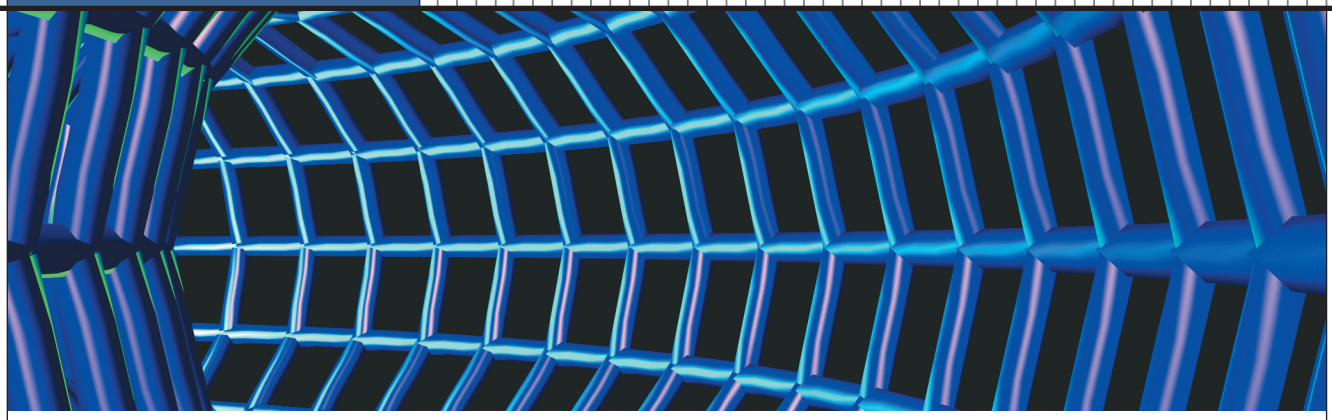

Using Mathematical Modeling in Provisioning a Heterogeneous Cloud Computing Environment

Sungkap Yeo and Hsien-Hsin S. Lee, *Georgia Institute of Technology*

Mathematical models demonstrate that to achieve optimal performance in a heterogeneous cloud infrastructure, the slowest node's response time should be no more than three times that of the fastest node.

Cloud computing is transforming the entire IT industry, high-performance computing (HPC), and personal data sharing and management. In cloud computing, computing power is supplied as a utility, similar to electricity or water. As such, service providers can centrally manage, maintain, and upgrade computing resources, offloading the burden from small business owners or those who do not have the expertise or budget to handle the fast-changing computing infrastructure.

Using the cloud for HPC can substantially reduce the total cost of ownership by eliminating the need to maintain large-scale parallel machines and their energy-consuming power and cooling systems.[1,2] From a cost-effectiveness perspective, there are tradeoffs in terms of resource provisioning given that a target task can be parallelized, a common case for throughput-oriented computing.

For example, assume that an HPC job, which can be perfectly parallelized, takes eight hours to complete using one computing node. If the cloud computing service provider charges for a job on a per-machine-hour basis (that is, based on the accumulated machine time), instead of running it on one node for eight hours, the job can be finished in one hour on eight machines with eight times speedup with the same utility charge (eight machine hours).

One trend that complicates this tradeoff is the heterogeneity in a cloud computing environment. Although a cloud service provider can start with near-homogeneous computing nodes, the facility will likely grow more heterogeneous over time due to upgrades and replacement. Therefore, not only will each computing node's performance and capability continue to deviate, the new computing nodes will also provide better performance for the same amount of power due to technology scaling and architectural innovation. Because of this heterogeneity, response times will vary significantly depending on provisioning policies. To mitigate this variation and guarantee quality of service, the cloud provider might want to dismiss the slowest computing nodes. The question is how slow a physical node can be for a given task to maintain its optimal computing quality in terms of execution time and energy cost.

