CloudSurf: a platform for monitoring public-cloud networks

Valerio Persico*, Antonio Montieri◇, Antonio Pescapé*,◇
*University of Napoli Federico II (Italy) and ◇NM2 srl (Italy)
valerio.persico@unina.it, montieri@nm-2.com, pescape@unina.it

***Abstract*—Despite customers increasingly depend on cloud systems, they are not advertised with any information about either the design or the performance figures of the network infrastructures that support them. In this paper, we introduce CloudSurf, an open-source platform we have designed, implemented, and recently publicly released [5]. CloudSurf allows to monitor public-cloud networking infrastructures from the customer viewpoint through *non-cooperative* approaches, i.e. without relying on information restricted to the cloud provider or to entities playing a privileged role with respect to the provision of cloud services. After having identified a set of desirable features, we discuss the design of the platform, also showing its effectiveness through a set of use cases. Thanks to CloudSurf it is possible to go beyond the coarse information about public-cloud network performance today available. The information collected by CloudSurf can guide customers in performing cloud-services configuration, thus allowing them to improve cloud-network performance, to understand its variability, and to reduce costs.**

I. Introduction

A huge number of applications is today delivered through the cloud, and organizations more and more depend on this general purpose technology [12]. Many companies are riding the wave and provide a wide range of public-cloud services that rapidly evolve over time, being backed by infrastructures with increasing complexity. However, the dependence of the industry on cloud systems has grown much faster than the understanding of the performance limits and dynamics of these environments.

A fitting example in this regard is represented by the network infrastructures that support these complex systems: while all providers grant high-performance network connectivity to their customers they rarely provide more than qualitative information about its performance or its design, mainly due to security and commercial reasons [14], [18]. Indeed, huge investments in networking have been made [13], to support cloud traffic which is expected to grow in volume at 33-percent CAGR from 2014 to 2019, thus accounting for more than 83-percent of total datacenter traffic by 2019 [8]. Investments aim at improving the performance of the cloud networking infrastructure in all its composing areas, i.e., (i) the *intra-datacenter* network that connects cloud resources placed in the same datacenter; (ii) the *inter-datacenter* wide-area network that connects datacenters located in geographically distributed regions; and (iii) the *cloud-to-user* network that is the collection of network paths between cloud datacenters and external hosts connected to the public Internet.

Although each of these network areas has different characteristics and constraints in accordance to the applications that rely on it [10], [19], [6], no detail about either the strategies implemented to manage the network or its expected performance is usually made public by cloud providers. Therefore cloud customers have to face a number of practical limitations. For instance, they are not aware of how network resources are allocated to the cloud services available or have no information about how network performance may vary over time. Definely, they are not able to either perform informed choices among the different services or compare different providers.

Adopting *non-cooperative approaches* (i.e. implementing methodologies that do not require a privileged point of view over the cloud environment and thus can be enforced also without any help from the provider) is the natural solution to the described situation. In this context, cloud networking performance has recently attracted the interest of the scientific community and the literature provides a number of examples in this regard [17], [16], [15], [19]. However, the outcome of these pioneering works is sometimes limited in scope for what concerns the performance of the network, and the obtained results conflict in some cases. The different points in time and the lack of knowledge about the specific conditions in which these analyses were performed as well as the different methodologies and tools adopted, make these kind of analyses hardly repeatable and do not ease the comparison of the results. In addition, the monetary cost of the experimentations needed for obtaining valuable information and the need for advanced expertise in cloud-network monitoring activities (e.g., selection and configuration of the network monitoring tools, setup of the cloud environment, etc.) further exacerbate the described scenario.

In order to overcome the limits of the state of the art we propose the CloudSurf platform. It is designed to perform cloud network monitoring activities from the general customer's angle and aims at providing the customers—even those with no specific expertise in performance monitoring—with a powerful and versatile tool to easily investigate the performance of cloud networks *on demand* and according their specific needs. Moreover, CloudSurf also allows to easily share monitoring results among the community of users, thus leading to save experimentation expenses of users.

