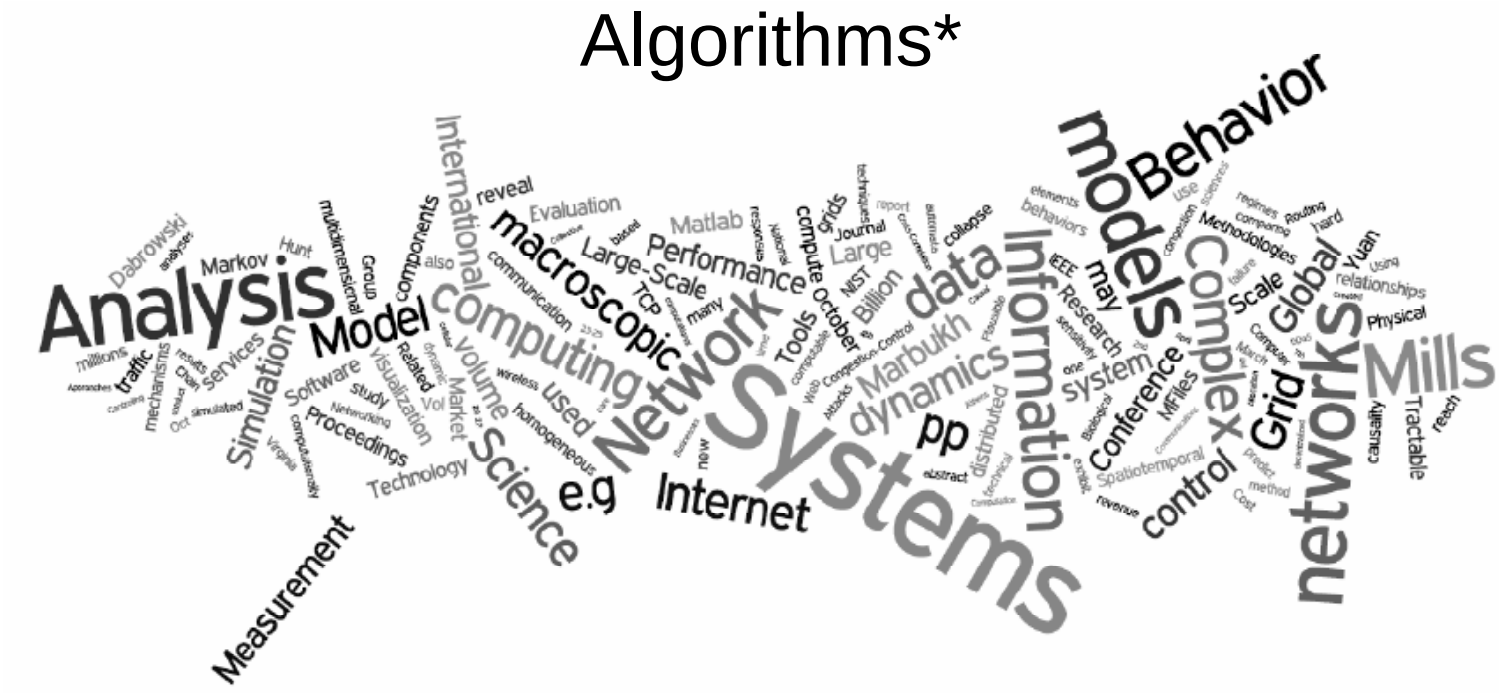
Study of Proposed Internet Congestion Control Algorithms*



Kevin L. Mills, NIST
(joint work with D. Y. Cho, J. Filliben, D. Genin and E. Schwartz)
March 24, 2010

Information Technology Laboratory

*performed under the NIST Complex Systems program: http://www.nist.gov/itl/cxs/index.cfm

image generated with http://www.wordle.net/

More information @ http://www.antd.nist.gov/emergent_behavior.shtml

Measurement Science for Complex Information Systems

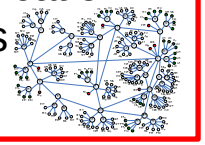
Understand & Predict Behavior

in large distributed information systems

by using mathematical & statistical techniques

applied by scientists to study physical systems.

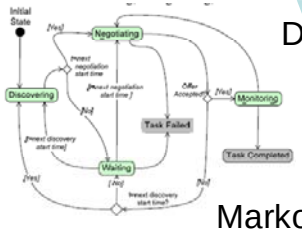
Communication Networks



Computational Clouds

Computational Grids

Dabrowski
Hunt



Markov models
Perturbation analysis

$$p_{rj}^{(new)} = \begin{cases} p_{rj}^{(old)} + m_{prim} & j = c^{\uparrow} \\ \left[p_{rj}^{(old)} - w \cdot m_{prim}\right]^{+} & j = c^{\downarrow} \\ \left[p_{rj}^{(old)} - (1-w) \cdot m_{prim} \dfrac{p_{rj}^{(old)}}{\sum_{k \neq c^{\uparrow}, c^{\downarrow}} p_{rk}}\right]^{+} & j \neq c^{\uparrow}, c^{\downarrow} \end{cases} \quad (6)$$

Genin
Marbukh

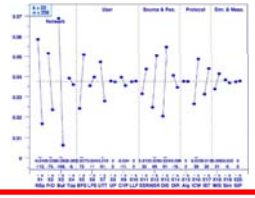
Differential equations
Fluid flow simulators

$$\frac{dW^N}{dt}(t) = \frac{N}{T} - \frac{1}{2}\sum_{i=1}^{N} W_i^N(t) P_i^N(t)$$

$$\frac{dw}{dt}(t) = \frac{1}{T} - \frac{1}{2} w(t) p(t),$$

$$\frac{dw}{dt}(t) = \frac{1}{T} - \frac{1}{2}\frac{w(t) p_q(w(t-T)) w(t-T)}{T}$$

Filliben
Mills



Reduced scale DE simulators
OFF experiment designs
Cluster analysis
Principal components analysis
Correlation analysis
Multidimensional visualizations

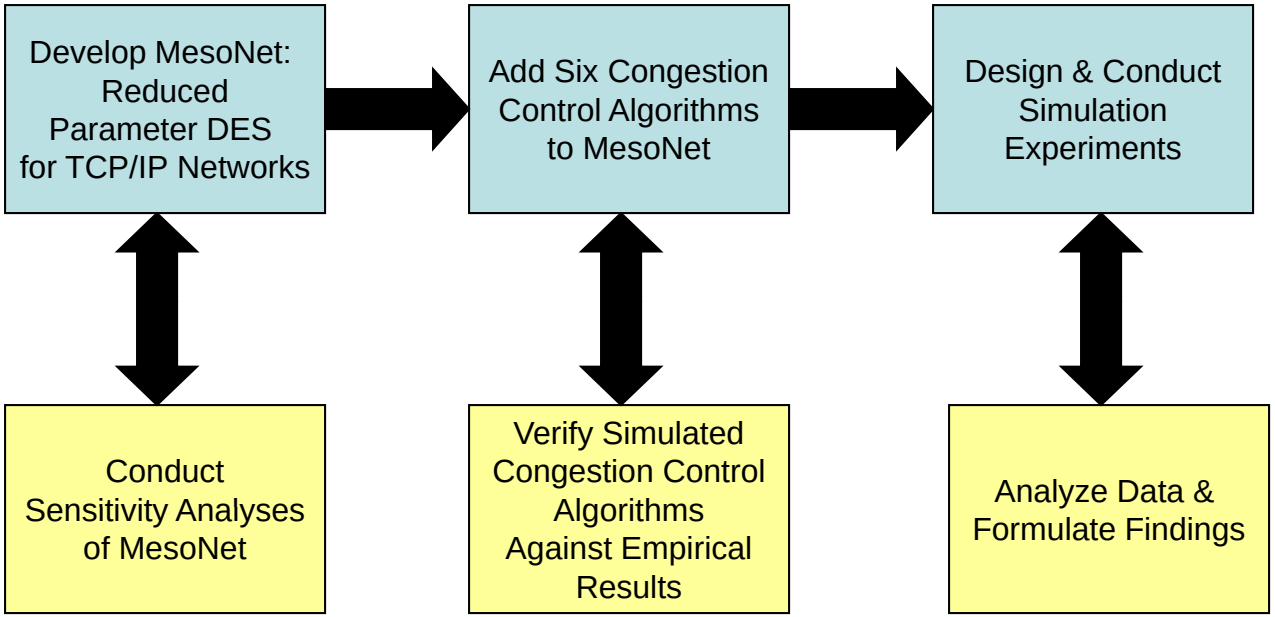
Other contributors: DY Cho, Edward Schwartz, Peter Mell, Jian Yuan, Zanxin Xu, Cedric Houard, Brittany Devine

Outline

- Technical approach → (slides 4-7)
- Overview of experiments → (slides 8-10)
- Some technical flavor → (slides 11-26)
  - Selected analysis techniques interspersed among → (slides 14-16 & 20-26)
  - Selected experiment details → (slides 11-13 & 17-19)
- Findings → (slides 27-34)
  - Utility and safety → (slides 27-32)
  - Characteristics of individual congestion control algorithms → (slides 33-34)
- Recommendations → (slides 35)
- Open discussion

Technical Approach

Our study is fairly comprehensive: large, fast topologies and wide-range of conditions



Algorithms Studied

BIC TCP
CTCP
FAST
FAST-AT
HS TCP
HTCP
Scalable TCP

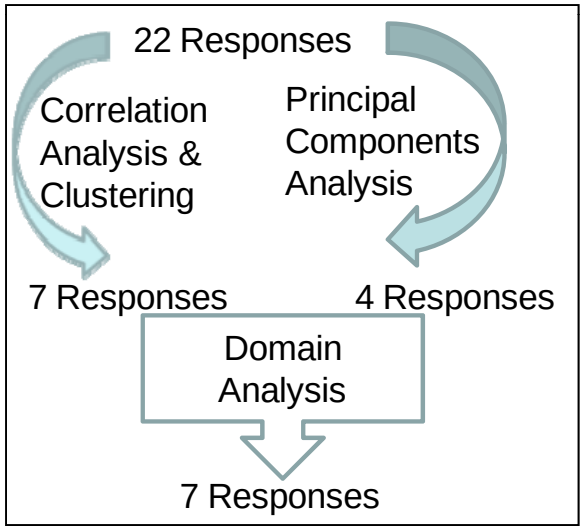
Topologies with up to 278,000 sources; backbone speeds up to 384 Gbps; loss rates between 10^{-9} and 50%; simulated durations of 25 – 60 mins; traffic including Web browsing and software and movie downloads; long-lived flows; temporary spatiotemporal congestion and recovery; algorithms homogeneous and mixes of alternates together with standard TCP; buffer sizes include *RTT* x *C* and *RTT* x *C*/sqrt(*n*); propagation delays from 6 to 200 ms; initial slow start threshold from 43 to $2^{31}/2$

Simulating large, fast networks across many conditions and congestion control algorithms requires reduction – model responses & parameters

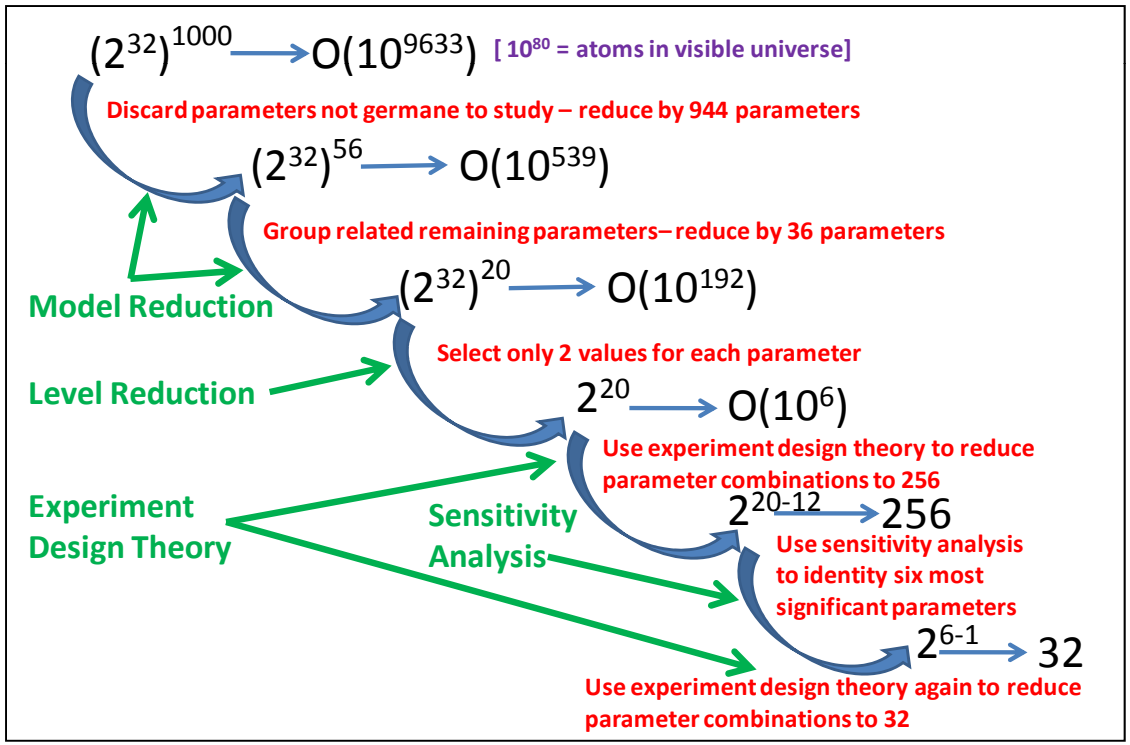
$$y_1, \ldots, y_z = f(x_{1|[1,\ldots,\ell]} \ldots, x_{p|[1,\ldots,\ell]})$$