0018-9162/11/$26.00 © 2011 IEEE Published by the IEEE Computer Society

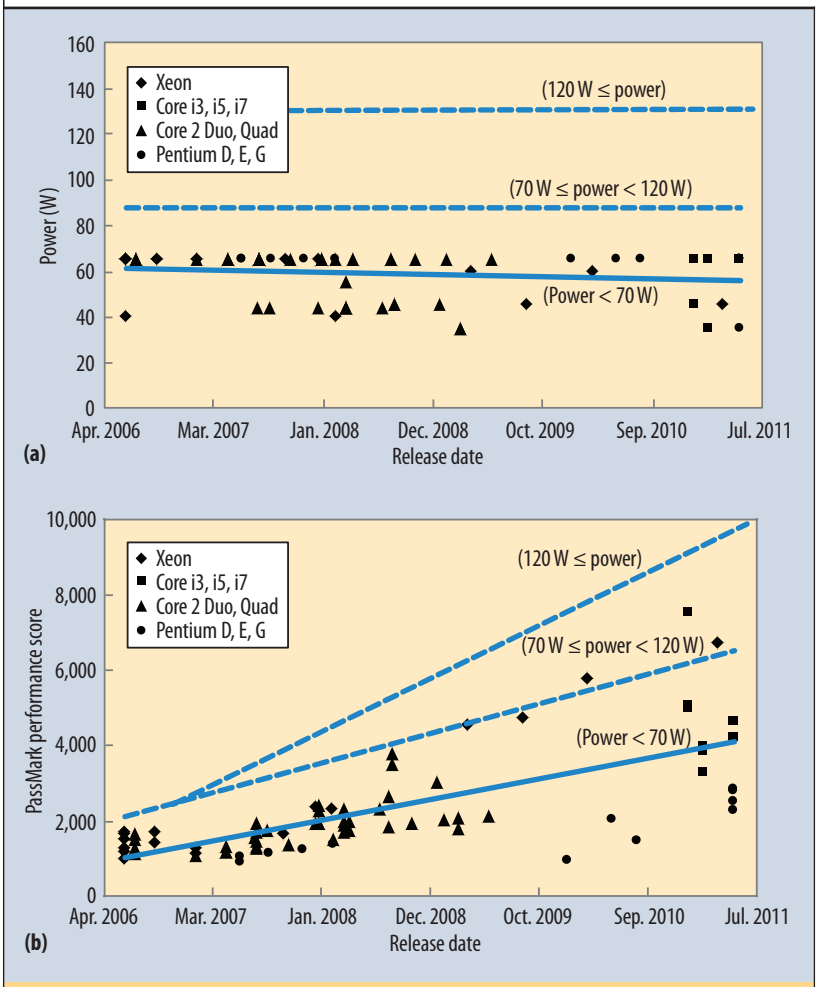

Figure 1. Power consumption and performance of Intel's CPUs since 2006: (a) power consumption and (b) performance. Over time, newer CPUs achieve higher performance than the older ones without compromising the power consumption.

To tackle this issue, we established a mathematical model based on statistics for a heterogeneous cloud environment. To understand optimal provisioning in a cloud, we used this model to evaluate the tradeoff of a task's execution time and energy consumption.

CLOUD COMPUTING MODEL

For this study, we assume the workload is perfectly parallelizable, which is often the case for throughput-oriented computing in HPC and transactional processing applications. For example, the most common cloud computing application is file transferring on the Web. Servers in the cloud can process all the requests received by a Web service at the same time individually and independently. Therefore, the cloud can achieve n times speedup when n nodes are deployed if and only if the number of concurrent users is always larger than or equal to n.

Next, we assume that an entire workload can be evenly divided into m smaller job units without affecting the workload's scalability. We also assume that m is larger than n, where n represents the maximum number of virtual machines in the cloud. (For simplicity, $m = kn$, where k is a positive integer.) In this study, one job unit represents the smallest task running to the end on a single physical node without interruption. However, we do not consider intermittent context switches within one job unit as interruption as long as the task keeps running on the same physical node.

In addition, we do not allow a virtual machine to be migrated among physical nodes during a job unit's execution because this migration will not only include the executable image but also all the architectural states, including the memory footprint. Data migration on interconnected cloud computing nodes would likely cause significant performance degradation due to peer-to-peer communication.

Cloud power and performance behavior

Before detailing power and performance in a heterogeneous cloud, we present a scenario from a cloud administrator's perspective. Typically, cloud service providers begin their cloud computing business with several near-homogeneous computing nodes. Over time, the cloud provider will replace some of the old computing nodes with newer nodes featuring the latest technologies. Gradually, the capability and performance of all machines in the cloud will become more heterogeneous. Although previous studies considered heterogeneity at the microarchitectural[3] and system levels,[4] they all assumed heterogeneity in the same generation of manufacturing technology. We consider computing heterogeneity in a broader sense.

