Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks

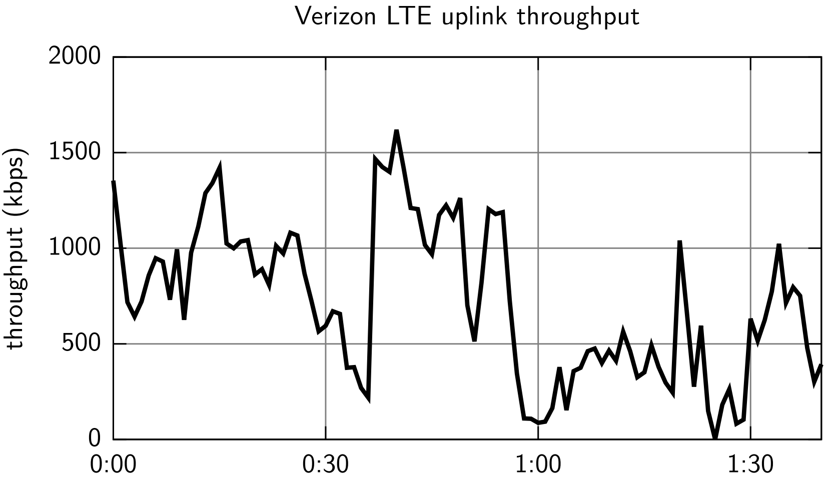
Keith Winstein, Anirudh Sivaraman, and Hari Balakrishnan

M.I.T. CSAIL

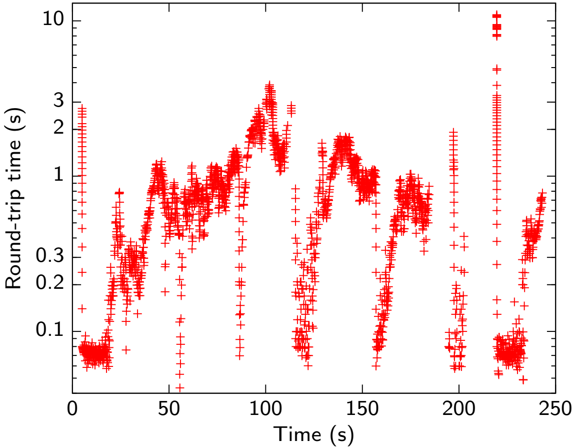
http://alfalfa.mit.edu

April 5, 2013

Cellular networks are **variable**



Cellular networks are **too reliable**

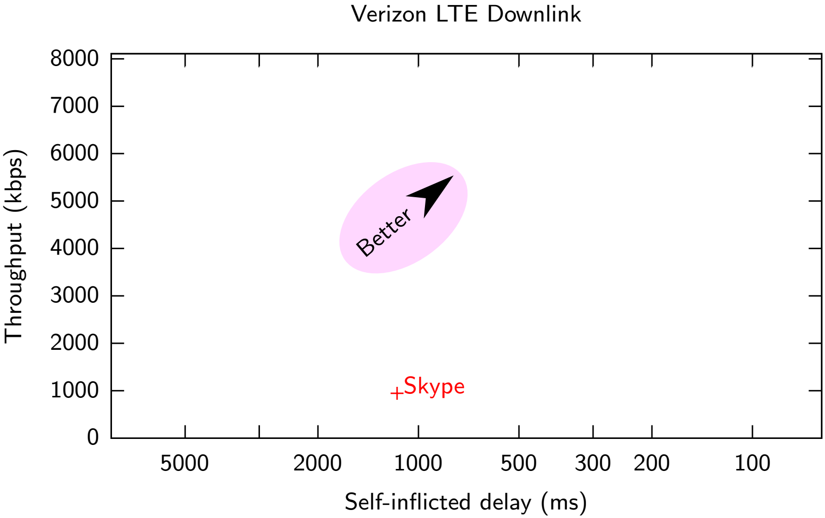


(Verizon LTE, one TCP download.)

Interactive apps work **poorly**

- We measured cellular networks while driving:
  - **Verizon LTE**
  - Verizon 3G (1xEV-DO)
  - AT&T LTE
  - T-Mobile 3G (UMTS)
- Then ran apps across replayed network trace:
  - **Skype** (Windows 7)
  - Google Hangout (Chrome on Windows 7)
  - Apple Facetime (OS X)

Performance summary



Why is performance so bad?

- Exiting schemes **react** to congestion signals.
  - Packet loss.
  - Increase in round-trip time.
- Feedback comes too late.
- The killer: **self-inflicted queueing delay**.
- Throughput overshoot means a queue filling up.

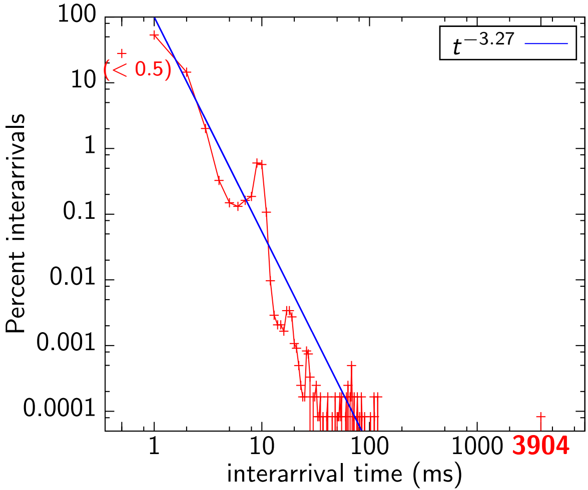
Sprout's goal

- Most throughput
- Bounded risk of delay $>$ 100 ms

Bounded risk of delay

- **Infer** link speed from interarrival distribution.
- **Predict** future link speed.
  - Don't wait for congestion.
- **Control:** Send as much as possible, but require:
  - 95% chance all packets arrive within 100 ms.

