Serverless Boom or Bust?
An Analysis of Economic Incentives

Xiayue Charles Lin
UC Berkeley

Joseph E. Gonzalez
UC Berkeley

Joseph M. Hellerstein
UC Berkeley

Abstract

Serverless computing is a new paradigm that promises to free cloud users from the burden of having to provision and manage resources. However, the degree to which serverless computing will replace provisioned servers remains an open question.

To address this, we develop an economic model that aims to quantify the value of serverless to providers and customers. A simple model of incentives for rational providers and customers allows us to see, in broad strokes, when and why serverless technologies are worth pursuing. By characterizing the conditions under which mutually beneficial economic incentives exist, our model suggests that many classes of customers can already benefit from switching to a serverless model and taking advantage of autoscaling at today's price points. Our model also helps characterize technical research directions that would be likely to have impact in the market.

1 Introduction

Over the past decade, the advent of cloud computing has caused many users to move their hardware infrastructure away from on-premises deployments and into large-scale shared resources [6]. Today, there are signs pointing towards another architectural shift, towards a "serverless" paradigm of *autoscaling* and *pay-as-you-go*, which gives users access to cloud computing without the burden of provisioning and managing resources. Underlying this architectural shift is a more fundamental movement in the economic model of computing towards paying for consumption instead of capacity.

There are many research and development initiatives underway that promise to improve the state of serverless computing in practice, such as those surveyed in recent articles [6, 13]. Here, we do not study how to improve serverless computing, but rather the circumstances under which consumption-based pricing is likely to succeed in the market. How can we quantitatively reason about the value of serverless computing to different customers – should we expect it to be a niche product, or can we reasonably expect widespread adoption? While existing work has opined that serverless can add significant value to customers and hypothesized about serverless dominance [13], there has not been any work quantifying this added value for customers who are not yet on serverless today. Similarly, there have been calls for research into pricing models and incentives [12] as well as better ways to build serverless systems [6], but this work does not address the impact that these would have have on the market for cloud computing.

If we can develop an understanding of the factors that shape the serverless trajectory, then we can reason about where the serverless community can best focus their efforts today. Thus, in this paper, we attempt to identify the economic incentives that result in both a customer and a cloud provider mutually preferring serverless over non-serverless ("serverful") offerings. The assumption behind this approach is that the existence of such economic incentives between two parties is, regardless of the short-term situation, a strong predictor for eventual long-term behaviour. To identify these economic incentives, in Section 2 we model the customer and the cloud provider as decision makers. In Section 3, we examine today's serverless market and categorize customers into three groups on their parameters, each with different paths to increased serverless adoption. Section 4 uses this framework to examine the value of *autoscaling*. Experiments using recently-released workload data from Microsoft Azure Functions [16] suggest that many classes of workloads may already benefit from using serverless autoscaling.

2 Modelling Providers and Customers

In this paper, we model customers and providers around one central decision that both providers and customers must make: the tradeoff between pricing for capacity ("serverful") versus consumption ("serverless"). Discussion of the limitations of this approach can be found in Section 5.

2.1 A Tradeoff Model for the Provider

The tradeoff that a provider faces is as follows. Consider a provider who has just rented some resource R (e.g. a virtual machine) to a customer, priced at rate p_r. By doing so, the provider incurs an opportunity cost: for that exact period of time, they could have spun up their own VM instead of renting it to the customer, and used that VM to serve serverless function requests instead, priced at some rate p_s.

The provider's tradeoff can thus be stated as:

> "Given an available resource R, should the provider use it to sell VM rentals or serverless function executions?"

If we assume the provider is rational, they will make the decision that is the most profitable. Over some period of time t, selling a VM rental would grant the provider revenue of $p_r \cdot t$, whereas selling serverless functions would grant revenue of $p_s \cdot t \cdot c$, where c is the expected utilization ratio of a VM executing serverless functions. This leads us to the following:

Definition 2.1.1 (The Serverless Provider Condition)**.** Given an available resource R, and that resource's p_r, p_s, c as defined above, a rational provider should use R to sell a serverless product if

$$c > \frac{p_r}{p_s}$$

This expression is agnostic to the characteristics of the serverful product offered; each specific configuration of the VM as well as its quality of service can be considered its own product offering.

