The Price Is Right: Towards Location-Independent Costs in Datacenters

Hitesh Ballani[†] hiballan@microsoft.com
Paolo Costa[†‡] costa@imperial.ac.uk
Thomas Karagiannis[†] thomkar@microsoft.com
Ant Rowstron[†] antr@microsoft.com

[†]Microsoft Research Cambridge, UK
[‡]Imperial College London, UK

ABSTRACT

The performance and cost for tenants in today's datacenters depends on the location of their virtual machines within the datacenter. However, a tenant's location is a knob for the provider and is of no interest to the tenant. Hence, this paper argues for *location independent tenant costs in datacenters*. We show how a change in today's IaaS offerings, coupled with a simple pricing scheme, can achieve this. We discuss how such a pricing model can be implemented and show that the consequent increase in system throughput can lead to a win-win situation– tenant costs are location independent and lower while provider revenue increases too.

Categories and Subject Descriptors: C.2.3 [Computer-Communication Networks]: Network Operations
General Terms: Design, Economics, Performance
Keywords: Datacenter, Price, Network cost, Location independence

1. INTRODUCTION

Today's cloud datacenters offer users on-demand computing resources. While the CPU and memory on compute instances (virtual machines, or VMs) are dedicated resources, the network connecting them is shared. Consequently, *the internal network bandwidth achieved by a tenant's instances depends on their location.* Here "location" is a catch-all phrase for inter-related factors such as the placement of tenant instances, neighboring tenants sharing network paths and their load, etc.

Such location dependence means that network performance for tenants can vary significantly [1–4]. This, in turn, impacts the performance for a wide variety of applications; from user-facing web services [2,5] on one end to data-parallel (MapReduce like), HPC and scientific applications [2,6–9] on the other. Location dependent performance impacts the cost too. In cloud settings, tenants pay based on the time they occupy their VMs and since occupancy time is influenced by network performance, tenant cost can be affected by location.

We take the position that the location of virtual machines in a datacenter is a knob for the provider but is of no interest to the tenant. Consequently, the fact that tenant location can impact their performance and their cost is unfair. Many recent proposals address the performance issue by providing location-independent network bandwidth, either through richer topologies [10,11] or through rate limiting [12–14]. However, as we discuss later, the former solution may be an overkill while the latter requires tenants to express their demands and results in network fragmentation.

In this paper, we defer on the performance goal and aim to ensure that tenants costs, in spite of (possibly) variable network performance, do not depend on the location of their virtual machines. To this end, we show how a change in today's IaaS offerings coupled with a simple pricing scheme can achieve *location independent costs in multi-tenant datacenters.* With the proposed model, each virtual machine comes with dedicated aggregate network bandwidth to other VMs for the same tenant. Tenants can, of course, send at more than this *base bandwidth.* As today, tenants are charged for each unit of time they occupy a VM. When a VM generates traffic at less than the base bandwidth, the tenant pays a fixed VM occupancy cost. However, when the VM generates traffic at a higher rate, the tenant cost is proportional to the amount of data transferred.

Hence, a tenant is charged based on whether processing (including local I/O) or the network is its bottleneck resource. We show that such "dominant resource pricing" (*DRP*) ensures the tenant costs do not depend on their network performance and hence, are location independent. Further, by ensuring a lower bound on the network bandwidth for each instance, DRP bounds tenant application performance. Such reining of outliers increases system throughput and actually results in tenant costs lower than today.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Hotnets '11, November 14–15, 2011, Cambridge, MA, USA.
Copyright 2011 ACM 978-1-4503-1059-8/11/11 ...$10.00.

We present a strawman DRP design involving existing techniques for smart VM placement and network fair queuing. While many practical challenges remain, our analysis indicates that both tenant and providers can accrue a lot of benefits by addressing them. An interesting consequence of the proposed design is that it leads to a better alignment of provider and tenant interests than the status quo. Providers have an incentive to improve tenant performance so as to maximize system throughput. This can, in turn, lead to increased provider revenue and presumably, reduced tenant costs.

