Optimization of IaaS Cloud including Performance, Availability, Power Analysis

Networking 2014

Trondheim, Norway

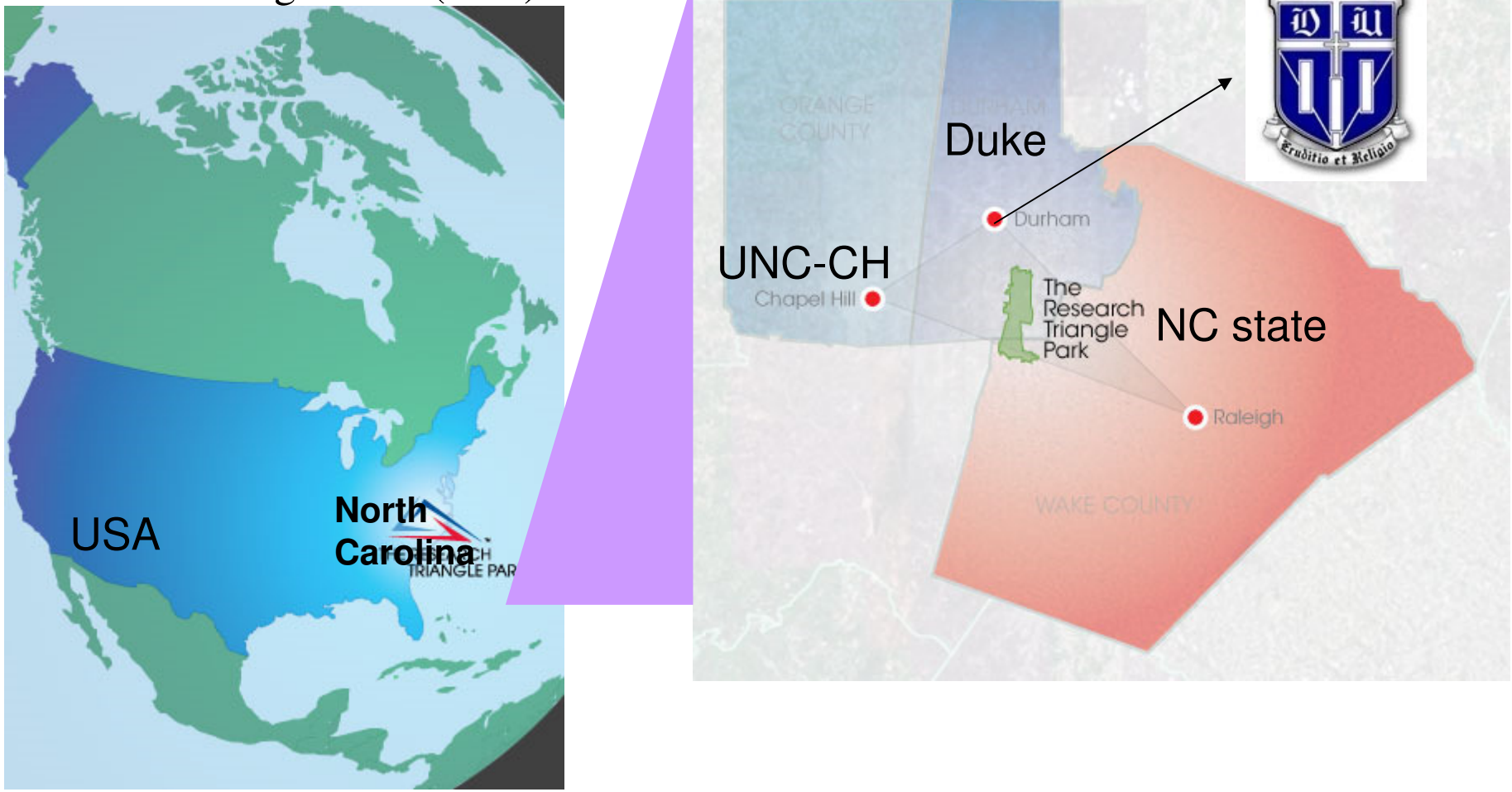
June 2, 2014

Prof. Kishor S. Trivedi
Duke High Availability Assurance Lab (**DHAAL**)
Department of Electrical and Computer Engineering
Duke University, Durham, NC 27708-0291
Phone: (919) 660-5269 E-mail: ktrivedi@duke.edu
URL: www.ee.duke.edu/~ktrivedi

Copyright © 2014 by K.S. Trivedi

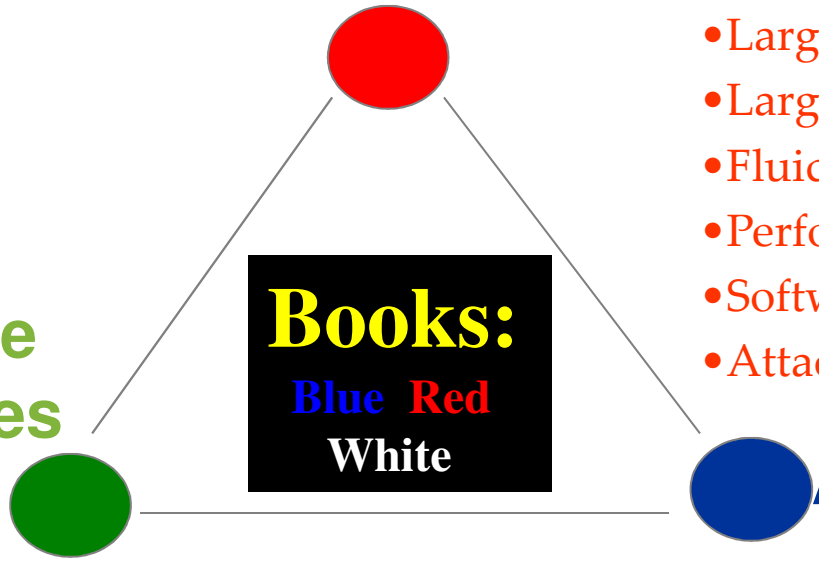
Duke University

Research Triangle Park (RTP)



Copyright © 2014 by K.S. Trivedi

DHAAL Research Triangle



Stochastic modeling methods & numerical solution methods:

- Large Fault trees, Stochastic Petri Nets,
- Large/stiff Markov & non-Markov models
- Fluid stochastic Petri Nets
- Performability & Markov reward models
- Software aging and rejuvenation
- Attack countermeasure trees

HARP (NASA), SAVE (IBM),
IRAP (Boeing)
SHARPE, SPNP, SREPT

Reliability/availability/performance
Avionics, Space, Power systems,
Transportation systems,
Automobile systems
Computer systems (hardware/software)
Telco systems
Computer Networks
Virtualized Data center
Cloud computing

Copyright © 2014 by K.S. Trivedi

Books Authored by Trivedi

- Probability and Statistics with Reliability, Queuing, and Computer Science Applications, first edition, Prentice-Hall, 1982; Second edition, John Wiley, 2001 (Bluebook)

- Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package, Kluwer, 1996 (Redbook)

- Queuing Networks and Markov Chains, John Wiley, first edition, 1996; second edition, 2006 (White book)

Copyright © 2014 by K.S. Trivedi

Talk outline

- Overview of Cloud Computing
- Cloud Capacity Planning
  - Availability Model for IaaS Cloud
  - Performance Model for IaaS Cloud
  - Power Model for IaaS Cloud

Copyright © 2014 by K.S. Trivedi

An Overview of Cloud Computing

Copyright © 2014 by K.S. Trivedi

Key characteristics

- **On-demand self-service:**
  Provisioning of computing capabilities without human intervention

- **Resource pooling:**
  Shared physical and virtualized environment

- **Rapid elasticity:**
  Through standardization and automation, quick scaling

- **Metered Service:**
  Pay-as-you-go model of computing

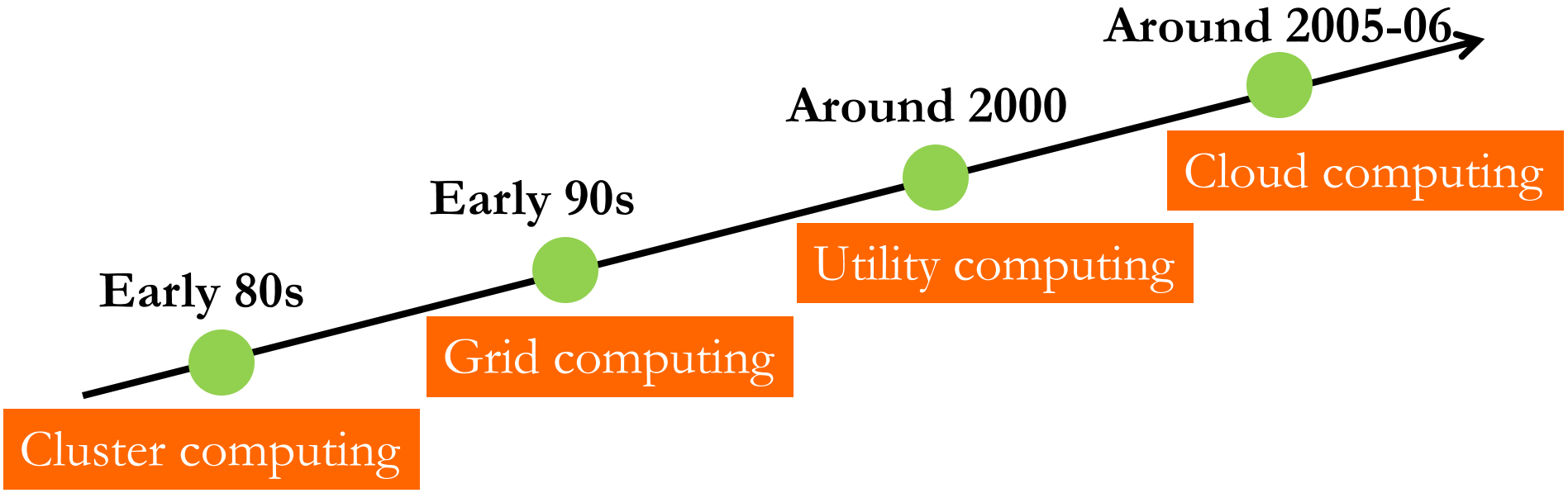
Many of these characteristics are borrowed from Cloud's predecessors!

Source: P. Mell and T. Grance, "The NIST Definition of Cloud Computing", October 7, 2009

Copyright © 2014 by K.S. Trivedi

Evolution of cloud computing

- Time line of evolution



*Source: http://seekingalpha.com/article/167764-tipping-point-gartner-annoints-cloud-computing-top-strategic-technology

Copyright © 2014 by K.S. Trivedi

Cloud Service models

- **Infrastructure-as-a-Service (IaaS) Cloud:**
  Examples: Amazon EC2

- **Platform-as-a-Service (PaaS) Cloud:**
  Examples: Microsoft Windows Azure, Google AppEngine

- **Software-as-a-Service (SaaS) Cloud:**
  Examples: Gmail, Google Docs

Copyright © 2014 by K.S. Trivedi

Deployment models

- **Private Cloud:**
  - Cloud infrastructure solely for an organization
  - Managed by the organization or third party
  - May exist on premise or off-premise
- **Public Cloud:**
  - Cloud infrastructure available for use for general users
  - Owned by an organization providing cloud services
- **Hybrid Cloud:**
  - Composition of two or more clouds (private or public)

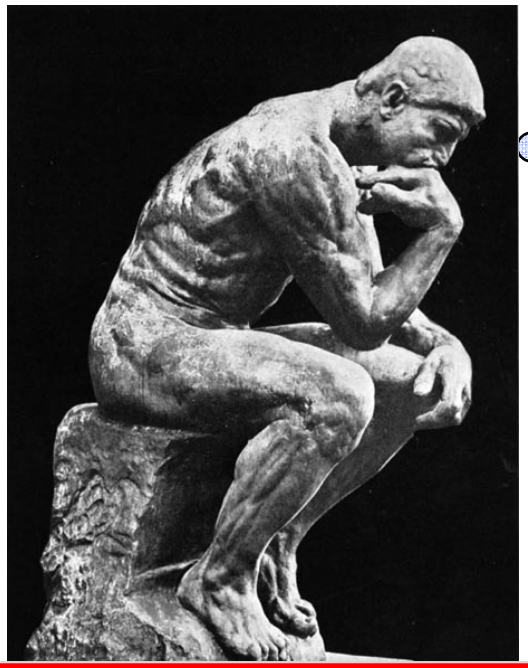
Copyright © 2014 by K.S. Trivedi

Stochastic Model Driven Capacity Planning for an IaaS Cloud,

R. Ghosh, F. Longo, R. Xia, V. Naik, and K. Trivedi,
IEEE Trans. On Services Computing, 2014 (to appear)

Copyright © 2014 by K.S. Trivedi

SLA driven capacity planning



What is the optimal #PMs so that total cost is minimized?

Large sized cloud, large # configurations to search

Copyright © 2014 by K.S. Trivedi

Capacity Planning Problem

Determine the number of Physical Machines

That

Minimize the overall cost

Copyright © 2014 by K.S. Trivedi

Duke/IBM project on cloud computing

Joint work with
Rahul Ghosh, Ruofan Xia and Dong Seong Kim (Duke),
Francesco Longo (Univ. of Messina)
Vijay Naik, Murthy Devarakonda and Daniel Dias
(IBM T. J. Watson Research Center)

Copyright © 2014 by K.S. Trivedi

Cost components

- Capital Expenditure (CapEx)
  - Infrastructure cost
- Operational Expenditure (OpEx)
  - Penalty due to violation of different SLA metrics
    - Cost of job rejection due to insufficient resources
    - Cost of downtime
  - Cost of carrying out repairs
  - Power usage cost

Copyright © 2014 by K.S. Trivedi

Three Pools of Servers (PMs)

- To reduce power usage costs, physical machines are divided into three pools [IBM Research Cloud]
  - Hot pool (high performance & high power usage)
  - Warm pool (medium performance & power usage)
  - Cold pool (lowest performance & power usage)

Copyright © 2014 by K.S. Trivedi

System Operation Details

- Failure/Repair (Availability):
  - Servers may fail and get repaired.
  - A minimum number of operational hot servers are required for the system to function.
  - Servers in other pools may be temporarily assigned to the hot pool to maintain system operation (migration).

- Job Arrival/Service (Performance):
  - New jobs may be rejected if existing workload is heavy so all resources are occupied.

