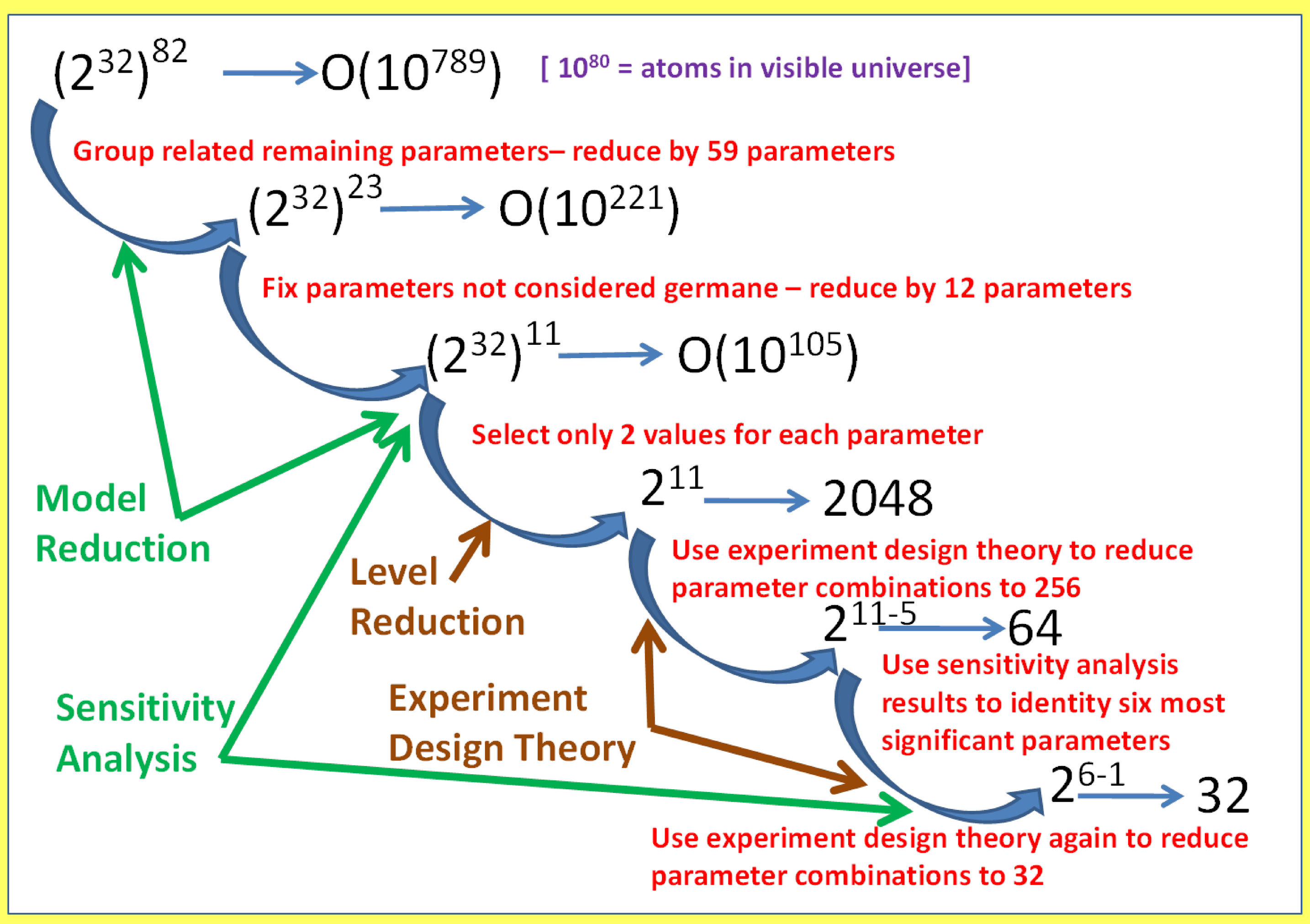
Efficient Sensitivity Analysis Method for Large Cloud Simulations

Kevin Mills, James Filliben and Chris Dabrowski from NIST

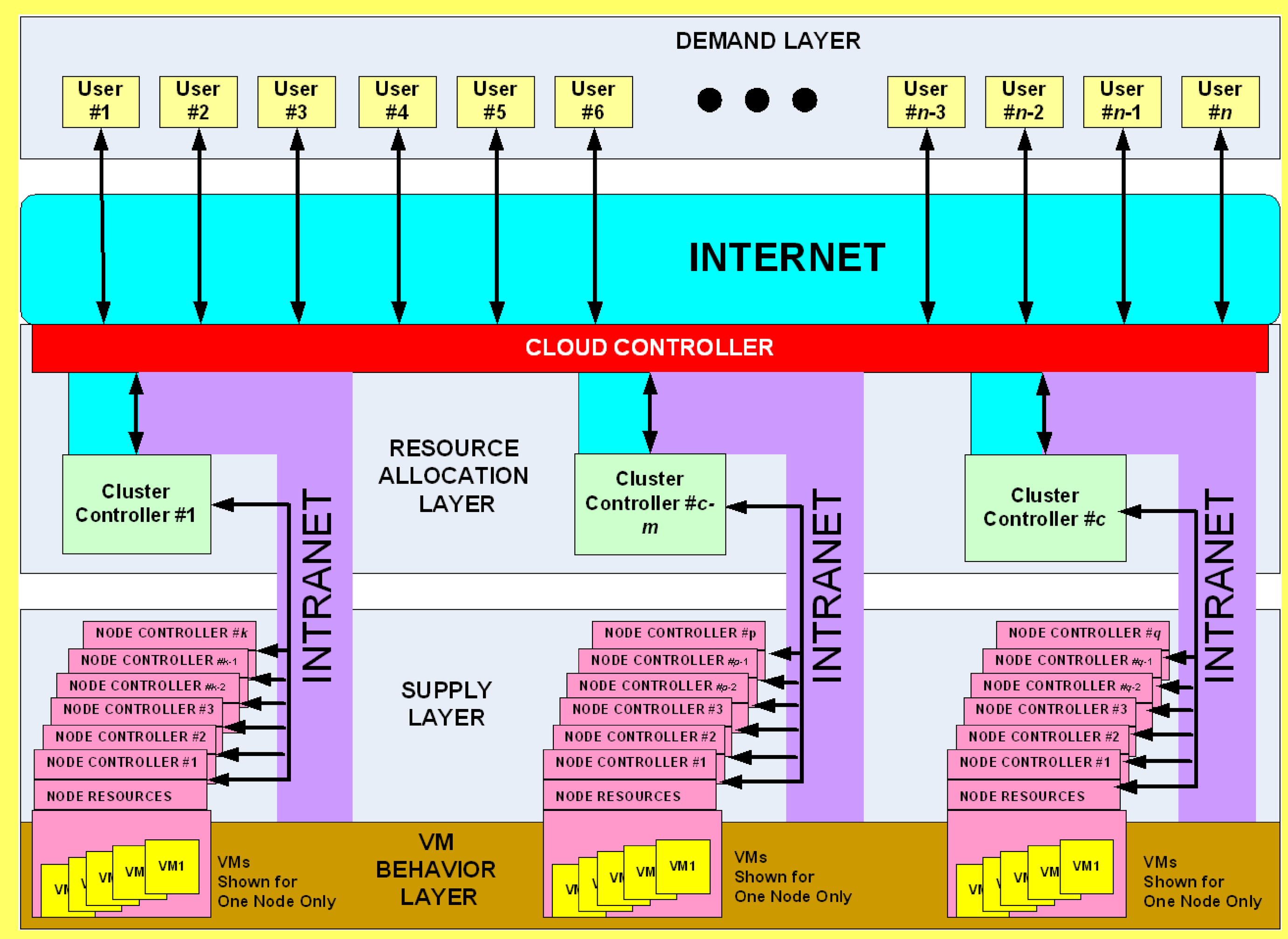
Proceedings of IEEE Cloud 2011, Washington D.C., July 4-9, 2011

We developed an objective method to compare distributed control algorithms in simulations of large systems. Our method has four steps: (1) develop a reduced-parameter model (i.e., Koala), (2) determine the most significant model behaviors and the parameters that most influence those behaviors, (3) construct a set of parameter combinations under which control algorithms should be compared and (4) use multi-dimensional data analysis techniques to find patterns revealing significant similarities and differences among the algorithms. This work describes steps (1) and (2) as applied to Koala, an IaaS (Infrastructure-as-a-Service) cloud simulator.

Theory



Koala Simulator



Experiment Design

2-Level OFF Experiment Designs Reduce # of Parameter Combinations, While Improving Global Coverage and Minimizing Error in Effect Estimates in comparison with comparable Factor-at-a-Time (FAT) Designs

We selected two pairs of level settings (SA1 & SA2) and two system sizes (small & large)

Adopted 2-Level (2^{11-5}) "Resolution IV" OFF experiment design, requiring 64 simulations per experiment

Instantiated 4 designs, and simulated 6 repetitions (different random number seeds) with the 2 smaller designs

Required (6 x 2 + 2) x 64 = 896 simulations

	SA1-small and SA1-large		SA2-small and SA2-large	
Parameter	Plus Level	Minus Level	Plus Level	Minus Level
x1	1200 hours	600 hours	1600 hours	200 hours
x2	500 (SA1-small) 5000 (SA1-large)	250 (SA1-small) 2500 (SA1-large)	750 (SA2-small) 7500 (SA2-large)	125 (SA2-small) 1250 (SA2-large)
x3	PU1 = 0.2 PU2 = 0.2 PU3 = 0.1 PU4 = 0.1 WS1 = 0.15 WS2 = 0.07 WS3 = 0.03 PS1 = 0.1 PS2 = 0.01 MS1 = 0.1 MS3 =0.01 DS1 = 0.10 DS2 = 0.01	PU1 = 1/6 PU2 = 1/6, WS1 = 1/6 MS1 = 1/6 PS1 = 1/6 DS1 = 1/6	PU1 = 0.4 PU2 = 0.4 PU3 = 0.1 PU4 = 0.05 PU5 = 0.025 PU6 = 0.025	WS1 = 0.25 WS2 = 0.15 WS3 = 0.1 PS1 = 0.35 PS2 = 0.04 PS3 = 0.01 DS1 = 0.08 DS2 = 0.015 DS3 = 0.005
x4	8 hours (α = 1.2)	4 hours (α = 1.2)	12 hours (α = 1.2)	2 hours (α = 1.2)
x5	20 (SA1-small) 40 (SA1-large)	10 (SA1-small) 20 (SA1-large)	30 (SA2-small) 40 (SA2-large)	5 (SA2-small) 10 (SA2-large)
x6	200 (SA1-small) 1000 (SA1-large)	100 (SA1-small) 500 (SA1-large)	400 (SA2-small) 1500 (SA2-large)	50 (SA2-small) 250 (SA2-large)
x7	C22 = 1.0	C8 = 0.25 C14 = 0.25 C18 = 0.25 C22 = 0.25	C14 = 0.2 C16 = 0.2 C18 = 0.2 C20 = 0.2 C22 = 0.2	C2 = 0.1 C4 = 0.1 C6 = 0.1 C8 = 0.1 C10 = 0.1 C12 = 0.1 C16 = 0.1 C22 = 0.3
x8	Percent Allocated	Least-Full First	Percent Allocated	Least-Full First
x9	Next-Fit	First-Fit	Next-Fit	First-Fit
x10	4	1	8	1
x11	10^{-3} to 10^{-8}	10^{-4} to 10^{-9}	10^{-2} to 10^{-7}	10^{-5} to 10^{-10}

