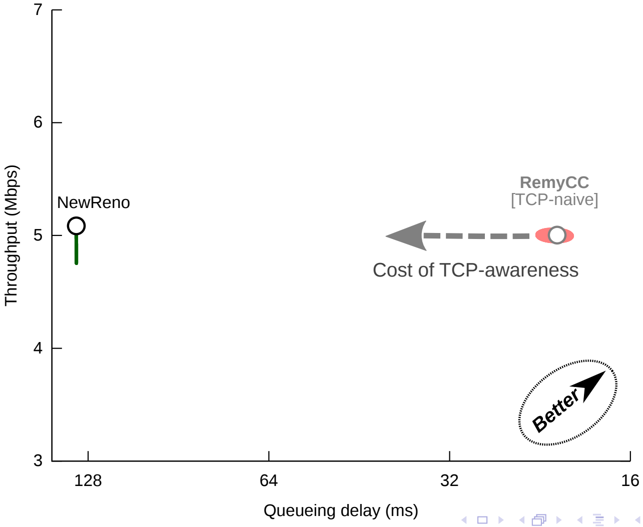
When the model is wrong about the topology



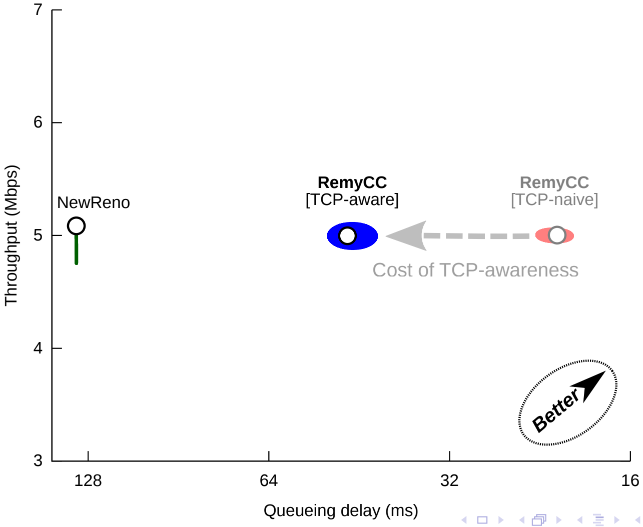
RemyCC competing against itself



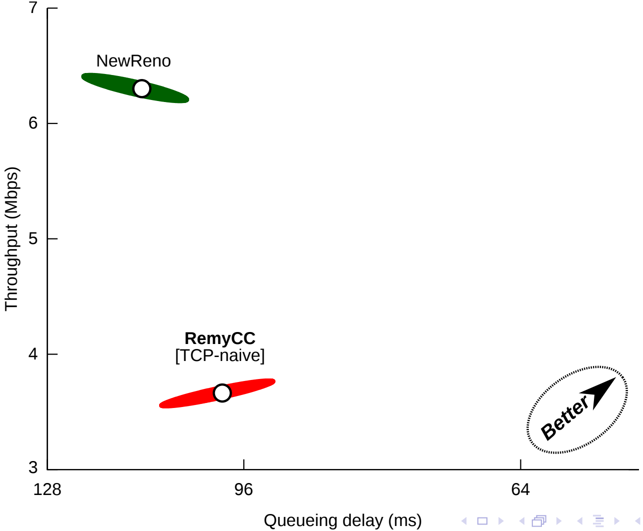
RemyCC competing against itself



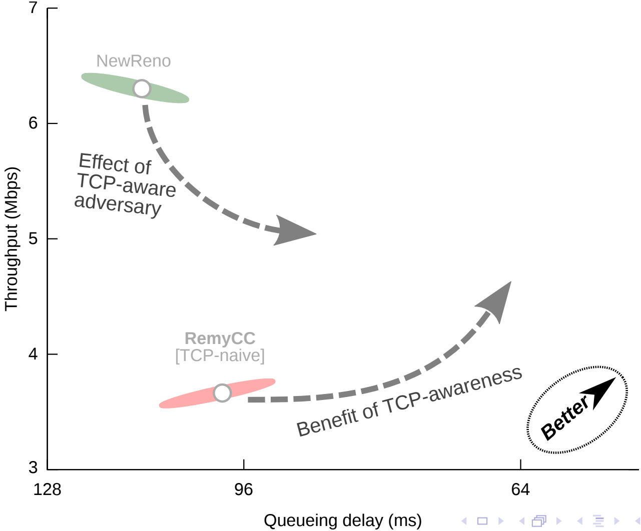
RemyCC competing against itself



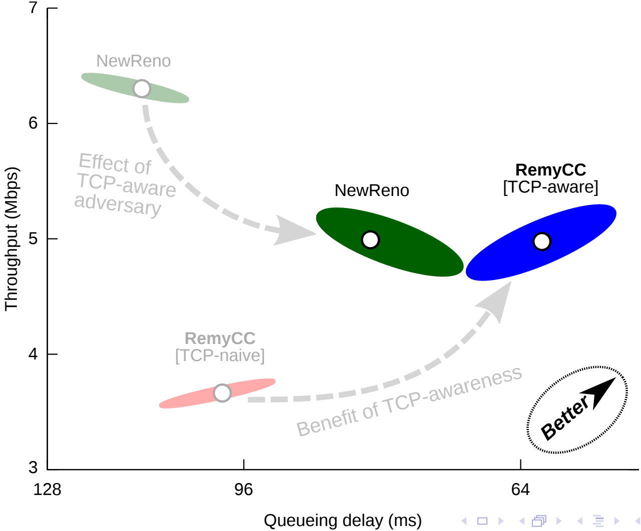
RemyCC competing against TCP NewReno



RemyCC competing against TCP NewReno



RemyCC competing against TCP NewReno



Coping with diverse application requirements

Related Work

- PAC
- Transfer learning

Limitations and future work

- Better characterization of optimal protocols
- Extending protocol-generation to in-network algorithms as well.
- Characterize model imperfections between simulation and the real world.
- Why are the results the way they are?