IDENTIFYING FAILURE SCENARIOS IN COMPLEX SYSTEMS BY PERTURBING MARKOV CHAIN MODELS

Christopher Dabrowski Fern Hunt

Information Technology Laboratory
U.S. National Institute of Standards and Technology
Gaithersburg MD

Presented at

2011 Pressure Vessels & Piping Conference (PVP 2011)

Materials and Fabrication (MF-20)

Modeling and Monitoring of Uncertainty in
PVP or Complex Systems/Networks

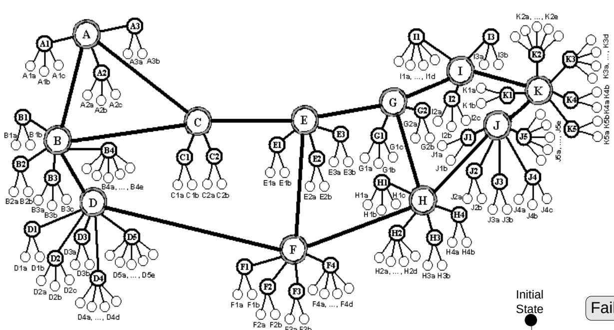
Baltimore, MD
Thursday, July 21, 2011

Summary

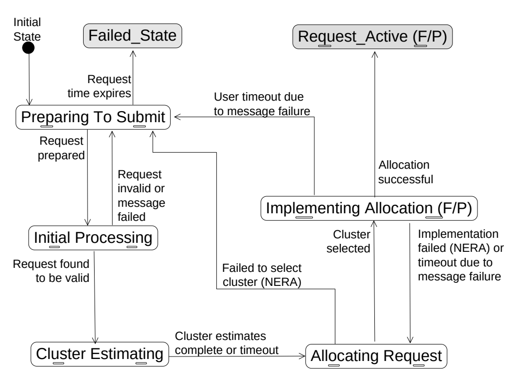
- **Goal:** To develop scalable modeling tools for monitoring real-world complex systems and predicting catastrophic performance degradations.

- **Use *Discrete Time Markov Chain* (*DTMC*):**
  - Develop detailed *time-inhomogeneous* model of system behavior that can represent evolution from normal conditions to failure states.
  - Perturb DTMC *transition probability matrices* (TPMs) to simulate alternative system evolutions.

  → ***Identify failure scenarios***



System to state model



		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
1	Initial	0.995	0.005	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	Thinking	0	0.962	0.038	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	Submitting	0	ε	0.873	0.122	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0001
4	Transferring_User_Request	0	0	0.022	ε	0.978	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	Initiating_Request_Session	0	0	ε	0	0	1-2ε	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	Preparing_Cluster_Estimate_Requests	0	0	ε	0	0	0	1-2ε	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	Transferring_Estimate_Request	0	0	ε	0	0	0	ε	0.993	0	0	0	0	0	0	0.007	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	Allocating_Minimum	0	0	ε	0	0	0	0	0	0.248	0.752	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	Allocating_Maximum	0	0	ε	0	0	0	0	0	0	ε	0.464	0.536	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	Transferring Failure_Estimate	0	0	ε	0	0	0	0	0	0	ε	0	0	0	0	1-2ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	Allocating Partial	0	0	ε	0	0	0	0	0	0	ε	0	1-3ε	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	Recording_Allocation	0	0	ε	0	0	0	0	0	0	ε	0	0	1-3ε	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	Transferring_Allocation_Estimate	0	0	ε	0	0	0	0	0	0	0	0	0	0	1-2ε	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	Selecting_Next_Cluster	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0.168	0	0.402	0.429	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	Selection Failing	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	ε	1-3ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	Transferring_Failure_Response	0	0	0.952	0	0	0	0	0	0	0	0	0	0	ε	0	0.048	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	Transferring_Implementation_Request (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0.012	0	0	ε	0	0.133	0	0.855	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	Transferring_Implementation_Request (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0.012	0	0	0	ε	0	0.053	0	0.934	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	Queued (F)or_Implementation (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	ε	0	1-3ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	Queued (F)or_Implementation (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	ε	0	1-3ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	Verifying_Allocation (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0.821	0.061	0	0	0	0	ε	0	0	0.118	0	0	0	0	0	0	0
22	Verifying_Allocation (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	ε	0.684	0	0	0	0	ε	0	0	0.313	0	0	0	0	0	0	0
23	Launching_Instances (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0.485	0	0.496	0.018	0	0	0	0	0	0	0	0	0	0	0
24	Launching_Instances (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0.317	0	0.587	0.096	0	0	0	0	0	0	0	0	0	0
25	Reallocating_VM_Instances (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	1-5ε	0	ε	0	ε	0	ε	0	0	0	0	0	0	0	0	0	0
26	Reallocating_VM_Instances (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	1-4ε	0	ε	0	0	ε	0	0	0	0	0	0	0	0	0	0
27	Recording_Launch (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	1-3ε	0	0	0	0	0	0	0	0	0
28	Recording_Launch (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	1-3ε	0	0	0	0	0	0	0	0
29	Rolling_Back_Implementation	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1-2ε	0	0	0	0	0	0	0
30	Transferring_Cluster_Success_Response (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	0	1-3ε	0	0	0	0	0	0
31	Transferring_Cluster_Success_Response (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	0	1-3ε	0	0	0	0	0
32	Transferring_Implementation_Failure	0	0	ε	0	0	0	0	0	0	0	0	0	0	1-2ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0
33	Preparing_Grant (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1-ε	0	0	0	0
34	Preparing_Grant (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1-ε	0	0	0
35	Transferring_Grant (F)	0	0	0.028	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.077	0	0.895	0	0
36	Transferring_Grant (P)	0	0	0.014	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.038	0	0.948	0
37	Request_Active (F)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
38	Request_Active (P)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
39	Failed_State	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

State model to Perturbable TPM

Problem and solution approach

- To identify failure scenarios in a complex system, it is advantageous to model more extensive range of possible system states—can lead to large, detailed models

  however, perturbation of large DTMCs may involves search spaces that increase exponentially with model size.

- **Solution approach** (*to be shown*)**:** Use minimal s-t cut set analysis on directed graph of DTMC in combination with other techniques:
  - Detailed DTMC, time-inhomogeneous representation (sets of TPMs for different time periods),
  - Model perturbation and
  - Markov simulation modeling

  in order to…..