2.2 A Task-Based Model for Cloud Customers

Consider a customer who has rented a VM R, for some period of time t, paying $p_r \cdot t$. Suppose that the customer accomplished some bag of tasks $F = \{f_1, \cdots, f_n\}$ using their rental. The customer could have instead executed these tasks as individual serverless function invocations. This would have costed the customer $p_s \cdot t_s$, where $t_s := \sum_i t(f_i)$ is the total time spent in serverless function executions.

This may suggest that customers are incentivized to use serverless simply when it is cheaper ($p_s \cdot t_s < p_r \cdot t_r$). However, this fails to capture that customers may get differing amounts of *net value* from executing on a serverless platform. For instance, developers may choose to not use serverless products for reasons such as performance and predictability requirements, or architectural transition costs.

We denote the customer's net value of serverless executions as v_s, and of serverful executions as v_r. A rational customer will use serverless when the net value is worth the cost of compute resources. This leads to the following:

Definition 2.2.1 (The Serverless Customer Condition)**.** Given that a customer has rented t_r time of some resource to accomplish some bag of tasks F, and that these tasks would take time t_s to execute through serverless functions, the rational customer should prefer consumption based pricing if

$$\frac{p_s}{p_r} < \frac{t_r}{t_s} \cdot \frac{v_s}{v_r}$$

The first term, p_s/p_r, is the premium that serverless products can charge for consumption-based pricing. It is upper-bounded by the product of two terms that characterize an individual customer; t_r/t_s represents how inefficiently a customer utilizes provisioned capacity, and v_s/v_r represents the change in net value from switching to a serverless product.

As the two right-hand terms characterize an individual customer, we will often need to refer to the terms together. We will thus denote $\alpha := \frac{t_r}{t_s} \cdot \frac{v_s}{v_r}$ for simplicity, and call it the *customer's preference for serverless*.

2.2.2 How Value Varies Across Cloud Customers

Serverless execution might be less appealing to cloud customers for any number of arbitrary reasons. To make this analysis tractable, we make the following observation: At any given point in time, a cloud customer wishing to have some task done always has the *option* to execute it on an existing serverless platform. Every time a cloud customer *declines* this option, they do so only because this option fundamentally produces less net value for the customer.

These decision factors differ across customers, and thus v_s/v_r ratios will also vary. We discuss the relationship between net value and various decision factors:

Transition cost. The simplest reason a customer might not execute a task on serverless is that they have not yet written the code to do. If the customer already has written the task to execute on their serverful architecture, then v_s captures some amortized cost for a business decision maker to transition between architectures.

Performance. Given that it is possible to execute a task on serverless, the next concern is the operational penalty of any degradation in performance – due to data shipping, queueing latency, lack of specialized hardware, or other factors that customers would otherwise be able to optimize in a serverful architecture. Tasks that are less sensitive to response time imply $v_s \approx v_r$. Other tasks may be very sensitive to response time, and actual costs for this sensitivity have previously been characterized [3, 18].

Operational Risk. Finally, it may be the case that certain customers are unable to shift their workload onto serverless due to operational reasons. Other customers may also be concerned about the operational risk posed by vendor lock-in. These customers have correspondingly lower values of v_s.

While accurately determining the value derived from cloud compute products is largely up to individual customers themselves, we characterize these common factors that are likely to affect many customers in predictable ways.

3 Serverless Markets of Today and Tomorrow

Having defined the conditions under which both a provider and a customer will prefer serverless, we can combine the conditions to reason about the serverless market and how various actions would shape it in the future.

Figure 1 is a representation of the serverless market today. The vertical axis represents the "true" value of a customer's α, and the horizontal axis has each individual customer in the market sorted by their true α values descending.

The exact distribution of α is beyond our ability to measure, but we can nonetheless partition the entire market into three groups based on ranges of true α values, even if unknown.

Group A ($\alpha > c^{-1}$) are the customers who satisfy the Serverless Provider Condition (Definition 2.1.1). These are the customers that providers are able to profitably sell serverless offerings to today. (The customers actually purchasing serverless today are a subset of this ($\alpha > p_s/p_r$), which we will call Group A'. To get a sense for what the p_s/p_r value is today, consider AWS's 1-core 1-GB-memory offerings at the time of writing (t2.micro @ \$0.0116 per hour, AWS Lambda @ \$0.06 per hour), which leads to a p_s/p_r of around 5.17. Later in Section 4, we will see how this particular value suggests that serverless has significant appeal today.)

