**RC25129 (W1103-127) March 21, 2011**
**Computer Science**

# IBM Research Report

## Power-Performance Trade-offs in IaaS Cloud: A Scalable Analytic Approach

**Rahul Ghosh[1], Vijay K. Naik[2], Kishor S. Trivedi[1]**

[1]Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708
USA

[2]IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598
USA

**Research Division**
**Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich**

# Power-Performance Trade-offs in IaaS Cloud: A Scalable Analytic Approach

Rahul Ghosh*, Vijay K. Naik† and Kishor S. Trivedi*
*Duke University, USA, †IBM T. J. Watson Research Center, USA
{rg51, kst}@ee.duke.edu*, vkn@us.ibm.com†

***Abstract*—Optimizing for performance is often associated with higher costs in terms of capacity, faster infrastructure, and power costs. In this paper, we quantify the power-performance trade-offs by developing a scalable analytic model for joint analysis of performance and power consumption for a class of Infrastructure-as-a-Service (IaaS) clouds with tiered service offerings. The tiered service offerings are provided by configuring physical machines into three pools with different response time and power consumption characteristics. Using interacting stochastic sub-models approach, we quantify power-performance trade-offs. We summarize our modeling approach and highlight key results on the effects of physical machine pool configurations on consumed power and achievable performance in terms of response time and ability to service requests. The approach developed here can be used to manage power consumption and performance by judiciously configuring physical machine pools.**

## I. Introduction

**Background and system description.** Recently, research efforts have been made to quantify the environmental impact of large IT infrastructure [1]. In [2], authors show how clouds can be used to reduce power consumptions even in office and home computing environments. In this paper, we describe a scalable analytic approach for joint analysis of performance and power consumption for a specific type of Infrastructure-as-a-Service (IaaS) cloud. In IaaS cloud (e.g., Amazon EC2 [3], IBM Smart Cloud Enterprise [4]), when a request is processed, a pre-built image is used to create one or more Virtual Machine (VM) instances [5]–[7]. When the VM instances are deployed on Physical Machines (PMs), they are provisioned with request specific CPU, RAM, and disk capacity. When provisioning a VM, the two primary sources of delay are: (i) PM and hypervisor readiness delays and (ii) network delays associated with image transfer. To minimize the effects of these delays on IaaS service performance, we consider a class of IaaS cloud system, where the physical machines are configured into different readiness state and VM instances are provisioned using multiple image pre-provisioning and caching strategies. For our analysis, *we assume that PMs are partitioned into three PM pools: hot (i.e., running), warm (turned on, but not ready) and cold (turned off).* A pre-instantiated VM can be provisioned and brought to ready state on hot PMs with minimum provisioning delay. Instantiating a VM from an image and provisioning it on a warm PM needs additional provisioning time. PMs in the cold pool need additional startup time to be turned on before a VM deployment. In the subsequent discussions we use the term "job" to mean a user request for provisioning a VM and making it available for use by a cloud user. The response time performance metric corresponds to the time elapsed from the time a user submits a request until the VM is available for the user to use. The service time corresponds to the time the VM is active on the PM. We assume that all requests are homogeneous and each request is for one VM with fixed size CPU cores and RAM. User requests (i.e., jobs) are submitted to a global resource provisioning decision engine (RPDE) that processes requests on a first-come, first-served (FCFS) basis as follows. The request at the head of the queue is provisioned on a hot PM if there is capacity to run a VM on one of the hot PMs. If no hot PM is available, a PM from the warm pool is used for provisioning the requested VM. If warm PMs are all busy, a PM from the cold pool is used. If none of these PMs are available, the request is rejected. When a running job exits, the capacity used by that VM is released and becomes available for provisioning the next job. Note that in this type of cloud, the partitioning of PMs into tiered readiness states gives operators a handle on the energy usage when the system is not saturated, but it can also result in response time delays when there is a mismatch between the size of the partitions and the demand, especially when there is large variation in the arrival rates.

**Problem Statement.** By varying sizes of hot, warm, and cold pools, we analyze the trade-offs in maintaining a PM in a hot pool vs in a warm or a cold pool. Partitioning the PMs in three pools allows for an optimization by reducing provider's costs without large provisioning delays for *all* VMs. Intuitively, having more PMs in warm or cold pools can reduce power consumption and operational costs, at the cost of potentially increasing response time delays. In this paper, we quantify the power-performance relations using stochastic analytic models and observe that intuition-based classification of PMs does not always lead to desired results.

