Transport Architectures for an Evolving Internet

Keith Winstein

MIT Computer Science and Artificial Intelligence Laboratory

March 5, 2014

CSAIL

Joint work with Anirudh Sivaraman, Pratiksha Thaker, and Hari Balakrishnan.

The Internet evolves

In 20 years, computer networks have seen dramatic change:

- Wi-Fi
- Cellular networks
- Datacenters
- 10 GigE
- Transoceanic links

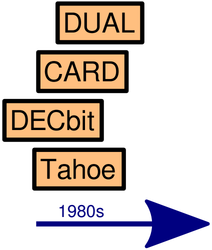
- Ubiquitous mobility
- Huge amounts of streaming video

Coping with change

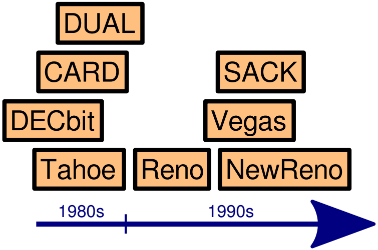
How should users deal with an evolving network?

One approach: design new protocols.

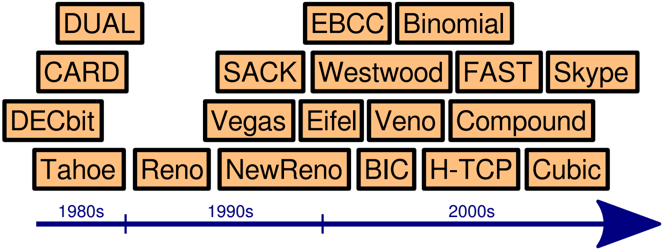
The march of congestion-control protocols



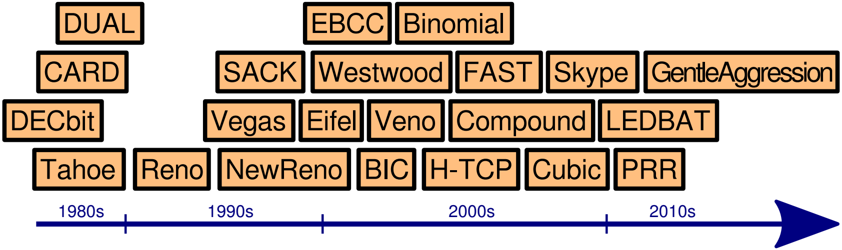
The march of congestion-control protocols



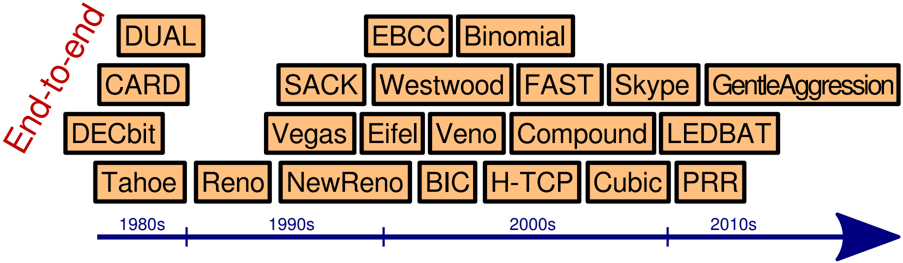
The march of congestion-control protocols



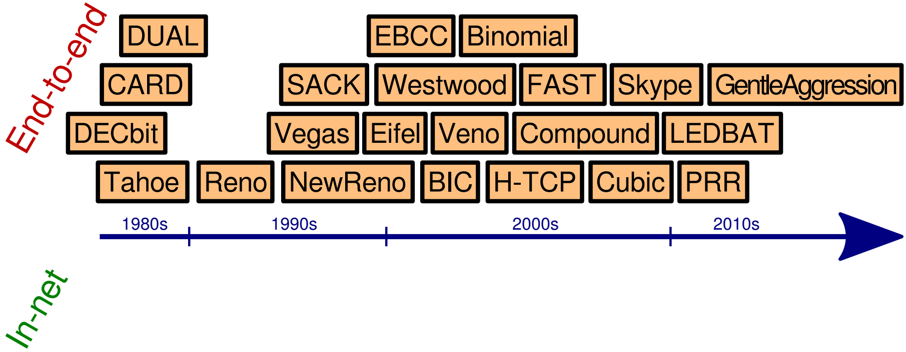
The march of congestion-control protocols



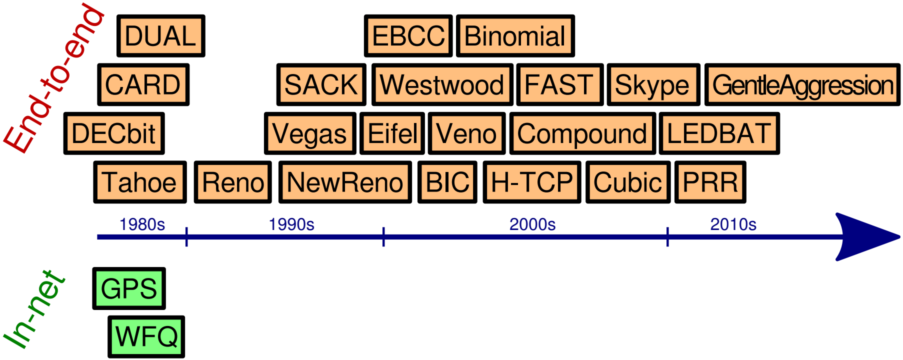
The march of congestion-control protocols



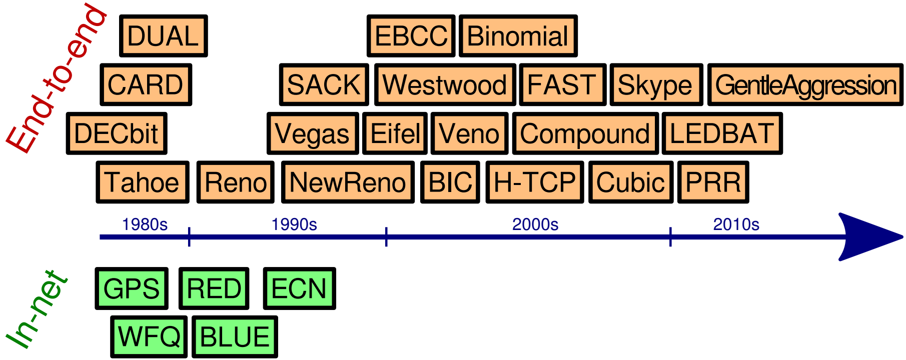
The march of congestion-control protocols



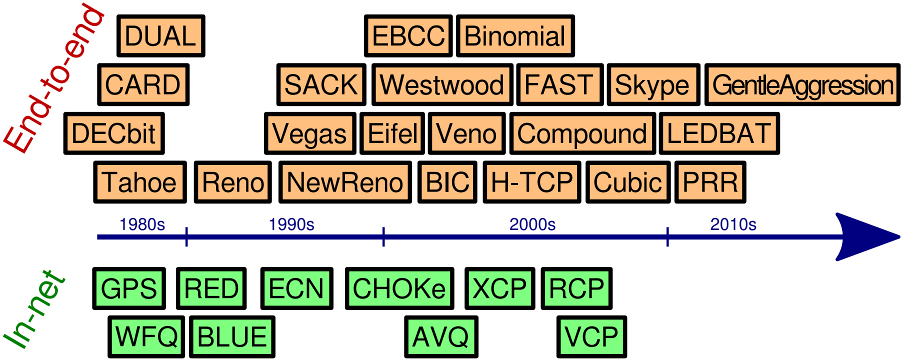
The march of congestion-control protocols



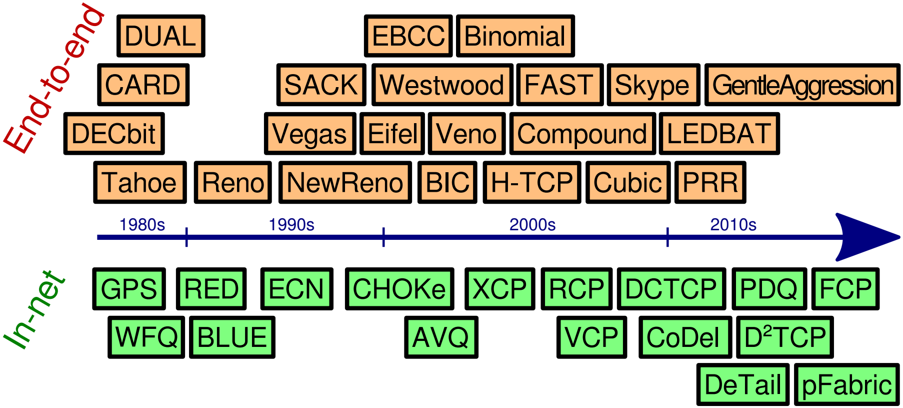
The march of congestion-control protocols



The march of congestion-control protocols



The march of congestion-control protocols



Declarative design

Systems with a **model** and a **mission**.

Explicitness in systems design

Model: explicit statement of assumptions about the problem

Mission: objective that the application wants

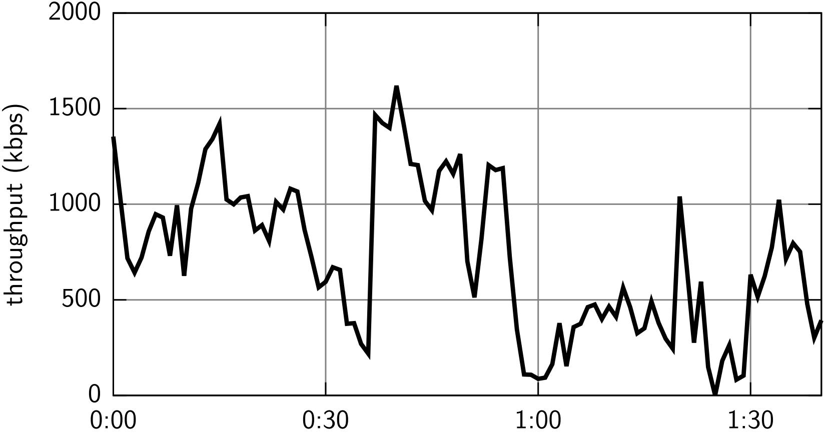
Explicit design considerations → **freedom to make changes**

Sprout: a transport protocol designed for variability

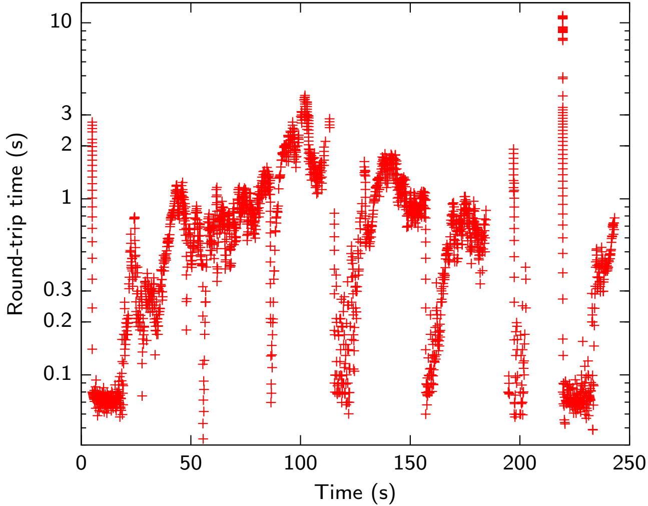
Observation:

Videoconferences perform poorly over cellular networks.

Verizon LTE uplink throughput



Verizon LTE ping delay during one TCP download

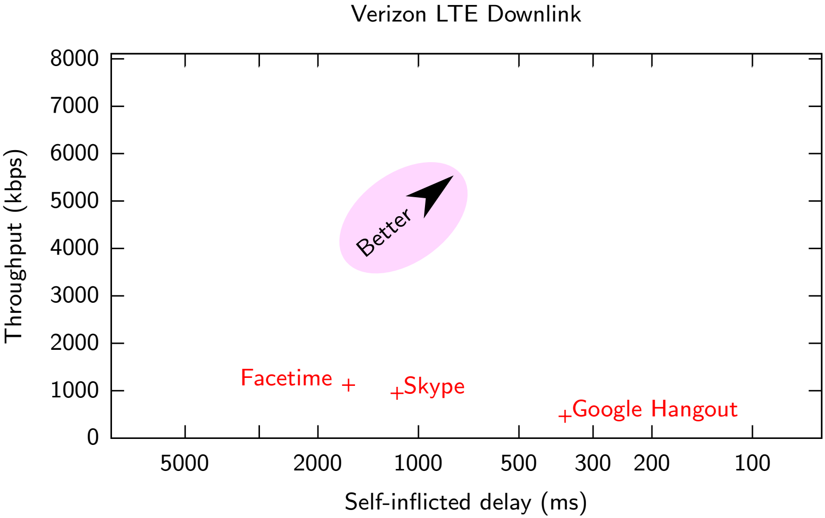


Interactive apps work poorly

- 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 Hangouts (Chrome on Windows 7)
  - Apple Facetime (OS X)

Skype's performance

Performance summary



What's wrong?

- Existing schemes **react** to congestion signals.
  - Packet loss.
  - Increase in round-trip time.
- Feedback comes too late.
- The killer: **self-inflicted queueing delay**.

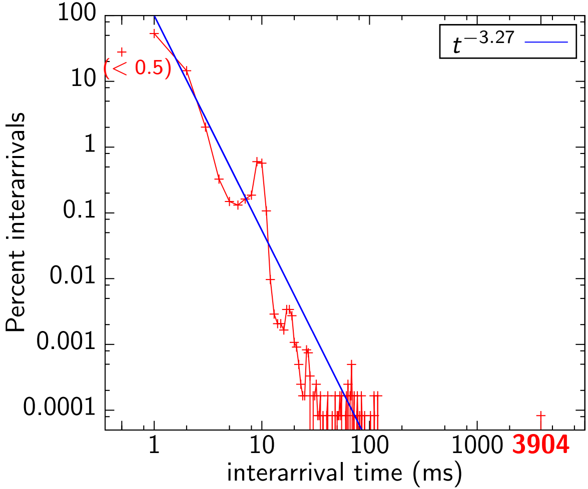
Sprout's **mission**

- Most throughput
- Bounded risk of delay > 100 ms

Bounded risk of delay

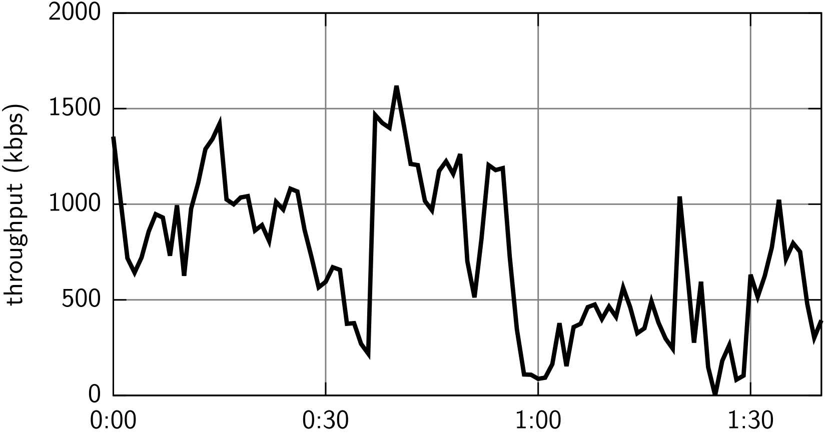
- **Model** variation in link speed
- **Infer** current link speed
- **Predict** future link speed
  - Don't wait for congestion
- **Control:** Send as much as possible, but require:
  - 95% chance all packets arrive within 100 ms

Model: packet deliveries looks like flicker noise

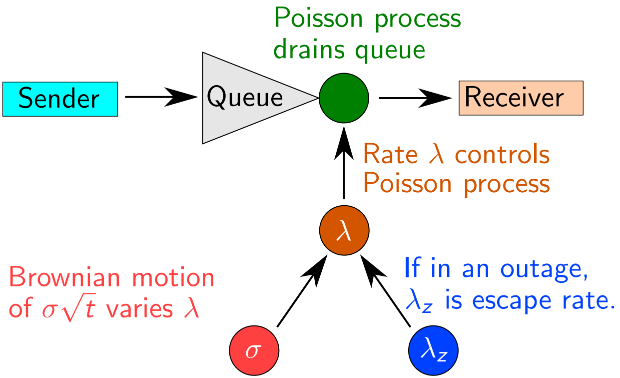


(Verizon LTE, phone stationary.)

Model: average rate looks like random walk



Sprout's **model**



Sprout's model parameters

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

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

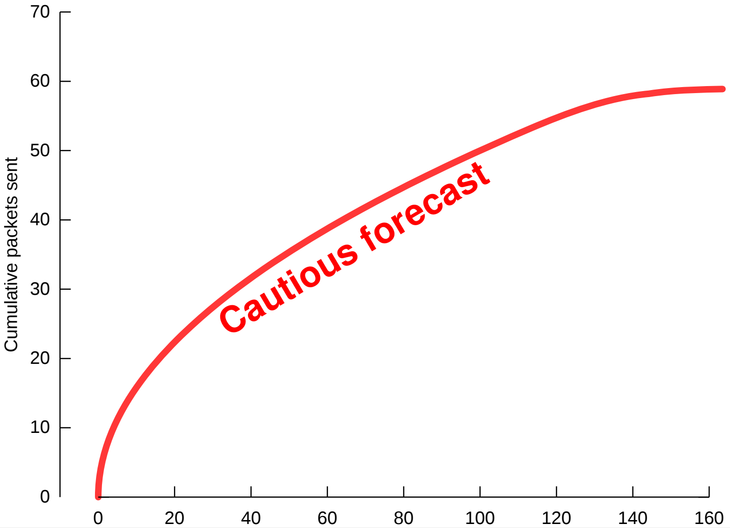
Infer: current link speed

- **Observe** packets received every τ
- **Update** P(λ)

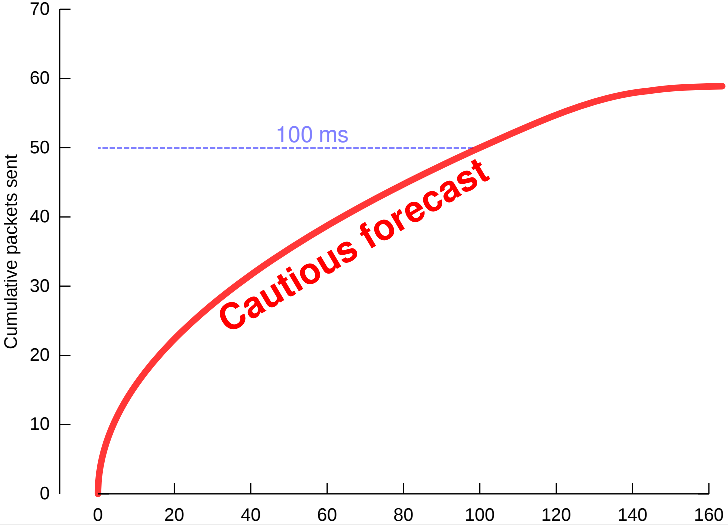
Predict: future link speed

- **Evolve** model forward
- **Forecast** 5th percentile cumulative packets

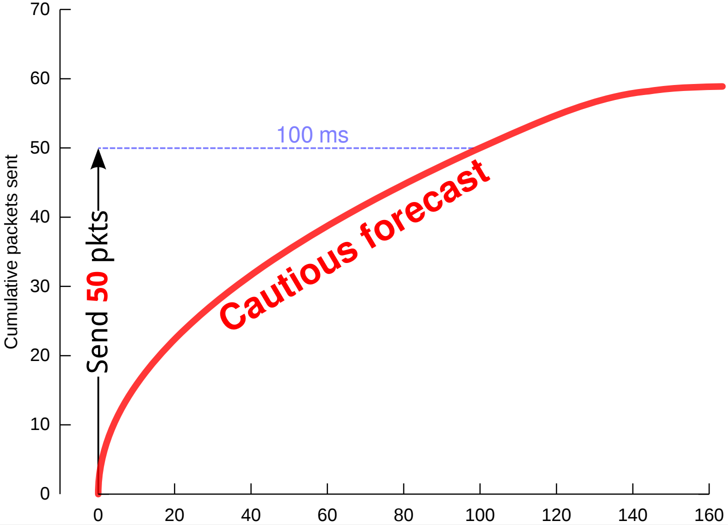
Control: fill up 100 ms forecast window



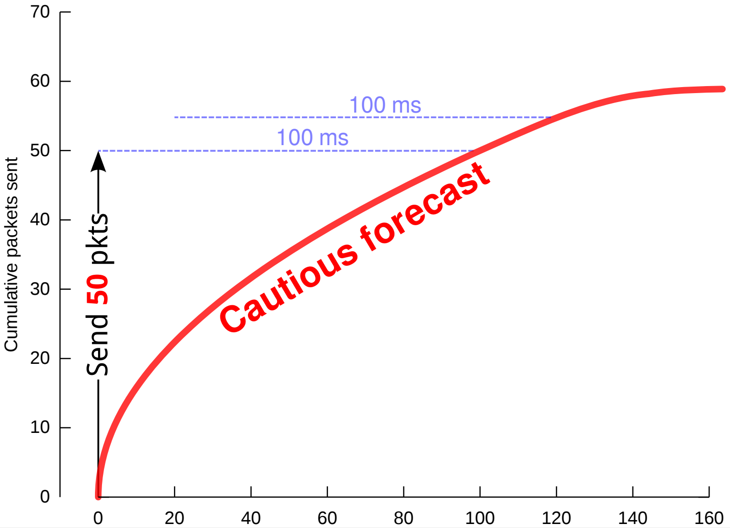
Control: fill up 100 ms forecast window



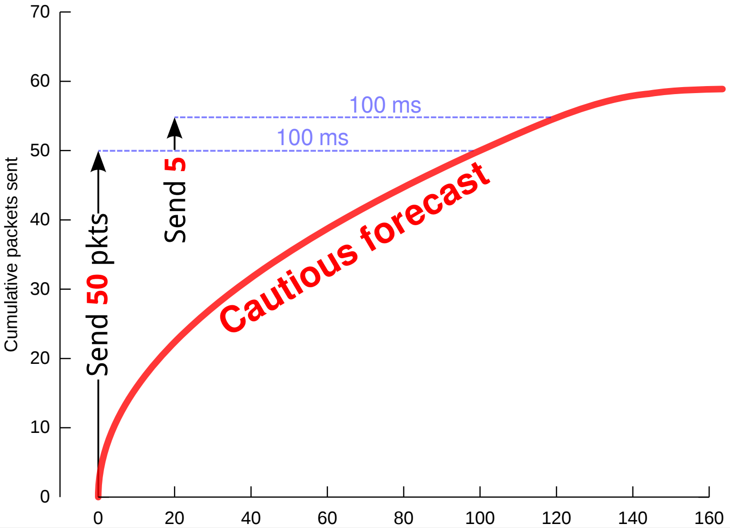
Control: fill up 100 ms forecast window



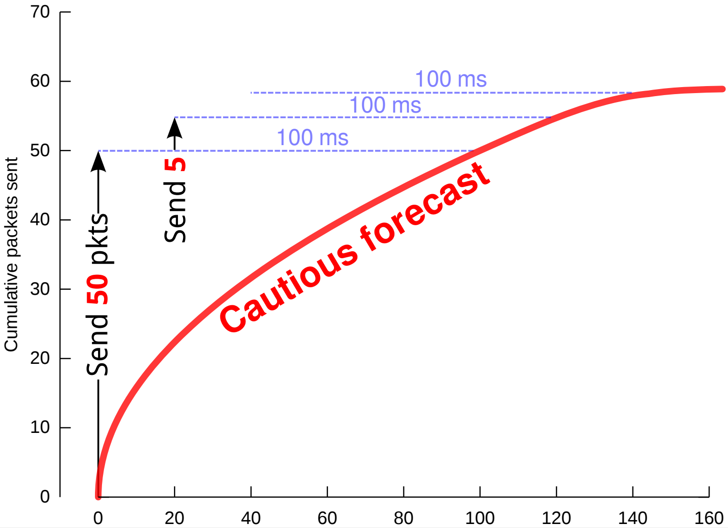
Control: fill up 100 ms forecast window



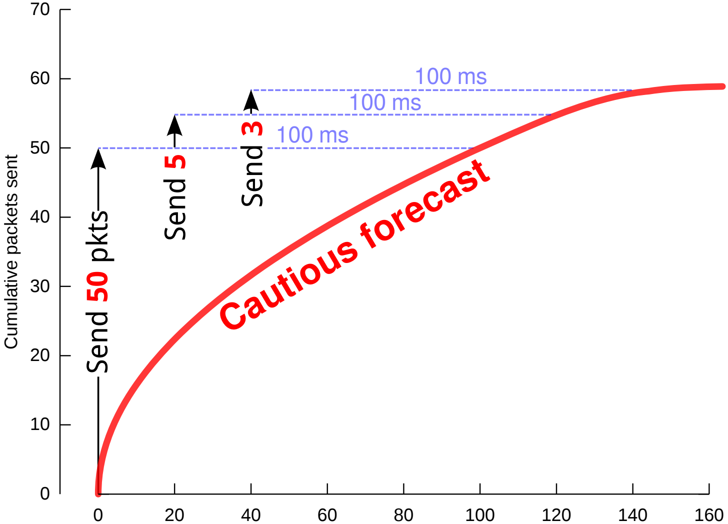
Control: fill up 100 ms forecast window



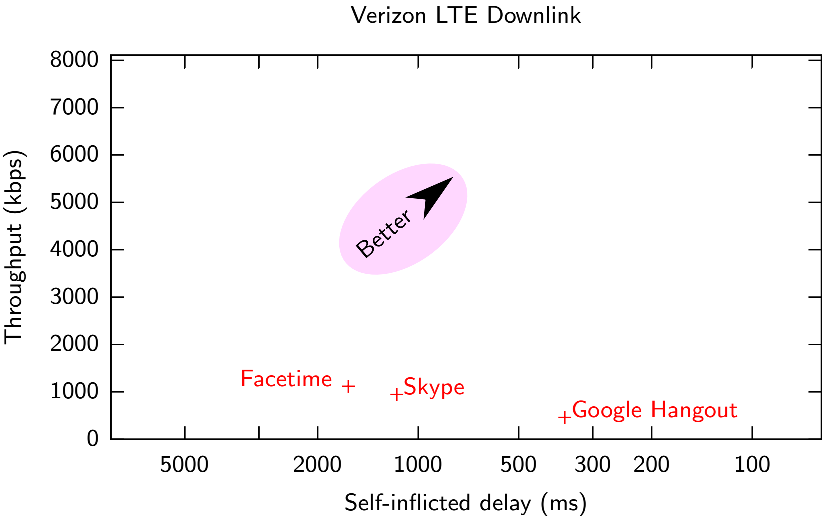
Control: fill up 100 ms forecast window

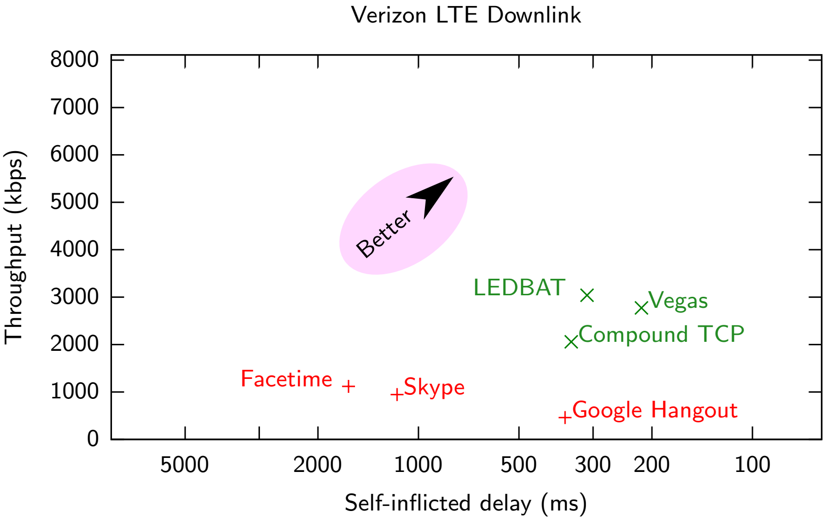


Control: fill up 100 ms forecast window

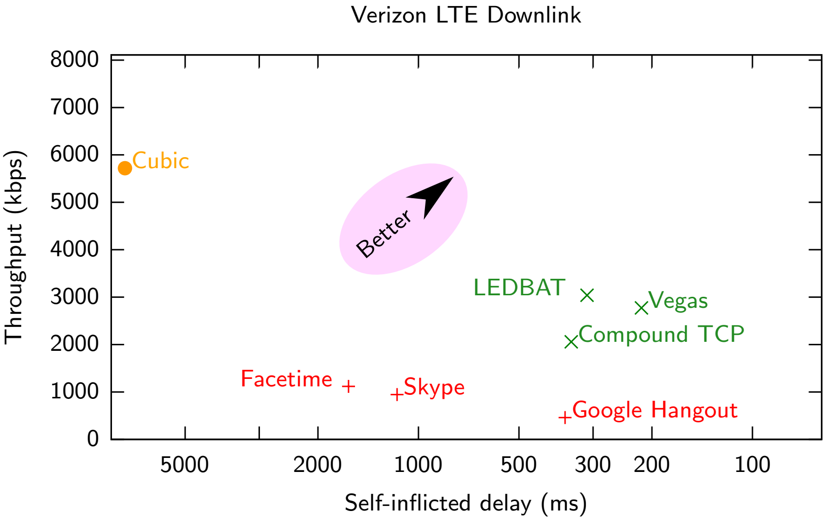


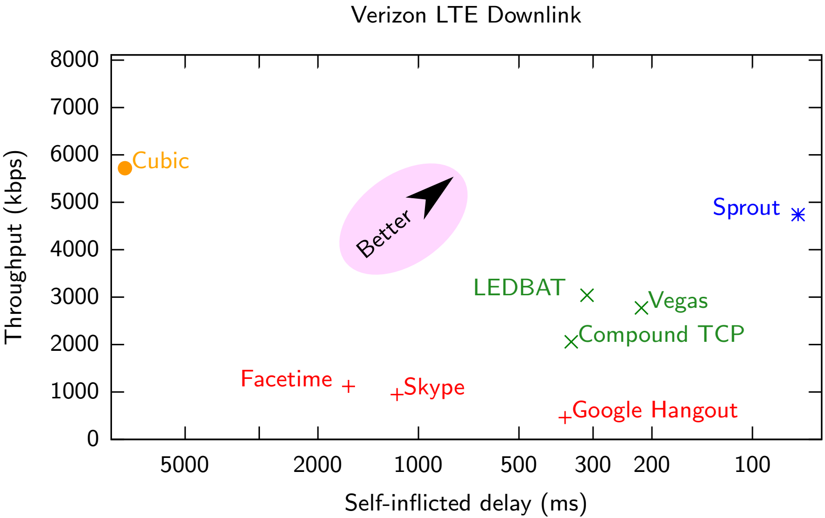
Sprout's results

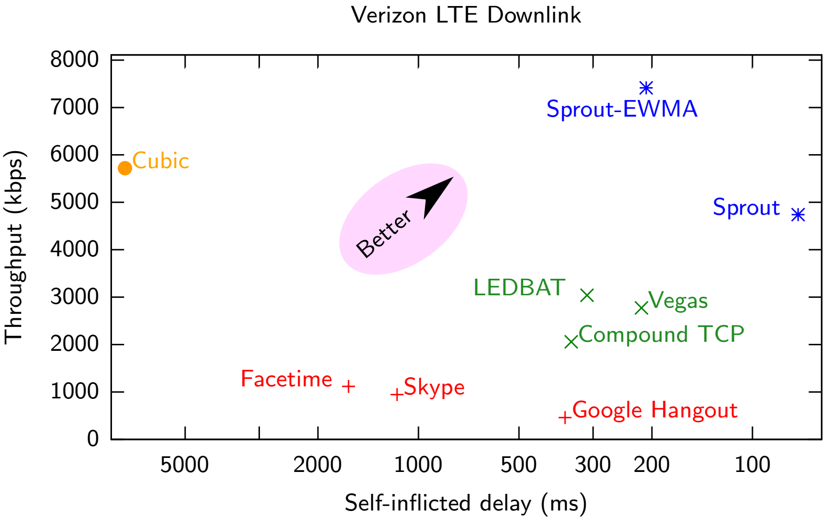

Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
Better
Facetime
Skype
Google Hangout
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)