Response State-Space — Stimulus State-Space

Multidimensional Response Reduction



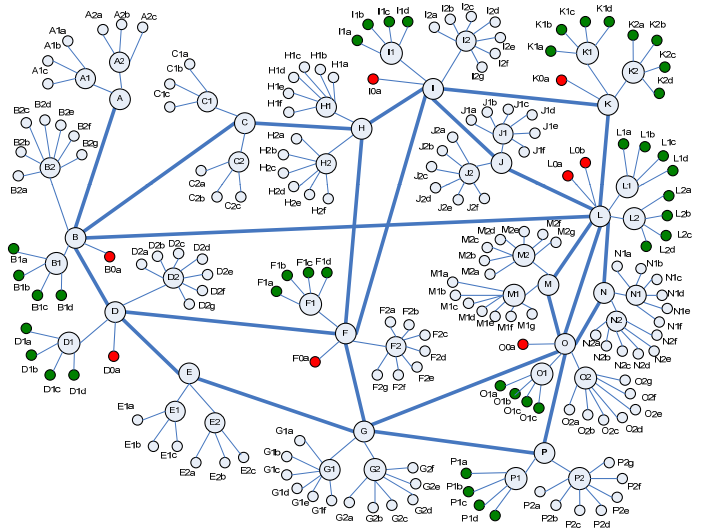
Parameter Reduction



MesoNet – a 20-parameter TCP/IP Network Model

Category	Identifier	Name
Network Configuration	X1	Topology
	X2	Propagation Delay
	X3	Network Speed
	X4	Buffer Provisioning
Sources & Receivers	X5	Number of Sources & Receivers
	X6	Distribution of Sources
	X7	Distribution of Receivers
	X8	Source & Receiver Interface Speeds
User Behavior	X9	Think Time
	X10	Patience
	X11	Web Object Size for Browsing
	X12	Proportion & Sizes of Larger File Downloads
	X13	Selected Spatiotemporal Congestion
	X14	Long-lived Flows
Protocols	X15	Congestion Control Algorithms
	X16	Initial Congestion Window Size
	X17	Initial Slow Start Threshold
Simulation & Measurement Control	X18	Measurement Interval Size
	X19	Simulation Duration
	X20	Startup Pattern

Parameter	Value	Speed Relationships		Speed Scaling with X3	
		Router Class	Speed	X3 = 800	X3 = 1600
s1	X3				
s2	4	Backbone	*s1* x *BBspeedup*	1600	3200
s3	10	PoP	*s1*/ *s2*	400	800
BBspeedup	2	**N**-Class	*s1*/ *s2*/ *s3*	40	80
Bfast	2	**F**-Class	*s1*/ *s2*/ *s3* x *Bfast*	80	160
Bdirect	10	**D**-Class	*s1*/ *s2*/ *s3* x *Bdirect*	400	800



Class	#routers	srcs/router	#srcs	%srcs	rcvrs/router	#rcvrs	%rcvrs
N-class	122	90	10,980	31.6	960	117,120	95.3
F-class	40	540	21,600	62.2	120	4,800	3.9
D-class	8	270	2,160	6.2	120	960	0.8

Flow class	%flows
NN-flows	30.1
FN-flows	60.5
FF-flows	2.4
DN-flows	6.1
DF-flows	0.74
DD-flows	0.05

Adopt 2-Level Orthogonal Fractional Factorial Designs

Sample 2^{9-4} design

Factor->	x1	x2	x3	x4	x5	x6	x7	x8	x9
Condition	--	--	--	--	--	--	--	--	--
1	-1	-1	-1	-1	-1	+1	+1	+1	+1
2	+1	-1	-1	-1	-1	+1	-1	-1	-1
3	-1	+1	-1	-1	-1	-1	+1	-1	-1
4	+1	+1	-1	-1	-1	-1	-1	+1	+1
5	-1	-1	+1	-1	-1	-1	-1	+1	-1
6	+1	-1	+1	-1	-1	-1	+1	-1	+1
7	-1	+1	+1	-1	-1	+1	-1	-1	+1
8	+1	+1	+1	-1	-1	+1	+1	+1	-1
9	-1	-1	-1	+1	-1	-1	-1	-1	+1
10	+1	-1	-1	+1	-1	-1	+1	+1	-1
11	-1	+1	-1	+1	-1	+1	-1	+1	-1
12	+1	+1	-1	+1	-1	+1	+1	-1	+1
13	-1	-1	+1	+1	-1	+1	+1	-1	-1
14	+1	-1	+1	+1	-1	+1	-1	+1	+1
15	-1	+1	+1	+1	-1	-1	+1	+1	+1
16	+1	+1	+1	+1	-1	-1	-1	-1	-1
17	-1	-1	-1	-1	+1	-1	-1	-1	-1
18	+1	-1	-1	-1	+1	-1	+1	+1	+1
19	-1	+1	-1	-1	+1	+1	-1	+1	+1
20	+1	+1	-1	-1	+1	+1	+1	-1	-1
21	-1	-1	+1	-1	+1	+1	+1	-1	+1
22	+1	-1	+1	-1	+1	+1	-1	+1	-1
23	-1	+1	+1	-1	+1	-1	+1	+1	-1
24	+1	+1	+1	-1	+1	-1	-1	-1	+1
25	-1	-1	-1	+1	+1	+1	+1	1	-1
26	+1	-1	-1	+1	+1	+1	-1	-1	+1
27	-1	+1	-1	+1	+1	-1	+1	-1	+1
28	+1	+1	-1	+1	+1	-1	-1	+1	-1
29	-1	-1	+1	+1	+1	-1	-1	+1	+1
30	+1	-1	+1	+1	+1	-1	+1	-1	-1
31	-1	+1	+1	+1	+1	+1	-1	-1	-1
32	+1	+1	+1	+1	+1	+1	+1	+1	+1

Sample experiment using 9 parameters

1. Selected appropriate $n = 2^{p-k}$ design template
2. Select two values for each parameter
3. Substitute parameter levels in template
4. Fix remaining (11) model parameters

Probes combinations with balance and orthogonality

All 32: $\frac{16}{-}\ \frac{16}{+}\ X_i$ Balance

All $\binom{32}{2}$: X_j $\begin{matrix} + & \boxed{\begin{matrix} 8 & 8 \\ 8 & 8 \end{matrix}} & + \\ - & & - \end{matrix}$ X_i Orthogonality

Resolution IV design – no main effects are confounded with two-term interactions

2-Level Designs Support Convenient Data Analysis Techniques

Summary of Our Experiments Comparing Congestion Control Algorithms

How do the algorithms react to and recover from spatiotemporal congestion?

Experiment #1a – Large (up to 278,000 sources), Fast (up to 192 Gbps backbone) network; Web browsing; 25 minutes simulated; 3 Time Periods; large ($2^{32}/2$) initial slow-start threshold (*sst*); all sources use same alternate congestion control algorithm

Experiment #1b – Same as #1a except smaller (up to 27,800 sources), slower (up to 28.8 Gbps backbone) network; low (100) initial *sst*

How do the algorithms improve flow throughputs and affect TCP flows?

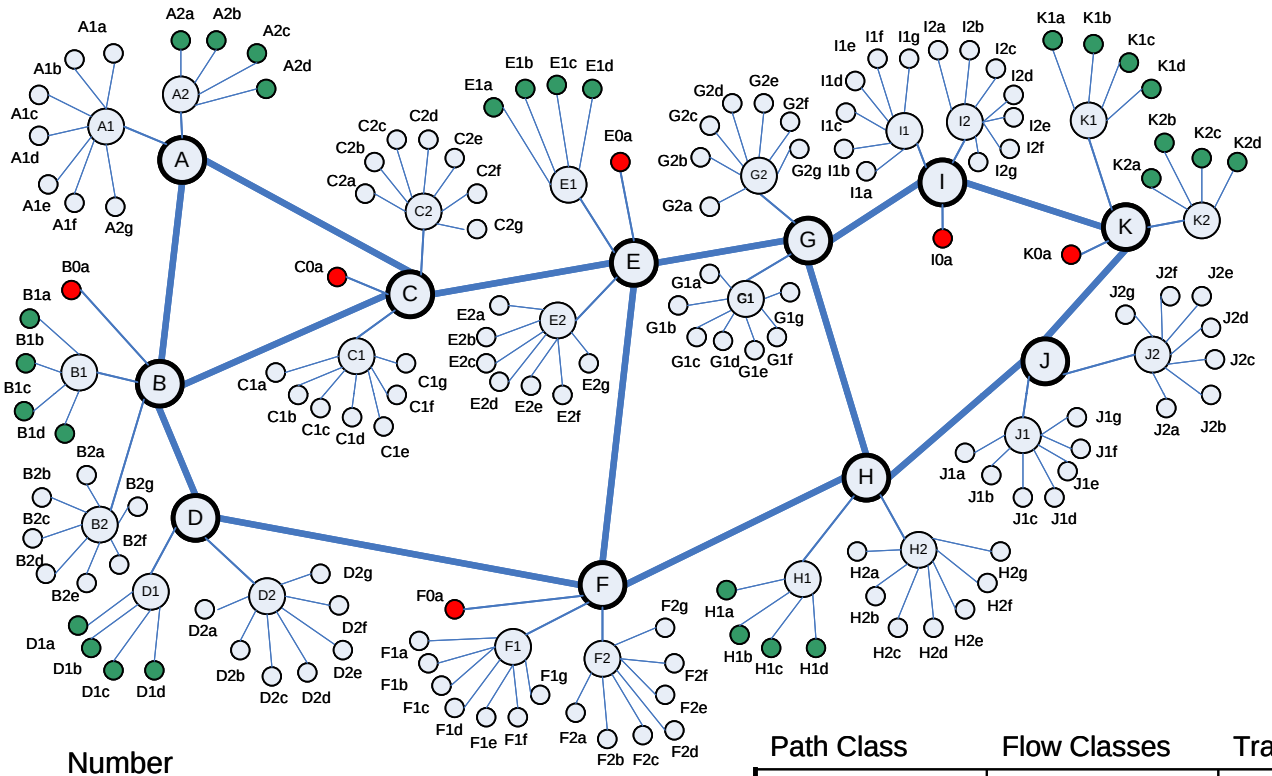
Experiment #2a – Small (up to 26,085 sources), Slow (up to 38.4 Gbps backbone) Network; Web browsing plus downloading software and movies; 60 minutes simulated; large ($2^{32}/2$) initial *sst* ; some sources use standard TCP and some use alternate congestion control algorithm

Experiment #2b – Same as #2a except low (100) initial *sst*

Experiment #2c – Same as #2a except larger (up to 261,792 sources), faster (up to 384 Gbps backbone) network

Overview of Experiments

x1 - All experiments used the same three-tier topology based on the Abilene backbone

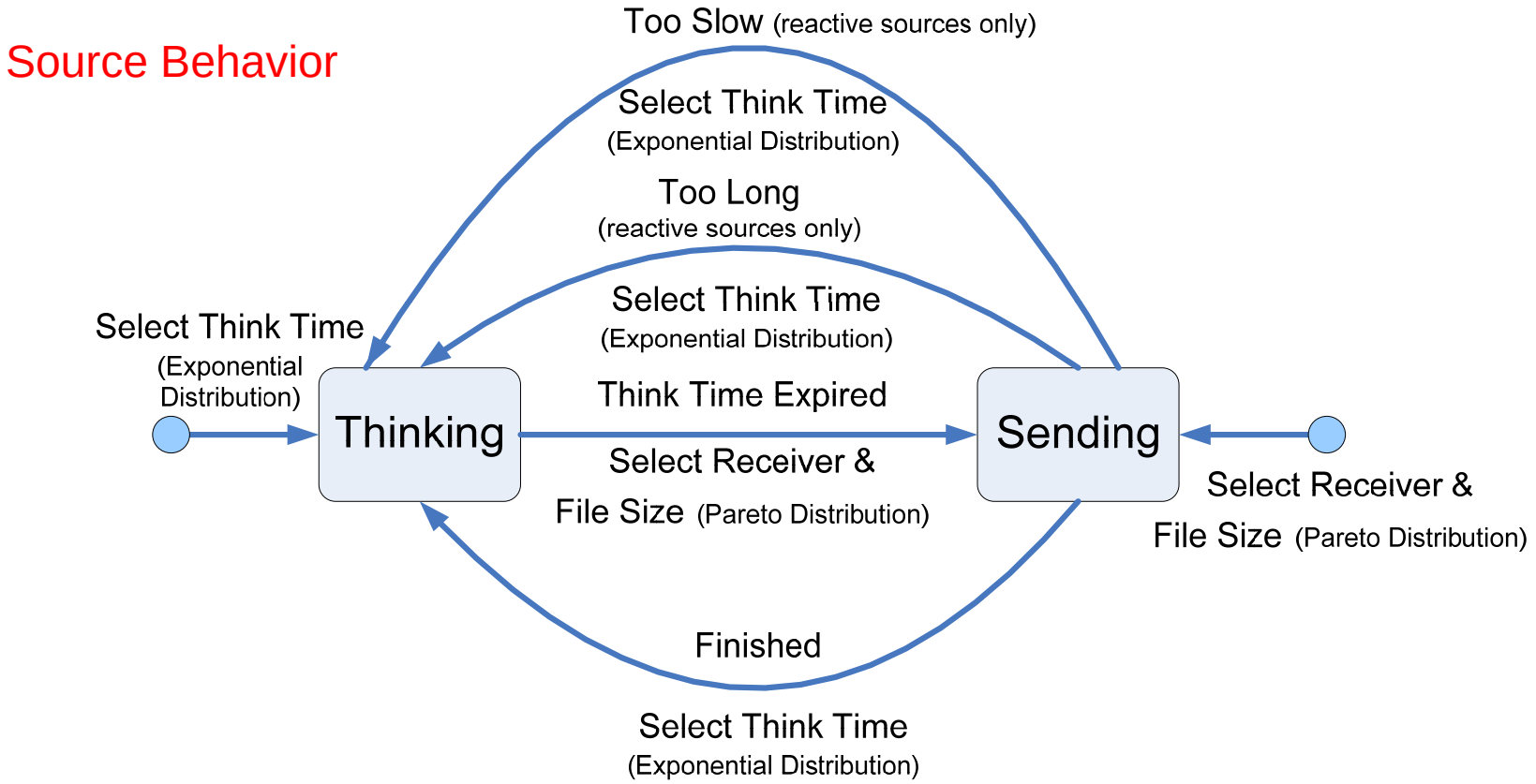


Router Type	Number
Backbone	11
PoP	22
D-class Access	6
F-class Access	28
N-class Access	105