- Server operation consumes power depending on the server status (i.e., the pool it is in and the number of active VMs)

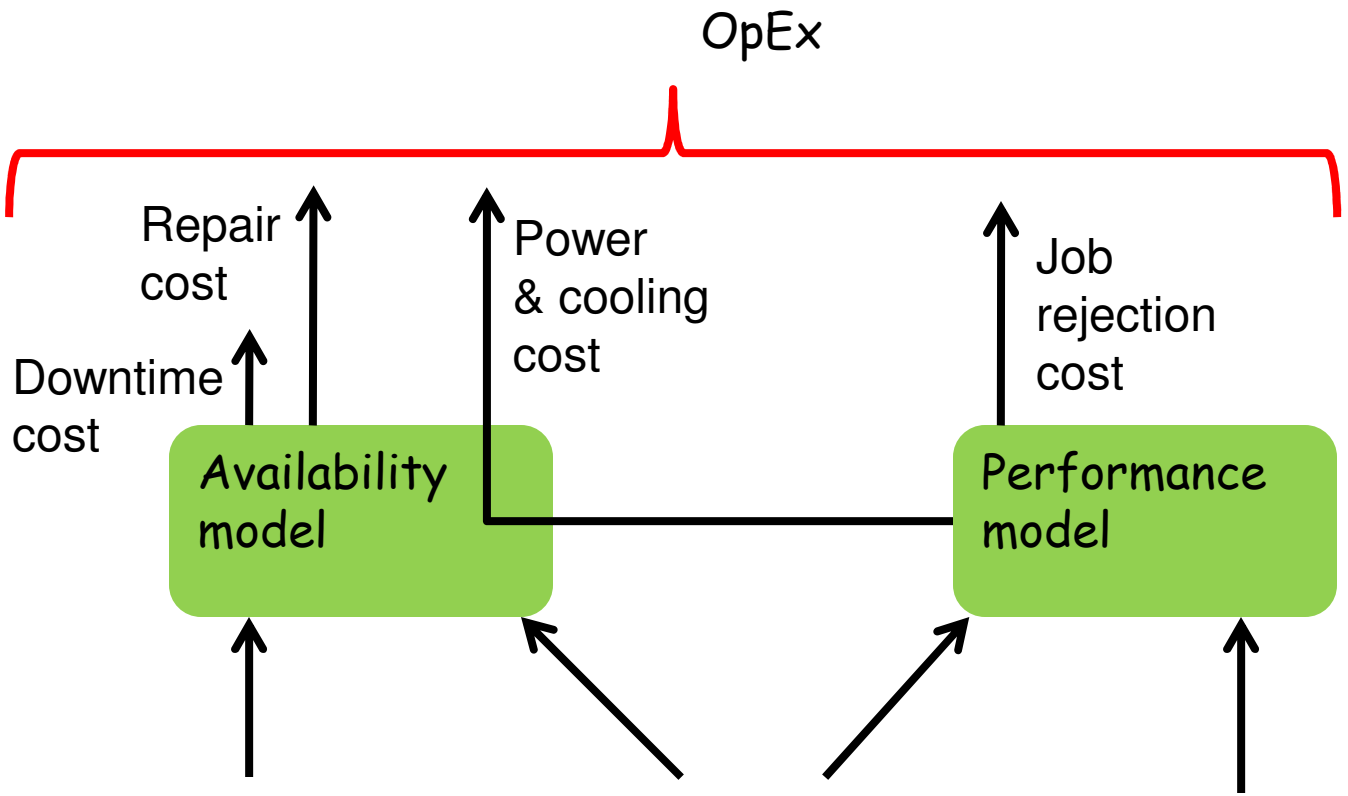
Copyright © 2014 by K.S. Trivedi

Optimization Problem

- Determine the number of PMs in each pool: n_h, n_w, n_c so as to minimize $CapEx(n_h, n_w, n_c) + OpEx(n_h, n_w, n_c)$
  - n_h, n_w, n_c: number of servers in the hot, warm, cold pool
- CapEx function can be "easily" determined
- OpEx for each vector of PMs in each pool needs to be computed
- For a search-based optimization algorithm, this OpEx computation needs to be done many times
  - We need an efficient algorithm for doing this
- Scalable models are developed for such a computation

Copyright © 2014 by K.S. Trivedi

High level view of developed models for OpEx



Copyright © 2014 by K.S. Trivedi

Cost Component – Part I

- Infrastructure cost:

$$C_{inf} = (n_h + n_w + n_c) \cdot C_f + (\left\lceil \frac{n_h}{n_s} \right\rceil + \left\lceil \frac{n_w}{n_s} \right\rceil + \left\lceil \frac{n_c}{n_s} \right\rceil) C'_f$$

C_f : cost of each server

n_h, n_w, n_c: number of servers in the hot, warm, cold pool

n_s: the number of servers on a chassis

C'_f : the cost of a server chassis

- Rejection cost:

$$C_{rej} = \rho_{reject} \cdot C_t \cdot L$$

ϱ_{reject}: task rejection rate from the **performance model**

C_t: cost of each task rejection

L : length of the operation period

Copyright © 2014 by K.S. Trivedi

Cost Components – Part II

- Repair cost:

$$C_{rep} = (r_h + r_w + r_c) \cdot C_r \cdot L$$

r_h, r_w, r_c : Mean number of repairs per time unit in the hot, warm and cold pool respectively

C_r : Cost of each repair

L: Length of time of operation

- Downtime cost:

$$C_{down} = C_d \cdot \max\{0, DT - DT_{th}\}$$

C_d : Revenue loss per time unit because of downtime

DT: Total steady state downtime in minutes for the Cloud during operational period L (hr) (from **Availability model**)

DT_{th}: Threshold on downtime beyond which downtime cost is incurred

Copyright © 2014 by K.S. Trivedi

Cost Components – Part III

- Power and cooling cost:

$$p_a(x,y,z) = p_h(x) \cdot p_w(y) \cdot p_c(z)$$

$$C_{pow} = \left(\sum_{x=0}^{n_h} \sum_{y=0}^{n_w} \sum_{z=0}^{n_c} p_a(x,y,z) W_{(x,y,z)} \right) \cdot C_p \cdot L$$

$p_h(x), p_w(y), p_c(z)$: the probability that there are x, y, z servers in the hot, warm, cold pool respectively. Computed from the **availability model**.

$W_{(x, y, z)}$: the power consumption when there are x, y, z servers running in the hot, warm, cold respectively. Computed from the **performance model**.

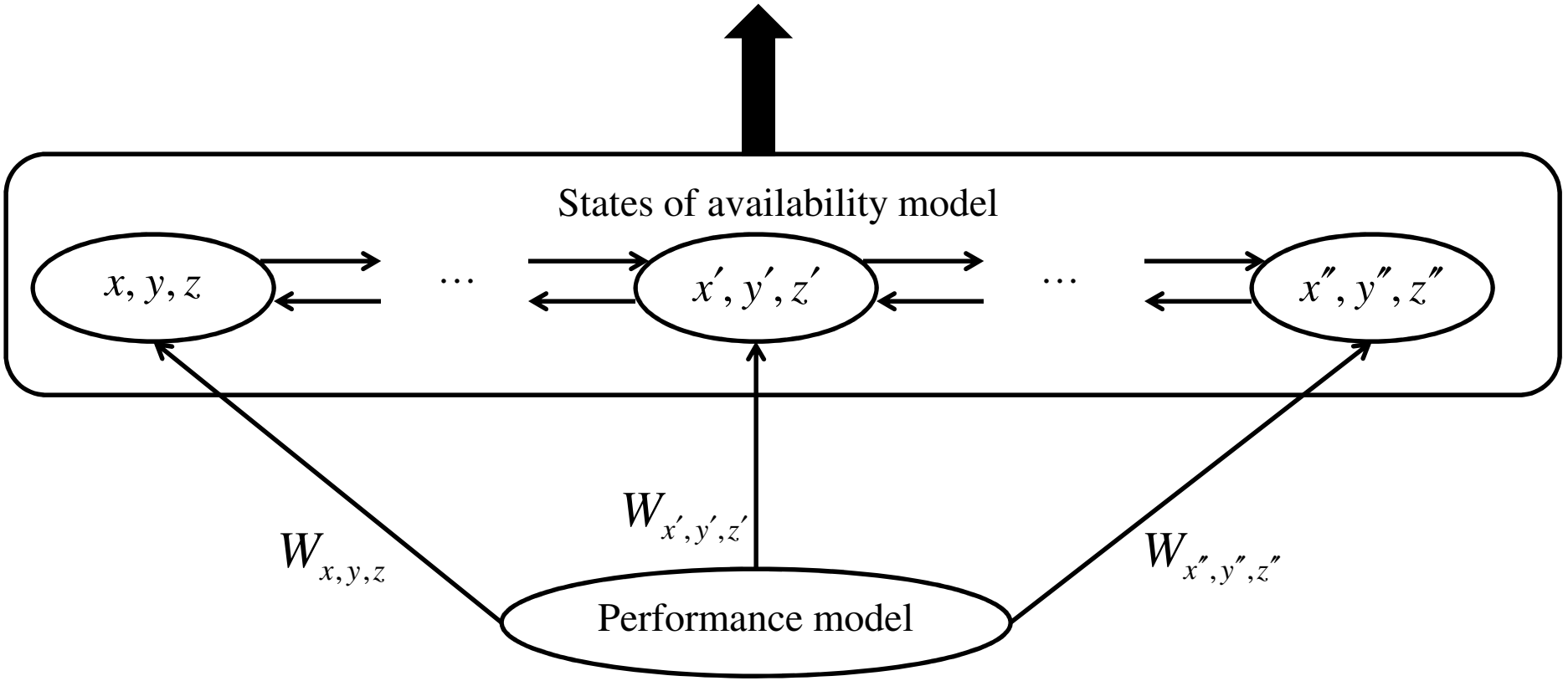
C_p: cost of per unit of power consumption

L: length of the operation period

Copyright © 2014 by K.S. Trivedi

Power and cooling cost

Overall power consumption and cooling cost as expected steady state reward rates



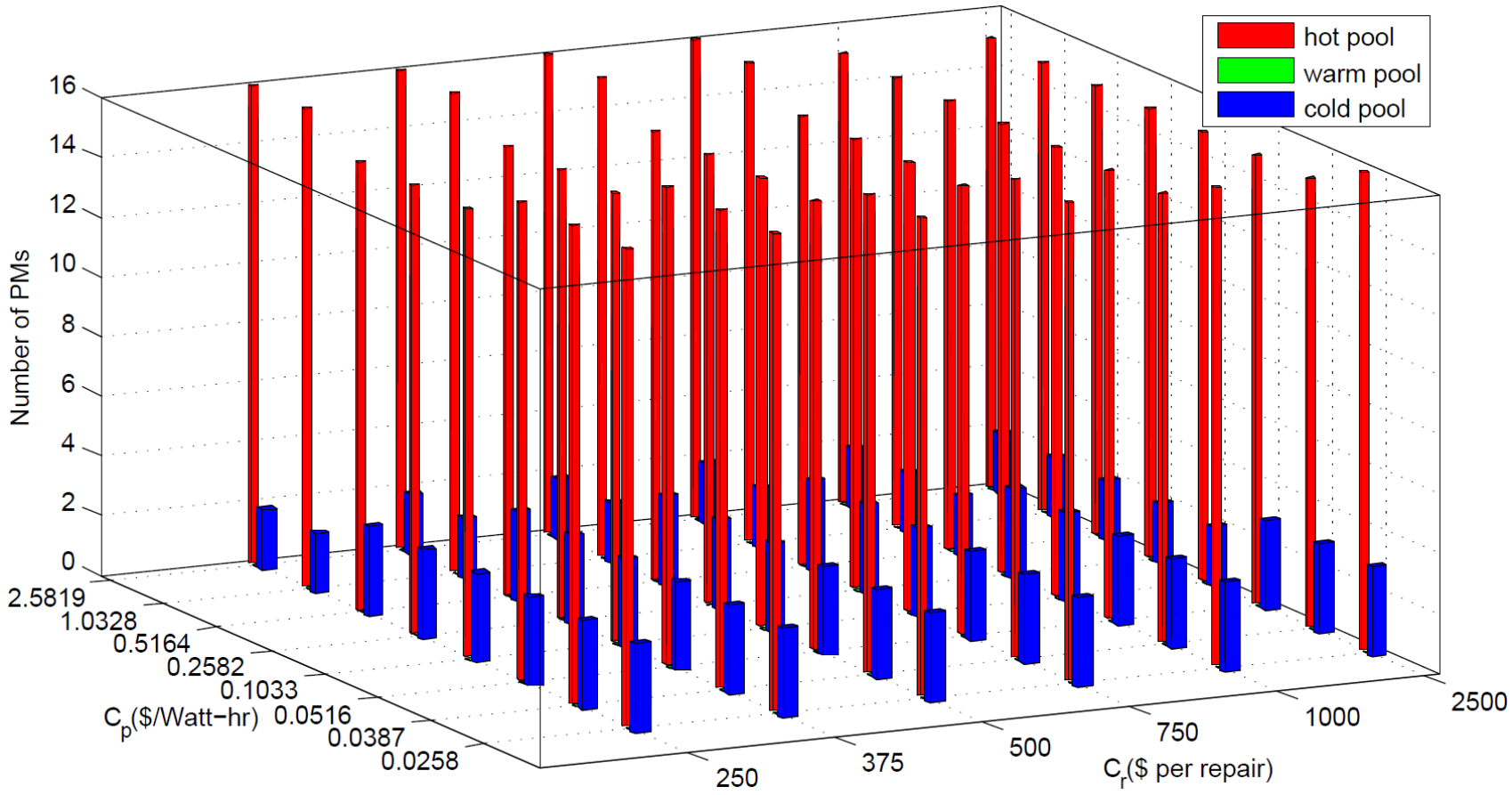
Optimization Problems and Solution Approach

- The problems are nonlinear and (in general) non-convex.
- We use Simulated Annealing but other search algorithms can be used as well.
- For each vector of values of the number of severs in each pool, we need and efficient method of computing the job rejection probability, downtime and the power usage cost → scalable **availability, performance and power** models are needed

Copyright © 2014 by K.S. Trivedi

Sample Results

- Optimal configurations in different problem instances



Copyright © 2014 by K.S. Trivedi

Comparison with intuition based approach

- Consider a case where OpEx involves only:
  - Power consumption and cooling costs
- An example of intuition based capacity planning in such scenario:
  - More PMs in hot pool $\longrightarrow$ higher power cost
  - More PMs in cold pool $\longrightarrow$ lower power cost
- In our previous paper (DSN workshop DCDV 2011), we showed, such intuition based approach does not always hold true
  - When the workload arrival rate is high, PMs in the cold pool will act as PMs in the hot pool
  - Cold pool power consumption will be almost same as the hot pool

Copyright © 2014 by K.S. Trivedi

How do we develop scalable performance and availability and power models to compute OpEx ?