We reviewed the power and performance trends of commercial microprocessors over the past few years and used our observations to justify our model assumption. We first plotted the thermal design power (TDP) numbers and the PassMark performance scores[5] for several processors under 70 W, including Pentium, Core 2, Core i3/5/7, and Xeon. This included all commercial desktop and server processors from Intel from January 2006 to February 2011, except Celeron processors and certain processors that did not report TDP or PassMark results. The solid line in Figure 1 shows their asymptotic power consumption and performance trend between 2006 and 2011. The dashed lines without individual dots show the trends of two other

machine groups based on their TDP: 70 W to 120 W and more than 120 W.

We applied regression methods to estimate the relationship between power and performance over time. Taking all the samples into account, we plotted our regression models for power and performance (solid lines in Figure 1). As Figure 1b shows, the performance continues to improve for each machine group across different proliferations or generations. On the other hand, the TDP trend in Figure 1a shows negligible growth. More interestingly, the TDP trends for the two lower-power machine classes are decreasing. This decrease is the consequence of a recent awareness of the power wall, which gradually increases the heat dissipation cost. For the same reason, we anticipate that the power grade of future processors will remain below the bar. This also implies that with the same power budget, newer machines can deliver higher performance. In other words, performance per power (a metric derived by dividing the performance score by the power consumption) continues to grow over time. For example, the 95 W Core i7 (Lynnfield), released in September 2009, achieves higher performance than the 95-W Pentium D (Presler), released in January 2006. This difference is largely attributable to technological advances in microarchitecture as well as scaled-down feature size and supply voltage.

Given these observations, we define our model of power and performance for a future heterogeneous cloud based on two assumptions.

First, the computing nodes in the cloud we analyze are heterogeneous, having different microarchitectures fabricated using different processes. Thus, the cloud provides varied capability and process technologies. Second, the performance capabilities of these computing nodes are uniformly distributed (from low to high) but consume exactly the same amount of power.

The rationale behind this second assumption is twofold. First, for a given power budget, the performance of each machine class continues to improve linearly while their power envelope remains pretty much unchanged. In other words, the power efficiency measured by performance per power improves over time. Second, when a datacenter phases out some computing nodes due to an upgrade, it can safely deploy new computing nodes only when these upgrades' aggregated power consumption does not exceed the original. Otherwise, the datacenter must also upgrade its power delivery infrastructure as well as its cooling capacity to accommodate the new servers. Given this overhead, we anticipate that the replacement and upgrade will be done without altering the power delivery infrastructure. Therefore, we assume that the newly deployed servers will improve performance linearly across different machine proliferations while using the same amount of power.

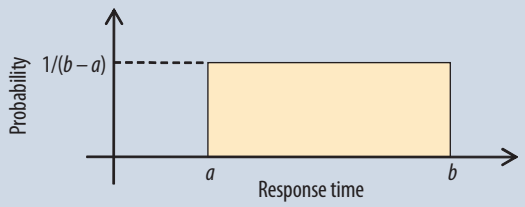

Figure 2. Probability distribution function (PDF) of the execution time of a job unit when there are n virtual machines. The execution time is uniformly distributed from a seconds (the fastest node) to b seconds (the slowest node).

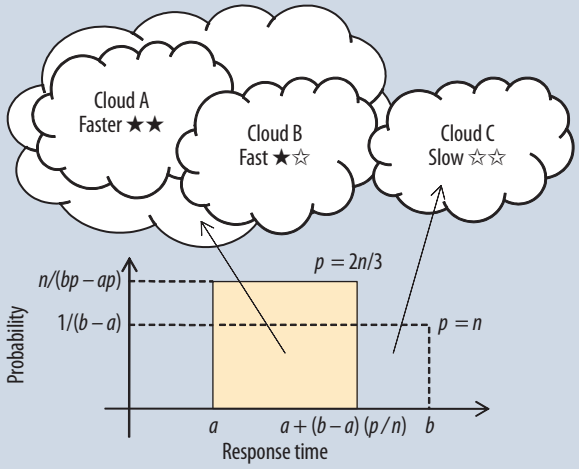

Figure 3. Probability distribution function (PDF) of the execution time of a job unit when there are $2n/3$ virtual machines. The worst-case response time is improved by dismissing the one-third of physical nodes with the lowest performance.

To express this distribution mathematically, we assume that the response time for executing a job unit in such a cloud is uniformly distributed from a seconds (the fastest node) to b seconds (the slowest node). Figure 2 illustrates the probability distribution function (PDF) of the response time for executing a job unit in this cloud.