The paper is organized as follows: we first identify the desirable features for a platform for monitoring public-cloud networks from the customer point of view (Sec. II); we then describe the design of the platform (Sec. III); We also show the effectiveness of CloudSurf presenting some results associated to some of the use cases implemented by the platform (Sec. IV); finally, concluding remarks are drawn (Sec. V).

II. Desirable Features

In this section we discuss the set of desirable features we believe a platform such as CloudSurf should have. They have been identified by taking into account both the obstacles that cloud customers face in understanding the performance figures offered by public-cloud networks and the limitations of the approaches already implemented for monitoring these networks.

Adoption of non-cooperative monitoring approaches. As the platform is thought to be fully oriented to the general customer, the approaches implemented have to require no cooperation of any privileged entity. As the monitoring activities do not rely on this advanced information, they have to primarily leverage *active monitoring approaches*.

Ease of use. Customers taking advantage of cloud services have different backgrounds and potentially have not an advanced expertise in network monitoring activities and related issues. Therefore, a desirable property of the platform is to be easy to use, such that it can be leveraged by the wide community of people interested in cloud network performance.

Comprehensiveness. As the providers offer a rapidly evolving wide range of services with different characteristics and properties, it is desirable for the platform to possibly deal with the most popular providers and all the options made available by each of them. In addition customers may be interested in different properties of the networks (e.g., minimum bandwidth guaranteed, maximum throughput achievable, latency). Accordingly the platform has to allow customers to perform the experimentation suited for gathering the aspects of interest.

Predictability of the experimentation cost. Non-cooperative approaches require to interact with the provider as general customers do. This implies that the user of the platform is subjected to the pay-as-you-go paradigm, in accordance to the terms of contract of each provider. Cloud network experimentations could be very costly (especially when repeated analyses are needed or when traffic has to be generated at high rates) as providers usually charge customers not only for the computation or memory capabilities of the Virtual Machines (VMs), but also for the traffic generated by them. It is desirable that the platform would be able to estimate the cost that the customers would be subjected to.

Easy sharing of the analysis results. According also to the cost of the experimentations, it would be desirable that the platform would provide an easy way to share the outcome of the analyses with the community of people interested in them.

III. Design and Implementation

Driven by the desirable features discussed above, in this section we describe the design of CloudSurf.

The CloudSurf platform includes all the basic components required to perform cloud-network monitoring activities acting as a general customer: (i) the cloud probes, (ii) the master, and (iii) the results repository. Fig. 1 shows an overview of the architecture and its main components. CloudSurf has been implemented in Python and has been released under the Affero GPL (AGPL) license [5]. It is designed to support cloud monitoring activities for the leading public cloud providers: Amazon Web Services [2] and Microsoft Azure [4]. Considering that these two providers own 40% of the cloud market together [3], CloudSurf is designed to support most of the customers leveraging cloud services.

In the following, details about the design and the functionalities of the platform are provided.

A. Cloud probes

The *cloud probes* are remote measurement servers deployed by the master on demand, i.e. when the user of the platform requires some experiments to be performed. The probes are thought to be passive entities, as they wait for instructions once deployed. As the CloudSurf platform implements non-cooperative approaches through active measurements, cloud probes act as the endpoints of the experiments, and may play the role of both the sender and the receiver, as required by the master. In more details, the probes integrate a number of active monitoring tools (e.g., *nuttcp* [11], *ping*, *paris traceroute* [7], or *D-ITG* [9]) that inject traffic into the network to measure its characteristics. The set of the tools integrated covers a number of different active experimentations, such as the estimation of the available link capacities, the measurement of the achievable network throughput, the evaluation of the network latency, and the tracing of network path. In addition the cloud probes implement a number of utility functions, allowing the master to cope with management issues (e.g., verifying if the probe is alive or getting the state of a given experiment).

The remote measurement service exposed by each probe is implemented through the XML-RPC protocol [1]. XML-RPC is a remote procedure call protocol which uses XML to encode its calls and HTTP as a transport mechanism. The parameter types allow nesting of parameters into maps and lists, thus larger structures can be transported. Therefore, XML-RPC can be used to transport objects or structures both as input and as output parameters. Identification of clients for authorization purposes can be achieved using popular HTTP security methods.