**Key contributions.** Based on the scalable interacting stochastic models approach, as described in our previous work [6], [7], we make the following contributions in this paper: (1) using Markov reward approach [8], we show how power consumption can be computed from the models described in [6], [7] and (2) through careful exploration of different cloud parameters and configurations, we show that optimal grouping of PMs requires sound understanding of power-performance trade-offs that exist in specific IaaS cloud environments.

Rest of the paper is organized as follows. Section II presents interacting stochastic models approach and describe computations of power consumption and performance metrics. Trade-offs in power-consumption and performance are

shown through numerical results in Section III. We conclude this work and outline future research in Section IV.

## II. Interacting Stochastic Models Approach

Our main motivation behind using an interacting sub-models approach is the following. A global monolithic model that captures all the details of a cloud service, tends to be complex and error-prone. Even by using methods of automated generation of models such as stochastic Petri-nets, such models become intractable and may not scale to large size clouds. We use interacting sub-models approach which reduces complexity of analysis without significantly affecting the accuracy. Final solution of the overall model is obtained by fixed-point iteration over individual sub-models. In this paper, we make the simplifying assumption that all inter-event times are exponentially distributed and thus all our sub-models are homogeneous continuous time Markov chains (CTMC).

**Resource provisioning decision model.** Details of this model can be found in [6], [7]. Input parameters for this sub-model are: (i) job arrival rate ($\lambda$), (ii) mean search delays to find a PM from hot/warm/cold pool that can be used for resource provisioning ($1/\delta_h$, $1/\delta_w$ and $1/\delta_c$ respectively), (iii) probabilities that a hot/warm/cold PM can accept a job for resource provisioning ($P_h$, $P_w$ and $P_c$ respectively) and (iv) maximum number of jobs in RPDE ($N$). Among all the input parameters $P_h$, $P_w$ and $P_c$ are computed as outputs from the VM provisioning models described later. From this model, we can compute job rejection probability due to buffer full ($P_{block}$), rejection probability due to insufficient capacity ($P_{drop}$) and hence, job rejection probability $P_{reject} = P_{block} + P_{drop}$. We can also compute mean queuing delay ($E[T_{q\_dec}]$) and mean decision delay ($E[T_{decision}]$) conditional upon the job being accepted.

**VM provisioning models.** VM provisioning models capture the actual provisioning and deployment of requested VMs to accepted jobs. For each hot, warm and cold PM, we have one CTMC which keeps track of the number of assigned and running VMs. VM provisioning model of a pool is the union of individual provisioning models of each PM in that pool.

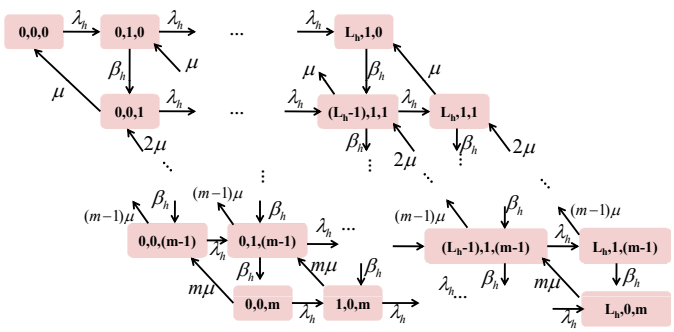


Figure 1: VM provisioning model for a hot PM.

Figure 1 shows VM provisioning model for a hot PM. Conceptually, the overall hot pool is modeled by a set of independent hot PM models. Note that only one PM model needs to be solved. States of the model in Figure 1 are indexed by $(i, j, k)$, where, $i$ denotes the number of jobs in the queue, $j$ denotes the number of VMs currently being provisioned, $k$ denotes the number of VMs on a PM which have already been deployed. Input parameters for a hot PM CTMC are: (i) effective job arrival rate to each hot PM ($\lambda_h$), (ii) rate at which VM instantiation, provisioning and configuration occurs ($\beta_h$), (iii) job service rate ($\mu$), (iv) buffer size of hot PM ($L_h$) and (v) maximum number of VMs that can be deployed on a PM ($m$). Assuming a total of $n_h$ PMs in the hot pool, $\lambda_h$ is given by $\lambda(1-P_{block})/n_h$. Observe that $P_{block}$ is computed from resource provisioning decision model. Although, we show FCFS provisioning of VMs, parallel provisioning of VMs can be captured by using a state dependent multiplier to $\beta_h$. From hot PM model, we compute the steady state probability that a hot PM can accept a job for VM provisioning. Output of the hot pool model is the probability $P_h$ that at least one PM in hot pool can accept a job for provisioning.

