Pricing the Cloud

Ian A. Kash iankash@microsoft.com
Peter B. Key peter.key@microsoft.com

October 20, 2015

Abstract

Current cloud pricing schemes are fairly simple, but what is the future of cloud pricing? We discuss a number of the economic issues shaping the cloud marketplace, and open questions they yield. We then explore what the current state of research in economics and computer science has to say about some of these questions and what it suggests for future evolution of cloud pricing.

Keywords: J.4.a Economics ¡ J.4 Social and Behavioral Sciences ¡ J Computer Applications J.8.e Electronic commerce ¡ J.8 Internet Applications ¡ J Computer Applications K.1.a Markets ¡ K.1 The Computer Industry ¡ K Computing Milieux

1 Introduction

Grossman[10] informally defined *clouds* as "cluster[s] of distributed computers providing on-demand resources ...over the Internet" at scale, which still serves as a useful definition. The scale here refers to data-center size unit, and the largest data centers may have of the order of 100,000 servers. This scale naturally gives savings, and an influential white paper from Microsoft[11] argued the benefits of outsourcing IT infrastructure form in-house or private-cloud to public cloud. The shift from owned-infrastructure to public cloud has accelerated over past few years, helped also by simple interfaces, and visualization. Moreover, cloud computing itself is becoming richer. Although the dominant cloud services still primarily sell computing instances (Amazon EC2, Google's Compute Engine, Microsoft's Azure Compute), now different types of resources are also being offered (such as Storage), Platform as a Service(PaaS) offerings such as App services are appearing, as well as more sophisticated products, such as Azure ML.

In simple economic terms, current cloud providers form an oligopoly. There are certain natural constraints on capacity, which argues the equilibrium pricing for raw resource approximates a Cournot equilibrium. This equilibrium gives prices above the competitive price (where prices equals production costs), however with decreasing resource costs, this essentially becomes an almost frictionless commodity market. For example, Microsoft has committed to match Amazon's pricing for basic infrastructure. Hence the natural reaction of the providers is to want to provide service differentiation, and create a richer set of services which have high added value to the users, hence the desire to "move up the stack", from vanilla IaaS to rich PaaS offerings and beyond to SaaS (Software as a Service) and complete solutions. The benefits to end users are higher value services. For providers, the potential profits from these richer services provide the needed incentive to build market share in the commodity market. Orthogonal to this is the use of variants of price discrimination that charge different amounts for (essentially the) same products as a way to capture more of the value created, and hence earn more revenue. We can interpret current forms of tiered or menu pricing for compute instances as examples of this.

But what of the future, for cloud economics and cloud pricing? With more and more computation occurring in the cloud, this is an important question. In this article, we look at what the "Econ-CS" research field (by which we mean the cross-disciplinary field that lies at the intersection of Computer Science, Game Theory, Operations Research, and Economics) has to say about this question.

2 Goods and Services

The jury is still out on what the fundamental "goods" are with respect to pricing the cloud. It is a messy place with different types of resources available (storage, bandwidth), different application types (server, batch), and different service types (IaaS,SaaS), all of which come with variations in attributes such as quality of service measures and service level agreements. The model that has been most commonly been adopted to date is a "utility" model: find something whose use can be measured and charge per unit, much like electricity and water markets work on the consumer-facing side. Thus we have AWS and Azure offering prices for VM hours, GB of storage, and external bandwidth. Even more sophisticated services such as load balancing and database use are metered. To some extent this is reasonable because the amount of capacity used can be viewed as a proxy for both the value created for customers and the costs of the provider, but in many cases it seems a crude one. Perhaps the biggest virtue of this approach is that it is simple, both to understand and to implement.

Electricity is a good reference point in this respect; power companies charge residential customers for usage as a fixed rate because it is all their technology allows them to do. One of their motivations to moving towards smart meters is the eventual ability for finer-grained pricing that reflects the fact that their costs vary throughout the day with demand (as exists today for larger customers). In contrast, airline yield management teams engage in exquisite price discrimination where every customer on a flight may have paid a different price based on market segmentation, shifts in demand over time, and the full context in which that seat is purchased.