Significant Behaviors

Correlation Analysis & Clustering (CAC) Reduces Dimensionality

We identified an **8-dimensional response space** within the 40 responses

Compute correlation coefficient (r) for all response pairs

Examine frequency distribution for all $|r|$ to determine threshold for correlation pairs to retain; $|r| > 0.65$, here

Create clusters of mutually correlated pairs; each cluster represents one dimension

Select one response from each cluster to represent the dimension; we selected response with largest mean correlation that was not in another cluster*

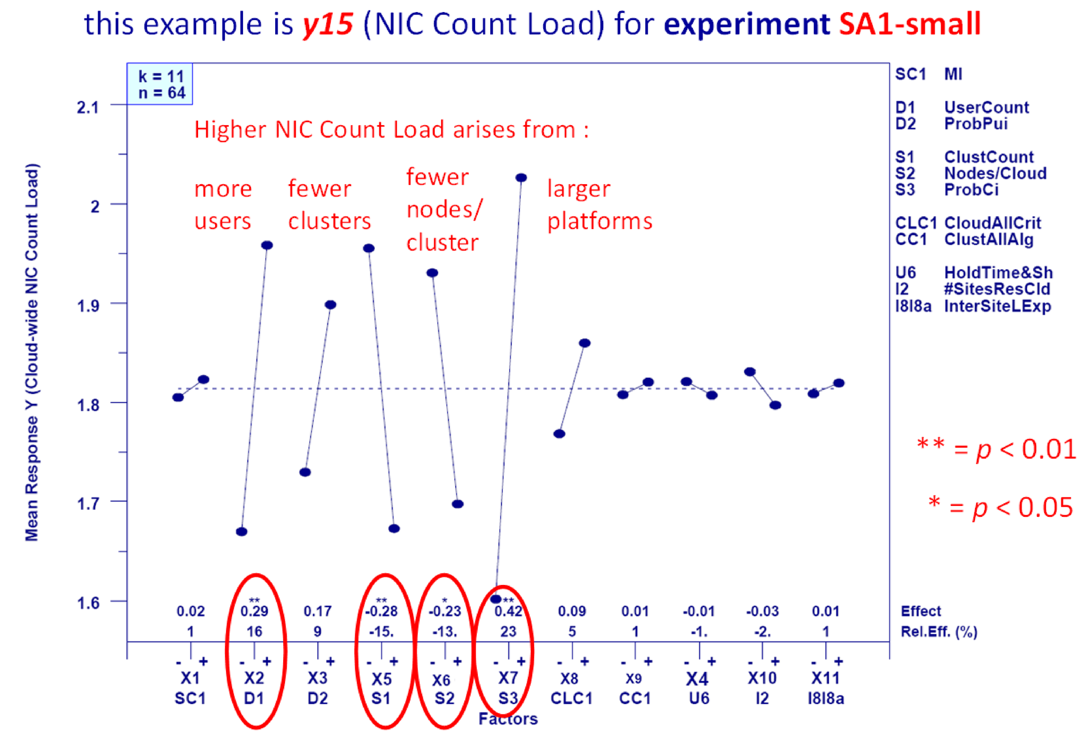
Response Dimension	SA1-small (9 dimensions)	SA1-large (8 dimensions)	SA2-small (10 dimensions)	SA2-large (9 dimensions)
Cloud-wide Demand/Supply Ratio	y1, y2, **y3**, y5, y6, y8, y9, y10, y13, y23, y24, y25, y29, y30, y32, y34, y36, y38	y1, y2, **y3**, y5, y6, y7, y8, y9, y10, y13, y23, y34, y25, y29, y30, y32, y33, y34, y36, y38	y1, **y2**, y3, y5, y6, y8, y9, y10, y11, y13, y14, y15, y23, y24, y25, y38	y1, y2, y3, y5, y6, y8, y9, **y23**, y24, y25, y38
Cloud-wide Resource Usage	y10, y11, y12, y13, y14, **y15**	y10, y11, y12, y13, y14, **y15**	**y10**, y11, y12, y13, y14, y15	**y10**, y11, y12, y13, y14, y15
Variance in Cluster Load	y16, y17, y18, y19,y20, y21, **y26**, y27	y16, y17, y18, y19,y20, y21, **y26**, y27	y16, y18, y19, y20, y21, y26, **y27** / **y17** (Mem. Util)	y16, y17, y18, **y19**,y20, y21, y26, y27
Mix of VM Types	y34, **y35** (WS) / **y31** (MS)	**y31** (MS)	y12, y14, y15, y30, y31, y33, y34, y35, **y36**	y14, y15, y30, **y31**, y33, y34, y35 / y15, **y36** (DS)
Number of VMs	y29, **y37**	**y37**	y29, **y37**	**y29**
User Arrival Rate	**y4**	**y4**	**y4**	**y4**, y37
Reallocation Rate	**y7**, y22	y7, **y22**	**y7** (cluster) / **y22** (node)	y7, **y22**
Variance in Choice of Cluster	**y28**	**y28**	**y28**	**y28**

*Not possible for cloud-wide resource usage in SA2-small, so we selected response with highest mean correlation.

Main Effects Analysis

Main Effects Analysis (MEA) Identifies Significant Influence of Input Parameters on Response Variables

We applied MEA to response variables selected using CAC – this example is ***y15*** (NIC Count Load) for **experiment SA1-small**



Significant Parameters

Most significant parameters determined through MEA of the responses selected using CAC

We computed percent of responses influenced (Ψ) for each parameter, weighting $p < 0.05$ at ½ and $p < 0.01$ at 1:

$$\Psi = (|\{y \mid p < 0.01\}| + ½\,|\{y \mid p < 0.05\}|) / |\{y\}| \times 100$$

Computed average Ψ for each parameter, weighting experiment Ψ by number of repetitions

		Input Parameter										
Experiment	Weight	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11
SA1 small	6/14	1	57	22	11	44	29	30	12	0	1	0
SA1 large	1/14	0	69	13	25	44	56	31	25	0	13	0
SA2 small	6/14	2	73	38	10	45	62	10	17	1	0	0
SA2 large	1/14	0	56	50	11	39	56	6	11	0	0	0
Avg. Ψ	Est.	1	65	30	12	44	47	20	15	0	1	0

green = major influence; yellow = modest influence; orange = minor influence; gray = no influence

Most significant parameters: *x2* (# users), *x5* (# clusters), and *x6* (# nodes/cluster)
Moderately influential parameters: *x3* (user types) and *x7* (platform types)
Somewhat influential parameters: *x4* (user hold time) and *x8* (cluster-selection algorithm)
No influence : *x1* (measurement interval), *x9* (node-selection algorithm), *x10* (geo-distribution of cloud components), and *x11* (packet loss prob.)

http://www.nist.gov/itl/antd/emergent_behavior.cfm

Comparing VM-Placement Algorithms for On-Demand Clouds

Kevin Mills, James Filliben and Chris Dabrowski from NIST

Proceedings of IEEE CloudCom 2011, Athens, Nov. 29-Dec. 1, 2011

We developed an objective method to compare distributed control algorithms in simulations of large systems. Our method has four steps: (1) develop a reduced-parameter model (i.e., Koala), (2) determine the most significant model behaviors and the parameters that most influence those behaviors, (3) construct a set of parameter combinations under which control algorithms should be compared and (4) use multi-dimensional data analysis techniques to find patterns revealing significant similarities and differences among the algorithms. This work describes steps (3) and (4) as applied to Koala, an IaaS (Infrastructure-as-a-Service) cloud simulator.

VM Placement Algorithms

Criteria for Choosing a Cluster

Identifier	Criterion Name
LLF	Least-Full First
PAL	Percent Allocated
RAN	Random

Heuristics for Choosing Nodes

Identifier	Heuristic Name
FF	First Fit
LF	Least-Full First
MF	Most-Full First
NF	Next Fit
RA	Random
TP	Tag & Pack

User Types

User Type	VM Type(s)	Max-Min VMs	Max-Max VMs	User Type	VM Type(s)	Max-Min VMs	Max-Max VMs
PU1	M1 small	10	100	PS1	C1 medium	3	10
				PS2		10	50
PU3		100	500	PS3		50	100
PU5		500	1000	WS1	M1 large M2 xlarge C1 xlarge	1	3
PU2	M1 large	10	100	WS2	M1 large M2 xlarge C1 xlarge	3	9
PU4		100	500	WS3	M1 large M2 xlarge C1 xlarge	9	12
PU6		500	1000	DS1	M4 xlarge	10	100
MS1	M1 xlarge	10	100	DS2		100	500
MS3	M1 xlarge	100	500	DS3		500	1000

VM Types

VM Type	Virtual Cores #	Virtual Cores Speed (GHz)	Virtual Block Devices #	Virtual Block Devices Size (GB) of Each	# Virtual Network Interfaces	Memory (GB)	Instruct. Arch.	Price in $/Hour
M1 small	1	1.7	1	160	1	2	32-bit	0.12
M1 large	2	2	2	420	2	8	64-bit	0.34
M1 xlarge	4	2	4	420	2	16	64-bit	0.96
C1 medium	2	2.4	1	340	1	2	32-bit	0.17
C1 xlarge	8	2.4	4	420	2	8	64-bit	0.68
M2 xlarge	8	3	1	840	2	32	64-bit	1.00
M4 xlarge	8	3	2	850	2	64	64-bit	2.00

Platform Type	Physical Cores #	Physical Cores Speed (GHz)	Memory (GB)	# Physical Disks by Size 250 GB	500 GB	750 GB	1000 GB	# Network Interfaces	Instruct. Arch.
C8	2	2.4	32	0	3	0	0	1	64-bit
C14	4	3	64	0	4	0	3	2	64-bit
C18	8	3	128	0	0	4	3	4	64-bit
C22	16	3	256	0	0	0	7	4	64-bit