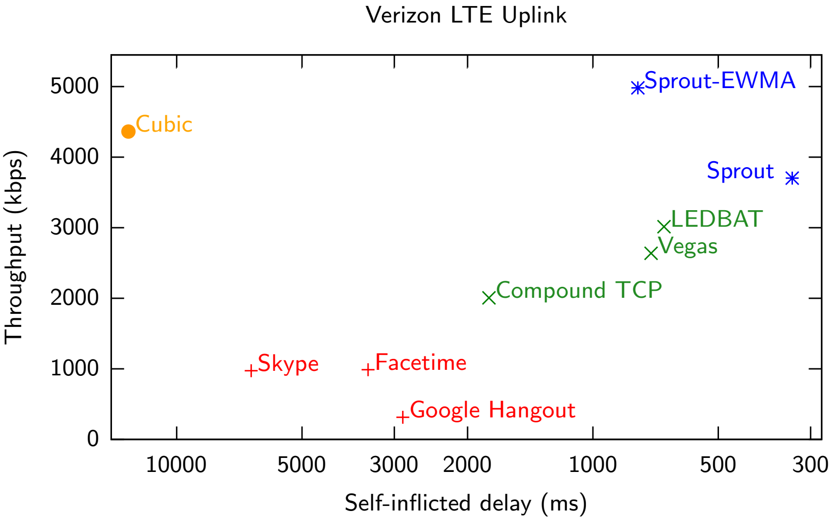


Verizon LTE Downlink
Throughput (kbps)
8000
7000
6000
5000
4000
3000
2000
1000
0
Better
LEDBAT
Vegas
Compound TCP
Facetime
Skype
Google Hangout
5000
2000
1000
500
300
200
100
Self-inflicted delay (ms)


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


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

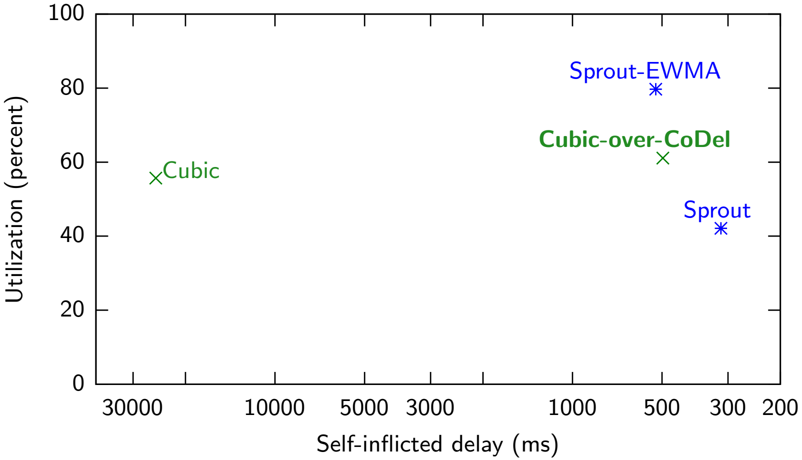

Verizon LTE Uplink
Throughput (kbps)
5000
4000
3000
2000
1000
0
10000
5000
3000
2000
1000
500
300
Self-inflicted delay (ms)
Sprout-EWMA
Cubic
Sprout
LEDBAT
Vegas
Compound TCP
Skype
Facetime
Google Hangout

Overall results on 8 links

Verizon 3G/LTE, AT&T LTE, T-Mobile 3G uplink and downlink:

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×

Sprout is end-to-end, but comparable to in-net control



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

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

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

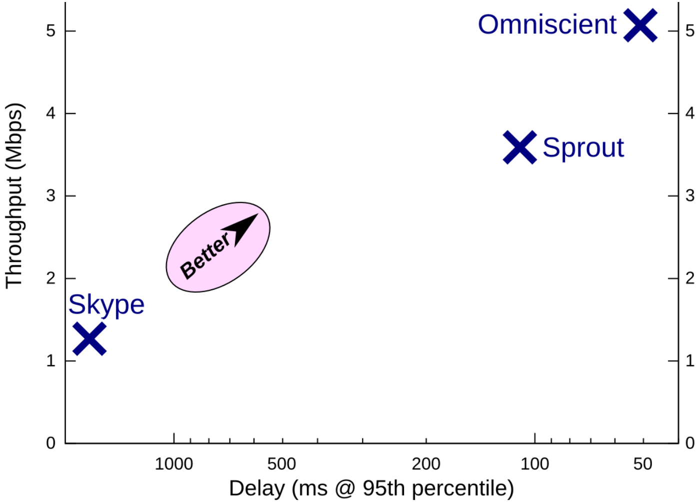
- Turnkey network emulator, evaluation
- Sender, receiver run in Linux containers
- **Mission**: maximize throughput/delay
- 4th prize: $20
- 3rd prize: $30
- 2nd prize: $40
- (If beat Sprout) 1st prize: **Co-authorship on future paper**

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

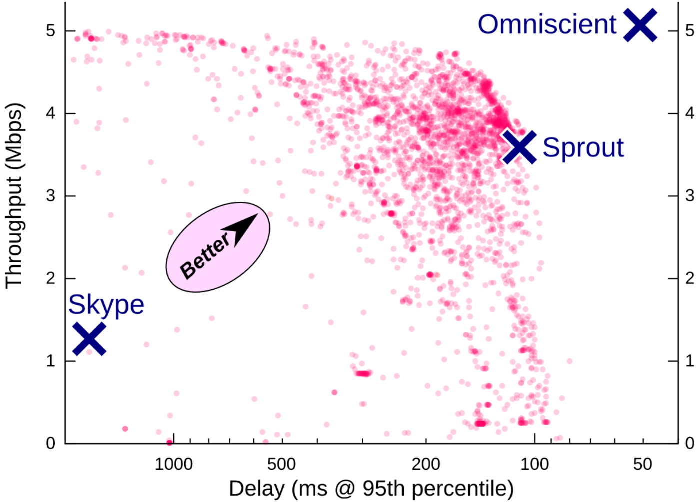
- Turnkey network emulator, evaluation
- Sender, receiver run in Linux containers
- **Mission**: maximize throughput/delay
- 4th prize: \$20
- 3rd prize: \$30
- 2nd prize: \$40
- (If beat Sprout) 1st prize: **Co-authorship on future paper**

Anirudh Sivaraman, KW, Pauline Varley, Somak Das, Joshua Ma, Ameesh Goyal, João Batalha, and Hari Balakrishnan, **Protocol Design Contests**, *in submission*

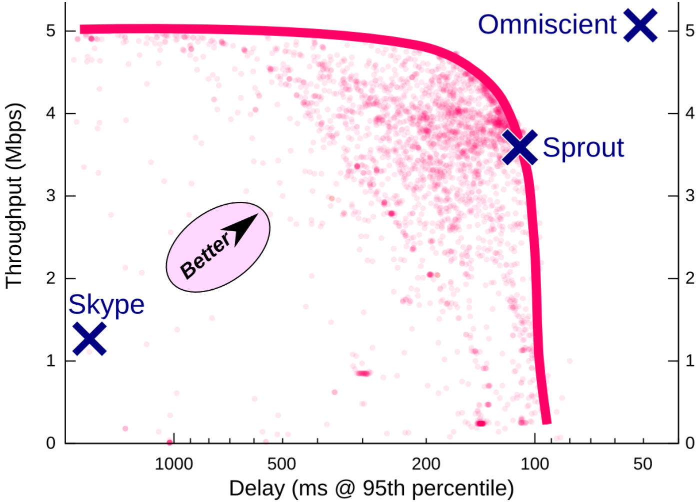
Baseline


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



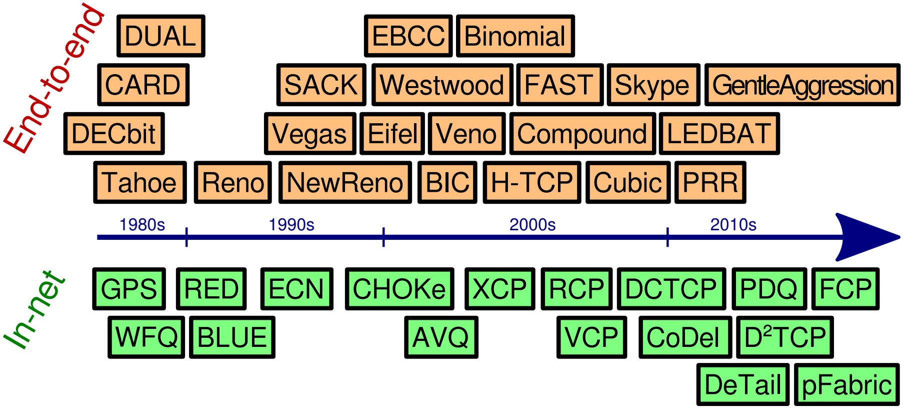
Sprout was on the frontier



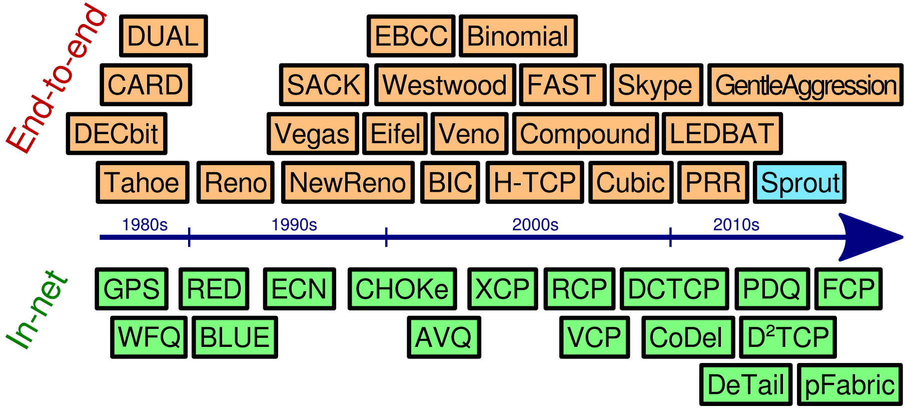
Limitations

- Sprout wants to control all of the traffic on a queue.
  - Cells generally have **per-user** queues. . .
  - . . . but Wi-Fi and wired networks usually don't.
- What if cell link *isn't* the bottleneck?
- Assumption: application always has data to send

Sprout's mark



Sprout's mark



Now that we have 40+ algorithms...

- Sprout for cellular networks?
- Wireless-TCP for Wi-Fi?
- High-BDP-TCP for transoceanic links?
- Datacenter-TCP for datacenters?
- CoDel for cable modems?
- **TBA-TCP for tomorrow's networks?**

Rational choice of scheme is challenging

$\boxed{\text{Cubic}}$ **vs.** $\boxed{\text{Compound}}$

- Different missions?
- Different assumptions about network?
- One scheme just plain better?

Networks constrained by a fuzzy idea of TCP's assumptions

- Mask stochastic loss
- Bufferbloat
- Mask out-of-order delivery
- No parallel/multipath routing

Advice for Internet Subnetwork Designers (RFC 3819) is 21,000 words!

Apps hack around TCP

- Open lots of flows
- Goose slow start
- Add pacing
- Give up and do it yourself

Apps hack around TCP

- Open lots of flows
- Goose slow start
- Add pacing
- Give up and do it yourself

Apps hack around TCP

- Open lots of flows
- Goose slow start
- Add pacing
- Give up and do it yourself

Apps hack around TCP

- Open lots of flows
- Goose slow start
- Add pacing
- Give up and do it yourself

Apps hack around TCP

- Open lots of flows
- Goose slow start
- Add pacing
- Give up and do it yourself

Chrome (QUIC)
BitTorrent (μTP)
Mosh (SSP)
IBM Aspera (fasp)

Idea: computer-generated protocols

Transport layer should adapt to **whatever**:

- network does
- application wants

Idea: computer-generated protocols

Transport layer should adapt to **whatever**:

- network does (**model**)
- application wants (**mission**)

What we built

Remy: a program that generates congestion-control schemes offline

Input:

- Assumptions about network and workload (**model**)
- Application's objective (**mission**)

Output: CC algorithm for a TCP sender (RemyCC)

Time: hours to days

The basic question of congestion control

At this moment, do I:

- send a packet
- not send a packet?

Missions of congestion control

Maximize

- $\sum_i \log\left[\text{throughput}_i\right]$ (proportionally fair throughput)

Missions of congestion control

Maximize

- $$\sum_i \log\left[\text{throughput}_i\right]$$ (proportionally fair throughput)
- $$\sum_i \log\left[\frac{\text{throughput}_i}{\text{delay}_i}\right]$$ (proportionally fair throughput/delay)

Missions of congestion control

Maximize

- $$\sum_i \log\left[\text{throughput}_i\right]$$ (proportionally fair throughput)
- $$\sum_i \log\left[\frac{\text{throughput}_i}{(\text{delay}_i)^{\delta}}\right]$$ (proportionally fair throughput/delay)

Missions of congestion control

Maximize

- $\sum_i \log\left[\text{throughput}_i\right]$ (proportionally fair throughput)
- $\sum_i \log\left[\frac{\text{throughput}_i}{\left(\text{delay}_i\right)^{\delta}}\right]$ (proportionally fair throughput/delay)
- $\min_i \text{throughput}_i$ (max-min throughput)

Minimize

- mean flow completion time
- page load time

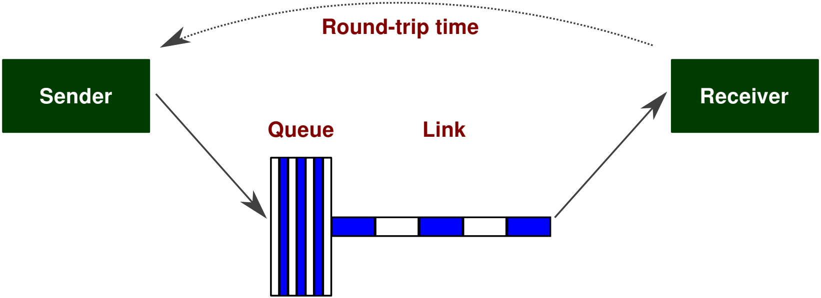
Prevent

- pathological behavior
- congestion collapse

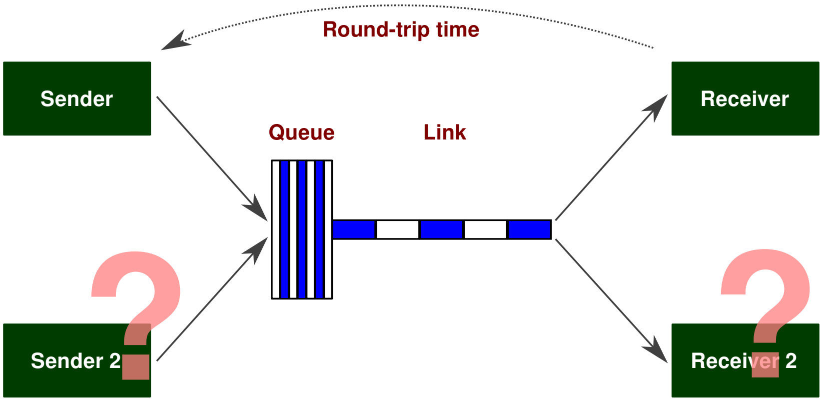
Encoding the designer's prior assumptions

- **Model** of network uncertainty
  - Link speed distribution
  - Delay distribution
  - Topology distribution
- **Model** of workload
  - Web browsing
  - MapReduce
  - videoconferencing
  - streaming video (YouTube/Netflix)

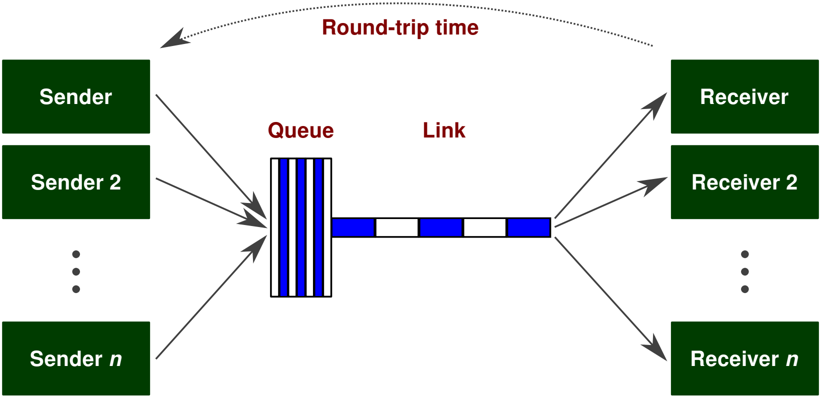
Dumbbell network



Dumbbell network



Dumbbell network



Superrational congestion control

At this moment,* do I:

- send a packet
- not send a packet?

Superrational congestion control

At this moment,* do I:

- send a packet
- not send a packet?

* Assuming every node is running the same algorithm.

Remy: tractable search for best policy

- Best decision given all history: not tractable
- Instead, **summarize the history**

A RemyCC tracks four congestion signals

r_ewma$_\alpha$: **short-term** moving average of interval between acks
"How fast are packets arriving (now)?"

r_ewma$_\beta$: **long-term** moving average of same
"How fast are packets arriving (smoothed)?"

s_ewma: moving average of interval between acked timestamps
"How fast was I sending?"

rtt_ratio: ratio of last RTT to smallest RTT so far
"How long is the queue?"

Why these four features?

- We can measure the benefit of each!
- Removing any one hurts
  - losing r_ewma_α hurts the most
- More signals increase search time
- ...but others might help on some networks

A RemyCC maps each state to an action

$$\textsc{RemyCC}(\mathit{r\_ewma}_{\alpha\beta}, \mathit{s\_ewma}, \mathit{rtt\_ratio}) \rightarrow \langle m, b, \tau \rangle$$

m	Multiple to congestion window
b	Increment to congestion window
τ	Minimum interval between two outgoing packets

Runtime for a RemyCC

On ack:

- $\langle m, b, \tau \rangle \leftarrow \textsc{RemyCC}(r\_ewma_{\alpha\beta}, s\_ewma, rtt\_ratio)$
- $\texttt{cwnd} \leftarrow m \cdot \texttt{cwnd} + b$

Send packet if:

- $\texttt{cwnd} > \texttt{FlightSize}$, and
- last packet sent $> \tau$ ago

Remy's job

Find piecewise-continuous $\textsc{RemyCC}()$ that optimizes expected value of objective function

Remy example: 2D state space

On ack:

$$\langle m, b, \tau \rangle \leftarrow \textsc{RemyCC}(\mathit{s\_ewma}, \mathit{r\_ewma}_{\alpha}, \quad \mathit{r\_ewma}_{\beta}, \mathit{rtt\_ratio} \quad)$$

Remy example: 2D state space

On ack:

$\langle m, b, \tau \rangle \leftarrow \textsc{RemyCC}(s\_ewma, r\_ewma_{\alpha},$ [redacted]$)$

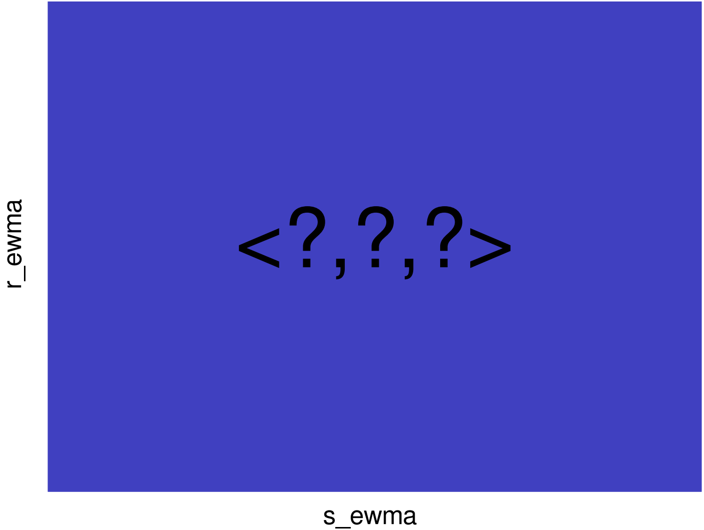
Remy example: **model**

Quantity	Distribution	Units
Link speed	Uniform(10, 20)	Mbps
RTT	Uniform(100, 200)	ms
n	Uniform(1, 16)	
“On” process “Off” process	$\exp[\mu = 5]$ same	seconds

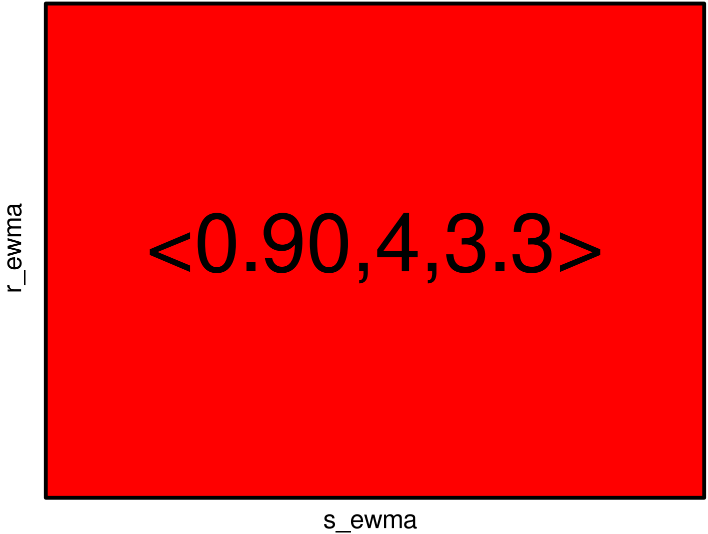
Remy example: **mission**

$$\sum_i \log \left[\frac{\text{throughput}_i}{\text{delay}_i} \right]$$

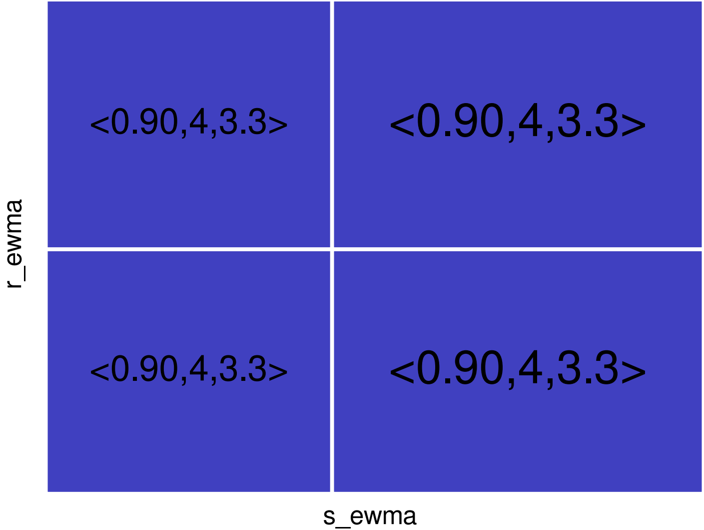
One action for all states. Find the best value.



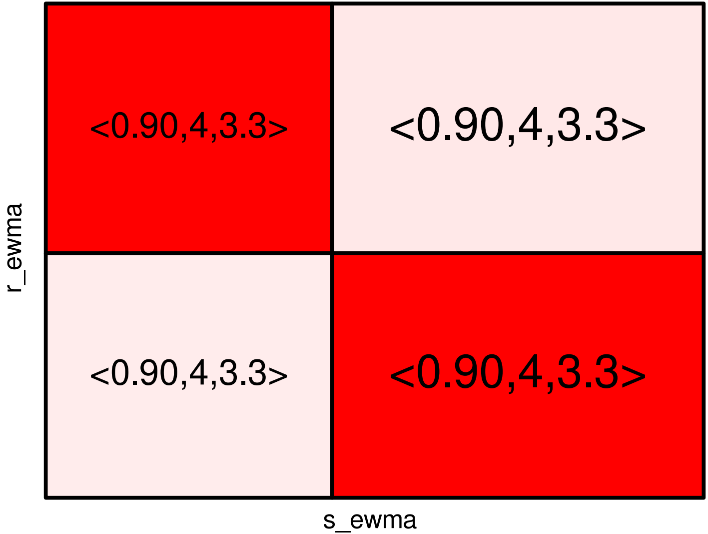
The best (single) action. Now split it on median.



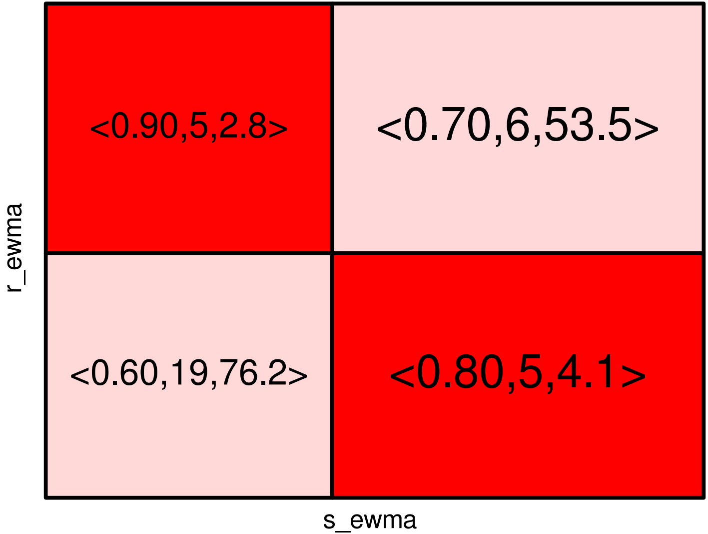
Simulate



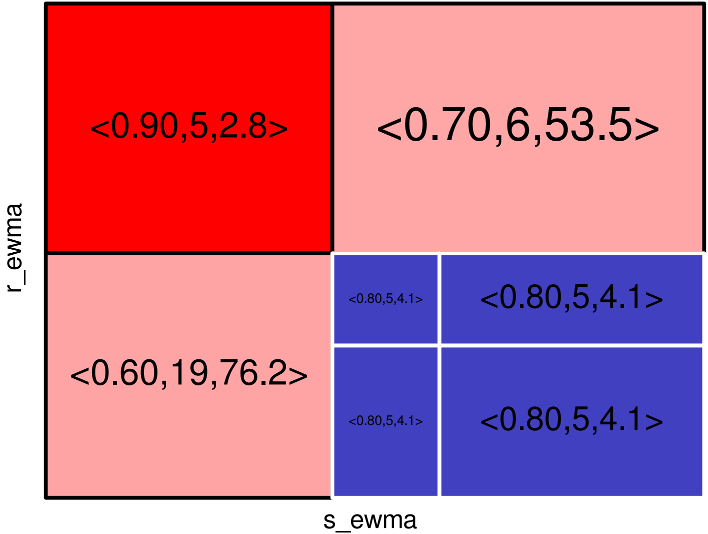
Optimize each of the new actions



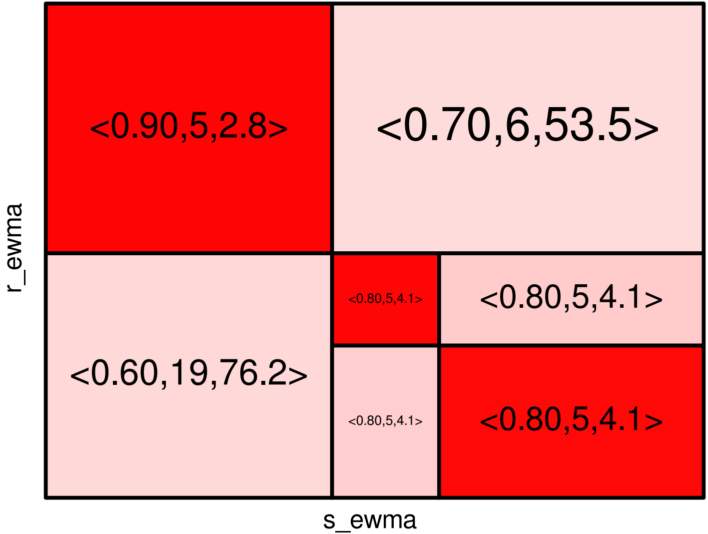
Now split the most-used rule



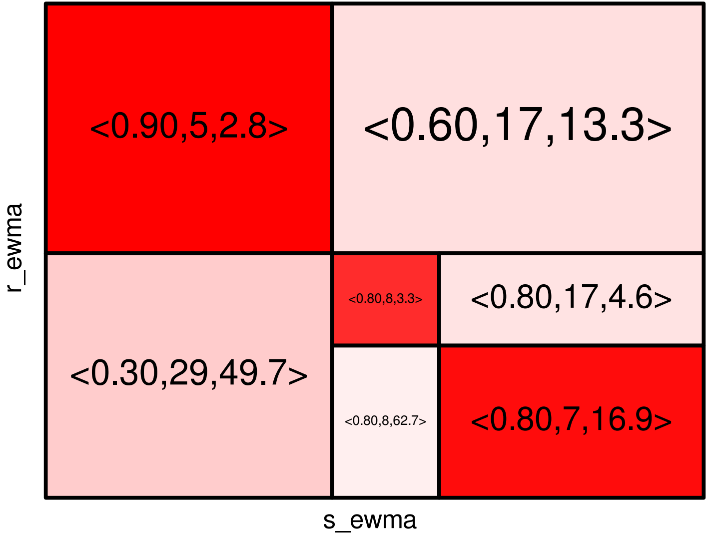
Simulate


<0.90,5,2.8>
<0.70,6,53.5>
<0.60,19,76.2>
<0.80,5,4.1>
<0.80,5,4.1>
<0.80,5,4.1>
<0.80,5,4.1>
r_ewma
s_ewma

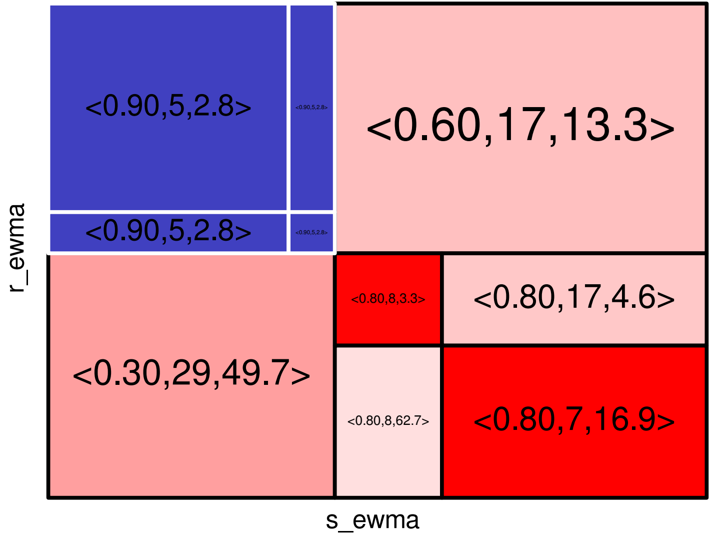
Optimize


<0.90,5,2.8>
<0.70,6,53.5>
<0.80,5,4.1>
<0.80,5,4.1>
<0.60,19,76.2>
<0.80,5,4.1>
<0.80,5,4.1>
r_ewma
s_ewma