According to the proposed approach, the probes generate and receive two types of traffic: (i) the *control traffic* exchanged between a probe and the entity that controls it; (ii) the *measurement traffic* generated by the probes and exchanged among them. Experiments implemented by the probes can

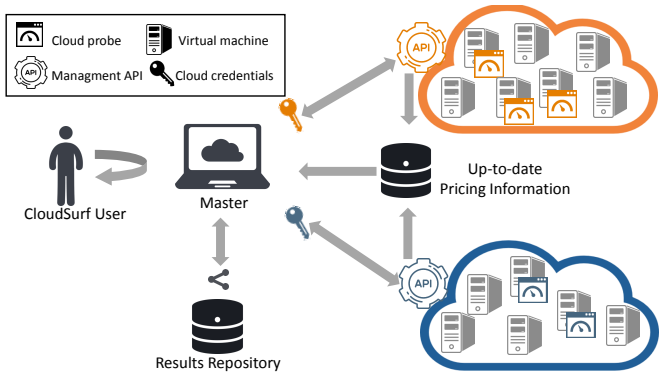

Fig. 1: CloudSurf Architecture.

be classified as *one-sided* or *two-sided*. In the former case, the experiment does not require control over the destination, i.e. the sender does not require an active process listening at the destination side. In the second case, the experiment does require that at the receiver side a process has been activated. Accordingly, in the case of two-sided experiments the entity that orchestrates the experiments has to coordinate the two probes involved, while in the case of one-sided experiments there is not such a need.

B. Master

The *master* is the entity in charge of orchestrating the overall monitoring process and is directly interfaced to the customer. The interaction with the user defines both the exact configuration in which the measurement experiments have to be carried out, and the type and duration of experiments to execute. CloudSurf users can directly control a set of factors of interest, whose combination identifies a number of scenarios in which they may operate. This approach eases as much as possible the understanding of the precise conditions in which the analysis is performed and allows to identify the major factors impacting the performance perceived by the users, fostering also the ability to replicate the analysis in exactly the same conditions [17], [16].

Setup of the experimental scenario. The factors under the control of the CloudSurf users to uniquely identify experimental scenarios can be divided in two major groups: (i) *deployment* and (ii) *experimental configuration* factors.

Deployment factors can be further classified as *common* or *provider-specific*, depending if they reflect common practices enforced by any public cloud provider or not, respectively. The former include: (i) the type (e.g., general purpose, compute optimized, storage optimized, etc.) and the size (e.g., small, medium, large, etc.) of the VMs to be leveraged, chosen between among the ones made available by the provider; (ii) the cloud region, in which a user can deploy his cloud resources, chosen from the ones belonging to the infrastructure of the cloud provider. Provider-specific deployment factors depend upon the different strategies implemented by providers in the offers they make available. For instance, the possibility to select among different availability zones made available by Amazon [2], or the affinity group and the virtual network configurations provided by Microsoft Azure [4].

Experimental configuration factors are related to the modes in which the traffic is generated. CloudSurf allows the users to select (i) the transport protocol (i.e. TCP or UDP) and (ii) the size of the generated packets, and the rate at which they are injected into the network. Finally, VMs can be terminated and recreated from scratch after each experiment, in order to evaluate the impact of VM placement transparently implemented by providers on the experimental results (VM relocation).

After the user has defined the setup of the parameters of the experimental campaign, the master performs the tasks described in the following.