Platform Types

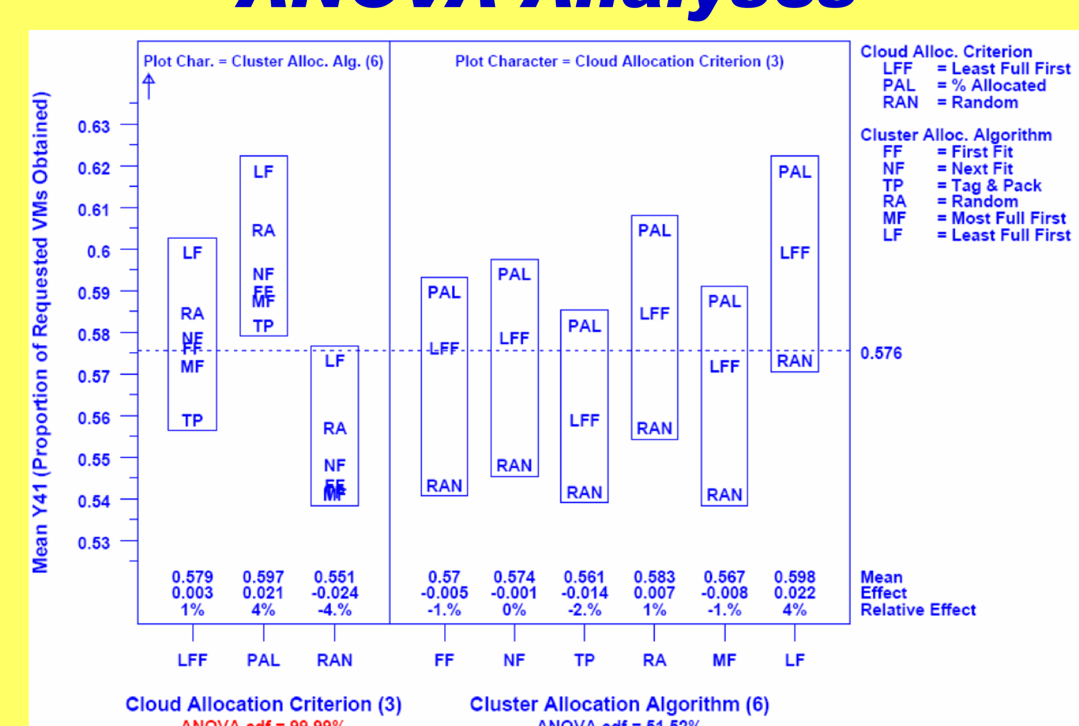
Experiment Design

Layer	Parameter	Parameter Name	Plus (1) Level	Minus (-1) Level
Demand Layer	*x1*	Number of users	2500	250
	x2	Probability of a user's type	PU1 = 0.20 PU2 = 0.20 PU3 = 0.10 PU4 = 0.10 MS1 = 0.10 MS3 = 0.01 PS1 = 0.10 PS2 = 0.01 WS1 = 0.15 WS2 = 0.07 WS3 = 0.03 DS1 = 0.10 DS2 = 0.01	PU1 = 1/6 PU2 = 1/6 MS1 = 1/6 PS1 = 1/6 WS1 = 1/6 DS1 = 1/6
	x3	Average (& shape) of user's holding time	8 hours (a = 1.2)	4 hours (a = 1.2)
Supply Layer	*x4*	Number of clusters	20	10
	x5	Number of nodes per cluster	1000	100
	x6	Probability of a node's platform configuration type	C22 = 1.0	C8 = 0.25 C14 = 0.25 C18 = 0.25 C22 = 0.25

Responses Evaluated

Category	ID	Response Name	Definition
User	*y1*	User Request Rate	(Requests by All Users / # User Cycles)
	y2	NERA Rate	(NERAs / Requests by All Users)
	y3	Full Grant Rate	(Full Grants / (Full Grants + Partial Grants))
	y4	User Arrival Rate	(# User Cycles / Simulated Hours)
	y5	User Give-up Rate	(# Users that Gave Up / # User Cycles)
	y6	Grant Latency	Weighted Avg. Delay in Granting VMs to Users that Got VMs
	y40	User Success Rate	((Full Grants + Partial Grants)/# User Cycles)
	y41	Avg. Fraction VMs Obtained	(Allocated VMs/Requested VMs)
	y42	Avg. *RunInstance* Response Time	Weighted avg. for successful allocations
Cloud	*y7*	Reallocation Rate	(# Times Alternate Cluster Chosen / Requests Granted)
	y8	Full Grant Proportion	(Avg. Fraction Clusters Offering Full Grants)
	y9	NERA Proportion	(Avg. Fraction Clusters Reporting NERA)
	y10	vCore Utilization	(Avg. Fraction of Virtual Cores Used in Cloud)
	y11	Memory Utilization	(Avg. Fraction of Memory in Use in Cloud)
	y12	Disk Space Utilization	(Avg. Fraction of Disk Space in Use in Cloud)
	y13	pCore Load	(Avg. Virtual Cores Allocated / Physical Cores in Cloud)
	y14	Disk Count Load	(Avg. Virtual Disks Allocated / Physical Disks in Cloud)
	y15	NIC Count Load	(Avg. Virtual NICs Allocated / Physical NICs in Cloud)
Cluster	*y16*	vCore Utilization Variance	Avg. Variance in vCore Utilization across Clusters
	y17	Memory Utilization Variance	Avg. Variance in Memory Utilization across Clusters
	y18	Disk Space Utilization Variance	Avg. Variance in Disk Space Utilization across Clusters
	y19	pCore Load Variance	Avg. Variance in pCore Load across Clusters
	y20	Disk Count Variance	Avg. Variance in Disk Count Load across Clusters
	y21	NIC Count Variance	Avg. Variance in NIC Count Load across Clusters
	y22	Node Reallocation Rate	(# Times Alternate Node Chosen / VMs Allocated)
	y23	Cluster NERA Rate	(# NERAs / # Responses Avg. across Clusters)
	y24	Cluster Full-Grant Rate	(# Full Grants / # Responses Avg. across Clusters)
	y25	Allocation Rate	(Times Cluster chosen / Cluster offered Avg. across Clusters)
	y26	Standard Deviation-NERA	Stand. Dev. in Avg. NERA Rate across Clusters
	y27	Standard Deviation-Full-Grant	Stand. Dev. in Avg. Full-Grant Rate across Clusters
	y28	Standard Deviation-Allocation Rate	Stand. Dev. in Allocation Rate across Clusters
VMs	*y29*	Current Instances	Avg. # VM Instances Extant in Cloud
	y30	M1small Instances	Fraction of Current Instances that are M1 small VMs
	y31	M1large Instances	Fraction of Current Instances that are M1 large VMs
	y32	M1xlarge Instances	Fraction of Current Instances that are M1 xlarge VMs
	y33	C1medium Instances	Fraction of Current Instances that are C1 medium VMs
	y34	C1xlarge Instances	Fraction of Current Instances that are C1 xlarge VMs
	y35	M2xlarge Instances	Fraction of Current Instances that are M2 xlarge VMs
	y36	M4xlarge Instances	Fraction of Current Instances that are M4 xlarge VMs
Internet/ Intranet	*y37*	WS Message Rate	Avg. # WS Messages Send Per Simulated Hour
	y38	Intra-Site Messages	(# WS Messages Sent with Sites / # WS Messages Sent)
Revenue	*y39*	Aggregate Revenue in $/Hour	Calculated from *y29* through *y36* & VM prices

ANOVA Analyses



$$F = \frac{f_2}{f_1} \cdot \frac{\sum_{i=1}^{3}\sum_{j=1}^{6}\sum_{k=1}^{32}(x_{ijk} - \bar{x})^2}{\sum_{i=1}^{3}\sum_{j=1}^{6}\sum_{k=1}^{32}(x_{ijk} - \overline{x_k})^2}$$

Summary of 84 ANOVA Tests

Category	ID	Response Name	ANOVA Cdf Cloud Crit (3)	ANOVA Cdf Cluster Alg (6)
User	*y1*	User Request Rate	99.96	62.19
	y2	NERA Rate	100	22.33
	y3	Full Grant Rate	100	2.75
	y4	User Arrival Rate	99.87	77.15
	y5	User Give-up Rate	94.63	98.6
	y6	Grant Latency	98.01	96.11
	y40	User Success Rate	95.86	98.02
	y41	Avg. Fraction VMs Obtained	99.99	51.52
	y42	Avg. *RunInstance* Response Time	37.35	97.49
Cloud	*y7*	Reallocation Rate	99.99	9.5
	y8	Full Grant Proportion	100	0.02
	y9	NERA Proportion	100	0.4
	y10	vCore Utilization	67.85	99.81
	y11	Memory Utilization	98.97	91.47
	y12	Disk Space Utilization	97.29	96.27
	y13	pCore Load	67.85	99.81
	y14	Disk Count Load	96.76	97.56
	y15	NIC Count Load	99.78	79.49
Cluster	*y16*	vCore Utilization Variance	100	1.28
	y17	Memory Utilization Variance	100	0.09
	y18	Disk Space Utilization Variance	100	0.14
	y19	pCore Load Variance	100	1.28
	y20	Disk Count Variance	100	0.42
	y21	NIC Count Variance	100	1.02
	y22	Node Reallocation Rate	100	6.09
	y23	Cluster NERA Rate	100	0.19
	y24	Cluster Full-Grant Rate	100	0.06
	y25	Allocation Rate	99.88	77.64
	y26	Standard Deviation-NERA	63.92	61.08
	y27	Standard Deviation-Full-Grant	99.73	30.95
	y28	Standard Deviation-Allocation Rate	100	0.02
VMs	*y29*	Current Instances	99.98	50.54
	y30	M1small Instances	99.99	35.85
	y31	M1large Instances	60.58	99.02
	y32	M1xlarge Instances	99.83	77.1
	y33	C1medium Instances	99.97	27.57
	y34	C1xlarge Instances	82.1	99.89
	y35	M2xlarge Instances	74.62	99.97
	y36	M4xlarge Instances	99.95	66.03
Internet/ Intranet	*y37*	WS Message Rate	99.7	83.74
	y38	Intra-Site Messages	89	99.05
Revenue	*y39*	Aggregate Revenue in $/Hour	99.99	44.51