  → **Identify small parts of the model – *critical state transitions* – that can be directly perturbed to change system performance.**

  (*thus avoiding large search space*)

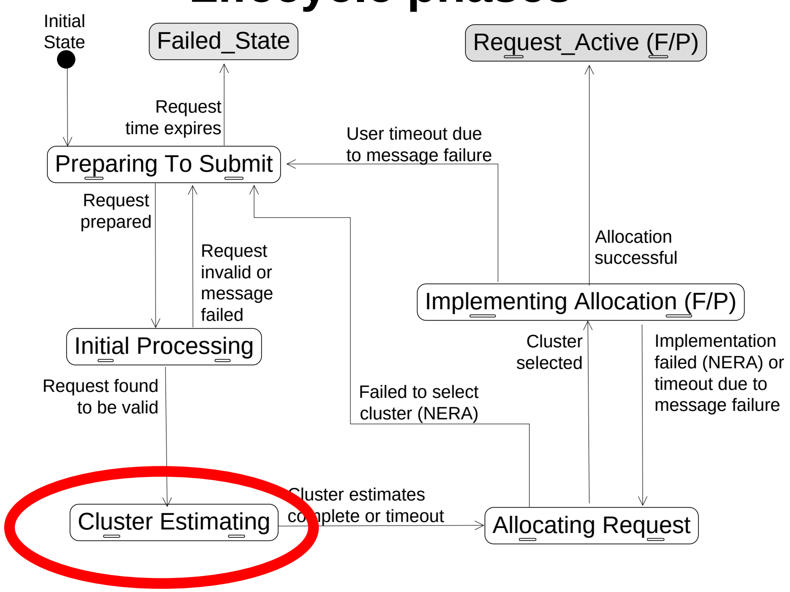
Outline

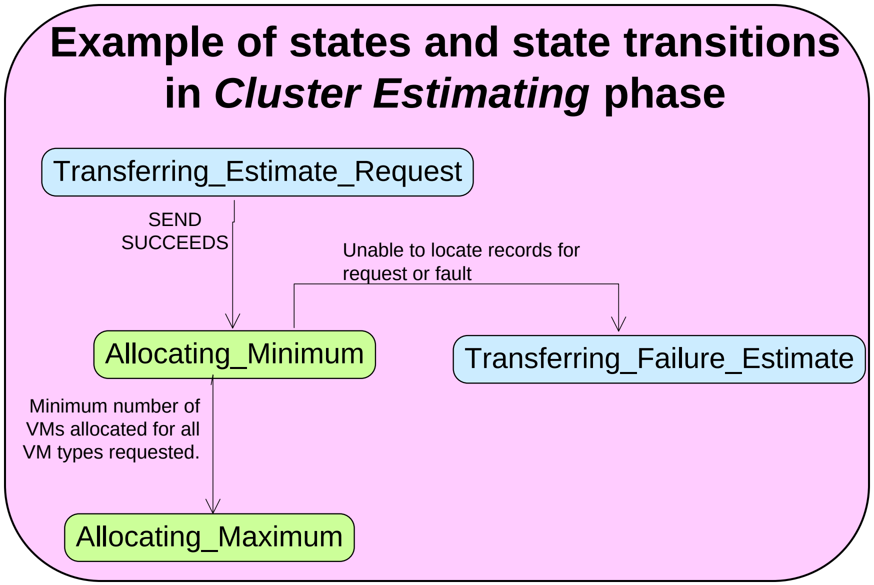
1. ***DTMC concepts and model development***
2. Perturbing a DTMC to identify a failure scenario
3. Using minimal s-t cut set analysis to reduce search for failure scenarios
4. Summary/Conclusions and future directions

State model of a cloud computing system

- Large-scale simulation for a cloud computing system [Mi2010]
  - Clouds “rent” compute resources – virtual machines or VMs (CPUs, memory, disk)
- Focus: Process of requesting and allocating VMs (computing resources)
  - ***Lifecycle of user requests -*** phases/stages in request process
  - Each phase is decomposable into detailed states and state transitions
  - Total of 39 states and 139 state transitions
  - **Request Active** (state) grant of VMs (resources) to users; **Failed State** rejection.

Lifecycle phases





Detailed model allows more precise analysis

Decomposed state model for *cluster estimating* phase

Summary of phase

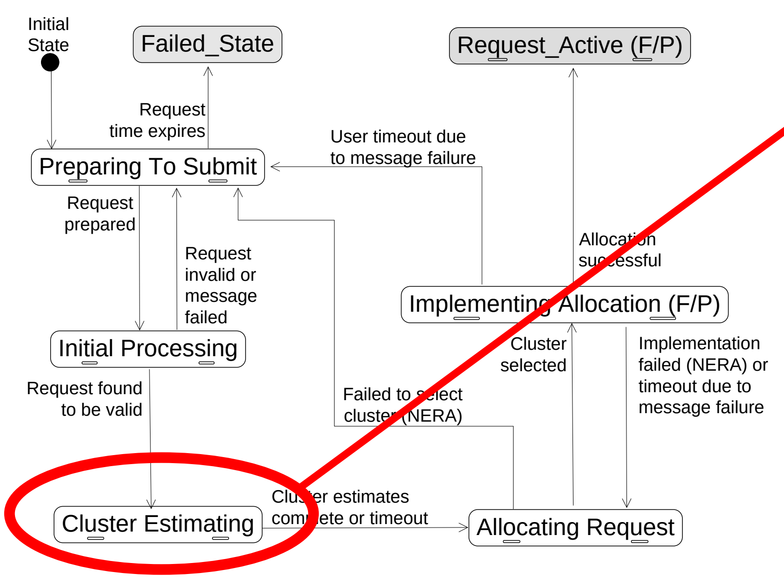
Cloud controller obtains estimates from clusters of ability to provide VMs to satisfy a user request.