Copyright © 2014 by K.S. Trivedi

Our goals in the IBM Cloud project

- Develop a comprehensive analytic modeling approach
- High fidelity
- Scalable and tractable

Copyright © 2014 by K.S. Trivedi

Our approach

- Monolithic analytic (Markov) models will not work as they will suffer largeness and hence not scalable
- Our approach:
  - overall system model decomposed into a set of sub-models
  - sub-model solutions composed via an interacting Markov chain approach
  - Fixed-Point problem solved via successive substitution
  - scalable and tractable

Copyright © 2014 by K.S. Trivedi

Scalable Analytic Model for IaaS Cloud Availability and Downtime

[paper in IEEE Trans. On Cloud Computing 2014]

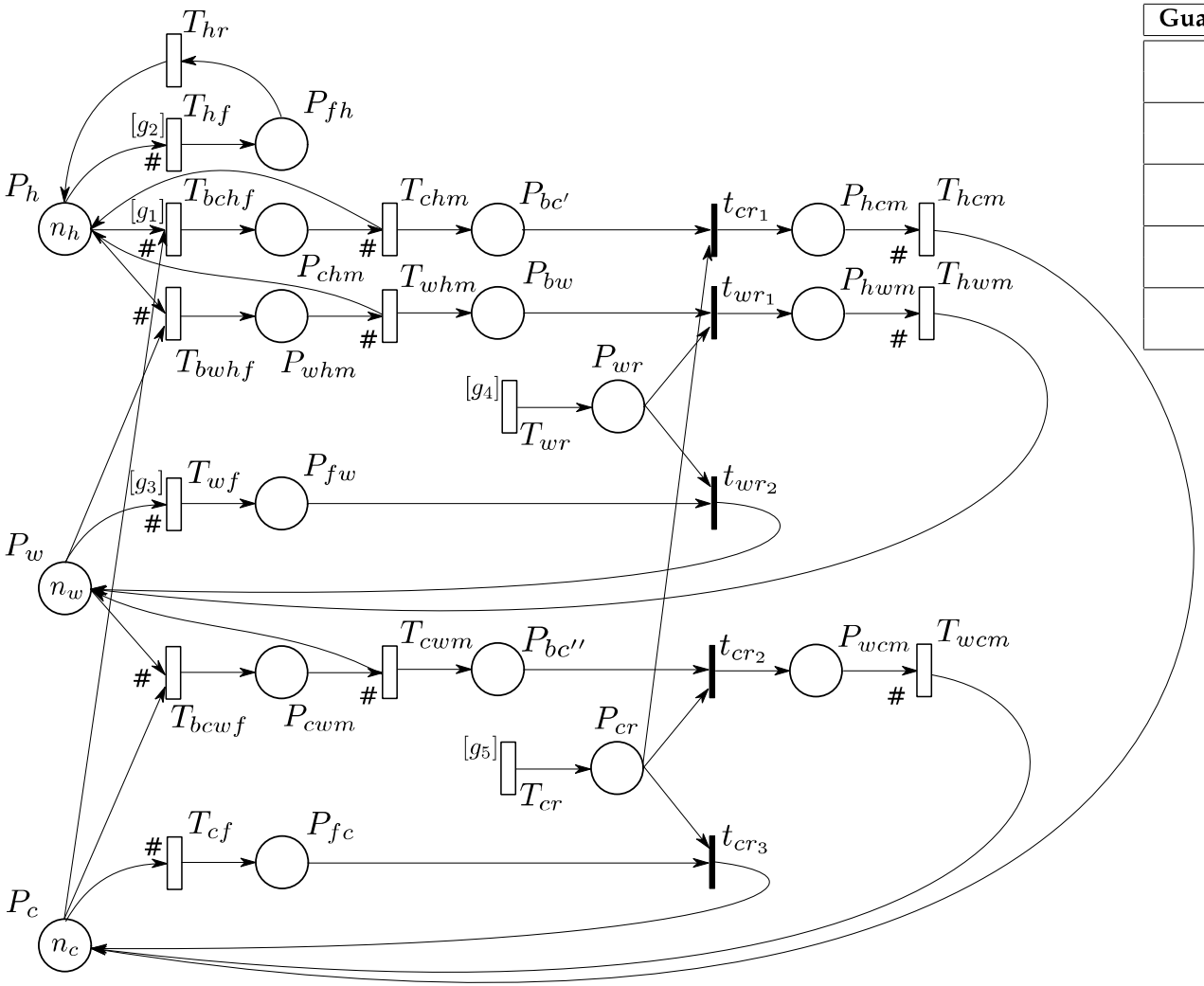
Copyright © 2014 by K.S. Trivedi

Analytic model

- Markov model (CTMC) is too large to construct by hand.
  - The number of PMs in each pool can be large
  - PMs can migrate among pools
- We use a high level formalism of stochastic Petri net (the flavor known as stochastic reward net (SRN)).
- SRN models can be automatically converted into underlying Markov (reward) model and solved for the measures of interest such as DT (downtime)
- For very large number of PMs even decomposed models are not enough; we resort to discrete-even simulation; same SRN model can be simulated via our software package (SPNP)

Copyright © 2014 by K.S. Trivedi

Monolithic SRN Model



Guard functions	Values
$[g_1]$	1 if $\#P_w = 0$ 0 otherwise
$[g_2]$	1 if $\#P_w = 0$ and $\#P_c = 0$ 0 otherwise
$[g_3]$	1 if $\#P_c = 0$ 0 otherwise
$[g_4]$	1 if $\#P_{fw} + \#P_{bw} > 0$ 0 otherwise
$[g_5]$	1 if $\#P_{fc} + \#P_{bc'} + \#P_{bc''} > 0$ 0 otherwise

Copyright © 2014 by K.S. Trivedi

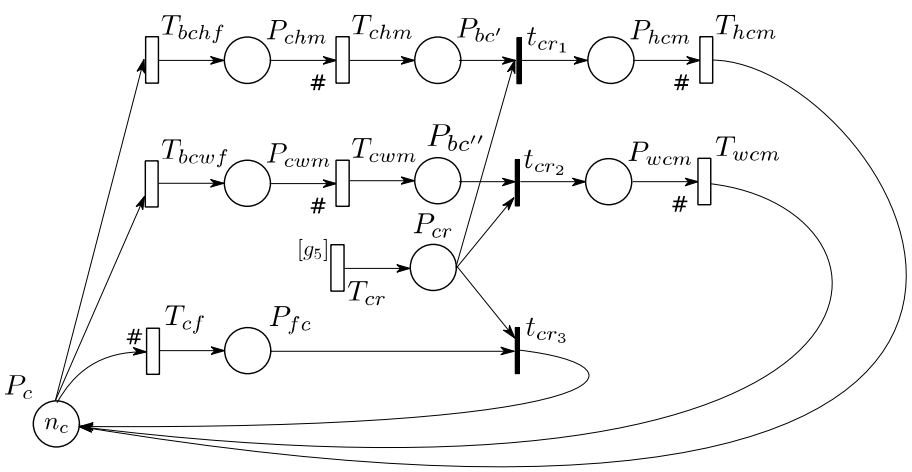
Monolithic Model

- Monolithic SRN model is automatically translated into CTMC or Markov Reward Model
- However the model not scalable as state-space size of this model is extremely large

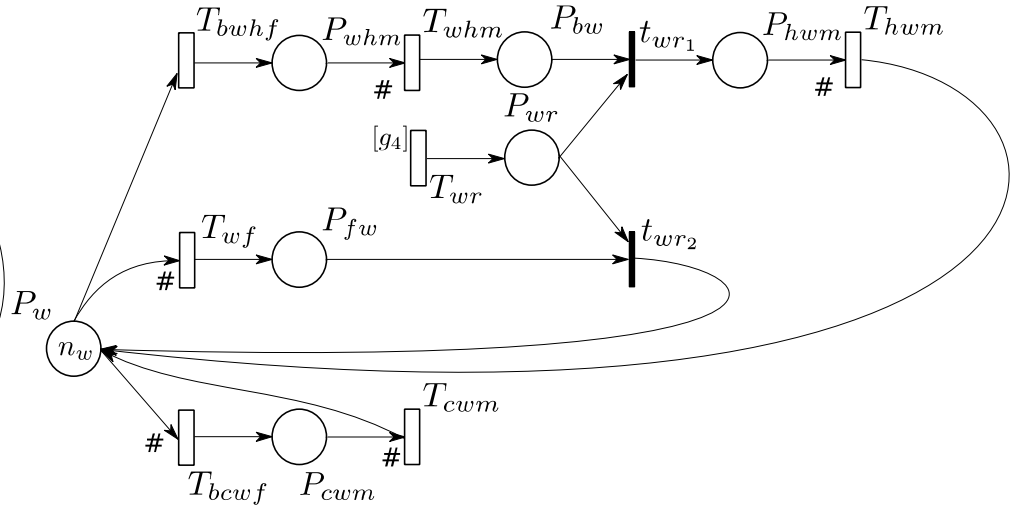
#PMs per pool	#states	#non-zero matrix entries
3	10, 272	59, 560
4	67,075	453, 970
5	334,948	2, 526, 920
6	1,371,436	11, 220, 964
7	4,816,252	41, 980, 324
8	Memory overflow	Memory overflow
10	-	-

Copyright © 2014 by K.S. Trivedi

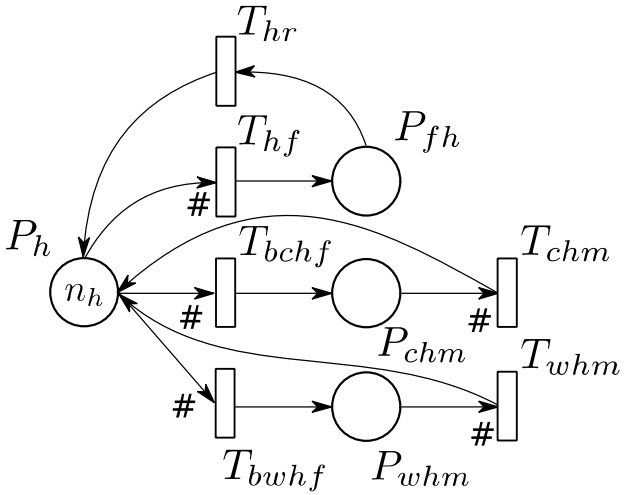
Decompose into Interacting Sub-models



SRN sub-model for cold pool



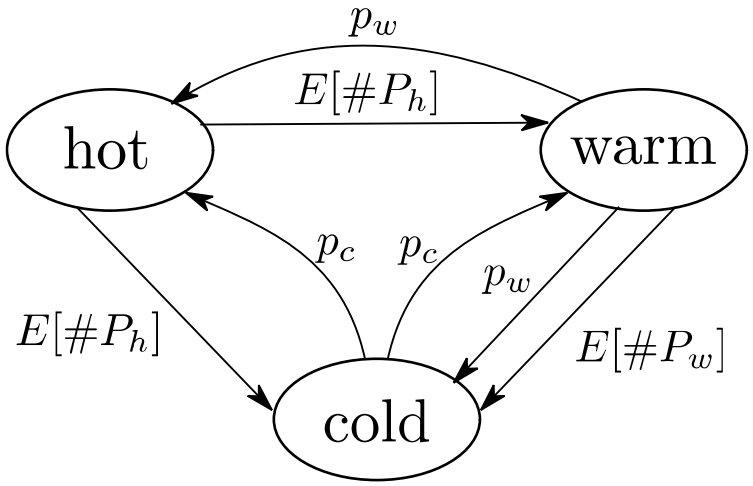
SRN sub-model for warm pool



SRN sub-model for hot pool

Copyright © 2014 by K.S. Trivedi

Import graph and model outputs



- **Model outputs:**
  - mean number of PMs in each pool ($E[\#P_h]$, $E[\#P_w]$, and $E[\#P_c]$)
  - Downtime in minutes per year

Copyright © 2014 by K.S. Trivedi

Many questions

- Existence of Fixed Point (easy)
- Uniqueness
- Rate of convergence
- Accuracy
- Scalability

Copyright © 2014 by K.S. Trivedi

Monolithic vs. interacting sub-models

- #states, #non-zero entries

n	Monolithic model		Interacting sub-models	
	#states	#non-zero entries	#states	#non-zero entries
3	10,272	59,560	196	588
4	67,075	453,970	491	1,768
5	334,948	2,526,920	1,100	4,518
6	1,371,436	11,220,964	2,262	10,272
7	4,816,252	41,980,324	3,770	18,434
8	Memory overflow	Memory overflow	6,939	36,316
10	-	-	20,460	118,710
20	-	-	21,273	106,300
40	-	-	271,543	1,481,000
60	-	-	1,270,813	7,148,100
80	-	-	3,859,083	22,051,600
100	-	-	9,196,353	53,055,500

Copyright © 2014 by K.S. Trivedi

Monolithic vs. interacting sub-models

- Downtime [minutes per year]
  - k is the #PM in hot pool to have the Cloud available
  - results differ only after the 8th significant figure (not reported in table)

n	k	Monolithic m.	Interacting s-m.
7	7	3,664.527	3,664.527
	6	10.978	10.978
	5	0.018	0.018
6	6	3,142.591	3,142.591
	5	7.847	7.847
	4	0.010	0.010
5	5	2,620.134	2,620.134
	4	5.235	5.235
	3	0.005	0.005
4	4	2,097.154	2,097.154
	3	3.143	3.143
	2	0.002	0.002
3	3	1,573.651	1,573.651
	2	1.572	1.572
	1	0.0005	0.0005

Copyright © 2014 by K.S. Trivedi

Interacting sub-models vs. simulation

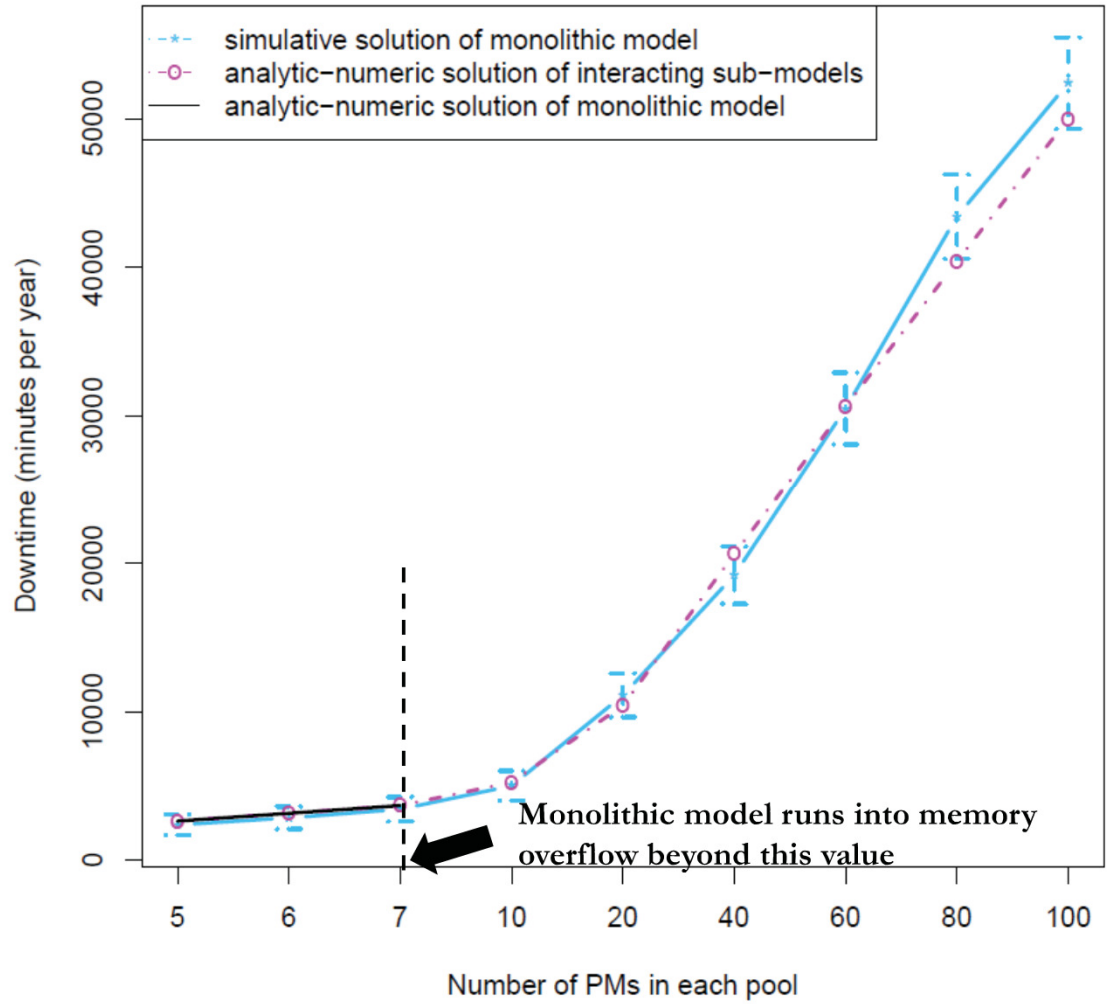
- Mean number of non-failed PMs
  - n is the initial #PMs in each pool
- Numeric solution of interacting sub-models in the c.i. of simulation solution

n	Interacting sub-models			Simulation		
	hot	warm	cold	hot	warm	cold
5	4.9950	4.9576	4.9824	4.9955, [4.9942, 4.9968]	4.9650, [4.9613, 4.9687]	4.9868, [4.9845, 4.9891]
10	9.9900	9.9134	9.9645	9.9905, [9.9886, 9.9924]	9.9269, [9.9215, 9.9323]	9.9687, [9.9652, 9.9722]
20	19.980	19.818	19.927	19.979, [19.976, 19.982]	19.823, [19.815, 19.832]	19.933, [19.928, 19.938]
40	39.960	39.593	39.846	39.963, [39.959, 39.966]	39.607, [39.593, 39.621]	39.849, [39.841, 39.857]
60	59.940	59.301	59.756	59.939, [59.935, 59.944]	59.298, [59.278, 59.317]	59.758, [59.748, 59.769]
80	79.920	78.896	79.655	79.910, [79.904, 79.916]	78.913, [79.886, 78.940]	79.656, [79.643, 79.669]
100	99.900	98.275	99.540	99.891, [99.884, 99.897]	98.297, [98.258, 99.335]	99.534, [99.519, 99.550]