Infer: link speed from flicker noise process

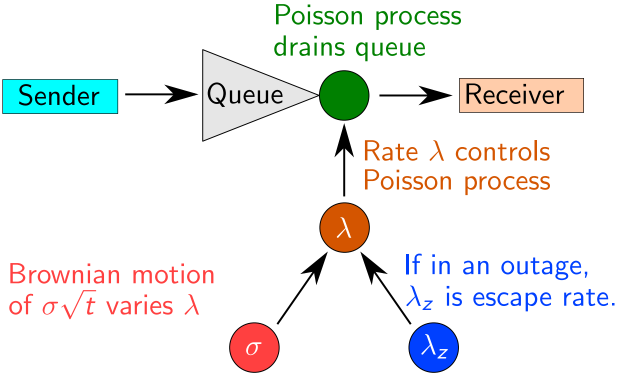


(Verizon LTE, phone stationary.)

Predict: future link speed

- Model evolution of speed as **random walk**.
  - (Brownian motion)
- Cautious forecast: 5th percentile cumulative packets
- Receiver makes forecast; sends back to sender in ack
- Almost all precalculated

Sprout's model

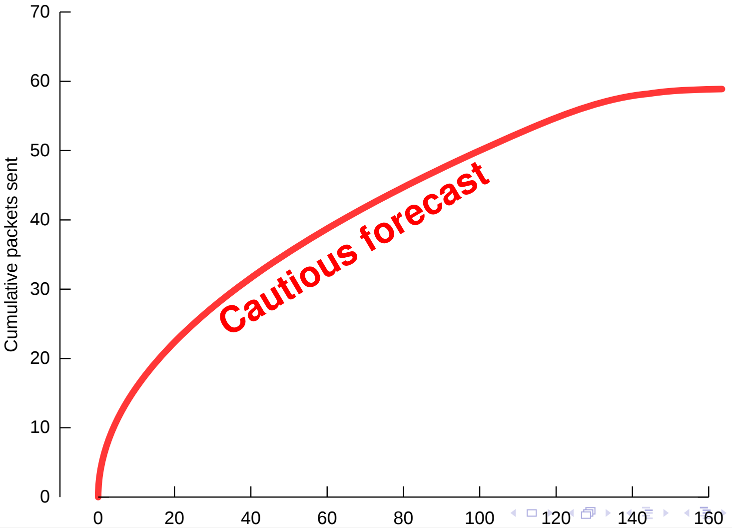


Parameters

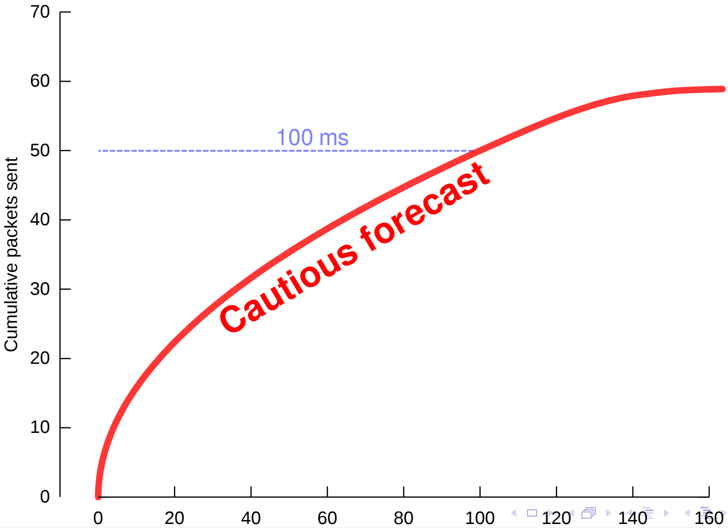
Volatility σ: fixed @	$200\ \frac{\text{pkts}/s}{\sqrt{s}}$
Expected outage time $1/\lambda_z$:	$1\ s$
Tick length:	$20\ ms$
Forecast length:	$160\ ms$
Delay target:	$100\ ms$
Risk tolerance:	5%

All source code was **frozen before data collection began**.