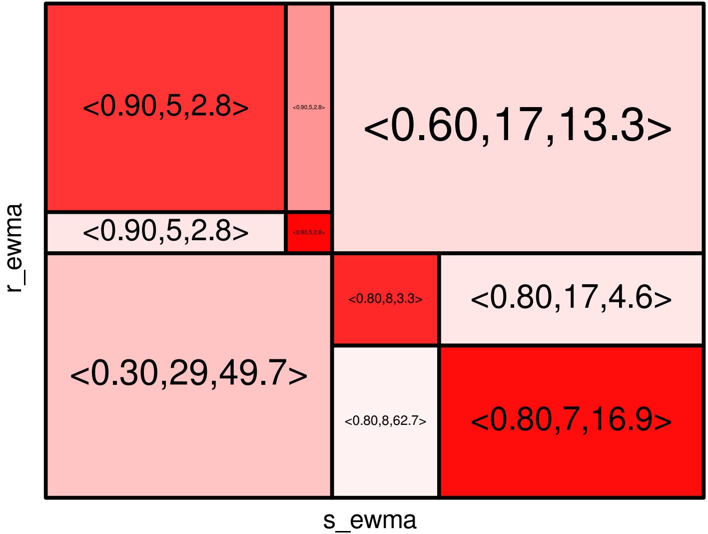
Split


<0.90,5,2.8>
<0.60,17,13.3>
<0.30,29,49.7>
<0.80,8,3.3>
<0.80,17,4.6>
<0.80,8,62.7>
<0.80,7,16.9>
r_ewma
s_ewma

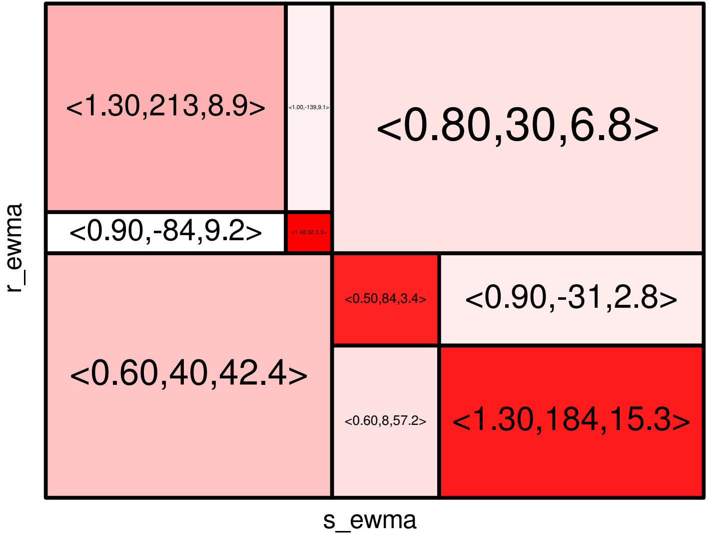
Simulate


<0.90,5,2.8>
<0.90,5,2.8>
<0.60,17,13.3>
<0.90,5,2.8>
<0.90,5,2.8>
<0.80,8,3.3>
<0.80,17,4.6>
<0.30,29,49.7>
<0.80,8,62.7>
<0.80,7,16.9>
r_ewma
s_ewma

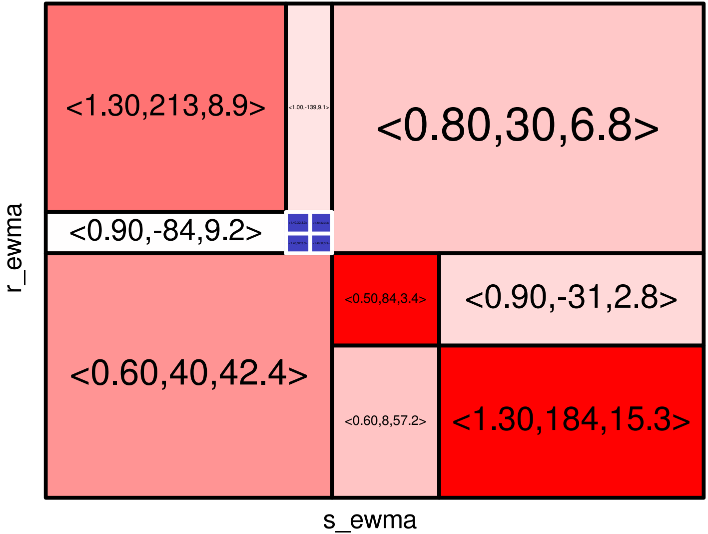
Optimize


<0.90,5,2.8>
<0.90,5,2.8>
<0.60,17,13.3>
<0.90,5,2.8>
<0.90,5,2.8>
<0.80,8,3.3>
<0.80,17,4.6>
<0.30,29,49.7>
<0.80,8,62.7>
<0.80,7,16.9>
r_ewma
s_ewma

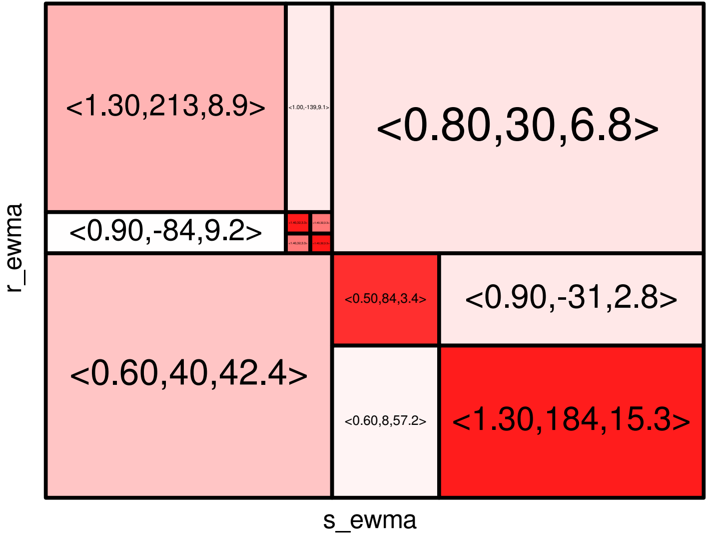
Split


<1.30,213,8.9>
<1.00,-139,9.1>
<0.80,30,6.8>
<0.90,-84,9.2>
<1.40,32,3.3>
<0.50,84,3.4>
<0.90,-31,2.8>
<0.60,40,42.4>
<0.60,8,57.2>
<1.30,184,15.3>
r_ewma
s_ewma

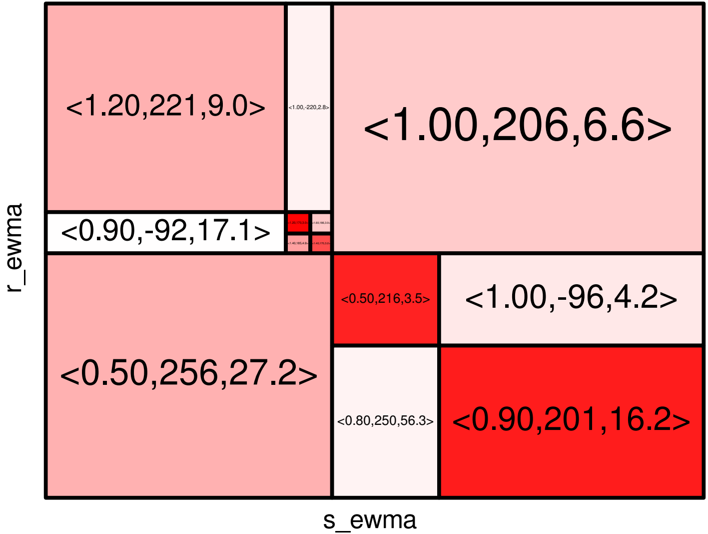
Simulate


<1.30,213,8.9>
<1.00,-139,9.1>
<0.80,30,6.8>
<0.90,-84,9.2>
<0.50,84,3.4>
<0.90,-31,2.8>
<0.60,40,42.4>
<0.60,8,57.2>
<1.30,184,15.3>
r_ewma
s_ewma

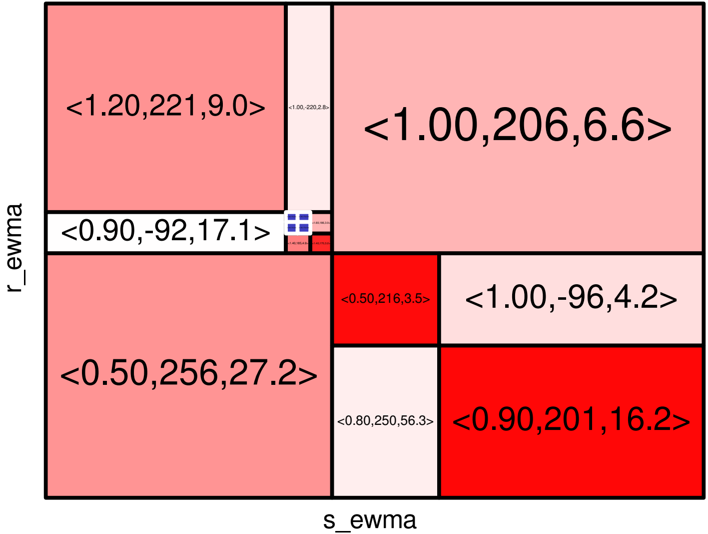
Optimize


<1.30,213,8.9>
<1.00,-139,9.1>
<0.80,30,6.8>
<0.90,-84,9.2>
<0.50,84,3.4>
<0.90,-31,2.8>
<0.60,40,42.4>
<0.60,8,57.2>
<1.30,184,15.3>
r_ewma
s_ewma

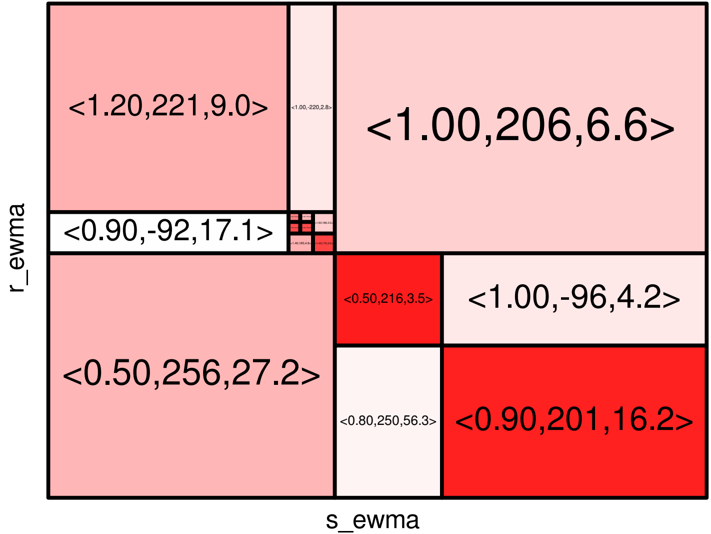
Split


<1.20,221,9.0>
<1.00,-220,2.8>
<1.00,206,6.6>
<0.90,-92,17.1>
<1.20,170,3.0>
<1.40,185,4.9>
<1.40,170,3.2>
<0.50,216,3.5>
<1.00,-96,4.2>
<0.50,256,27.2>
<0.80,250,56.3>
<0.90,201,16.2>
r_ewma
s_ewma

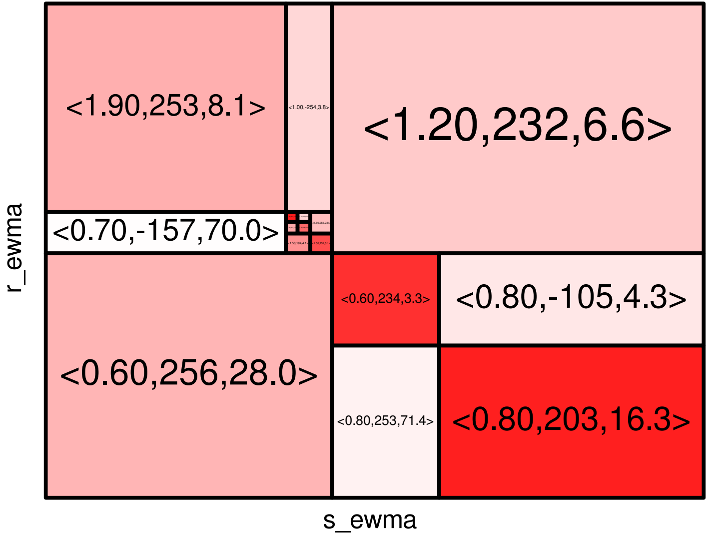
Simulate


<1.20,221,9.0>
<1.00,-220,2.8>
<1.00,206,6.6>
<0.90,-92,17.1>
<0.50,216,3.5>
<1.00,-96,4.2>
<0.50,256,27.2>
<0.80,250,56.3>
<0.90,201,16.2>
r_ewma
s_ewma

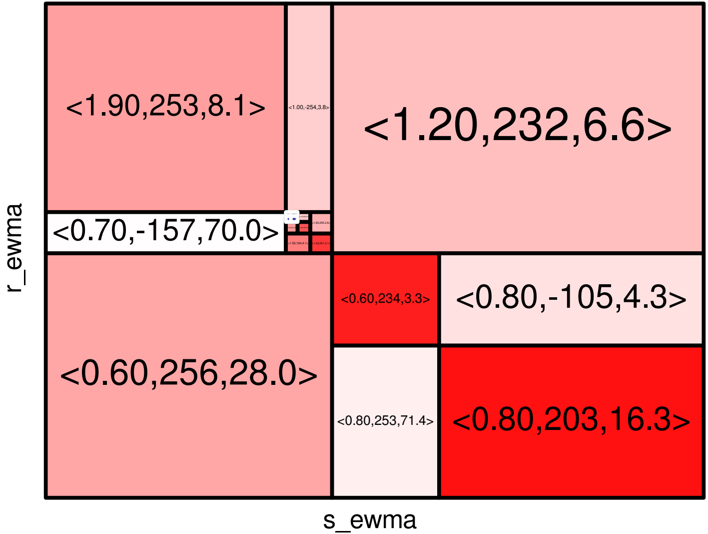
Optimize


<1.20,221,9.0>
<1.00,-220,2.8>
<1.00,206,6.6>
<0.90,-92,17.1>
<0.50,216,3.5>
<1.00,-96,4.2>
<0.50,256,27.2>
<0.80,250,56.3>
<0.90,201,16.2>
r_ewma
s_ewma

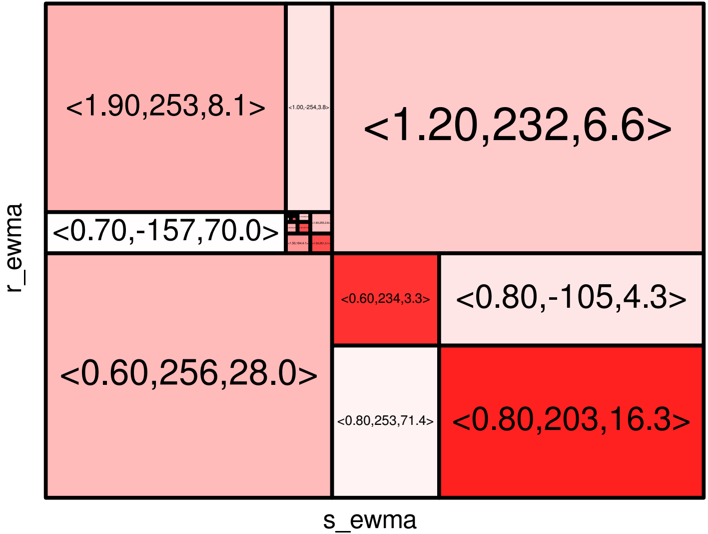
Split


<1.90,253,8.1>
<1.00,-254,3.8>
<1.20,232,6.6>
<0.70,-157,70.0>
<1.80,255,2.8>
<1.50,194,4.1>
<1.50,251,3.1>
<0.60,234,3.3>
<0.80,-105,4.3>
<0.60,256,28.0>
<0.80,253,71.4>
<0.80,203,16.3>
r_ewma
s_ewma

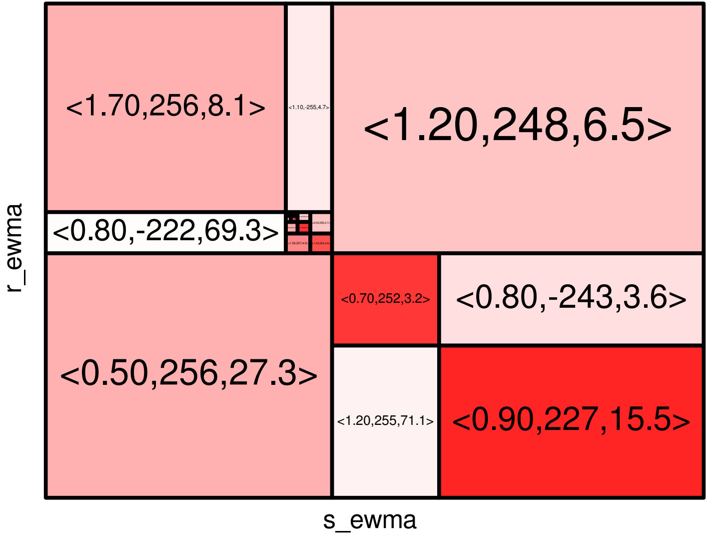
Simulate


r_ewma
<1.90,253,8.1>
<1.00,-254,3.8>
<1.20,232,6.6>
<0.70,-157,70.0>
<0.60,234,3.3>
<0.80,-105,4.3>
<0.60,256,28.0>
<0.80,253,71.4>
<0.80,203,16.3>
s_ewma

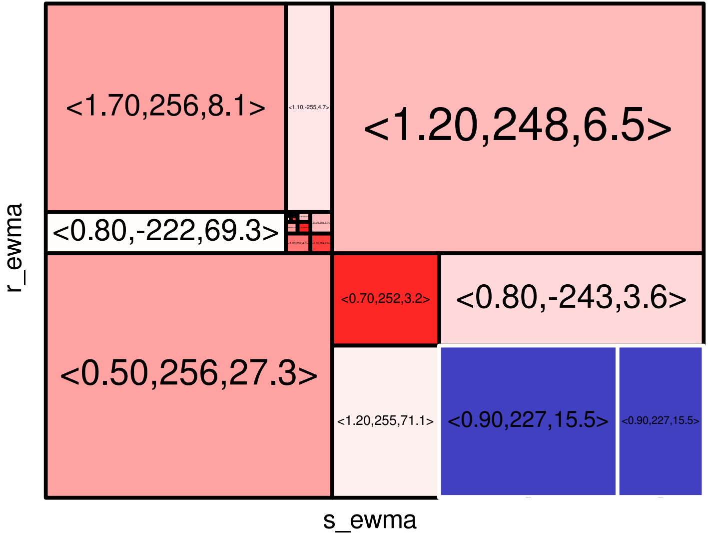
Optimize


<1.90,253,8.1>
<1.00,-254,3.8>
<1.20,232,6.6>
<0.70,-157,70.0>
<0.60,234,3.3>
<0.80,-105,4.3>
<0.60,256,28.0>
<0.80,253,71.4>
<0.80,203,16.3>
r_ewma
s_ewma

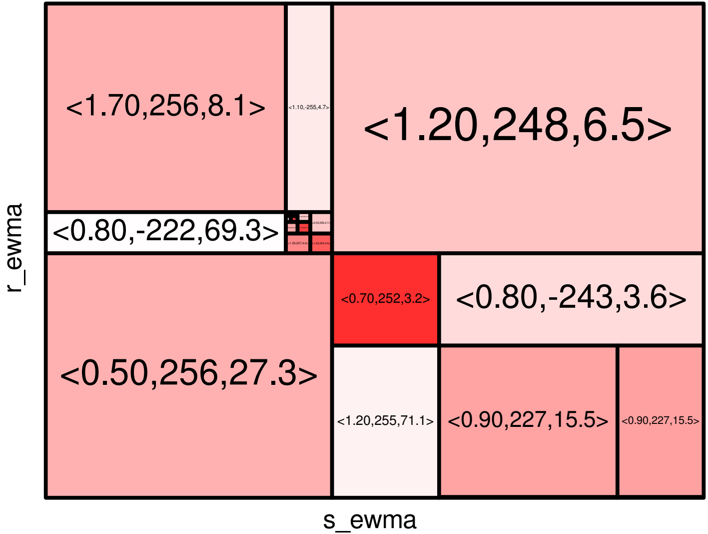
Split


<1.70,256,8.1>
<1.10,-255,4.7>
<1.20,248,6.5>
<0.80,-222,69.3>
<0.70,252,3.2>
<0.80,-243,3.6>
<0.50,256,27.3>
<1.20,255,71.1>
<0.90,227,15.5>
r_ewma
s_ewma

Simulate

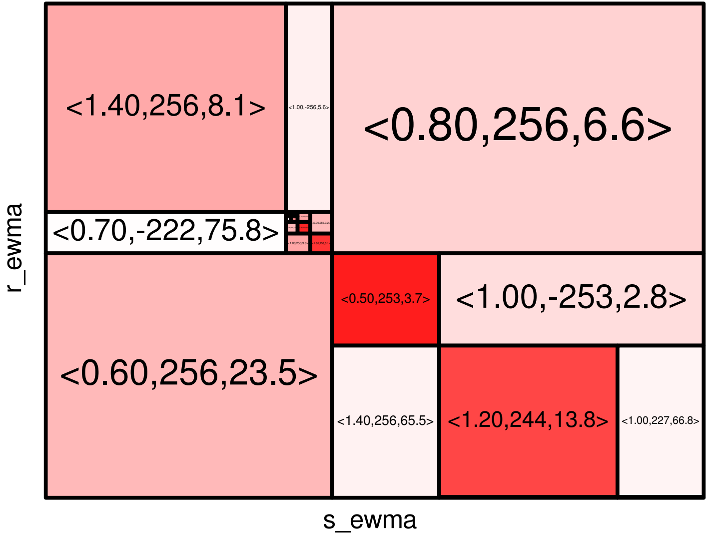

<1.70,256,8.1>
<1.10,-255,4.7>
<1.20,248,6.5>
<0.80,-222,69.3>
<0.70,252,3.2>
<0.80,-243,3.6>
<0.50,256,27.3>
<1.20,255,71.1>
<0.90,227,15.5>
<0.90,227,15.5>
r_ewma
s_ewma

Optimize

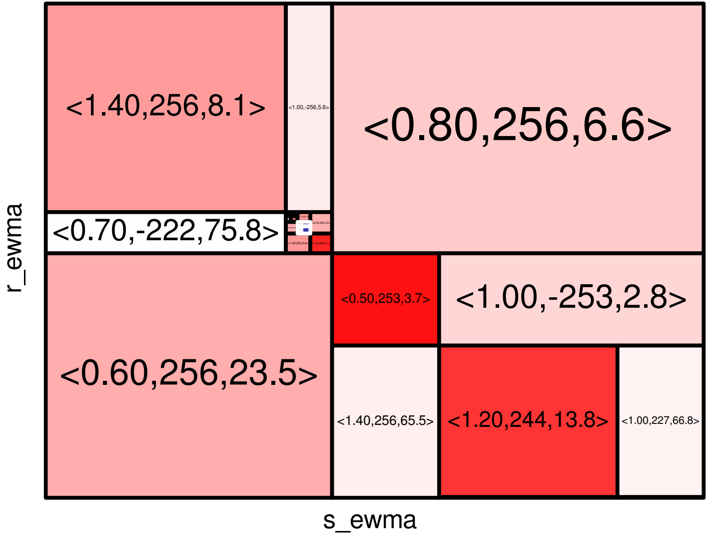


<1.70,256,8.1>
<1.10,-255,4.7>
<1.20,248,6.5>
<0.80,-222,69.3>
<0.70,252,3.2>
<0.80,-243,3.6>
<0.50,256,27.3>
<1.20,255,71.1>
<0.90,227,15.5>
<0.90,227,15.5>
r_ewma
s_ewma

Split

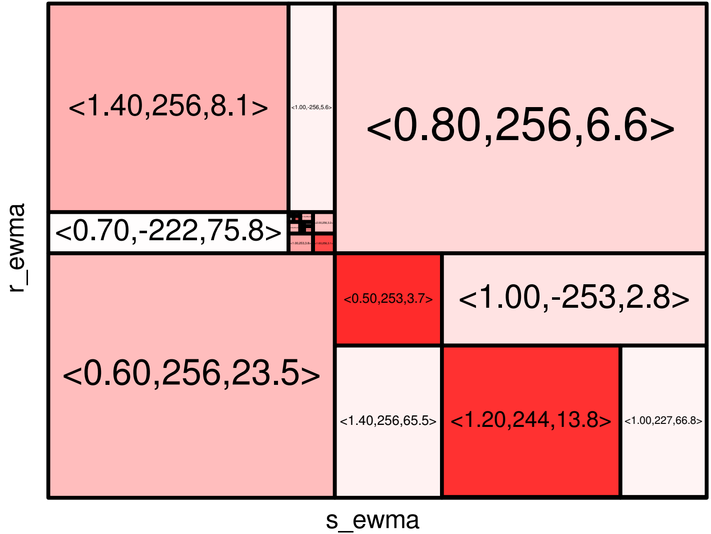

<1.40,256,8.1>
<1.00,-256,5.6>
<0.80,256,6.6>
<0.70,-222,75.8>
<0.50,253,3.7>
<1.00,-253,2.8>
<0.60,256,23.5>
<1.40,256,65.5>
<1.20,244,13.8>
<1.00,227,66.8>
r_ewma
s_ewma

Simulate

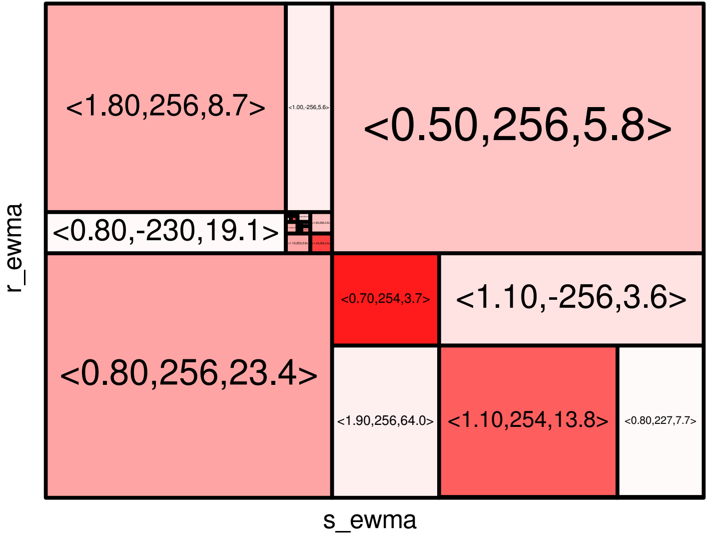


<1.40,256,8.1>
<1.00,-256,5.6>
<0.80,256,6.6>
<0.70,-222,75.8>
<0.50,253,3.7>
<1.00,-253,2.8>
<0.60,256,23.5>
<1.40,256,65.5>
<1.20,244,13.8>
<1.00,227,66.8>
r_ewma
s_ewma

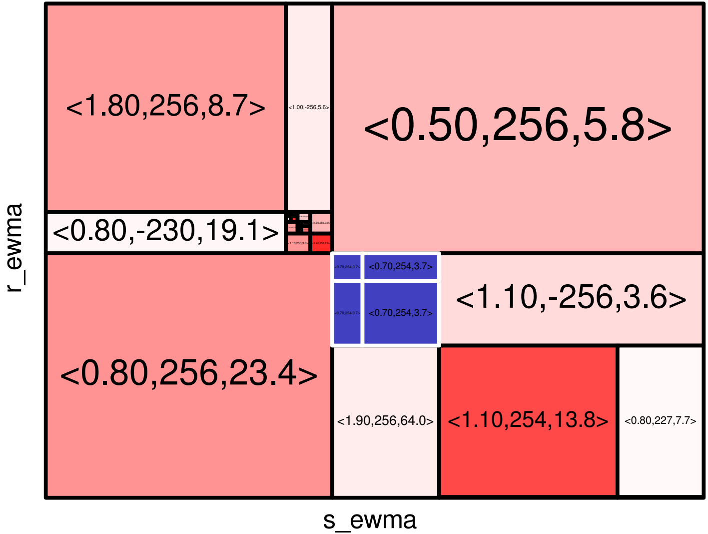
Optimize


<1.40,256,8.1>
<1.00,-256,5.6>
<0.80,256,6.6>
<0.70,-222,75.8>
<0.50,253,3.7>
<1.00,-253,2.8>
<0.60,256,23.5>
<1.40,256,65.5>
<1.20,244,13.8>
<1.00,227,66.8>
r_ewma
s_ewma

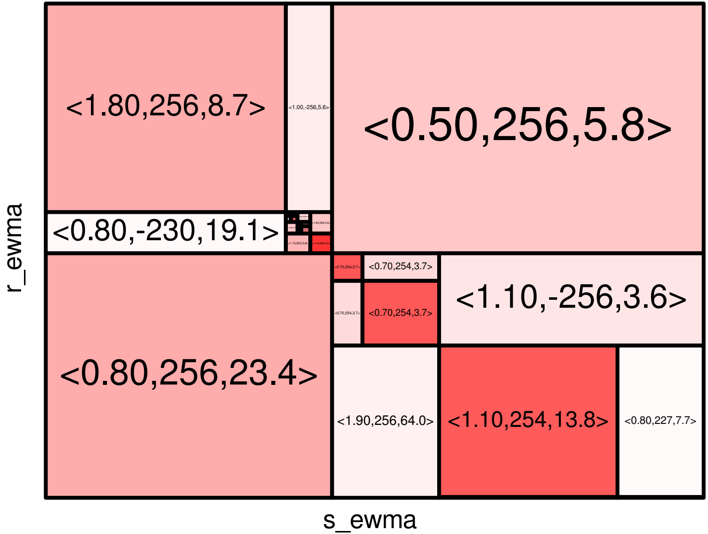
Split


<1.80,256,8.7>
<1.00,-256,5.6>
<0.50,256,5.8>
<0.80,-230,19.1>
<0.70,254,3.7>
<1.10,-256,3.6>
<0.80,256,23.4>
<1.90,256,64.0>
<1.10,254,13.8>
<0.80,227,7.7>
r_ewma
s_ewma

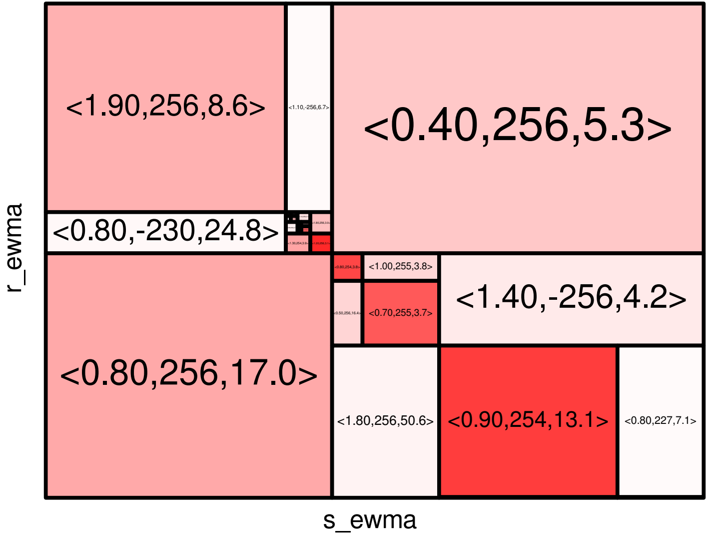
Simulate


<1.80,256,8.7>
<1.00,-256,5.6>
<0.50,256,5.8>
<0.80,-230,19.1>
<0.70,254,3.7>
<0.70,254,3.7>
<1.10,-256,3.6>
<0.80,256,23.4>
<1.90,256,64.0>
<1.10,254,13.8>
<0.80,227,7.7>
r_ewma
s_ewma