Copyright © 2014 by K.S. Trivedi

Analytic-Numeric vs. simulative solutions

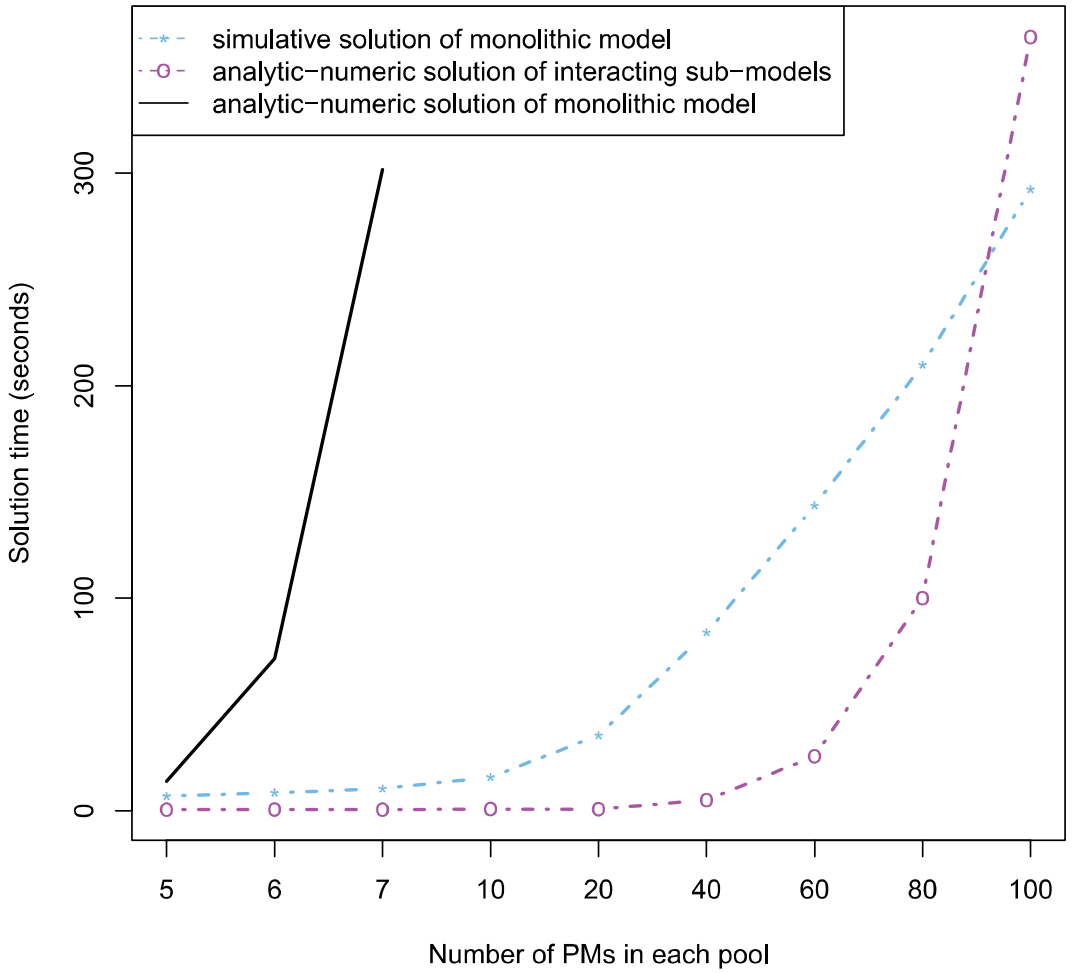
- Downtime [minutes per year]



Copyright © 2014 by K.S. Trivedi

Analytic-Numeric vs. simulative solution

- Solution times [seconds]



Copyright © 2014 by K.S. Trivedi

Availability Model Summary

Comparison index	Metric	Configuration	Monolithic model	Interacting sub-models	Simulation
Accuracy	Downtime (sec)	7 PMs per pool	3664.527	3664.527	3416.4
		100 PMs per pool	-	49993	52402
Scalability	Max #PMs per pool	-	≤ 7	> 100	> 200
	#states	7 PMs per pool	$4,816,252$	$3,770$	-
	#states	100 PMs per pool	-	$9,196,353$	-
Efficiency	Solution time (sec)	7 PMs per pool	301.700	0.584	10.434
		100 PMs per pool	-	364.025	294.346

Copyright © 2014 by K.S. Trivedi

Performance Modeling and Analysis for IaaS Cloud

[paper in Proc. IEEE PRDC 2010; FGCS 2013]

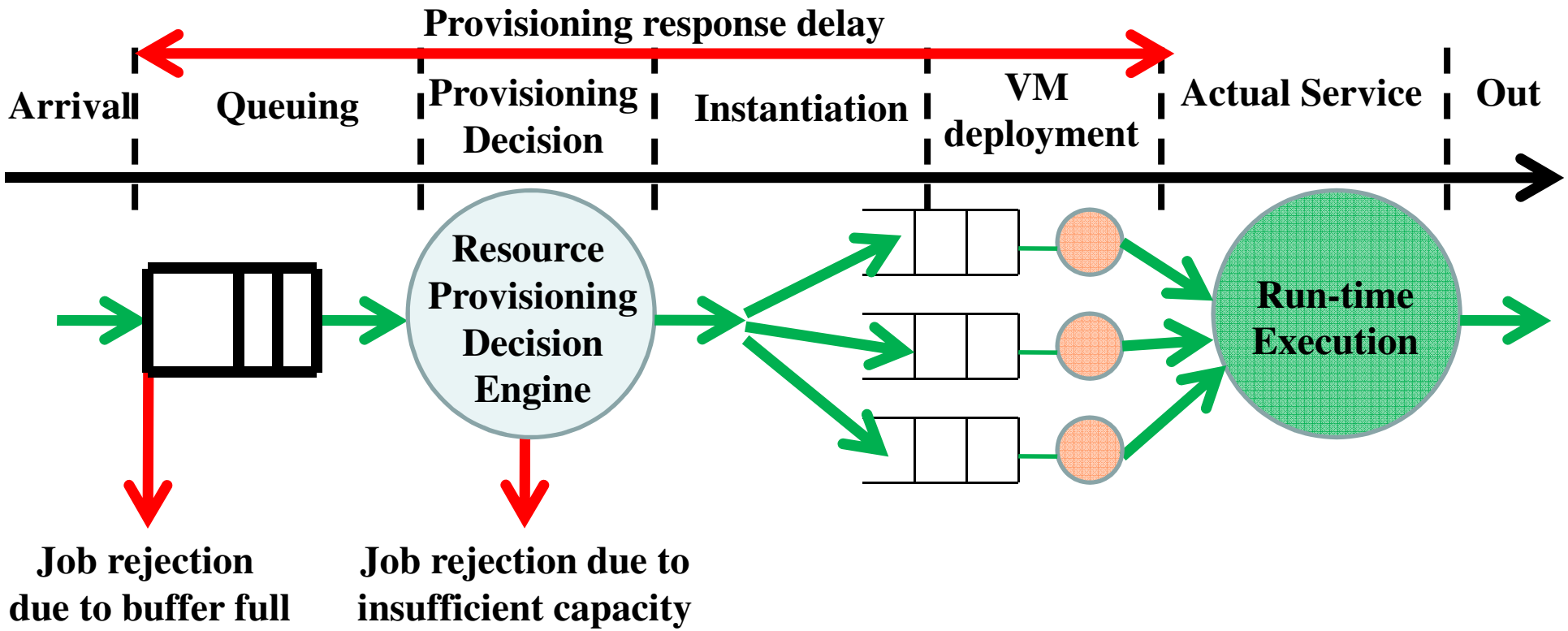
Copyright © 2014 by K.S. Trivedi

System model

- **Current Assumptions [will be relaxed soon]**
  - Homogenous requests
  - All physical machines (PMs) are identical.

Copyright © 2014 by K.S. Trivedi

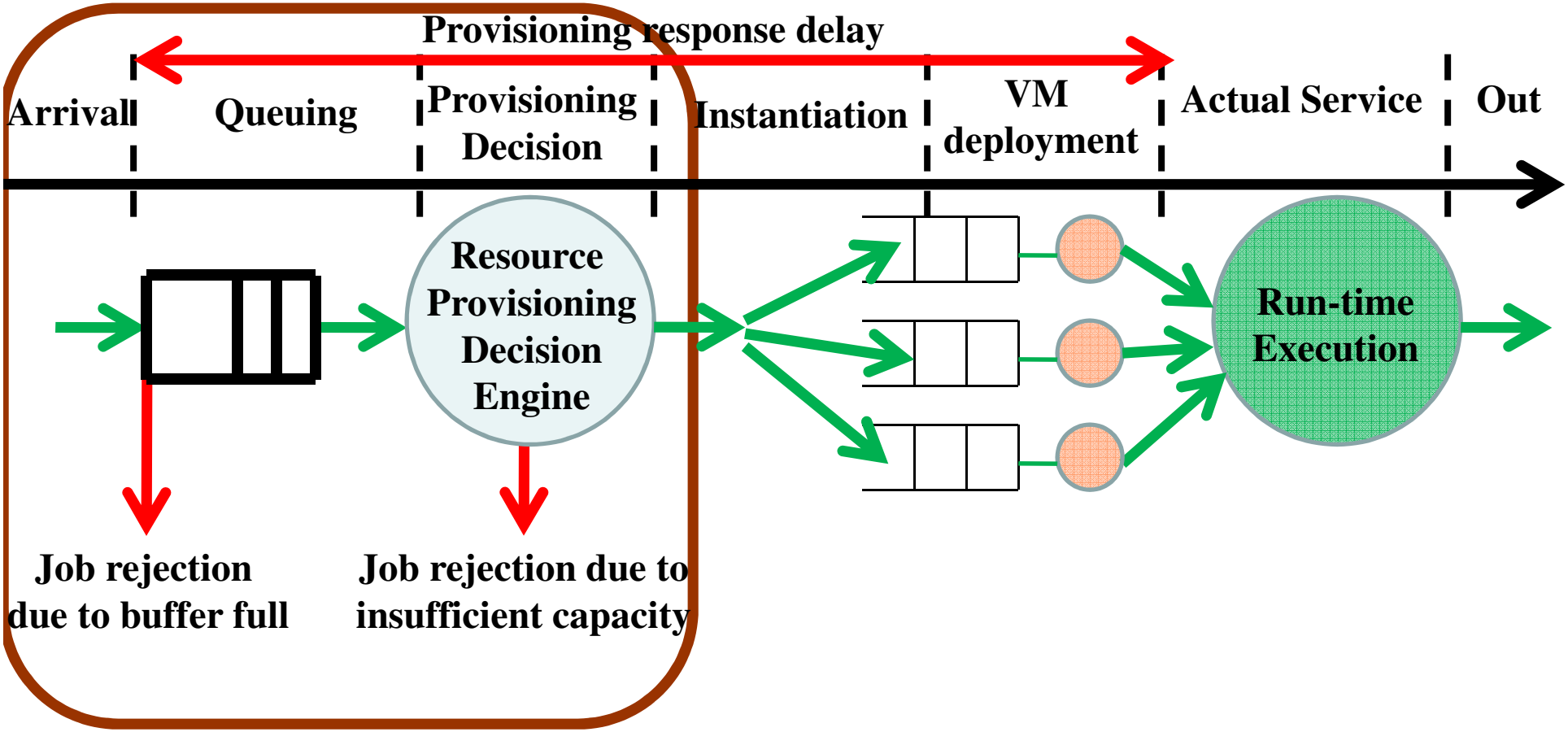
Life-cycle of a job inside a IaaS cloud



- **Provisioning and servicing steps:**
  - (i) resource provisioning decision,
  - (ii) VM provisioning and
  - (iii) run-time execution

Copyright © 2014 by K.S. Trivedi

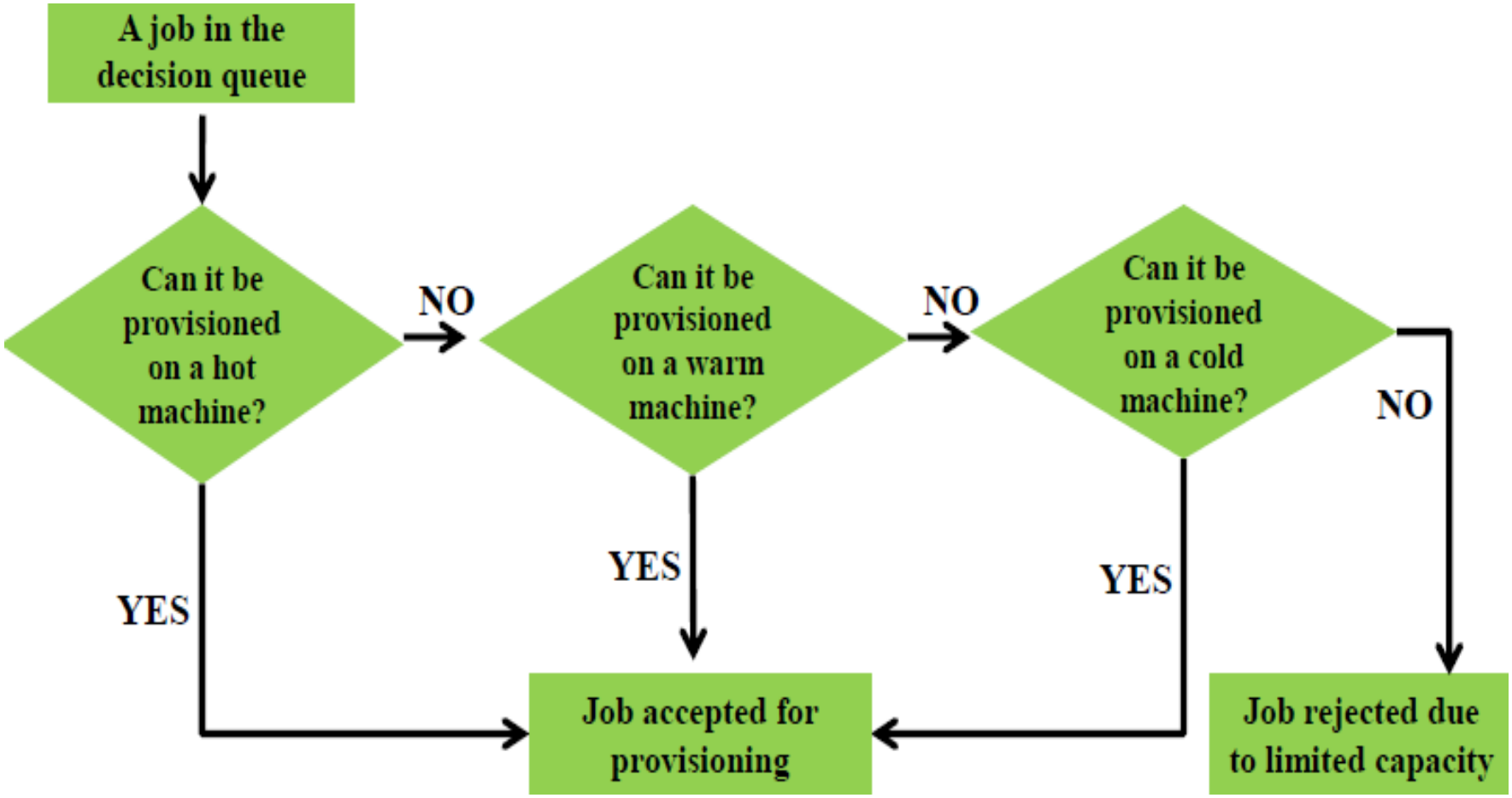
Resource provisioning decision engine (RPDE)