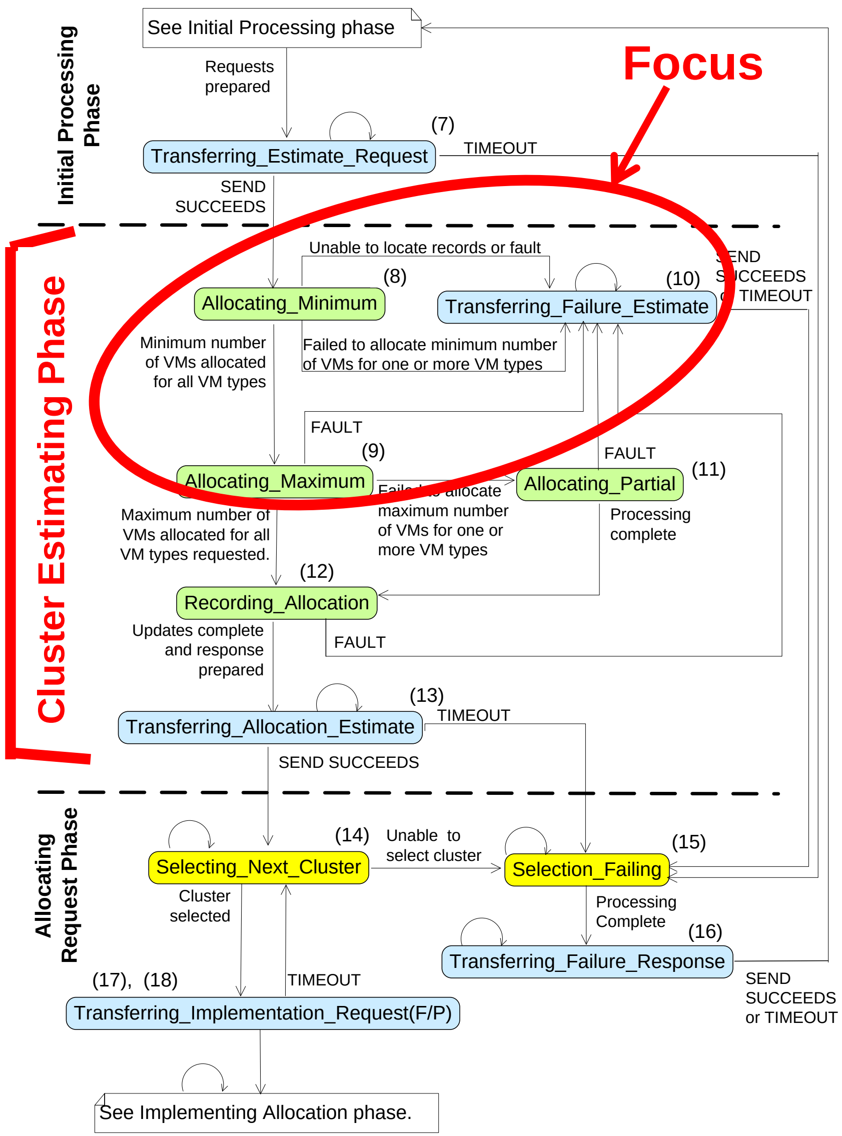
Partial Grant (*Allocating Minimum*)

Full Grant (*Allocating Maximum*)

then

1. Controller selects cluster to implement
2. If cluster successful request (eventually) reaches *Request_Granted* state.
3. Or, if no cluster can → *Failed_State*



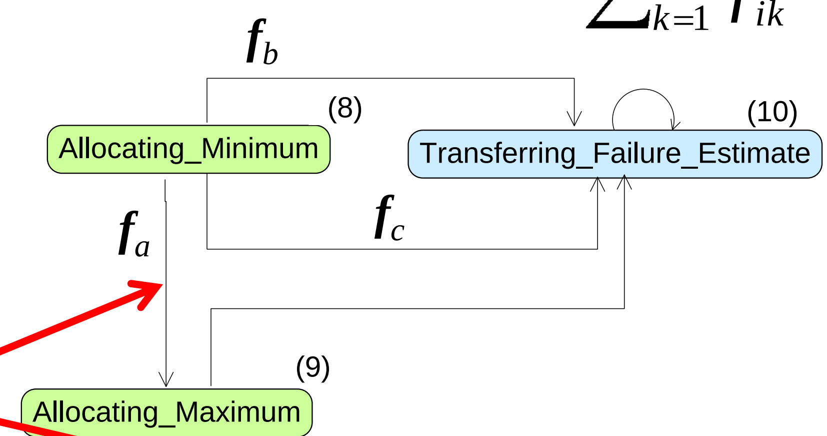


Building a Discrete Time Markov Chain (DTMC) model

- *DTMCs* are state models where probability of transition from one state to another does not depend on past history: $\Pr(X_{n+1} = x | X_n = x_n, \ldots, X_1 = x_1) = \Pr(X_{n+1} = x | X_n = x_n)$ for sequence of states $X_n, X_{n+1}, X_{n+2}\ldots\ldots$
- Probability state i transitions to state j, $\boldsymbol{p_{ij}}$, is the proportion of total number of transitions from state i to other states, where $\boldsymbol{f_{ij}}$ are frequencies.

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

Observe system (large scale simulation) and obtain frequencies for all transitions



Example:

$$p\,(\textit{Allocating\_Minimum} \rightarrow \textit{Allocating\_Maximum}) = \boxed{\frac{f_a}{f_a + f_b + f_c}}$$

→ ***Produce Transition Probability Matrix (TPM)*** (example submatrix)

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

*where p (ε) = $1.0e^{-6}$ without perturbation

INFORMATION TECHNOLOGY LABORATORY NIST
National Institute of Standards and Technology