Tenant applications use resources beyond the two, VMs and internal network, we focus on; for instance, the cloud storage service, external network, etc. The basic idea of charging tenants for their bottleneck resource can be extended to achieve fixed tenant costs in such multi-resource environments. However, this requires performance bounds for each resource which may not be trivial to offer. For example, implementing storage bandwidth guarantees is an open research problem. In this context, the simple, two-resource DRP proposal in this paper represents a first stab at the broader goal of location independent tenant costs in datacenters.

2. LOCATION INDEPENDENT PERFORMANCE

The shared nature of the network in cloud datacenters implies that tenant network performance can vary depending on its "location", i.e., where the tenant's VMs are located, other VMs in the network vicinity and their load, what transport protocols are being used. Recent measurement studies have observed a variation of almost an order of magnitude for bandwidth between VMs in the same datacenter [1–4,9,14,15]. For *long-running*, web facing applications, this implies unpredictable and possibly poor performance. For *job-like*, data parallel applications, network variability impacts completion time and hence, cost.

To counter these problems, two classes of proposals make network performance independent of tenant location. First, richer topologies like fat tree networks [10,11] and the Amazon ClusterCompute network do away with network oversubscription. With such topologies, each physical machine is limited only by its uplink. Assuming machines with 4 VM slots and a 1Gbps interface, a fat tree network ensures a VM can communicate with any other VM at rate $\geq$ 250Mbps. However, *such topologies may be excessive* given the locality of typical datacenter communication patterns [16,17]. This is especially true for cloud settings where a tenant's VMs mostly communicate with other VMs for the same tenant.

The second class of solutions offer bandwidth guarantees to tenants by packing their VMs in a bandwidth aware fashion while using rate limiting to ensure individual VMs do not exceed their allocation [12–14]. However, *this requires tenants to specify their bandwidth demands*. Further, reserving network bandwidth leads to fragmentation of the underlying network since spare capacity is not utilized.

3. LOCATION INDEPENDENT COSTS

This paper advocates that tenant cost should not depend on where a tenant's VMs are allocated within a datacenter. To this end, we propose and evaluate a pricing scheme that yields location independent costs.

Below we discuss a few pricing schemes and use a simple tenant job as a running example to explain them. As part of the job, each tenant VM processes some local data and sends L bytes of processed data over the network to another VM for the same tenant.

3.1 Today's world: VM-only Pricing

Today, tenants pay for VM occupancy. The *price* of a VM is a flat $\$k$ per unit time.[1] Hence, a MapReduce job on N VMs costs $\$kNT$, where T is the time to complete the job. Note that the flat VM price gives the illusion that the network is "free". However, since the completion time T depends on network performance, the *total tenant cost* (kNT) has a hidden network component and can vary with location.

For our example job, a VM that is able to transfer its output at rate b will take $\frac{L}{b}$ seconds to complete and will cost $\$k * \frac{L}{b}$. This tenant cost depends on the network performance achieved (i.e., b). Hence, VM-only pricing does not yield location independent costs.

3.2 Network-only Pricing

A simple way to ensure location independent costs is to charge tenants only for the data they transfer between their VMs. Hence, for each VM, tenants are charged per unit time based on the outbound traffic rate. The price of a VM sending at rate b is $\$k_b * b$, where k_b is the provider specified bandwidth charge. For our example job, a VM transferring its output at rate b will complete in $\frac{L}{b}$ seconds and will cost $k_b * b * \frac{L}{b} = \$k_b L$. Such network-only pricing is independent of the network performance achieved (i.e., b) and hence, location independent. However, such pricing is not very practical since tenants do not pay for VM occupancy. For instance, a tenant whose VMs do not generate any inter-VM traffic would get the VMs for free!

3.3 Dominant Resource Pricing

A practical pricing scheme should include a VM occupancy component so that tenants pay for VM usage. Thus, the challenge is to incorporate an occupancy

[1]For Amazon EC2, $k = \$0.085$/hr for small VMs. This does not include storage and external data transfer costs that are beyond the VM costs.

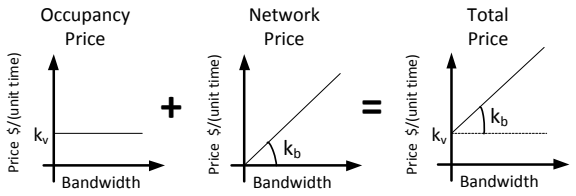

Figure 1: Charging tenants for occupancy and network transfers does not yield location independence.