Copyright © 2014 by K.S. Trivedi

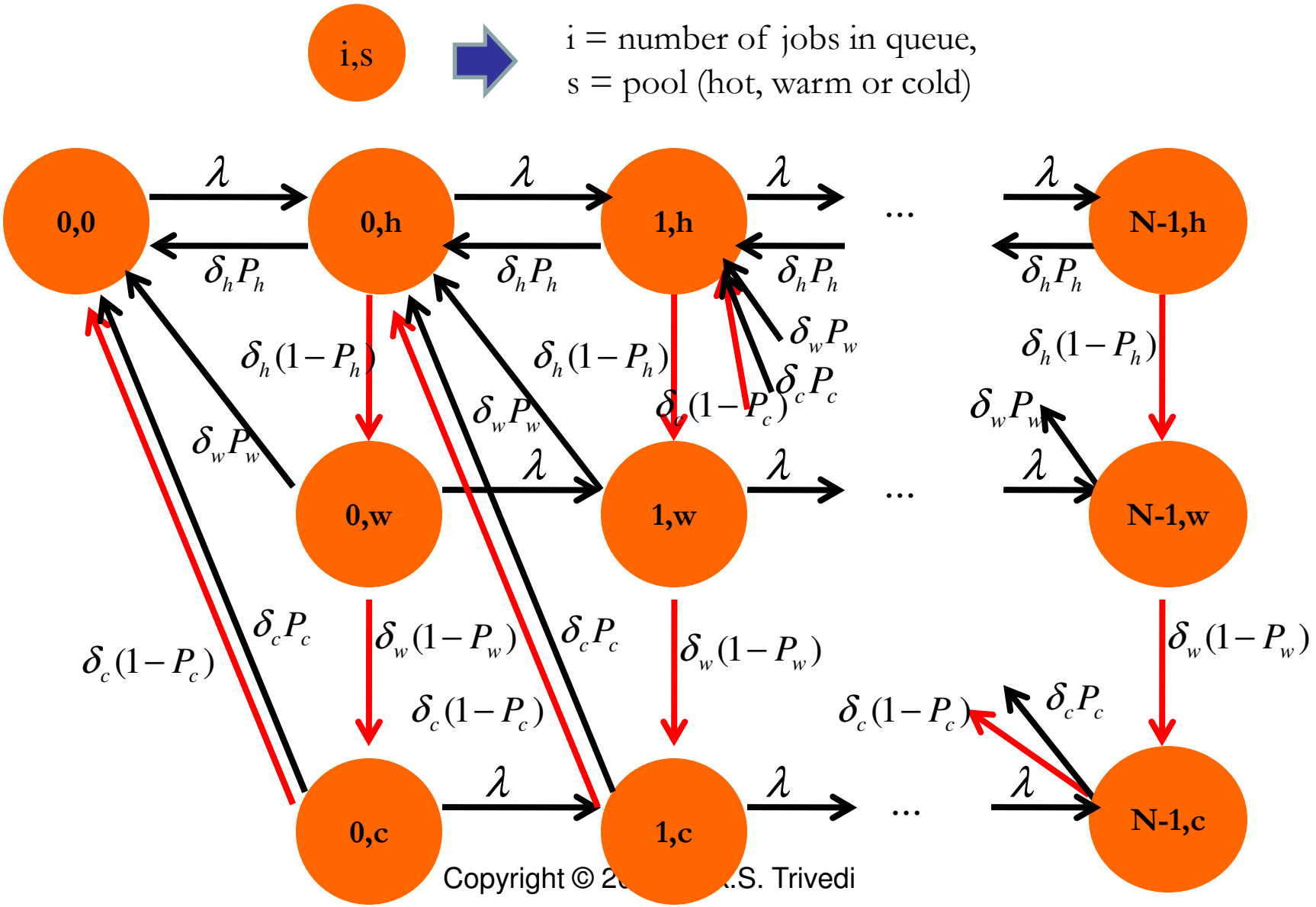
Resource provisioning decision engine (RPDE)

- Flow-chart:



Copyright © 2014 by K.S. Trivedi

CTMC model for RPDE



Copyright © 2 .S. Trivedi

Generator Matrix of the RPDE model

	0,0	0,h	0,w	0,c	1,h	1,w	1,c	2,h	2,w	2,c	$\cdots$	N–2,h	N–2,w	N–2,c	N–1,h	N–1,w	N–1,c
0,0	$*_0$	λ															
0,h	$\delta_h P_h$	$*_1$	$\delta_h(1-P_h)$		λ												
0,w	$\delta_w P_w$		$*_2$	$\delta_w(1-P_w)$		λ											
0,c	δ_c			$*_3$			λ										
1,h		$\delta_h P_h$			$*_1$	$\delta_h(1-P_h)$		λ									
1,w		$\delta_w P_w$				$*_2$	$\delta_w(1-P_w)$		λ								
1,c		δ_c					$*_3$			λ							
2,h					$\delta_h P_h$			$*_1$	$\delta_h(1-P_h)$		$\ddots$						
2,w					$\delta_w P_w$				$*_2$	$\delta_w(1-P_w)$	$\ddots$						
2,c					δ_c					$*_3$	$\ddots$						
$\vdots$											$\ddots$						
N–2,h												$*_1$	$\delta_h(1-P_h)$		λ		
N–2,w													$*_2$	$\delta_w(1-P_w)$		λ	
N–2,c														$*_3$			λ
N–1,h												$\delta_h P_h$			$*''_1$	$\delta_h(1-P_h)$	
N–1,w												$\delta_w P_w$				$*''_2$	$\delta_w(1-P_w)$
N–1,c												δ_c					$*''_3$

The generator matrix possesses significant structure and may be solved through matrix geometric method.

Copyright © 2014 by K.S. Trivedi

Closed form solution of RPDE sub-model

Let, $W = \dfrac{\lambda}{\delta_h(1-P_h)}$ $X = \dfrac{\delta+\lambda_c}{\delta_w(1-P_w)}$

$Y = \dfrac{\lambda+\delta_w}{\delta_h(1-P_h)}$ $Z = \dfrac{\lambda}{\delta_w(1-P_w)}$

It can be shown:

$$\pi_{(0,c)} = \frac{\lambda}{\delta_h P_h XY + \delta_w P_w X + \delta_c}\pi_{(0,0)} = A_{(0,c)}\pi_{(0,0)}$$

where,

$$A_{(0,c)} = \frac{\lambda}{\delta_h P_h XY + \delta_w P_w X + \delta_c}$$

Copyright © 2014 by K.S. Trivedi

Closed form solution of RPDE sub-model

Also, we can compute $\pi_{(0,w)}$ and $\pi_{(0,h)}$ w.r.t. $\pi_{(0,0)}$:

$$\pi_{(0,w)} = \frac{\lambda X}{\delta_h P_h XY + \delta_w P_w X + \delta_c}\pi_{(0,0)} = XA_{(0,c)}\pi_{(0,0)} = A_{(0,w)}\pi_{(0,0)}$$

$$\pi_{(0,h)} = \frac{\lambda XY}{\delta_h P_h XY + \delta_w P_w X + \delta_c}\pi_{(0,0)} = XYA_{(0,c)}\pi_{(0,0)} = A_{(0,h)}\pi_{(0,0)}$$

where,

$$A_{(0,w)} = \frac{\lambda X}{\delta_h P_h XY + \delta_w P_w X + \delta_c} = XA_{(0,c)}$$

$$A_{(0,h)} = \frac{\lambda XY}{\delta_h P_h XY + \delta_w P_w X + \delta_c} = XYA_{(0,c)} = YA_{(0,w)}$$

Copyright © 2014 by K.S. Trivedi

Closed form solution of RPDE sub-model

Similarly, other state probabilities can be derived in terms of $\pi_{(0,0)}$

$$\pi_{(k,h)} = A_{(k,h)}\pi_{(0,0)} \qquad \pi_{(k,w)} = A_{(k,w)}\pi_{(0,0)} \qquad \pi_{(k,c)} = A_{(k,c)}\pi_{(0,0)}$$

Where,

$$A_{(-1,h)} = 1$$

$$A_{(k,w)} = XA_{(k,c)} - ZA_{(k-1,c)}$$

$$A_{(k,h)} = YA_{(k,w)} - WA_{(k-1,w)}$$

$$\text{with } 1 \leq k \leq (N-2)$$

$$A_{(k,c)} = \frac{(\lambda+\delta_h)A_{(k-1,h)}}{\delta_h P_h XY + \delta_w P_w X + \delta_c} + \frac{\delta_h P_h YZA_{(k-1,c)} + \delta_h P_h WA_{(k-1,w)}}{\delta_h P_h XY + \delta_w P_w X + \delta_c} + \frac{\delta_w P_w ZA_{(k-1,c)} - \lambda A_{(k-2,h)}}{\delta_h P_h XY + \delta_w P_w X + \delta_c}$$

Finally, normalization is provided by:

$$\pi_{(0,0)} + \sum_{i=0}^{N-1}\left(\pi_{(i,h)} + \pi_{(i,w)} + \pi_{(i,c)}\right) = 1$$

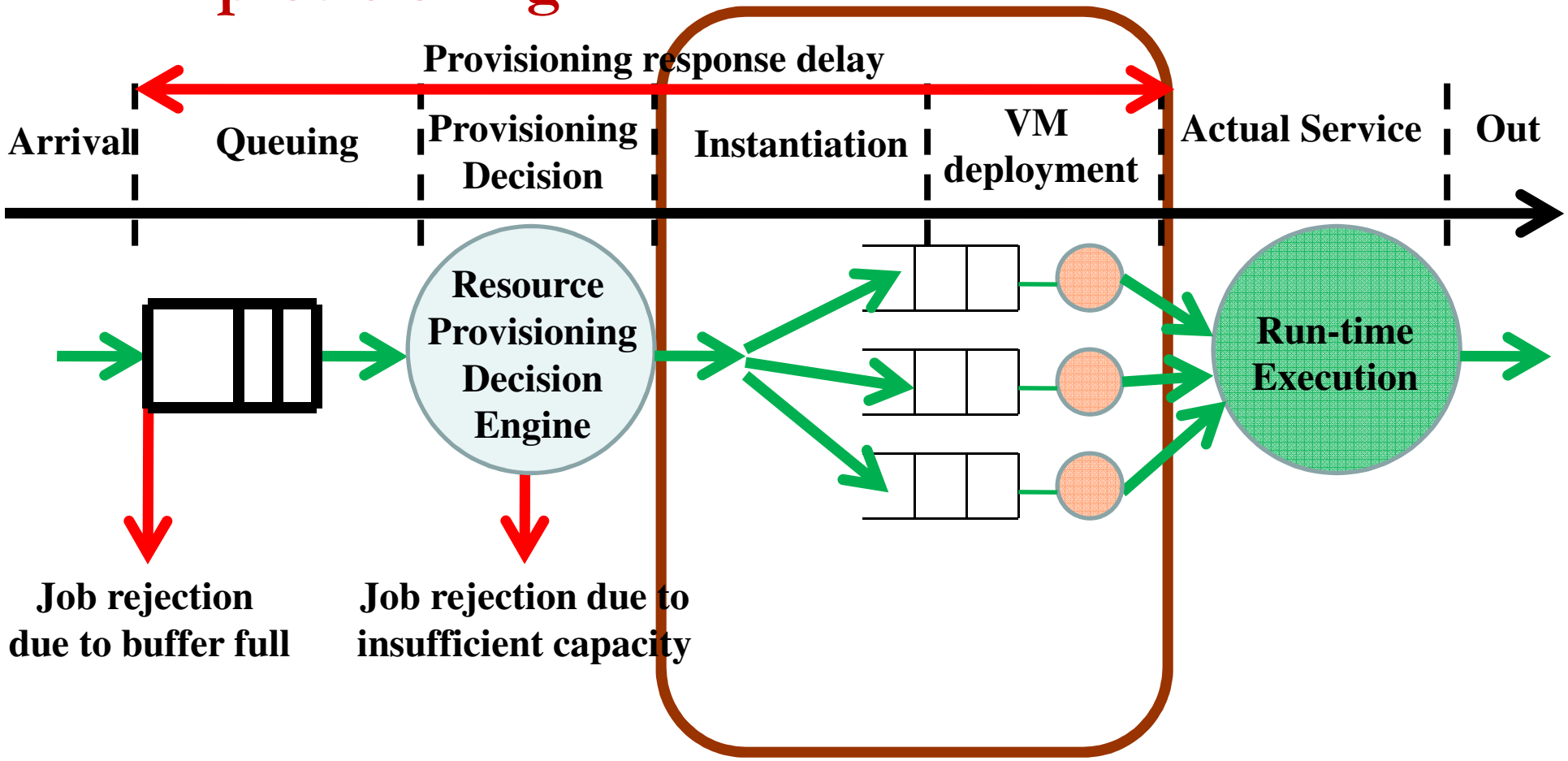
Copyright © 2014 by K.S. Trivedi

RPDE model: parameters & measures

- **Input Parameters:**
  - λ –arrival rate: data collected from cloud
  - $1/\delta_h, 1/\delta_w, 1/\delta_c$ – mean search delays for resource provisioning decision engine: from searching algorithms or measurements
  - P_h, P_w, P_c – probability of being able to provision: computed from VM provisioning model
  - N – maximum # jobs in RPDE: from system/server specification
- **Output Measures:**
  - Job rejection probability due to buffer full (P_{block})
  - Job rejection probability due to insufficient capacity (P_{drop})
  - Sum of the above two is the overall rejection probability (ϱ_{reject})
  - Mean decision delay for an accepted job ($E[T_{decision}]$)
  - Mean queuing delay for an accepted job ($E[T_{q\_dec}]$)