Group B ($c^{-1} > \alpha > 1$) are customers who have $\alpha > 1$, but do not satisfy the Serverless Provider Condition. These customers would prefer consumption-based over capacity-based pricing if possible, but cloud providers are not yet able to profitably deliver serverless products to.

Group C ($\alpha < 1$) are the customers who innately prefer provisioning their own capacity over serverless offerings, even if they were offered at the same price.

Group C, where perhaps the majority of cloud customers might reside, will benefit from increases in serverless value (v_s) such that their α increases above 1. Section 2.2.2 outlines various improvements that can increase v_s. Some of these are potentially low-hanging fruit; for instance, every customer that would want a GPU is in Group C with $\alpha = 0$, simply because serverless with specialized hardware does not yet widely exist. Similarly, prohibitively expensive data movement [9] and unpredictable queuing latencies [20] are also known restrictions for existing serverless architectures. Lines of research [17, 21] that would address these factors are thus particularly promising for the market, as they would enable serverless for large classes of these "locked-out" customers.

As serverless offerings improve, customers begin moving from Group C into Group B. The onus then shifts to the serverless providers to improve the utilization of their serverless offerings, which brings down c^{-1} and widens the range of customers that providers can sell serverless to. Improving utilization involves addressing cold starts and idle time, and there is plenty of active research in this area [15, 16]. One unexplored direction that we find interesting is time-varying pricing in the vein of public utilities; aggregate load for serverless is

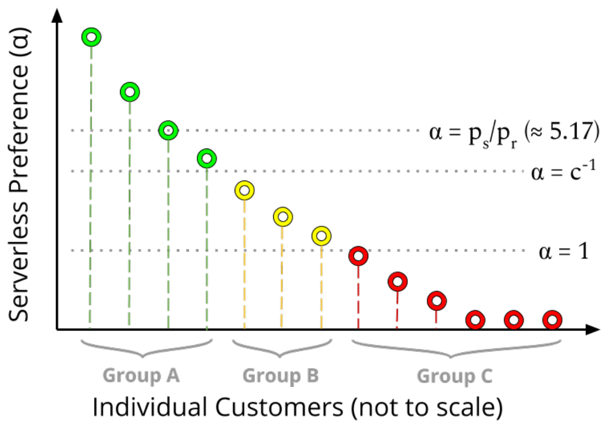

Figure 1: *Customers in the serverless market, sorted by their preference for serverless $\alpha := \frac{t_r}{t_s} \cdot \frac{v_s}{v_r}$. This conceptually partitions the market into three groups based on true α values that benefit differently from advances in serverless. The properties of these groups, and how they might enter the serverless market, are described in Section 3.*

shown to follow diurnal and weekly patterns [16]. Using time-varying pricing for flattening such patterns is well-studied in the economic literature [10].

Finally, Group A' customers are fully incentivized to use serverless products, and will do so, unless they are unaware of the value added or are otherwise imperfectly rational. In the next section, we will look at a scenario which suggests that there may be many such customers who are immediately able to benefit from adopting serverless today.

4 The Value of Autoscaling

One of the primary appeals of serverless is its ability to provide autoscaling. While this is intuitively an attractive feature, it is unclear how to precisely reason about its benefit. As a result, there is some tendency for cloud customers to view autoscaling products as primarily suited for bursty workloads [6], even if the fundamental benefits of consumption-based pricing might span far beyond that.

Consider a customer who continuously makes resource provisioning decisions in the face of a changing workload. We model this situation as a sequence of individual decisions to *start* up a new resource, or to *stop* an existing resource and deprovision it. A customer can then be defined by the sequence of individual resources it provisions, with each resource being defined by its start-time, end-time, and task utilization over that period.

This decomposes a customer's aggregate resources into individual resources, each of which has its own separate α value. To assess the value that autoscaling adds in a consumption-based model, we only need to treat each resource used as a separate resource allocation, and see which individual resources satisfy the Serverless Customer Condition (2.2.1).