Pricing of Internet or network services also provides a natural comparison for cloud pricing. Pricing telephony evolved towards non-linear, time-varying, usage-sensitive pricing for long-distance, coupled with flat fee for connection and local traffic; as the provisioning cost of telephony has declined, so has the usage sensitive component of the pricing for fixed network telecoms. The primary good for Internet pricing is bandwidth, which is provided through congestible resources. Thus it looked as though it was a natural candidate for non-linear usage-sensitive pricing [16]. But instead flat rate pricing became almost universal. Current practice is to offer end-customers limited menu pricing, with a flat fee plus a fixed additional fee dependent on the maximum bandwidth rate and maximum bandwidth transferred, which is similar to mobile telephony pricing.

These flat-fee unlimited use prices appears to fly in the face of economic theory which argues that usage sensitive pricing is optimal. Odlyzko[18] argues that declining provisioning costs mean that customer's demand for simplicity override the benefits of usage-sensitive charging, while Sundararajan[20] argues from a theoretical model that any positive transaction cost associated with implementing a usage based charging scheme makes it optimal for sellers of *information goods* to offer customers a combination of usage based pricing and unlimited use fixed-fee. An information good is defined as having zero variable production costs, which does not exactly hold for bandwidth provision and is quite far from the truth for cloud providers. Thus, we expect cloud pricing to take a somewhat different route.

Compute capability or Computational Assets? One common task customers want to do on the cloud is run batch tasks such as MapReduce jobs. What is the object of value that should be priced here, the cluster and software associated with running the job or a service that takes the job and runs it for you? Both business models exist, even within the same company (e.g. Google offers Cloud SQL to allow customers to provision their own SQL databases as well as BigQuery to allow customers to simply submit individual queries). In such instances, pricing needs to interact accordingly, but how should this be done in practice?

Constraints and Desires. In some types of cloud systems, there is no choice about when to do work: when a user makes a request of a web-facing system it needs to be responded to immediately. In others, there is some flexibility about when work is scheduled, for example a job that needs to be run sometime between the close of business one day and the start of business the next. How should cloud services allow customers to express this flexibility? Provide no guarantees and leave it up to the customer to decide when to do the work each day? Provide a reservation service? Accept jobs with hard constraints about when they are run and an SLA about meeting those constraints? Allow a more expressive utility function that explains how value changes depending on when the work is completed? Similarly, how should providers share the information that they have? For example, what historical data should Amazon provide about bids in its spot markets? Such combined scheduling and pricing models are an active area of research in a variety of contexts, and the right answer remains unclear [13].

Unidimensional or Multidimensional? What are the fundamental elements involved in computation? Should they all be bundled together and sold as a unit as a notional VM with associated connectivity, bandwidth, storage etc or sold as a flexible computational entity where all these aspects can be customized? We call the former unidimensional, because the only decision a customer need make is how many of these units are required (of course in practice there may be a small number of different options rather than literally a single type of VM), and the latter multidimensional because the customer must express preferences along a number of axes. From an economic perspective, these two models lead to very different techniques, with those for a unidimensional world much better understood. In the remainder of this article, we examine how research has begun to address some of the questions we have raised, roughly splitting it using this lens of unidimensional vs multidimensional approaches.

3 Pricing unidimensional offerings

The simplest type of Cloud Pricing is for "unidimensional" resources, such as basic IaaS offerings or raw compute power. Since resources are continually being sold, this argues for a stochastic demand model, where resources behave as a queuing system. Within this framework, customers can value jobs differently, and also have different valuations for response time.

3.1 A hybrid market : Pay as You Go and a Spot Market

Abhishek et al. [1] model customers as having different values for jobs by assuming that customers belong to different classes, where each class has a different value v_i for job completion, a different distribution for waiting time "costs" $c \sim F_i(c)$, and a different arrival rate. The market is a hybrid one, comprising a Pay as You Go (PAYG) market offering a fixed priced, p, per unit time, and a Spot Market with a variable clearing price. A job pays an amount m for using the resource, and if it spends a time w in the system, then the payoff to a class i customer with waiting time cost c is $v_i - cw - m$, the difference between the value to the customer and the cost incurred. Hence the expected pay-off to a job entering the PAYG market is $v - (c + p)$, where we have assumed the expected service time is normalized to one and the PAYG market is assumed to have sufficient capacity to serve all demands with negligible queueing time. The Spot Market has finite capacity and runs an auction mechanism, eliciting "bids" from customers and giving priority to those jobs which bid more, pre-empting jobs as necessary, where pre-empted jobs rejoin the queue. Hence the expected cost to a job in the spot market is $\hat{w} + \hat{m}$, where the expected waiting time $\hat{w} \geq 1$, since the job incurs possible additional delay while waiting or preempted.