Innovations in Measurement Science

Result is set of TPMs for *m* time periods

Summary TPM -- weighted average of m periods

To

From

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
1	Initial	0.995	0.005	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	Thinking	0	0.962	0.038	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	Submitting	0	ε	0.873	0.122	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0001
4	Transferring_User_Request	0	0	0.022	ε	0.978	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	Initiating_Request_Session	0	0	ε	0	0	1-2ε	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	Preparing_Cluster_Estimate_Requests	0	0	ε	0	0	0	1-2ε	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	Transferring_Estimate_Request	0	0	ε	0	0	0	ε	0.993	0	0	0	0	0	0	0.007	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	Allocating_Minimum	0	0	ε	0	0	0	0	0	0.248	0.752	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	Allocating_Maximum	0	0	ε	0	0	0	0	0	0	ε	0.464	0.536	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	Transferring Failure_Estimate	0	0	ε	0	0	0	0	0	0	ε	0	0	0	0	1-2ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	Allocating Partial	0	0	ε	0	0	0	0	0	0	ε	0	1-3ε	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	Recording_Allocation	0	0	ε	0	0	0	0	0	0	ε	0	0	1-3ε	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	Transferring_Allocation_Estimate	0	0	ε	0	0	0	0	0	0	0	0	0	0	1-2ε	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	Selecting_Next_Cluster	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0.168	0	0.402	0.429	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	Selection Failing	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	ε	1-3ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	Transferring_Failure_Response	0	0	0.952	0	0	0	0	0	0	0	0	0	0	ε	0	0.048	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	Transferring_Implementation_Request (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0.012	0	0	ε	0	0.133	0	0.855	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	Transferring_Implementation_Request (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0.012	0	0	0	ε	0	0.053	0	0.934	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	Queued_for_Implementation (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	ε	0	1-3ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	Queued_for_Implementation (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	ε	0	1-3ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	Verifying_Allocation (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0.821	0.061	0	0	0	0	ε	0	0	0.118	0	0	0	0	0	0	0
22	Verifying_Allocation (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	ε	0.684	0	0	0	0	ε	0	0	0.313	0	0	0	0	0	0	0
23	Launching_Instances (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0.485	0	0.496	0.018	0	0	0	0	0	0	0	0	0	0	0
24	Launching_Instances (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0.317	0	0.587	0.096	0	0	0	0	0	0	0	0	0	0
25	Reallocating_VM_Instances (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	1-5ε	0	ε	0	ε	0	ε	0	0	0	0	0	0	0	0	0	0
26	Reallocating_VM_Instances (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	1-4ε	0	ε	0	0	ε	0	0	0	0	0	0	0	0	0	0
27	Recording_Launch (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	1-3ε	0	0	0	0	0	0	0	0	0
28	Recording_Launch (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	1-3ε	0	0	0	0	0	0	0	0
29	Rolling_Back_Implementation	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1-2ε	0	0	0	0	0	0	0
30	Transferring_Implementation_Success (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	0	1-3ε	0	0	0	0	0	0
31	Transferring_Implementation_Success (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	0	1-3ε	0	0	0	0	0
32	Transferring_Implementation_Failure	0	0	ε	0	0	0	0	0	0	0	0	0	0	1-2ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ε	0	0	0	0	0	0	0
33	Preparing_Grant (F)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1-ε	0	0	0	0
34	Preparing_Grant (P)	0	0	ε	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1-ε	0	0	0
35	Transferring_Grant (F)	0	0	0.028	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.077	0	0.895	0	0
36	Transferring_Grant (P)	0	0	0.014	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.038	0	0.948	0
37	Request_Active (F)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
38	Request_Active (P)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
39	Failed_State	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- ***Key Concept:*** Observation of system over time yields series of TPMs for ***m*** successive time periods to form *a piece-wise homogenous DTMC* [Ro2004]

  → captures change over time.

- ***Absorbing states*** (tasks enter and never exit)
  - Requests Active (F/P) &
  - Failed_State

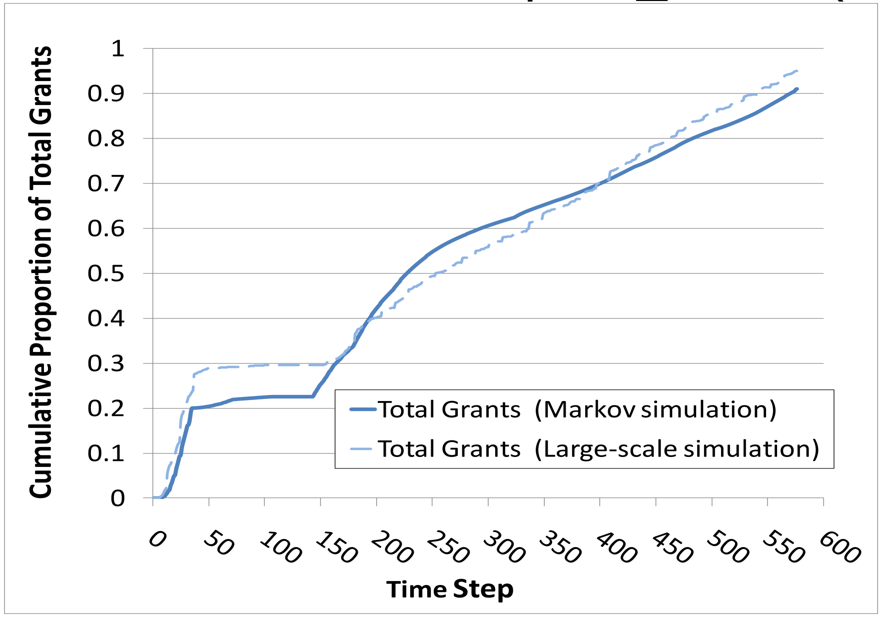
→ ***absorbing chain*** [Ke1976]

DTMC can simulate evolution of cloud computing system

- Set of TPMs, Q_i, for successive time periods (3600 s)
- System evolves in discrete time steps (100 s per step)
- Vector v_n shows system state at any step n:
  - consists of 39 elements → one for each state
- Matrix multiplication: $Q^T \cdot v_n = v_{n+1}$ with Q_i for related time period.

End system state vector $\mathbf{v}_{576}$ approximates result of large scale simulation, i.e., *Total Grants* or Request_Active (F/P)

16-hour period
(576 time steps and 16 TPMs)



Outline

1. DTMC concepts and model development
2. ***Perturbing a DTMC to identify a failure scenario***
3. Using minimal s-t cut set analysis to reduce search for failure scenarios
4. Summary/Conclusions and future directions

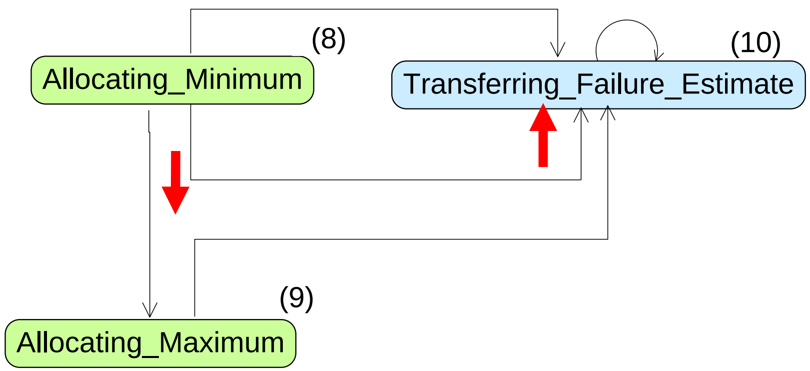
TPM perturbation

- *Modifying state transition probabilities changes behavior and outcome of Markov simulation*