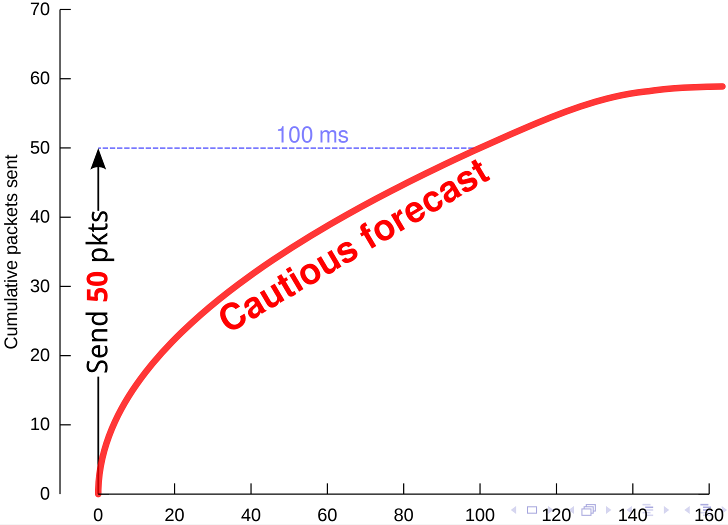
Control: fill up 100 ms forecast window



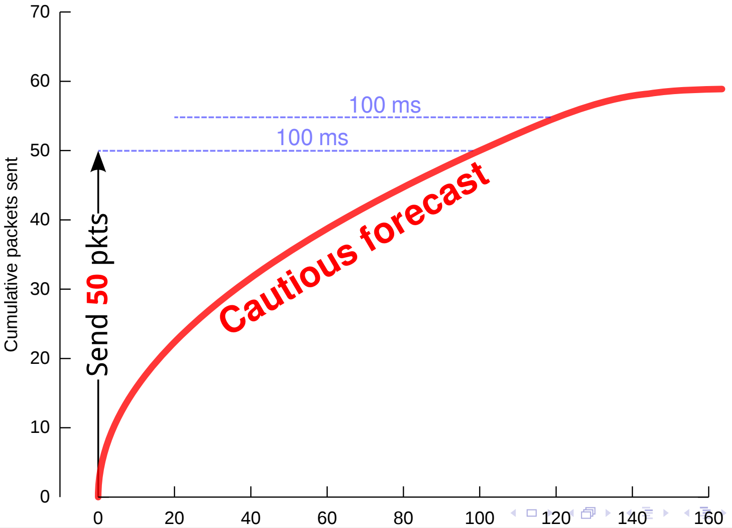
Control: fill up 100 ms forecast window



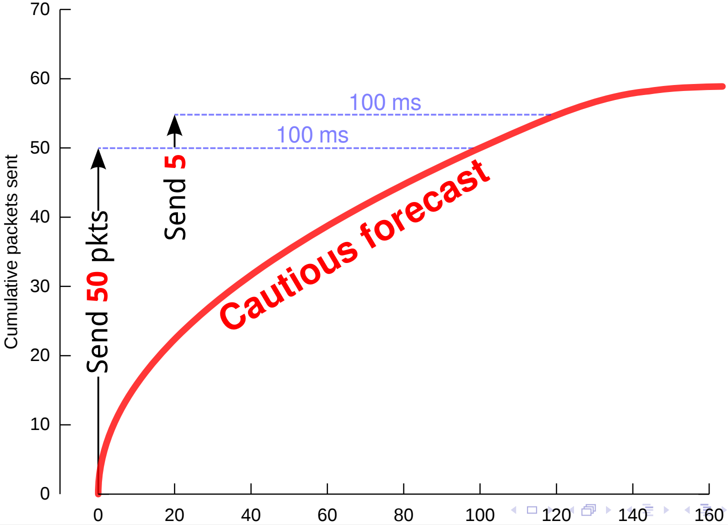
Control: fill up 100 ms forecast window



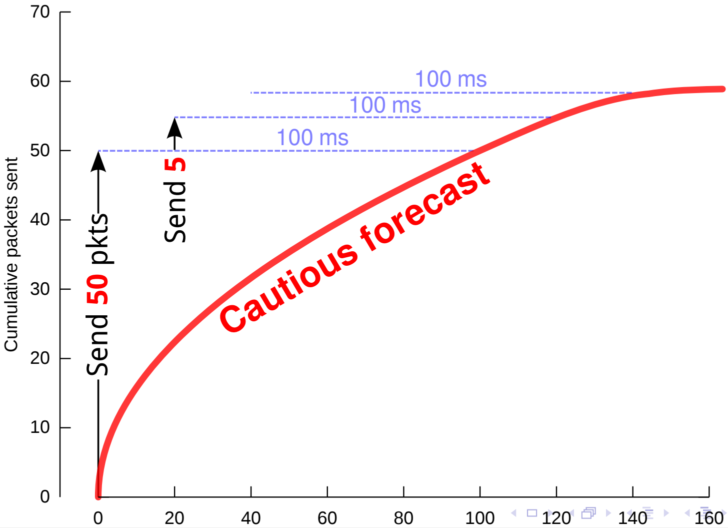
Control: fill up 100 ms forecast window



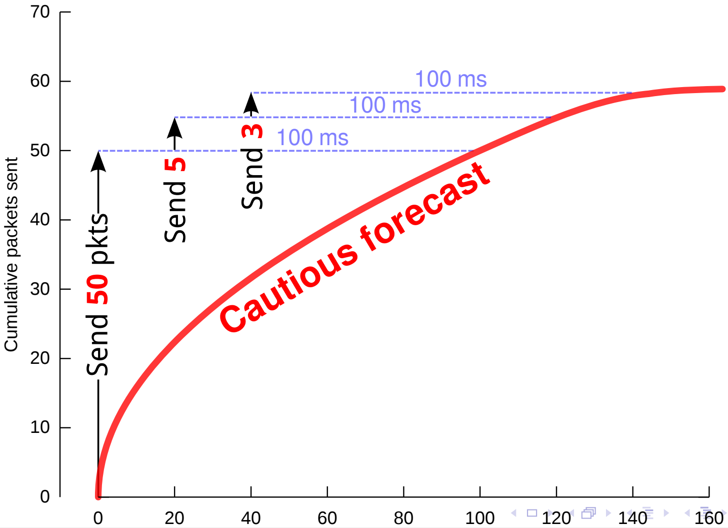
Control: fill up 100 ms forecast window