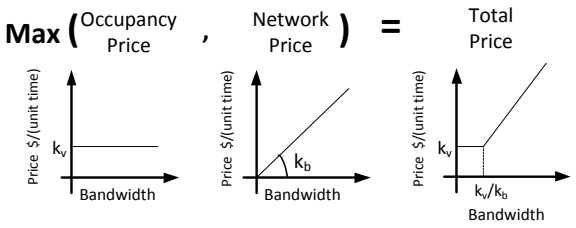

Figure 2: Dominant resource pricing: charging tenants for the greater of the occupancy and network price.

charge while ensuring location independence. To illustrate this challenge, we first consider a simple scheme where tenants are charged for occupancy and network separately. The resulting price curve is shown in Figure 1 which assumes tenants pay a flat VM occupancy charge ($\$k_v$ per unit time) and a network charge that is proportional to the bandwidth they achieve ($\$k_b * b$ per unit time). Hence, the price for each VM is $\$(k_v + k_b b)$ per unit time. For our example job, a VM that transfers its output at rate b will cost $(k_v + k_b b) * \frac{L}{b}$. This cost depends on the network performance achieved and hence, the pricing scheme is not location independent.

Instead, consider a pricing scheme where tenants are charged based on the greater of the two prices– occupancy and network price. Figure 2 shows the resulting price curve as a function of the VM's outbound traffic rate. When a VM generates traffic at a rate lower than $\frac{k_v}{k_b}$ (*base bandwidth* or B_{base}), the VM occupancy price dominates and tenants pay this flat price. However, when a VM generates traffic at a higher rate, the VM price is proportional to the sending rate and hence, the amount of data transferred. Overall, a VM sending at rate b is priced as follows:

$$\begin{array}{rcll} \frac{\text{VM Price}}{\text{unit time}} & = & k_v, & b < \frac{k_v}{k_b} \\ & = & k_b * b, & b \geq \frac{k_v}{k_b} \end{array}$$

For our example job, a VM that transfers its output at a rate b greater than base bandwidth costs $k_b * b * \frac{L}{b} = \$k_b L$, which is independent of network performance. In case the output is transferred at a lower rate, the VM costs $\frac{k_v L}{b}$. This cost depends on the network. However, the provider can ensure location independence by guaranteeing that a VM is always able to send network traffic at a rate greater than B_{base}. If a VM still generates traffic at a lower rate, it is due to the job characteristics and local bottlenecks, and not a consequence of the underlying network performance.

The pricing scheme above, combined with aggregate bandwidth guarantees for VMs, results in location independent costs. Note that, in effect, tenants are charged for their bottleneck resource. When a VM does not use up its guaranteed base bandwidth because the processing or local I/O is the bottleneck, tenants pay for occupancy. Alternatively, when the VM is network limited, tenants pay based on the amount of data transferred. Thus, we call this pricing scheme "Dominant Resource Pricing" (DRP).[2]

3.4 Implementing DRP

A DRP realization should satisfy two main design goals:

1. Base Bandwidth guarantee. Achieving location independence with DRP requires a minor modification to IaaS semantics: a VM, apart from dedicated storage and processing, is also bundled with guaranteed network bandwidth to other VMs for the same tenant. Each VM can send and receive at base bandwidth, and can thus be seen as being connected to the rest of the datacenter by a virtual link whose capacity is B_{base}. This is akin to the hose model [19]. Note that two VMs sending to the same destination VM may be bottlenecked at the destination, and are only guaranteed to get a combined rate of B_{base} instead of $2\mathrm{B}_{base}$. Consequently, a tenant's inter-VM traffic pattern can also cause VMs to not utilize their guaranteed bandwidth.

2. Work conserving allocation. Tenants should not be limited to sending only at the base bandwidth. Any spare capacity should be used by network flows from VMs that can send at higher rate.