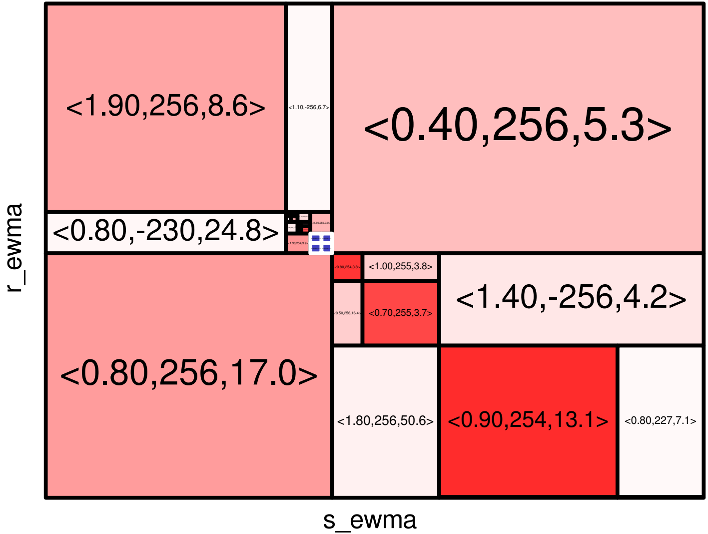
Optimize


<1.80,256,8.7>
<1.00,-256,5.6>
<0.50,256,5.8>
<0.80,-230,19.1>
<0.70,254,3.7>
<0.70,254,3.7>
<1.10,-256,3.6>
<0.80,256,23.4>
<1.90,256,64.0>
<1.10,254,13.8>
<0.80,227,7.7>
r_ewma
s_ewma

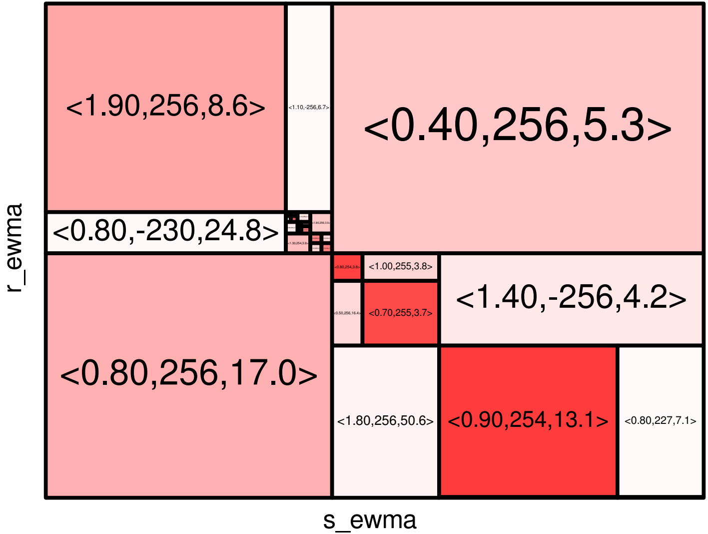
Split


<1.90,256,8.6>
<1.10,-256,6.7>
<0.40,256,5.3>
<0.80,-230,24.8>
<1.00,255,3.8>
<1.40,-256,4.2>
<0.70,255,3.7>
<0.80,256,17.0>
<1.80,256,50.6>
<0.90,254,13.1>
<0.80,227,7.1>
r_ewma
s_ewma

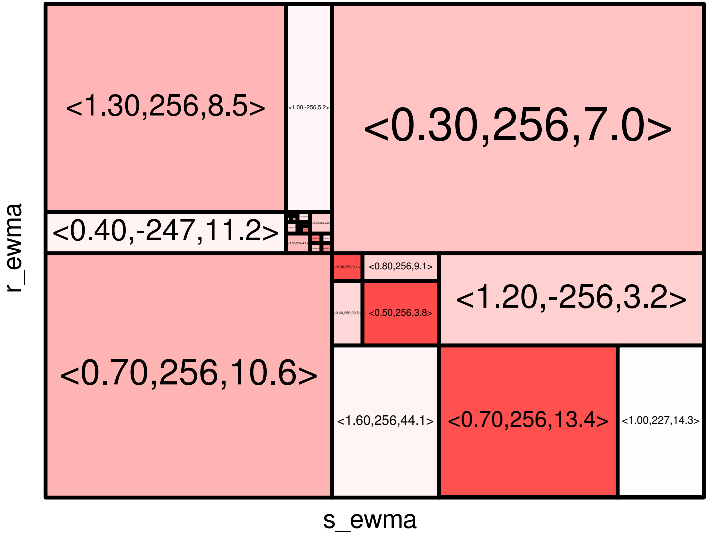
Simulate


r_ewma
s_ewma
<1.90,256,8.6>
<1.10,-256,6.7>
<0.40,256,5.3>
<0.80,-230,24.8>
<1.00,255,3.8>
<1.40,-256,4.2>
<0.70,255,3.7>
<0.80,256,17.0>
<1.80,256,50.6>
<0.90,254,13.1>
<0.80,227,7.1>

Optimize

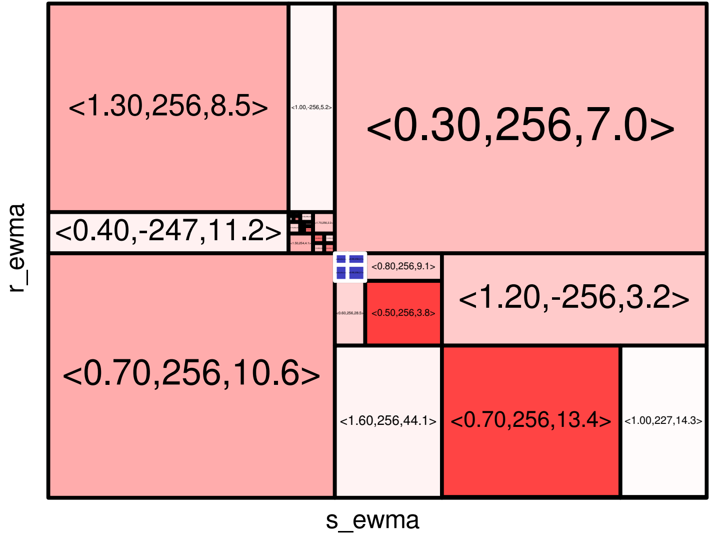

<1.90,256,8.6>
<1.10,-256,6.7>
<0.40,256,5.3>
<0.80,-230,24.8>
<1.00,255,3.8>
<1.40,-256,4.2>
<0.70,255,3.7>
<0.80,256,17.0>
<1.80,256,50.6>
<0.90,254,13.1>
<0.80,227,7.1>
r_ewma
s_ewma

Split



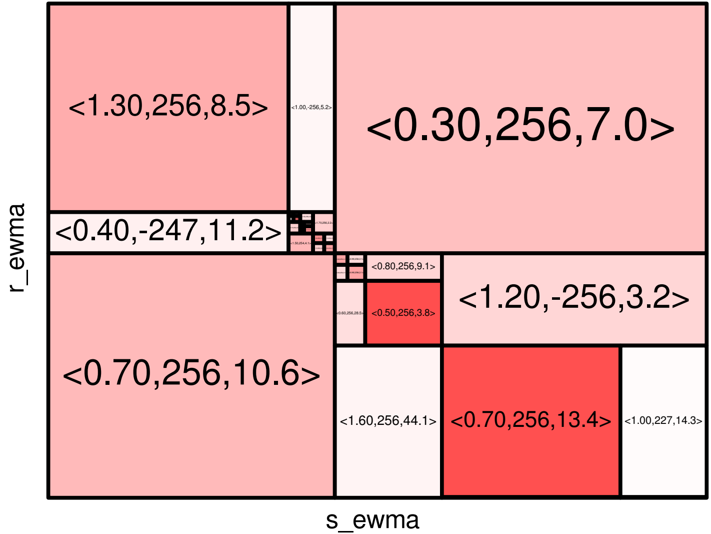
<1.30,256,8.5>
<1.00,-256,5.2>
<0.30,256,7.0>
<0.40,-247,11.2>
<1.70,256,2.2>
<1.50,254,4.1>
<0.90,256,3.1>
<0.80,256,9.1>
<0.60,256,28.5>
<0.50,256,3.8>
<1.20,-256,3.2>
<0.70,256,10.6>
<1.60,256,44.1>
<0.70,256,13.4>
<1.00,227,14.3>
r_ewma
s_ewma

Simulate

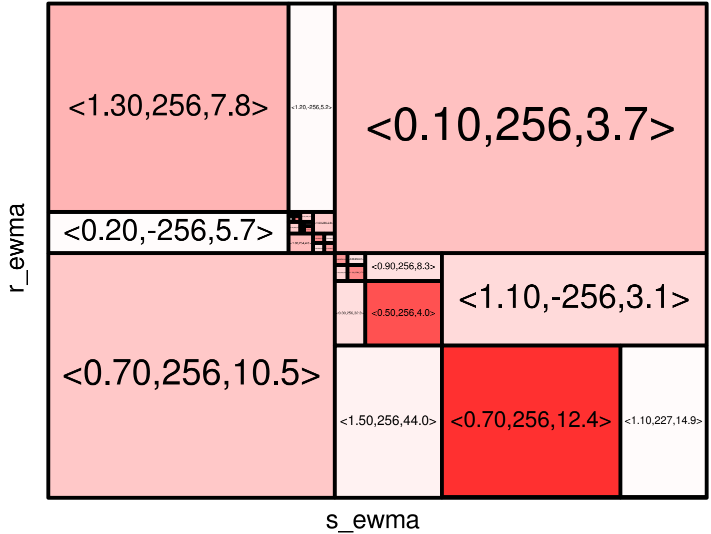


<1.30,256,8.5>
<1.00,-256,5.2>
<0.30,256,7.0>
<0.40,-247,11.2>
<0.80,256,9.1>
<1.20,-256,3.2>
<0.50,256,3.8>
<0.70,256,10.6>
<1.60,256,44.1>
<0.70,256,13.4>
<1.00,227,14.3>
r_ewma
s_ewma

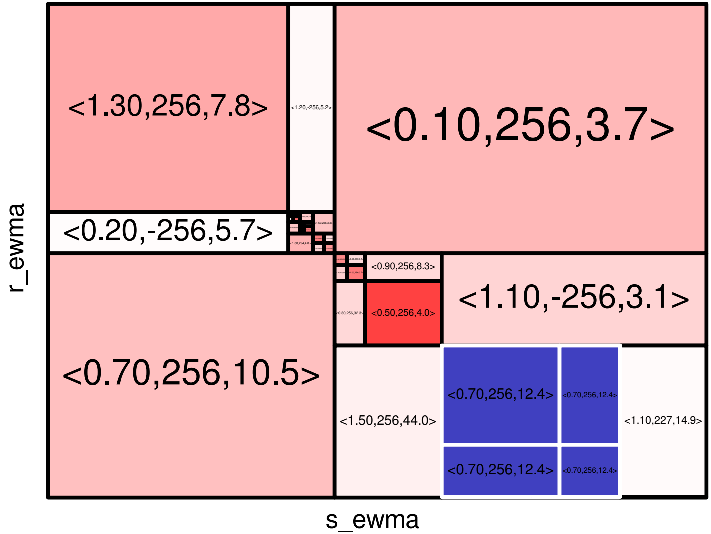
Optimize


<1.30,256,8.5>
<1.00,-256,5.2>
<0.30,256,7.0>
<0.40,-247,11.2>
<1.50,254,4.1>
<0.80,256,9.1>
<0.60,256,28.5>
<0.50,256,3.8>
<1.20,-256,3.2>
<0.70,256,10.6>
<1.60,256,44.1>
<0.70,256,13.4>
<1.00,227,14.3>
r_ewma
s_ewma

Split

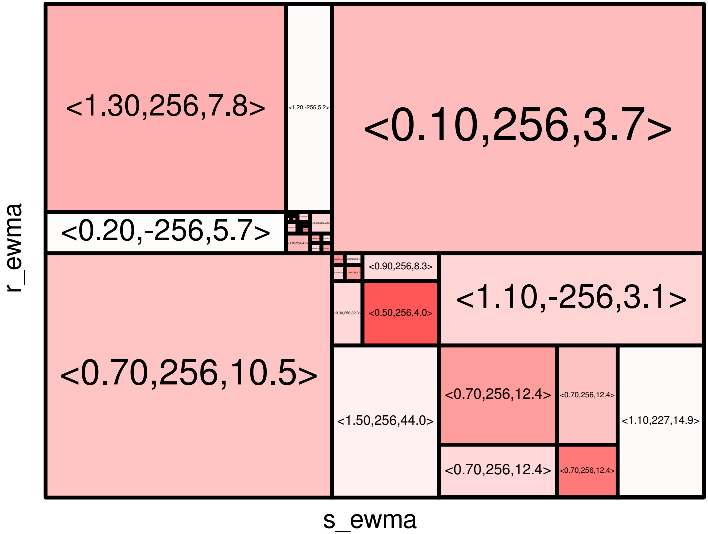

<1.30,256,7.8>
<1.20,-256,5.2>
<0.10,256,3.7>
<0.20,-256,5.7>
<0.90,256,8.3>
<1.10,-256,3.1>
<0.30,256,32.3>
<0.50,256,4.0>
<0.70,256,10.5>
<1.50,256,44.0>
<0.70,256,12.4>
<1.10,227,14.9>
r_ewma
s_ewma

Simulate



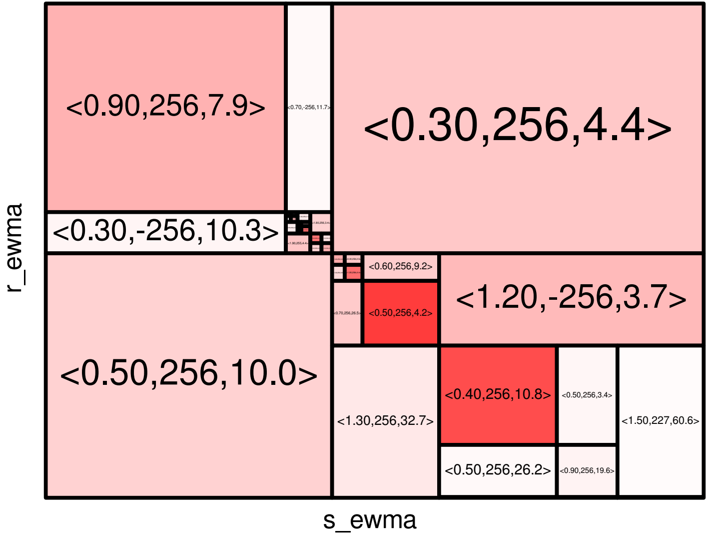
<1.30,256,7.8>
<1.20,-256,5.2>
<0.10,256,3.7>
<0.20,-256,5.7>
<0.90,256,8.3>
<0.30,256,32.3>
<0.50,256,4.0>
<1.10,-256,3.1>
<0.70,256,10.5>
<1.50,256,44.0>
<0.70,256,12.4>
<0.70,256,12.4>
<0.70,256,12.4>
<0.70,256,12.4>
<1.10,227,14.9>
r_ewma
s_ewma

Optimize

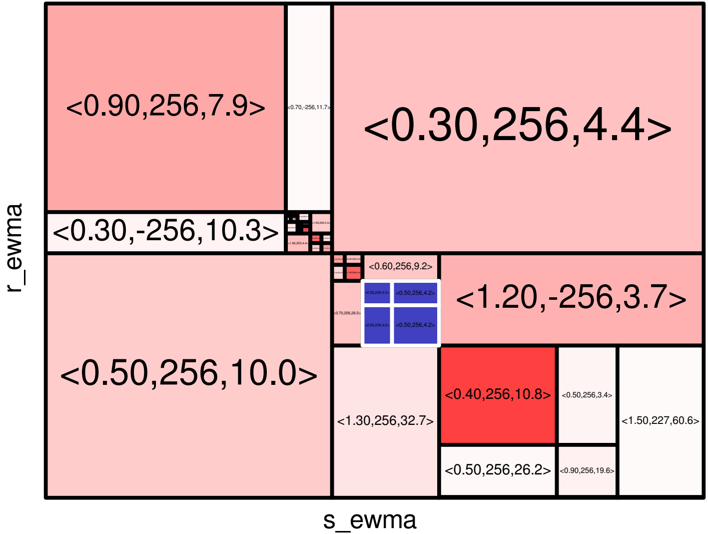


<1.30,256,7.8>
<1.20,-256,5.2>
<0.10,256,3.7>
<0.20,-256,5.7>
<0.90,256,8.3>
<0.50,256,4.0>
<1.10,-256,3.1>
<0.70,256,10.5>
<1.50,256,44.0>
<0.70,256,12.4>
<0.70,256,12.4>
<1.10,227,14.9>
<0.70,256,12.4>
<0.70,256,12.4>
r_ewma
s_ewma

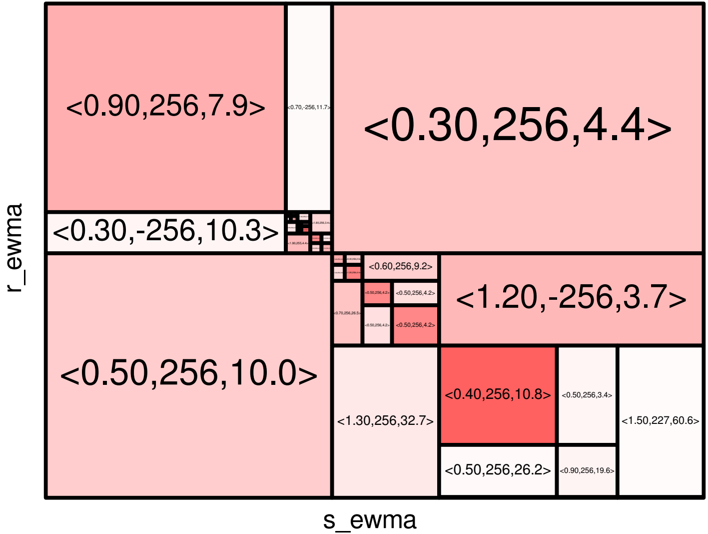
Split


r_ewma
s_ewma
<0.90,256,7.9>
<0.70,-256,11.7>
<0.30,256,4.4>
<0.30,-256,10.3>
<0.60,256,9.2>
<1.20,-256,3.7>
<0.50,256,10.0>
<0.70,256,26.5>
<0.50,256,4.2>
<1.30,256,32.7>
<0.40,256,10.8>
<0.50,256,3.4>
<1.50,227,60.6>
<0.50,256,26.2>
<0.90,256,19.6>

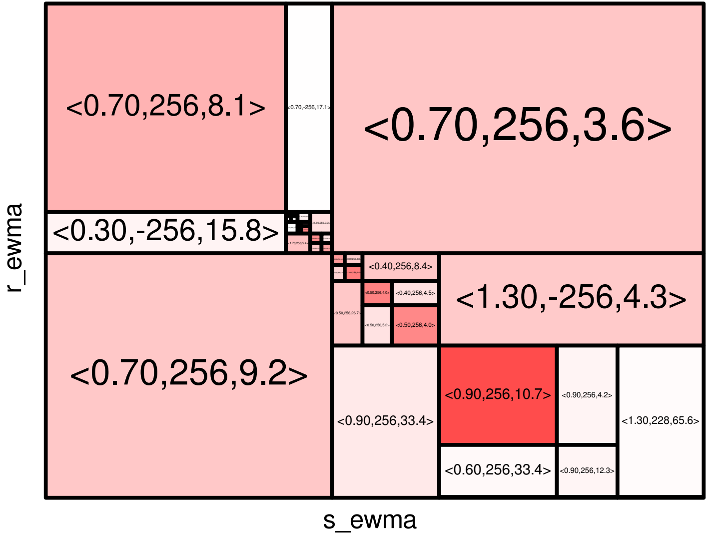
Simulate


<0.90,256,7.9>
<0.70,-256,11.7>
<0.30,256,4.4>
<0.30,-256,10.3>
<0.60,256,9.2>
<0.50,256,4.2>
<0.50,256,4.2>
<0.70,256,26.5>
<1.20,-256,3.7>
<0.50,256,10.0>
<1.30,256,32.7>
<0.40,256,10.8>
<0.50,256,3.4>
<1.50,227,60.6>
<0.50,256,26.2>
<0.90,256,19.6>
r_ewma
s_ewma

Optimize

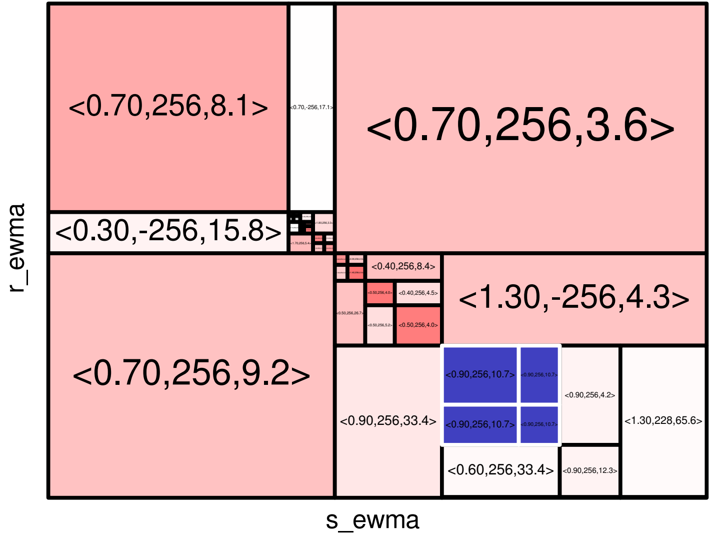

<0.90,256,7.9>
<0.70,-256,11.7>
<0.30,256,4.4>
<0.30,-256,10.3>
<0.60,256,9.2>
<1.20,-256,3.7>
<0.50,256,4.2>
<0.50,256,4.2>
<0.50,256,4.2>
<0.50,256,4.2>
<0.50,256,10.0>
<1.30,256,32.7>
<0.40,256,10.8>
<0.50,256,3.4>
<1.50,227,60.6>
<0.50,256,26.2>
<0.90,256,19.6>
r_ewma
s_ewma

Split



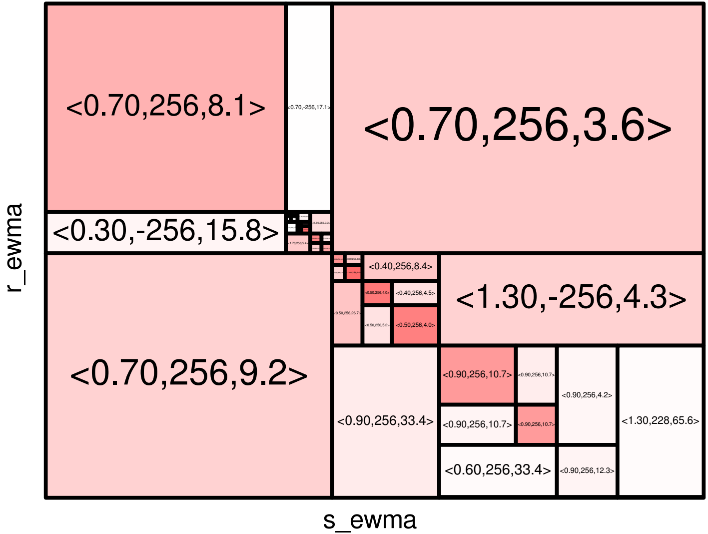
<0.70,256,8.1>
<0.70,-256,17.1>
<0.70,256,3.6>
<0.30,-256,15.8>
<0.40,256,8.4>
<0.50,256,4.0>
<0.40,256,4.5>
<0.50,256,26.7>
<0.50,256,5.2>
<0.50,256,4.0>
<1.30,-256,4.3>
<0.70,256,9.2>
<0.90,256,33.4>
<0.90,256,10.7>
<0.90,256,4.2>
<1.30,228,65.6>
<0.60,256,33.4>
<0.90,256,12.3>
r_ewma
s_ewma

Simulate

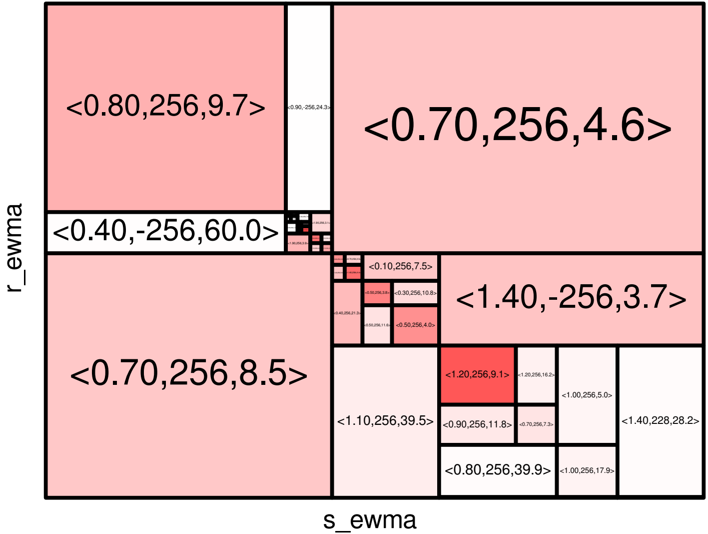


<0.70,256,8.1>
<0.70,-256,17.1>
<0.70,256,3.6>
<0.30,-256,15.8>
<0.40,256,8.4>
<0.50,256,4.0>
<0.40,256,4.5>
<1.30,-256,4.3>
<0.50,256,26.7>
<0.50,256,5.2>
<0.50,256,4.0>
<0.70,256,9.2>
<0.90,256,10.7>
<0.90,256,10.7>
<0.90,256,4.2>
<0.90,256,33.4>
<1.30,228,65.6>
<0.90,256,10.7>
<0.90,256,10.7>
<0.60,256,33.4>
<0.90,256,12.3>
r_ewma
s_ewma

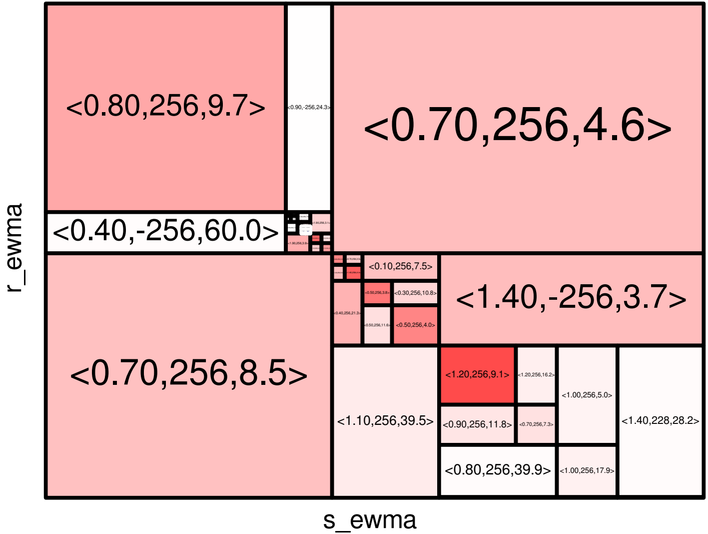
Optimize


<0.70,256,8.1>
<0.70,-256,17.1>
<0.70,256,3.6>
<0.30,-256,15.8>
<1.70,256,5.4>
<0.40,256,8.4>
<0.50,256,4.0>
<0.40,256,4.5>
<0.50,256,26.7>
<0.50,256,5.2>
<0.50,256,4.0>
<1.30,-256,4.3>
<0.70,256,9.2>
<0.90,256,10.7>
<0.90,256,10.7>
<0.90,256,4.2>
<0.90,256,33.4>
<0.90,256,10.7>
<0.90,256,10.7>
<1.30,228,65.6>
<0.60,256,33.4>
<0.90,256,12.3>
r_ewma
s_ewma

Split

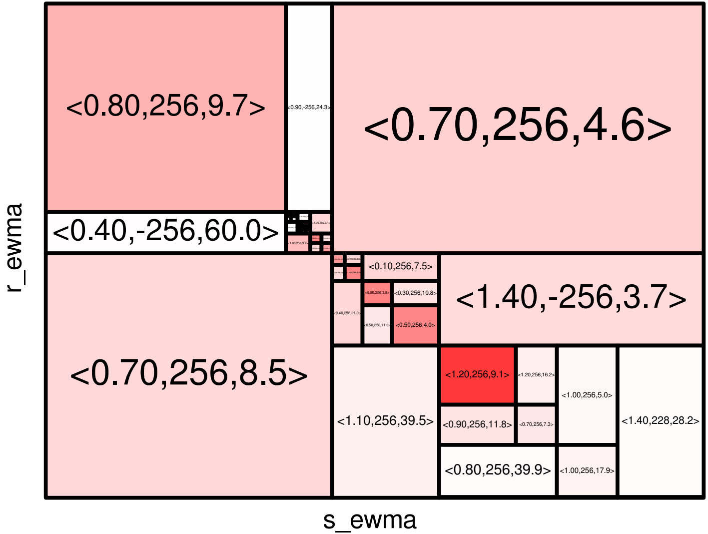

<0.80,256,9.7>
<0.90,-256,24.3>
<0.70,256,4.6>
<0.40,-256,60.0>
<0.10,256,7.5>
<0.50,256,3.8>
<0.30,256,10.8>
<0.40,256,21.3>
<0.50,256,11.6>
<0.50,256,4.0>
<1.40,-256,3.7>
<0.70,256,8.5>
<1.10,256,39.5>
<1.20,256,9.1>
<1.20,256,16.2>
<1.00,256,5.0>
<1.40,228,28.2>
<0.90,256,11.8>
<0.70,256,7.3>
<0.80,256,39.9>
<1.00,256,17.9>
r_ewma
s_ewma

Simulate



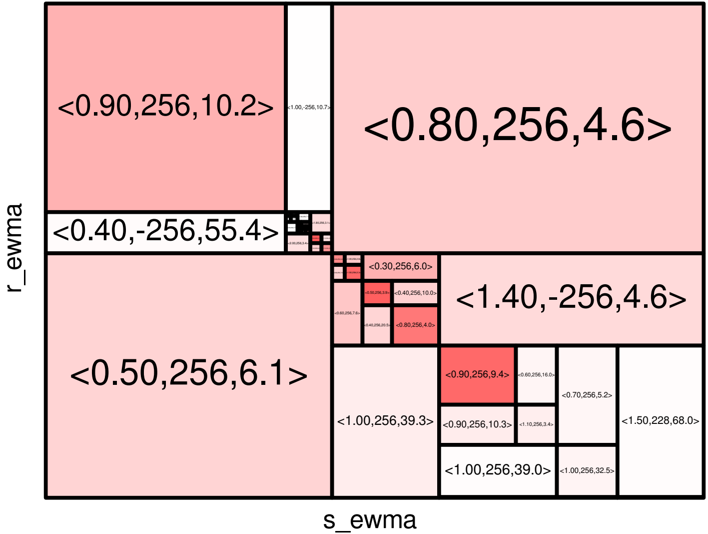
<0.80,256,9.7>
<0.90,-256,24.3>
<0.70,256,4.6>
<0.40,-256,60.0>
<0.10,256,7.5>
<0.30,256,10.8>
<0.50,256,4.0>
<1.40,-256,3.7>
<0.70,256,8.5>
<1.10,256,39.5>
<1.20,256,9.1>
<1.20,256,16.2>
<1.00,256,5.0>
<1.40,228,28.2>
<0.90,256,11.8>
<0.70,256,7.3>
<0.80,256,39.9>
<1.00,256,17.9>
r_ewma
s_ewma