All flows transit the backbone

Path Class	Flow Classes	Traffic Class
Very Fast (VF)	**DD**-flows	Web-centric
Fast (F)	**DF**-flows	
	FF-flows	
Typical (T)	**DN**-flows	
	FN-flows	
	NN-flows	Peer-2-Peer

Tier 4 is Sources and Receivers



For simplicity, the state diagram omits a flow connection phase that occurs prior to sending, and also the potential for connection failure after which a source reenters the thinking state

Design for Experiment #1a

Algorithms Compared

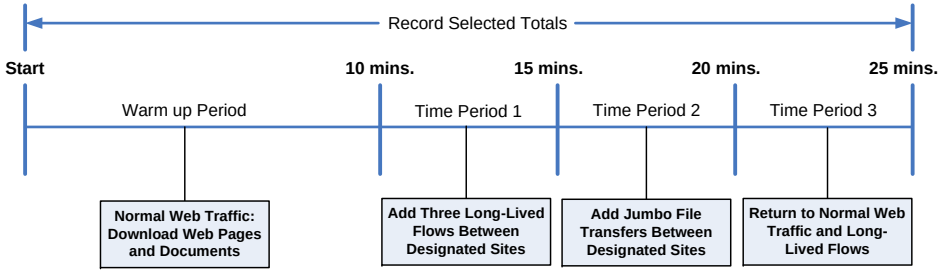
Identifier	Label	Name of Congestion-Avoidance Algorithm
1	BIC	Binary Increase Congestion Control
2	CTCP	Compound Transmission Control Protocol
3	FAST	Fast Active-Queue Management Scalable Transmission Control Protocol
4	HSTCP	High-Speed Transmission Control Protocol
5	HTCP	Hamilton Transmission Control Protocol
6	Scalable	Scalable Transmission Control Protocol
7	TCP	Transmission Control Protocol (Reno)

Parameters Varied (OFF 2^{6-1})

Parameter	Definition	PLUS (+1) Value	Minus (-1) Value
x2	Propagation Delay Multiplier	2	1
x3	Network Speed	8000 p/ms	4000 p/ms
x4	Buffer Sizing Algorithm	*RTT* x *C*	*RTT* x *C* /sqr(*n*)
x6	Source Distribution	Uniform(.33/.33/.33)	Skewed(.1/.6/.3)
x9	Avg. Think Time	5 s	2.5 s
x11	Avg. Size for Web Object	100 packets	50 packets

baseSources = 1000

Spatiotemporal Scenario



Parameters Fixed

x5	Number Sources	2 (*baseSources* = 1000)
x7	Receiver Dist.	0.6/0.2/0.2
x8	Prob. *Hfast*	0.4
x10	User Patience	infinite
x12	Large Files	*Fp* = 0.1;*Fx* = 10
x13	ST Congestion	*Jon*=0.6;*Joff*=0.9;*Jx*=100
x14	Long Flows	3
x15	Algorithm	Appropriate One
x16	Initial *cwnd*	2 packets
x17	Initial *sst*	$2^{31}/2$ packets
x18	MI	200 ms
x19	Duration	25 mins.
x20	Startup Pattern	25%;8%;17%;50%

Design for Experiment #1a

Domain View of Experiment #1a

Router Speeds

Router	PLUS (+1)	Minus (-1)
Backbone	192 Gbps	96 Gbps
POP	24 Gbps	12 Gbps
Normal Access	2.4 Gbps	1.2 Gbps
Fast Access	4.8 Gbps	2.4 Gbps
Directly Connected Access	24 Gbps	12 Gbps

Propagation Delays

	Min	Avg	Max
PLUS (+1)	12	81	200
Minus (-1)	6	41	100

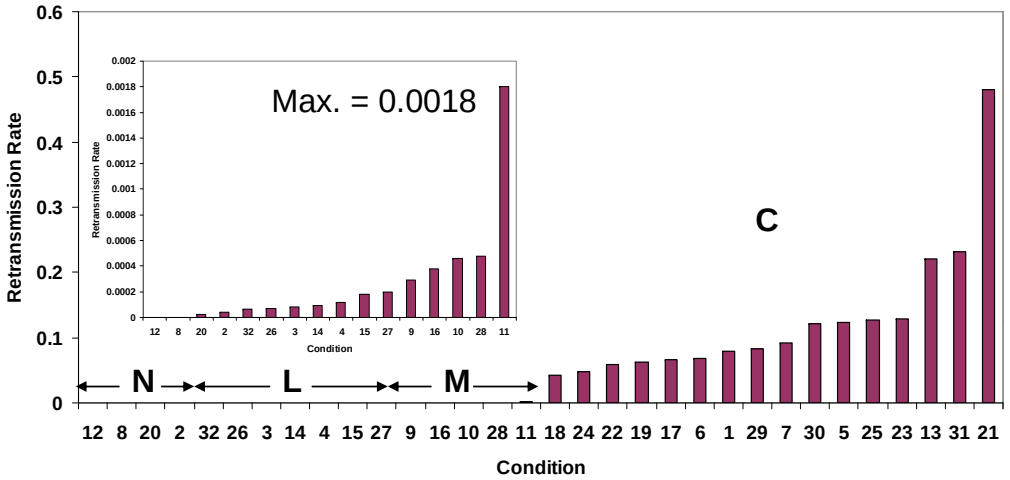
Number of Sources

PLUS (+1)	Minus (-1)
278,000	174,600

Router Buffer Sizes

	PLUS (+1)			Minus (-1)		
Router	Min	Avg	Max	Min	Avg	Max
Backbone	325,528	732,437	1,302,110	1,153	2,606	4,654
POP	40,691	91,555	162,764	221	505	908
Access	6,470	14,557	25,879	91	207	369

Congestion Conditions



N = none, L = Low, M = Moderate, C = Congested

224 Total Runs (32 conditions x 7 algorithms)

Statistic	Flows Completed	Data Packets Sent
Avg. Per Condition	74,033,116	6,912,373,746
Min. Per Condition	40,966,013	3,146,870,571
Max. Per Condition	154,914,953	11,917,420,154
Total All Runs	16,583,418,069	1,548,371,719,084

Selected Response Measurements for Experiment #1

Macroscopic Behavior

Response	Definition
y42	Average number of connecting flows
y1	Average number of active (i.e., connected) flows
y43	Average number of active flows in initial slow start
y44	Average number of active flows in normal congestion-control mode
y45	Average number of active flows in alternate congestion-control mode
y3	Average packets output per measurement interval
y5	Average flows completed per measurement interval
y6	Average retransmission rate
y7	Average smoothed round-trip time (SRTT)
y8	Average round-trip queuing delay
y2	Average congestion-window increases per active flow
y4	Average congestion window per active flow

Goodput on Flow Classes

Response	Definition
y9	Average goodput (pps) for DD flows
y13	Average goodput (pps) for DF flows
y21	Average goodput (pps) for FF flows
y17	Average goodput (pps) for DN flows
y25	Average goodput (pps) for FN flows
y29	Average goodput (pps) for NN flows

Goodput on Long-Lived Flows

Response	Definition
y33	Average goodput (pps) for the long-distance flow (L1)
y34	Average goodput (pps) for the medium-distance flow (L2)
y35	Average goodput (pps) for the short-distance flow (L3)

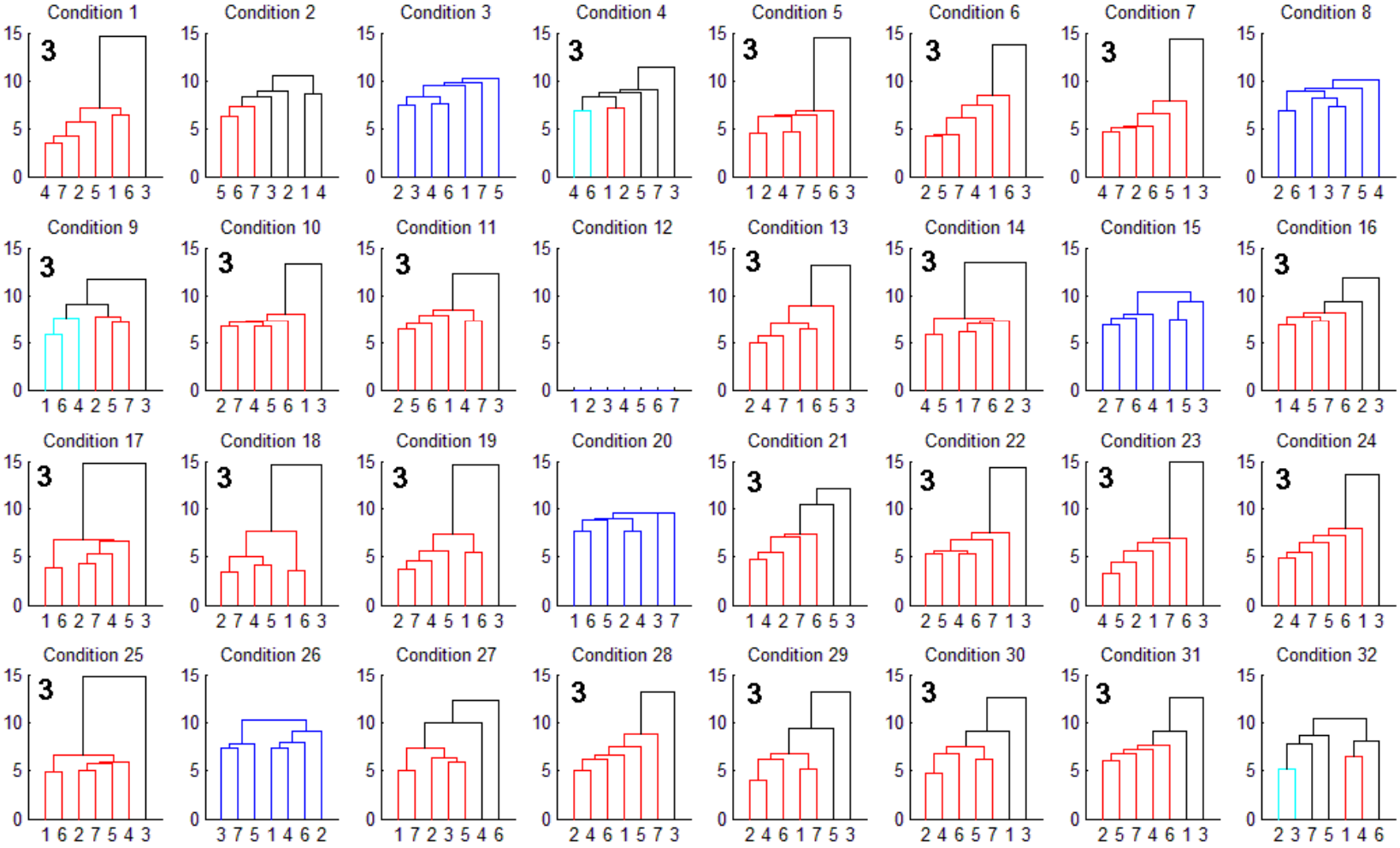
Aggregate Measures

Response	Definition
T.y1	Aggregate packets input
T.y2	Aggregate packets output
T.y3	Aggregate flows connected
T.y4	Aggregate flows completed
T.y5	Average SYNs sent per flow

Buffer Utilization on Selected Routers

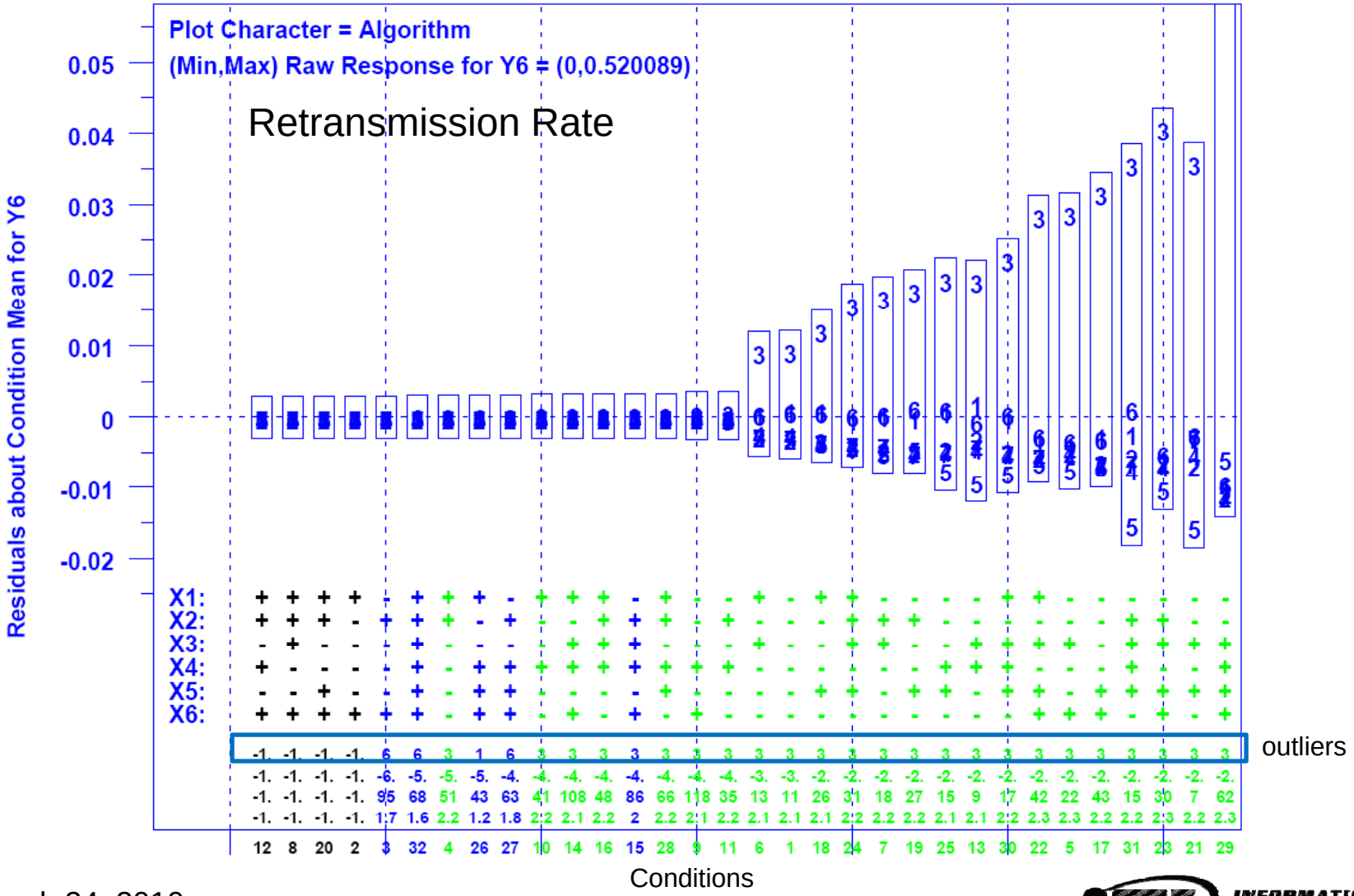
Response	Definition
y36	Average buffer saturation for router B0a
y37	Average buffer saturation for router C0a
y38	Average buffer saturation for router E0a
y39	Average buffer saturation for router F0a
y40	Average buffer saturation for router I0a
y41	Average buffer saturation for router K0a