They adopt a static equilibrium model and assume customers are rational agents, and choose the option that maximizes their expected payoff, whereas the market provider chooses the price p and to maximize expected revenue. Under mild assumptions, they show that behavior can be characaterized regardless of the precise auction mechanism used, and results are insensitive both to the distribution of arrival times and to the service time distribution. They show that, for each class i there is a cut-off c_i below which jobs participate in the Spot Market, that the payment function $m(c)$ must be increasing in c, and that there is a unique vector of cut-offs, $c(p)$ for a given price p.

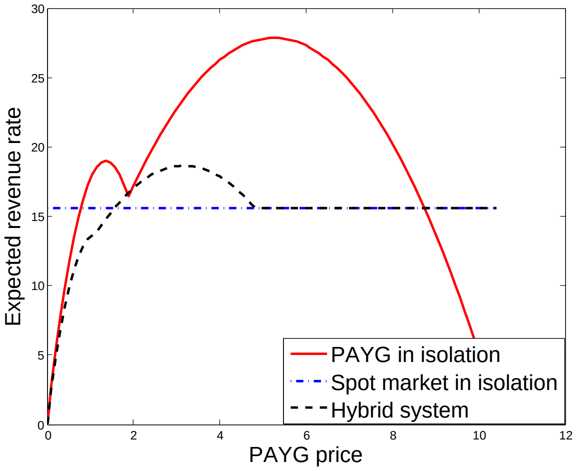

Figure 1: Fixed Pricing, Spot Pricing and a Hybrid Market.

More surprisingly, typically PAYG raises more revenue that the Hybrid mechanism: indeed, it is always the case when all classes participate in PAYG - i.e. when for the optimal hybrid price p^h, $p^h \leq v_i - c_i(p^h)$ for all i. Figure 1 shows an example when not all classes participate in the Spot Market (for high price) and still PAYG raises more revenue. Why does this happen? In any Hybrid mechanism there is no way to prevent high-value low-waiting-time-cost jobs choosing the Spot market when they would have been willing to pay a higher PAYG price; in other words, "rich" jobs can choose the "cheaper" class, with a consequent loss of revenue. This assumes that there are no extra costs associated with preemption. If such costs are sufficiently high they would allow the spot market to avoid cannibalizing the primary market. Of course one way to avoid the cannibalization effect would be using a "damaged goods" approach, where delay is deliberated introduced into the secondary market as a way of ensuring that significant costs are felt.Alternatively, a spot market may be run for reasons other than revenue optimality, such as to gain market share by attracting new small-scale customers.

Their findings are consistent with anecdotal evidence that Amazon makes it difficult to operate in their spot market at scale, and with the findings of Ben-Yehuda et al. [2] who found Amazon controls reserve prices and causes them to spike, suggesting that Amazon may be making its spot market behave like a menu-priced market. Even in cases where a spot market would be beneficial from a revenue perspective, it may be desirable not to run one to avoid the complexity it creates for customers.

3.2 Adding time

Even when a single good is being priced, there is room for considerable richness beyond simply asking about willingness to pay (whether through a fixed price or an auction). A simple extension involves adding time, specifically to deal with the scheduling and pricing of batch jobs (e.g., MapReduce, DryadLINQ, or SCOPE), which extends the idea of admission control. For many such jobs, it is critical that they are completed by a particular deadline. For others, there may not be a hard deadline but the value of the job depends on how soon they can be completed. A line of work (e.g. [13]) looks at this scheduling problem, using algorithms based on linear programming approaches. While the primary focus is on deciding which jobs to schedule and when, this work has also looked at how to charge prices such that it is optimal for job owners to truthfully reveal how their value for the job changes depending on when it is done.