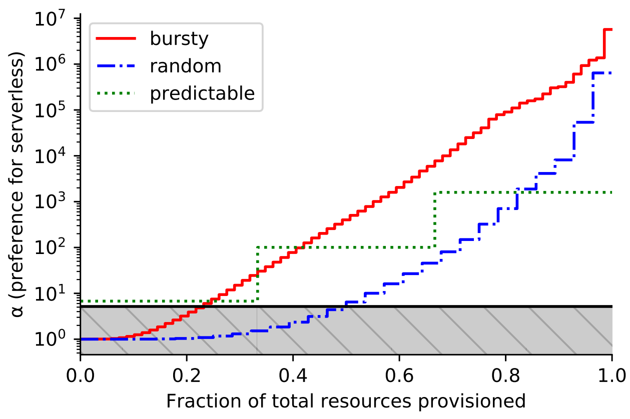

Figure 2: *Under static peak provisioning, the preference for serverless α_i for each resource provisioned. The shaded area represents $\alpha < p_s/p_r$ today; above this line, customers should prefer serverless. Customers running bursty workloads should begin preferring serverless after provisioning one-quarter of peak; for Poisson-random workloads, serverless should be preferred after provisioning for around half of peak.*

That is, a customer can immediately benefit from autoscaling if *any* of the resources it provisioned has $\alpha_i > p_s/p_r$, simply by serverlessly executing the tasks otherwise assigned to those specific resources.

4.1 Experiments

We will now simulate different classes of workloads against different resource provisioning strategies and show that serverless autoscaling can provide value to many different classes of workloads. Surprisingly, we find that serverless autoscaling can provide value even for workloads that are not particularly bursty.

For workload generation, we use recently released statistics from all workloads processed by Microsoft Azure Functions [16]. These statistics include a distribution of function execution duration, as well as distributions for the mean (μ) and coefficient of variation (c_v) of the interarrival time (IAT).

These statistics show a heavy tail of arbitrarily bursty functions, but also entirely static workloads, and functions arriving by a Poisson distribution, which is representative of many datacenter workloads [14].

For each of these classes, we choose representative parameters within the provided distributions and generate synthetic workloads consisting of task arrival times and task durations. For task arrival times, we fit Gamma distributions to $\mu = 60s$, $c_v = 0.1$ for predictable arrivals, $\mu = 1s$, $c_v = 1$ for random-Poisson arrivals, and $\mu = 1s, c_v = 5$ for random-bursty arrivals. These chosen values are in the 25-90th percentile of workloads, except for $\mu = 1s$, which is more representative of average executions. For all workloads, we use the full distribution of function durations provided in [16].

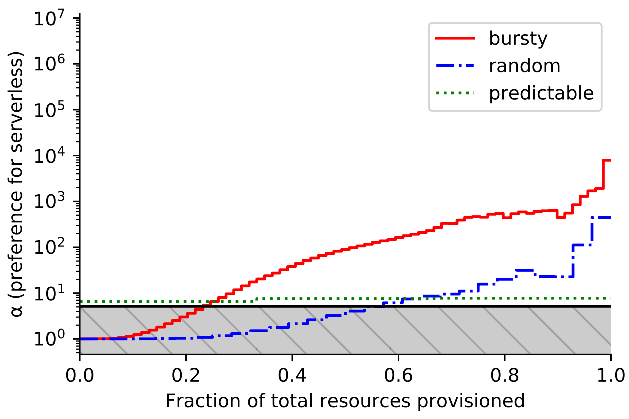

Figure 3: *Preference of serverless, but under oracle provisioning in 1-minute windows. Oracle provisioning does not significantly change the breakeven points at which customers should begin preferring serverless (cf. Figure 2).*

4.2 Peak vs. Oracle Provisioning

For each workload class, we evaluate the marginal preference of serverless for abstract cloud compute resources provisioned during a 24-hour simulation. We assume the customer is able to perfectly schedule and migrate tasks, and thus needs only k abstract resources to run k concurrent tasks.

The marginal preference of serverless for a customer who statically provisions for peak is shown in Figure 2. We also plot $\alpha < p_s/p_r$ value as a shaded gray box. For a customer running a random workload, they are economically better off executing remaining work on serverless after they have provisioned for about half of peak – in the figure, this is where the line crosses above the gray area. For a bursty workload, this point occurs even sooner, after the customer has provisioned for only a quarter of peak. Surprisingly, even the predictable workload shows preference for serverless, likely simply due to sparse utilization from periodic arrivals.