Cluster Analyses Over All Macroscopic Responses



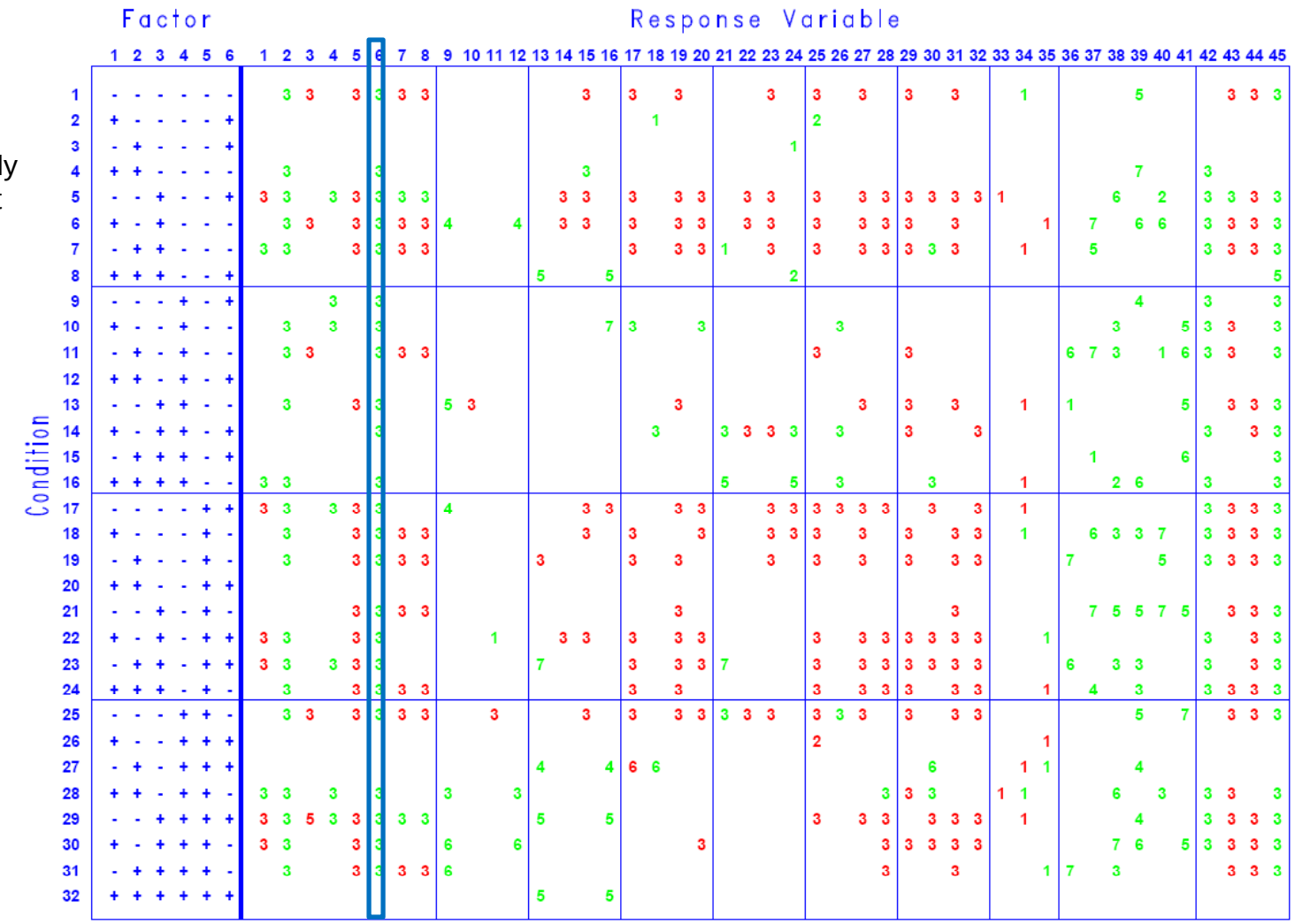
Algorithm 3 stands out

Each Response Subjected to Detailed Analysis



All Detailed Analyses Reflected in Condition-Response Summary

Statistically significant outliers



Design for Experiment #2a/b

Algorithms Compared

Identifier	Label	Name of Congestion-Avoidance Algorithm
1	BIC	Binary Increase Congestion Control
2	CTCP	Compound Transmission Control Protocol
3	FAST	Fast Active-Queue Management Scalable Transmission Control Protocol
4	FAST-AT	FAST with α-tuning Enabled
5	HSTCP	High-Speed Transmission Control Protocol
6	HTCP	Hamilton Transmission Control Protocol
7	Scalable	Scalable Transmission Control Protocol

Parameters Varied (OFF 2^{9-4})

Parameter	Definition	PLUS (+1) Value	Minus (-1) Value
x2	Propagation Delay Multiplier	2	1
x3	Network Speed	1600 p/ms	800 p/ms
x4	Buffers (*RTT* x *C* x *Qfactor*)	*Qfactor* = 1	*Qfactor* = 0.5
x5	Source Multiplier	3	2
x8	Probability of Fast Source	0.7	0.3
x9	Avg. Think Time	7.5 s	5 s
x11	Avg. Size for Web Object	150 packets	100 packets
x12	Probability of Large Files	*Fp*=0.04;*Sp*=0.004;*Mp*= 0.0004	*Fp*=0.02;*Sp*=0.002;*Mp*= 0.0002
x15	Probability of Alternate Alg.	0.7	0.3

baseSources=100 & File Size Multipliers: *Fx*=10;*Sx*=1000;*Mx*=10,000

24 Flow Groups

Identifier	Path Class	Interface Speed	File Type
1	VERY FAST	FAST	Movie
2	VERY FAST	NORMAL	Movie
3	FAST	FAST	Movie
4	FAST	NORMAL	Movie
5	TYPICAL	FAST	Movie
6	TYPICAL	NORMAL	Movie
7	VERY FAST	FAST	Service Pack
8	VERY FAST	NORMAL	Service Pack
9	FAST	FAST	Service Pack
10	FAST	NORMAL	Service Pack
11	TYPICAL	FAST	Service Pack
12	TYPICAL	NORMAL	Service Pack
13	VERY FAST	FAST	Document
14	VERY FAST	NORMAL	Document
15	FAST	FAST	Document
16	FAST	NORMAL	Document
17	TYPICAL	FAST	Document
18	TYPICAL	NORMAL	Document
19	VERY FAST	FAST	Web Object
20	VERY FAST	NORMAL	Web Object
21	FAST	FAST	Web Object
22	FAST	NORMAL	Web Object
23	TYPICAL	FAST	Web Object
24	TYPICAL	NORMAL	Web Object

Parameters Fixed

Parameter	Definition	Value
x6	Source Distribution	.1/.6/.4
x7	Receiver Distribution	.6/.2./.2
x10	User Patience	infinite
x13	Spatiotemporal Congestion	none
x14	Long-Lived Flows	none
x16	Initial *cwnd*	2 packets
x17	Initial *sst*	#2a ($2^{31}/2$) or #2b (100)
x18	Meas. Int. Size	200 ms
x19	Simulation Duration	60 mins
x20	Startup Pattern	25%; 8%;17%;50%

Design for Experiment #2a/b

Domain View of Experiment #2a/2b

Router Speeds

Router	PLUS (+1)	Minus (-1)
Backbone	38.4 Gbps	19.2 Gbps
POP	4.8 Gbps	2.4 Gbps
Normal Access	480 Mbps	240 Mbps
Fast Access	960 Mbps	720 Mbps
Directly Connected Access	4.8 Gbps	2.4 Gbps

Propagation Delays

	Min	Avg	Max
PLUS (+1)	12	81	200
Minus (-1)	6	41	100

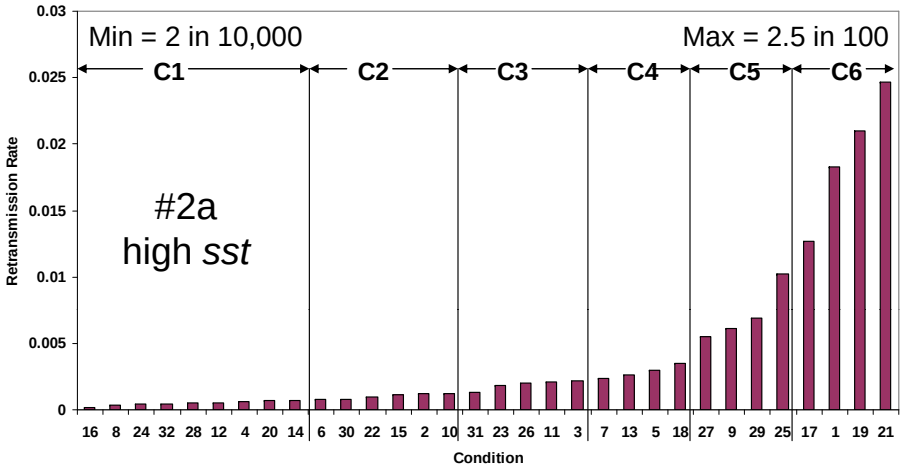
Number of Sources

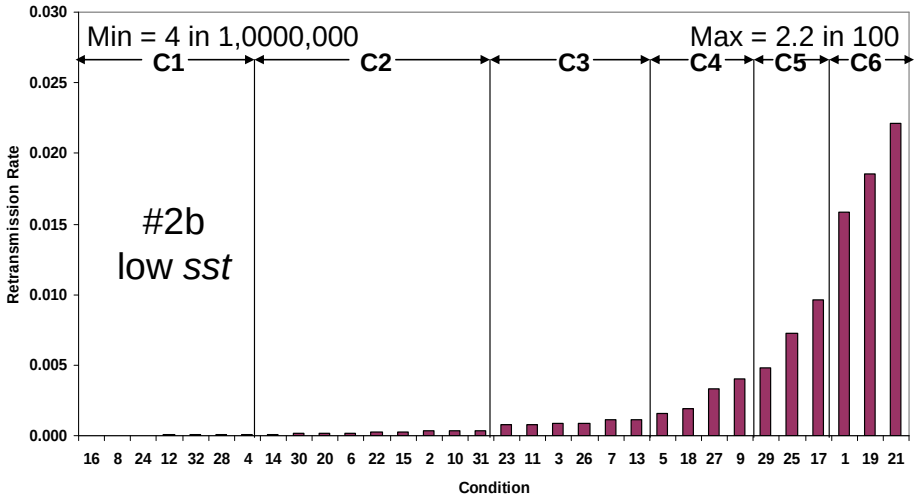
PLUS (+1)	Minus (-1)
26,085	17,355

Router Buffer Sizes

	x2 1.0			x2 0.5		
Router	Min	Avg	Max	Min	Avg	Max
Backbone	65,105	146,487.30	260,422	32,553	73,243.50	130,211
POP	8,138	18,310.75	32,553	4,096	9,155.25	16,276
Access	1,294	2,911.60	5,176	647	1,455.82	2,588

Congestion Conditions



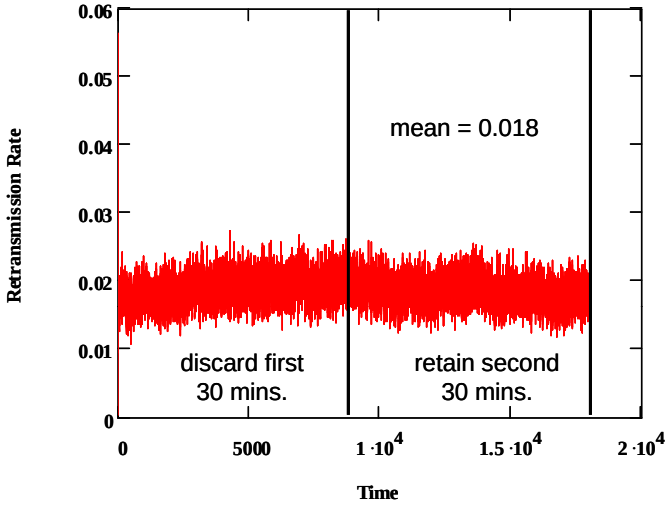


Selected Response Measurements for Experiment #2

Macroscopic Behavior

Response	Definition
y1	Average number of active flows
y2	Average number of flows in initial slow-start
y3	Average number of flows using normal congestion avoidance
y4	Average number of flows using alternate congestion avoidance
y5	Average number of flows attempting to connect
y6	Average aggregate packets output by the network every measurement interval
y7	Average number of flows completed per measurement interval
y8	Average size of congestion window per flow
y9	Average number of congestion-window increases per flow per measurement interval
y10	Average retransmission rate
y11	Average smoothed round-trip time
y12	Aggregate number of flows completed
y13	Proportion of completed flows that were Web objects
y14	Proportion of completed flows that were document downloads
y15	Proportion of completed flows that were service-pack downloads
y16	Proportion of completed flows that were movie downloads