3.3 Pricing Storage

For many cloud offerings however, control resides with the customer, and the provider is left guessing how the resources will be used. A natural alternative is to ask the customer for information about future resource usage and price accordingly. Ceppi and Kash [7] explore pricing for storage, which asks in advance for predictions about how much will be used in each month, in contrast to current pricing which charges per month based on the total that was used. While complicated probabilistic information could in principle be requested, to keep things simple from the perspective of the customer they assume this consists solely of lower and upper bounds on usage over a time interval, with the provider using its own models to understand how that will affect usage in each period. For the provider this is helpful information to make capacity planning decisions, and in particular allows for more efficient operation by reducing both the amount of storage that must be kept free to allow for future use and the frequency with which a customer exceeds the amount of storage locally available and ends up getting its data split to, e.g., another rack.

While this information is useful, getting it presents a significant pricing challenge for the cloud. Since tighter estimates are more useful to the provider, these should be rewarded with lower prices. At the same time, this information is only useful if it is accurate, so the prices need to be such that customers are not incentivized to report inaccurate estimates. They provide a pricing scheme that provides these incentives, as well as ensuring that the provider covers its costs. The main idea is to quote customers a price per GB per month, just as is done today, except that these prices are personalized based on the customer's report. If the report proves accurate (i.e. the customer does not violate the lower or upper bound), that is all the customer pays. If the bound is violated, the customer pays an additional penalty charge based on how badly it was violated, and these penalty charges are carefully calculated to provide correct incentives.

While they focus on pricing one particular aspect of cloud services and eliciting one particular piece of information about future usage, this is an area ripe for further exploration. Better understanding of customer plans more broadly could lead to substantially higher utilization of resources, and thus substantially lower costs.

3.4 Competition in the Market Place

Competition is important, but there are few existing models of competition in the cloud. Anselmi et al. [4] look at a stylized tiered-model of the cloud where users seek service from Service providers (SaaS), who themselves buy resources from providers (IaaS or PaaS), and consider both congestion and pricing. In this vertical market structure, under their model the profits of the IaaS or PaaS providers decrease as competition intensifies, whereas that of the service providers does not; in effect the SaaS providers maintain their market power.

Looking back at the (much simpler) Internet pricing literature is salutary: simple modes offering different levels of QoS looked appealing initially, but then looked fragile in the face of competition. As in the case of the so-called "Paris Metro Pricing" proposal [17], where better QoS was provided solely by charging more for a "better" service, taking its name from pricing used at one time in the Paris metro, where first class and second class carriages were identical, but tickets cost more for first class carriages, and hence were likely to be less crowded. Gibbens et al [9] showed that such differential pricing was not sustainable under competition, since an operator offering say two levels of service by splitting capacity with differential charging, would lose out to a competitor offering a single price.

4 Problems of multidimensional goods

Multidimensional approaches explicitly grapple with the fact that VMs are not monolithic entities but are instead bundles that bring together resources such as CPU, memory, and bandwidth, illustrated in Figure 2, where each demand has a minimum and maximum requirement for a resource. Since different applications have different needs for these resources, gains from trade are possible. But realizing those trades requires confronting issues at a variety of levels: implementation, information, and mechanism design (i.e., how resources allocation and pricing decisions are made).

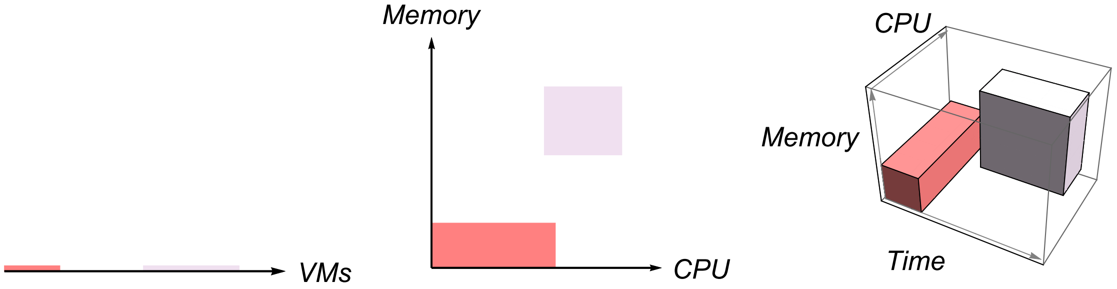

Figure 2: Unidimensional and Multi-Dimensional Demand Profiles.