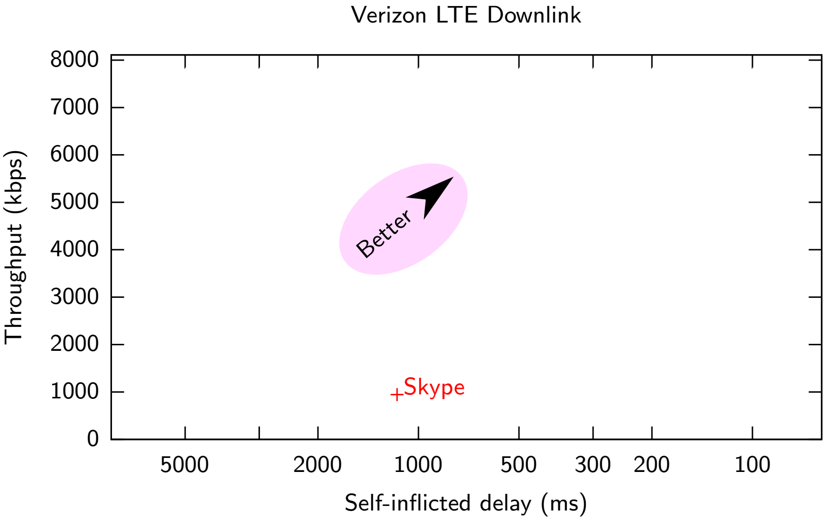


Control: fill up 100 ms forecast window

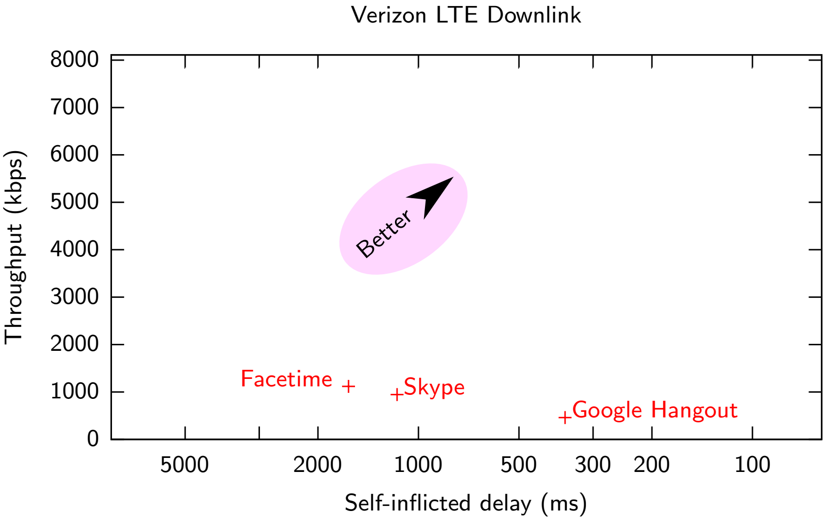


Control: fill up 100 ms forecast window

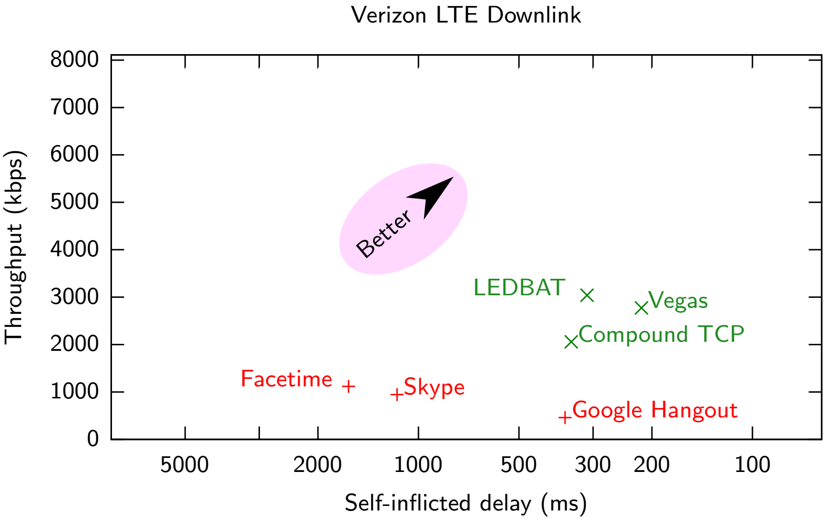


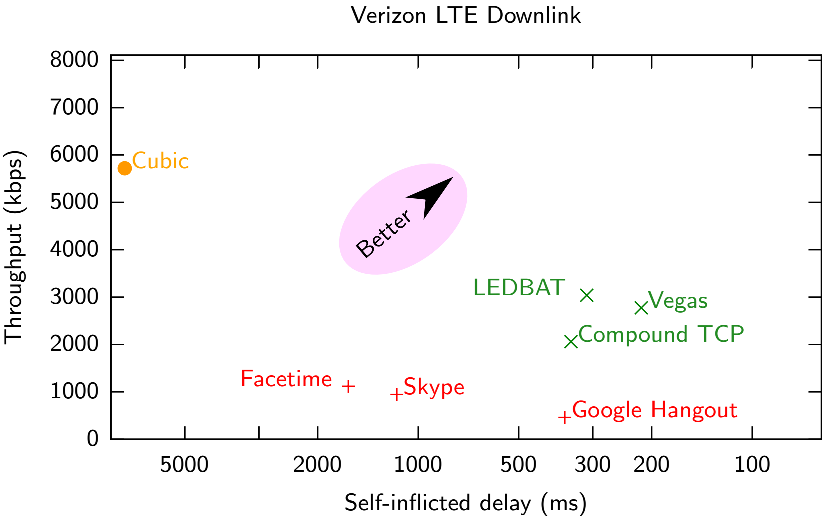


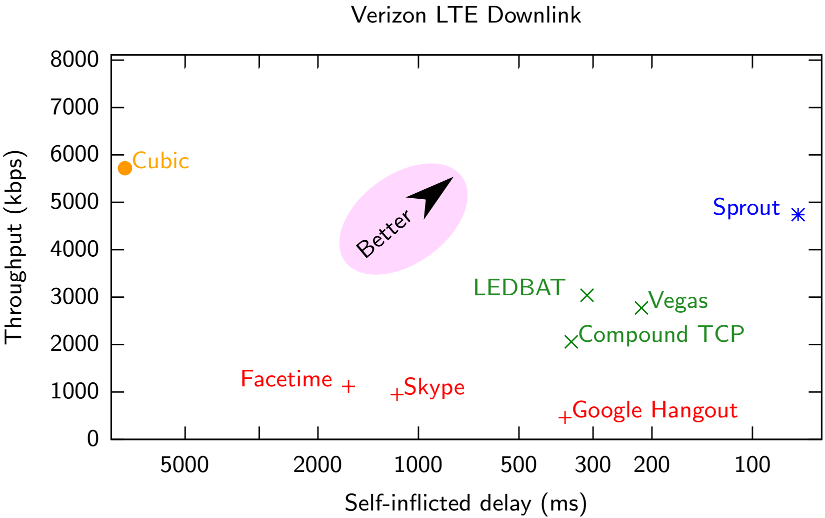
Verizon LTE Downlink
8000
7000
6000
5000
4000
3000
2000
1000
0
Throughput (kbps)
Better
Skype
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)