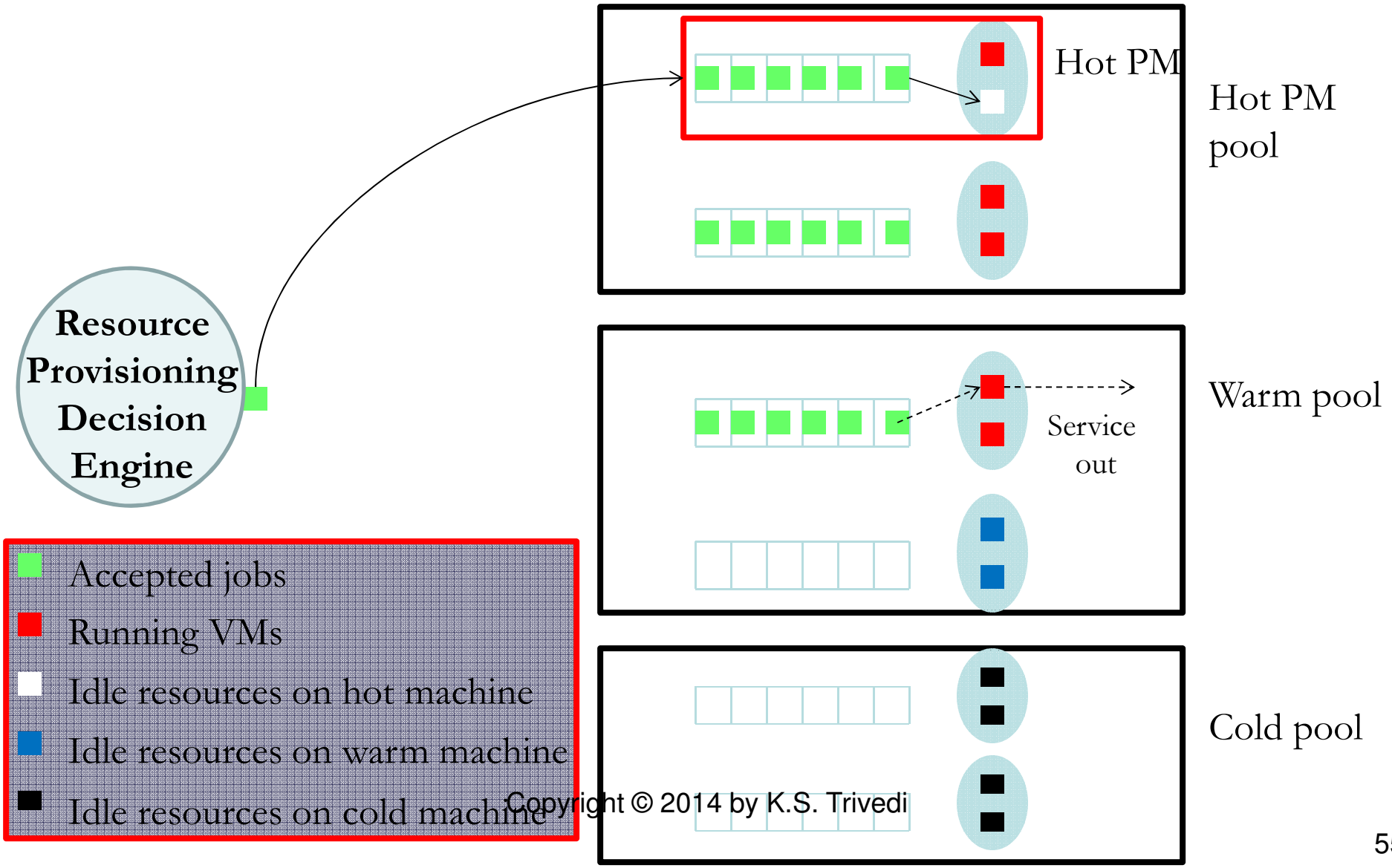
Copyright © 2014 by K.S. Trivedi

VM provisioning



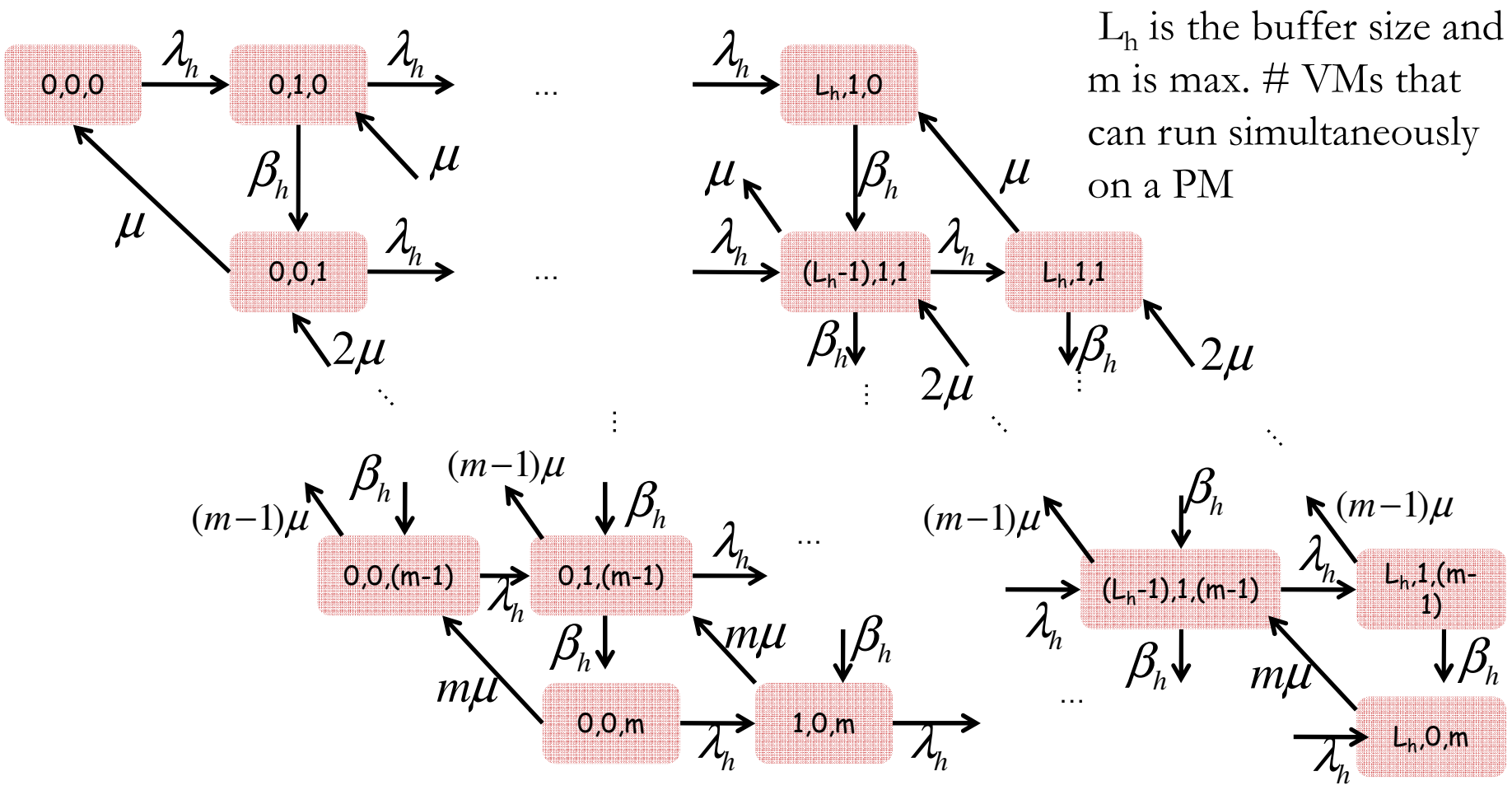
Copyright © 2014 by K.S. Trivedi

VM provisioning model



Copyright © 2014 by K.S. Trivedi

VM provisioning model for each hot PM



L_h is the buffer size and m is max. # VMs that can run simultaneously on a PM

i,j,k → i = number of jobs in the queue, j = number of VMs being provisioned, k = number of VMs running

Copyright © 2014 by K.S. Trivedi

Generator Matrix of the hot PM model

	000	001	002	003	010	011	012	103	110	111	112	203	210	211	212	303	310	311	312
000	$*_0$				λ														
001	μ	$*_{01}$				λ													
002		2μ	$*_{02}$				λ												
003			3μ	$*_{03}$				λ											
010		β			$*_{10}$				λ										
011			β		μ	$*_{11}$				λ									
012				β		2μ	$*_{12}$				λ								
103							3μ	$*_{13}$				λ							
110						β			$*_{10}$				λ						
111							β		μ	$*_{11}$				λ					
112								β		2μ	$*_{12}$				λ				
203											3μ	$*_{13}$				λ			
210										β			$*_{10}$				λ		
211											β		μ	$*_{11}$				λ	
212												β		2μ	$*_{12}$				λ
303															3μ	$*_{13}$			
310														β			$*_{20}$		
311															β		μ	$*_{21}$	
312																β		2μ	$*_{22}$

The generator matrix when $L_h = 3$ and $m = 3$. It possesses a block structure that facilitates a matrix geometric solution.

Copyright © 2014 by K.S. Trivedi

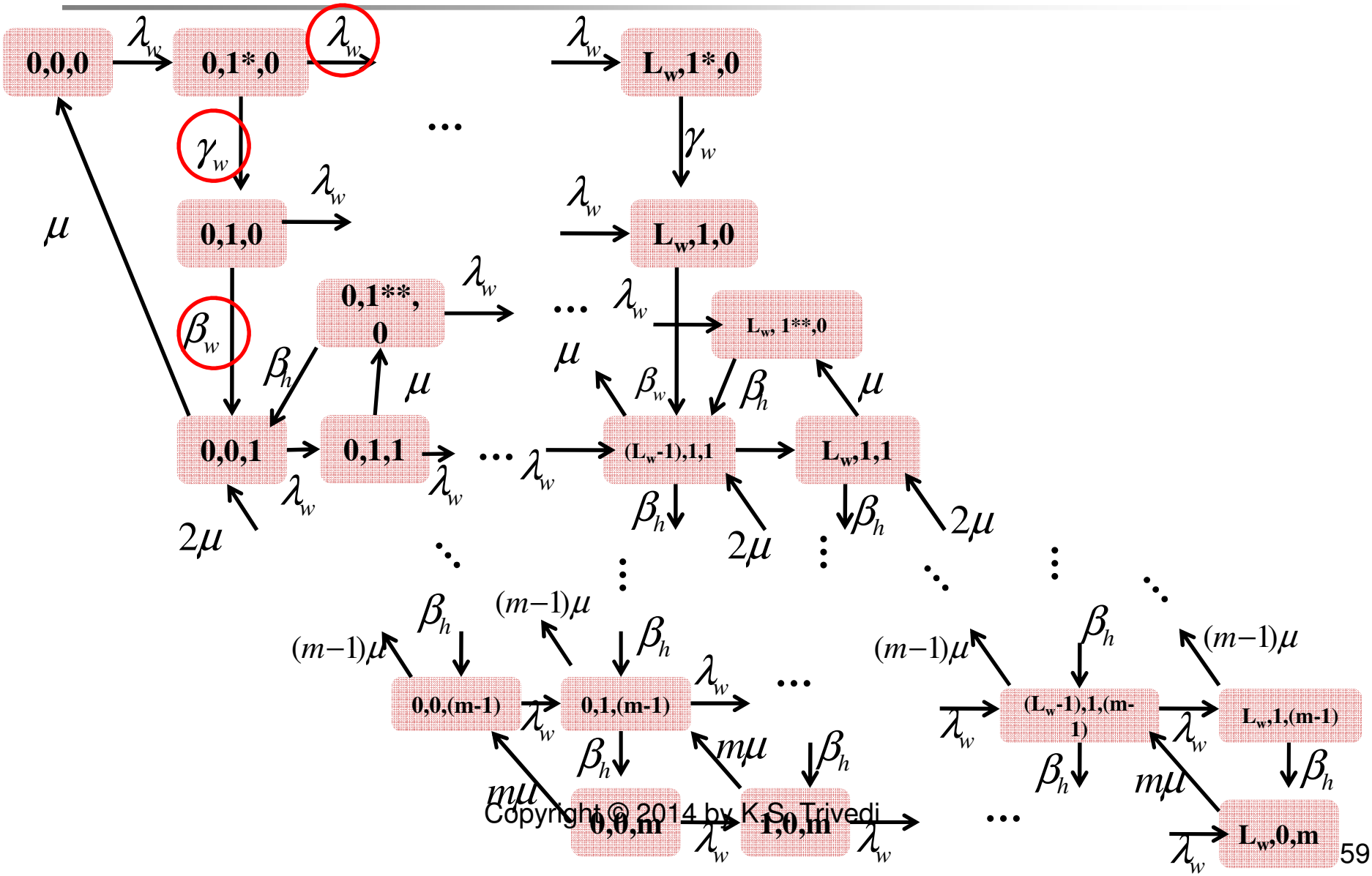
VM provisioning model (for each hot PM)

- Input Parameters:
  - $\lambda_h = \dfrac{\lambda(1-P_{block})}{n_h}$
  - $1/\beta_h$ can be measured experimentally
  - $1/\mu$ obtained from the lower level run-time model
  - P_{block} obtained from the resource provisioning decision model
- Hot pool model is the set of n_h independent hot PM models
- Output Measure:
  - P_h = prob. that a job is accepted in the hot pool $= 1-(\sum_{i=0}^{m-1} \varphi^{(h)}_{(L_h,1,i)} + \varphi^{(h)}_{(L_h,0,m)})^{n_h}$

where, $(\sum_{i=0}^{m-1} \varphi_{(L_h,1,i)}^{(h)} + \varphi_{(L_h,0,m)}^{(h)})$ is the steady state probability that a PM can not accept job for provisioning - from the solution of the Markov model of a hot PM on the previous slide