Computed from time series

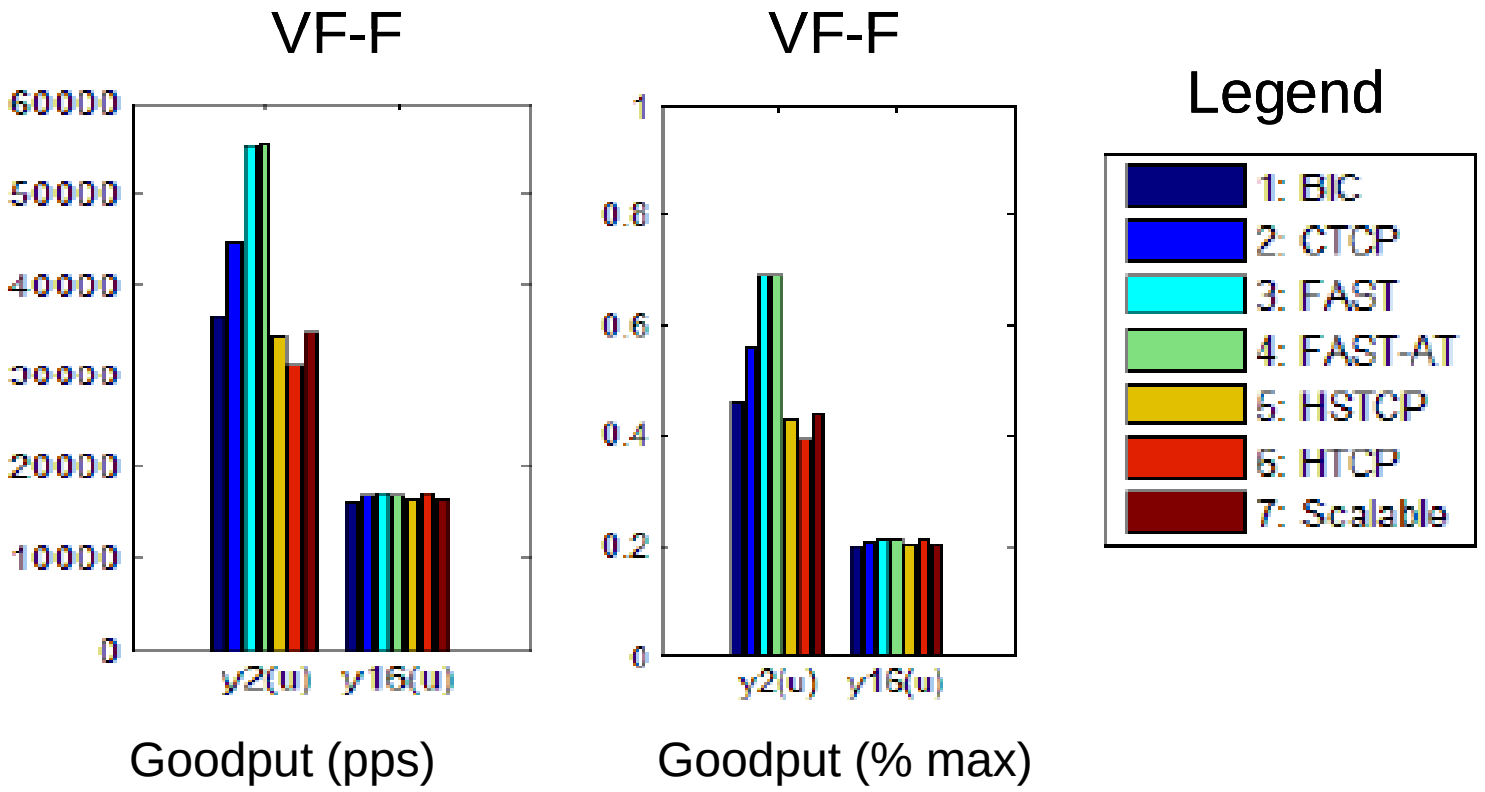


48 Goodput Measures (2 Per Flow Group x 24 Flow Groups)

Response	Definition
y2(u)	Avg. Goodput (pps) for flows using alternate algorithm
y16(u)	Avg. Goodput (pps) for flows using standard TCP

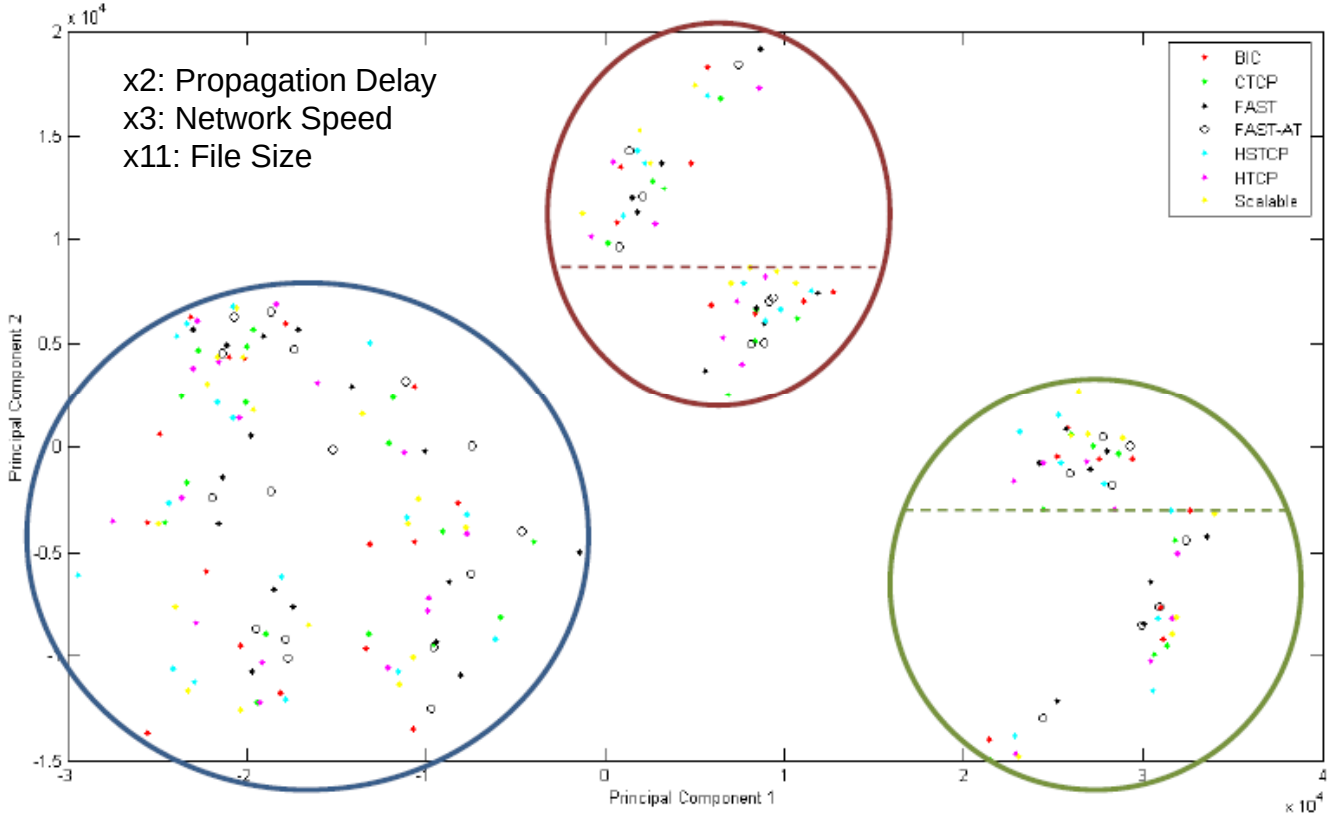
Experiment #2 Uses Analysis Techniques from Experiment #1 and Additional Techniques

Comparative Goodput bar graphs



Flows transferring movies on very fast paths with fast interface speeds (low *sst*)

Principal Components Analysis of Goodputs (high *sst*)

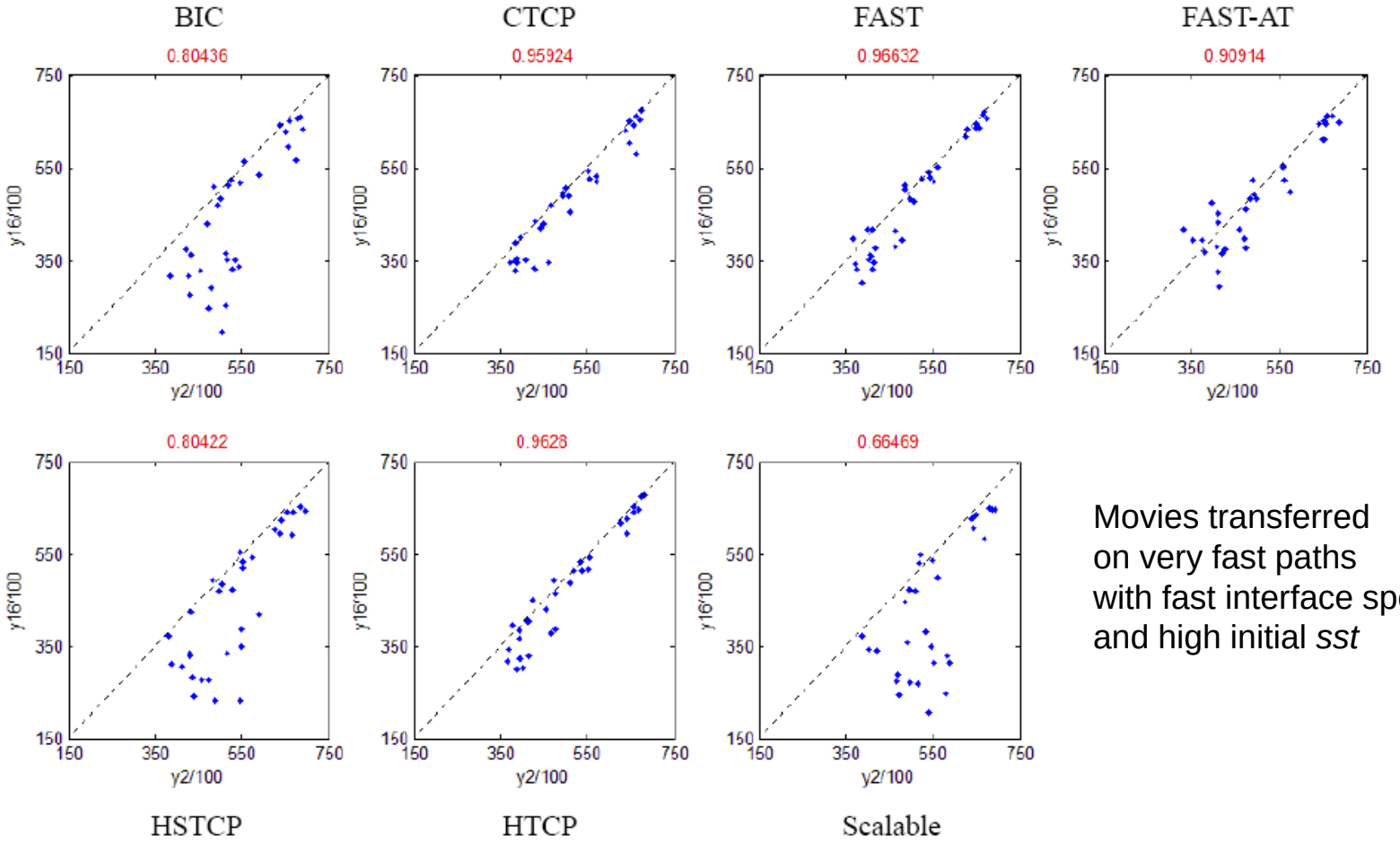


Group 1: lower network speed
Group 2: higher network speed, longer propagation delay (above line smaller file size, below line larger file size)
Group 3: higher network speed, shorter propagation delay (above line smaller file size, below line larger file size)

Suggests that under high initial *sst* congestion control algorithm not significant