Experiment-cost estimation. Before running the experiment, the master predicts the cost which the user is subjected to, that depends on the charges imposed by the specific provider, according to the pay-as-you-go model. Two quotas can be identified: (i) the expense associated to the requested cloud resources (e.g., VMs leased, object storage, etc.), (ii) the expense related to the network traffic generated. These two quotas are heavily impacted by the specific configuration of the cloud environment and by the type and duration of the experiment, chosen by the user. For instance, an experiment involving two large-sized VMs deployed in different datacenters has a cost higher than an experiment between two small-sized VMs deployed in the same datacenter. The general formulation implemented by CloudSurf for estimating this cost is the following:

$$Exp\_cost = \left\lceil \frac{D}{3600} \right\rceil * (C_{VM}^{sender} + C_{VM}^{receiver}) + R * C_{Traffic}$$

where D is the duration of the experiment (in seconds), C_{VM}^{sender} is the (hourly) cost of the sender VM, $C_{VM}^{receiver}$ is the (hourly) cost of the receiver VM, R is the rate of measurement traffic (in GB/s), and $C_{Traffic}$ is the charge that the provider imposes to the data transfer (€/GB). The provider-dependent parameters in the above formula (i.e. C_{VM}^{sender}, $C_{VM}^{receiver}$, and $C_{Traffic}$) are gathered taken advantage of a *cost-estimation module*, that periodically updates this pricing information extracting it from the pricing pages of the providers.

Cloud-environment setup and deployment of probes. Based on the outcome of the cost estimation, the user can decide to withdraw the experiment (e.g., if the predicted cost is too high) or to continue. If so, the master performs the cloud-environment setup and allocates the cloud resources in order to accomplish the monitoring activities required by the user. The monitoring probes are deployed by leveraging the IaaS paradigm by interacting with the public-cloud providers through their public interface. In more details, in the setup phase, the master is in charge of: (i) managing user's credentials, (ii) configuring firewalling rules, (iii) launching VMs, and (iv) deploying the monitoring probes onto them. At the end of the monitoring activities the master decommissions the VMs and restores the state of the environment to carry it to its initial conditions.

In order to setup the cloud environment, the master interacts with the different providers through the cloud management API exposed. CloudSurf leverages the Python implementation of the interfaces made available by the providers. To interact through this interface, the master needs a proper level of authorization from the user. Once the credentials have been arranged, the platform is able to autonomously configure the cloud environment.

Before deploying the measurement probes, the VMs have to be launched, properly configuring their firewall rules. The firewall must be configured in order to let both the measurement traffic and control traffic pass. In order to access the VMs via SSH indeed, for each experiment a couple of PEM-encoded RSA keys is created: the public one is deployed onto the VMs through the management API, while the private one is locally stored and adopted for interacting with the VMs for the successive tasks. It is worth noting that thanks to this step, afterwards the CloudSurf platform can interact with the VMs with no need to pass through the provider's API. This also means that the steps after the VM creation are the same whichever the involved provider is.

Once these tasks have been completed, the VMs can be launched. No other activity can be performed before the VMs are in the running state. As the startup time may vary with the provider or the resources leased, the CloudSurf platform takes advantage of the *status-check* functionality made available by the providers to be sure that the VMs launched are actually running. When the status check returns a positive outcome, the probes can be deployed on the newly instantiated VMs, i.e. the tools needed are installed and configured on the cloud VMs, and the measurement servers are actually started.

Experiment life-cycle management. When the measurement probes are available, i.e. the master can leverage the measurement services they expose, the experimentations can be actually launched, coordinating the probes through the XML-RPC protocol (see Fig. 2). Each measurement campaign is identified by a structure containing all the information needed to define an experiment. In detail, the IDs of the sender and the receiver VMs involved in the campaign, an experiment list parameter representing a list of experiment objects, and other management parameters useful to schedule the experiment launches. Each experiment object contains all the information to execute measurement activity (e.g., tool, experiment duration, bitrate and protocol of the traffic generated, etc.). Leveraging this information, the master drives the sender and possibly the receiver VM, according to the fact that the experiment is two-sided or one-sided, respectively.