EXAMPLE:

Decrease *p* (*Allocating_Minimum* → *Allocating_Maximum*)

Increase *p* (*Allocating_Minimum* → *Transferring_Failure_Estimate*)



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

→ *changes proportion of requests that enter absorbing states*, **Request_Active (F/P)** or *Total Grants*

(*Note: parenthesized numbers indicate TPM row number)

Markov simulation to predict performance degradation

- Simulation of perturbed critical transitions over multiple time periods (time inhomogeneous evolution) drives down performance
- *Can be related to failure scenarios:*

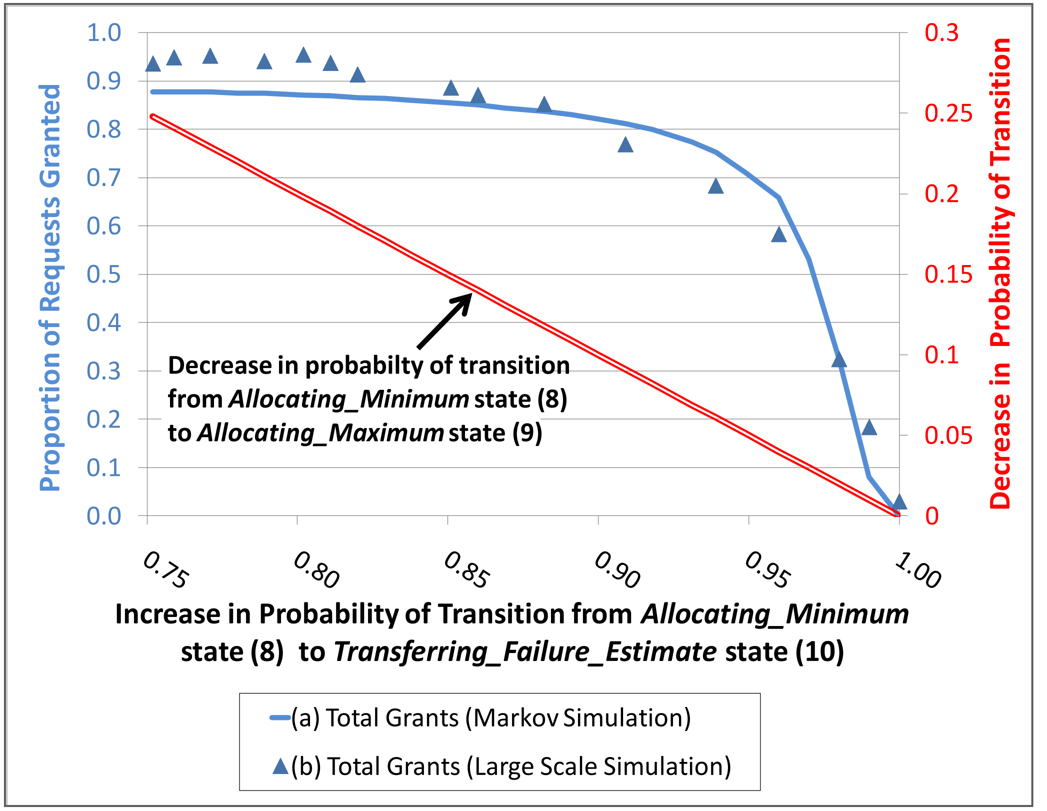
  ex. Cluster databases inaccessible to a software or hardware fault.

EXAMPLE:

Decrease *p* (*Allocating_Minimum* → *Allocating_Maximum*)

Increase *p* (*Allocating_Minimum* → *Transferring_Failure_Estimate*)

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



→ *Predict the performance of the system being modeled.*

Perturbation of combinations of critical transitions

- Multiple critical transitions can be perturbed together to reveal more complicated scenarios
- *Failure scenario:* impact of multiple (possibly related) software failures

Example: Perturbation of state transitions involving two different states

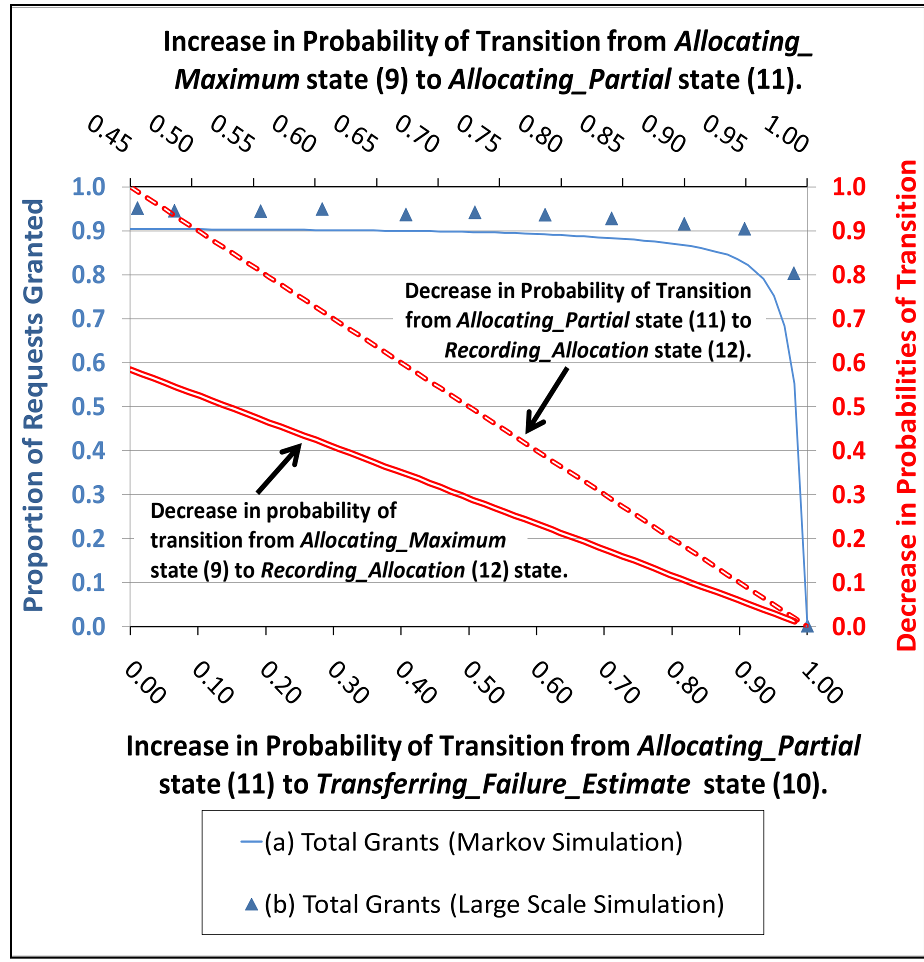
Decrease *p* (*Allocating_Maximum* → *p* (*Recording_Allocation*)