**Total power consumption in hot pool.** Using Markov reward approach [8], we can compute the power consumption per hot PM. We assume when no VM is running, hot PM consumes an idle power of $h_l$. Power consumption of a VM with average resource utilization is assumed to be $v_a$. For each state $(i, j, k)$ of the CTMC in Figure 1, we assign a reward rate: $r_{(i,j,k)} = h_l + kv_a$. Steady state power consumption in each hot PM ($s_h$) can be computed as expected steady state reward rate. Total power consumption in hot pool ($H_p$) is given by: $H_p = n_h s_h$. Observe, although we use a very simple power consumption model for VM execution, such Markov reward approach can be extended to more accurate power models as well.

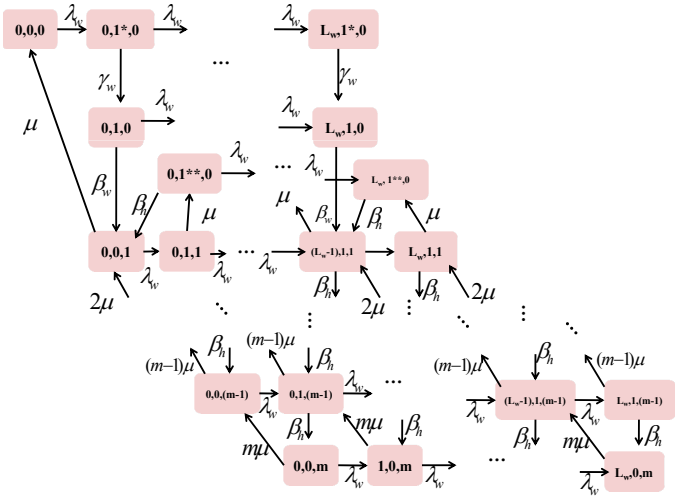


Figure 2: VM provisioning model for a warm PM.

CTMC for a warm PM (Figure 2) is similar to the hot

PM model, with few differences: (i) Effective arrival rate to each warm PM is $\lambda_w$. (ii) When no VM is running or being provisioned, a warm PM is turned on but maintains a low power state. Upon a job arrival in this state, the warm PM requires some additional startup time to make it ready to use. Rate at which warm PM is made ready for use is $\gamma_w$. (iii) Rate of VM instantiation, provisioning and configuration for first VM is $\beta_w$ and for subsequent VMs, rate is $\beta_h$. Buffer size of each warm PM is $L_w$.

Assuming a total of $n_w$ PMs in the warm pool, $\lambda_w$ is given by $\lambda(1-P_{block})(1-P_h)/n_w$. From warm PM model, we compute the steady state probability that a warm PM can accept a job VM provisioning. Output of the warm pool model is the probability $P_w$ that at least one PM in warm pool can accept a job for provisioning.

**Total power consumption in warm pool.** Reward rates attached to different states of warm PM model are summarized in Table I. *We assume that idle power consumptions during different startup phases of warm PM are less compared to the idle power consumption in hot PM.* Specifically, we assume: $w_{l_1} \leq w_{l_2} \leq w_{l_3} \leq h_l$. Once the warm PM is ready to use, power consumption is similar to that of a hot PM. Steady state power consumption in each warm PM ($s_w$) can be computed as expected steady state reward rate. Total power consumption in warm pool ($W_p$) is given by: $W_p = n_w s_w$.

Table I: Warm PM reward rates for power consumption.

| CTMC states in Figure 2 | Reward rates |
|---|---|
| $(0,0,0)$ | $w_{l_1}$ |
| $(i,1^*,0),\ 0 \leq i \leq L_w$ | $w_{l_2}$ |
| $(i,1,0),\ 0 \leq i \leq L_w$ | $w_{l_3}$ |
| $(i,1^{**},0),\ 0 \leq i \leq L_w$ | $h_l$ |
| $(0,0,k),\ 1 \leq k \leq m$ | $h_l + kv_a$ |
| $(i,1,k),\ 0 \leq i \leq L_w,\ 1 \leq k \leq (m-1)$ | $h_l + kv_a$ |
| $(i,0,m),\ 1 \leq i \leq L_w$ | $h_l + mv_a$ |