Means Per Cluster-Choice

Category	ID	LLF	PAL	RAN
User	*y1*	7.461	8.386	7.696
	y2	0.444	0.506	0.450
	y3	0.624	0.574	0.514
	y4	37324	35878	37170
	y5	0.066	0.074	0.067
	y6	9044	10488	9526
	y40	0.925	0.915	0.923
	y41	0.579	0.597	0.551
	y42	0.278	0.277	0.278
Cloud	*y7*	0.000052	0.000084	0.000057
	y8	0.438	0.332	0.389
	y9	0.481	0.587	0.537
	y10	0.774	0.791	0.783
	y11	0.188	0.197	0.199
	y12	0.413	0.428	0.418
	y13	0.774	0.791	0.783
	y14	0.964	0.997	0.948
	y15	1.591	1.645	1.554
Cluster	*y16*	0.0017	0.019	0.0071
	y17	0.0009	0.0034	0.0015
	y18	0.0022	0.0086	0.0038
	y19	0.0017	0.019	0.0071
	y20	0.018	0.052	0.024
	y21	0.045	0.127	0.052
	y22	0.00015	0.00015	0.00008
	y23	0.507	0.606	0.562
	y24	0.421	0.323	0.375
	y25	0.19	0.232	0.232
	y26	0.01	0.01	0.011
	y27	0.008	0.011	0.015
	y28	0.034	0.058	0.02
VMs	*y29*	21808	22139	20365
	y30	0.355	0.354	0.333
	y31	0.308	0.311	0.307
	y32	0.138	0.142	0.151
	y33	0.057	0.053	0.052
	y34	0.025	0.022	0.025
	y35	0.026	0.023	0.026
	y36	0.091	0.096	0.106
Internet/ Intranet	*y37*	60867	62677	60841
	y38	0.977	0.977	0.977
Revenue	*y39*	11322	11706	11624

Means Per Node-Selection

Category	ID	FF	LF	MF	NF	TP	RA
User	*y1*	7.643	8.450	7.692	7.710	7.871	7.718
	y2	0.460	0.493	0.458	0.462	0.455	0.470
	y3	0.566	0.593	0.563	0.57	0.555	0.577
	y4	37138	35624	37188	36938	37051	36807
	y5	0.065	0.080	0.065	0.067	0.067	0.069
	y6	10130	8636	10439	9643	10420	8848
	y40	0.925	0.908	0.925	0.923	0.922	0.921
	y41	0.57	0.598	0.567	0.574	0.561	0.583
	y42	0.278	0.276	0.278	0.279	0.277	0.278
Cloud	*y7*	0.000063	0.000064	0.000068	0.000073	0.000055	0.000063
	y8	0.387	0.387	0.378	0.389	0.385	0.39
	y9	0.529	0.55	0.536	0.528	0.536	0.532
	y10	0.789	0.761	0.812	0.786	0.764	0.78
	y11	0.198	0.188	0.204	0.196	0.191	0.193
	y12	0.419	0.428	0.424	0.421	0.402	0.424
	y13	0.789	0.761	0.812	0.786	0.764	0.78
	y14	0.958	1.013	0.958	0.97	0.928	0.99
	y15	1.58	1.639	1.597	1.592	1.542	1.631
Cluster	*y16*	0.0085	0.008	0.0127	0.0097	0.008	0.008
	y17	0.0019	0.0020	0.0022	0.0019	0.0019	0.0017
	y18	0.0045	0.0054	0.0053	0.0050	0.0046	0.0045
	y19	0.0085	0.0089	0.0127	0.0097	0.0080	0.0080
	y20	0.029	0.036	0.032	0.032	0.029	0.029
	y21	0.067	0.088	0.080	0.074	0.065	0.073
	y22	0.00013	0.00012	0.00013	0.00014	0.00011	0.00012
	y23	0.555	0.569	0.562	0.552	0.558	0.553
	y24	0.373	0.375	0.364	0.376	0.373	0.378
	y25	0.228	0.192	0.237	0.216	0.232	0.201
	y26	0.011	0.009	0.013	0.010	0.010	0.009
	y27	0.012	0.010	0.015	0.011	0.012	0.010
	y28	0.037	0.040	0.037	0.037	0.035	0.038
VMs	*y29*	21237	22244	21020	21409	20824	21888
	y30	0.344	0.356	0.342	0.348	0.341	0.352
	y31	0.306	0.315	0.304	0.305	0.311	0.312
	y32	0.144	0.149	0.145	0.147	0.135	0.142
	y33	0.054	0.053	0.053	0.053	0.056	0.054
	y34	0.025	0.018	0.026	0.024	0.027	0.022
	y35	0.027	0.019	0.028	0.026	0.029	0.023
	y36	0.100	0.090	0.103	0.097	0.101	0.095
Internet/ Intranet	*y37*	61018	63016	61223	61156	60571	61785
	y38	0.977	0.977	0.977	0.977	0.976	0.977
Revenue	*y39*	11603	11529	11683	11587	11362	11541

http://www.nist.gov/itl/antd/emergent_behavior.cfm

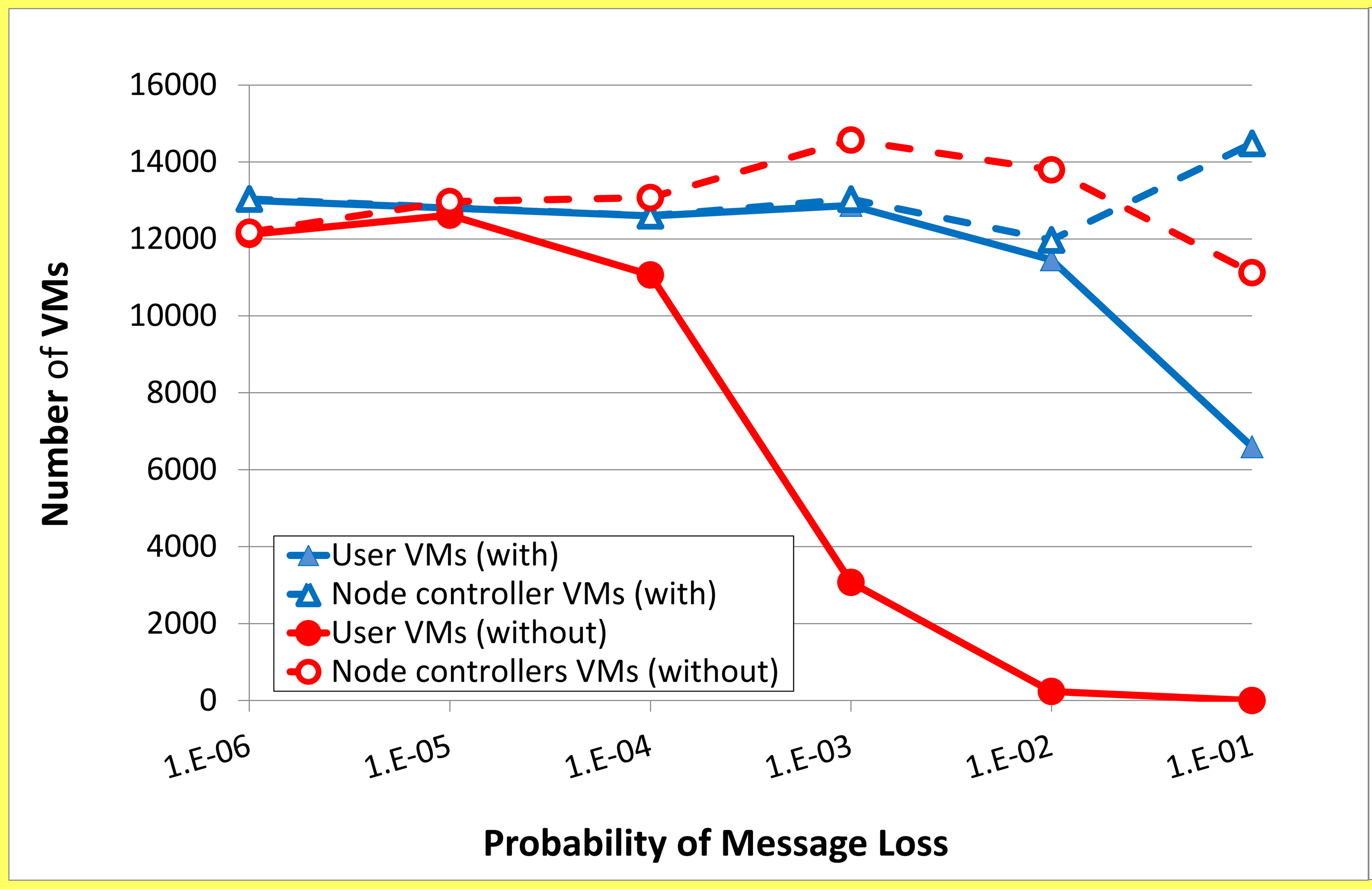
VM Leakage and Orphan Control in Open-Source Clouds