Optimize

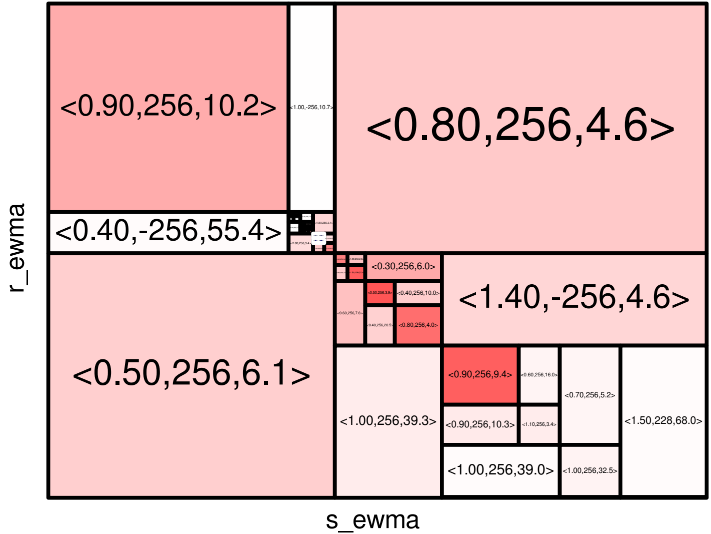


<0.80,256,9.7>
<0.90,-256,24.3>
<0.70,256,4.6>
<0.40,-256,60.0>
<0.10,256,7.5>
<1.40,-256,3.7>
<0.50,256,3.8>
<0.30,256,10.8>
<0.40,256,21.3>
<0.50,256,11.6>
<0.50,256,4.0>
<0.70,256,8.5>
<1.20,256,9.1>
<1.20,256,16.2>
<1.00,256,5.0>
<1.10,256,39.5>
<0.90,256,11.8>
<0.70,256,7.3>
<1.40,228,28.2>
<0.80,256,39.9>
<1.00,256,17.9>
r_ewma
s_ewma

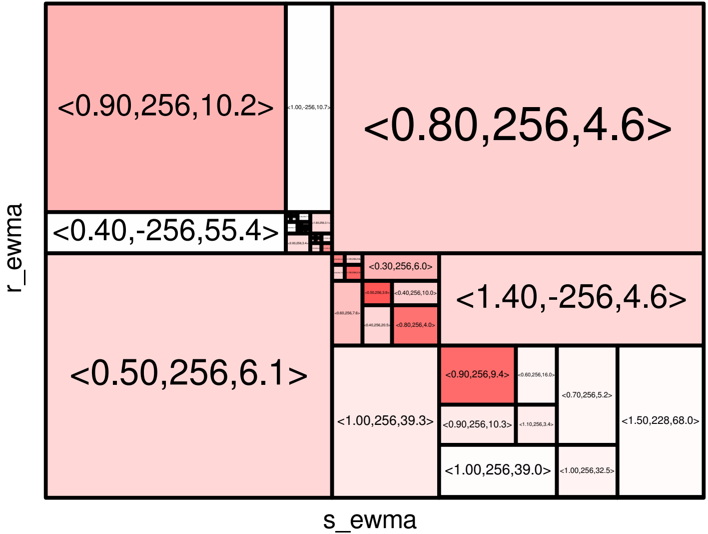
Split


<0.90,256,10.2>
<1.00,-256,10.7>
<0.80,256,4.6>
<0.40,-256,55.4>
<0.30,256,6.0>
<0.50,256,3.9>
<0.40,256,10.0>
<1.40,-256,4.6>
<0.60,256,7.6>
<0.40,256,20.5>
<0.80,256,4.0>
<0.50,256,6.1>
<0.90,256,9.4>
<0.60,256,16.0>
<0.70,256,5.2>
<1.00,256,39.3>
<0.90,256,10.3>
<1.10,256,3.4>
<1.50,228,68.0>
<1.00,256,39.0>
<1.00,256,32.5>
r_ewma
s_ewma

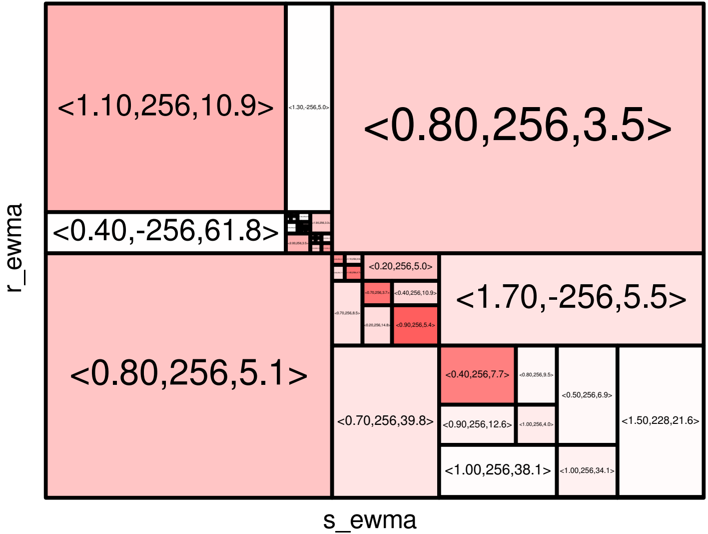
Simulate


<0.90,256,10.2>
<1.00,-256,10.7>
<0.80,256,4.6>
<0.40,-256,55.4>
<0.30,256,6.0>
<0.50,256,3.9>
<0.40,256,10.0>
<0.60,256,7.6>
<0.40,256,20.5>
<0.80,256,4.0>
<1.40,-256,4.6>
<0.50,256,6.1>
<1.00,256,39.3>
<0.90,256,9.4>
<0.60,256,16.0>
<0.70,256,5.2>
<1.50,228,68.0>
<0.90,256,10.3>
<1.10,256,3.4>
<1.00,256,39.0>
<1.00,256,32.5>
r_ewma
s_ewma

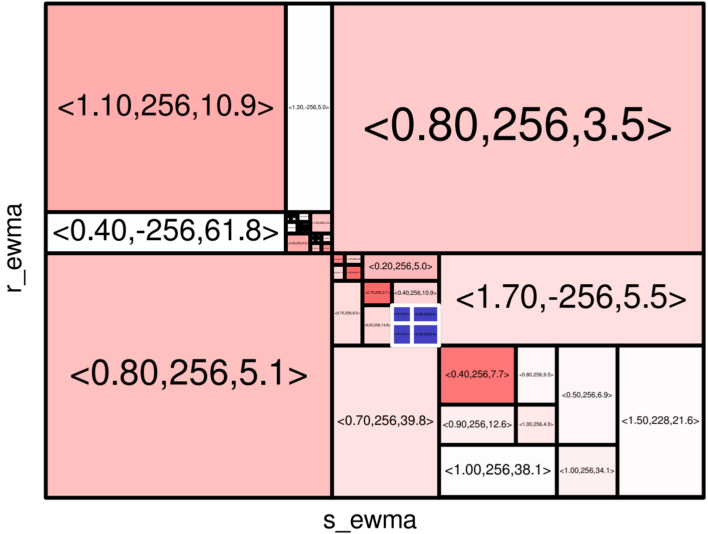
Optimize


<0.90,256,10.2>
<1.00,-256,10.7>
<0.80,256,4.6>
<0.40,-256,55.4>
<1.80,256,3.1>
<2.00,256,3.4>
<0.30,256,6.0>
<0.50,256,3.9>
<0.40,256,10.0>
<0.60,256,7.6>
<0.40,256,20.5>
<0.80,256,4.0>
<1.40,-256,4.6>
<0.50,256,6.1>
<1.00,256,39.3>
<0.90,256,9.4>
<0.60,256,16.0>
<0.70,256,5.2>
<1.50,228,68.0>
<0.90,256,10.3>
<1.10,256,3.4>
<1.00,256,39.0>
<1.00,256,32.5>
r_ewma
s_ewma

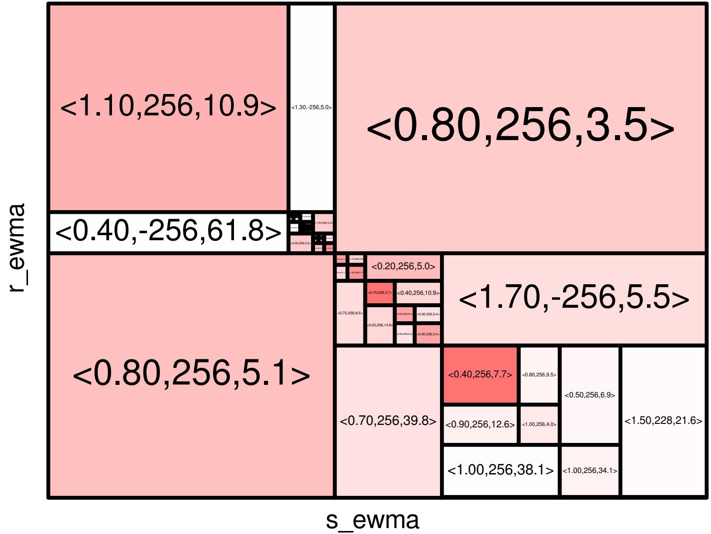
Split


<1.10,256,10.9>
<1.30,-256,5.0>
<0.80,256,3.5>
<0.40,-256,61.8>
<0.20,256,5.0>
<0.70,256,3.7>
<0.40,256,10.9>
<1.70,-256,5.5>
<0.70,256,8.5>
<0.20,256,14.8>
<0.90,256,5.4>
<0.80,256,5.1>
<0.70,256,39.8>
<0.40,256,7.7>
<0.80,256,9.5>
<0.50,256,6.9>
<1.50,228,21.6>
<0.90,256,12.6>
<1.00,256,4.0>
<1.00,256,38.1>
<1.00,256,34.1>
r_ewma
s_ewma

Simulate

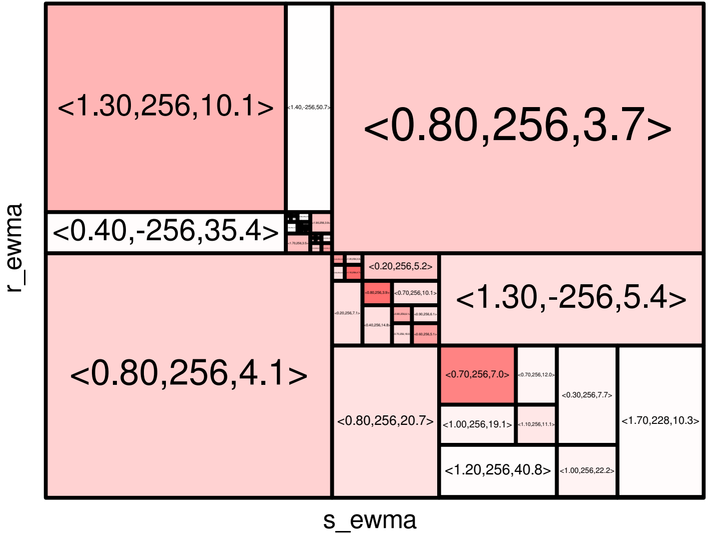

<1.10,256,10.9>
<1.30,-256,5.0>
<0.80,256,3.5>
<0.40,-256,61.8>
<0.20,256,5.0>
<0.70,256,3.7>
<0.40,256,10.9>
<1.70,-256,5.5>
<0.70,256,8.5>
<0.20,256,14.8>
<0.80,256,5.1>
<0.40,256,7.7>
<0.80,256,9.5>
<0.50,256,6.9>
<0.70,256,39.8>
<1.50,228,21.6>
<0.90,256,12.6>
<1.00,256,4.0>
<1.00,256,38.1>
<1.00,256,34.1>
r_ewma
s_ewma

Optimize

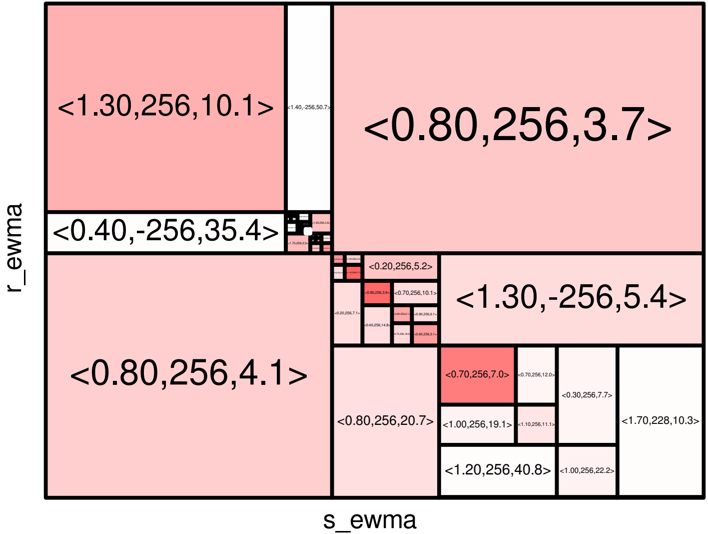


r_ewma
s_ewma
<1.10,256,10.9>
<1.30,-256,5.0>
<0.80,256,3.5>
<0.40,-256,61.8>
<0.20,256,5.0>
<1.70,-256,5.5>
<0.40,256,10.9>
<0.80,256,5.1>
<0.40,256,7.7>
<0.80,256,9.5>
<0.50,256,6.9>
<1.50,228,21.6>
<0.70,256,39.8>
<0.90,256,12.6>
<1.00,256,4.0>
<1.00,256,38.1>
<1.00,256,34.1>

Split

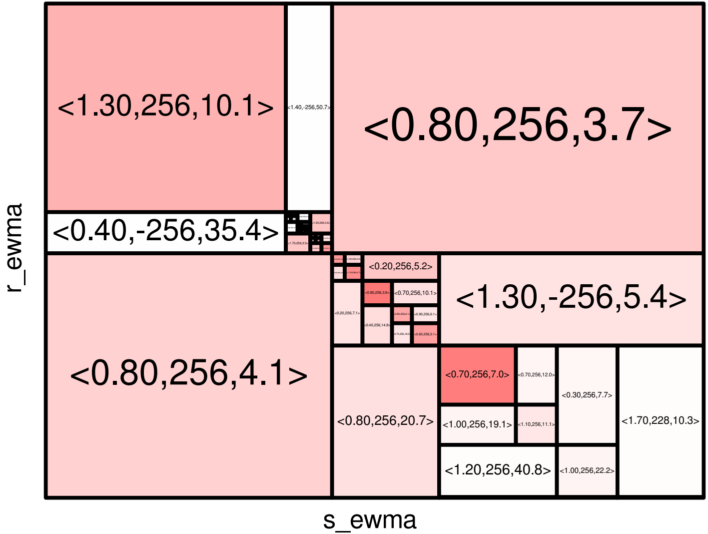

<1.30,256,10.1>
<1.40,-256,50.7>
<0.80,256,3.7>
<0.40,-256,35.4>
<0.20,256,5.2>
<1.30,-256,5.4>
<0.70,256,10.1>
<0.20,256,7.1>
<0.40,256,14.8>
<0.80,256,4.1>
<0.70,256,7.0>
<0.70,256,12.0>
<0.30,256,7.7>
<0.80,256,20.7>
<1.00,256,19.1>
<1.10,256,11.1>
<1.70,228,10.3>
<1.20,256,40.8>
<1.00,256,22.2>
r_ewma
s_ewma

Simulate

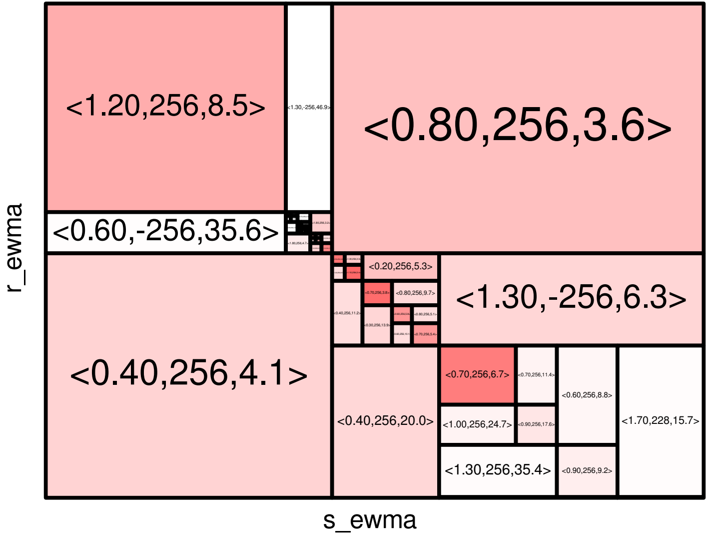


r_ewma
s_ewma
<1.30,256,10.1>
<1.40,-256,50.7>
<0.80,256,3.7>
<0.40,-256,35.4>
<0.80,256,4.1>
<0.20,256,5.2>
<1.30,-256,5.4>
<0.80,256,3.9>
<0.70,256,10.1>
<0.20,256,7.1>
<0.40,256,14.8>
<0.90,256,6.1>
<0.60,256,5.1>
<0.70,256,7.0>
<0.70,256,12.0>
<0.30,256,7.7>
<0.80,256,20.7>
<1.00,256,19.1>
<1.10,256,11.1>
<1.70,228,10.3>
<1.20,256,40.8>
<1.00,256,22.2>

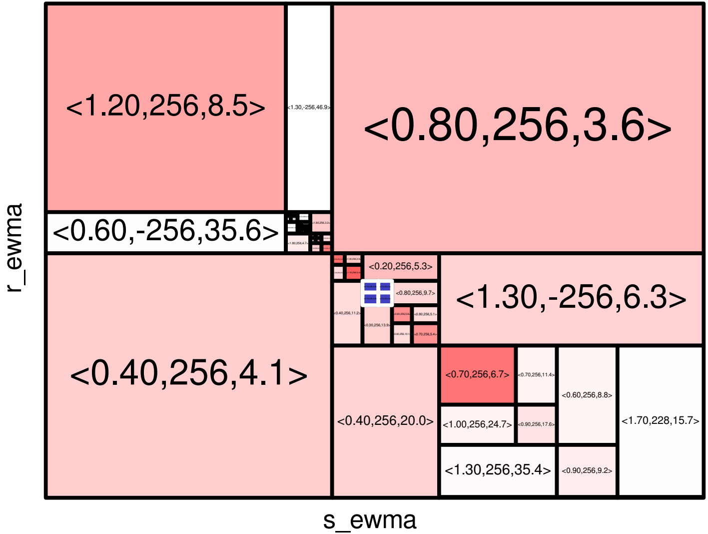
Optimize


r_ewma
s_ewma
<1.30,256,10.1>
<1.40,-256,50.7>
<0.80,256,3.7>
<0.40,-256,35.4>
<0.20,256,5.2>
<0.80,256,3.9>
<0.70,256,10.1>
<0.20,256,7.1>
<0.40,256,14.8>
<0.90,256,6.1>
<0.60,256,5.1>
<1.30,-256,5.4>
<0.80,256,4.1>
<0.70,256,7.0>
<0.70,256,12.0>
<0.30,256,7.7>
<0.80,256,20.7>
<1.00,256,19.1>
<1.10,256,11.1>
<1.70,228,10.3>
<1.20,256,40.8>
<1.00,256,22.2>

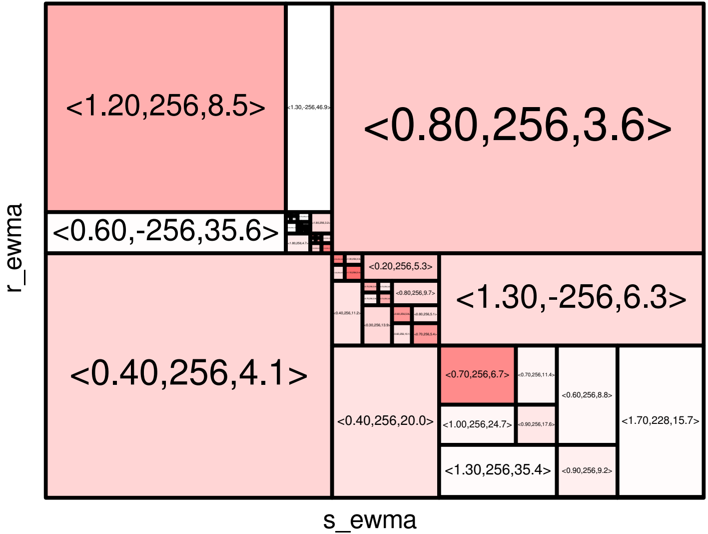
Split


r_ewma
s_ewma
<1.20,256,8.5>
<1.30,-256,46.9>
<0.80,256,3.6>
<0.60,-256,35.6>
<0.20,256,5.3>
<1.30,-256,6.3>
<0.80,256,9.7>
<0.40,256,4.1>
<0.70,256,6.7>
<0.70,256,11.4>
<0.60,256,8.8>
<0.40,256,20.0>
<1.00,256,24.7>
<0.90,256,17.6>
<1.70,228,15.7>
<1.30,256,35.4>
<0.90,256,9.2>

Simulate

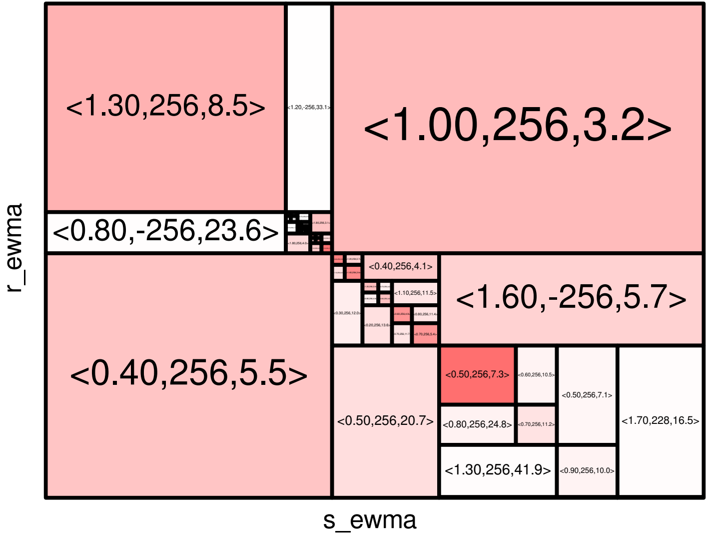

<1.20,256,8.5>
<1.30,-256,46.9>
<0.80,256,3.6>
<0.60,-256,35.6>
<0.20,256,5.3>
<0.80,256,9.7>
<1.30,-256,6.3>
<0.40,256,4.1>
<0.40,256,11.2>
<0.30,256,13.9>
<0.80,256,5.1>
<0.70,256,5.4>
<0.70,256,6.7>
<0.70,256,11.4>
<0.60,256,8.8>
<0.40,256,20.0>
<1.00,256,24.7>
<0.90,256,17.6>
<1.70,228,15.7>
<1.30,256,35.4>
<0.90,256,9.2>
r_ewma
s_ewma

Optimize

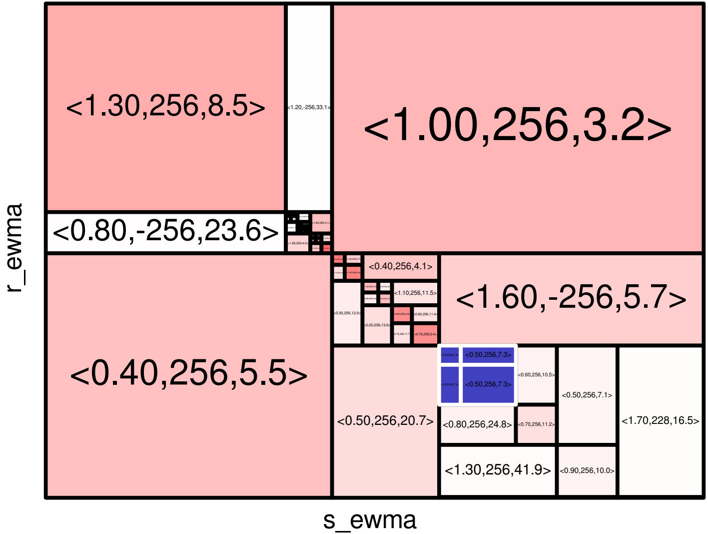


<1.20,256,8.5>
<1.30,-256,46.9>
<0.80,256,3.6>
<0.60,-256,35.6>
<0.20,256,5.3>
<0.80,256,9.7>
<1.30,-256,6.3>
<0.40,256,4.1>
<0.70,256,6.7>
<0.70,256,11.4>
<0.60,256,8.8>
<0.40,256,20.0>
<1.00,256,24.7>
<0.90,256,17.6>
<1.70,228,15.7>
<1.30,256,35.4>
<0.90,256,9.2>
r_ewma
s_ewma

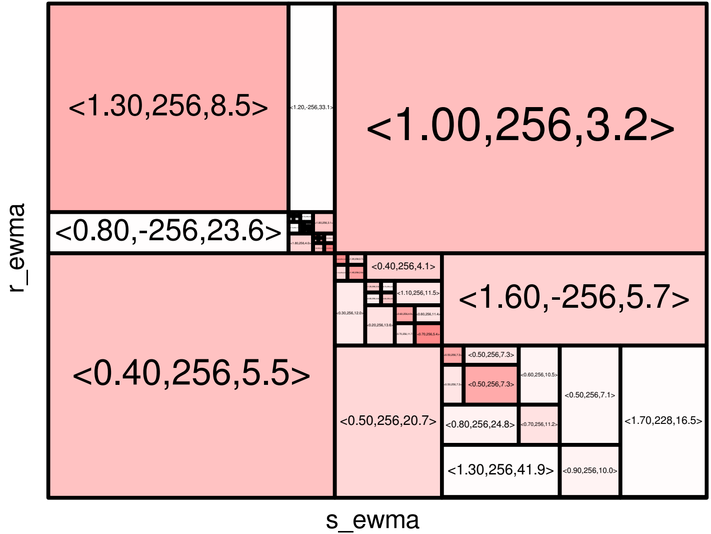
Split


<1.30,256,8.5>
<1.20,-256,33.1>
<1.00,256,3.2>
<0.80,-256,23.6>
<0.40,256,4.1>
<1.10,256,11.5>
<1.60,-256,5.7>
<0.30,256,12.0>
<0.20,256,13.6>
<0.40,256,5.5>
<0.50,256,20.7>
<0.50,256,7.3>
<0.60,256,10.5>
<0.50,256,7.1>
<1.70,228,16.5>
<0.80,256,24.8>
<0.70,256,11.2>
<1.30,256,41.9>
<0.90,256,10.0>
r_ewma
s_ewma

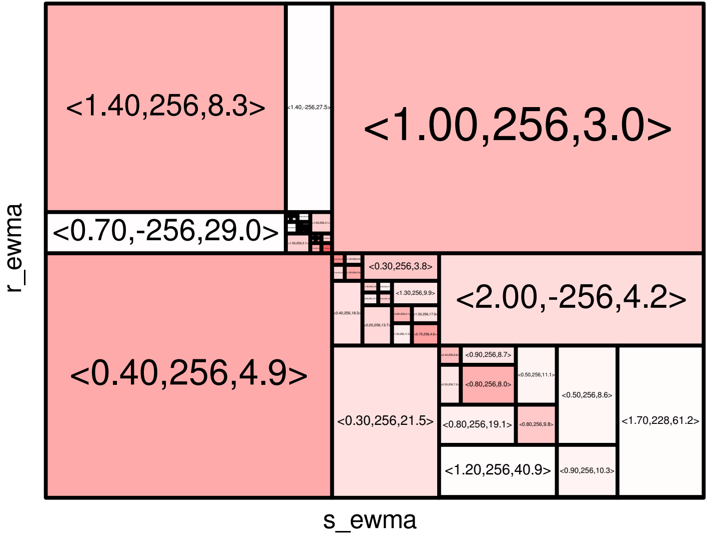
Simulate


r_ewma
s_ewma
<1.30,256,8.5>
<1.20,-256,33.1>
<1.00,256,3.2>
<0.80,-256,23.6>
<0.40,256,4.1>
<1.10,256,11.5>
<1.60,-256,5.7>
<0.40,256,5.5>
<0.50,256,7.3>
<0.50,256,7.3>
<0.60,256,10.5>
<0.50,256,7.1>
<0.50,256,20.7>
<1.70,228,16.5>
<0.80,256,24.8>
<0.70,256,11.2>
<1.30,256,41.9>
<0.90,256,10.0>

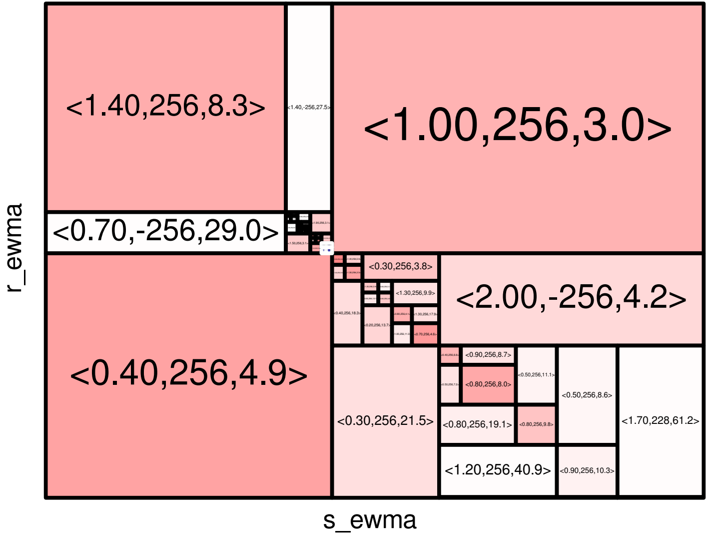
Optimize


r_ewma
s_ewma
<1.30,256,8.5>
<1.20,-256,33.1>
<1.00,256,3.2>
<0.80,-256,23.6>
<0.40,256,4.1>
<1.10,256,11.5>
<1.60,-256,5.7>
<0.30,256,12.0>
<0.20,256,13.6>
<0.80,256,11.4>
<0.70,256,5.4>
<0.40,256,5.5>
<0.50,256,7.3>
<0.50,256,7.3>
<0.60,256,10.5>
<0.50,256,7.1>
<0.50,256,20.7>
<0.80,256,24.8>
<0.70,256,11.2>
<1.70,228,16.5>
<1.30,256,41.9>
<0.90,256,10.0>

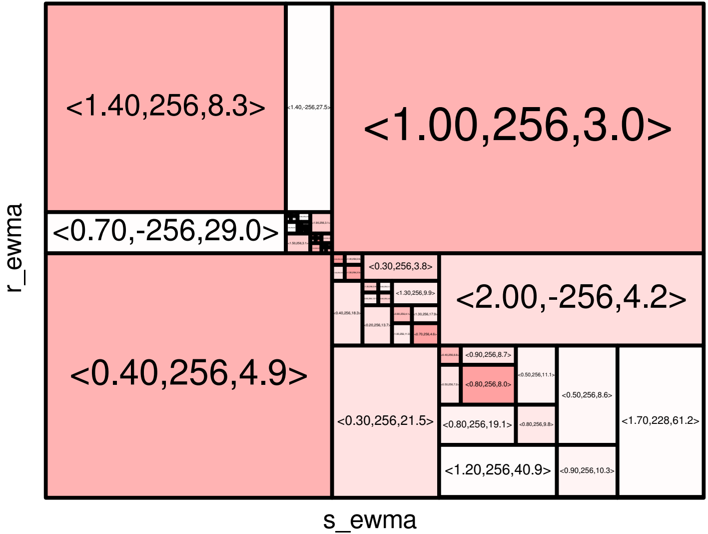
Split


<1.40,256,8.3>
<1.40,-256,27.5>
<1.00,256,3.0>
<0.70,-256,29.0>
<0.30,256,3.8>
<2.00,-256,4.2>
<1.30,256,9.9>
<0.40,256,18.3>
<0.20,256,13.7>
<1.30,256,17.9>
<0.70,256,4.6>
<0.40,256,4.9>
<0.30,256,21.5>
<0.90,256,8.7>
<0.50,256,11.1>
<0.80,256,8.0>
<0.50,256,8.6>
<1.70,228,61.2>
<0.80,256,19.1>
<0.80,256,9.8>
<1.20,256,40.9>
<0.90,256,10.3>
r_ewma
s_ewma