An Analysis Technique

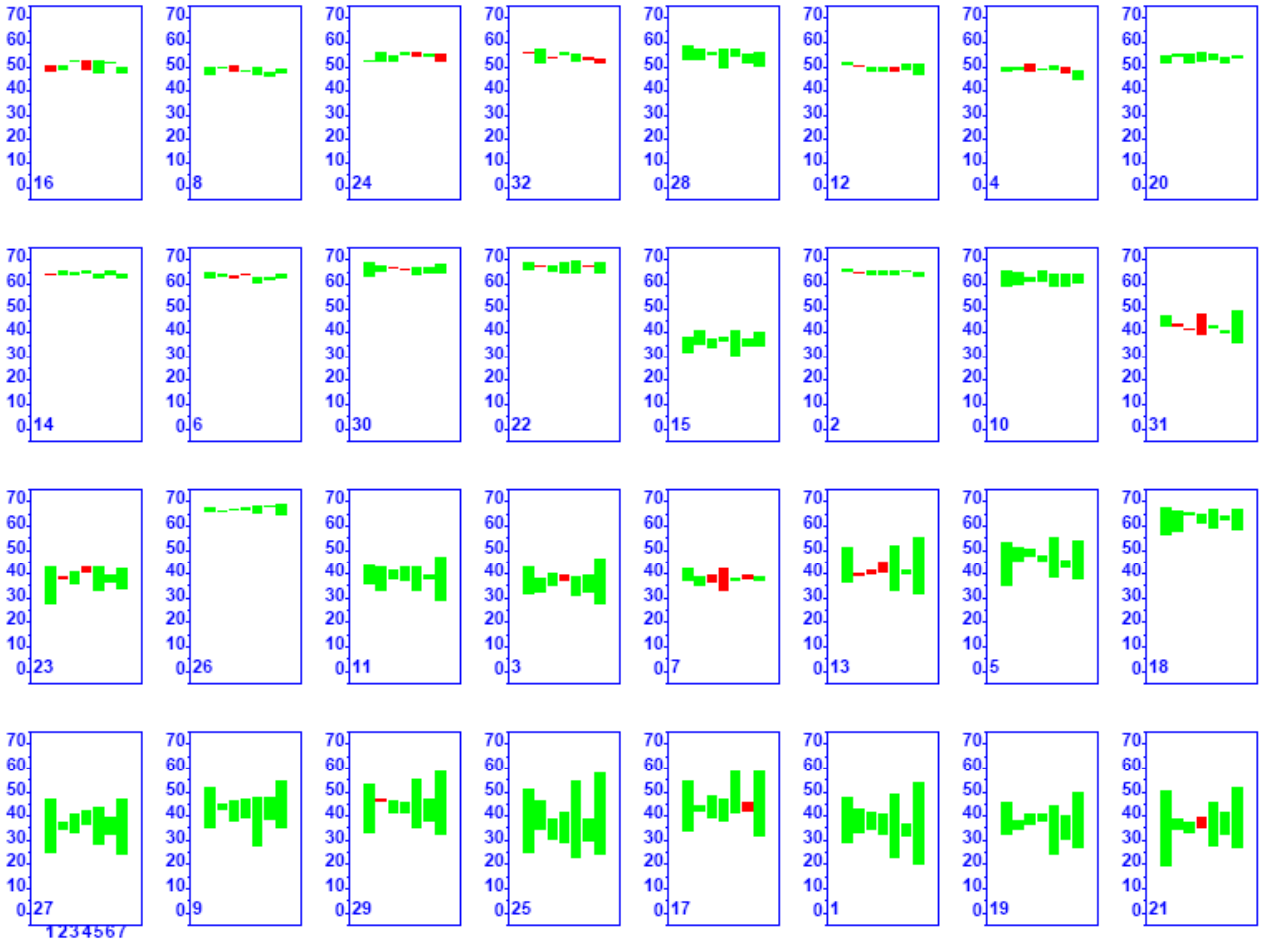
Biplots of Avg. Goodputs on alternate flows vs. TCP flows



Movies transferred
on very fast paths
with fast interface speeds
and high initial *sst*

Under many conditions Scalable, HSTCP and BIC flows achieve higher goodput than TCP flows

Histograms of Avg. Goodput differences between alternate flows and TCP flows



Movies transferred on very fast paths with fast interface speeds and high initial *sst*

Under higher congestion Scalable, HSTCP and BIC flows achieve higher goodput than TCP flows

Goodput rank matrix – CTCP flows under high initial *sst*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
16	4	5	2	3	4	4	6	5	1	5	1	3	7	4	6	3	5	5	7	4	6	7	7	6
8	5	5	2	4	3	4	5	5	4	3	2	5	7	7	3	5	6	6	3	6	5	6	6	6
24	6	5	2	2	2	3	5	7	2	1	1	1	3	7	6	6	7	5	6	4	6	6	6	6
32	7	4	4	1	3	2	4	4	6	6	2	2	1	2	7	6	6	6	5	2	6	6	6	6
28	4	1	1	2	2	2	6	1	5	3	2	2	6	3	3	6	5	6	3	7	4	6	6	6
12	4	1	3	3	3	2	1	4	3	5	2	2	5	5	7	6	3	6	7	4	7	7	7	7
4	5	4	1	2	2	2	4	3	2	2	2	2	4	3	6	6	6	6	2	5	7	6	6	6
20	4	7	1	3	2	1	2	4	1	1	2	2	2	3	7	7	7	6	1	7	7	7	7	7
14	6	7	2	2	1	2	5	4	3	2	2	2	4	7	5	7	6	6	1	4	6	6	6	6
6	4	7	1	1	1	2	7	3	2	2	2	2	5	4	7	6	5	6	7	6	5	5	6	6
30	5	1	2	2	1	3	1	4	1	1	2	2	7	6	6	5	6	6	7	2	6	6	6	6
22	2	3	3	3	1	2	4	5	3	1	1	2	5	6	6	6	6	6	3	6	6	6	6	6
15	6	6	4	1	5	1	7	5	2	7	2	1	7	7	6	7	7	7	6	7	7	7	7	7
2	1	1	5	3	3	4	2	5	3	3	2	2	2	5	6	4	1	6	7	6	4	6	6	6
10	5	5	2	4	4	3	2	6	2	5	2	3	7	7	5	6	6	6	3	4	5	6	6	6
31	4	5	3	2	2	1	6	5	3	2	1	2	4	5	5	6	3	4	3	4	7	5	5	5
23	1	3	2	2	5	2	5	4	1	2	1	1	4	3	6	7	7	7	3	2	6	6	7	7
26	1	1	2	3	4	4	7	1	2	2	3	3	2	1	2	5	3	4	4	5	4	5	4	4
11	3	1	1	1	4	6	6	2	2	2	2	2	7	4	7	6	5	5	1	4	7	6	5	5
3	2	1	2	2	2	4	6	7	2	1	2	2	5	7	6	6	4	3	4	7	7	5	5	5
7	5	5	1	2	4	5	3	3	1	1	1	2	7	6	6	7	4	6	2	7	7	7	7	7
13	1	3	3	2	3	2	4	5	2	2	2	2	6	1	6	7	6	6	5	1	7	7	6	6
5	4	5	2	2	3	2	4	5	2	2	2	2	4	6	5	6	5	6	4	4	6	6	6	6
18	4	2	1	2	4	3	1	4	2	2	3	3	3	1	6	5	4	4	3	2	5	5	5	5
27	1	4	3	1	5	1	2	3	2	2	3	2	7	6	6	2	3	1	5	7	7	5	4	4
9	1	6	2	4	3	5	3	5	2	2	3	4	2	4	2	4	2	2	5	6	4	4	5	4
29	3	6	6	4	6	2	5	6	3	1	3	3	7	5	4	5	5	5	7	6	5	5	5	5
25	4	5	4	5	4	4	3	5	3	3	4	4	6	5	4	4	3	3	6	6	5	4	5	5
17	2	2	2	6	5	6	4	4	3	4	3	3	5	4	2	2	1	1	4	7	6	6	3	4
1	4	4	6	6	2	6	4	4	1	3	3	3	6	5	4	4	3	3	3	4	5	5	5	5
19	1	7	4	3	5	4	4	5	2	3	5	3	7	5	7	7	6	5	1	3	7	7	7	7
21	3	4	2	4	3	1	7	5	1	4	2	2	4	5	3	1	3	2	4	5	4	4	5	5
	VF F M	VF N M	F F M	F N M	T F M	T N M	VF F SP	VF N SP	F F SP	F N SP	T F SP	T N SP	VF F D	VF N D	F F D	F N D	T F D	T N D	VF F WO	VF N WO	F F WO	F N WO	T F WO	T N WO

CTCP provides higher relative Goodput on smaller files

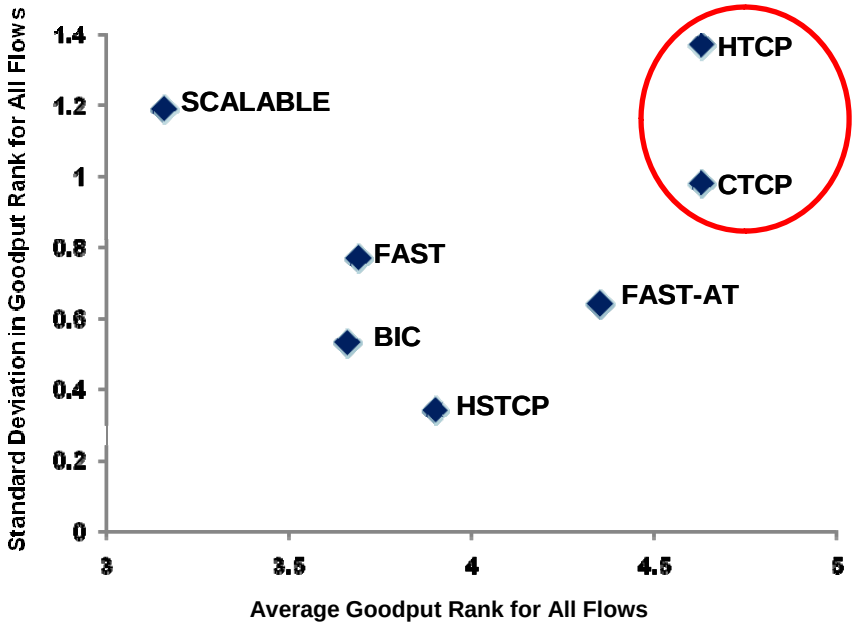
Goodput rank matrix – TCP flows competing with CTCP flows under high initial *sst*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
16	3	1	5	7	3	6	6	1	3	5	3	7	6	1	6	7	5	6	2	6	5	5	5	5
8	6	1	3	3	1	2	7	7	3	6	6	7	1	7	4	3	6	6	3	7	4	5	6	6
24	2	3	3	1	1	7	7	3	5	6	6	7	7	2	6	6	6	6	5	5	6	6	6	6
32	1	1	2	7	7	6	2	2	4	6	5	6	2	3	7	7	6	6	4	2	7	7	6	7
28	4	7	3	3	5	4	4	1	6	4	3	6	2	3	4	5	6	5	3	4	7	6	6	6
12	6	3	5	5	4	6	5	3	4	6	5	6	2	3	5	3	6	7	6	3	7	7	6	7
4	4	4	3	6	3	5	3	6	7	6	7	7	5	1	7	4	5	6	7	5	7	7	6	6
20	7	4	2	5	7	4	3	2	7	7	7	6	5	7	4	7	7	7	3	7	7	7	7	7
14	4	7	7	7	1	7	1	5	6	6	5	6	3	2	4	5	6	6	4	6	7	6	6	6
6	5	1	5	3	6	7	6	7	6	6	7	7	7	4	5	5	6	6	2	5	5	5	6	6
30	5	4	5	7	6	4	5	6	6	5	7	6	2	4	5	6	6	6	5	5	6	6	6	6
22	6	6	1	7	2	3	6	7	4	7	4	6	3	5	6	6	6	6	4	1	6	6	6	6
15	6	7	7	7	7	7	7	7	6	7	7	7	4	7	7	7	7	7	6	6	7	7	7	7
2	5	7	5	5	3	6	6	4	6	7	4	5	4	1	6	6	6	6	1	6	6	6	6	6
10	4	7	3	6	6	7	5	5	6	6	6	6	4	4	5	7	6	6	5	2	6	6	6	6
31	6	4	7	6	7	3	6	4	3	7	6	7	1	2	4	7	6	6	4	6	6	6	6	6
23	6	5	4	3	3	4	1	6	6	7	7	7	2	6	6	7	7	6	5	4	7	7	7	7
26	4	3	7	5	5	5	3	3	2	5	6	5	7	7	5	5	5	5	5	1	5	5	5	5
11	2	7	6	5	2	5	3	5	6	5	3	7	5	7	5	6	5	5	6	2	5	6	5	5
3	5	7	6	5	7	7	2	3	5	7	6	5	3	7	6	5	5	5	6	7	6	5	5	5
7	1	7	5	4	1	6	4	2	7	2	4	6	2	3	6	7	7	7	2	5	7	7	7	7
13	4	6	4	6	6	7	5	7	6	7	6	6	1	3	7	6	6	6	3	1	6	7	6	6
5	5	4	7	7	2	2	7	6	6	6	7	6	5	6	7	6	6	6	5	6	6	6	6	6
18	2	6	3	7	2	5	5	6	4	5	6	6	7	7	5	6	5	5	1	7	5	5	5	5
27	6	6	3	2	3	2	7	4	3	6	5	6	6	6	6	4	5	5	4	7	7	7	5	5
9	7	5	6	4	2	4	4	4	4	5	6	5	7	4	4	4	5	5	2	4	4	4	5	5
29	7	6	5	7	3	6	5	5	2	6	7	6	2	5	5	5	5	5	6	3	5	5	5	5
25	7	7	5	4	3	7	6	6	7	2	5	3	7	5	5	5	5	5	4	3	5	5	5	5
17	6	7	3	7	7	7	6	4	5	7	2	4	4	5	6	6	5	5	4	6	5	6	5	5
1	6	5	1	2	3	7	7	6	5	4	7	5	7	5	5	5	5	5	7	5	5	5	5	5
19	5	6	5	6	7	7	6	6	2	7	6	6	5	3	7	7	6	7	1	4	7	7	7	7
21	6	5	5	4	6	6	4	5	6	2	7	5	6	6	4	5	6	6	4	6	6	6	6	6
	VF FM	VF NM	F FM	F NM	T FM	T NM	VF FSP	VF NSP	F FSP	F NSP	T FSP	T NSP	VF FD	VF ND	F FD	F ND	T FD	T ND	VF FWO	VF NWO	F FWO	F NWO	T FWO	T NWO

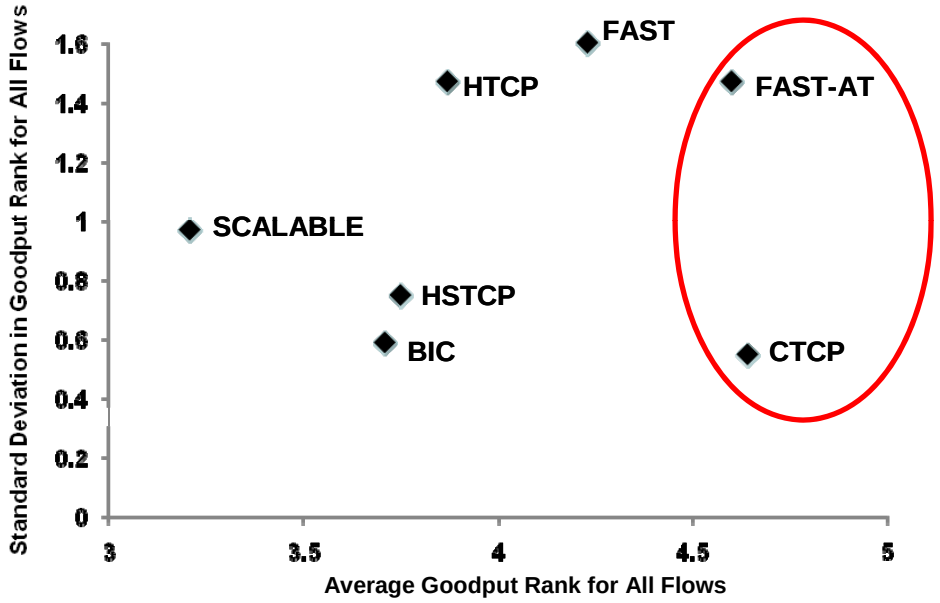
TCP flows achieve high relative Goodput when competing with CTCP flows

Average and Standard Deviation in Goodput Ranks

high initial *sst*



low initial *sst*



CTCP achieves relatively high ranking Goodput for its flows and competing TCP flows

Utility and Safety

1. *Increase rate*: How quickly can the maximum transmission rate be achieved?