On the other hand, we assume that the cloud service provider can improve the worst-case response time by dismissing physical nodes with the least performance. For example, when a cloud service provider decides to retire one-third of its physical nodes from the slowest batch, we assume that the new response time for executing a job unit of this cloud becomes a uniform distribution from a seconds to $(a + 2b)/3$ seconds, represented by $U(a, (a + 2b)/3)$. As such, we assume that the maximum number of virtual machines that can be allocated on this cloud also shrinks in the same ratio.

Figure 3 shows the impact of retiring one-third of a cloud service provider's physical nodes from the cloud. The variable p in this figure represents the maximum number of virtual machines that can be allocated on the cloud, while

n represents the maximum number of virtual machines for the original cloud as shown in Figure 2. Moreover, the PDF in Figure 3 shows the improved worst-case response time as a result of removing one-third of the physical nodes from the slowest side.

Nevertheless, in the given response time PDF, we did not assume that a particular virtual machine can pick a physical node at a particular speed. Rather, when a cloud's PDF is given, we assumed that a virtual machine's behavior in this cloud follows the PDF in a statistical manner. In other words, we assumed that virtual machines will be uniformly distributed across the physical nodes.

Although dispatching more jobs to newly deployed servers with higher power efficiency increases energy efficiency, this is not the case for a datacenter, for two reasons. First, for a datacenter, it is important to balance the power draw across the AC phases.[6] The balance will break when jobs are distributed to only certain computing racks. Second, we want to minimize the number of hotspots for a datacenter, a common consequence of unbalanced workloads. Hotspots generally cause higher machine failure rates and require additional attention and effort to remove the heat.

Given that each computing node consumes the same amount of power, energy consumption will be proportional to the total execution time.

Execution time and energy consumption

We define the execution time of a given workload on a cloud as the time required to finish a workload consisting of m job units. When some job units are assigned to more than one virtual machine, the execution time, in our definition, is bounded by the virtual machine that finishes last. For example, when an animator renders a movie comprising m independent frames, the movie cannot be released before the last frame finishes rendering. In addition, when comparing the performance of cloud configurations, we use as the baseline the case of executing the same amount of workload on a virtual machine running on the fastest node. When we use more virtual machines to execute the workload in parallel, we use slower nodes to accomplish the task. As a result, the parallelized version could reduce the overall effectiveness of energy consumed in the cloud.

Energy consumption is the total energy needed to complete a given workload. In particular, when some physical nodes finish their assigned job units before the others, we assume that these nodes will not consume energy while waiting for the others to finish. This is because, in a real-world scenario, these nodes will either be assigned to other tasks or moved to a near-zero power state to save energy.[7] In addition, given that each computing node consumes the same amount of power, energy consumption as defined will be proportional to the total execution time. Therefore, we calculate a parallelized workload's utility consumption as the summation of each virtual machine's execution time.

To quantify the effectiveness of resource provisioning in a cloud, we use the energy-delay product (EDP),[8] which we calculate by multiplying the execution time (seconds) with the energy consumption (joules). We will use this metric in our subsequent evaluation when provisioning resources (that is, the number of virtual machines to allocate to achieve optimal energy efficiency).

ANALYTICAL EVALUATION

Next, we use analytical models, based on our assumptions, to compare each configuration's EDP to the baseline EDP.

Baseline

The baseline of our study assumes that the entire job is performed on one virtual machine running on the fastest physical node. In this case, the fastest physical node can retire a job unit every a seconds. Because there are m independent job units in the entire workload, the baseline configuration takes ma seconds to finish. This configuration consumes $W \times ma$ joules for completing the entire workload, where W represents a physical node's power. Thus, the EDP of this study's baseline is

$$\mathrm{EDP}_{\mathrm{base}} = (W \times ma)(ma) = Wm^2a^2.$$

Expectation-based analysis

We use an expectation-based analysis to determine a cloud model's execution time and energy consumption. We use a new distribution function to represent the execution time of a virtual machine with more than one job unit.

Execution time distribution across virtual machines. The PDF of the response time when using p virtual machines is given by $U(a, (a + ((b - a)p)/n))$, as Figure 3 illustrates. However, when a virtual machine is responsible for more than one job unit (that is, m/p units), the virtual machine's total execution time cannot be modeled the same way. Rather, we model it as the summation of independently selected m/p samples from Figure 3. When we add independent samples from a uniform distribution, the summation's distribution function tends to approach a normal distribution according to the central limit theorem.[9] This theorem proves that when we add more independent samples into the summation, the summation's distribution will become more like a normal distribution. In addition, the summation of 12 samples is known to be good enough

to satisfy the central limit theorem.[9] In this case, we assume that a virtual machine is responsible for more than 12 job units by letting $m \geq 12n$ (that is, $m \geq 12p$ because $p \leq n$).