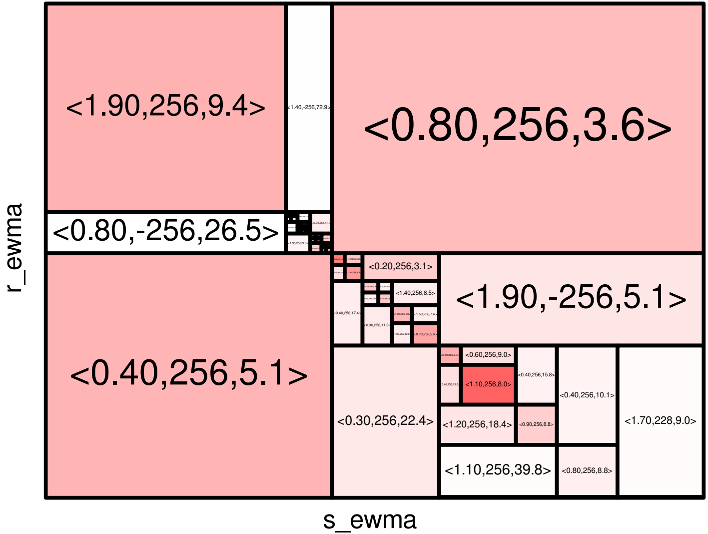
Simulate


<1.40,256,8.3>
<1.40,-256,27.5>
<1.00,256,3.0>
<0.70,-256,29.0>
<0.40,256,4.9>
<0.30,256,3.8>
<1.30,256,9.9>
<2.00,-256,4.2>
<0.90,256,8.7>
<0.80,256,8.0>
<0.50,256,11.1>
<0.50,256,8.6>
<0.30,256,21.5>
<0.80,256,19.1>
<0.80,256,9.8>
<1.70,228,61.2>
<1.20,256,40.9>
<0.90,256,10.3>
r_ewma
s_ewma

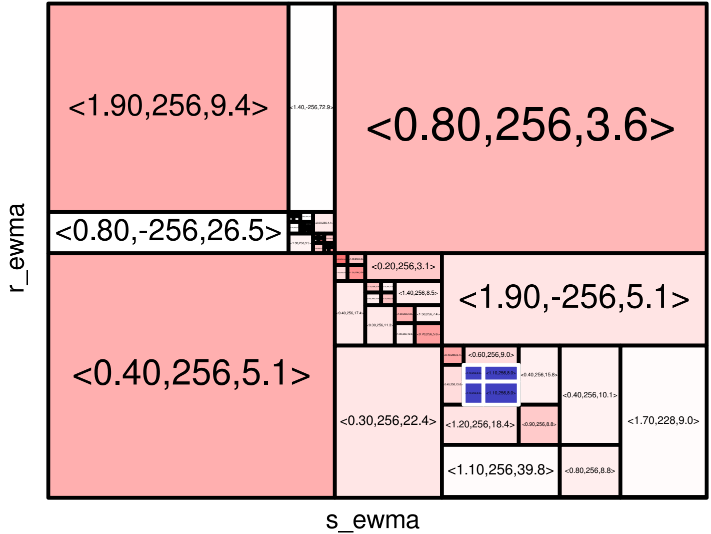
Optimize


<1.40,256,8.3>
<1.40,-256,27.5>
<1.00,256,3.0>
<0.70,-256,29.0>
<0.30,256,3.8>
<2.00,-256,4.2>
<1.30,256,9.9>
<0.40,256,18.3>
<0.20,256,13.7>
<1.30,256,17.9>
<0.70,256,4.6>
<0.40,256,4.9>
<0.90,256,8.7>
<0.50,256,11.1>
<0.80,256,8.0>
<0.50,256,8.6>
<1.70,228,61.2>
<0.30,256,21.5>
<0.80,256,19.1>
<0.80,256,9.8>
<1.20,256,40.9>
<0.90,256,10.3>
r_ewma
s_ewma

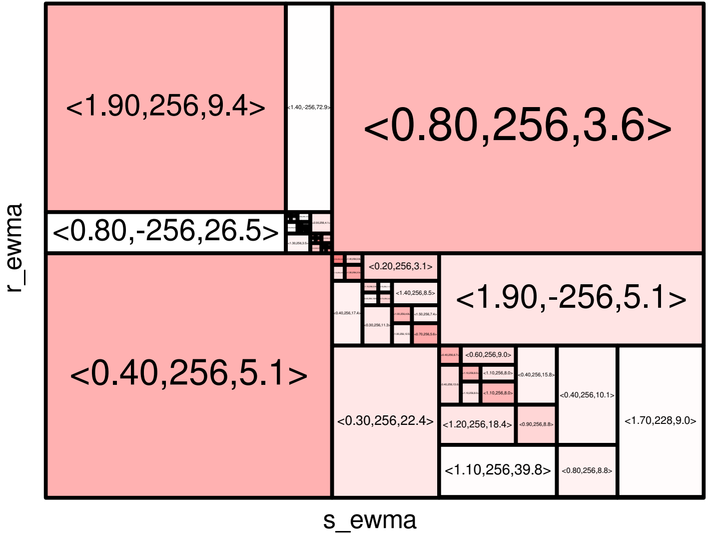
Split


<1.90,256,9.4>
<1.40,-256,72.9>
<0.80,256,3.6>
<0.80,-256,26.5>
<0.20,256,3.1>
<1.90,-256,5.1>
<1.40,256,8.5>
<0.40,256,5.1>
<0.60,256,9.0>
<1.10,256,8.0>
<0.40,256,15.8>
<0.40,256,10.1>
<0.30,256,22.4>
<1.20,256,18.4>
<0.90,256,8.8>
<1.70,228,9.0>
<1.10,256,39.8>
<0.80,256,8.8>
r_ewma
s_ewma

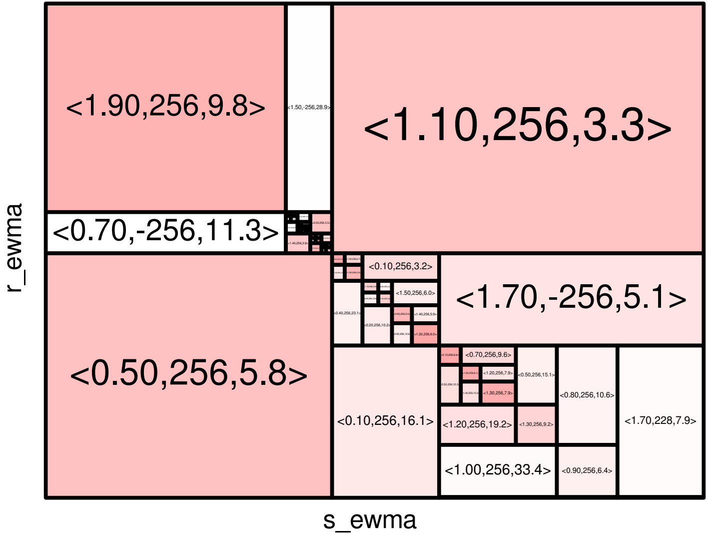
Simulate


<1.90,256,9.4>
<1.40,-256,72.9>
<0.80,256,3.6>
<0.80,-256,26.5>
<0.40,256,5.1>
<0.20,256,3.1>
<1.90,-256,5.1>
<1.40,256,8.5>
<0.60,256,9.0>
<0.40,256,15.8>
<0.40,256,10.1>
<0.30,256,22.4>
<1.20,256,18.4>
<0.90,256,8.8>
<1.70,228,9.0>
<1.10,256,39.8>
<0.80,256,8.8>
r_ewma
s_ewma

Optimize

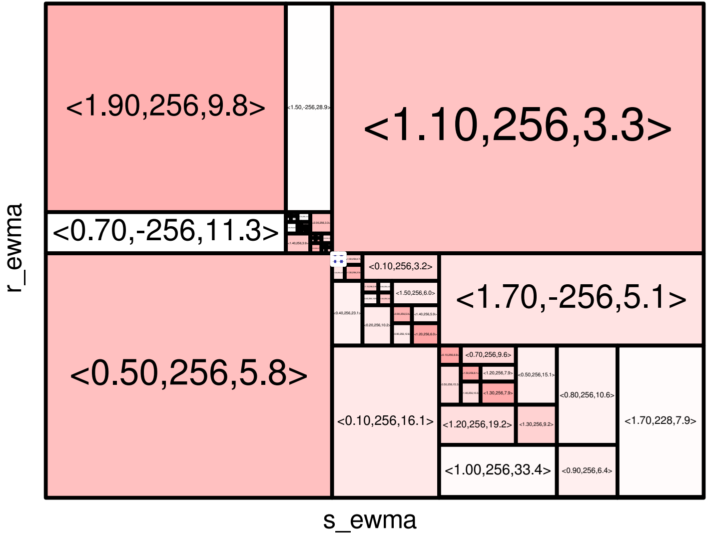

<1.90,256,9.4>
<1.40,-256,72.9>
<0.80,256,3.6>
<0.80,-256,26.5>
<0.20,256,3.1>
<1.40,256,8.5>
<1.90,-256,5.1>
<0.40,256,5.1>
<0.60,256,9.0>
<1.10,256,8.0>
<0.40,256,15.8>
<1.10,256,8.0>
<0.40,256,10.1>
<0.30,256,22.4>
<1.20,256,18.4>
<1.70,228,9.0>
<1.10,256,39.8>
<0.80,256,8.8>
r_ewma
s_ewma

Split

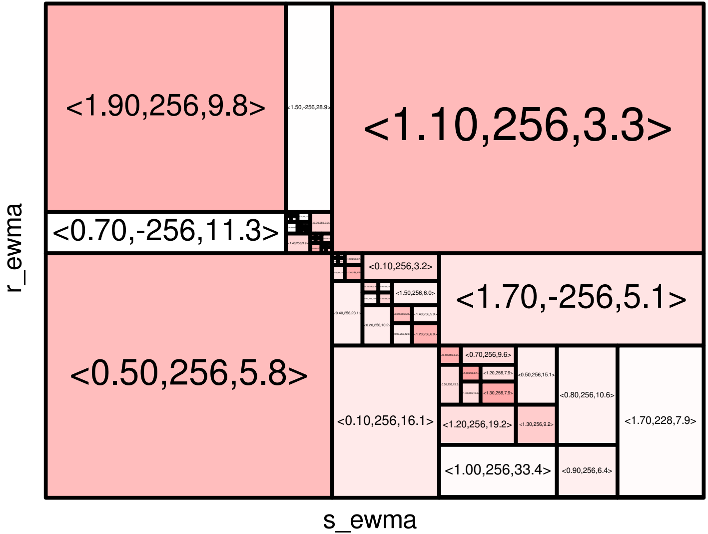


<1.90,256,9.8>
<1.50,-256,28.9>
<1.10,256,3.3>
<0.70,-256,11.3>
<1.40,256,3.9>
<0.10,256,3.2>
<1.50,256,6.0>
<1.70,-256,5.1>
<0.40,256,23.1>
<0.20,256,10.2>
<1.40,256,5.9>
<1.20,256,6.0>
<0.50,256,5.8>
<0.70,256,9.6>
<1.20,256,7.9>
<0.50,256,15.1>
<1.30,256,7.9>
<0.80,256,10.6>
<0.10,256,16.1>
<1.20,256,19.2>
<1.30,256,9.2>
<1.70,228,7.9>
<1.00,256,33.4>
<0.90,256,6.4>
r_ewma
s_ewma

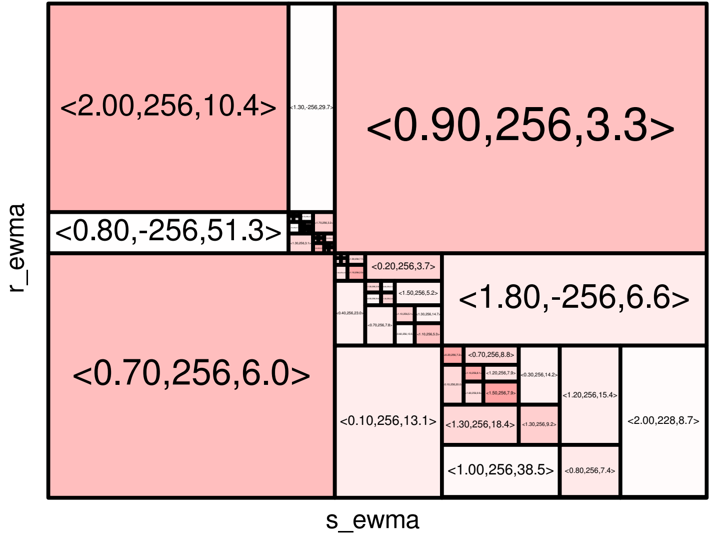
Simulate


<1.90,256,9.8>
<1.50,-256,28.9>
<1.10,256,3.3>
<0.70,-256,11.3>
<0.10,256,3.2>
<1.50,256,6.0>
<1.70,-256,5.1>
<0.40,256,23.1>
<0.20,256,10.2>
<1.40,256,5.9>
<1.20,256,6.0>
<0.50,256,5.8>
<0.70,256,9.6>
<1.20,256,7.9>
<0.50,256,15.1>
<1.30,256,7.9>
<0.80,256,10.6>
<0.10,256,16.1>
<1.20,256,19.2>
<1.30,256,9.2>
<1.70,228,7.9>
<1.00,256,33.4>
<0.90,256,6.4>
r_ewma
s_ewma

Optimize

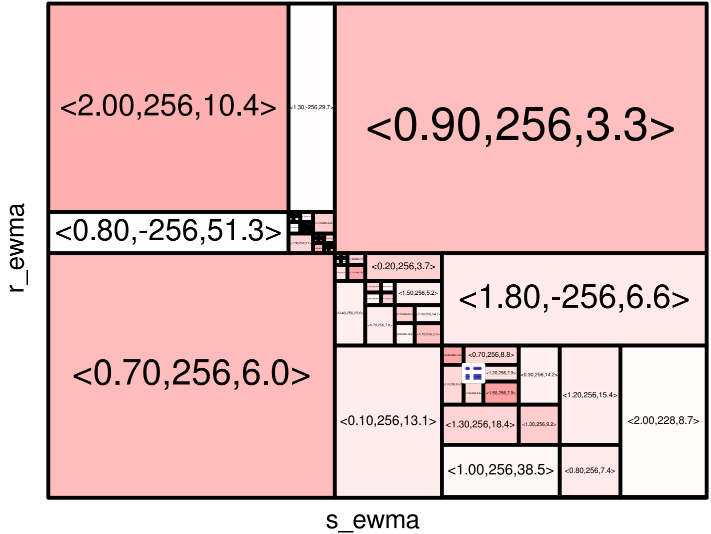

r_ewma
s_ewma
<1.90,256,9.8>
<1.50,-256,28.9>
<1.10,256,3.3>
<0.70,-256,11.3>
<0.50,256,5.8>
<0.10,256,3.2>
<1.50,256,6.0>
<1.70,-256,5.1>
<0.40,256,23.1>
<0.20,256,10.2>
<1.40,256,5.9>
<1.20,256,6.0>
<0.70,256,9.6>
<1.20,256,7.9>
<1.30,256,7.9>
<0.50,256,15.1>
<0.80,256,10.6>
<0.10,256,16.1>
<1.20,256,19.2>
<1.30,256,9.2>
<1.70,228,7.9>
<1.00,256,33.4>
<0.90,256,6.4>

Split

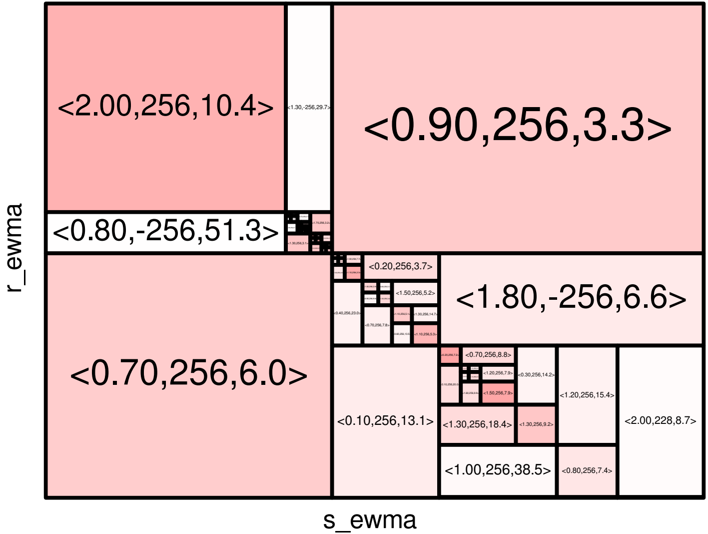


r_ewma
s_ewma
<2.00,256,10.4>
<1.30,-256,29.7>
<0.90,256,3.3>
<0.80,-256,51.3>
<0.20,256,3.7>
<1.50,256,5.2>
<1.80,-256,6.6>
<0.70,256,6.0>
<0.70,256,8.8>
<0.30,256,14.2>
<1.20,256,15.4>
<0.10,256,13.1>
<1.30,256,18.4>
<2.00,228,8.7>
<1.00,256,38.5>
<0.80,256,7.4>

Simulate

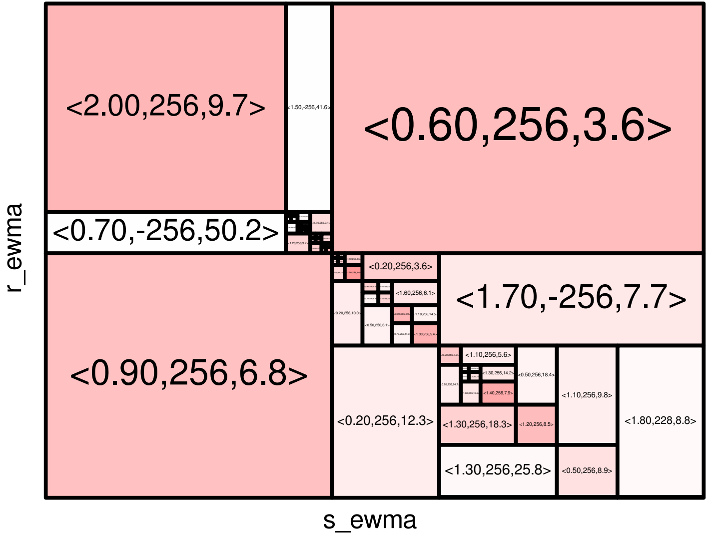

r_ewma
<2.00,256,10.4>
<1.30,-256,29.7>
<0.90,256,3.3>
<0.80,-256,51.3>
<0.70,256,6.0>
<0.20,256,3.7>
<1.50,256,5.2>
<1.80,-256,6.6>
<0.70,256,8.8>
<0.10,256,13.1>
<1.20,256,15.4>
<2.00,228,8.7>
<1.30,256,18.4>
<1.00,256,38.5>
<0.80,256,7.4>
s_ewma

Optimize

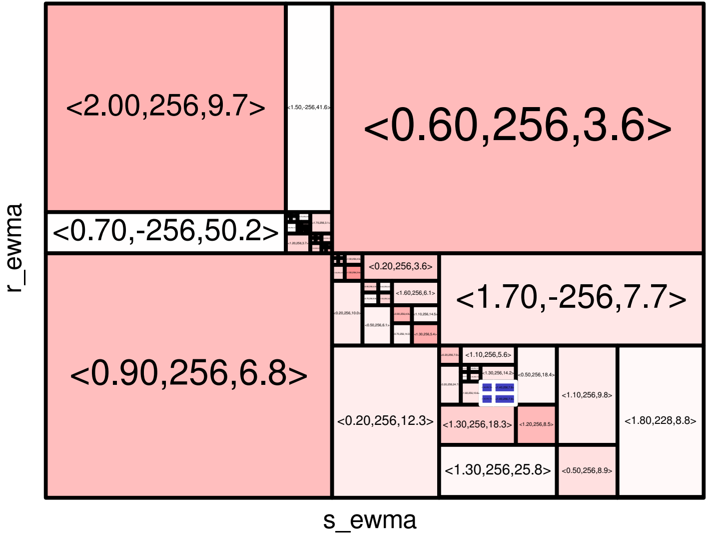


<2.00,256,10.4>
<1.30,-256,29.7>
<0.90,256,3.3>
<0.80,-256,51.3>
<0.20,256,3.7>
<1.50,256,5.2>
<1.80,-256,6.6>
<0.40,256,23.0>
<0.70,256,7.8>
<1.30,256,14.7>
<1.10,256,5.3>
<0.70,256,6.0>
<0.70,256,8.8>
<1.20,256,7.9>
<0.30,256,14.2>
<1.50,256,7.9>
<1.20,256,15.4>
<0.10,256,13.1>
<1.30,256,18.4>
<1.30,256,9.2>
<2.00,228,8.7>
<1.00,256,38.5>
<0.80,256,7.4>
r_ewma
s_ewma

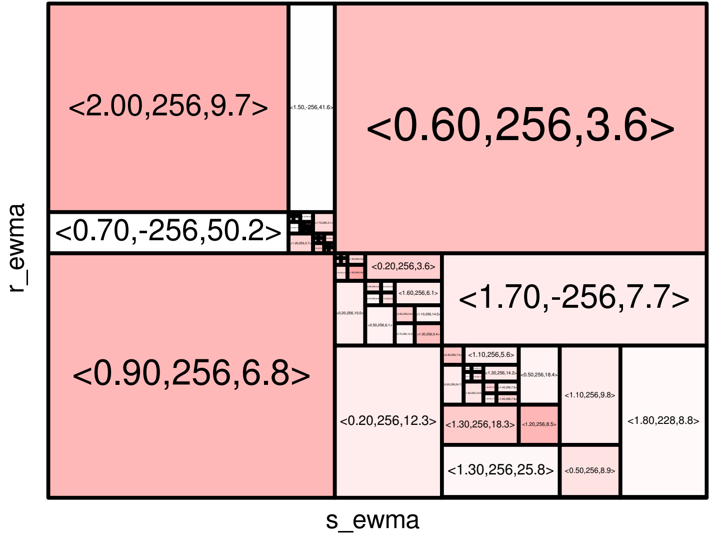
Split


<2.00,256,9.7>
<1.50,-256,41.6>
<0.60,256,3.6>
<0.70,-256,50.2>
<0.20,256,3.6>
<1.60,256,6.1>
<1.70,-256,7.7>
<0.20,256,10.0>
<0.50,256,6.1>
<1.10,256,14.5>
<1.30,256,5.4>
<0.90,256,6.8>
<1.10,256,5.6>
<1.30,256,14.2>
<0.50,256,18.4>
<1.40,256,7.9>
<1.10,256,9.8>
<0.20,256,12.3>
<1.30,256,18.3>
<1.20,256,8.5>
<1.80,228,8.8>
<1.30,256,25.8>
<0.50,256,8.9>
r_ewma
s_ewma

Simulate

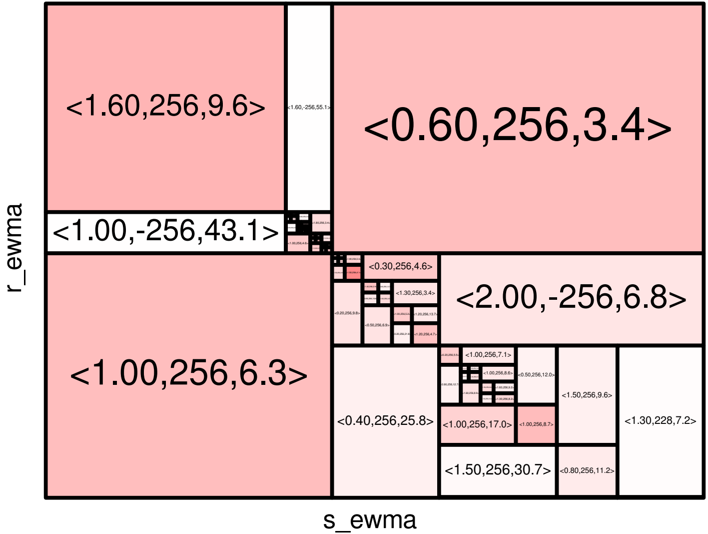

<2.00,256,9.7>
<1.50,-256,41.6>
<0.60,256,3.6>
<0.70,-256,50.2>
<0.20,256,3.6>
<1.60,256,6.1>
<1.70,-256,7.7>
<0.20,256,10.0>
<0.50,256,6.1>
<1.10,256,14.5>
<1.30,256,5.4>
<0.90,256,6.8>
<1.10,256,5.6>
<1.30,256,14.2>
<0.50,256,18.4>
<1.10,256,9.8>
<0.20,256,12.3>
<1.30,256,18.3>
<1.20,256,8.5>
<1.80,228,8.8>
<1.30,256,25.8>
<0.50,256,8.9>
r_ewma
s_ewma

RemyCC

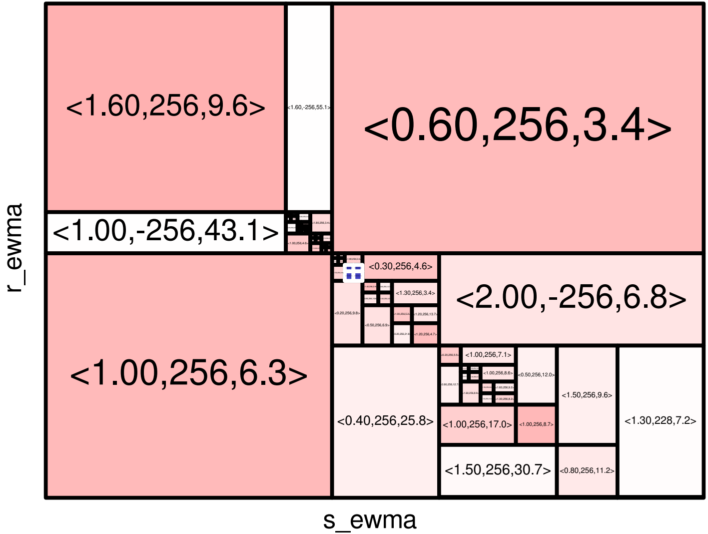


r_ewma
s_ewma
<2.00,256,9.7>
<1.50,-256,41.6>
<0.60,256,3.6>
<0.70,-256,50.2>
<0.20,256,3.6>
<1.60,256,6.1>
<1.70,-256,7.7>
<0.90,256,6.8>
<1.10,256,5.6>
<0.50,256,18.4>
<1.10,256,9.8>
<0.20,256,12.3>
<1.30,256,18.3>
<1.80,228,8.8>
<1.30,256,25.8>
<0.50,256,8.9>

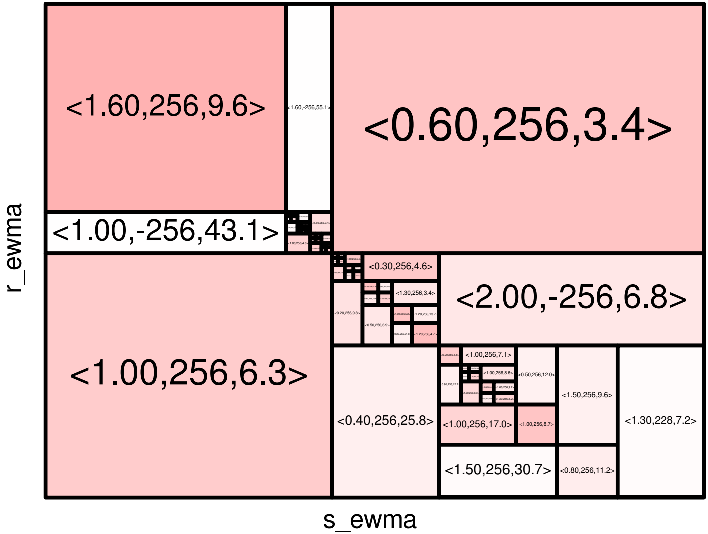
RemyCC


<1.60,256,9.6>
<1.60,-256,55.1>
<0.60,256,3.4>
<1.00,-256,43.1>
<1.00,256,6.3>
<0.30,256,4.6>
<1.30,256,3.4>
<2.00,-256,6.8>
<1.00,256,7.1>
<1.00,256,8.6>
<0.50,256,12.0>
<1.50,256,9.6>
<0.40,256,25.8>
<1.00,256,17.0>
<1.00,256,8.7>
<1.30,228,7.2>
<1.50,256,30.7>
<0.80,256,11.2>
r_ewma
s_ewma

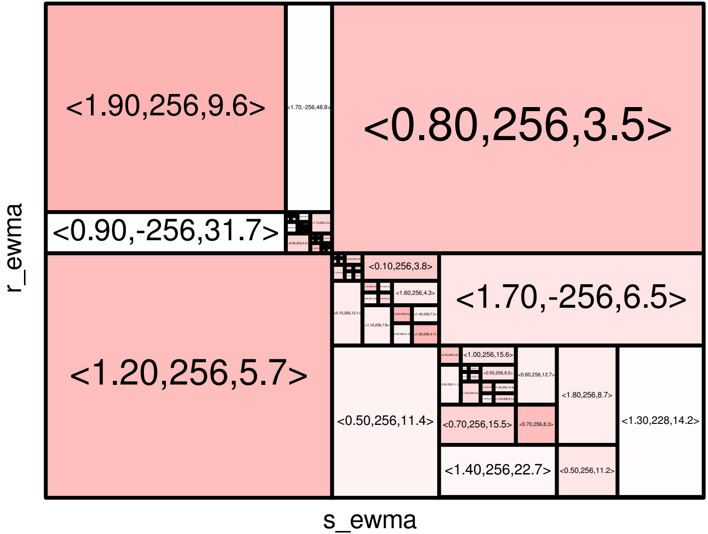
RemyCC


<1.60,256,9.6>
<1.60,-256,55.1>
<0.60,256,3.4>
<1.00,-256,43.1>
<0.30,256,4.6>
<1.30,256,3.4>
<2.00,-256,6.8>
<1.00,256,6.3>
<1.00,256,7.1>
<0.50,256,12.0>
<1.50,256,9.6>
<0.40,256,25.8>
<1.00,256,17.0>
<1.00,256,8.7>
<1.30,228,7.2>
<1.50,256,30.7>
<0.80,256,11.2>
r_ewma
s_ewma

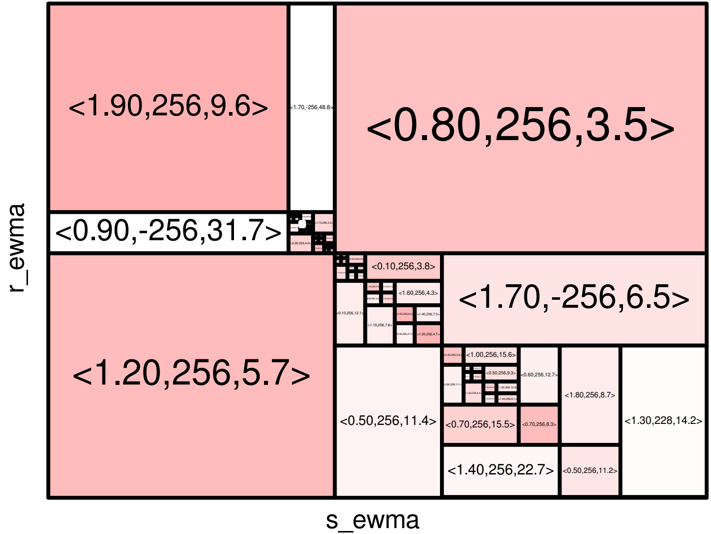
RemyCC


<1.60,256,9.6>
<1.60,-256,55.1>
<0.60,256,3.4>
<1.00,-256,43.1>
<1.00,256,6.3>
<0.30,256,4.6>
<1.30,256,3.4>
<2.00,-256,6.8>
<1.00,256,7.1>
<0.50,256,12.0>
<1.50,256,9.6>
<0.40,256,25.8>
<1.00,256,17.0>
<1.00,256,8.7>
<1.30,228,7.2>
<1.50,256,30.7>
<0.80,256,11.2>
r_ewma
s_ewma