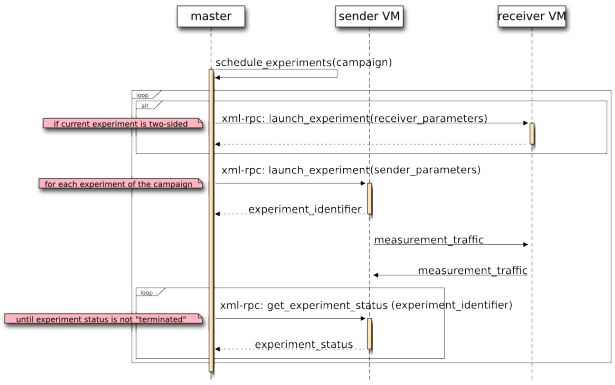

Fig. 2: CloudSurf experiment life-cycle management.

Results collection. At the end of the experimental campaign, the master gathers all the results temporarily stored on the probes. As the CloudSurf platform allows to take advantage of a number of different tools—each with its own output format—the gathered results are forcedly heterogeneous. In order to solve this heterogeneity issue, the master relies on a *parsing module*, in charge of translating the raw output into an homogeneous format. In more details, for each tool supported, the master has a parsing routine that translates the unstructured heterogeneous source into a JSON-encoded homogeneous format. For each experiment key details are saved, such as: the start time, the duration, the tool adopted. Finally for each parameter investigated with the experiment (e.g., throughput, latency, loss, etc.), both the instantaneous values (when available) and the synthetic information are saved to ease the analysis of the results.

C. *Results repository*

The *results repository* is a community archive where the JSON-encoded results containing the experimental information are uploaded by the CloudSurf master after having been compressed. Sharing their results allows the users to access to information related to a wider set of configurations and scenarios, reducing also the costs to perform these experimentations directly.

IV. Experimental results

In the following we report some of the results obtained through the platform to show the effectiveness of CloudSurf in monitoring the different cloud network areas previously identified.

A. *Monitoring* intra-datacenter *network performance*

Fig.3 shows the mean and the standard deviation of the achieveable TCP throughput measured over 24 hours performing repeated 5-minute-long experiments between general-purpose VMs of the same size (M, L, and XL) in the Amazon datacenter located in Ireland. Achievable throughput strongly depends on the VM size. In addition—as also shown by the small standard deviation—throughput variability over time is markedly limited. Interestingly, both M and L VMs are advertised to have *moderate* networking performance by the provider [2] although experimental results show how L-sized VMs stably outperform M VMs. A user leveraging XL VMs can achieve throughput values 230.9% higher than when using an M VM, on average.

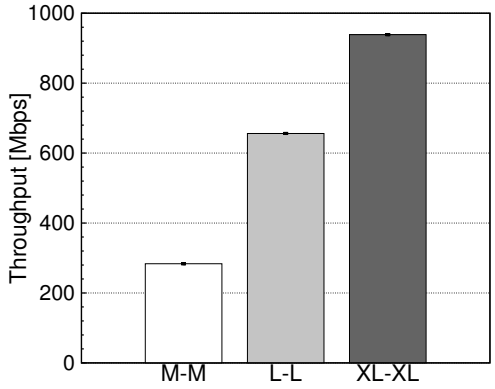

Fig. 3: Mean and standard deviation of the maximum TCP throughput achievable between Amazon VMs of different sizes deployed in the same datacenter placed in the Ireland region.

B. *Monitoring* inter-datacenter *network performance*

Fig.4 reports an overall picture of the TCP throughput performance obtained among four different Amazon and Azure geographically distributed datacenters (Ireland, North Virginia, Singapore, and Sao Paulo). Each sample represents the mean of a 5-minute-long experiment. Network throughput heavily varies with the specific pair of regions taken into account, being the path interconnecting Ireland and North Virginia the one with the best performance among those tested. In this case VM size proved to be non-influential: for instance M

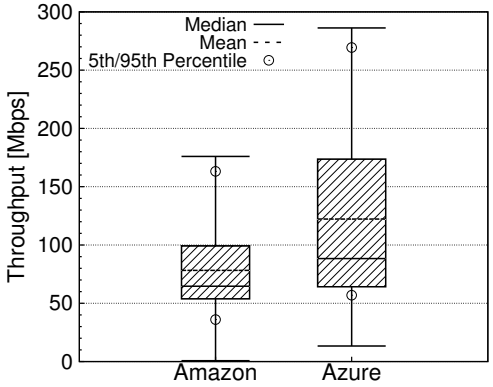

Fig. 4: TCP throughput distribution across different regions for Amazon and Azure. Each sample represents the mean of a 5-minute-long experiment. Azure performs better on average ($+56\%$).

and XL VMs reported the same performance for Amazon, although they are advertised to have *moderate* and *high* network performance, respectively. On average, Azure inter-datacenter network performs better ($+56\%$).