Now our goal is to obtain the mean and variance of the normal distribution representing the total execution time of a virtual machine responsible for m/p job units. First, we calculate the mean and variance for the original uniform distribution, $U(a,(a + ((b - a)p)/n))$:

$$\text{Mean} = \frac{1}{2}\left(a + a + \frac{(b-a)p}{n}\right) = a + \frac{(b-a)p}{2n} \text{ and}$$

$$\text{Variance} = \frac{1}{12}\left(a + \frac{(b-a)p}{n} - a\right)^2 = \left(\frac{(b-a)p}{\sqrt{12}n}\right)^2$$

The central limit theorem shows that the summation of m/p independent samples from this distribution will become a normal distribution with the following mean and variance:

$$N\left(\frac{m}{p}\left(a + \frac{(b-a)p}{2n}\right), \left(\sqrt{\frac{m}{p}} \times \frac{(b-a)p}{\sqrt{12}n}\right)^2\right) = N(\mu, \sigma^2) \quad (1)$$

For convenience, we use μ and σ^2 to denote the distribution's mean and variance. All in all, when using p virtual machines, each machine's execution time will follow the normal distribution, $N(\mu, \sigma^2)$. The ultimate question is, "How many seconds will it take to finish the entire workload?" To answer this question, we must first determine the expectation of the largest sample from $N(\mu, \sigma^2)$ when we must pick p samples. Because the overall execution time depends on the slowest virtual machine that finishes last, the largest of p samples will give the total execution time.

Expectation of the largest sample. Before finding the largest sample's expectation, we discuss the same question for the standard normal distribution, $N(0, 1)$. Let $pdf(x)$ be the PDF of the standard normal distribution. In this PDF, let y be the largest sample among randomly chosen p samples. For each case out of p cases, the probability of y being the largest sample is given as follows:

$$\text{Probability} = pdf(y) \times \left(\int_{-\infty}^{y} p\,df(x)dx\right)^{p-1}$$

Equation 2 gives the expectation of the variable y.

$$\int_{-\infty}^{\infty} p \times y \times p\,df(y) \times \left(\int_{-\infty}^{y} p\,df(x)dx\right)^{p-1} dy = ExB(p) \quad (2)$$

For convenience, $ExB(p)$ denotes the expectation of the largest sample among p samples from the standard normal distribution. In addition, by substituting $pdf(x)$ in Equation 2 with Equation 3, we can find the numerical values of $ExB(p)$ for various p. We show the results in the middle column of Table 1.

Table 1. Expectation of the largest sample ($ExB(p)$) from $N(0, 1)$.

Number of samples (p)	Value using Equation 2	Value using the experimental method
1	0.00000	–0.00001
2	0.56419	0.56419
4	1.02938	1.02938
8	1.42360	1.42356
16	1.76599	1.76591
32	2.06967	2.06968
64	2.34373	2.34368
128		2.59461
256		2.82679
512		3.04392

$$pdf(x) = \frac{1}{\sqrt{2\pi}}\exp(-x^2/2) \quad (3)$$

Because Equation 2's complexity grows exponentially as p increases, we cannot find the exact numerical values of $ExB(p)$ for $p > 64$. To address this shortcoming, we propose a more scalable way of approximating the values in Table 1. In this solution, we first implement a random number generator that produces random numbers from the standard normal distribution. Using this random number generator, our solution picks p independent random samples and remembers the largest sample among them. This operation continues for a long enough time (for example, to produce the results in Table 1, our software repeated this operation more than 100 million times) and averages the largest samples. This experimental method generates the exact numerical values of $ExB(p)$, as shown in the third column of Table 1, after averaging more than 100 million trials. As a comparison of the second and third columns in the table shows, the mathematical accuracy is slightly compromised in exchange for scalability. However, we do not expect the tiny error rate to affect our analysis and conclusion.