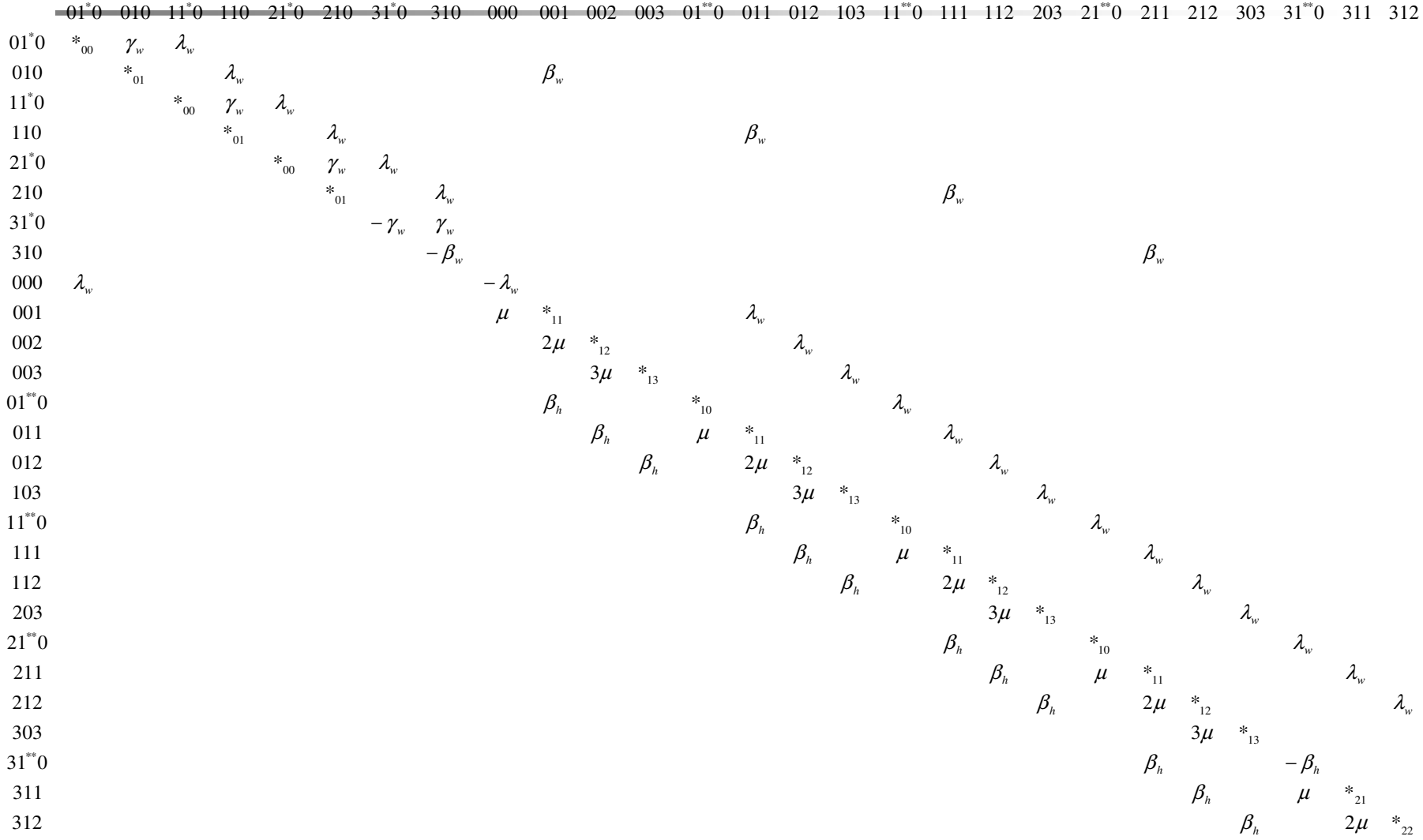
Copyright © 2014 by K.S. Trivedi

VM provisioning model for each warm PM



Copyright © 2014 by K. S. Trivedi

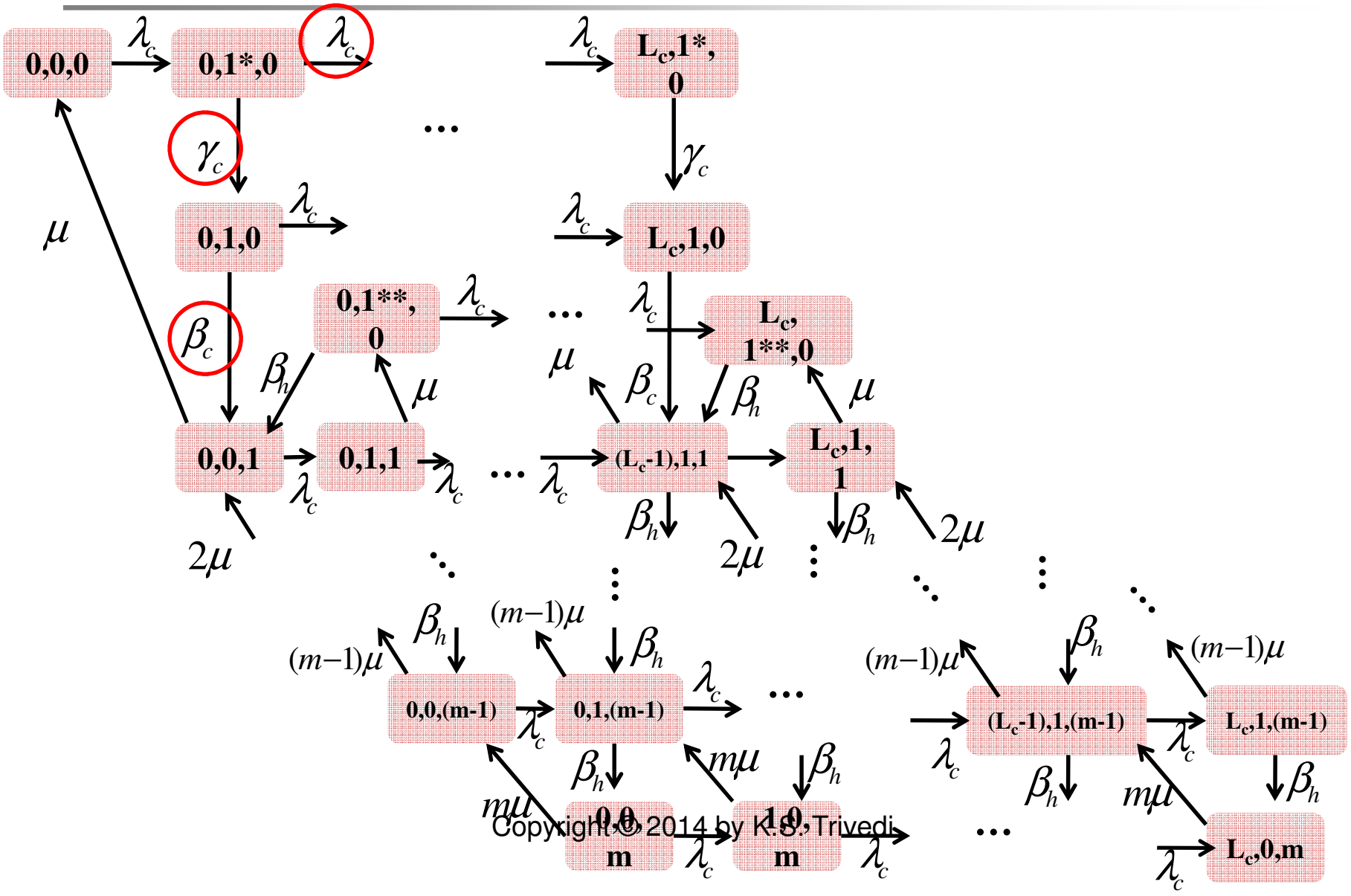
Generator Matrix for warm pool model



$L_w = 3$ and $m = 3$. The matrix may be solved using matrix-analytical method.

Copyright © 2014 by K.S. Trivedi

VM provisioning model for each cold PM



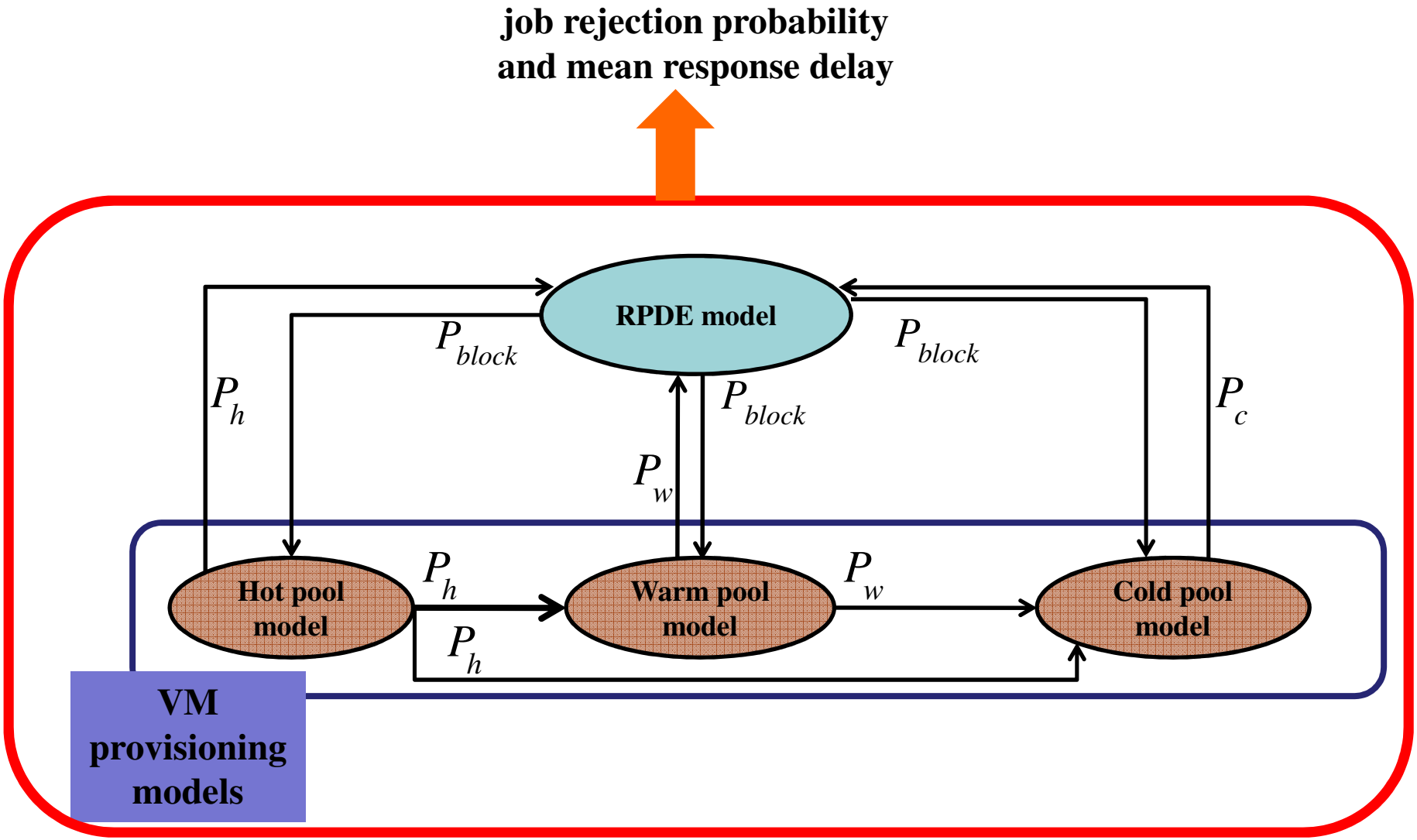
Copyright © 2014 by K.S. Trivedi

VM provisioning model: Summary

- Warm/cold PM model is similar to hot PM, except:
  - Effective job arrival rate
  - For first job, warm/cold PM requires additional start-up time
  - Mean provisioning delay for a VM for the first job is longer
- Outputs of hot, warm and cold pool models:
  - Probabilities (P_h, P_w, P_c) that at least one PM in hot/warm/cold pool can accept a job

Copyright © 2014 by K.S. Trivedi

Import graph for performance models



Copyright © 2014 by K.S. Trivedi

Fixed-point iteration

- To solve hot, warm and cold PM models, we need P_{block} from resource provisioning decision model
- To solve provisioning decision model, we need P_h, P_w, P_c from hot, warm and cold pool model respectively
- This leads to a cyclic dependency among the resource provisioning decision model and VM provisioning models (hot, warm, cold)
- We resolve this dependency via fixed-point iteration
- Observe, our fixed-point variable is P_{block} and corresponding fixed-point equation is of the form:

$$P_{block} = f(P_{block})$$

Copyright © 2014 by K.S. Trivedi

Many questions

- Existence of Fixed Point (easy)
- Uniqueness
- Rate of convergence
- Accuracy
- Scalability

Copyright © 2014 by K.S. Trivedi

Performance measures comparison with monolithic model

- 1 PM per pool and 1 VM per PM

Jobs/hr	Mean RPDE queue length		Rejection probability	
	IMC	monolithic	IMC	Monolithic
1	9.0332e-07	9.2321e-07	9.8899e-06	1.1221e-03
5	4.1622e-05	4.3364e-05	4.2334e-02	8.0500e-02
10	2.3731e-04	2.4225e-04	2.3496e-01	2.6587e-01
15	6.3539e-04	6.4377e-04	3.9860e-01	4.1493e-01
20	1.2526e-03	1.2655e-03	5.1069e-01	5.1969e-01
25	2.0990e-03	2.1179e-03	5.8915e-01	5.9449e-01
30	3.1826e-03	3.2091e-01	6.4648e-01	6.4985e-01
35	4.5106e-03	4.5462e-03	6.8999e-01	6.9223e-01

- The error is between e-03 and e-07 for all the results.
- The number of states in monolithic model is 912 while in ISP model it is 21

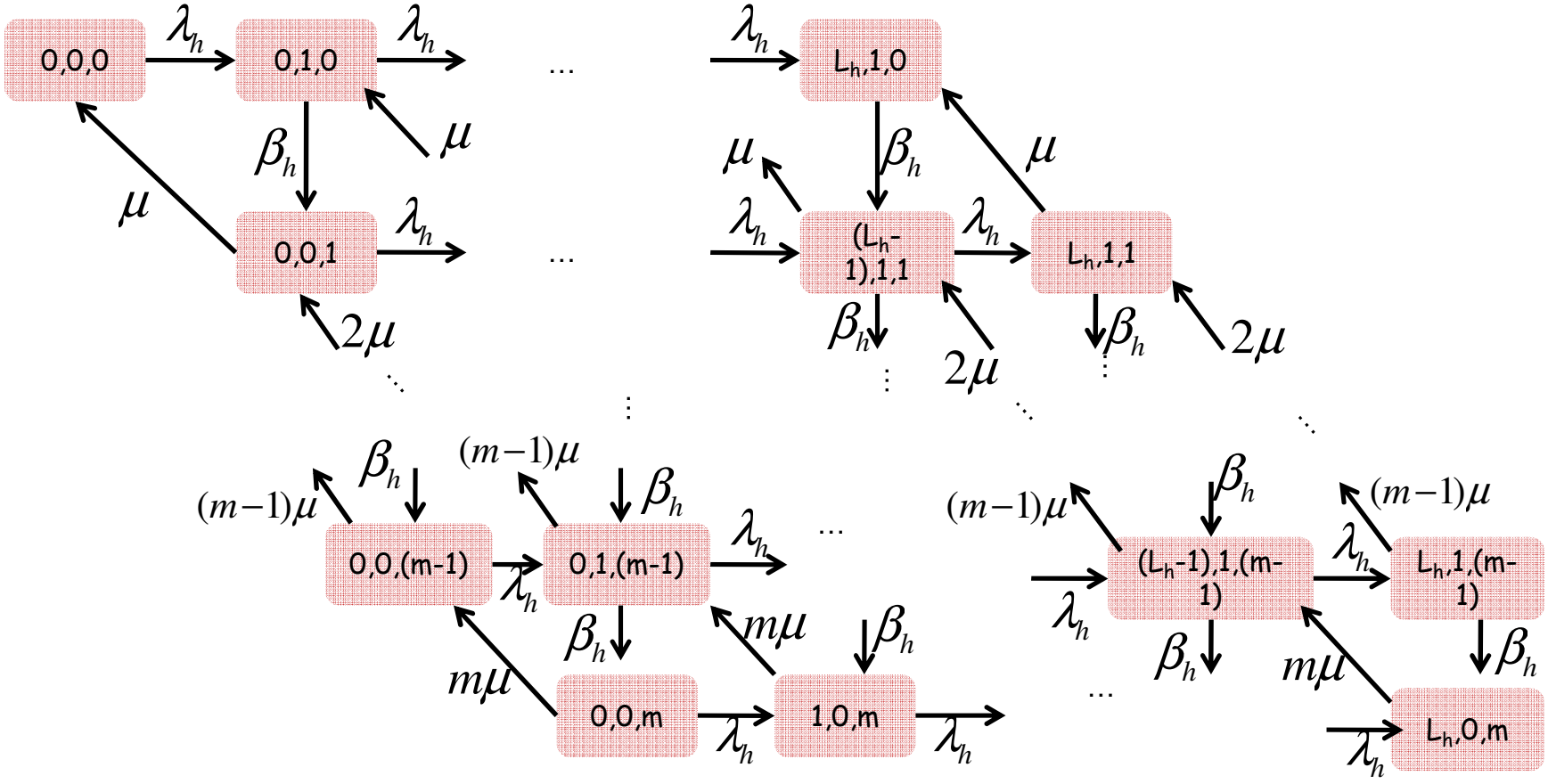
Copyright © 2014 by K.S. Trivedi

Power Quantification for IaaS Cloud

[paper in Proc. IEEE/IFIP DSN workshop DCDV 2011]