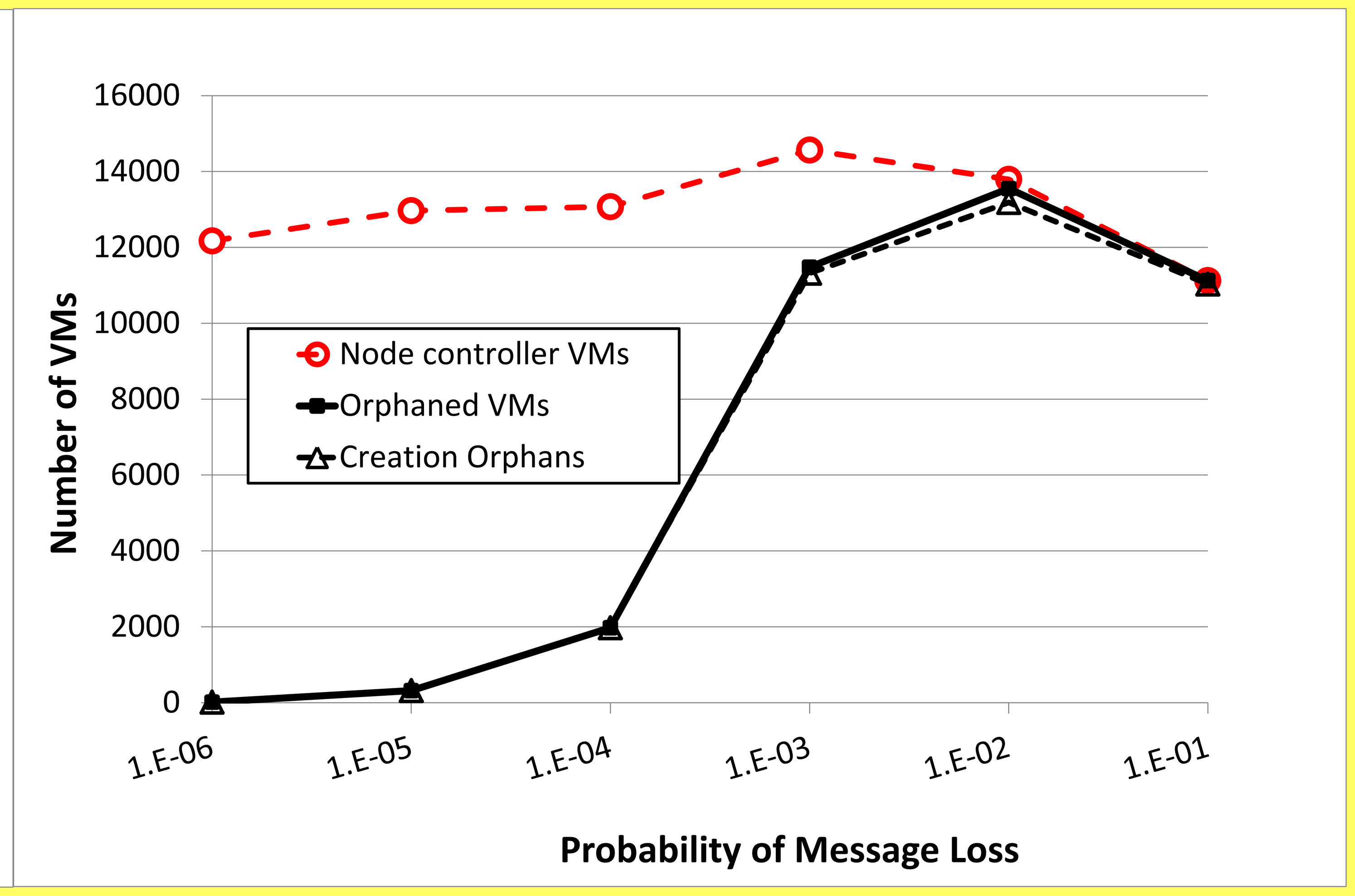
Chris Dabrowski & Kevin Mills from NIST

Proceedings of IEEE CloudCom 2011, Athens, Nov. 29-Dec. 1, 2011

A simple message discard attack initiated by a Trojan in compromised Web-server code leads to VM Leakage that exhausts resources in an IaaS cloud based on commonly available open-source code. Adding two orphan control processes, (1) creation orphan control and (2) persistent termination, mitigates the VM Leakage.

VMs as seen by cloud users and cloud nodes without (red) and with (blue) orphan control processes—the gaps denote VM leakage

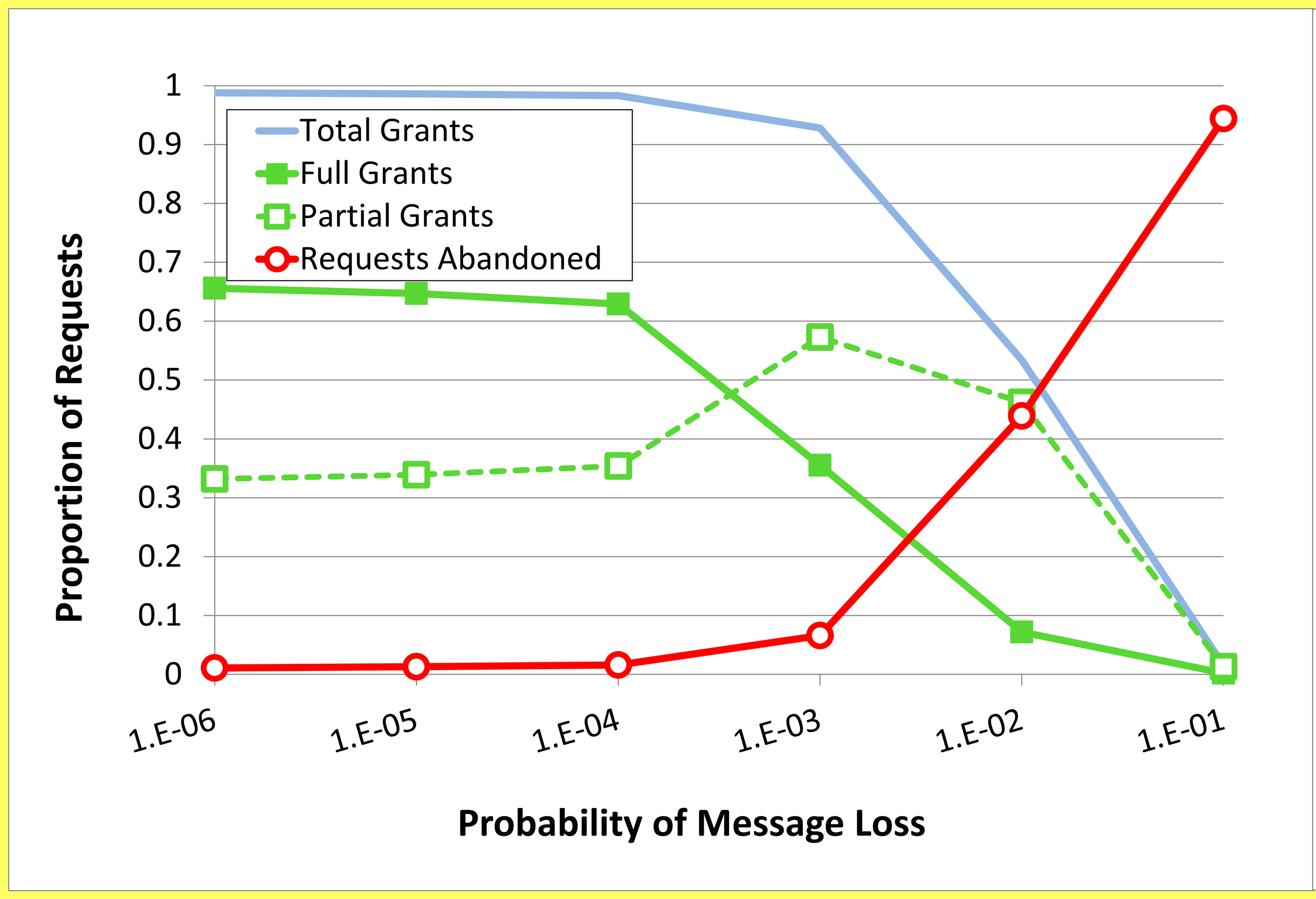


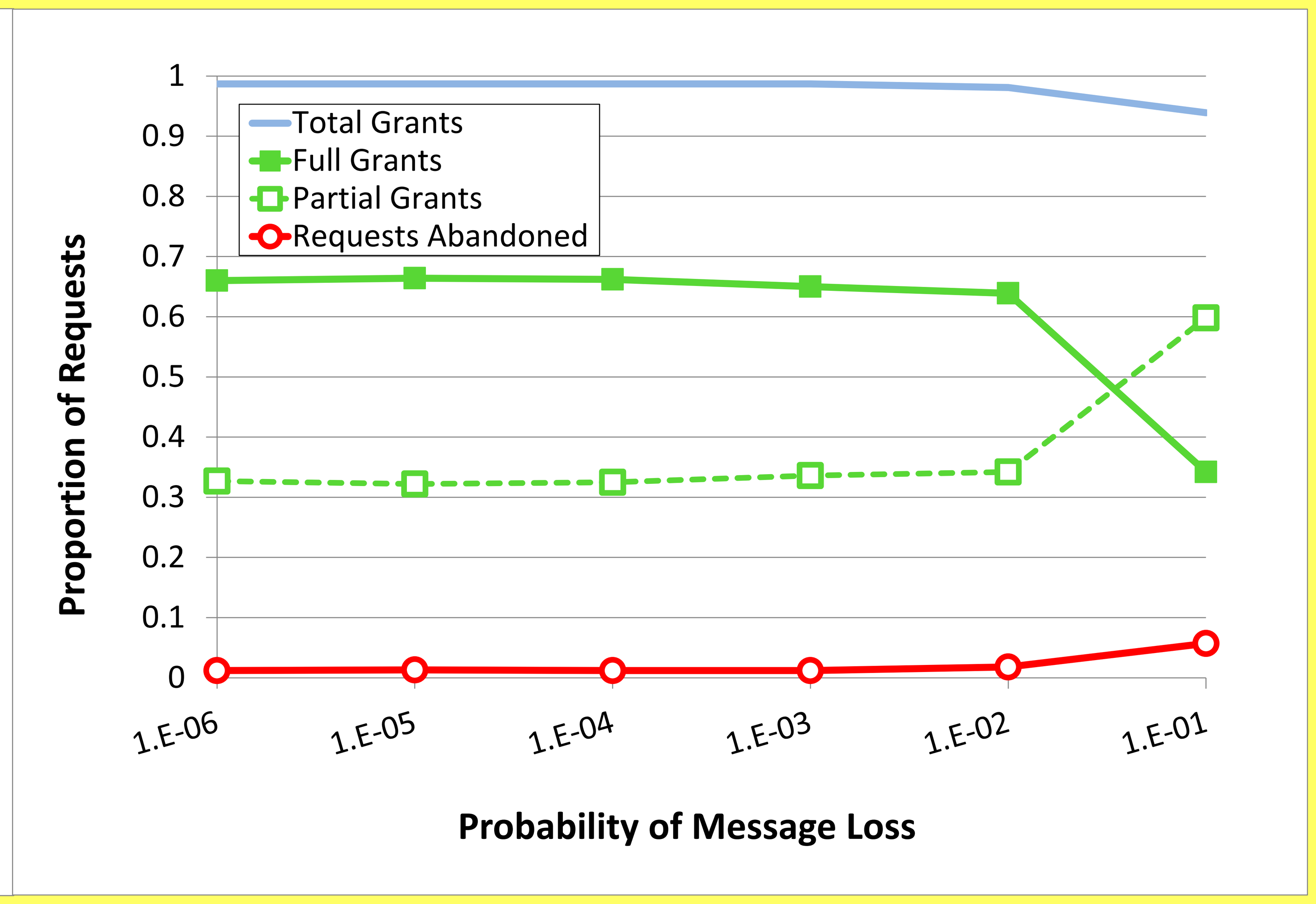


Leaked VMs without orphan control processes—most leaked VMs are creation orphans

Without orphan control processes, most leaked VMs are creation orphans because VMs must be created before they can become orphans and user retries

Outcomes of User Requests without orphan control processes–Total Grants (categorized by Full and Partial) and Un-served Users





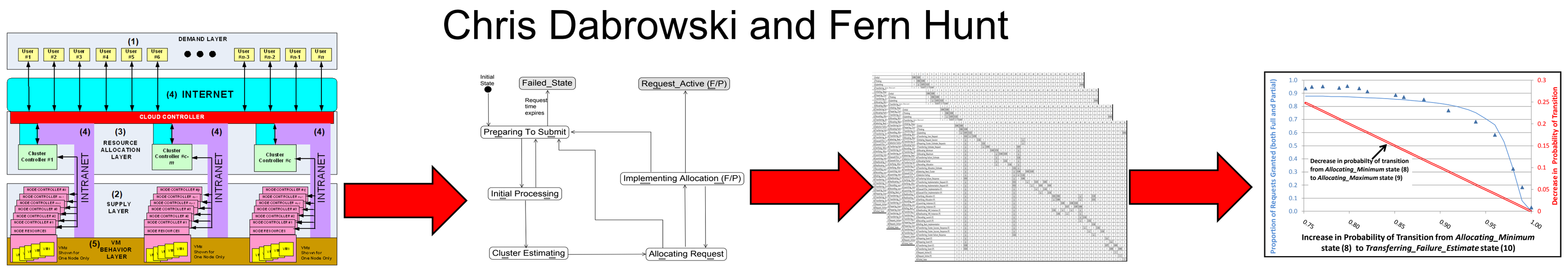
Outcomes of User Requests with orphan control processes–Total Grants (categorized by Full and Partial) and Un-served Users

Orphan control processes do not eradicate all leaked VMs at highest message loss rate due to existence of temporary termination orphans that occur because we limited persistent termination to final termination requests

http://www.nist.gov/itl/antd/emergent_behavior.cfm

IDENTIFY FAILURE SCENARIOS IN CLOUD SYSTEMS USING MARKOV CHAIN ANALYSIS

Chris Dabrowski and Fern Hunt



Problem: Identifying failure scenarios in distributed systems such as clouds is critical to understanding areas where performance may degrade. However, potential failure scenarios may be numerous and difficult to find.