Cold PM model (Figure 3) [6], [7] is similar to warm PM model and cold pool model is the set of $n_c$ independent cold PM models. Main differences between a warm and a cold PM model are - (i) effective arrival rates ($\lambda_w$ vs. $\lambda_c$), (ii) rate at which startup is executed ($\gamma_w$ vs. $\gamma_c$), (iii) initial VM provisioning rates ($\beta_w$ vs. $\beta_c$) and buffer sizes ($L_w$ vs. $L_c$). Cold PM is turned off when all VMs finish execution. Assuming a total of $n_c$ PMs in the cold pool, $\lambda_c$ is given by $\lambda(1-P_{block})(1-P_h)(1-P_w)/n_c$. From cold PM model, we compute the steady state probability that a cold PM can accept a job for VM provisioning. Output of the cold pool model is the probability $P_c$ that at least one PM in cold pool can accept a job for provisioning.

**Total power consumption in cold pool.** Reward rates attached to different states of cold PM model are summarized in Table II. Similar to the reward rate assignment in warm PM case, we assume that for a cold PM: $c_{l_1} \leq c_{l_2} \leq c_{l_3} \leq h_l$. Steady state power consumption in each cold PM

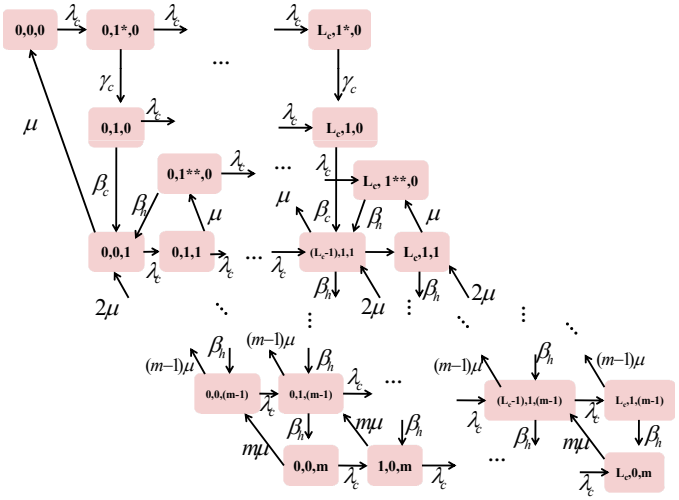


Figure 3: VM provisioning model for a cold PM.

Table II: Cold PM reward rates for power consumption.

| CTMC states in Figure 3 | Reward rates |
|---|---|
| $(0,0,0)$ | $c_{l_1}$ |
| $(i,1^*,0),\ 0 \leq i \leq L_c$ | $c_{l_2}$ |
| $(i,1,0),\ 0 \leq i \leq L_c$ | $c_{l_3}$ |
| $(i,1^{**},0),\ 0 \leq i \leq L_c$ | $h_l$ |
| $(0,0,k),\ 1 \leq k \leq m$ | $h_l + kv_a$ |
| $(i,1,k),\ 0 \leq i \leq L_c,\ 1 \leq k \leq (m-1)$ | $h_l + kv_a$ |
| $(i,0,m),\ 1 \leq i \leq L_c$ | $h_l + mv_a$ |

($s_c$) can be computed as expected steady state reward rate. Total power consumption in cold pool ($C_p$) is given by: $C_p = n_c s_c$. Thus, total power consumption ($T_p$) across all pools is given by: $T_p = H_p + W_p + C_p$.

Probabilities $P_h$, $P_w$ and $P_c$ as obtained respectively from hot, warm and cold pool models are used in resource provisioning decision model as input parameters. Using import graphs, it can be shown [6], [7] that there are input-output dependencies among VM provisioning models and resource provisioning decision model. Such dependencies can be resolved via fixed-point iteration for which existence of a solution can be proved. VM provisioning models also provide the mean queuing delay ($E[T_{q\_vm}]$), and the conditional mean VM provisioning delay ($E[T_{prov}]$). Thus, mean response delay can be computed as: $E[T_{resp}] = E[T_{q\_dec}] + E[T_{decision}] + E[T_{q\_vm}] + E[T_{prov}]$.