The study of the largest sample in the standard normal distribution gives us an idea about the $ExB(p)$ for other normal distributions. Let a random variable X follow $N(\mu, \sigma^2)$ with $\mu \neq 0$, $\sigma \neq 1$, $\sigma \neq 0$, and a derived random variable $Y = (X - \mu)/\sigma$. Then, Y follows $N(0, 1)$ by recalling the property that if X follows $N(\mu, \sigma^2)$ and a and b are real numbers, then $aX + b$ follows $N(a\mu + b,(a\sigma)^2)$. From Equation 2, the expectation of the largest sample for Y is $ExB(p)$ because $Y = (X - \mu)/\sigma, X = Y\sigma + \mu$; and the expectation of the largest sample for X is $ExB(p) \cdot \sigma + \mu$. Now, we can calculate the expectation of the largest sample for any arbitrary normal distribution.

Execution time and energy consumption analysis. In our model, each of the p virtual machines is responsible for m/p job units, and the response time for each job unit

follows $U(a, (a + ((b - a)p)/n))$. We use the following equation to calculate the expectation of the time required on a virtual machine finishing last:

$$\begin{aligned}&\text{Execution time}\\&= \mu + ExB(p) \times \sigma\\&= \frac{m}{p}\left(a + \frac{(b-a)p}{2n} + ExB(p) \times \sqrt{\frac{m}{p}} \times \frac{(b-a)p}{\sqrt{12}n}\right)\\&= \frac{ma}{2np}\left(2n + \left(\frac{b}{a} - 1\right)p + ExB(p) \times p^{3/2}\sqrt{\frac{1}{3m}} \times \left(\frac{b}{a} - 1\right)\right)\\&= \frac{ma}{2np}\left(2n + \left(\frac{b}{a} - 1\right)p + Unbalance\left(\frac{b}{a}, p, m\right)\right)\end{aligned}$$

In this equation, we name the second term *unbalance*, which becomes zero if and only if every virtual machine finishes at the same time:

$$Unbalance\left(\frac{b}{a}, p, m\right) = ExB(p) \times p^{3/2}\sqrt{\frac{1}{3m}} \times \left(\frac{b}{a} - 1\right) \quad (4)$$

For example, a higher deviation from the normal distribution indicates that the random samples from this distribution are more spread out, increasing the probability of having more deviated samples. In our case, because we model a virtual machine's finishing time by picking a sample from Equation 1, more deviated samples indicate that the workload assignment is unbalanced among virtual machines executing this workload. In particular, a larger b/a will lead to a larger σ^2 in Equation 1 and a larger $Unbalance((b/a), p, m)$ in Equation 4. Hence, we can conclude that a larger b/a value causes a more unbalanced workload distribution among virtual machines, degrading the overall performance. Also note that $Unbalance((b/a), p, m)$ is directly proportional to $1/\sqrt{m}$. Because m is independent of p and b/a, changing the value of m will not affect other variables in Equation 4. This implies that a very large m will eventually zero out Equation 4. Thus, we can express the execution time when $m \to \infty$ as

$$\begin{aligned}&\text{Execution time } (when\ m \to \infty)\\&= \frac{ma}{2np}\left(2n + \left(\frac{b}{a} - 1\right)p\right)\end{aligned}$$

We also evaluate the energy consumption probabilistically. Because performance is bounded by the execution time of the virtual machine finishing last, we must calculate the expectation of the largest sample from Equation 1. In contrast, to evaluate the utility consumption, we must focus on the average execution time of p virtual machines. This is because, in a normal distribution, the probability for having $\mu + \alpha$ samples is exactly the same as having $\mu - \alpha$ samples. This fact indicates that the odds of having a virtual machine consuming α seconds more than the average is the same as having a virtual machine consuming α seconds less than the average. Therefore, we conclude that the expectation of the total execution time is given by $\mu \times p$, the number of virtual machines. Given the power of a physical node in the cloud is W, the total energy consumption will be as follows:

$$\begin{aligned}\text{Energy consumption} &= W \times \frac{m}{p}\left(a + \frac{(b-a)p}{2n}\right)p\\&= W \times m\left(a + \frac{(b-a)p}{2n}\right),\end{aligned}$$

$$\begin{aligned}EDP_{\exp}(p) &= \frac{Wm^2a^2}{4n^2p} \times\\&\left(\left(2n + \left(\frac{b}{a} - 1\right)p + Unbalance\left(\frac{b}{a}, p, m\right)\right)\left(2n + \left(\frac{b}{a} - 1\right)p\right)\right)\\&= \frac{EDP_{\text{base}}}{4n^2p} \times\\&\left(\left(2n + \left(\frac{b}{a} - 1\right)p + Unbalance\left(\frac{b}{a}, p, m\right)\right)\left(2n + \left(\frac{b}{a} - 1\right)p\right)\right)\end{aligned}$$