Objective: To perturb Discrete Time Markov Chains (DTMCs) of cloud system behavior to identify potential failure scenarios more quickly than through detailed large-scale simulation or use of test beds.

Steps:

(1) Using *Koala* as proxy for real-world cloud, develop detailed state model of cloud behavior and convert to time-inhomogeneous DTMC.

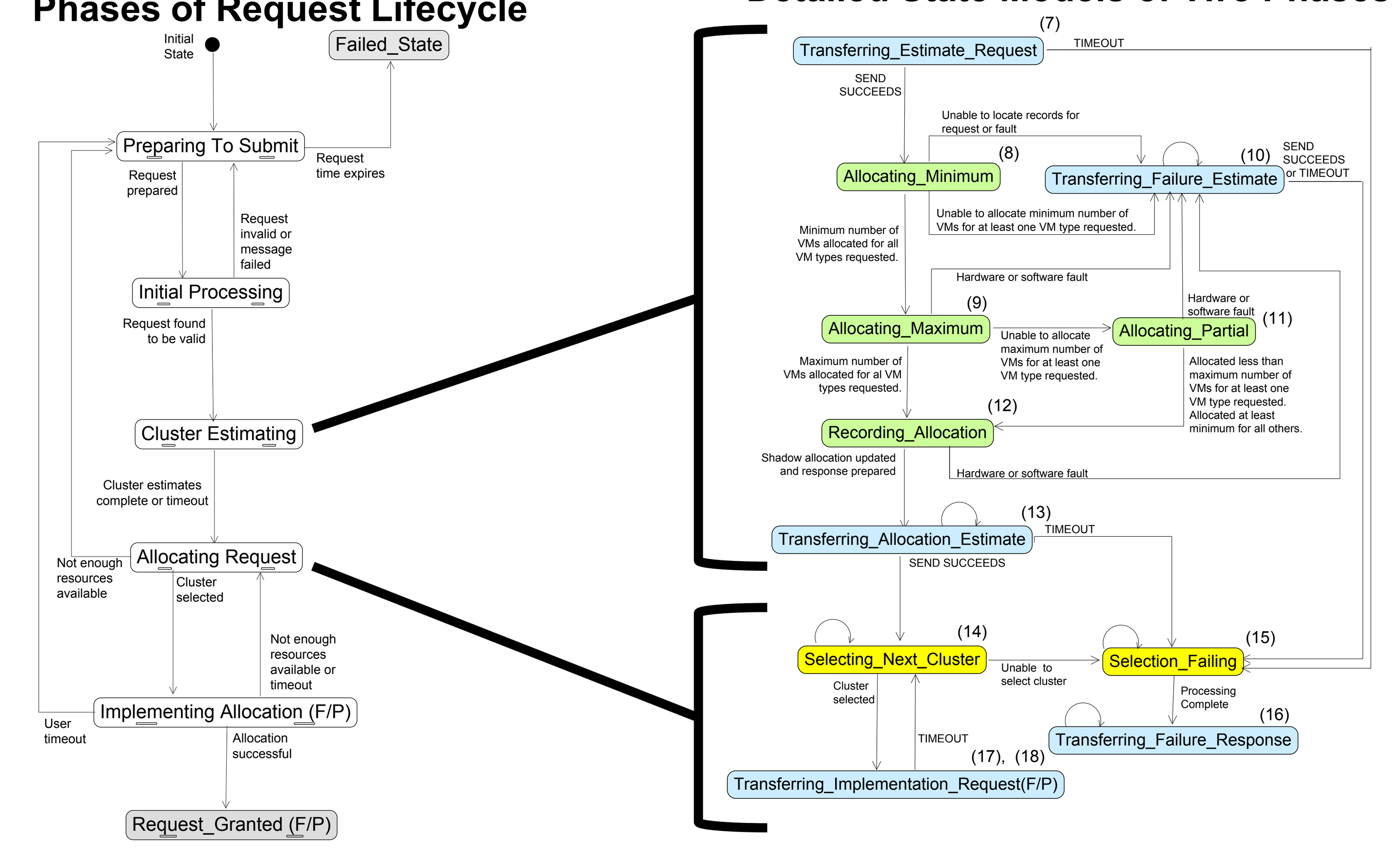
(2) Find minimal s-t cut sets in a directed graph of cloud DTMC to identify critical state transitions that break paths to desirable system goal states.

(3) Perturb critical state transitions to describe potential failure scenarios, create predictive performance curves, and find performance thresholds.

State Model of Resource Request in Cloud

A detailed representation of the states that a cloud system (*Koala*) may enter under normal and failure conditions, shown for two of the five major phases.

High-Level Model of Phases of Request Lifecycle | **Detailed State Models of Two Phases**

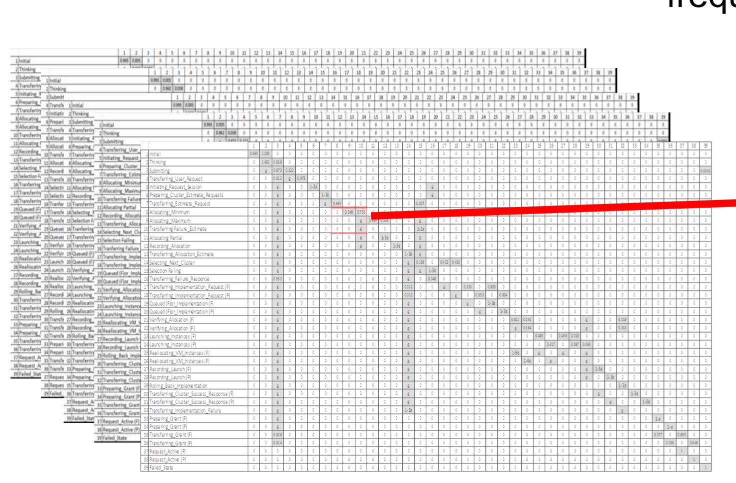


Creating a Discrete Time Markov Chain

- Observe *Koala* (as proxy for real-world system) to derive set of transition probability matrices (TPMs) that describe probabilities of transition between states over different time periods → form a time-inhomogeneous DTMC.
- Generate1000 time period TPMs of 3600 s each.

$$p_{ij} = \frac{f_{ij}}{\sum_{k=1}^{n} f_{ik}}$$

Given states s_i, s_j, $i, j = 1\ldots n$ where $n = 39$, p_{ij}, is the probability of transitioning from state i to state j, written as $s_i \rightarrow s_j$. This probability is estimated by calculating the frequency of $s_i \rightarrow s_j$, or f_{ij}, divided by the sum of the frequencies of s_i to all other states.



		8	9	10
8	Allocating_Minimum	0	0.248	0.752
9	Allocating_Maximum	0	0	ε
10	Transferring Failure_Estimate	0	0	ε

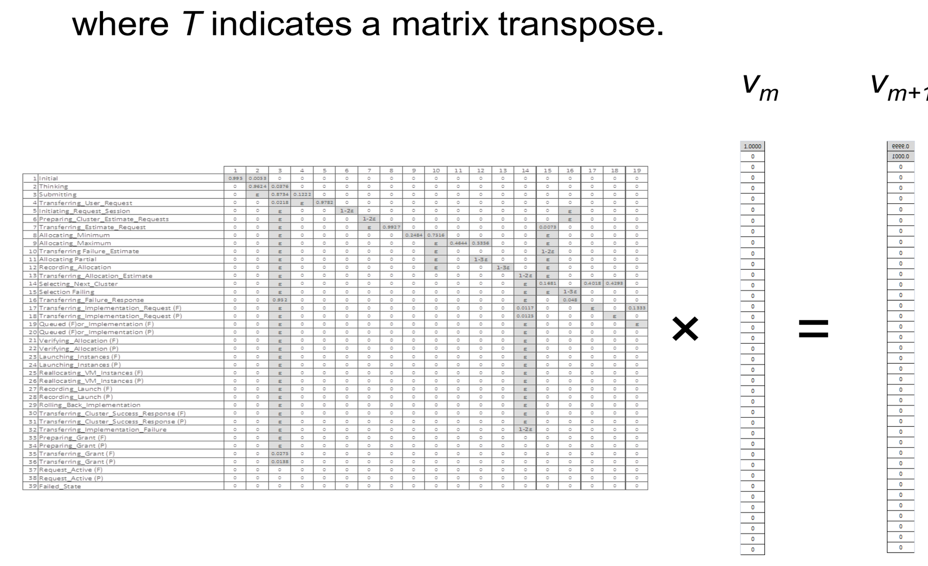
Using the DTMC to simulate large-scale system (*Koala*) behavior

- Markov chains can emulate *Koala* to capture high-level system behavior, but in two orders of magnitude less computational time.

To evolve system state in discrete time steps, multiply state vector v_m (at time step m) by the TPM, Q^{tp}, for the applicable time period tp to produce a new system state vector v_{m+1},

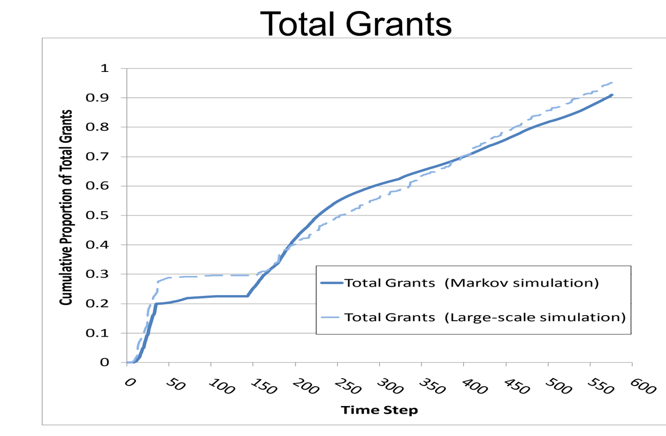
$$(Q^{tp})^T * v_m = v_{m+1}, \text{ where } tp = \textit{integral value}\,(m/S) + 1$$

where T indicates a matrix transpose.