## III. Numerical Results and Discussions

We exercise the models described earlier to compute two power consumption metrics: (1) percentage of power consumption per pool and (2) total power consumption; as well as two performance metrics: (1) job rejection probability and (2) mean response delay. Using SHARPE [9], we show the trade-offs among the power consumption metrics and performance metrics as we change the number of PMs per pool and job arrival rate. **Values of key parameters.** We assumed a wide range of values for our model parameters so

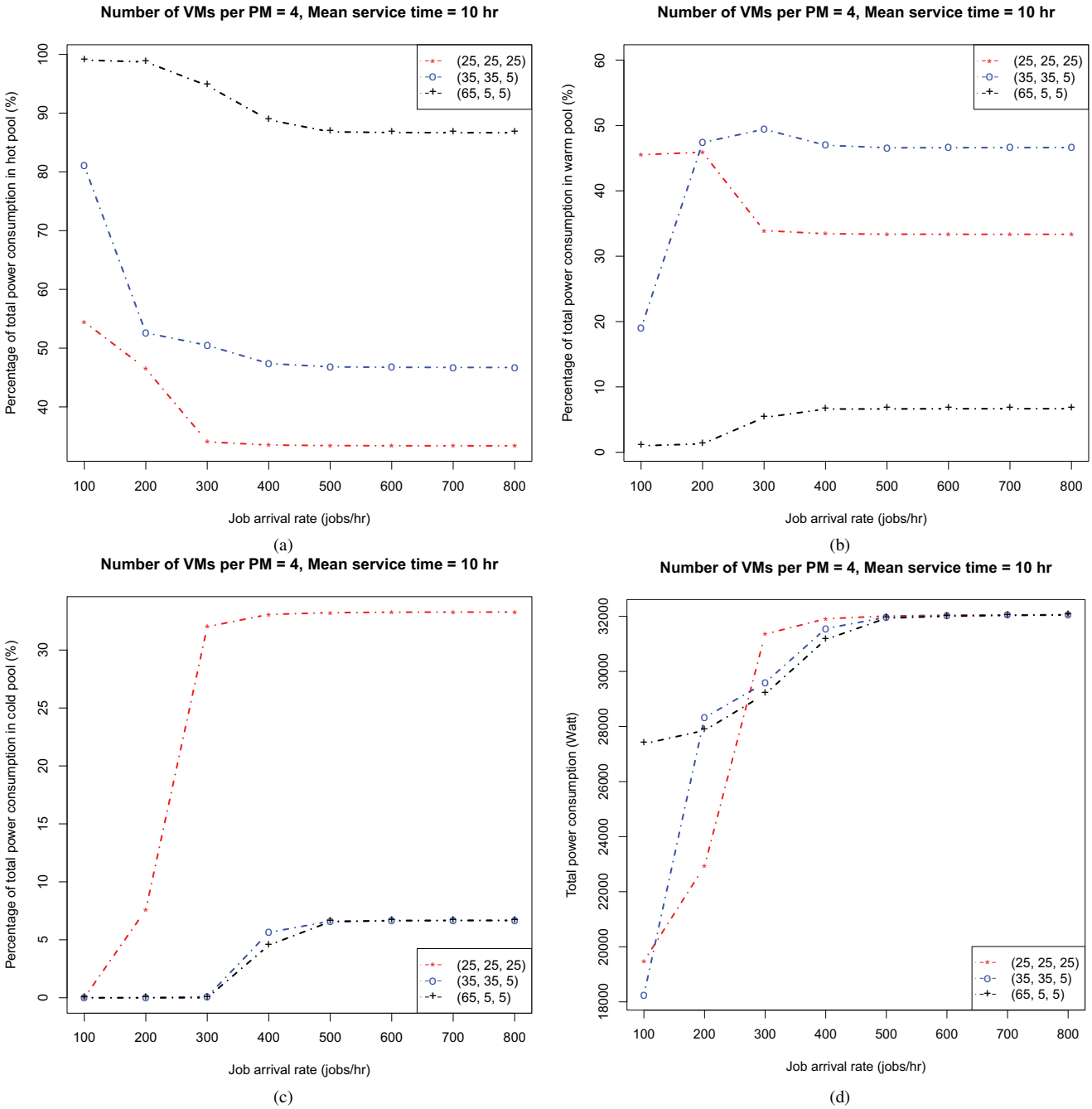


Figure 4: (a), (b), (c): Percentage of total power consumption in each pool and (d) total power consumption.