Below we sketch a strawman design that satisfies these goals. To guarantee the base bandwidth for tenants, the allocation of their VMs to physical machines should account for the corresponding bandwidth requirements. In recent work, Oktopus [14] presents algorithms for such allocation. The key idea is to ensure that each network link that connects a tenant's VMs has enough capacity to satisfy their bandwidth guarantees. The bandwidth needed for a tenant on a link is the tenant's *bandwidth quota* on the link and depends on the number of VMs for the tenant on either side of the link. The bandwidth quota can, in turn, be used to guide the allocation of VMs. This allocation algorithm is detailed in [14], and is used by DRP to allocate VMs while satisfying goal 1.

The second goal requires that flows that can use spare

[2]The name is inspired by Dominant Resource Fairness [18] which defines fairness for multi-resource settings in terms of a user's dominant share.

capacity be allowed to do so. A proportionally fair way to distribute a link's spare capacity is to give each tenant a fraction of the spare capacity that is proportional to the tenant's "quota" on the link. Consider a link of capacity C and a set A of tenants whose VMs are sending traffic across it. Proportional allocation implies that the flows for a tenant i with bandwidth quota Q_i on this link should achieve an aggregate rate given by: (Bandwidth Quota + Proportional share of spare capacity) $= Q_i + (C - \sum_A Q_j) * \frac{Q_i}{\sum_A Q_j} = \frac{CQ_i}{\sum_A Q_j}$.

The analysis above shows that distributing a link's total capacity in proportion to tenant quotas for the link ensures that the bandwidth guarantees for individual VMs are met and the spare capacity is distributed in a fair fashion. Such allocation of link bandwidth can be achieved through weighted fair queuing (WFQ) in the network. Specifically, network switches are configured with per-tenant weights on each of their ports. The weight for a tenant is its quota on the outbound link, as determined by the VM allocation algorithm, and applies to all traffic from the tenant's VMs. This approach is work conserving since any unused capacity is distributed amongst flows that can use it. Assuming a proper WFQ implementation with per tenant queues, this design also ensures that, irrespective of the transport protocol used (UDP or TCP), a tenant's traffic cannot adversely impact other tenants.

3.5 Design Alternatives

The design sketch above relies on network support. Switches need to be configured with weights for all tenants whose traffic may traverse them. While the VM allocation algorithm can minimize the number of tenants whose flows traverse the core network, this may still mean hundreds of traffic classes for core switches, an order of magnitude more than what is supported by today's switches. Thus, DRP would require switches to be modified to deal with the scale and churn of tenants.

Motivated by the deficiencies of the current design, we are working on end host based approaches to assign weights to tenant traffic. The fact that a single entity owns a datacenter and can instrument all machines makes such an approach particularly attractive. A few recent proposals follow this tact and implement mechanisms on end hosts to achieve various bandwidth sharing goals [14,15,20]. With DRP, the goal is to associate a weight to traffic from *all* VMs for a tenant, and share network bandwidth in a weighted fashion without per-tenant state in the network.

Overall, we admit that many challenges remain. However, the thesis of this paper is the notion of location independent pricing and what it brings to multi-tenant environments. Thus, the rest of this paper focuses on evaluating the benefits of DRP to illustrate that the design exercise is worth pursuing.

4. PRELIMINARY RESULTS

We use simulations to evaluate DRP. While preliminary, our results indicate that apart from decoupling tenant costs from location, DRP can actually improve tenant performance and reduce their costs.

4.1 Simulation setup

Our simulator coarsely models a multi-tenant datacenter. It uses three-level simple tree topology– racks of 40 machines with 1Gbps links connected using a hierarchy of switches. The results in this section involve a datacenter with 16,000 machines and 4 VMs per machine, resulting in 64,000 VM slots. The simulated physical network has an oversubscription of 10:1, a conservative value for today's datacenters [10].

Tenant jobs. As with our example job in Section 3, tenant jobs comprise tasks on individual VMs and a set of flows transferring data between the tasks. Each tenant task is a source and destination for one flow, and all flows for a given tenant are of the same length. While admittedly naive, we believe this is a good first-order representation of the network component of real world data processing jobs that often run in datacenters. The number of VMs making up a job and the amount of data being shuffled between these VMs are both exponentially distributed with means of 50 and 50GB respectively. The "ideal completion time" for a job is its completion time if it were running in isolation. For a job with L bytes of data to be shuffled between VMs, the ideal completion time is $\frac{L}{1Gbps}$.