Increase *p* (*Allocating_Maximum* → *p* (*Allocating_Partial*)

Decrease *p* (*Allocating_Partial* → *p* (*Recording_Allocation*)

Increase *p* (*Allocating_Partial* → *p* (*Transferring_Failure_Estimate*)

		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



Computability of finding critical state transitions

Unfortunately, there may be many combinations of perturbations to examine in a large problem.

- Combinations in which the transition probability of one column is raised while the transition probabilities of one or more other non-zero columns in the same row is lowered involves 115 possible perturbations.
- When perturbing different combinations of rows together to find combinations of state transitions in different rows which together are critical, the figure increases by a factor of

$$\binom{n-4}{r}$$

where ***n*** = number of states (39) and ***r*** = number of rows in combination:
  - 5355 perturbations to examine all possible combinations of two rows
  - 58, 905 perturbations to examine all possible combinations of three rows

→ Brute force search over all combinations infeasible

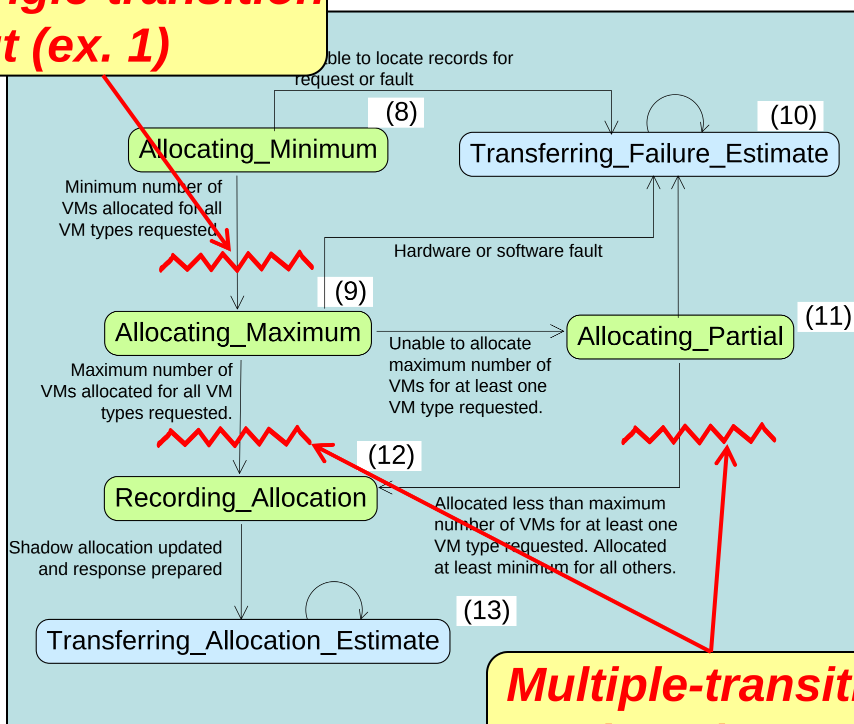
Outline

1. DTMC concepts and model development
2. Perturbing a DTMC to identify a failure scenario
3. ***Using minimal s-t cut set analysis to reduce search for failure scenarios***
4. Summary/Conclusions and future directions

Minimal s-t cut set analysis

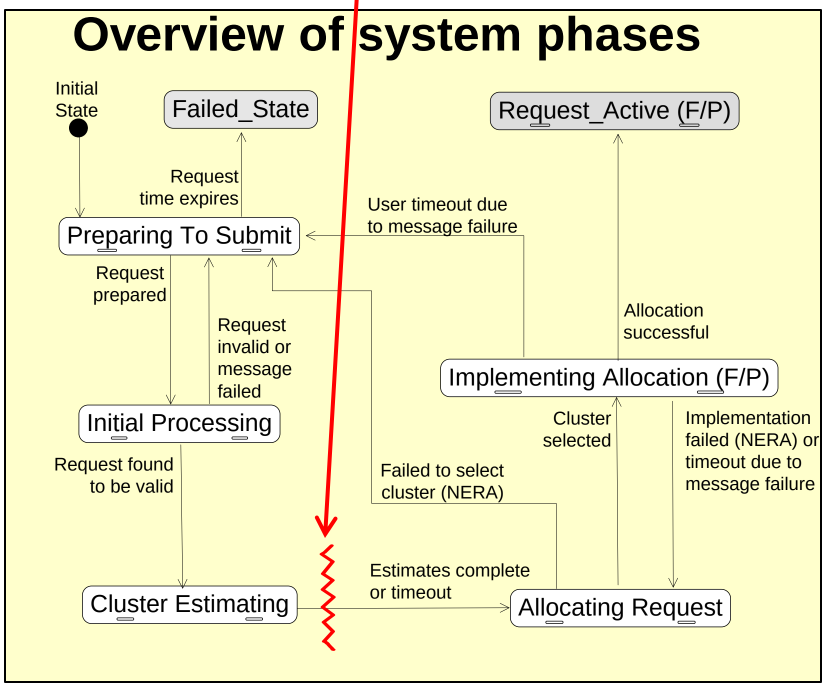
- A DTMC is a directed graph
- Minimal s-t cut set: edges (transitions) that disconnect all paths from vertex ***s*** (*Initial* state) to vertex ***t*** --desired absorbing states *Request_Active (F/P)*
  - → *Cut sets contain critical transitions where perturbation drives down performance*