Copyright © 2014 by K.S. Trivedi

Power Consumption from Hot PM Model



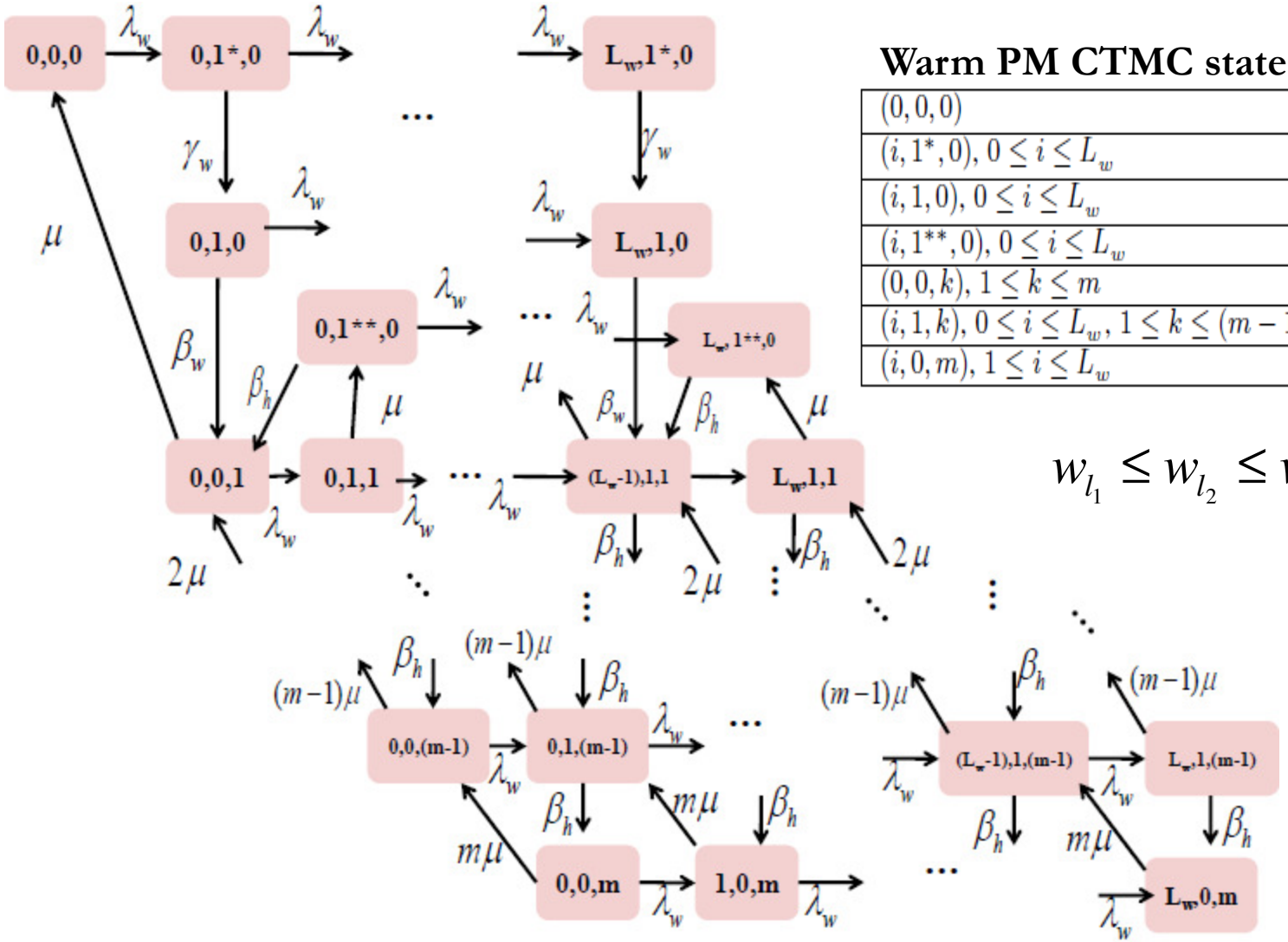
Hot PM idle power consumption (no VM): h_l

Additional power consumption of each running VM with average resource utilization: v_a

For each state (i, j, k) of the CTMC, we assign a reward rate: $r(i, j, k) = h_l + kv_a$

Copyright © 2014 by K.S. Trivedi

Power Consumption from Warm PM Model

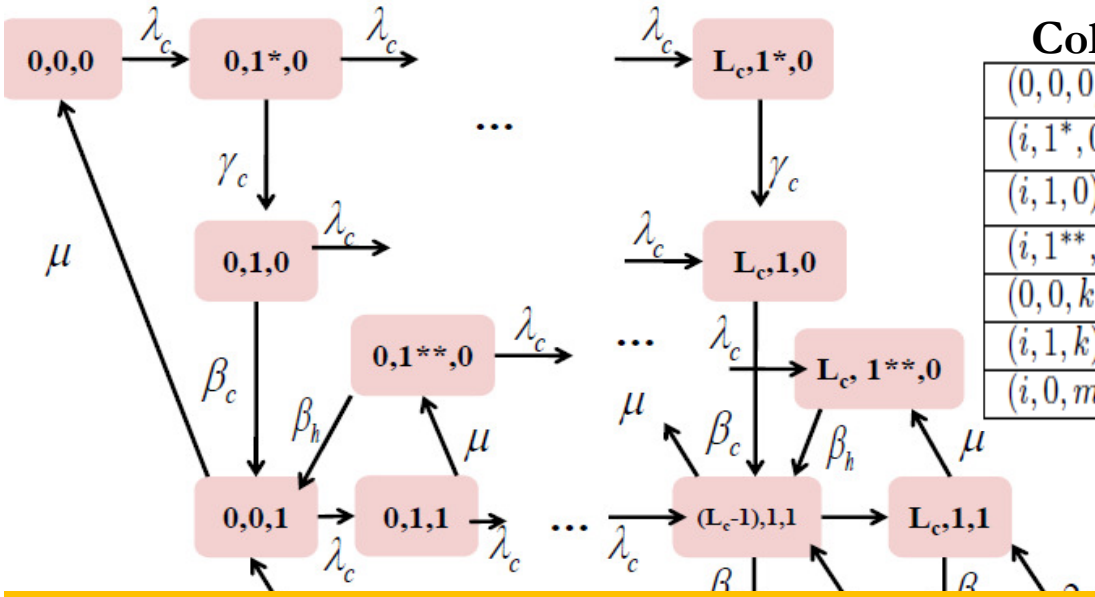


Warm PM CTMC states	Reward rates
$(0,0,0)$	w_{l_1}
$(i,1^*,0),\ 0 \le i \le L_w$	w_{l_2}
$(i,1,0),\ 0 \le i \le L_w$	w_{l_3}
$(i,1^{**},0),\ 0 \le i \le L_w$	h_l
$(0,0,k),\ 1 \le k \le m$	$h_l + kv_a$
$(i,1,k),\ 0 \le i \le L_w,\ 1 \le k \le (m-1)$	$h_l + kv_a$
$(i,0,m),\ 1 \le i \le L_w$	$h_l + mv_a$

$$w_{l_1} \le w_{l_2} \le w_{l_3} \le h_l$$

Copyright © 2014 by K.S. Trivedi

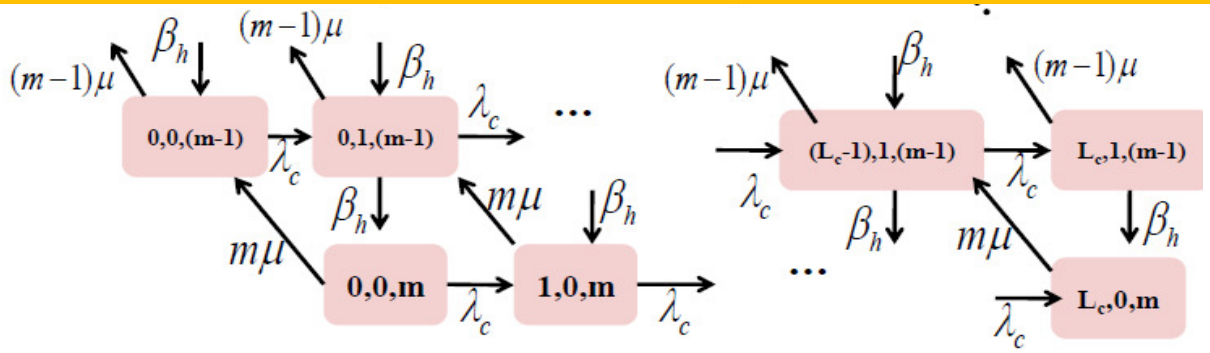
Power Consumption from Cold PM Model



Cold PM CTMC states	Reward rates
$(0,0,0)$	c_{l_1}
$(i,1^*,0),\ 0 \le i \le L_c$	c_{l_2}
$(i,1,0),\ 0 \le i \le L_c$	c_{l_3}
$(i,1^{**},0),\ 0 \le i \le L_c$	h_l
$(0,0,k),\ 1 \le k \le m$	$h_l + kv_a$
$(i,1,k),\ 0 \le i \le L_c,\ 1 \le k \le (m-1)$	$h_l + kv_a$
$(i,0,m),\ 1 \le i \le L_c$	$h_l + mv_a$

$$c_{l_1} \le c_{l_2} \le c_{l_3} \le h_l$$

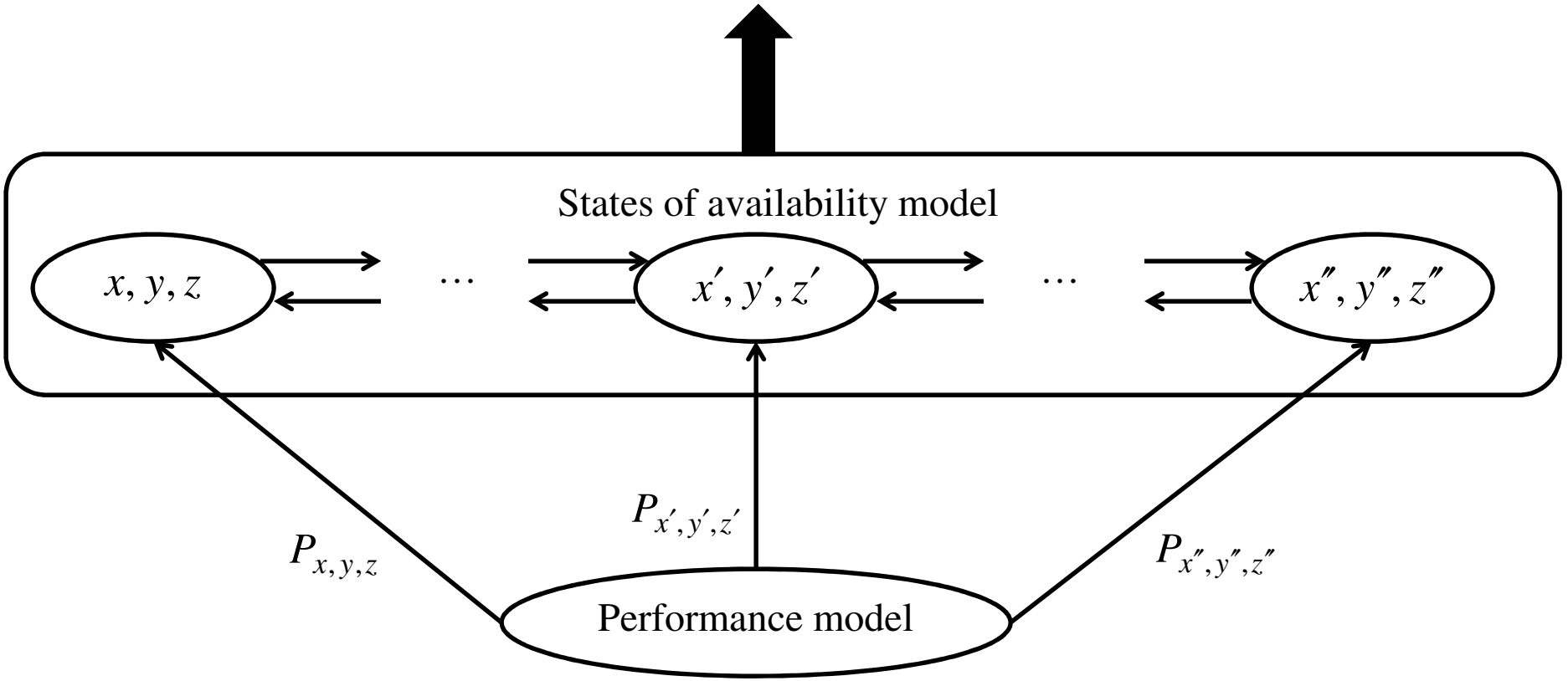
Net power consumption is sum of power consumptions in hot, warm and cold pool



Copyright © 2014 by K.S. Trivedi

Power and cooling cost

Overall power consumption and cooling cost as expected steady state reward rates



Conclusions

Copyright © 2014 by K.S. Trivedi

Conclusions

- Analytic models are powerful for the construction and numerical solution of various reliability, availability, performance, and power models
- For very complex systems such as clouds, hierarchical, fixed-point iterative and approximate solutions are needed.
  - Performance, availability and power consumption analysis can be done using such an approach
- Simulative and hybrid models/solutions should be used when absolutely necessary
- Models can then be used in
  - capacity planning
  - a feedback control setting for adapting to changes

Copyright © 2014 by K.S. Trivedi

Summary

- Performance analysis:
  - ✓ Developed scalable interacting stochastic sub-models for large Clouds
  - ✓ Analysis of provisioning delay and impact of different factors, e.g., arrival rate, system capacity, resource holding time etc.
    - R. Ghosh, F. Longo, V. K. Naik, and K. S. Trivedi, "Modeling and Performance Analysis of Large Scale IaaS Clouds", Elsevier Future Generation Computing Systems, July 2013.

Copyright © 2014 by K.S. Trivedi

Summary (contd.)

- Scalable Availability analysis:
  - ✓ Interacting stochastic sub-models for failure-repair analysis
    - F. Longo, R. Ghosh, V. K. Naik, and K. S. Trivedi, "A Scalable Availability Model for Infrastructure-as-a-Service Cloud", DSN, June 2011.
    - R. Ghosh, F. Longo, F. Frattini, S. Russo and K. S. Trivedi, "Scalable Analytics for IaaS Cloud Analytics", IEEE Trans. On Cloud Computing, accepted Feb 2014.

- Cost analysis, optimization and Cloud capacity planning:
  - ✓ Developed and solved optimization problems to minimize the total cost without violating the SLAs
    - R. Ghosh, F. Longo, R. Xia, V. K. Naik, and K. S. Trivedi, "Stochastic Model Driven Capacity Planning for an Infrastructure-as-a-Service Cloud", IEEE Trans. On Services Computing, accepted August 2013.

Copyright © 2014 by K.S. Trivedi

Thanks!

Copyright © 2014 by K.S. Trivedi