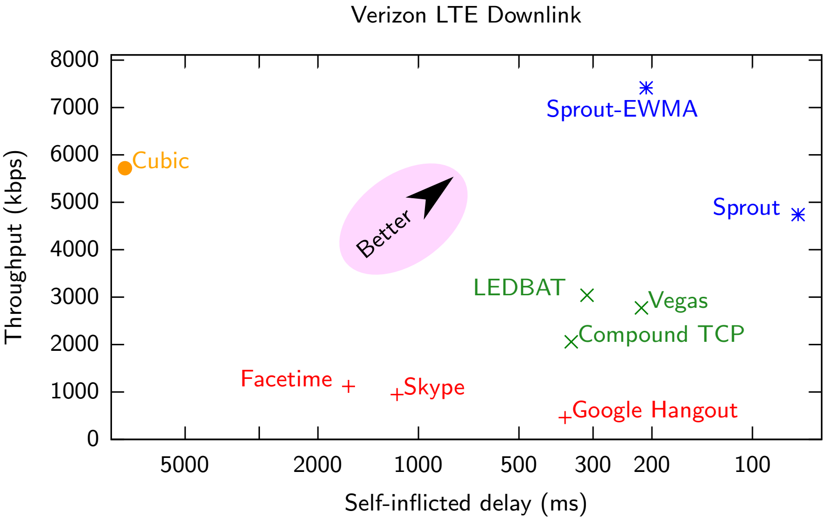


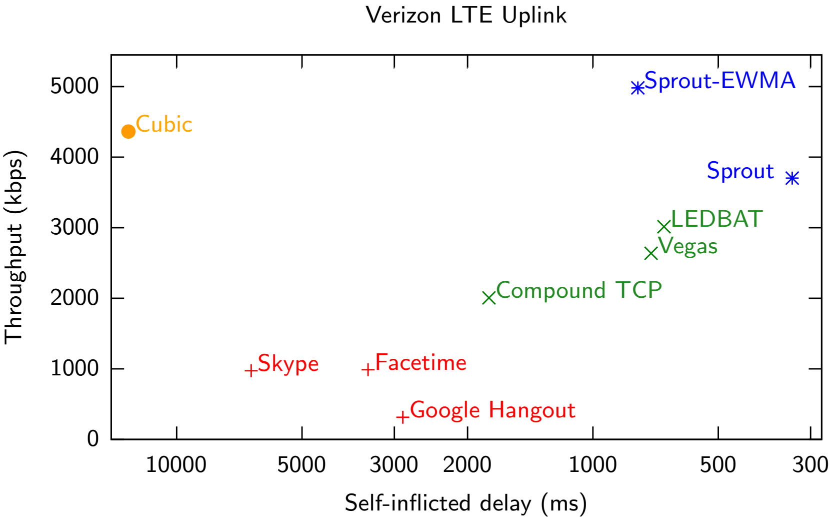
Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
Better
Facetime
Skype
Google Hangout
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)

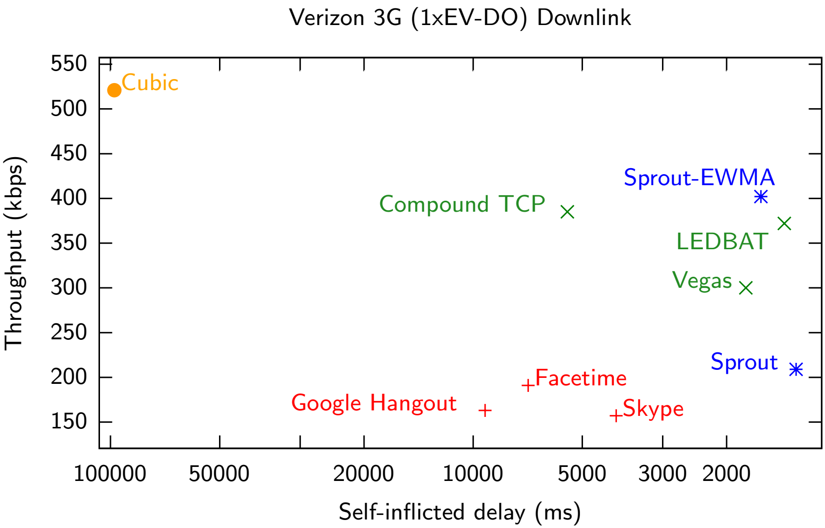

Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
Better
LEDBAT
Vegas
Compound TCP
Facetime
Skype
Google Hangout
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)

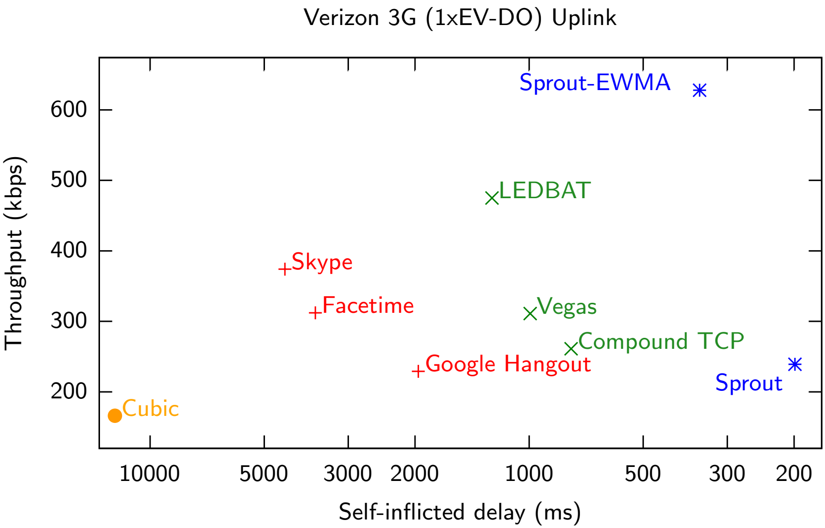

Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
Cubic
Better
LEDBAT
Vegas
Compound TCP
Facetime
Skype
Google Hangout
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)