that they can represent a large variety of IaaS cloud services. **(1) Reward rates for power consumptions.** Considering a blade server type PM, idle power consumption in a hot PM ($h_l$) was assumed to be 270 W [10]. Idle power consumptions in warm PM ($w_{l_1}, w_{l_2}, w_{l_3}$) are assumed to be within $20 - 50\%$ of $h_l$. For cold PMs, idle power consumptions ($c_{l_1}, c_{l_2}, c_{l_3}$) are assumed to be within $0-40\%$ of $h_l$. Two approaches for VM power metering are proposed in [11] and [12]. With SPEC2006 benchmarks, authors of [12] provide bounds on VM power consumptions for $100\%$ CPU utilization. Observing the CPU utilization pattern on SPEC2006 benchmarks as reported in [11], we assume that power consumption per VM under average CPU utilization is around $16 - 40$W. **(2) Number of PMs in each pool.** We assumed small ($2 - 10$ PMs in each pool), medium ($10 - 100$ PMs in each pool) and large (more than 100 PMs in each pool) size clouds. **(3) Maximum number of VMs on each PM.** We assumed that $1, 2, 4, 8, 16, 32$ or $64$ VMs are deployed on each PM. **(4) Job arrival rate** ($\lambda$)**.** We categorized arrival rates in three regions: (i) low ($10 - 500$

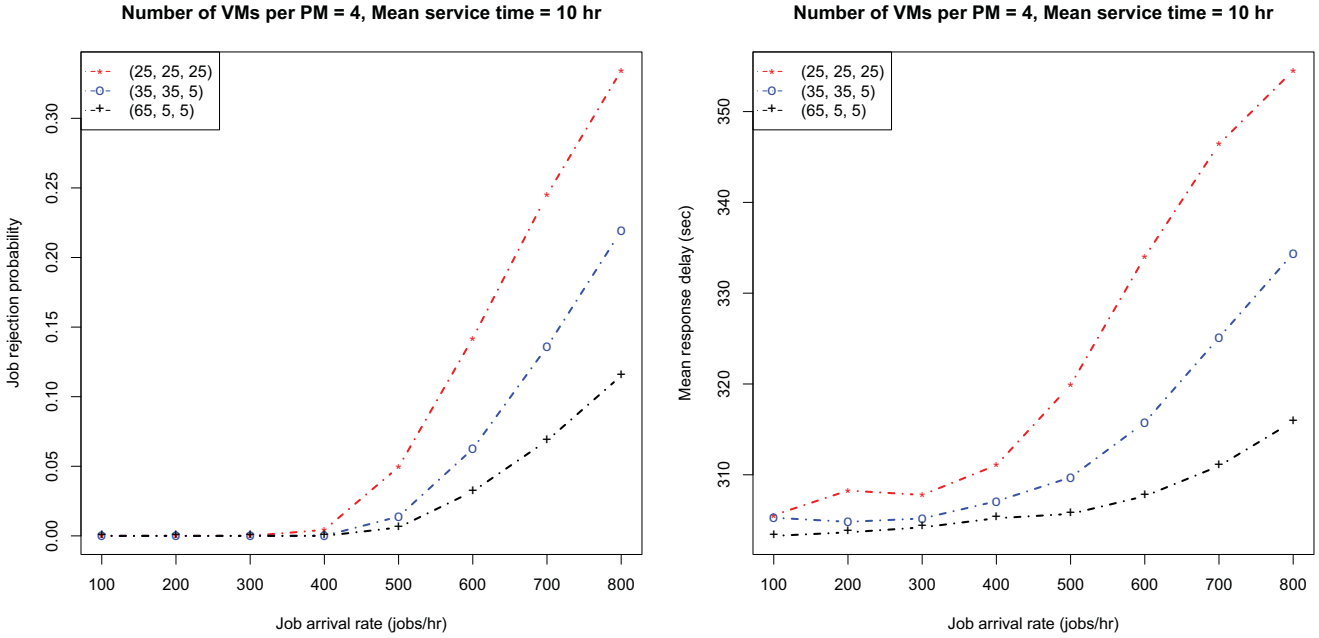


Figure 5: (a) Job rejection probability and (b) mean response delay.

jobs/hour), (ii) medium ($500-1500$ jobs/hour) and (iii) high (more than $1500$ jobs/hr). **(5) Mean job service time** $(1/\mu)$**.** In our analysis, we assumed three ranges for mean service times: (i) low (less than an hour), (ii) medium (1 hr - 1 day) and (iii) high (more than 1 day) [4]. **(6) Mean delay to search a PM from a pool** $(1/\delta_h, 1/\delta_w, 1/\delta_c)$**.** We assumed that the searching delay was independent of number of PMs or type (hot, warm, cold) of pool and varied the values from $1-5$ seconds. **(7) Mean time to provision a VM.** Values of $1/\beta_h$, $1/\beta_w$ and $1/\beta_c$ were assumed to be in the range of $1-10$ minutes, $2-20$ minutes and $5-30$ minutes respectively. **(8) Mean time to prepare a warm** $(1/\gamma_w)$**/ cold** $(1/\gamma_c)$ **PM ready for use.** Value of $1/\gamma_w$ was assumed in the range of 20 seconds to 2 minutes. Value of $1/\gamma_c$ was assumed in the range of 5-10 minutes.