2. *Loss/Recovery processing*:
   a. How much does the protocol reduce transmission rate upon a loss?
   b. How quickly does the protocol increase transmission rate after a reduction?

3. *Fairness*: How well do standard TCP flows do when competing with alternates?

4. *Utility bounds*: Under what circumstances can alternate congestion control algorithms provide improved user goodputs?

5. *Safety*: Will widespread deployment of alternate algorithms induce undesirable macroscopic characteristics in the Internet?

Increase Rate

- Assuming low congestion, setting of initial *sst* is a key factor
  - High initial *sst* – all algorithms (standard TCP included) achieved maximum transmission rate with the same (exponential) quickness
  - Low initial *sst* – alternate algorithms achieved maximum transmission rate more quickly than (linear) increase of standard TCP

- Under heavy congestion, setting of initial *sst* matters little because initial slow start terminates upon first packet loss and a flow enters congestion avoidance, where loss/recovery processing determines goodput

- On real TCP flows receivers may convey a window (*rwnd*) that can restrict goodput because sources pace transmission based on **min**(*cwnd*, *rwnd*). Typically, *rwnd* < *cwnd*. In our studies, we assume an infinite *rwnd* in order to compare effects of congestion control algorithms. Goodput on many TCP flows in a real network might be constrained by *rwnd*, so that alternate congestion control algorithms would provide little advantage over standard TCP. In fact, even TCP congestion control does not have much influence when *rwnd* < *cwnd.*

Loss/Recovery Processing

- One group of algorithms (Scalable TCP, BIC[1] and HSTCP) reduce transmission rate less than standard TCP after a packet loss
  - Unfair to TCP flows and to new flows using alternate algorithms

- Another group of algorithms (CTCP, FAST and FAST-AT) reduce transmission rate by ½ following a loss (HTCP is a hybrid with reduction between 20 and 50%)
  - These algorithms seek to obtain higher goodput by increasing transmission rate more quickly than standard TCP (the rate of increase varies with the algorithm)
  - HTCP reverts to TCP congestion avoidance for 1 s after each loss, which can lead to lower goodputs than other alternate algorithms

- Under extreme spatiotemporal congestion, most alternate algorithms have a low-window threshold and revert to standard TCP congestion avoidance procedures (giving no advantage to alternate procedures)
  - FAST and FAST-AT do not use TCP congestion avoidance under any conditions, which can lead to oscillatory behavior and increased loss rates

[1]Note that on repeated losses occurring close in time, BIC can reduce *cwnd* substantially more than standard TCP – thus, on paths with very severe congestion TCP can provide higher goodput than BIC

Fairness

- All alternate algorithms take steps to provide improved goodput over TCP – thus comparing fairness must consider relative performance of TCP flows when competing with flows using each of the alternate algorithms

- We found CTCP, HTCP and FAST-AT to be most fair to TCP flows
  - Under low initial *sst* FAST-AT is more unfair because of its quick increase in rate
  - Injecting more FAST-AT packets induced more losses in TCP flows, which could recover only linearly

- We found Scalable TCP, BIC and FAST to be most unfair to TCP flows
  - Established Scalable and BIC flows (on large files) tended to maintain higher transmission rates than TCP flows after losses, while FAST recovered more quickly, and these alternate algorithms induced more losses in TCP flows

- HSTCP appeared moderately fair to TCP flows, especially under conditions of lower congestion and under low initial *sst* – HSTCP appeared unfair under conditions of heavy congestion

- We found that Scalable TCP, BIC and HSTCP are also unfair to competing flows that are newly arriving

Utility Bounds

- We found that alternate congestion control algorithms could provide increased utility (goodput) for users – however, this utility would arise only under a specific combination of circumstances
  - Flow's *rwnd* must not be constraining flow transmission rate
  - Flow's initial *sst* must be relatively low
  - Flow must be transferring a large file
  - Flow's packets must be transiting a relatively uncongested path (i.e., experiencing only sporadic losses) or else users must be willing to tolerate marked unfairness in trade for increased gooput
- How likely is this combination of circumstances on a given Internet flow?
  - Certainly possible to engineer a network, or segments of a network, to provide specific users with improved goodput compared withTCP
  - We suspect a rather low probability for such circumstances to arise generally in the Internet
- We conclude that alternate congestion control algorithms can provide improved user goodput – however, most users seem unlikely to benefit very often

Safety

- We can answer this only in part – additional cautionary findings may be possible
  - We simulated either homogeneous networks where all flows used one congestion control algorithm or mixes of TCP flows competing with flows using one alternate algorithm at a time
  - The real Internet could contain a mix of many different types of congestion algorithm
- For most algorithms we studied, under most conditions, we found little significant change in macroscopic network characteristics
- FAST and FAST-AT are exceptions to this general finding
  - Under high spatiotemporal congestion, where there were insufficient buffers to support flows transiting specific routers, FAST and FAST-AT entered an oscillatory behavior where the flow *cwnd* increased and decreased rapidly with large amplitude
  - Under such conditions the network showed increased loss and retransmission rates, a higher number of flows pending in the connecting state and a lower number of flows completed over time
- We recommend the need for additional study of FAST and FAST-AT prior to widespread deployment and use on the Internet

Characteristics of Individual Alternate Algorithms

1. *Implementation complexity*: How much code required to implement an algorithm?
2. *Activation trigger*: What causes a flow to switch from standard TCP congestion avoidance to alternate procedures?
3. *Goodput latency*: What is the time required for a flow to achieve maximum transmission rate?
4. *Recovery latency*: What is the time required for a flow to recover maximum transmission rate after a period of congestion (with sustained losses)?

Characteristics of Individual Alternate Algorithms

Algorithm	Implementation Complexity	Activation Trigger	Goodput Latency (avg)	Recovery Latency (avg)
BIC	high	14 packets	18.8 s	71.3 s
CTCP	moderate	41 packets	7.9 s	2.9 s
FAST	low	none	3.7 s	6.6 s
FAST-AT	moderate	none	3.7 s	26.0 s
HSTCP	low	31 packets	22.4 s	10.0 s
H-TCP	moderate	1 s w/o loss	16.6 s	10.0 s
Scalable TCP	low	16 packets	17.8 s	22.5 s

Recommendations

- Under some circumstances users may benefit from alternate congestion control algorithms – thus it makes sense to deploy such algorithms on the Internet
- Probability appears quite low that a specific user will see benefits on a particular file transfer
- Among the algorithms we studied, CTCP appears to provide the best balance of properties
  - Under low congestion, CTCP can increase transmission rate relatively quickly
  - CTCP reduces rate relatively quickly under sustained congestion and recovers maximum transmission rate quickly when congestion eases
  - CTCP appears relatively friendly to flows using standard TCP
  - CTCP seems unlikely to induce large shifts in the Internet's macroscopic properties
- FAST and FAST-AT have some appealing properties, especially with respect to achieving maximum transmission rate quickly on high-bandwidth, long-delay paths and recovering quickly from sporadic losses
  - However, when transiting highly congested paths with insufficient buffers to support flow volume, FAST and FAST-AT can enter a regime of oscillatory rates

ADDITIONAL DISCUSSION?

BACKUP SLIDES

Why are researchers proposing alternate Internet congestion control algorithms?

Standard TCP - 1 Gbps Path Between Chicago and Dublin

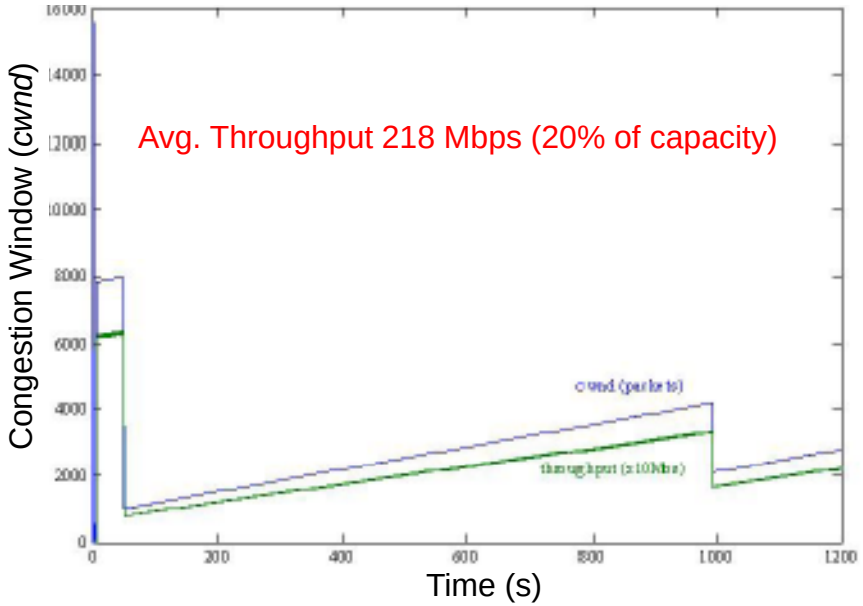


Figure 1 from Li et al. 2007. Experimental Evaluation of TCP Protocols for High-Speed Networks. *Transactions on Networking*. 15:5, 1109-1122.

Example Proposals

BIC
Compound TCP
CUBIC (not included in this study)
FAST
HSTCP
H-TCP
Scalable TCP

Some common themes among proposals:

(1) alterations only to congestion avoidance (not initial slow start)
(2) relative to TCP: most reduce *cwnd* less on packet loss and all increase *cwnd* faster
(3) most have mode switch between TCP and alternate behavior (FAST is a notable exception)

How are researchers evaluating proposed congestion control algorithms?

Analytical models of single long-lived flows

Blanc, A., Avrachenkov, K. and Collange, D. 2009. Comparing some high speed TCP versions under bernoulli losses. In *Proceedings of the International Workshop on Protocols for Future, Large-Scale and Diverse Network Transports (PFLDNet 2009)*, 59-64.

Simulation studies in small topologies

Jackson, T. and Smith, P. 2008. Building a Network Simulation Model of the TeraGrid Network. In *Proceedings of TeraGrid'08*.

Shimonishisi, H., Sanadidi, M. and Murase, T. 2007."Assessing Interactions among Legacy and High-Speed TCP Protocols. In *Proceedings of the 5th International Workshop on Protocols for Fast Long-Distance Networks.*

Empirical evaluations in small topologies

Li et al. 2007. Experimental Evaluation of TCP Protocols for High-Speed Networks. *Transactions on Networking*. 15:5, 1109-1122.

Lee, G., Lachlan, A., Tang, A. and Low, S. 2007. WAN-in-Lab: Motivation, Deployment and Experiments. In *Proceedings of the 5th International Workshop on Protocols for Fast Long-Distance Networks.*

Quantitative Summary of Our Experiments Comparing Congestion Control Algorithms

1152 simulations encompassing nearly 50 billion flows and 20 trillion packets and requiring > 14 processor years

Exp. #	Parameter Combinations	Algorithms Compared	Simulation Runs	Processor Hours	Simulated Flows	Simulated Packets
1a	32	7	224	16,598.4	$>16.5 \times 10^9$	$>3 \times 10^{12}$
1b	32	8	256	~1,658.0	$>2 \times 10^9$	$\sim 460 \times 10^9$
2a	32	7	224	5,857.2	$>2.5 \times 10^9$	$>1.5 \times 10^{12}$
2b	32	7	224	5,638.5	$>2.5 \times 10^9$	$>1.4 \times 10^{12}$
2c	32	7	224	94,355.3	$>26 \times 10^9$	$>14 \times 10^{12}$
All	160	(~7)	1152	124,107.4	$\sim 49.5 \times 10^9$	$>20.0 \times 10^{12}$