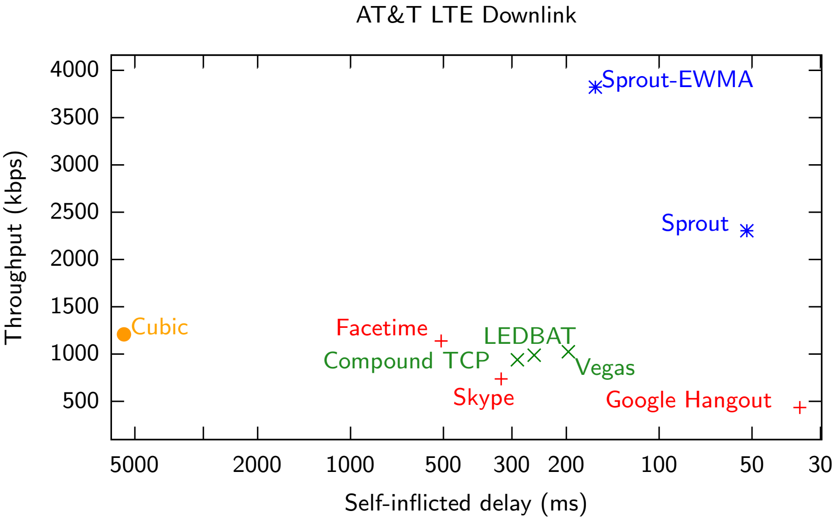

Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)
Cubic
Better
Sprout
LEDBAT
Vegas
Compound TCP
Facetime
Skype
Google Hangout

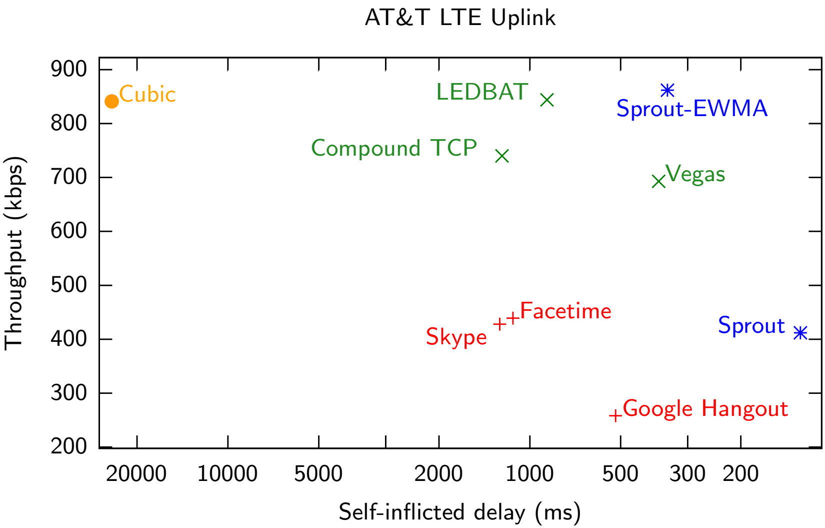

Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
Self-inflicted delay (ms)
5000
2000
1000
500
300
200
100
Sprout-EWMA
Cubic
Better
Sprout
LEDBAT
Vegas
Compound TCP
Facetime
Skype
Google Hangout

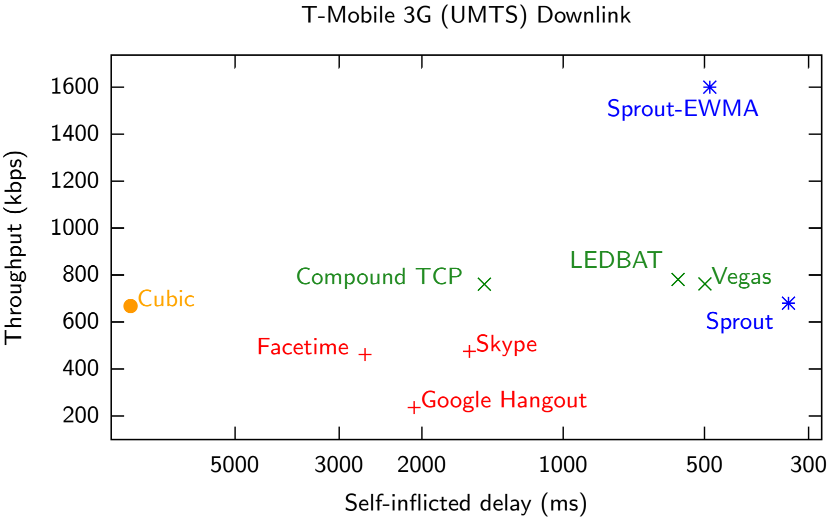

Verizon LTE Uplink
Throughput (kbps)
5000
4000
3000
2000
1000
0
Self-inflicted delay (ms)
10000
5000
3000
2000
1000
500
300
Cubic
Sprout-EWMA
Sprout
LEDBAT
Vegas
Compound TCP
Skype
Facetime
Google Hangout