**Effects of PM configurations on power consumption behavior.** We consider an example scenario where total 75 PMs are distributed differently among three pools. These configurations are: (1) each pool has 25 PMs; denoted by $(25, 25, 25)$, (ii) hot and warm pool have 35 PMs each; denoted by (35, 35, 5) and (iii) warm and cold pool have 5 PMs each; denoted by (65, 5, 5). Increasing or decreasing trends in percentage power consumptions in a specific configuration can be easily visualized if we observe Figures 4(a), (b) and (c) simultaneously. For example, as the arrival rate increases from 100 jobs/hr to 300 jobs/hr, in $(25, 25, 25)$, net increase (around $33\%$) in power consumption in cold pool exactly compensates the net decrease in power consumptions in hot (around $20\%$) and warm (around $13\%$) pools. Similarly, as the arrival rate increases from 100 jobs/hr to 300 jobs/hr, in $(35, 35, 5)$, net increase (around $30\%$) in warm pool power consumption is same as net decrease in hot pool power consumption and the cold pool power consumption is almost zero. Figure 4(a) shows percentage of total power dissipated in hot pool is maximum in (65, 5, 5) and minimum in (25, 25, 25). This is because, for $(65, 5, 5)$, most of the jobs are provisioned in hot pool which contains more than $85\%$ of total PMs as opposed to (25, 25, 25), which has only one-third of the total PMs in hot pool. With increased arrival rate more warm PMs and subsequently cold PMs start accepting jobs and dissipate more power. Figure 4(b) shows with increased arrival rate, percentage of total power consumption in warm pool is more in $(35, 35, 5)$ compared to $(25, 25, 25)$. This is because percentage of warm PMs in $(35, 35, 5)$ is higher and warm pool accepts more jobs. Compared to $(65, 5, 5)$ and $(35, 35, 5)$, larger fraction of cold PMs are present in $(25, 25, 25)$. Figure 4(c) shows that in $(25, 25, 25)$, percentage of power consumption by the cold pool is maximum among the three configurations.

**Power-performance trade-offs among different configurations.** Total power consumption for each of the three configurations is shown in Figure 4(d). Figures 5(a) and (b) show the changes in two performance metrics, respectively: (i) job rejection probability and (ii) mean response delay. Interestingly, when arrival rate is in the range of $300-500$ jobs/hr, total power consumption in $(65, 5, 5)$ is minimum compared to other two configurations. As a result, $(65, 5, 5)$ exhibits **smallest power consumption and minimum rejection probability and mean response delay**. This is because, in the range of $300-500$ jobs/hr, warm and cold PMs in $(25, 25, 25)$ and $(35, 35, 5)$ start behaving like hot PMs. So, advantage of having more PMs in warm

or cold pool almost vanishes. In case of $(65, 5, 5)$, most of the jobs get accepted in hot pool and arrival rates to warm and cold pool are low. Thus, probabilities of being in low power consumption states for warm and cold PMs are higher. Figure 4(d) also shows that beyond $500$ jobs/hr, all configurations are equivalent in terms of power consumption. However, at such higher arrival rates, job rejection probability (Figure 5(a)) and mean response delay (Figure 5(b)) are significantly lower in $(65, 5, 5)$ compared to other two configurations. Observe, there is a reduction in mean response delay for $(25, 25, 25)$, when arrival rate is increased from $200$ jobs/hr to $300$ jobs/hr. This is because, mean response delay has four components as described in Section II. Among these components, mean provisioning delay reduces in $(25, 25, 25)$, when more cold PMs start accepting jobs, behave like hot PMs and achieve less mean provisioning delay. At this arrival rate (near $300$ jobs/hr), such effect is not observed in other configurations as most of the jobs are provisioned in hot and warm pools. With further increase in arrival rate, net increase in mean queuing delay more than compensates the decrease in mean provisioning delay. As a result, mean response delay increases with arrival rate for all three configurations.