4.1 Delivering Flexibility

For implementation, actually providing such flexible bundles is not a trivial task. Some resources, like processor time or memory, can be handled by extending standard paradigms for scheduling and resource allocation on a single machine to datacenter settings. But what about resources such as bandwidth, where providing guarantees requires bringing together network design, VM placement, routing, and congestion control? The systems community has seen a large body of work on performance isolation for network, middlebox, and storage resources, as well as their combination into a "virtual datacenter [3]." As the ideas from this research percolate into datacenters, cloud service providers will begin to be able to address a problem with current pricing schemes: costs depend on factors outside the control of customers. For example, if a customer's VMs are located such that there is significant network contention, jobs will take longer to run, and, since VMs are charged based on how long they are used for, will cost more [19].

Once resources can be delivered, the cloud provider needs sufficient information to determine which resources should be delivered. From a customer perspective, this requires understanding both how a job or system will perform with different resource bundles and how to express this knowledge to the cloud provider. Ideally, such job performance profiling would be largely automatic. Jalaparti et al. [14] studied this problem with applications from three domains: data analytics, web-facing, and HPC. They found that in all three settings similar performance could be achieved with several quite different bundles of resources. For MapReduce jobs, they were able provide reasonable predictions for what performance would be with different bundles composed of network and compute resources. However, such batch jobs are perhaps the easiest setting in which to make such predictions, and much more work is needed in this area.

4.2 Fair Division

Once customers know what they want and how to explain this to the cloud provider and the cloud provider is able to deliver resources it promises, there remain significant challenges to determining how to operate the market that determines who gets what resources. While mechanism design is well understood in unidimensional settings, much less is known in multidimensional ones. However, one line of work has managed to bring together good properties in terms of both fairness and incentives. It assumes that the decision on whether or not to admit or schedule a request has already been made, and focuses on the consequent allocation. Further, it assumes that the system is work-conserving in the sense that it attempts to deliver as much of the available resources to each job as it can while being "fair" as opposed to simply delivering the amount required to meet an SLA.

Ghodsi et al. [8], extended earlier ideas about max-min fairness to multiple resources, under the assumption that people's preferences for them are "Leontief," which means they only want them in some fixed ratio. A good example is hot dogs and buns, where I only want a hot dog if I have a bun to go with it and vice versa. In a cloud setting, this makes sense when someone needs, e.g., a particular amount of compute and memory to create a useful VM but then can create a large number of copies of that VM which can all do useful work. In this setting, instead of maximizing the minimum total amount of resources a person gets, their algorithm maximizes the minimum amount of the resource that person requires the most of (relatively speaking). Thus, they called this approach Dominant Resource Fairness (DRF). This can be extended to deal with customers who have paid different amounts by weighting them appropriately. This allows systems with a number of key guarantees:

1. no one is worse off than if all resources had just been divided evenly,
2. no one prefers the bundle of resources someone else got to their own,
3. any left over resources are not usable by anyone, and
4. no one has an incentive to misreport about what the bundles of resources they need look like.

This idea also provides the inspiration for Mesos, a thin management layer now in Apache, which allows different applications (e.g. Hadoop and Spark) to share the same underlying pool of resources [12].

4.3 Fairness and Time: Dynamic Fairness

Systems like Mesos take a static view of the world, and try their best to maintain their fairness guarantees at each point in time. Kash and colleagues [15] study a version of the problem where not all applications necessarily exist at the same time, and show that these techniques extend to this setting. However, if resources cannot easily be taken away from an application there is an inherent tension between efficiency (putting resources to work now) and fairness (saving resources for later arrivals). They give two different algorithms, which each relaxes one of these two guarantees from DRF while preserving the other.