Single-transition cut (ex. 1)



Multiple-transition cut (ex. 2)

Cut all paths from Cluster Estimating phase to Requests Active (F/P)



Applying minimal s-t cut set analysis

- Use of algorithm to enumerate all cut sets in a directed graph [Pr1984]
- Results in 159 cut sets of 1 to 5 transitions in size
  - Ex. 33 cut sets of one and two transitions vs. 115 + 5355
  - 26 cut sets of three transition vs. 58,905

→ ***Reduces number of perturbation combinations to examine to focus on most critical***

→ ***<u>2x magnitude reduction in computation cost</u>***

Single-transition cuts

	Set of member transitions	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
1-5	{4, 5}	0.978
1-6	{6, 7}	0.978
1-7	{7, 8}	0.990
1-8	{13, 14}	0.991
1-9	{5, 6}	0.995
1-10	{12, 13}	1.000

Multiple-transition cuts

	Set of member transitions	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
2-4	{23, 27} {36, 38}	2	1.438
2-5	{23, 27} {31, 34}	2	1.499
2-6	{23, 27} {28, 31}	2	1.507
2-7	{23, 27} {34, 36}	2	1.507
2-8	{35, 37} {36, 38}	2	1.861
2-9	{31, 34} {35, 37}	2	1.922
2-10	{30, 33} {36, 38}	2	1.924
2-11	{28, 31} {35, 37}	2	1.930
2-12	{34, 36} {35, 37}	2	1.930
2-13	{27, 30} {36, 38}	2	1.931
2-14	{33, 35} {36, 38}	2	1.931
2-15	{30, 33} {31, 34}	2	1.985
2-16	{27, 30} {31, 34}	2	1.993
2-17	{31, 34} {33, 35}	2	1.993
2-18	{28, 31} {30, 33}	2	1.993
2-19	{30, 33} {34, 36}	2	1.993
2-20	{27, 30} {28, 31}	2	2.000
2-21	{27, 30} {34, 36}	2	2.000
2-22	{28, 31} {33, 35}	2	2.000
2-23	{33, 35} {34, 36}	2	2.000

Using minimal s-t cut sets to identify critical transitions and most likely failure scenarios

Further reducing 159 minimal s-t cut sets:

- Structural information
  - Ordering by number of transitions
    → *fewer transitions more likely to occur*
  - Ordering by probability
  - All transitions originate from same state (▼)

- Use of domain expertise to reduce selection
  - Ex. cut sets with transitions in same system component .
    - Cloud Controller (green)
    - Cluster Controller (orange)
    - Network (light blue)

→ ***Narrows down system (10-15) failure scenarios of greatest interest and likelihood.***

Single-transition cuts

	Set of member transitions	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
1-5	{4, 5}	0.978
1-6	{6, 7}	0.978
1-7	{7, 8}	0.990
1-8	{13, 14}	0.991
1-9	{5, 6}	0.995
1-10	{12, 13}	1.000

Multiple-transition cuts

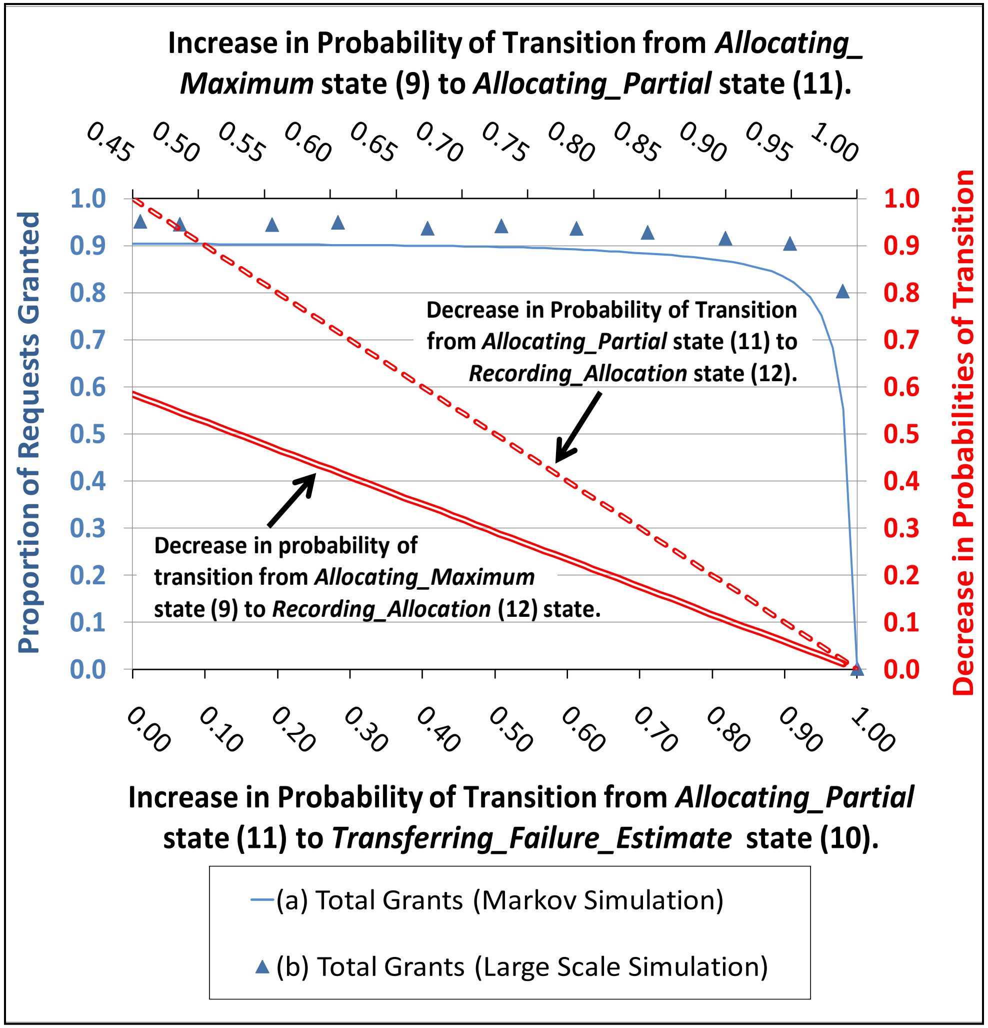
		Set of member transitions	Number of From States	Total Probabilty
	2-1	{14, 17} {14, 18}	1	0.895
▼	2-2	{9, 11} {9, 12}	1	1.000
▼	2-3	{9, 12} {11, 12}	2	1.395
	2-4	{23, 27} {36, 38}	2	1.438
	2-5	{23, 27} {31, 34}	2	1.499
	2-6	{23, 27} {28, 31}	2	1.507
	2-7	{23, 27} {34, 36}	2	1.507
	2-8	{35, 37} {36, 38}	2	1.861
	2-9	{31, 34} {35, 37}	2	1.922
	2-10	{30, 33} {36, 38}	2	1.924
	2-11	{28, 31} {35, 37}	2	1.930
	2-12	{34, 36} {35, 37}	2	1.930
	2-13	{27, 30} {36, 38}	2	1.931
	2-14	{33, 35} {36, 38}	2	1.931
	2-15	{30, 33} {31, 34}	2	1.985
	2-16	{27, 30} {31, 34}	2	1.993
	2-17	{31, 34} {33, 35}	2	1.993
	2-18	{28, 31} {30, 33}	2	1.993
	2-19	{30, 33} {34, 36}	2	1.993
	2-20	{27, 30} {28, 31}	2	2.000
	2-21	{27, 30} {34, 36}	2	2.000
	2-22	{28, 31} {33, 35}	2	2.000
	2-23	{33, 35} {34, 36}	2	2.000