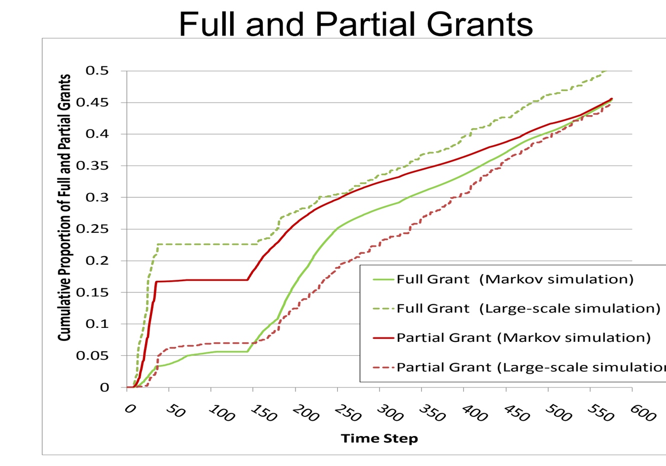


Repeated for 576 time steps in 16 hour simulated period, one time period per hour.

Total Grants



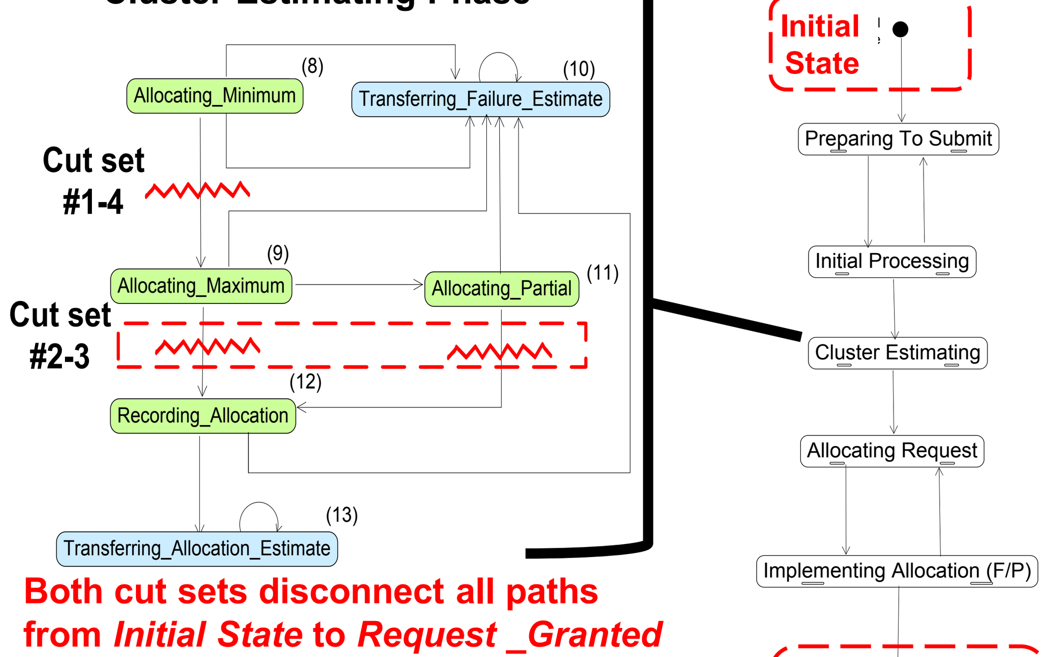
Full and Partial Grants



Using minimal s-t cut set analysis to find potential failure scenarios

- In a directed graph of the *Koala* DTMC, minimal s-t cut sets consist of *critical state transitions*, which if removed, disconnect all paths to absorbing *Requests_Granted* (*F/P*) state.
- Applying algorithm to find minimal s-t cut sets* to the *Koala* DTMC resulted in 159 cut sets. Examples of one and two-transition cut sets are shown.

Detailed Model of Cluster Estimating Phase | **High-Level Model of Request Lifecycle**



Both cut sets disconnect all paths from *Initial State* to *Request _Granted* (*F/P*) absorbing states.

One-transition cut sets

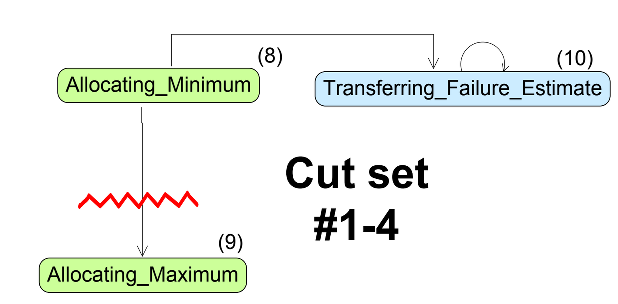
	Set of member transitions from Fig. 3	Total Probabilty
1-1	{1, 2}	0.001
1-2	{2, 3}	0.025
1-3	{3, 4}	0.124
1-4	{8, 9}	0.264
.....	....	....
1-10	{12, 13}	1.000

Two-transition cut sets

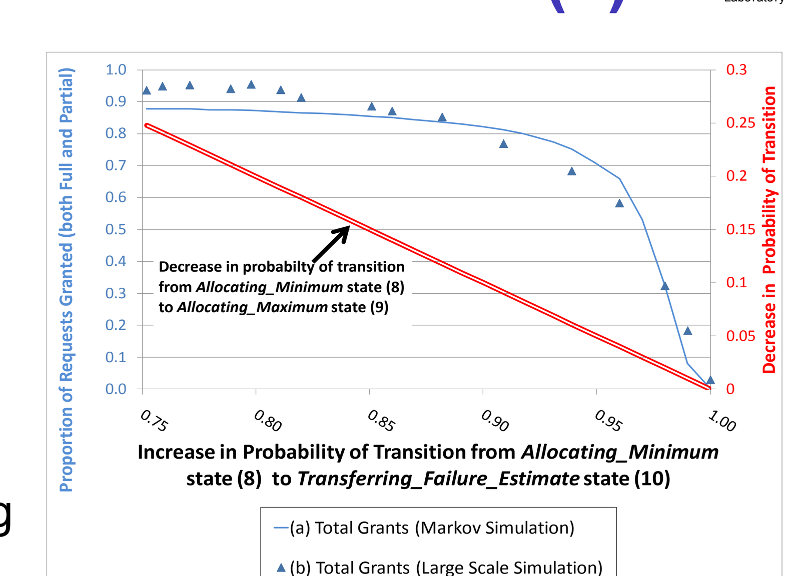
	Set of member transitions from Fig. 3	Number of From States	Total Probabilty
2-1	{14, 17} {14, 18}	1	0.895
2-2	{9, 11} {9, 12}	1	1.000
2-3	{9, 12} {11, 12}	2	1.395
.....	....	..	....
2-23	{33, 35} {34, 36}	2	2.000

*Provan S., and Ball M., 1984, "Computing Network Reliability in Time Polynomial in the Number of Cuts," *Operations Research*, 32(3), pp. 516–526.

Perturbing state transitions in a cut set to predict system behavior in failure scenario (1)



- Cut set #1-4 could relate to a scenario in which software or hardware failures make resource databases inaccessible, preventing clusters from computing minimum allocation estimates. Instead, clusters return failure estimates to the cloud controller.



Decline in total requests granted (Full and Partial) due to cluster estimation failure:
(a) As estimated by perturbing the DTMC; and
(b) As computed in *Koala* large-scale simulation.

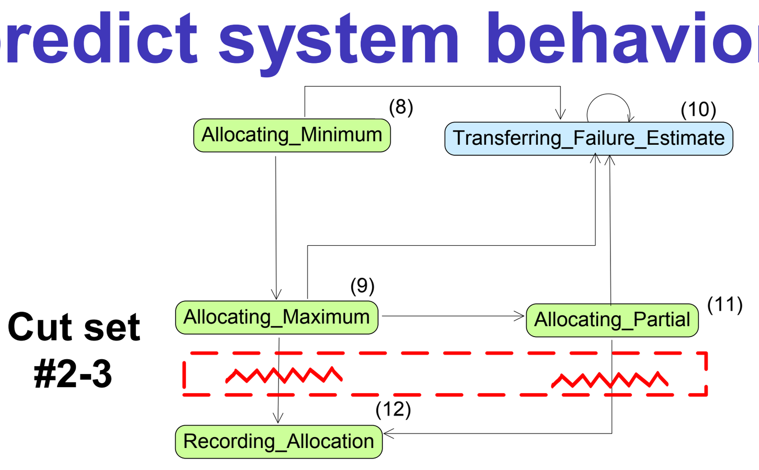
Blue curves show the resulting decrease in requests granted as estimated using the DTMC and as actually occurred in the *Koala* large-scale simulation. These curves are plotted against the left vertical axis. The right vertical axis provides units for the decrease in probability of the state transition.

Portions of TPM perturbed

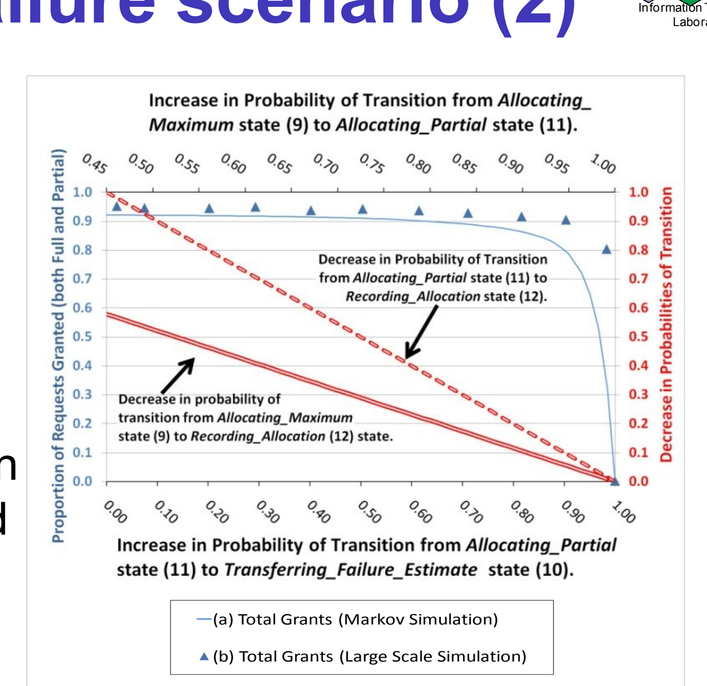
		8	9	10
8	Allocating_Minimum	0	0.248↓	0.752↑
9	Allocating_Maximum	0	0	ε
10	Transferring Failure_Estimate	0	0	ε

- Raise probabiltiy of *Allocating_Minimum* → *Transferring_Failure_Estimate*:TPM element {8, 10}
- Lower probablity of *Allocating_Minimum* → *Allocating_Maximum:* TPM elements {8, 9}.