Note that for these measurements, we have assumed $v_s \approx v_r$. While this is likely true for the workloads simulated (as they describe workloads already ported to or developed on serverless), the large values of α suggests that similar results hold for any task as long as $v_s \neq 0$. This reinforces the importance of enabling serverless for the "locked-out" customers described in Section 3, and suggests that simply making a task possible, even if not very performantly, can add significant value to customers and providers.

Statically provisioning for peak is in essence the "worst" a serverful customer can do, so we will also evaluate a customer with oracle abilities, provisioning only the resources it needs for each 5-minute window. Figure 3 shows the results for this customer. While the preference for serverless is significantly flattened towards the peak, the breakeven points for serverless preference are barely changed.

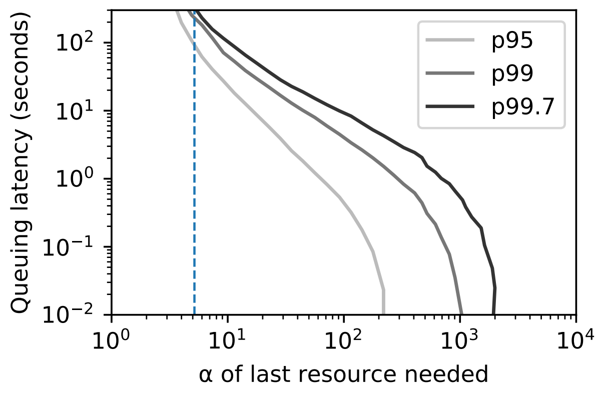

Figure 4: *A customer that underprovisions can improve queuing latency by provisioning more resources. This graph shows the* α *of the last resource needed to bring (p95, p99, p99.7) queuing latencies below any given target. The blue line denotes* $\alpha = p_s/p_r \approx 5.17$*, the boundary for preferring serverless today.*

4.3 Incentives for Quality of Service

Predictable performance is another desirable feature for a serverless platform, and a well-characterized subset of this problem is unpredictability due to cold start times [20]. To address this in AWS Lambda, Amazon has recently introduced *provisioned concurrency* [5], which allows users to pay to keep lambdas warm. Reintroducing provisioning decisions, however, is undesirable for a consumption-based pricing model.

A more elegant solution would simply be to charge for a guaranteed warm start. To determine the value of such a product to the customer, we perform a similar experiment, but this time, we allow the customer to underprovision and queue up tasks. Figure 4 is an experiment with a bursty workload under perfect scheduling and oracle provisioning in 10-minute windows. In this scenario, a customer wishing to bring p95 queuing latency down to the order of 10ms would have to provision resources with $\alpha > 100$. For a similar p99 target, they would have to provision resources with α nearing 1000. These numbers suggest that introducing serverless offerings, even with simple QoS guarantees, can provide much value to providers and customers who desire it.

5 Limitations

In this section, we will address some of the key assumptions and limitations behind our approach.

The biggest limitation is that the provider's infrastructure is viewed as a black box. We assume that variable costs of operation do not significantly differ when providing a serverless product. In practice, these costs include electrical power, which may be higher if the serverless product results in higher core utilization. There is also the variable cost of running components such as schedulers and warm pools.

Our model is also agnostic of heterogenity of resources and the resulting products offered. We assume that different configurations of VMs can be considered separate products for this model. Quality of service parameters such as response latencies or scaling limits are also configurations which we make no assumptions about. Real products may offer a baseline level of service (for instance, AWS Lambda at the time of writing has a scaling limit of 500 instances per minute, as well as well-documented response latencies); we do not consider this baseline QoS in our model, although we do discuss pricing for specific QoS levels in Section 4.

6 Related Work

It is interesting to compare the economics of serverless with the economics of the shift from on-premises computing to cloud computing. While cloud computing did offer the possibility of enabling new applications [8], it was also easily translated into a fixed-cost versus variable-cost tradeoff, which made it a favourable business decision [4]. In contrast, serverless computing may require the development of new systems and "enabling technologies" to broaden its appeal, and in this paper, we used an economic model to reason about the impact of these different improvements on the market.