Adopt 2-Level Orthogonal Fractional Factorial Designs

Sample 2^{9-4} design instantiated

Factor->	X2	X3	X4	X5	X7	X9	X11	X12	X15
Condition	--	--	--	--	--	--	--	--	--
1	1	800	0.5	3	0.7	5000	100	0.04/0.004/0.0004	0.7
2	1	1600	0.5	2	0.3	5000	100	0.04/0.004/0.0004	0.3
3	2	800	0.5	2	0.7	5000	100	0.02/0.002/0.0002	0.3
4	2	1600	0.5	3	0.3	5000	100	0.02/0.002/0.0002	0.7
5	1	800	1	2	0.3	5000	100	0.02/0.002/0.0002	0.7
6	1	1600	1	3	0.7	5000	100	0.02/0.002/0.0002	0.3
7	2	800	1	3	0.3	5000	100	0.04/0.004/0.0004	0.3
8	2	1600	1	2	0.7	5000	100	0.04/0.004/0.0004	0.7
9	1	800	0.5	3	0.3	7500	100	0.02/0.002/0.0002	0.3
10	1	1600	0.5	2	0.7	7500	100	0.02/0.002/0.0002	0.7
11	2	800	0.5	2	0.3	7500	100	0.04/0.004/0.0004	0.7
12	2	1600	0.5	3	0.7	7500	100	0.04/0.004/0.0004	0.3
13	1	800	1	2	0.7	7500	100	0.04/0.004/0.0004	0.3
14	1	1600	1	3	0.3	7500	100	0.04/0.004/0.0004	0.7
15	2	800	1	3	0.7	7500	100	0.02/0.002/0.0002	0.7
16	2	1600	1	2	0.3	7500	100	0.02/0.002/0.0002	0.3
17	1	800	0.5	2	0.3	5000	150	0.02/0.002/0.0002	0.3
18	1	1600	0.5	3	0.7	5000	150	0.02/0.002/0.0002	0.7
19	2	800	0.5	3	0.3	5000	150	0.04/0.004/0.0004	0.7
20	2	1600	0.5	2	0.7	5000	150	0.04/0.004/0.0004	0.3
21	1	800	1	3	0.7	5000	150	0.04/0.004/0.0004	0.3
22	1	1600	1	2	0.3	5000	150	0.04/0.004/0.0004	0.7
23	2	800	1	2	0.7	5000	150	0.02/0.002/0.0002	0.7
24	2	1600	1	3	0.3	5000	150	0.02/0.002/0.0002	0.3
25	1	800	0.5	2	0.7	7500	150	0.04/0.004/0.0004	0.7
26	1	1600	0.5	3	0.3	7500	150	0.04/0.004/0.0004	0.3
27	2	800	0.5	3	0.7	7500	150	0.02/0.002/0.0002	0.3
28	2	1600	0.5	2	0.3	7500	150	0.02/0.002/0.0002	0.7
29	1	800	1	3	0.3	7500	150	0.02/0.002/0.0002	0.7
30	1	1600	1	2	0.7	7500	150	0.02/0.002/0.0002	0.3
31	2	800	1	2	0.3	7500	150	0.04/0.004/0.0004	0.3
32	2	1600	1	3	0.7	7500	150	0.04/0.004/0.0004	0.7

Sample experiment using 9 parameters

1. Selected appropriate $n = 2^{p-k}$ design template
2. Select two values for each parameters
3. Substitute parameter levels in template
4. Fix remaining (11) model parameters

Fixed values assigned to remaining parameters

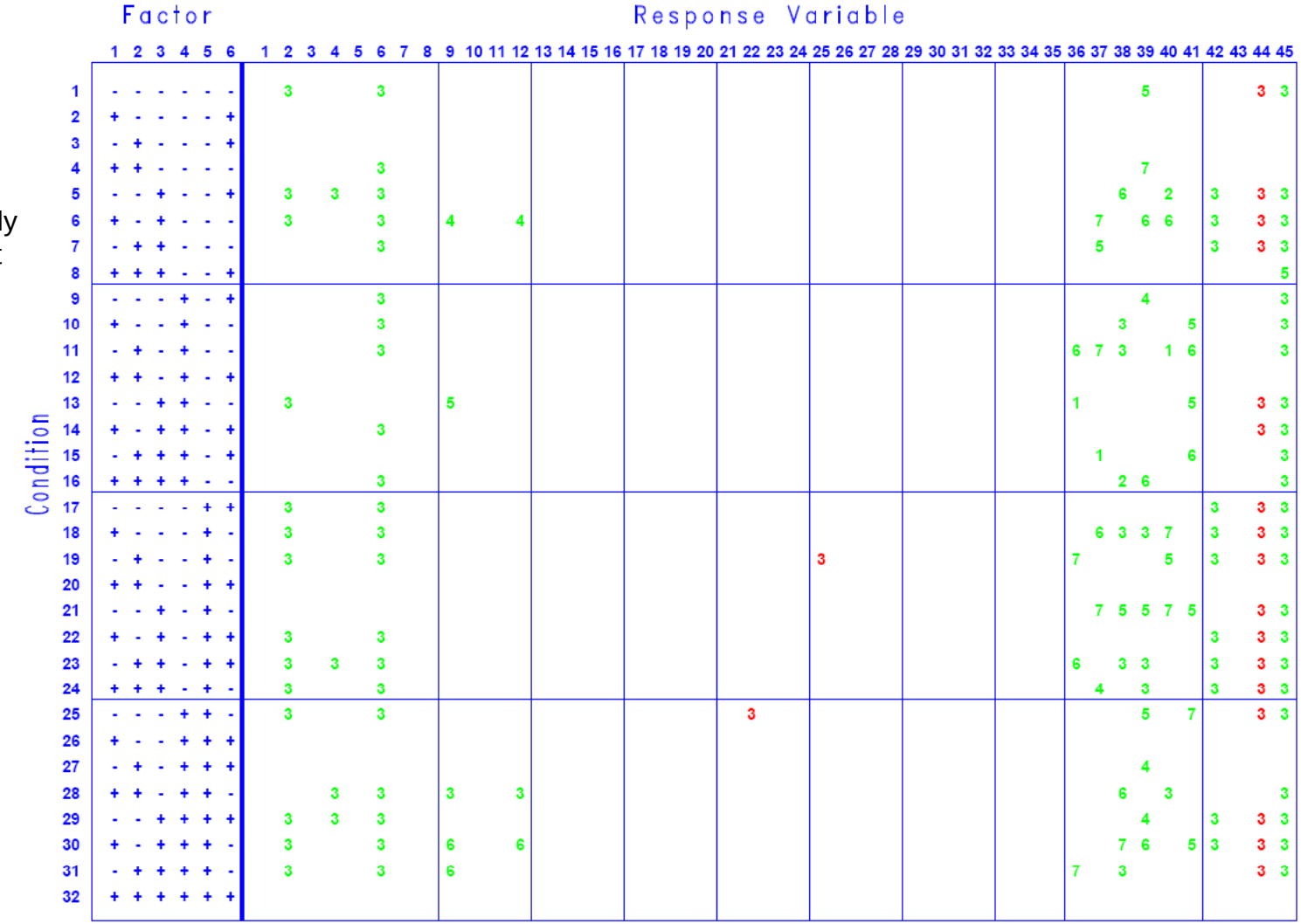
Parameter	Assigned Value
X1	Abilene Topology (Backbone: 11 routers and 14 links; 22 PoP routers; 139 Access routers)
X6	*probNs* = 0.1, *probNsf* = 0.6
X7	*probNr* = 0.6, *probNrf* = 0.2
X10	0 (all users have infinite patience)
X13	*Jon* = 1; *Joff* = 1; *Jx* = 1 (no explicit spatiotemporal congestion)
X14	no long-lived flows
X16	initial *cwnd* = 2 (default Microsoft Windows™ value)
X17	initial *sst* = $2^{31}/2$ (arbitrary large value)
X18	*M* = 200 ms
X19	*MI* = 18,000 (x .2 *M* =) 3600 s
X20	*prON* = 0.25, *prONsecond* = 0.08, *prONthird* = 0.17

baseSources = 100

Scale experiment up to a larger faster network simply, e.g., multiply X3 values by 10 and set *baseSources* = 1000

Filters Applied to Condition-Response Summaries

Statistically significant outliers & 10% Δ



Design for Experiment #1b

Experiment #1b (smaller, slower network and low initial *sst* and added FAST-AT)

Router Speeds

Router	PLUS (+1)	Minus (-1)
Backbone	28.8 Gbps	14.4 Gbps
POP	3.6 Gbps	1.8 Gbps
Normal Access	360 Mbps	180 Mbps
Fast Access	720 Mbps	360 Mbps
Directly Connected Access	3.6 Gbps	1.8 Gbps

Propagation Delays

	Min	Avg	Max
PLUS (+1)	12	81	200
Minus (-1)	6	41	100

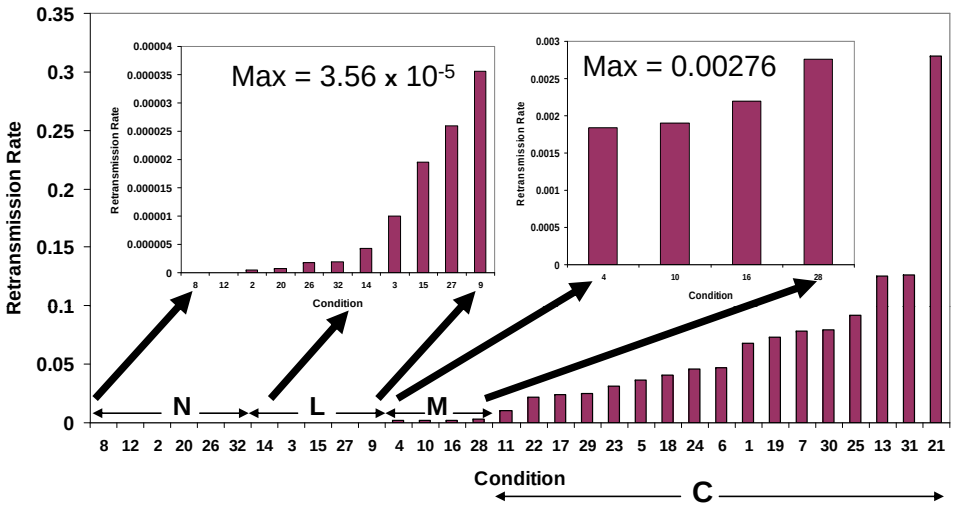
Number of Sources

PLUS (+1)	Minus (-1)
27,800	17,460

Router Buffer Sizes

	PLUS (+1)			Minus (-1)		
Router	Min	Avg	Max	Min	Avg	Max
Backbone	48,830	109,866	195,317	547	1,236	2,208
POP	6,104	13,734	24,415	105	240	431
Access	971	2,184	6,104	44	99	105

Congestion Conditions



256 Total Runs (32 conditions x 8 algorithms)

Statistic	Flows Completed	Data Packets Sent
Avg. per condition	8,329,266	897,379,391
Min. per condition	4,329,268	380,349,161
Max. per condition	16,729,532	1,749,461,097
Total all runs	2,132,292,096	229,729,124,182

Design for Experiment #2c

Domain View of Experiment #2c – Repeat #2a with larger, faster network

Router Speeds

Router	PLUS (+1)	Minus (-1)
Backbone	384 Gbps	192 Gbps
POP	48 Gbps	24 Gbps
Normal Access	4.8 Gbps	2.4 Gbps
Fast Access	9.6 Gbps	7.2 Gbps
Directly Connected Access	48 Gbps	24 Gbps

Propagation Delays

	Min	Avg	Max
PLUS (+1)	12	81	200
Minus (-1)	6	41	100

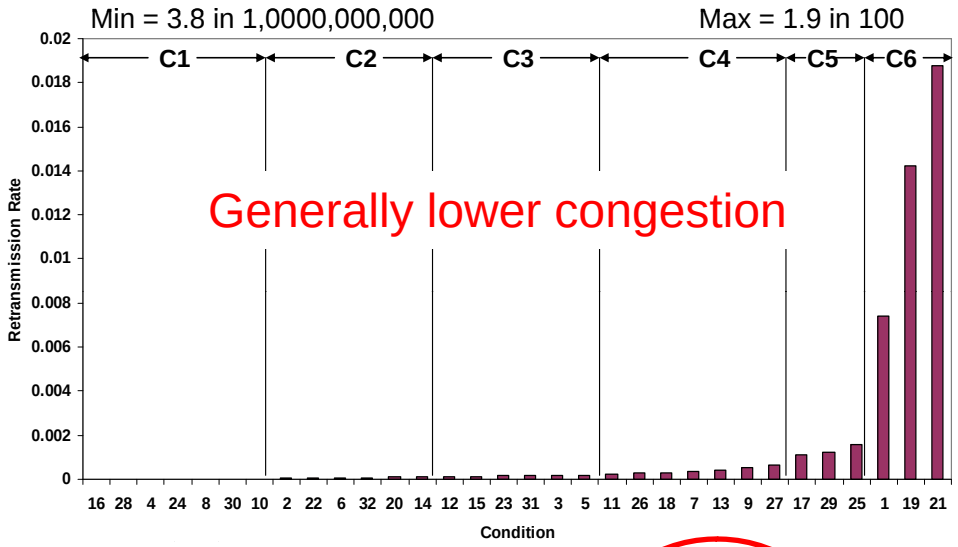
Number of Sources

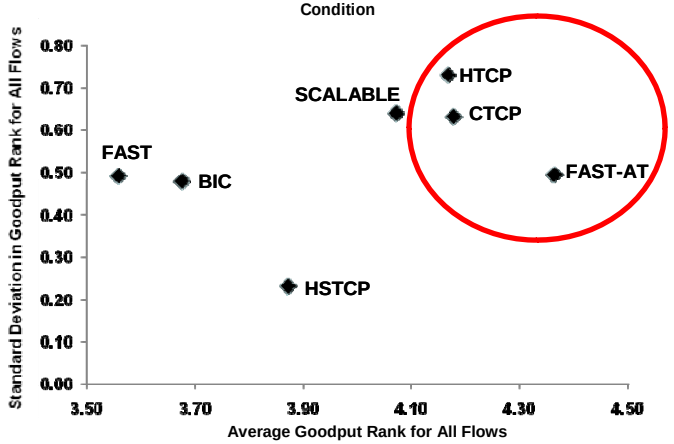
PLUS (+1)	Minus (-1)
261,792	174,600

Router Buffer Sizes

	x2 1.0			x2 0.5		
Router	Min	Avg	Max	Min	Avg	Max
Backbone	651,055	1,464,874	2,604,219	325,527	732,437	1,302,109
POP	81,382	183,110	325,528	40,691	91,555	162,764
Access	12,939	29,113	51,757	6,469	14,556	25,878

Congestion Conditions





Potential Future Work

- Study additional proposed congestion control algorithms
  - Of particular interest, CUBIC has replaced BIC as the congestion control algorithm enabled by default in Linux
- Consider scenarios where multiple alternate congestion control algorithms are mixed together in the same network
- Validate findings against live, controlled experiments configured in GENI (Global Environment for Network Innovation) or similar test bed environment
- Researchers could exploit our findings to propose improvements to the algorithms we studied – compensating for identified weaknesses, while retaining strengths
- Our findings might also help other researchers to improve future designs for additional congestion control algorithms