To capture the operation of today's cloud datacenters, we simulate tenant jobs arriving over time. We vary the job arrival rate to vary the **target occupancy** of the datacenter in terms of the number of VMs. Assuming Poisson tenant arrivals with an arrival rate of λ, the target occupancy in a datacenter with total M VM slots is $\frac{\lambda N}{M} \cdot \frac{L}{1Gbps}$, where N is the mean VMs per job and L is the mean amount of data shuffled. *Tenant jobs that cannot be accommodated on arrival are rejected.*

Baseline scenario. We compare DRP against a Baseline scenario representative of datacenter operation today. With Baseline, tenant VMs are greedily allocated close together. Further, tenant flow rates are given by their max-min fair share. As a contrast, DRP involves allocating VMs with base bandwidth guarantees [14]. Network bandwidth is distributed amongst tenants in a proportionally fair fashion.

4.2 Performance Analysis

We simulate the arrival and execution of 20,000 tenant jobs with varying target occupancy. Figure 3 shows the fraction of rejected job requests. Baseline rejects 10-28% more jobs than DRP, even though DRP only accepts a job if bandwidth guarantees of its VMs can be satisfied. The improvement offered by DRP can be

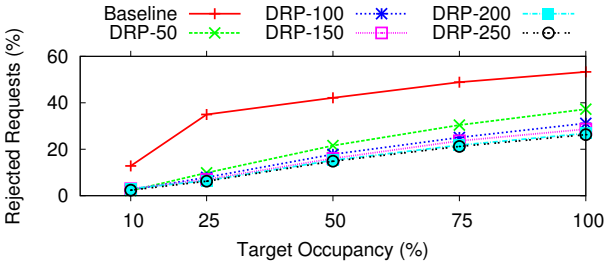

Figure 3: Percentage of requests that are rejected. Base Bandwidth is x Mbps for DRP-x.

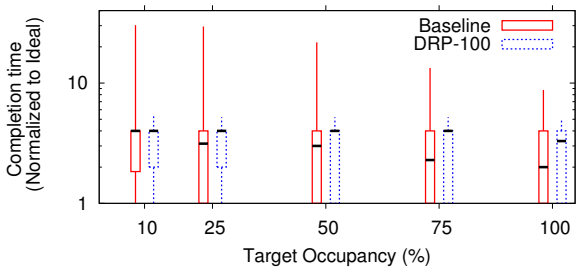

Figure 4: Completion time percentiles (normalized to ideal completion time). Plotted values are 1-25-50-75-95th percentiles.

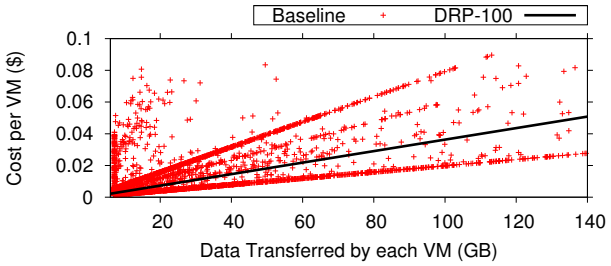

Figure 5: VM costs for tenant jobs with different amount of data transferred. Occupancy = 75%.

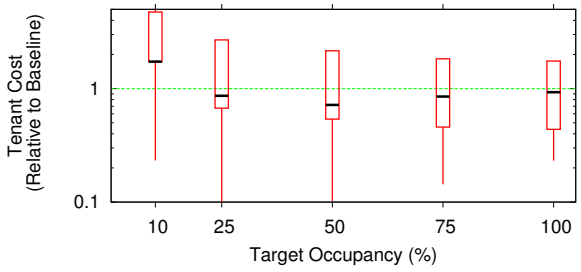

Figure 6: Tenants costs with DRP-100 relative to baseline (log-scale).