Other adjacent work in economics of cloud computing has focused on pricing from a tactical perspective. Optimal bidding for spot market instances was an active area of work [11, 22]. For serverless economics specifically, research has included case studies for operational decisions [1] and a proof-of-concept for cheaper web serving using serverless [19]. We are more interested in reasoning about the market and its direction as a whole, but these lines of research are invaluable for helping individual customers understand how to switch to serverless. Similarly, efforts to model customer satisfaction have primarily been in the context of technical performance; [2] uses statistical learning to adaptively find good cloud configurations for big data analytics, and [7] proposes that cloud providers can assist customers who quantitatively declare their satisfaction.

7 Conclusion

In this paper, we have presented a framework for quantifying the value of serverless to different cloud customers. Using this framework, we analyze the current serverless market and show that all cloud customers can be categorized into one of three groups, each of which needs *different* advancements in serverless in order to incentivize adoption. Finally, we apply this framework to quantitatively assess the value of autoscaling, a key feature of serverless. Our results suggest that serverless is significantly underadopted and that many cloud customers today may benefit from a serverless model.

8 Discussion Topics

Attitudes of cloud provider firms. How does the discussion in this paper relate to the attitudes of cloud provider firms with respect to their actual practices and business strategies? In this paper, we treated serverless FaaS offerings in the context of a standalone product being offered by a profit-seeking firm, but it may be the case that serverless today is primarily a way to make any amount of spare change from otherwise idle VMs. If so, how does it affect the analysis in this paper, and how much profit margin would be needed for providers to seriously consider more dedicated development of serverless offerings?

Long-term serverless pricing. Additionally, economic theory states that firms choosing between producing multiple goods will optimally choose a level that satisfies *allocative efficiency*, where the marginal cost of producing the good is equal to its price. However, one would expect the marginal cost of selling cloud server time to be similar regardless of the product that the server was running, and thus the only fundamental reason for differing prices is lower paid utilization from serverless offerings. Are serverless prices understood to be unreasonably high? Or is our model for the provider failing to capture additional factors that we are unaware of?

Facilitating high-value serverless. While serverless execution can be of less value than serverful execution ($v_s \leq v_r$), particularly in the sense of FaaS, this ignores the benefit of having resource provisioning and management handled by the provider. When these costs are significant, we might expect significant adoption for these classes of customers; for instance, autoscaling cloud storage services such as Amazon S3 have seen significant adoption.

We suspect there are many similar but untapped markets. Data science users, for instance, are particularly uninterested in managing or provisioning resources, and as a result, services such as Google Colaboratory and JupyterHub have already become quite popular, even though they are not truly autoscaling. Building true serverless autoscaling for these and other products, however, requires dedicated development – and, without a serverless option, customers remain "locked-out" of the serverless market, causing both customers and providers to suffer deadweight loss.

What other products and customers might benefit from serverless offerings? Given our hypothesized importance of building such products, are providers interested in building these offerings or leaving it to the community instead?

References

[1] Gojko Adzic and Robert Chatley. Serverless computing: Economic and architectural impact. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2017, page 884–889, New York, NY, USA, 2017. Association for Computing Machinery.

[2] Omid Alipourfard, Hongqiang Harry Liu, Jianshu Chen, Shivaram Venkataraman, Minlan Yu, and Ming Zhang. Cherrypick: Adaptively unearthing the best cloud configurations for big data analytics. In *Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation*, NSDI'17, page 469–482, USA, 2017. USENIX Association.

[3] Daniel An. Find out how you stack up to new industry benchmarks for mobile page speed, 2018. https://www.thinkwithgoogle.com/marketing-resources/data-measurement/mobile-page-speed-new-industry-benchmarks.

[4] Ergin Bayrak, John Conley, and Simon Wilkie. The economics of cloud computing. Vanderbilt University Department of Economics Working Papers 1118, Vanderbilt University Department of Economics, 2011.

[5] James Beswick. New for aws lambda – predictable start-up times with provisioned concurrency, 2019. https://aws.amazon.com/blogs/compute/new-for-aws-lambda-predictable-start-up-times-with-provisioned-concurrency/.

[6] Paul Castro, Vatche Ishakian, Vinod Muthusamy, and Aleksander Slominski. The rise of serverless computing. *Communications of the ACM*, 62(12):44–54, 2019.