Perturbing state transitions in a cut set to predict system behavior in failure scenario (2)



- Cut set #2-3 could relate to a failure scenario in which viruses or other faults cause widespread software process failures in clusters, which prevent completion of cluster allocation estimation computations. Instead, clusters return failure estimates to the controller.



Decline in total requests granted (Full and Partial) due to cluster estimation failure:
(a) As estimated by perturbing the DTMC; and
(b) As computed by *Koala* large-scale simulation.

Blue curves show the resulting decrease in requests granted as estimated using the DTMC and as actually occurred in the *Koala* large-scale simulation. These curves are plotted against the left vertical axis. The right vertical axis provides units for the decrease in probability of the state transition.

Portions of TPM perturbed

		9	10	11	12
9	Allocating_Maximum	0	ε	0.464↑	0.536↓
10	Transferring Failure_Estimate	0	ε	0.000	0.000
11	Allocating Partial	0	ε↑	0.000	1-3ε↓
12	Recording_Allocation	0	ε	0.000	0.000

•Raise *Allocating_ Maximum* → *Allocating_Partial*: TPM element {9, 11}
•Lower*Allocating_Maximum* → *Recording_ Allocation* (TPM element {9, 12})

•Raise *Allocating_Partial*→*Transferring_Failure_Estimate:* TPM element {11, 10}
•Lower *Allocating_Partial* → *Recording_Allocation*: TPM element {11, 12}

Ongoing Work

Apply methodology to larger problems and determine scalability

- Current model consists of 39 states and 139 transitions
- Includes user, cloud controller, and cluster behavior, but not node behavior or actual use of VMs

Apply methodology to different types of failure scenarios

For more information, see:

- Identifying Failure Scenarios in Complex Systems by Perturbing Markov Chain Models, by Christopher Dabrowski and Fern Hunt, *Proceedings of the 2011 Pressure Vessels and Piping Division Conference*. July 17–21, Baltimore, MD.
- Using Markov Chains and Graph Theory Concepts to Analyze Behavior in Complex Distributed Systems, by Christopher Dabrowski and Fern Hunt, *Proceedings of the 23rd European Modeling and Simulation Symposium*, September 12-14, 2011. Rome Italy.

Predicting Catastrophic Failure Scenarios in Cloud Systems (*Proposed Research*)

C. Dabrowski, J. Filliben, D. Genin, F. Hunt, K. Mills and S. Ressler from NIST

No effective methods exist to predict failure regimes in large distributed systems—the search space is large and causality is difficult to establish. Today, system operators wait for failures and diagnose, react and mitigate. We propose to leverage models and methods from the physical sciences to predict unforeseen dynamics and failure scenarios that could lead to spatiotemporal collapse in designs and deployments of large distributed systems. We intend to investigate and demonstrate our methods in the context of IaaS (Infrastructure-as-a-Service) clouds. Our work aims to improve cloud computing reliability, benefiting designers of distributed systems running on clouds, and those deploying clouds.

First Hard Problem

Complex information systems encompass an infeasible search space.

$$\underbrace{y_1, \ldots, y_m}_{\text{System Response Space}} = f(\underbrace{x_{1|[1,\ldots,k]}, \ldots, x_{n|[1,\ldots,k]}}_{\text{System Parameter Space}})$$

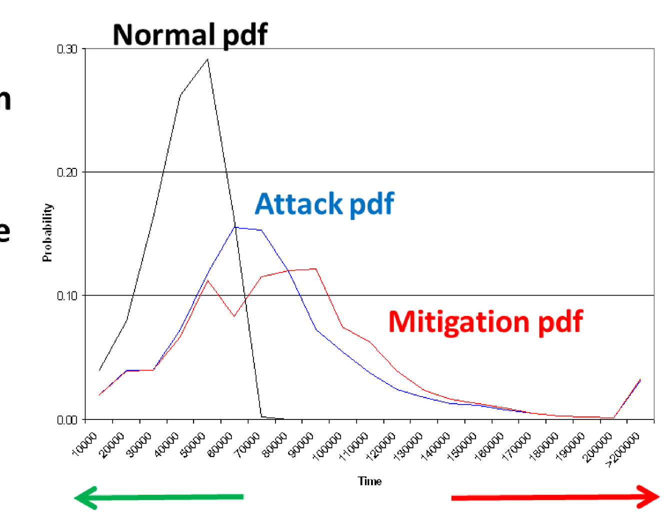
System Parameter Space = k^x, e.g., for k of 2^{32} and x of 1000, = $(2^{32})^{1000}$ = $O(10^{9633})$

Atoms in the visible universe = about 10^{80}

Second Hard Problem

Determining causality is difficult given only patterns of global system behavior.

For example, unexpected collapse in the mitigation probability density function of job completion times in a computing grid was unexplainable without more detailed data and analysis.



Possible Approaches

Plausible approaches to investigate:

- **Anti-Optimization + Directed Search** (e.g., genetic algorithms, simulated annealing, evolutionary strategies)
- **Markov Models + Graph & Perturbation Analyses** (e.g., transform system models into Markov models and use graph theory to find cut-sets and perturbation to explore failure trajectories)

Possible Research Team

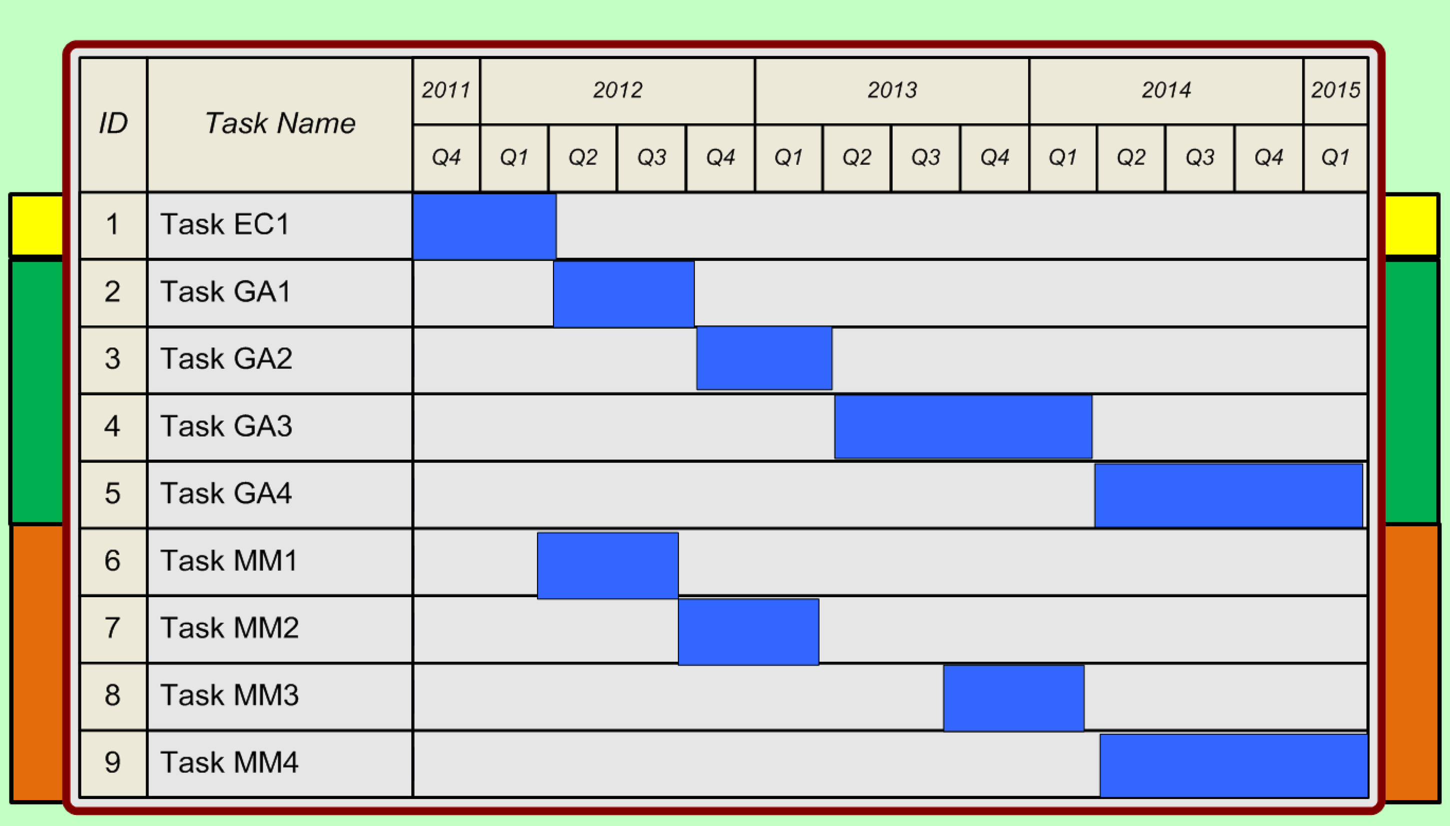
Great NIST team! Experienced in modeling and analyzing complex systems:

- Kevin Mills [PhD] *(Senior Research Scientist – Simulation & Genetic Algorithm)*
- Christopher Dabrowski *(Computer Scientist – Markov models & Graph Theory)*
- Fern Hunt [PhD] *(Mathematician – Markov Models and Eigenanalysis)*
- James Filliben [PhD] *(Statistician – Exploratory Data Analysis)*
- Sandy Ressler *(Computer Scientist – Information Visualization)*
- Daniel Genin [PhD] *(Mathematician – Analytical Models)*

Proposed Tasks

- Existing Commitments (EC1) – comparing virtual machine placement algorithms under suboptimal conditions
- Anti-Optimization + Genetic Algorithms (GA)
  - GA1: design/develop GA software for existing Koala simulator
  - GA2: test GA software on a known problem within Koala
  - GA3: design/develop IaaS model that includes recovery behaviors
  - GA4: use GA to explore new simulation for failure scenarios
- Markov Models + Graph & Perturbation Analysis (MM)
  - MM1: design/develop MM software that captures dependencies
  - MM2: test MM software on a known problem within Koala
  - MM3: generate MM from the new IaaS simulator (see GA3)
  - MM4: use graph & perturbation analysis to explore new MM for failure scenarios

Proposed Schedule



http://www.nist.gov/itl/antd/emergent_behavior.cfm