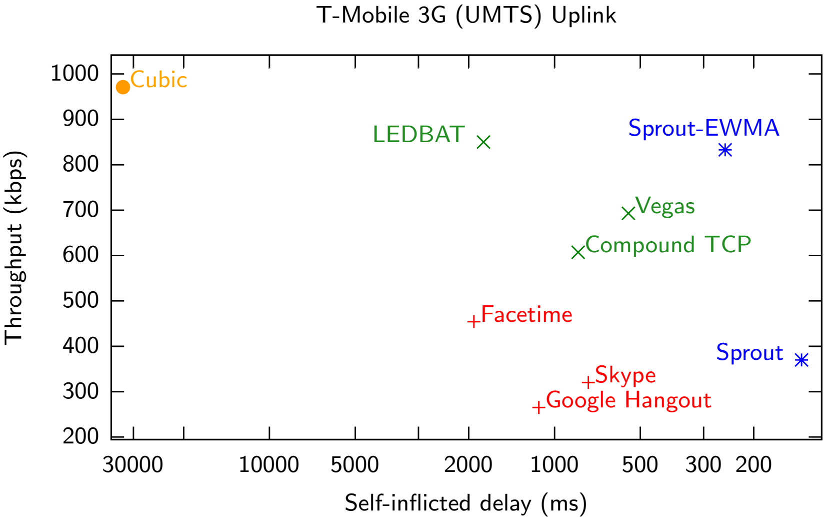

Verizon 3G (1xEV-DO) Downlink
Throughput (kbps)
Self-inflicted delay (ms)
550
500
450
400
350
300
250
200
150
100000
50000
20000
10000
5000
3000
2000
Cubic
Sprout-EWMA
Compound TCP
LEDBAT
Vegas
Sprout
Facetime
Google Hangout
Skype


Verizon 3G (1xEV-DO) Uplink
Sprout-EWMA
LEDBAT
Skype
Facetime
Vegas
Compound TCP
Google Hangout
Sprout
Cubic
600
500
400
300
200
10000
5000
3000
2000
1000
500
300
200
Throughput (kbps)
Self-inflicted delay (ms)


AT&T LTE Downlink
Throughput (kbps)
4000
3500
3000
2500
2000
1500
1000
500
5000
2000
1000
500
300
200
100
50
30
Self-inflicted delay (ms)
Sprout-EWMA
Sprout
Cubic
Facetime
LEDBAT
Compound TCP
Vegas
Skype
Google Hangout


AT&T LTE Uplink
Throughput (kbps)
900
800
700
600
500
400
300
200
Self-inflicted delay (ms)
20000
10000
5000
2000
1000
500
300
200
Cubic
LEDBAT
Sprout-EWMA
Compound TCP
Vegas
Facetime
Skype
Sprout
Google Hangout

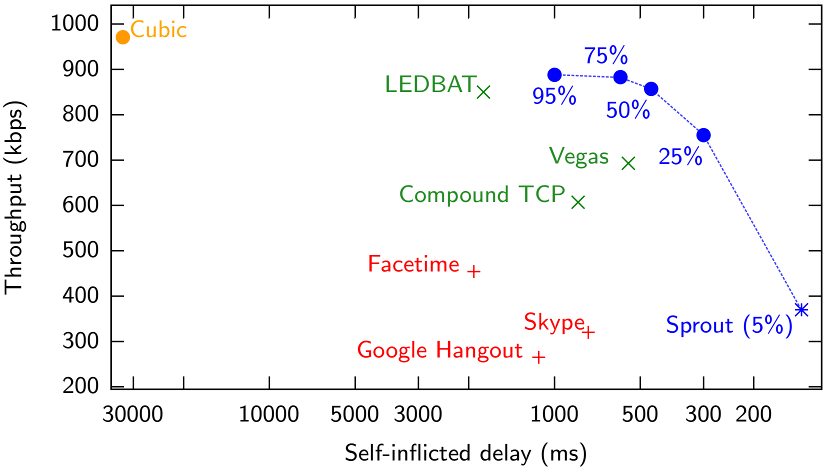

T-Mobile 3G (UMTS) Downlink
Throughput (kbps)
1600
1400
1200
1000
800
600
400
200
Self-inflicted delay (ms)
5000
3000
2000
1000
500
300
Sprout-EWMA
Sprout
LEDBAT
Vegas
Compound TCP
Cubic
Facetime
Skype
Google Hangout


T-Mobile 3G (UMTS) Uplink
Throughput (kbps)
1000
900
800
700
600
500
400
300
200
30000
10000
5000
2000
1000
500
300
200
Self-inflicted delay (ms)
Cubic
LEDBAT
Sprout-EWMA
Vegas
Compound TCP
Facetime
Sprout
Skype
Google Hangout

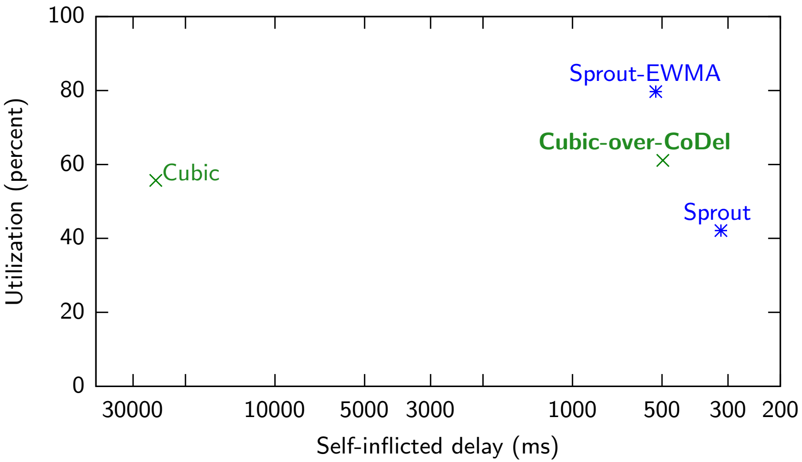
Overall results

Sprout vs.	Avg. speedup	Delay reduction
Skype	2.2×	7.9×
Hangout	4.4×	7.2×
Facetime	1.9×	8.7×
Compound	1.3×	4.8×
TCP Vegas	1.1×	2.1×
LEDBAT	Same	2.8×
Cubic	0.91×	79×

Varying risk tolerance



Competes with AQM even though end-to-end



Competing traffic **inside** Sprout tunnel

	Direct	via Sprout	Benefit
Cubic throughput	8336 kbps	3776 kbps	0.5× (= worse)
Skype throughput	78 kbps	490 kbps	6×
Skype 95% delay	6.0 s	0.17 s	35×

Replication by Stanford students (February–March 2013)

- Alterman & Quach reproduced some of our measurements
- `http://ReproducingNetworkResearch.wordpress.com/2013/03/12/1216/`
- Won best project award in Stanford networking class!