[7] Vojislav Dukic and Ankit Singla. Happiness index: Right-sizing the cloud's tenant-provider interface. In *11th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 19)*, Renton, WA, July 2019. USENIX Association.

[8] Armando Fox, Rean Griffith, Anthony Joseph, Randy Katz, Andrew Konwinski, Gunho Lee, D Patterson, Ariel Rabkin, Ion Stoica, et al. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, 28(13):2009, 2009.

[9] Joseph M. Hellerstein, Jose Faleiro, Joseph E. Gonzalez, Johann Schleier-Smith, Vikram Sreekanti, Alexey Tumanov, and Chenggang Wu. Serverless Computing: One Step Forward, Two Steps Back. In *CIDR 2019, 9th Biennial Conference on Innovative Data Systems Research*, Asilomar, CA, USA, 2019.

[10] Stephen P Holland and Erin T Mansur. The short-run effects of time-varying prices in competitive electricity markets. *The Energy Journal*, 27(4), 2006.

[11] David Irwin, Prashant Shenoy, Pradeep Ambati, Prateek Sharma, Supreeth Shastri, and Ahmed Ali-Eldin. The price is (not) right: Reflections on pricing for transient cloud servers. In *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9. IEEE, 2019.

[12] David Irwin and Bhuvan Urgaonkar. Research challenges at the intersection of cloud computing and economics. Technical report, USA, 2018.

[13] Eric Jonas, Johann Schleier-Smith, Vikram Sreekanti, Chia-Che Tsai, Anurag Khandelwal, Qifan Pu, Vaishaal Shankar, Joao Carreira, Karl Krauth, Neeraja Yadwadkar, Joseph E. Gonzalez, Raluca Ada Popa, Ion Stoica, and David A. Patterson. Cloud programming simplified: A berkeley view on serverless computing, 2019. https://arxiv.org/abs/1902.03383.

[14] Srikanth Kandula, Sudipta Sengupta, Albert Greenberg, Parveen Patel, and Ronnie Chaiken. The nature of data center traffic: measurements & analysis. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, pages 202–208, 2009.

[15] Edward Oakes, Leon Yang, Dennis Zhou, Kevin Houck, Tyler Harter, Andrea Arpaci-Dusseau, and Remzi Arpaci-Dusseau. SOCK: Rapid task provisioning with serverless-optimized containers. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*, pages 57–70, Boston, MA, July 2018. USENIX Association.

[16] Mohammad Shahrad, Rodrigo Fonseca, Íñigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini. Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider, 2020. https://arxiv.org/abs/2003.03423.

[17] Vikram Sreekanti, Chenggang Wu, Xiayue Charles Lin, Johann Schleier-Smith, Jose M. Faleiro, Joseph E. Gonzalez, Joseph M. Hellerstein, and Alexey Tumanov. Cloudburst: Stateful functions-as-a-service, 2020. https://arxiv.org/abs/2001.04592.

[18] Akamai Technologies. Akamai online retail performance report: Milliseconds are critical, 2017. https://www.akamai.com/uk/en/about/news/press/2017-press/akamai-releases-spring-2017-state-of-online-retail-performance-report.jsp.

[19] Mario Villamizar, Oscar Garcés, Lina Ochoa, Harold Castro, Lorena Salamanca, Mauricio Verano, Rubby Casallas, Santiago Gil, Carlos Valencia, Angee Zambrano, and et al. Cost comparison of running web applications in the cloud using monolithic, microservice, and aws lambda architectures. *Serv. Oriented Comput. Appl.*, 11(2):233–247, June 2017.

[20] Liang Wang, Mengyuan Li, Yinqian Zhang, Thomas Ristenpart, and Michael Swift. Peeking behind the curtains of serverless platforms. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*, pages 133–146, Boston, MA, July 2018. USENIX Association.

[21] Ryan Yang, Nathan Pemberton, and Jichan Chung. Pyplover: A system for gpu-enabled serverless instances. Technical report, University of California, Berkeley.

[22] Liang Zheng, Carlee Joe-Wong, Chee Wei Tan, Mung Chiang, and Xinyu Wang. How to bid the cloud. *ACM SIGCOMM Computer Communication Review*, 45(4):71–84, 2015.