Table III: Solution times for large scale IaaS clouds.

| Max. number of VMs per PM | Solution time (sec) |
|---|---|
| 2 | 0.698 |
| 4 | 0.705 |
| 8 | 0.711 |
| 16 | 0.721 |
| 32 | 0.739 |
| 64 | 0.783 |

**Scalability of proposed approach.** So far, we presented results to demonstrate cases of interesting power-performance trade-offs with relatively smaller number of PMs. However, our models can be used to analyze clouds with large number of PMs and VMs. Observe that, number of states in any sub-model is independent of number of PMs in the pools but VM provisioning models grow in size as the maximum number of VMs per PM is increased. Using $5000$ PMs in each pool ($15,000$ in total), RPDE buffer size of $1000$, we vary the the number of the VMs per PM and report the solution times in Table III. All models were solved using a desktop PC with Intel Core 2 Duo processor (E8400, 3.0 GHz) and 4 GB memory. Clearly, interacting sub-models approach facilitates power-performance trade-off analysis of large IaaS clouds with a reasonably small solution time.

## IV. Conclusions and Future Work

In this paper, we describe a scalable and fast analytic approach for power consumption and performance analysis of IaaS cloud. Using three classes of pools, we show the effects of PM pool configurations on performance and power consumption and discuss the trade-offs between the two. Our analysis shows that single fixed partitioning among different pools is not always the best strategy, especially under varying arrival rate – a common characteristics of the cloud. Using dynamic partitioning and adjusting the pool sizes to match expected arrival rates can result in optimal power-performance balance. Analysis presented here can be used to achieve power and performance objectives by managing the PM pool configurations. In future: (1) We will mathematically formulate several optimization problems such as: (i) for a given arrival rate, what is the optimal number of PMs in each pool that can minimize the total power consumption and do not violate the performance SLAs, i.e., upper bound on job rejection probability and mean response delay?, (ii) for a given arrival rate and given total power consumption budget, what is the optimal number of PMs in each pool that can minimize the job rejection probability and mean response delay? (2) Using the data collected from real cloud, we will design more detailed power consumption models for VM execution. (3) Sensitivity analysis will be carried out w.r.t. model parameters. (4) We will analyze more realistic cases e.g., heterogenous requests and different scheduling policies.

## ACKNOWLEDGMENTS

Research by Duke authors (Ghosh and Trivedi) was supported in part under a 2010 IBM Faculty Award and in part by NSF under Grant NSF-CNS-08-31325. Authors would like to thank Prof. Jogesh K. Muppala for his insightful comments and feedback on this work. This work was completed during Rahul's internship at IBM Research.

## References

[1] M. Marwah *et al.*, "Quantifying the sustainability impact of data center availability," *Sigmetrics Perf. Eval. Rev.*, 2010.
[2] K. Joshi *et al.*, "The case for energy-oriented partial desktop migration," in *HotCloud*, 2010.
[3] "Amazon EC2," http://aws.amazon.com/ec2.
[4] "IBM Smart Cloud Enterprise," http://www.ibm.com/cloud-computing/us/en/#!iaas.
[5] R. Ghosh, F. Longo, V. K. Naik, and K. S. Trivedi, "Quantifying resiliency of IaaS cloud," in *RACOS Workshop*, 2010.
[6] R. Ghosh, K. S. Trivedi, V. K. Naik, and D. S. Kim, "Performability analysis for Infrastructure-as-a-Service cloud: An interacting stochastic models approach," in *IBM Research Report, RC 25006*, 2010.
[7] R. Ghosh, K. S. Trivedi, V. K. Naik, and D. S. Kim, "Performability analysis for Infrastructure-as-a-Service cloud: An interacting stochastic models approach," in *PRDC*, 2010.
[8] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*. Wiley, 2001.
[9] K. S. Trivedi and R. Sahner, "SHARPE at the age of twenty two," *Sigmetrics Perf. Eval. Rev.*, March 2009.
[10] D. Meisner, B. Gold, and T. Wenisch, "Powernap: Eliminating server idle power," in *ASPLOS*, 2009.
[11] A. Kansal *et al.*, "Virtual machine power metering and provisioning," in *SoCC*, 2010.
[12] B. Krishnan, H. Amur, A. Gavrilovska, and K. Schwan, "Vm power metering: Feasibility and challenges," in *GreenMetrics*, 2010.