Using simulated failure scenarios as a basis for prediction

Example: Markov simulation and perturbation of ***cut set 2-3***:

- Corresponds to software failure scenario involving multiple faults/attacks.
- Simulation identifies threshold beyond which increased failure incidence causes drastic performance collapse

→ *Verified by large-scale simulation*



***Conclusion*: Study indicates approach can be used to predict potential for failure and is more tractable than exhaustive search**

Outline

1. DTMC concepts and model development
2. Perturbing a DTMC to identify a failure scenario
3. Using minimal s-t cut set analysis to reduce search for failure scenarios
4. ***Conclusions and Future Directions***

Conclusions

- Results show potential of approach to model system failure scenarios at reduced computation cost
  - Generally 2x less than brute force search
  - Three examples in paper—can be expanded

  → ***Indicates potential for predictive use***

- Approach uses techniques in combination
  - Large, detailed DTMC models and TPMs
  - Time inhomogeneous representation to capture change over time
  - Markov simulation and quantitative performance analysis (thresholds)
  - Minimal s-t cut set analysis

  <u>Use of all four in combination not previously reported</u>

- Areas of further work
  - Tractability for large problems
  - Applicability to other domains
  - Investigate other approaches to finding critical transitions [Hu2011, Da2010]

Tractability and generality of minimal s-t cut set analysis

- **<u>Tractability</u> : number of potential cut sets in big graphs poses barriers.**
  → ***Progress in application to larger problems***. See [Da2011b]:
  – Developed node contraction algorithm which finds minimal s-t cut sets probabilistically (though not guaranteed to find all)
  – Applied contraction algorithm to four large DTMC TPMs with as many as > 4.22×10^8 cut sets
  – Found most of most highly-ranked cut sets also found through cut set enumeration, with some exceptions.

- **<u>Generality</u>: application to other domains.**
  – In a smaller grid computing problem (7 states, 18 state transitions), minimal s-t cut set analysis was used to identify ***<u>all</u>*** critical state transitions found through brute force search of combinations [Da2011b].
  – Applied to domain of network congestion control algorithm modeling. See [Da2010]

Another issue: understanding effects of perturbation on *distant* states

- While Markov simulation of perturbed TPMs for cloud computing system was reasonably predictive of Total Grants of all Requests (full and partial),

→***Much harder to predict effect of perturbation on full and partial grants separately***

- Why? In large-scale simulation (or target real-world system), indirect effects occur between parts of a system that cannot be modeled as states that are in direct transition with each other.

  Ex. Failures of messages from cloud controllers to clusters reduces overall performance, but also increases resource availability. This leads to relative increase of full grants partial→ hence, full grants decline less than expected.

- Area of current interest and investigation

References

[Da 2010] Dabrowski, C., Hunt, F. Morrison, K. Improving Efficiency of Markov Chain Analysis of Complex Distributed Systems. NISTIR 7744, National Institute of Standards and Technology, Gaithersburg, MD

[Da2011b] Dabrowski, C. and Hunt F., Using Markov Chains and Graph Theory Concepts to Analyze Behavior In Complex Distributed Systems. To appear in *Proceedings of 2011 European Modelling and Simulation Conference (EMSS)*, Rome, Italy. September 2011.

[Hu2011] Hunt, F., Morrison, K., Dabrowski, C., Using Eigensystem Decomposition in Markov Chain Analysis of Complex Systems. To appear in *Proceedings of the the Nineteenth IASTED International Conference on Applied Simulation and Modelling (ASM 2011). Crete, Greece. June 2011.*

[Pr1984] Provan S., and Ball M., 1984, “Computing Network Reliability in Time Polynomial in the Number of Cuts,” *Operations Research*, 32(3), pp. 516–526.

[Ro2004] Rosenberg, D., Solan, E. and Vielle N., Approximating A Sequence of Observations By A Simple Process. *The Annals of Statistics*. Volume 32, Number 6. pp. 2742-2775. 2004.

Other publications of interest

Dabrowski, C., Hunt, F. Using Markov Chain Analysis to Study Dynamic Behavior in Large-Scale Grid Systems. *Seventh Australasian Symposium on Grid Computing and e-Research (AUSGRID 2009)*, Wellington, New Zealand**.** January 2009

Karger, D., Stein, C. A New Approach to the Minimum Cut Problem. *Journal of the ACM*. Volume 43, pp. 601-640.

Kemeny, J. and Snell, J., *Finite Markov Chains*. New York, Springer, 1976.

Mills K., Dabrowski C. Investigating Global Behavior in Computing Grids. *LNCS*, vol. 4124. pp. 120-136. Springer (2006)

Tsukiyama S, Shirakawa I, Ozaki H, and Ariyoshi H., An Algorithm to Enumerate All Cut Sets of a Graph in Linear Time per Cutset. *Journal of the Association of Computing Machinary*. Volume 27, Number 4. pp. 619-632. October 1980.