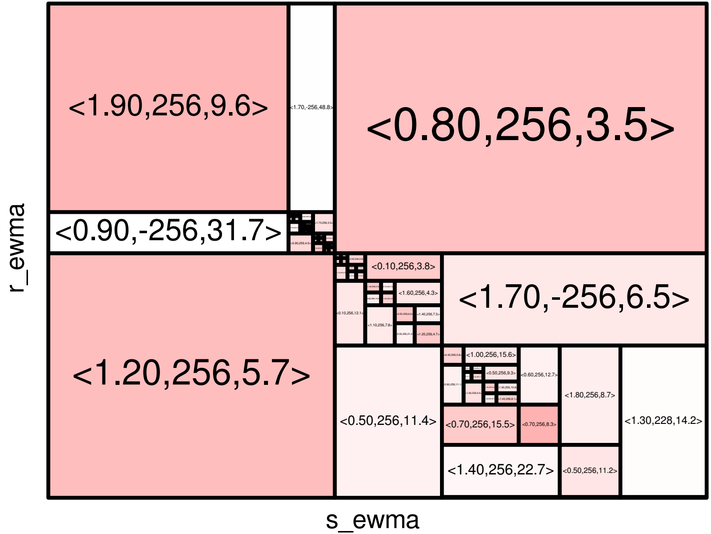
RemyCC


<1.90,256,9.6>
<1.70,-256,48.8>
<0.80,256,3.5>
<0.90,-256,31.7>
<1.70,256,3.3>
<0.90,256,4.0>
<0.10,256,3.8>
<1.60,256,4.3>
<1.70,-256,6.5>
<0.10,256,12.1>
<1.10,256,7.8>
<1.40,256,7.3>
<1.20,256,4.7>
<1.20,256,5.7>
<1.00,256,15.6>
<0.50,256,9.3>
<0.60,256,12.7>
<1.80,256,8.7>
<0.50,256,11.4>
<0.70,256,15.5>
<0.70,256,8.3>
<1.30,228,14.2>
<1.40,256,22.7>
<0.50,256,11.2>
r_ewma
s_ewma

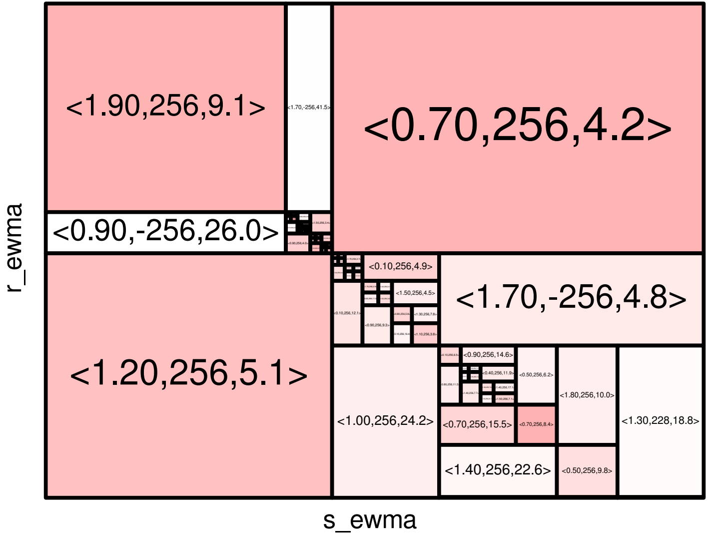
RemyCC


<1.90,256,9.6>
<1.70,-256,48.8>
<0.80,256,3.5>
<0.90,-256,31.7>
<0.10,256,3.8>
<1.60,256,4.3>
<1.70,-256,6.5>
<1.20,256,5.7>
<1.00,256,15.6>
<0.60,256,12.7>
<1.80,256,8.7>
<0.50,256,11.4>
<0.70,256,15.5>
<0.70,256,8.3>
<1.30,228,14.2>
<1.40,256,22.7>
<0.50,256,11.2>
r_ewma
s_ewma

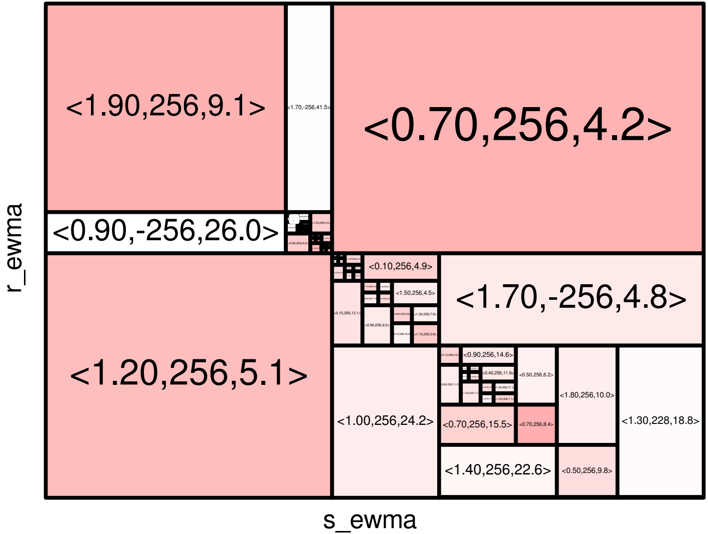
RemyCC


<1.90,256,9.6>
<1.70,-256,48.8>
<0.80,256,3.5>
<0.90,-256,31.7>
<0.10,256,3.8>
<1.60,256,4.3>
<1.70,-256,6.5>
<0.10,256,12.1>
<1.10,256,7.8>
<1.40,256,7.3>
<1.20,256,4.7>
<1.20,256,5.7>
<1.00,256,15.6>
<0.50,256,9.3>
<0.60,256,12.7>
<1.80,256,8.7>
<0.50,256,11.4>
<0.70,256,15.5>
<0.70,256,8.3>
<1.30,228,14.2>
<1.40,256,22.7>
<0.50,256,11.2>
r_ewma
s_ewma

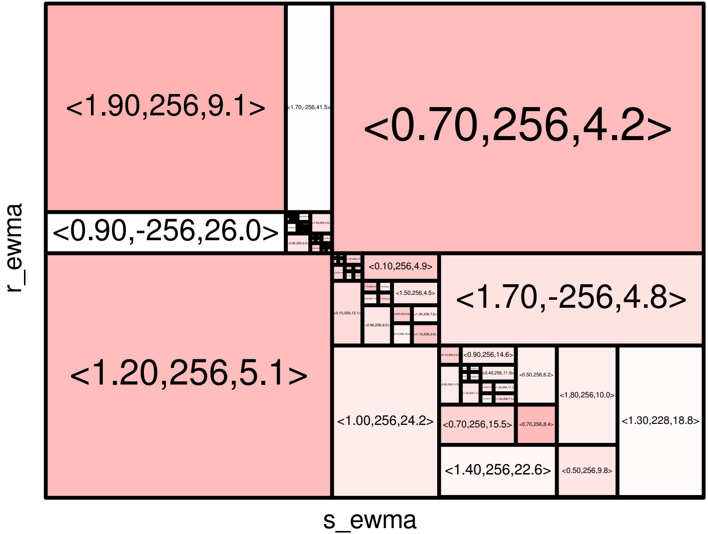
RemyCC


<1.90,256,9.1>
<1.70,-256,41.5>
<0.70,256,4.2>
<0.90,-256,26.0>
<0.10,256,4.9>
<1.70,-256,4.8>
<1.50,256,4.5>
<1.20,256,5.1>
<0.90,256,14.6>
<0.40,256,11.9>
<0.50,256,6.2>
<1.80,256,10.0>
<1.00,256,24.2>
<0.70,256,15.5>
<0.70,256,8.4>
<1.30,228,18.8>
<1.40,256,22.6>
<0.50,256,9.8>
r_ewma
s_ewma

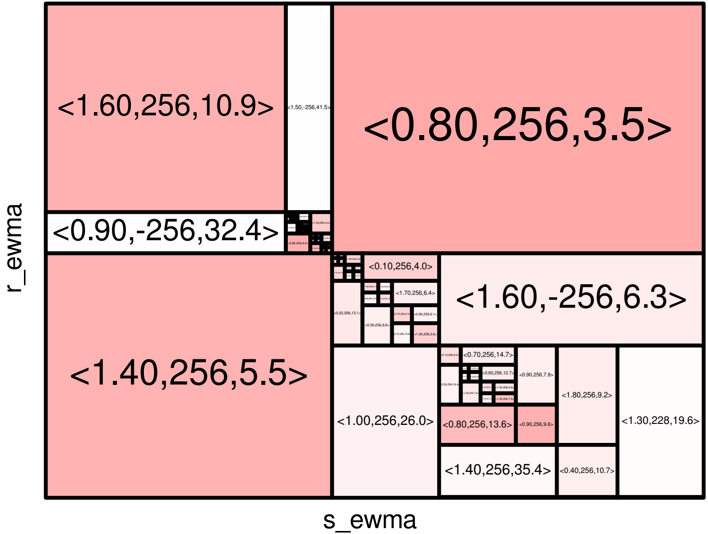
RemyCC


<1.90,256,9.1>
<1.70,-256,41.5>
<0.70,256,4.2>
<0.90,-256,26.0>
<0.10,256,4.9>
<1.50,256,4.5>
<1.70,-256,4.8>
<1.20,256,5.1>
<0.90,256,14.6>
<0.50,256,6.2>
<1.80,256,10.0>
<1.00,256,24.2>
<0.70,256,15.5>
<0.70,256,8.4>
<1.30,228,18.8>
<1.40,256,22.6>
<0.50,256,9.8>
r_ewma
s_ewma

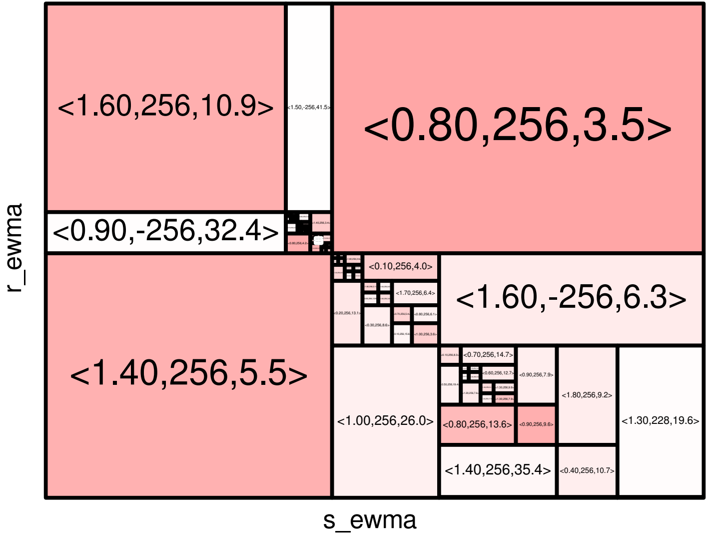
RemyCC


<1.90,256,9.1>
<0.70,256,4.2>
<0.90,-256,26.0>
<1.20,256,5.1>
<1.70,-256,4.8>
<0.10,256,4.9>
<1.50,256,4.5>
<0.90,256,14.6>
<0.50,256,6.2>
<1.80,256,10.0>
<1.00,256,24.2>
<0.70,256,15.5>
<1.30,228,18.8>
<1.40,256,22.6>
<0.50,256,9.8>
r_ewma
s_ewma

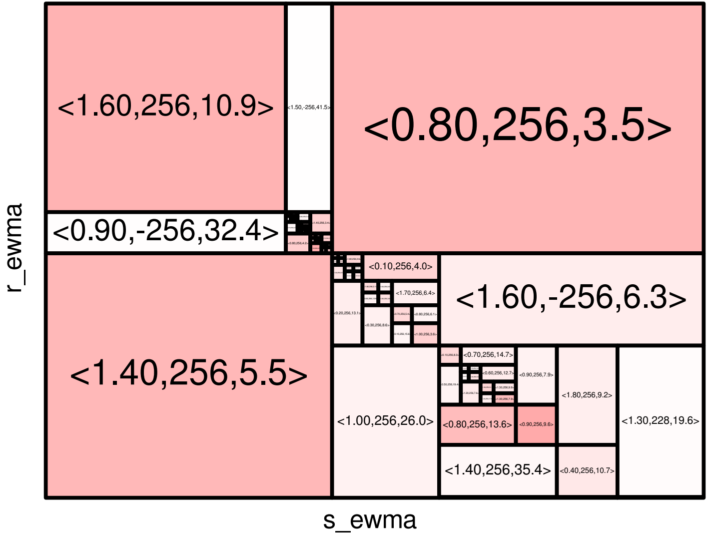
RemyCC


<1.60,256,10.9>
<1.50,-256,41.5>
<0.80,256,3.5>
<0.90,-256,32.4>
<0.10,256,4.0>
<1.70,256,6.4>
<1.60,-256,6.3>
<1.40,256,5.5>
<0.70,256,14.7>
<0.90,256,7.9>
<1.80,256,9.2>
<1.00,256,26.0>
<1.30,228,19.6>
<0.80,256,13.6>
<1.40,256,35.4>
<0.40,256,10.7>
r_ewma
s_ewma

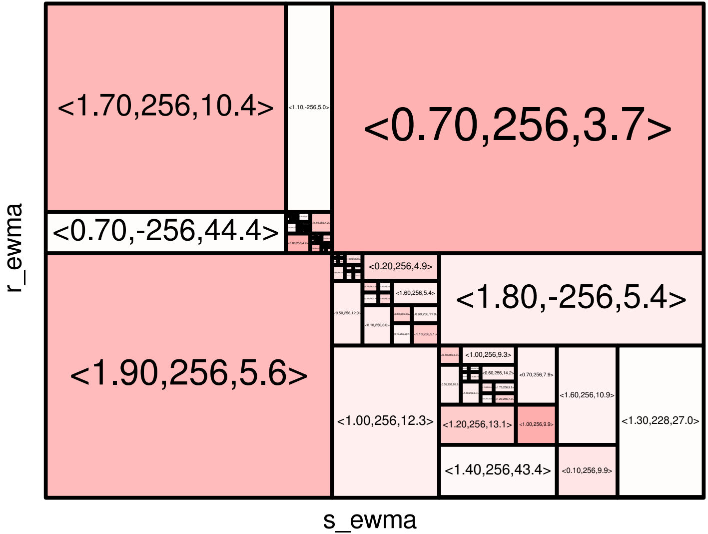
RemyCC


<1.60,256,10.9>
<1.50,-256,41.5>
<0.80,256,3.5>
<0.90,-256,32.4>
<0.10,256,4.0>
<1.70,256,6.4>
<1.60,-256,6.3>
<0.20,256,13.1>
<0.30,256,8.6>
<0.80,256,6.1>
<1.00,256,3.6>
<1.40,256,5.5>
<0.70,256,14.7>
<0.60,256,12.7>
<0.90,256,7.9>
<1.80,256,9.2>
<1.00,256,26.0>
<0.80,256,13.6>
<0.90,256,9.6>
<1.30,228,19.6>
<1.40,256,35.4>
<0.40,256,10.7>
r_ewma
s_ewma

RemyCC

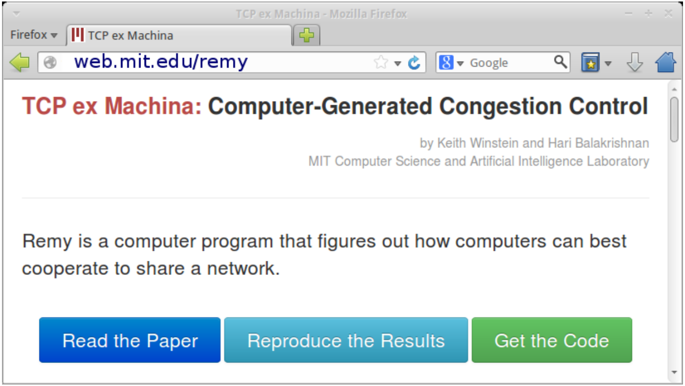

<1.60,256,10.9>
<1.50,-256,41.5>
<0.80,256,3.5>
<0.90,-256,32.4>
<0.10,256,4.0>
<1.70,256,6.4>
<1.60,-256,6.3>
<1.40,256,5.5>
<0.70,256,14.7>
<0.60,256,12.7>
<0.90,256,7.9>
<1.80,256,9.2>
<1.00,256,26.0>
<0.80,256,13.6>
<0.90,256,9.6>
<1.30,228,19.6>
<1.40,256,35.4>
<0.40,256,10.7>
r_ewma
s_ewma

RemyCC

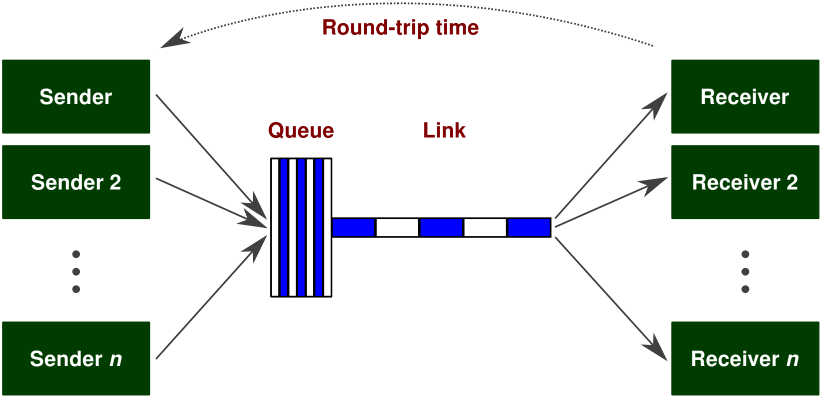

<1.70,256,10.4>
<1.10,-256,5.0>
<0.70,256,3.7>
<0.70,-256,44.4>
<0.20,256,4.9>
<1.80,-256,5.4>
<1.60,256,5.4>
<1.90,256,5.6>
<1.00,256,9.3>
<0.70,256,7.9>
<1.60,256,10.9>
<1.00,256,12.3>
<1.20,256,13.1>
<1.30,228,27.0>
<1.40,256,43.4>
<0.10,256,9.9>
r_ewma
s_ewma

Evaluation in ns-2

- End-to-end comparators: NewReno, Cubic, Compound, Vegas
- In-net comparators: Cubic-over-sfqCoDel, XCP
- Simulation setup published for replication



Scenario 1: fixed-rate network, homogenous senders



Scenario 1: details

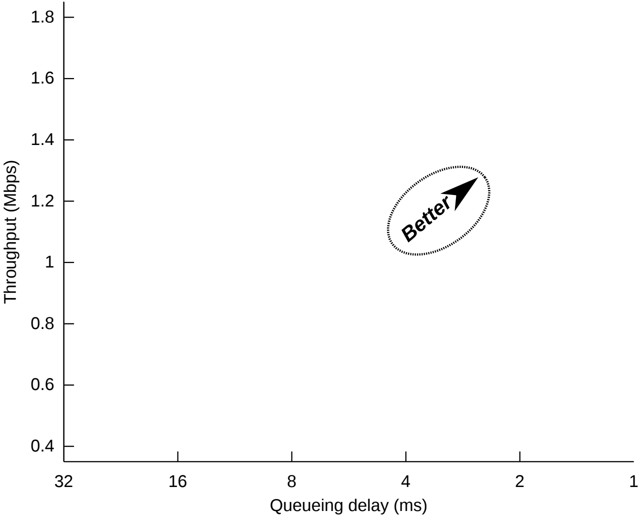
Quantity	Simulation parameter	Remy assumptions
Link speed	15 Mbps	Uniform(10, 20) Mbps
RTT	150 ms	Uniform(100, 200) ms
n	8	Uniform(1, 16)
"On" process	$\exp[\mu = 100]$ **kB**	$\exp[\mu = 5]$ **s**
"Off" process	$\exp\left[\mu = \frac{1}{2}\right]$ s	$\exp[\mu = \mathbf{5}]$ s

Remy objective:

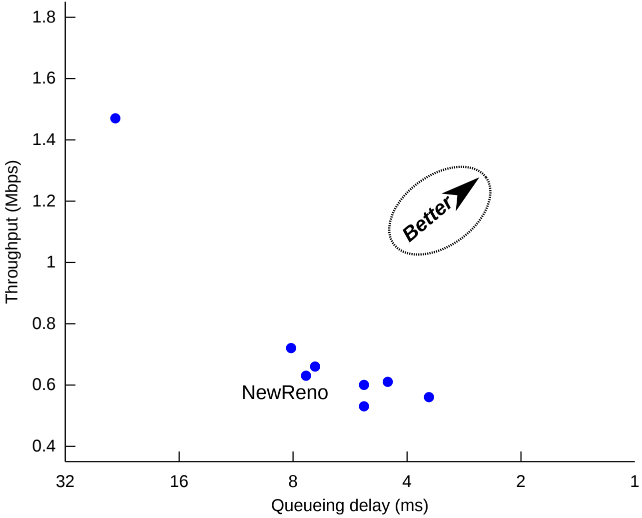
$$\sum_i \log \left[\frac{\text{throughput}_i}{(\text{delay}_i)^{\delta}} \right]$$

$$\delta \in \left\{ \frac{1}{10}, 1, 10 \right\}$$

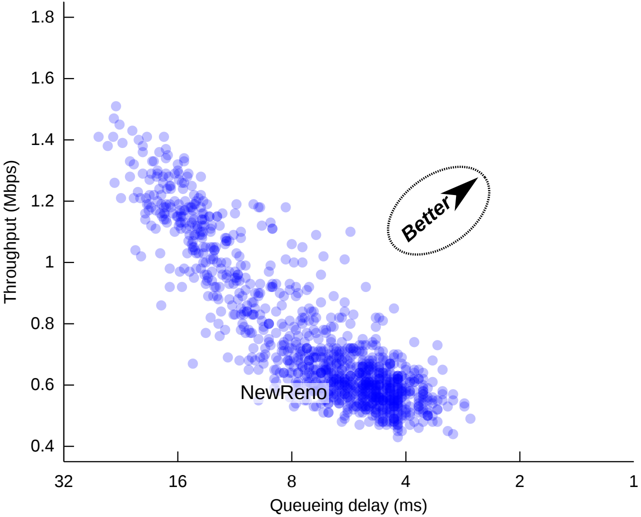
Scenario 1: throughput-delay plot



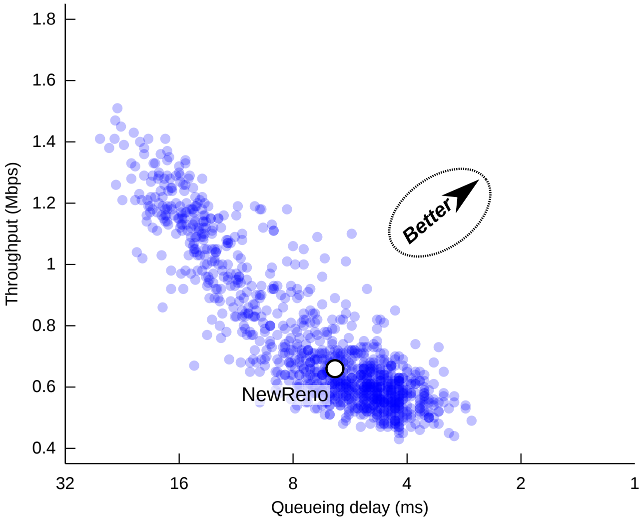
Scenario 1: throughput-delay plot



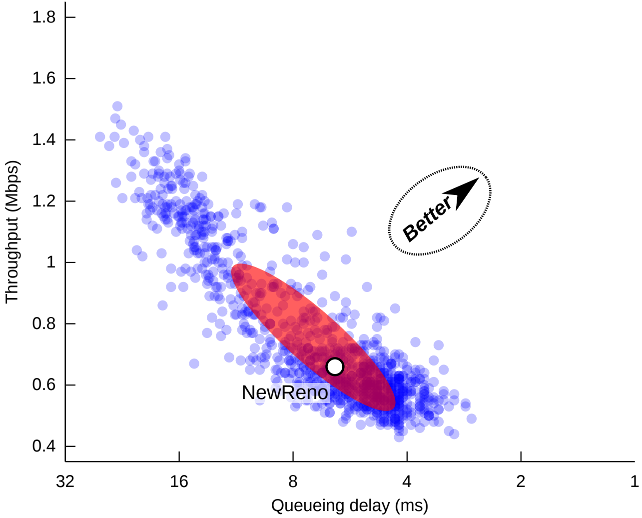
Scenario 1: throughput-delay plot



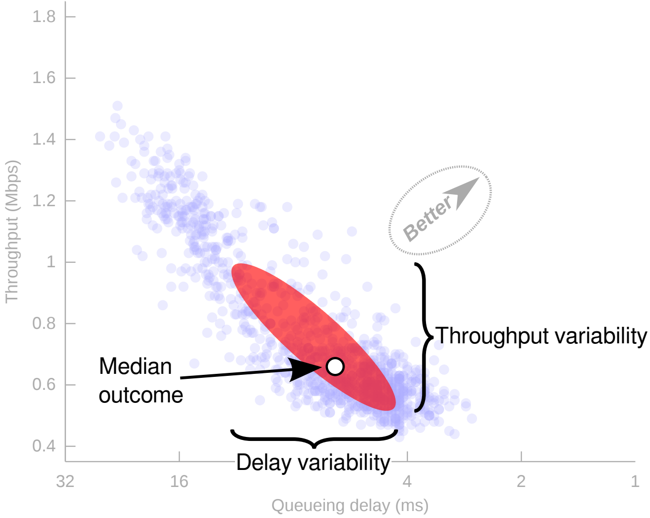
Scenario 1: throughput-delay plot



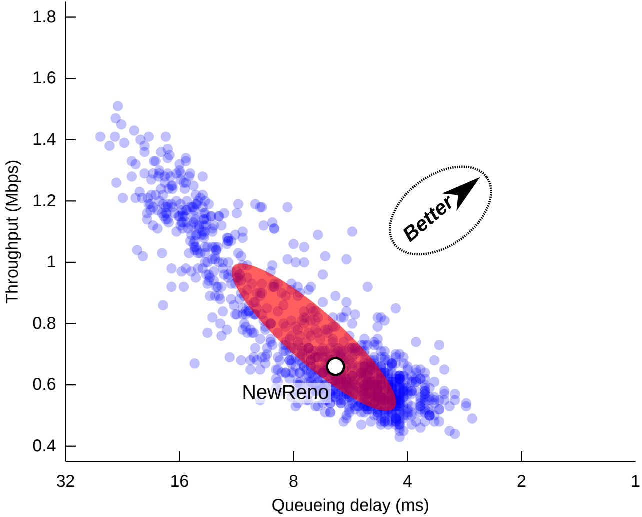
Scenario 1: throughput-delay plot



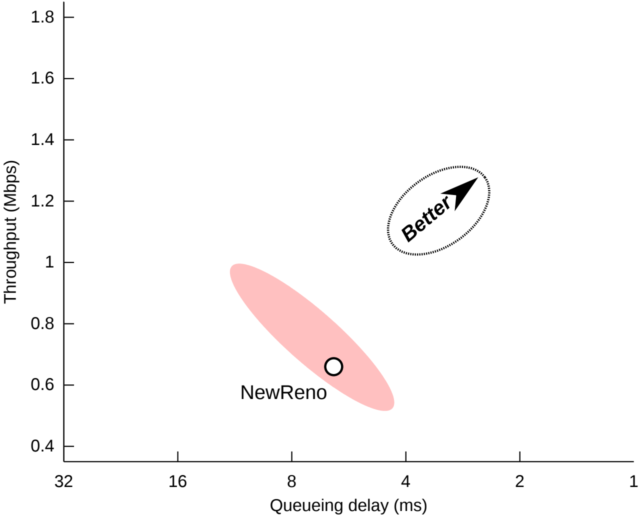
Scenario 1: throughput-delay plot



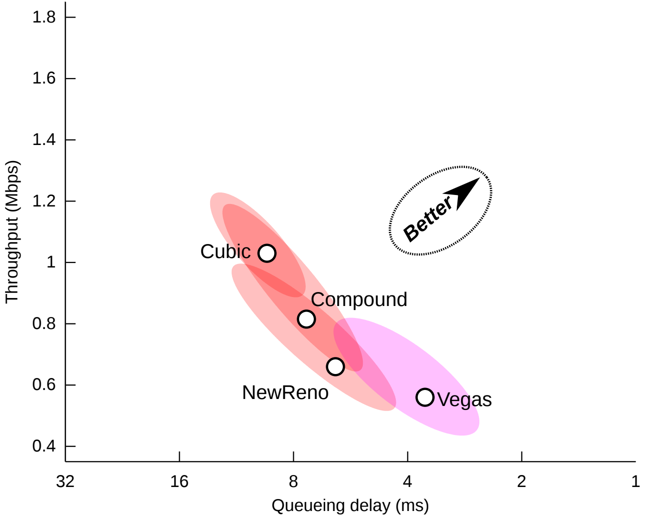
Scenario 1: throughput-delay plot



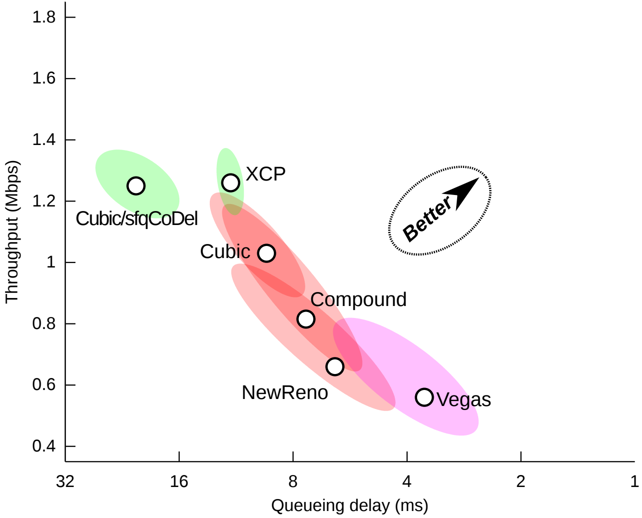
Scenario 1: throughput-delay plot



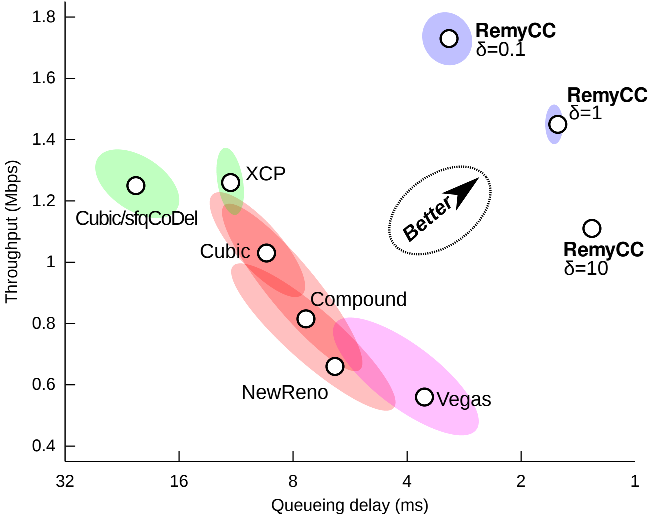
Scenario 1: throughput-delay plot



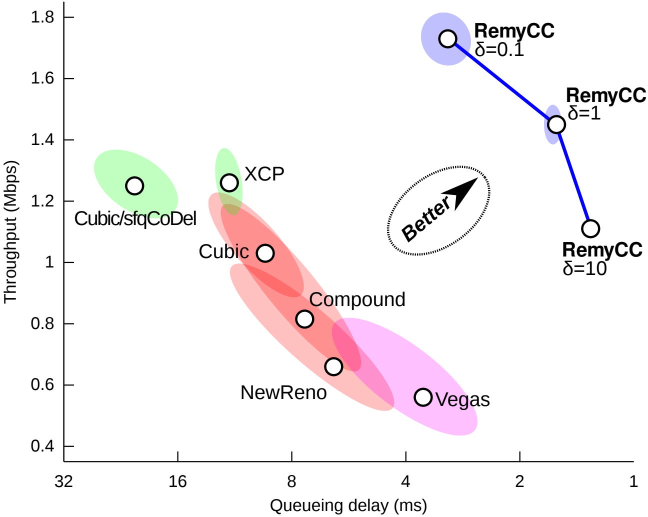
Scenario 1: throughput-delay plot



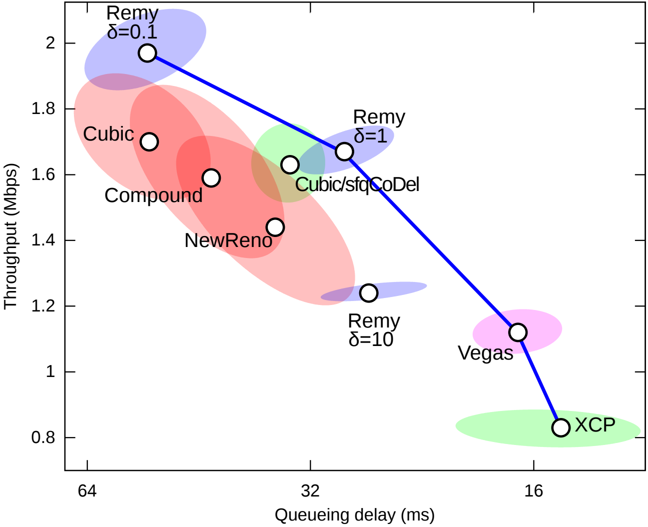
Scenario 1: throughput-delay plot



Scenario 1: throughput-delay plot



Scenario 2: Verizon LTE, $n = 8$

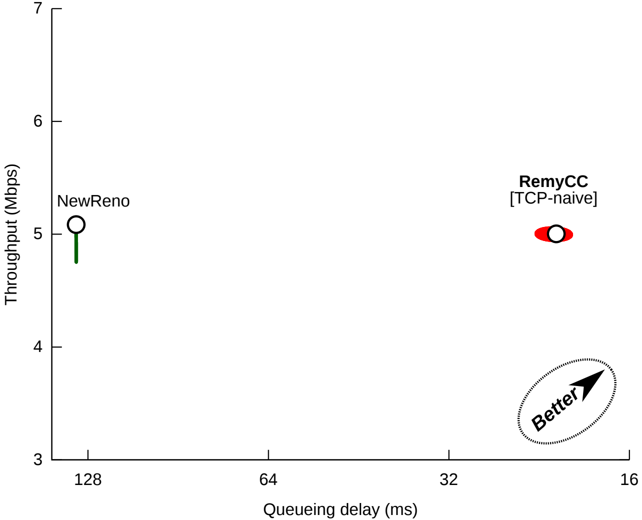


Remy as an instrument to study network science

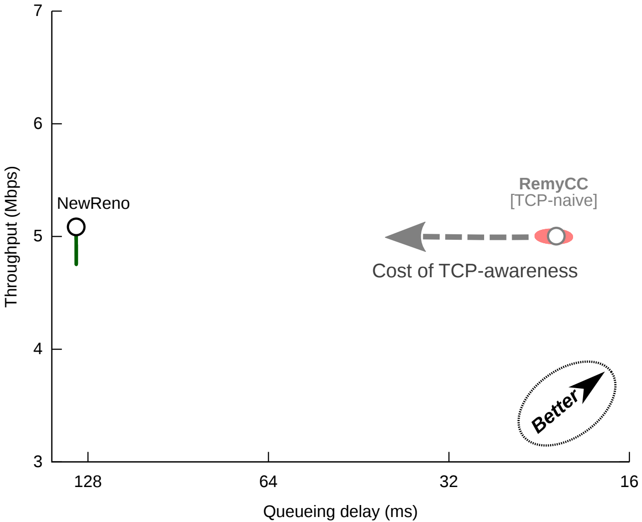
From the perspective of an endpoint, **what does it help to know** *about the network?*