Similarly, we calculate the EDP for $m \to \infty$ as follows:

$$EDP_{\exp, m \to \infty}(p) = \frac{EDP_{\text{base}}}{4n^2p} \times \left(2n + \left(\frac{b}{a} - 1\right)p\right)^2 \quad (5)$$

To visualize the effect of a large m in the $EDP_{\exp}$ metric, Figure 4 shows the *EDP* analysis for $m = 12n$, $m = 120n$, and $m \to \infty$ using the following coefficients: n = 16,384, $b/a = 1, 2, 3, 5$, and $ExB(p)$ from Table 1. To find the exact value p that makes the EDP metric a global minimum point, we take the derivative of Equation 5 with respect to p and set it to zero:

$$\begin{aligned}&\frac{d}{dp}\left(EDP_{\text{base}} \times \frac{(2n + (b/a - 1)p)^2}{4n^2p}\right) = 0\\&p = \frac{2n}{\frac{b}{a} - 1}\left(\because p > 0\right) \quad (6)\end{aligned}$$

In the example of $m \to \infty$ in Figure 4, we achieve the minimum EDP when $p = 2n/(b/a - 1) = 16{,}384$ in Figure 4c or $p = 2n/(b/a - 1) = 8{,}192$ in Figure 4d.

Again, $p = n$ must be fulfilled while maintaining Equation 6 to be energy-effective for all n virtual machines in the cloud. By combining two conditions, $p = n$ and Equation 6, we can calculate the requirement of b/a as $n = 2n/(b/a - 1)$; $b/a = 3$. This equation suggests that in a heterogeneous cloud computing environment with uniformly distributed performance, physical nodes that respond 3 times slower than the fastest node should not be used when attempting to minimize the EDP.

These models and analysis can be used to guide future deployment, allocation, and upgrades of cloud infrastructure to achieve optimal utility effectiveness. The findings presented here hold not only for a computing