explained through the performance of individual jobs. Figure 4 plots the percentiles for normalized completion time of jobs. With DRP, the median completion time increases by up to 50%. Part of the reason is that, compared to Baseline, there are more jobs running concurrently and, hence, there is greater network contention. However, the 95th-percentile completion time with DRP is significantly less than Baseline (3-5x less), which demonstrates that DRP reduces the disparity in job performance. Further, DRP bounds the worst-case completion time. For instance, with DRP-100, VMs have an aggregate bandwidth of at least 100Mbps, so the job completion time can at worst be 10x the ideal completion time. Thus, *by providing lower bounds on tenant network performance and preventing outlier jobs, DRP is actually able to accommodate more jobs.*

4.3 Cost Analysis

We now analyze tenant costs. With Baseline, each VM for a job costs \$kT, where T is the job completion time. We use k = \$0.085/hr based on the price of Amazon EC2 small VMs. With DRP, tenants are charged based on the pricing model in Section 3.3. We assume tenant VMs can generate data at sufficient rate, so a VM transferring L bytes of data costs $\$k_bL$. The bandwidth charge ($k_b$) is determined based on provider revenue neutrality. For each experiment, we calculate k_b such that the provider's total revenue with DRP is the same as with Baseline. For instance, for DRP-100 and target occupancy = 75%, we find k_b = \$3.63*10^{-4}/GB. Figure 5 is a scatter plot of the total cost per VM for different jobs transferring varying amounts of data. For Baseline, the VM cost depends on the job completion time and hence, network performance. Since the latter can vary significantly, VM costs for jobs shuffling the same amount of data vary too. As a contrast, *with DRP, VM costs are do not depend on tenant location. Instead, they are proportional to the amount of data transferred between a tenant's VMs.*

Beyond predictable costs, DRP can actually reduce tenant costs. Figure 6 shows percentiles for the ratio of tenant costs with DRP to Baseline. At very low occupancy, median tenant cost with DRP is higher since Baseline is able to ensure good performance for almost all tenants. However, as the target occupancy increases, tenants pay less with DRP. This is important since providers like Amazon operate their datacenters at 70-80% occupancy [21]. In the median case, tenants pay 7-30% less. As before, this results from the fact that DRP is able to accommodate more tenant jobs by avoiding outliers. The resulting increase in system throughput, in turn, lowers per tenant cost. Though the analysis above is based on provider revenue neutrality, we also consider a win-win scenario where the provider is able to increase its revenue and still pass on cost benefits to tenants. For instance, *providers can increase their revenue by 8% and reduce median tenant cost by 9% at an occupancy of 75%.*

5. DISCUSSION

The notion of location independent costs raises many issues. Here we discuss a few concerns relevant to DRP.

Pricing complexity. At first glance, tenants may

find the proposed pricing model less intuitive than today's VM-only pricing. With DRP, our goal was to ensure simple tenant costs, albeit at the expense of simple pricing. Simplified total costs is what tenants with *job-like applications* care most about.

We admit that today's pricing model appears more amenable to long-running, *user-facing services* (e.g., websites) since the tenant monthly bill is fixed. However, there is no lower bound on the network and hence, application performance. With DRP, a tenant's bill may depend on the inter-VM traffic which, in turn, probably depends on the number of user requests served. Many cloud components like storage and external bandwidth are already charged based on usage. Hence, even today, part of a tenant's bill depends on the popularity of their service and with DRP, the same will apply to VM costs.

Usage pricing. With DRP, VMs are priced based on occupancy while the network is priced based on usage. This is because the network is a shared resource. Proposals like [10–12,14] turn the network into a dedicated resource by guaranteeing the network bandwidth for tenants, thus allowing for occupancy based network pricing. However, this hurts datacenter efficiency since spare network capacity is wasted. DRP avoids this.

Spurring innovation. DRP better aligns tenant and provider interests than the status quo. Providers have flexibility regarding distributing spare network capacity; they are not obligated to divide it fairly. For instance, tenant flows could be prioritized based on their length which would cause the corresponding jobs to finish faster and free up VM slots for subsequent tenants. Such network scheduling can improve provider throughput and revenue by passing jobs faster through the system. Providers may even pass on some benefits to tenants in the form of reduced costs. In effect, DRP gives providers an incentive to improve tenant performance; as a contrast, it can be argued that today's pricing results in a disincentive for providers to do so.