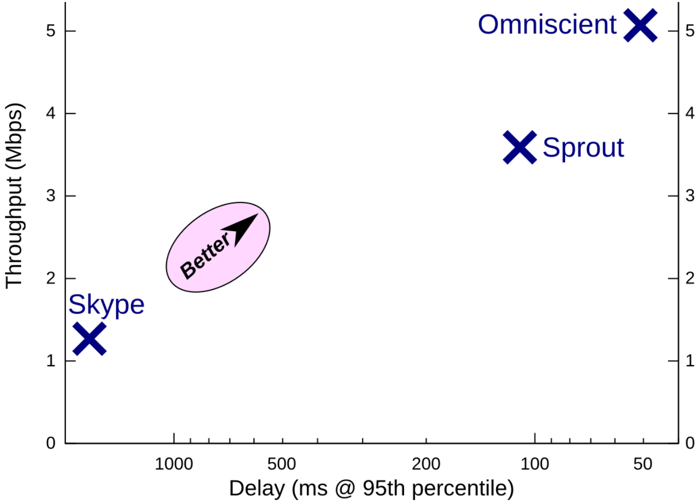
M.I.T. 6.829 contest (March–April 2013)

- Turnkey network emulator, evaluation
- Sender, receiver run in Linux containers
- 4th prize: $20
- 3rd prize: $30
- 2nd prize: $40
- (If beat Sprout) 1st prize:

M.I.T. 6.829 contest (March–April 2013)

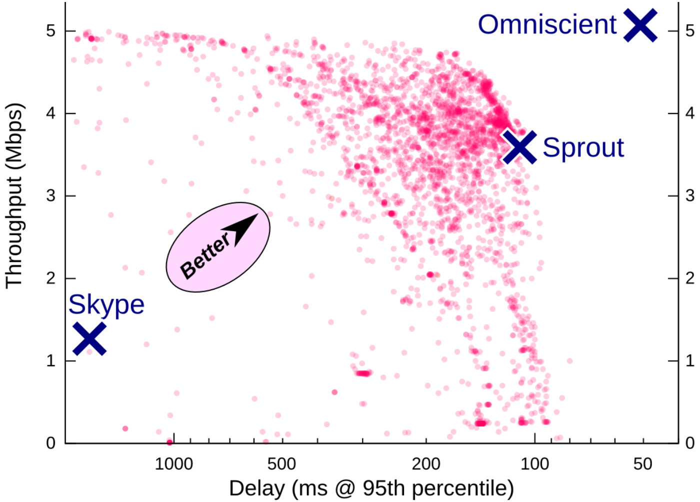
- Turnkey network emulator, evaluation
- Sender, receiver run in Linux containers
- 4th prize: $20
- 3rd prize: $30
- 2nd prize: $40
- (If beat Sprout) 1st prize: **Co-authorship on future paper**

Baseline

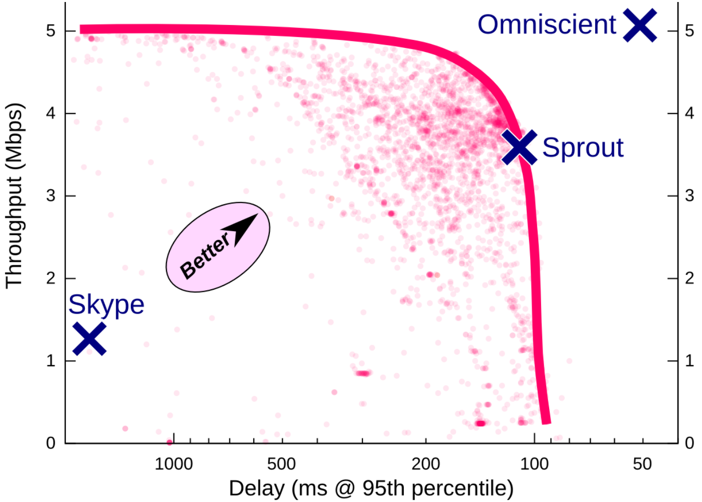


Omniscient
Sprout
Skype
Better
Throughput (Mbps)
0
1
2
3
4
5
1000
500
200
100
50
Delay (ms @ 95th percentile)

Land of 3,000 student protocols



Sprout is on the frontier



Limitations

- Only evaluated long-running flows.
- All testing data from Boston.
- User should wrap competing flows inside Sprout.
- If queue is full of another user's packets, an end-to-end scheme can't help.
  - Fortunately, cells have per-device queues...
  - ...but Wi-Fi generally doesn't.
- What about when the cell link *isn't* the bottleneck?

Our approach

- Pick a model, any model.
- All models are wrong, but they help anyway!
- See if it lands on the frontier.*

* (On a large set of real network paths or newly-collected traces.)

- Kaizen for congestion

Thank you

- Lakshminarayanan Subramanian
- Shuo Deng
- Jonathan Perry
- Katrina LaCurts
- Andrew McGregor
- Tim Shepard
- Dave Täht
- Michael Welzl
- Hannes Tschofenig
- Wireless@MIT members (http://wireless.csail.mit.edu)
- NSF & Shannon family (fellowship)

Sprout for controlled delay over cellular networks

- **Infer** link speed from interarrival distribution
- **Predict** future link speed
- **Control** risk of large delay with cautious forecast
- Yields 2–4× throughput of Skype, Facetime, Hangout
- Achieves 7–9× reduction in self-inflicted delay
- Matches active queue management **without router changes**
- Code and directions at `http://alfalfa.mit.edu`
- `alfalfa@mit.edu`