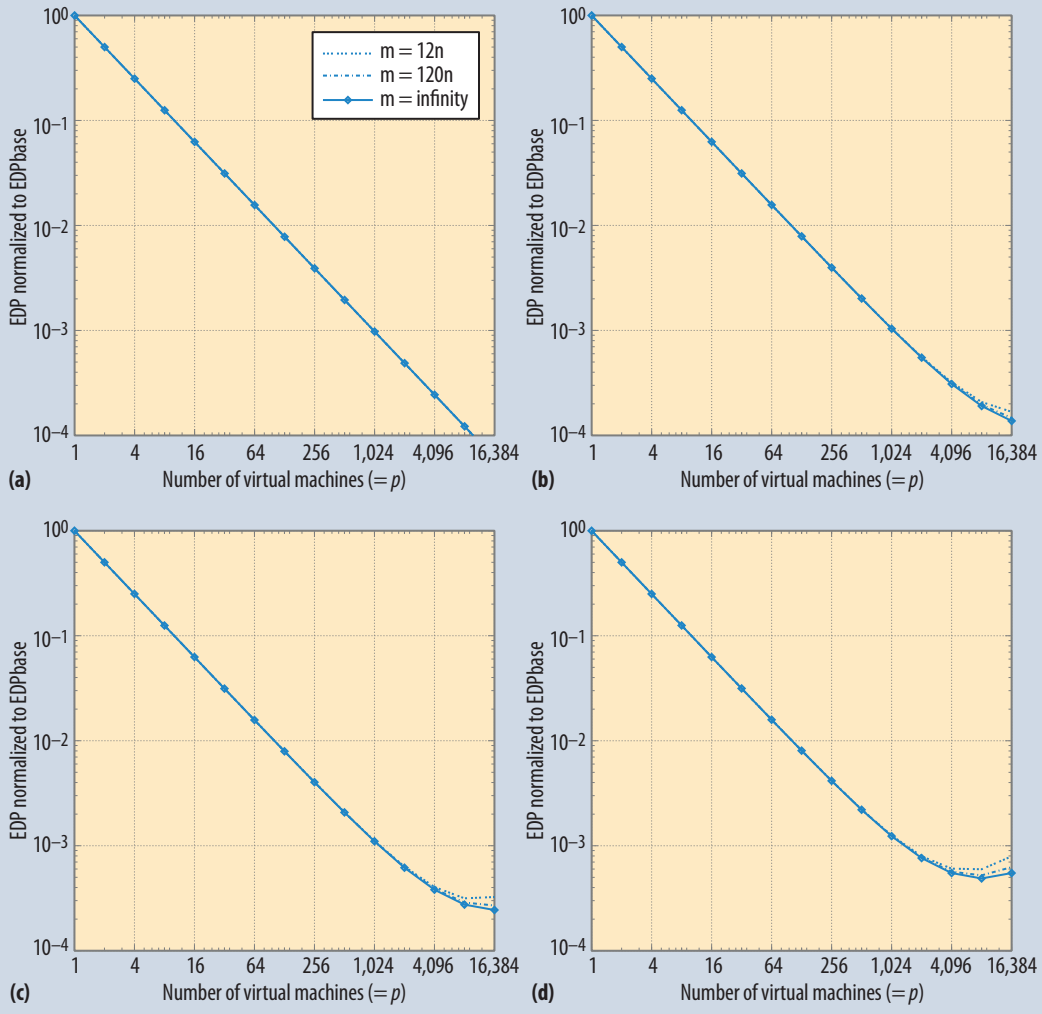

Figure 4. Example of the expectation-based analysis where the total number of available virtual machines is 16,384: (a) $b/a = 1$, (b) $b/a = 2$, (c) $b/a = 3$, (d) $b/a = 5$. When the response time of the slowest node b divided by that of the fastest node a is larger than 3, using all available virtual machines will compromise EDP.

environment operated by a single heterogeneous datacenter but also for a larger computing service comprising many datacenters of varying ages. Because more recent datacenters show better energy efficiency, the effectiveness of the collaboration of multigenerational datacenters can be analyzed in the same way. Our future work will study the power management techniques and scheduling algorithms for the heterogeneous cloud computing environments. C

References

1. M. Palankar et al., "Amazon S3 for Science Grids: A Viable Solution?" *Proc. 2008 Int'l Workshop Data-Aware Distributed Computing*, ACM Press, 2008, pp. 55-64.
2. L.A. Barroso, "The Price of Performance," *ACM Queue*, vol. 3, no. 7, 2005, pp. 48-53.
3. S. Ghiasi, T. Keller, and F. Rawson, "Scheduling for Heterogeneous Processors in Server Systems," *Proc. 2nd Conf. Computing Frontiers*, ACM Press, 2005, pp. 199-210.
4. R. Nathuji, C. Isci, and E. Gorbatov, "Exploiting Platform Heterogeneity for Power-Efficient Datacenters," *Proc. 4th Int'l Conf. Autonomic Computing* (ICAC 07), IEEE CS Press, 2007, pp. 5-14.
5. PassMark Software, "CPU Benchmarks;" www.cpubenchmark.net.
6. S. Pelley et al., "Power Routing: Dynamic Power Provisioning in the Datacenter," *Proc. 15th Int'l Conf. Architectural Support for Programming Languages and Operating Systems* (ASPLOS 10), ACM Press, 2010, pp. 231-242.
7. D. Meisner, B.T. Gold, and T.F. Wenisch, "PowerNap: Eliminating Server Idle Power," *Proc. 14th Int'l Conf. Architectural Support for Programming Languages and Operating Systems*, (ASPLOS 09), ACM Press, 2009, pp. 205-216.

8. R. Gonzalez and M. Horowitz, "Energy Dissipation in General Purpose Processors," *IEEE J. Solid-State Circuits*, vol. 31, no. 9, 1996, pp. 1277-1284.
9. J.A. Rice, *Mathematical Statistics and Data Analysis*, Duxbury Press, 2007.

Sungkap Yeo *is a PhD student in electrical and computer engineering at the Georgia Institute of Technology. His research interests include power management for datacenters and microarchitectural support for the future cloud computing environment. Yeo received an MS in electrical and computer engineering from Georgia Institute of Technology. Contact him at sungkap@gatech.edu.*

Hsien-Hsin S. Lee *is an associate professor in the School of Electrical and Computer Engineering at the Georgia Institute of Technology. His research interests include computer architecture, cybersecurity, and 3D integration. Lee received a PhD in computer science and engineering from the University of Michigan at Ann Arbor. He is a senior member of IEEE and the ACM. Contact him at leehs@gatech.edu.*