4.4 Fairness with Multiple Entities

Cloud platforms have a large ecosystem of applications and services running on them. This includes both services provided by the cloud provider and services built by customers and then sold onward to other customers. When such services communicate, there are three different economic relationships involved: two between the services and the cloud provider and one between the services. How should these multiple economic relationships affect the allocation of bandwidth? Ballani and colleagues [5] demonstrate that this scenario is already common and propose an answer to this question using a notion of "upper bound proportionality," which limits the bandwidth that a service can acquire regardless of the amount paid by those it communicates with. This prevents services that communicate widely from claiming a disproportionate share of the network.

4.5 Fairness in a Cooperative Game Setting

Customers' willingness-to-pay for resources is handled in the above fairness settings by using weighted allocations — effectively conflating a proportionally-fair principle with the chosen fairness method (such as DRF). An alternative approach is adopted by Blocq et al. [6] who introduce the Shared Assignment Game, borrowing ideas from co-operative game theory to discuss both allocation and pricing in a static context. In the Shared Assignment Game, sellers have multidimensional resources, and buyers need bundles of resources to execute their jobs. Buyers have values for their jobs, and the game is a *Cooperative* game, where buyers and sellers are the agents, and the objective is to find the coalition of agents that maximizes (say) social welfare — the aggregate welfare of the coalition. Clearly a coalition which doesn't have both sellers and buyers has zero value, and the value of a coalition is defined as the maximum welfare achievable from a feasible assignment, where a feasible assignment matches jobs to resources that respects capacity constraints. As a simple example, the sellers could be individual servers with multidimensional attributes (CPU,Memory, BW) and the buyers each have a number of jobs with associated values. Pricing is performed using the Shapley value as an instrument for revenue sharing: this takes the optimal welfare from the grand coalition of all agents, and apportions it, by calculating what each buyer or seller "contributes", by looking at their contribution to each sub-coalition, randomizing over the way sub-coalitions are formed. Such an apportionment is a "fair" division, in that it rewards those who contribute the most, and can be derived staring from an axiomatic approach to fairness. Calculating the allocation and the Shapley value are both computationally hard, so approximation methods are needed. Although such pricing is not strategy-proof, it is reasonably resistant to natural manipulations, such as "splits", where buyers spit their goods, or "bluffs" where fake goods are declared. Simulations suggest that using the Shapley value as a basis for pricing could improve both welfare and revenue. While this presents an interesting viewpoint, literally implementing it would introduce a number of difficulties. Nevertheless, it provides a useful perspective to inform pricing decisions.

5 Conclusion

We have described some of the pricing issues inherent in pricing the cloud, and some of the state-of-the art research work. We have deliberately focused on the simplest settings, such as pricing resources or jobs in an IaaS or PaaS setting, however the approaches taken and issues raised apply much more broadly, to SaaS offerings and beyond: for example, the fundamental dichotomy between unidimensional or multidimensional service specification and allocation applies generally. Within a multidimensional settings, time behaves as a dimension bringing its own unique issues. As services evolve, the type of resources may become even richer. At the present time, understanding of multidimensional pricing is embryonic: the work on fairness gives a handle on allocation, but even this is partial. The fairness framework doesn't account for the future effect on demand that a fair-allocation might have (demand externality), and isn't well integrated with temporal requirements. This is an important and fruitful area for research. Multidimensional scheduling and pricing offers greater potential for increasing both customer satisfaction and revenue; but militating against increasing complexity in pricing and scheduling is the customers' need for simplicity: pricing schemes need to be understandable by a user or their agent. A complementary strand of research and innovation is needed to understand how best to capture and reflect user requirements.

More broadly, it is clear that pricing is in its infancy in the cloud. The research frontier is moving rapidly, and we expect that in the coming years the approaches used by cloud providers will do so as well.

References

[1] Vineet Abhishek, Ian A. Kash, and Peter Key. Fixed and market pricing for cloud services. Working paper. Manuscript at http://arxiv.org/abs/1201.5621.

[2] Orna Agmon Ben-Yehuda, Muli Ben-Yehuda, Assaf Schuster, and Dan Tsafrir. Deconstructing amazon ec2 spot instance pricing. *ACM Trans. Econ. Comput.*, 1(3):16:1–16:20, September 2013.