How difficult is it to learn a good protocol, given an **imperfect** *model of the network?*

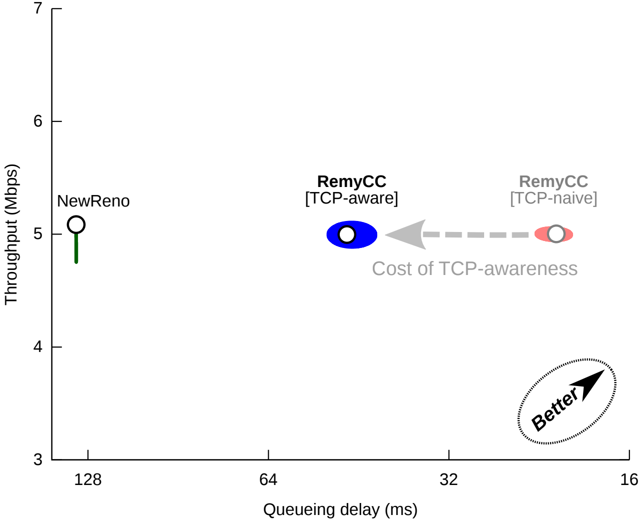
RemyCC competing against itself



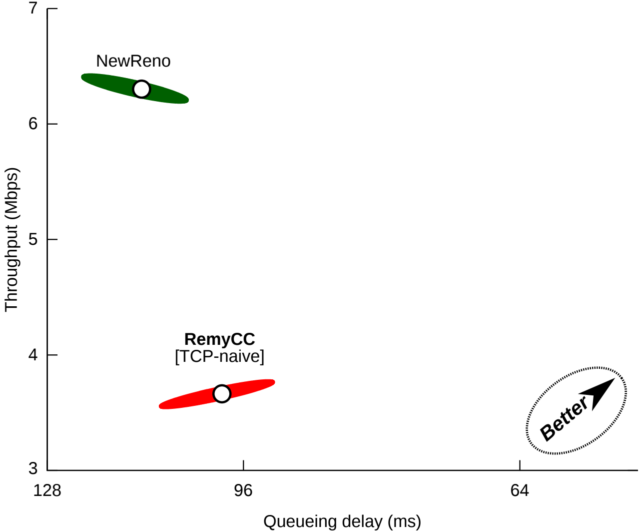
RemyCC competing against itself



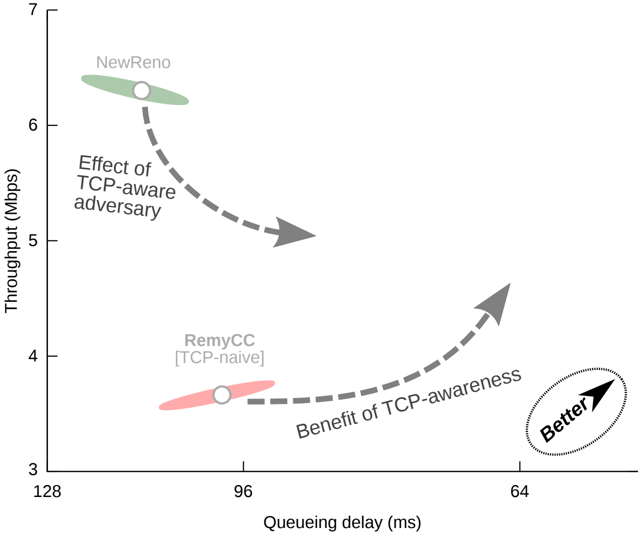
RemyCC competing against itself



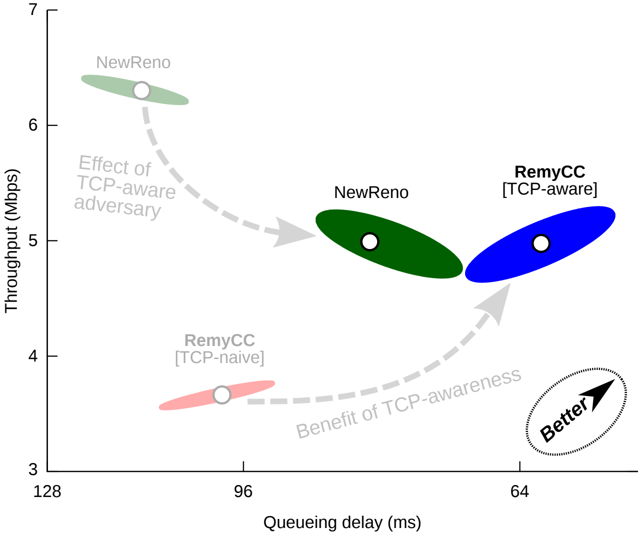
RemyCC competing against TCP NewReno



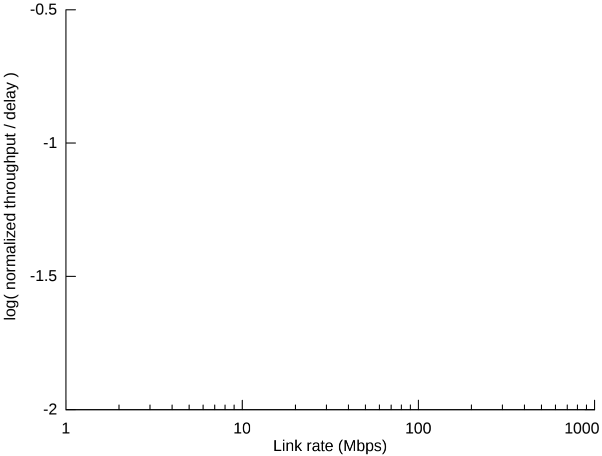
RemyCC competing against TCP NewReno



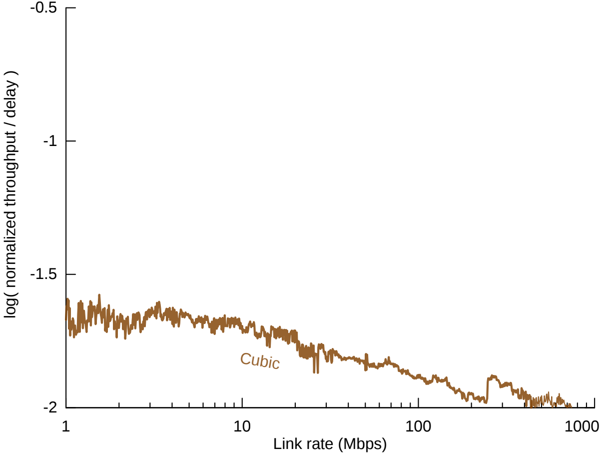
RemyCC competing against TCP NewReno



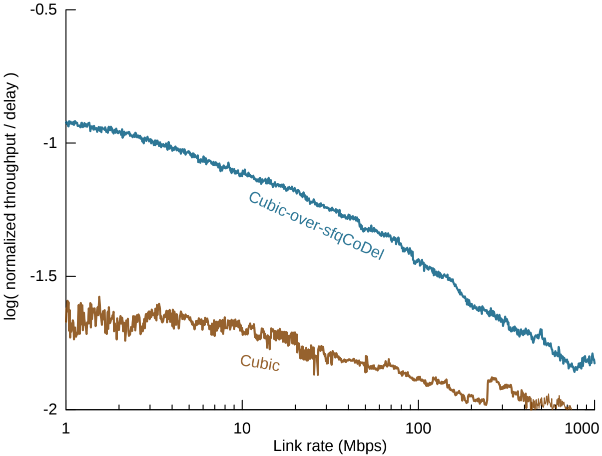
The cost of generality



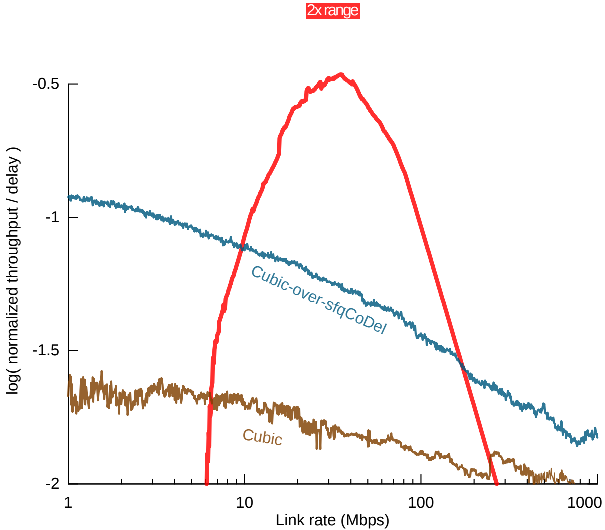
The cost of generality



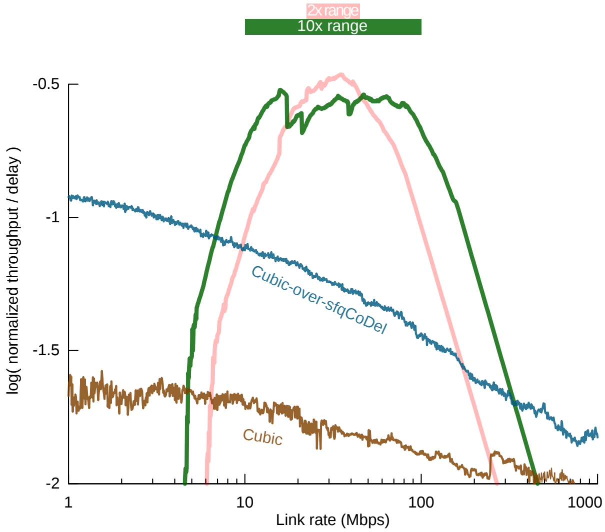
The cost of generality



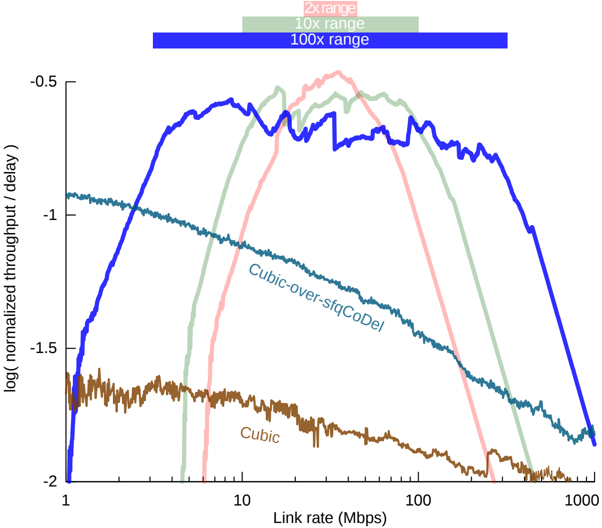
The cost of generality



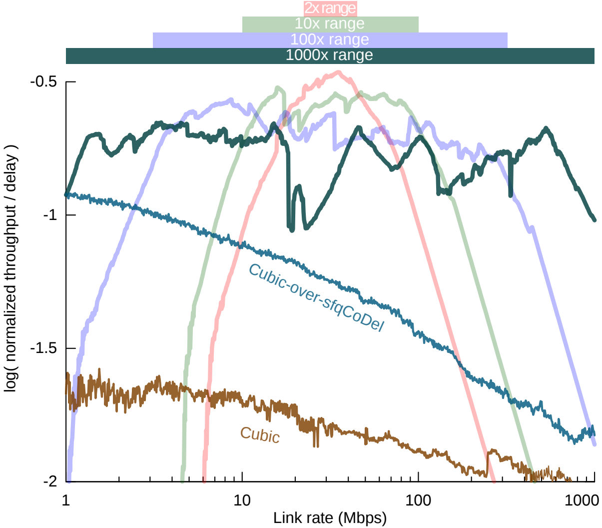
The cost of generality



The cost of generality



The cost of generality



Systems ex Machina

- Explicit design considerations → **freedom to make changes**
- "If this system is the answer, what's the question?"

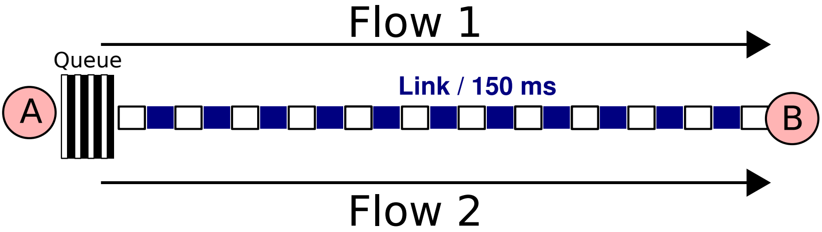
Sprout 2–4× the throughput and 7–9× less delay than Skype, etc.
Remy computer-generated protocol design

keithw@mit.edu

http://mit.edu/keithw

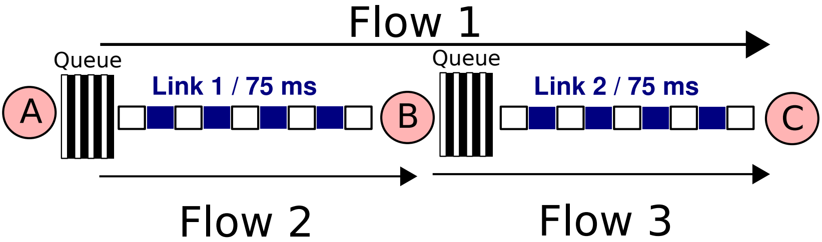
When the model is wrong about the topology

One bottleneck

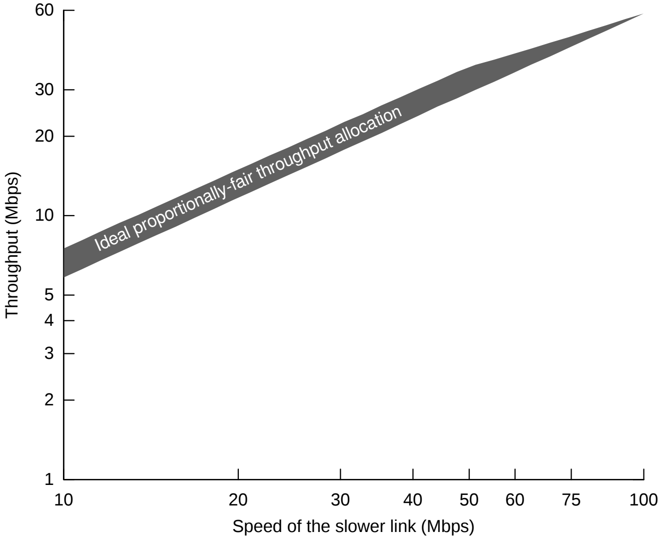


When the model is wrong about the topology

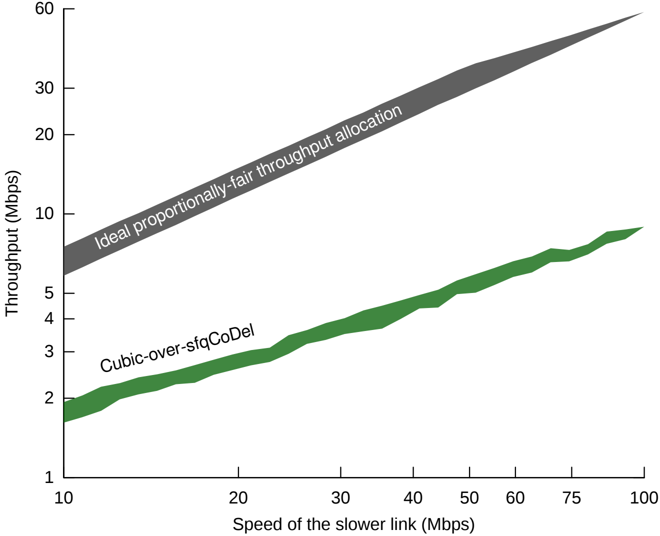
Two bottlenecks



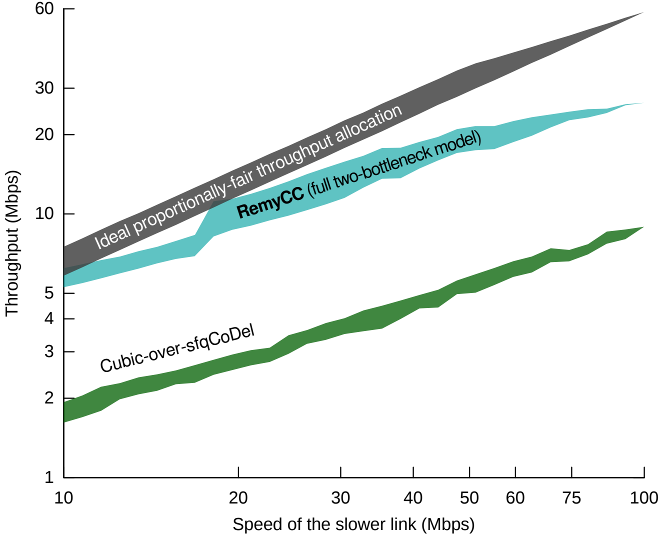
When the model is wrong about the topology



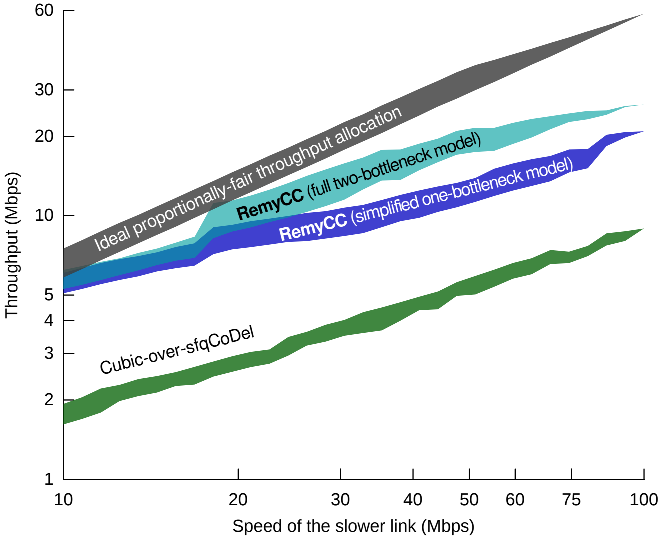
When the model is wrong about the topology

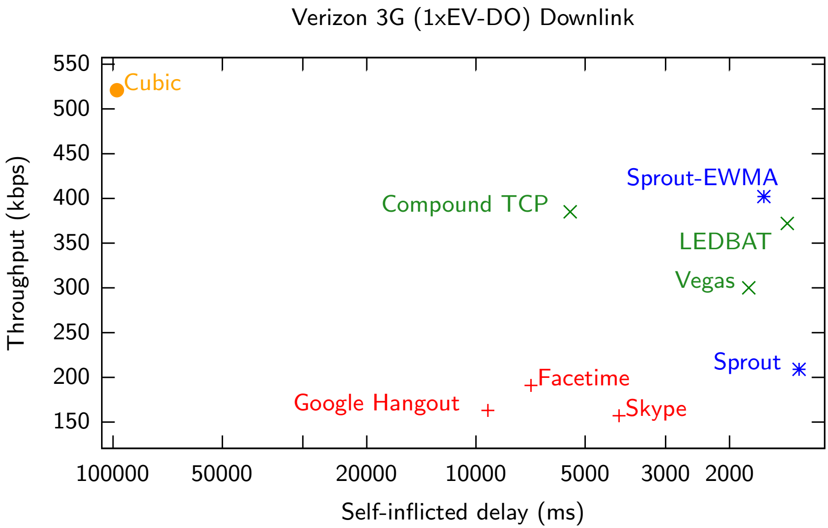


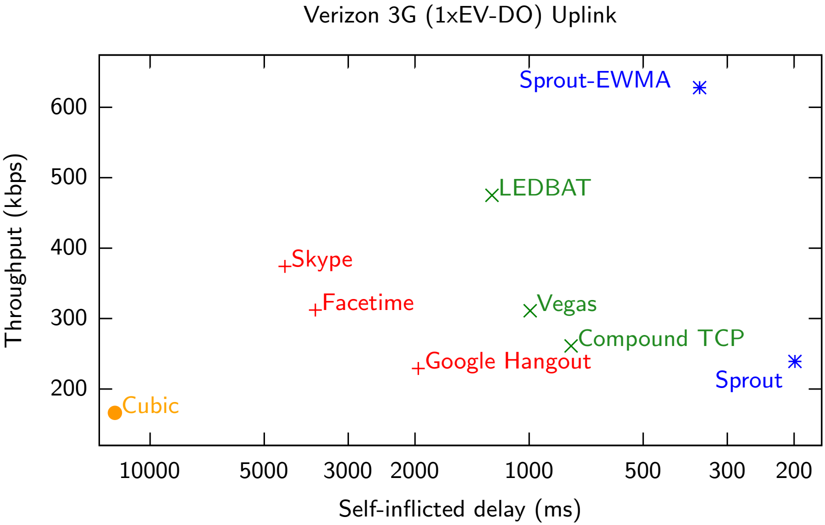
When the model is wrong about the topology

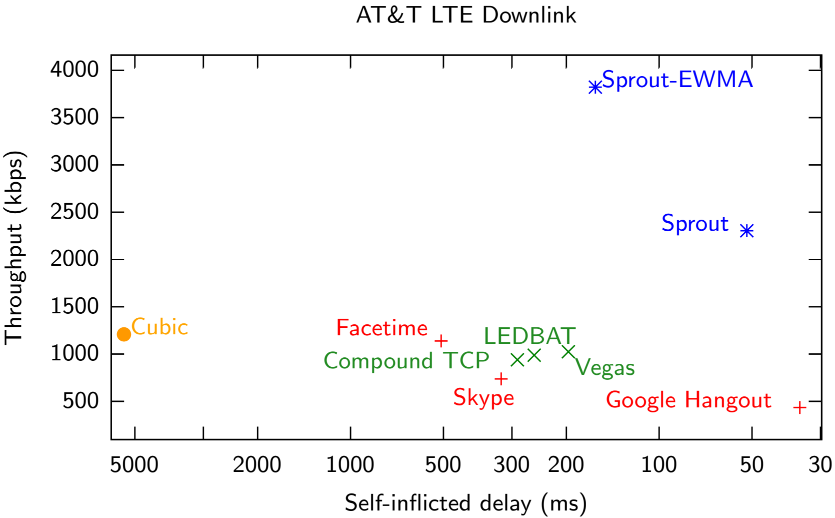


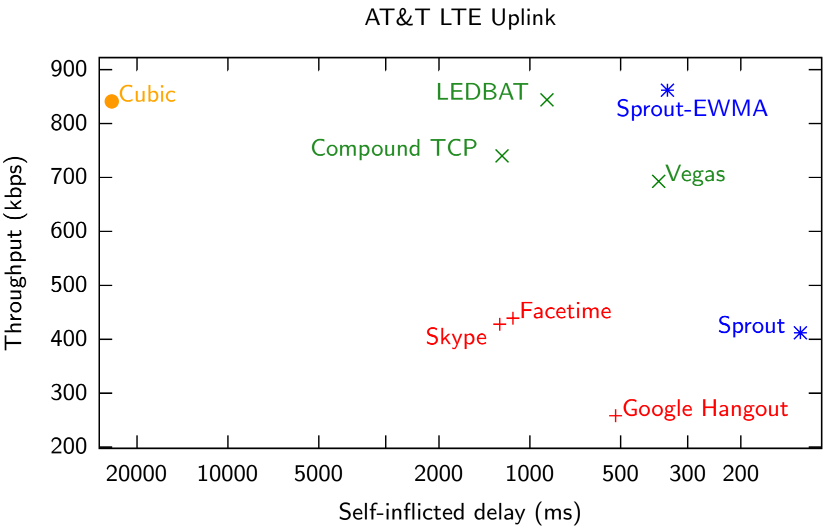
When the model is wrong about the topology

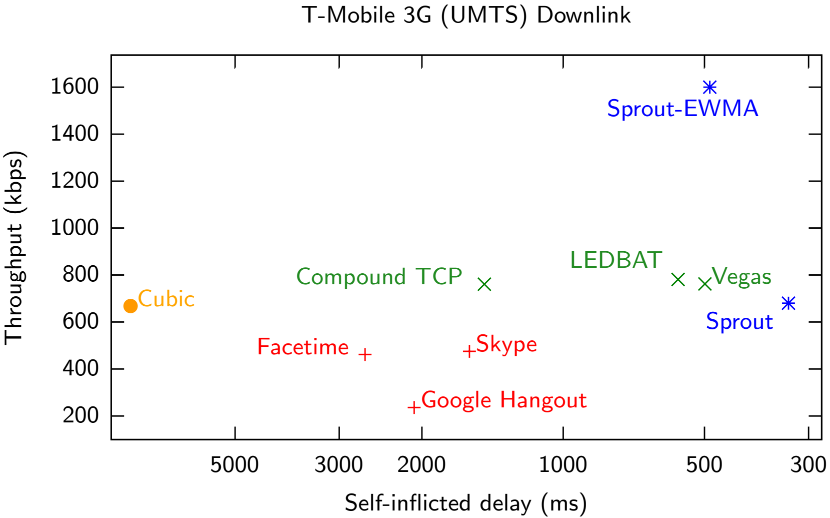


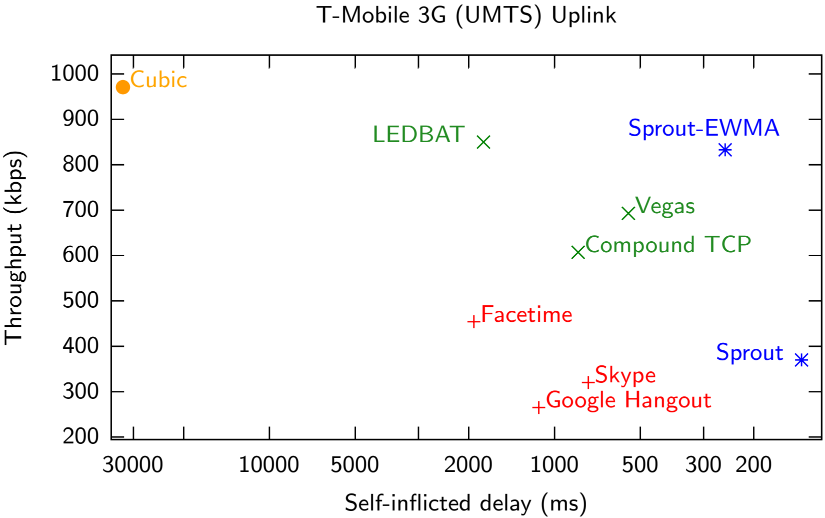

Verizon 3G (1xEV-DO) Downlink
Throughput (kbps)
550
500
450
400
350
300
250
200
150
Self-inflicted delay (ms)
100000
50000
20000
10000
5000
3000
2000
Cubic
Compound TCP
Sprout-EWMA
LEDBAT
Vegas
Sprout
Facetime
Google Hangout
Skype


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


AT&T LTE Downlink
Throughput (kbps)
4000
3500
3000
2500
2000
1500
1000
500
Self-inflicted delay (ms)
5000
2000
1000
500
300
200
100
50
30
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
20000
10000
5000
2000
1000
500
300
200
Self-inflicted delay (ms)
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
Sprout-EWMA
LEDBAT
Compound TCP
Vegas
Cubic
Sprout
Facetime
Skype
Google Hangout
5000
3000
2000
1000
500
300
Self-inflicted delay (ms)


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