Provider flexibility. DRP offers a lot of flexibility to providers. It allows for tiered pricing– various VM classes (small, medium, ...) with different occupancy charge ($k_{v1} < k_{v2} < \ldots$) and different base bandwidth ($\frac{k_{v1}}{k_b} < \frac{k_{v2}}{k_b} < \ldots$) can be offered. Similarly, providers can implement spot pricing with DRP. The price of a VM would be characterized by an occupancy charge ($k_v(t)$) that can vary over time to reflect the datacenter utilization. Each VM still comes with the same base bandwidth guaranty, so the bandwidth charge ($k_b(t)$) will vary accordingly. As today, providers can use this to improve overall utilization.

To summarize, this paper makes the case for location independent tenant costs and shows how charging tenants for their bottleneck resource can achieve this. By preventing jobs with very poor performance, such "fair" pricing even improves overall datacenter performance and can result in lower tenant costs. Moreover, by decoupling cost from performance, DRP aligns tenant interests with those of the provider and encourages the latter to innovate. While many implementation challenges remain to be addressed, the fact that DRP offers predictable costs and can thus remove a significant hurdle to cloud adoption makes this a promising exercise.

6. REFERENCES

[1] A. Li, X. Yang, S. Kandula, and M. Zhang, "CloudCmp: comparing public cloud providers," in *IMC*, 2010.

[2] J. Schad, J. Dittrich, and J.-A. Quiané-Ruiz, "Runtime measurements in the cloud: observing, analyzing, and reducing variance," in *VLDB*, 2010.

[3] "Measuring EC2 performance," http://bit.ly/48Wui.

[4] "Network performance in virtual infrastrucures," http://bit.ly/aZxdhn.

[5] A. Iosup, N. Yigitbasi, and D. Epema, "On the Performance Variability of Production Cloud Services," Delft University, Tech. Rep. PDS-2010-002, 2010.

[6] M. Zaharia, A. Konwinski, A. D. Joseph, Y. Katz, and I. Stoica, "Improving MapReduce Performance in Heterogeneous Environments," in *OSDI*, 2008.

[7] E. Walker, "Benchmarking Amazon EC2 for high performance scientific computing," *Login*, 2008.

[8] Q. He, S. Zhou, B. Kobler, D. Duffy, and T. McGlynn, "Case study for running HPC applications in public clouds," in *HPDC*, 2010.

[9] G. Ananthanarayanan, S. Kandula, A. Greenberg, I. Stoica, Y. Lu, B. Saha, and E. Harris, "Reining in the Outliers in Map-Reduce Clusters using Mantri," in *OSDI*, 2010.

[10] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "VL2: a scalable and flexible data center network," in *SIGCOMM*, 2009.

[11] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *SIGCOMM*, 2008.

[12] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "SecondNet: A Data Center Network Virtualization Architecture with Bandwidth Guarantees," in *CoNext*, 2010.

[13] P. Soares, J. Santos, N. Tolia, and D. Guedes, "Gatekeeper: Distributed Rate Control for Virtualized Datacenters," HP Labs, Tech. Rep. HP-2010-151, 2010.

[14] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards Predictable Datacenter Networks," in *SIGCOMM*, 2011.

[15] A. Shieh, S. Kandula, A. Greenberg, and C. Kim, "Sharing the Datacenter Network," in *NSDI*, 2011.

[16] S. Kandula, J. Padhye, and P. Bahl, "Flyways To Decongest Data Center Networks," in *HotNets*, 2009.

[17] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. S. E. Ng, M. Kozuch, and M. Ryan, "c-Through: Part-time Optics in Data Centers," in *SIGCOMM*, 2010.

[18] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant Resource Fairness: Fair Allocation of Multiple Resource Types," in *NSDI*, 2011.

[19] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. van der Merive, "A flexible model for resource management in virtual private networks," in *SIGCOMM*, 1999.

[20] B. Raghavan, K. Vishwanath, S. Ramabhadran, K. Yocum, and A. C. Snoeren, "Cloud control with distributed rate limiting," in *SIGCOMM*, 2007.

[21] "Amazon's EC2 Generating 220M," http://bit.ly/8rZdu.