[3] Sebastian Angel, Hitesh Ballani, Thomas Karagiannis, Greg O'Shea, and Eno Thereska. End-to-end performance isolation through virtual datacenters. In *Proceedings of the 11th USENIX conference on Operating Systems Design and Implementation*, pages 233–248. USENIX Association, 2014.

[4] Jonatha Anselmi, Danilo Ardagna, John Lui, Adam Wierman, Yunjian Xu, and Zichao Yang. The economics of the cloud: price competition and congestion. *ACM SIGecom Exchanges*, 13(1):58–63, 2014.

[5] Hitesh Ballani, Keon Jang, Thomas Karagiannis, Changhoon Kim, Dinan Gunawardena, and Greg O'Shea. Chatty tenants and the cloud network sharing problem. In *NSDI*, pages 171–184, 2013.

[6] Gideon Blocq, Yoram Bachrach, and Peter Key. The shared assignment game and applications to pricing in cloud computing. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '14, pages 605–612, Richland, SC, 2014. International Foundation for Autonomous Agents and Multiagent Systems.

[7] Sofia Ceppi and Ian A. Kash. Personalized payments for storage-as-a-service. In *NetEcon15*, 2015.

[8] Ali Ghodsi, Matei Zaharia, Benjamin Hindman, Andy Konwinski, Scott Shenker, and Ion Stoica. Dominant resource fairness: Fair allocation of multiple resource types. In *NSDI*, volume 11, pages 24–24, 2011.

[9] Richard Gibbens, Robin Mason, and Richard Steinberg. Internet service classes under competition. *IEEE Journal on Selected Areas in Communications*, 18(12):2490–2498, 2000.

[10] Robert L Grossman. The case for cloud computing. *IT professional*, 11(2):23–27, 2009.

[11] Rolf Harms and Michael Yamartino. The economics of the cloud. Technical report, Microsoft, 2010.

[12] Benjamin Hindman, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D Joseph, Randy H Katz, Scott Shenker, and Ion Stoica. Mesos: A platform for fine-grained resource sharing in the data center. In *NSDI*, volume 11, pages 22–22, 2011.

[13] Navendu Jain, Ishai Menache, Joseph Seffi Naor, and Jonathan Yaniv. A truthful mechanism for value-based scheduling in cloud computing. *Theory of Computing Systems*, 54(3):388–406, 2014.

[14] Virajith Jalaparti, Hitesh Ballani, Paolo Costa, Thomas Karagiannis, and Ant Rowstron. Bridging the tenant-provider gap in cloud services. In *Proceedings of the Third ACM Symposium on Cloud Computing*, page 10. ACM, 2012.

[15] Ian Kash, Ariel D Procaccia, and Nisarg Shah. No agent left behind: Dynamic fair division of multiple resources. *Journal of Artificial Intelligence Research*, pages 579–603, 2014.

[16] Jeffrey K MacKie-Mason and Hal R Varian. Pricing congestible network resources. *Selected Areas in Communications, IEEE Journal on*, 13(7):1141–1149, 1995.

[17] Andrew Odlyzko. Paris metro pricing for the internet. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 140–147. ACM, 1999.

[18] Andrew Odlyzko. Internet pricing and the history of communications. *Computer networks*, 36(5):493–517, 2001.

[19] Jörg Schad, Jens Dittrich, and Jorge-Arnulfo Quiané-Ruiz. Runtime measurements in the cloud: observing, analyzing, and reducing variance. *Proceedings of the VLDB Endowment*, 3(1-2):460–471, 2010.

[20] Arun Sundararajan. Nonlinear pricing of information goods. *Management Science*, 50(12):1660–1673, 2004.

6 Biographies

Ian Kash is a researcher at Microsoft Research. His research focuses on questions at the interface between computer science and economics. Ian received the BS degree in computer science from Carnegie Mellon University in 2004, and the MS and PhD degrees in computer science from Cornell University in 2007 and 2010 respectively. Contact him at iankash@microsoft.com

Peter Key is a Principal Researcher at Microsoft Research. His research interests focus on Networks, Economics, and Algorithms. He is a Fellow of the ACM, IEEE, IET and IMA. Peter has a PhD in Statistics from London University and MA in Mathematics from Oxford. Contact him at peter.key@microsoft.com