C. *Monitoring* cloud-to-user *network performance*

Fig. 5 shows the distributions of the performance in terms of goodput perceived when leveraging the *Amazon Simple Storage Service (S3)* for downloading contents (100 MiB) stored in 4 different cloud regions—North Virginia (US), Ireland (EU), Singapore (AP), and Sao Paulo (SA) from more than 70 vantage points spread worldwide. This analysis allows us to evaluate what are the datacenters that serve better a generic user without any assumption about his exact location. Considering the goodput average values from all the users, US, EU, SA, and AP cloud regions reported 3562.81, 2791.08, 1445.75, and 2018.76 KiB/s, respectively. Therefore two performance classes can be easily identified: US and EU versus SA and AP, where the former performs 45.5% better than the latter, on average. Counterintuitively, AP and SA are also associated to higher network-transfer costs with respect to EU and US.

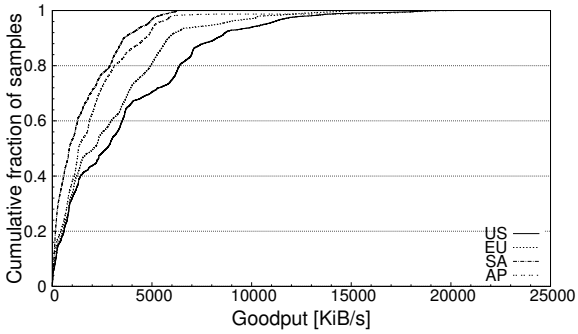

Fig. 5: Amazon S3 goodput performance for downloads of 100 MiB-sized objects performed by users spread worldwide. CDFs from each user to the four geo-distributed cloud regions are showed. US and EU perform better on average ($+45.5\%$) than AP and SA cloud regions.

V. Conclusion

Despite the huge investment made by providers in improving cloud network performance, they rarely advertise any specific information about the design or the performance of these network infrastructures. Consequently, cloud customers suffer from the lack of information about them.

In this paper we have proposed CloudSurf, an open-source monitoring platform designed to allow the general customers to monitor public-cloud networks. CloudSurf has been designed according to a set of desirable features here identified, and thanks to it a cloud customer can monitor the performance of the cloud networks in any scenario of interest with no specific expertise neither in managing cloud resources nor in utilizing network monitoring tools. A description of the design choices behind the development of CloudSurf has been provided, showing the details of its features and operating modes. Proposing a number of practical examples, we have shown the effectiveness of the platform in heavily improving the understanding of the public-cloud networks in different scenarios and use cases.

Acknowledgment

This work is partially funded by art. 11 DM 593/2000 for NM2 srl (Italy). The experimental work in this paper was also supported by a grant provided by Amazon AWS in Education.

References

[1] XML-RPC Specification. http://xmlrpc.scripting.com/spec.html, June 1999.

[2] Amazon Web Services website. http://aws.amazon.com/, June 2016.

[3] Aws remains dominant despite microsoft and google growth surges. https://www.srgresearch.com/articles/aws-remains-dominant-despite-microsoft-and-google-growth-surges, Feb. 2016.

[4] Microsoft Azure website. http://azure.microsoft.com/, June 2016.

[5] The CloudSurf Platform. http://traffic.comics.unina.it/cloudsurf, June 2016.

[6] V. K. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z. L. Zhang. Unreeling netflix: Understanding and improving multi-cdn movie delivery. In *INFOCOM, 2012 Proceedings IEEE*, pages 1620–1628, March 2012.

[7] B. Augustin, T. Friedman, and R. Teixeira. Measuring multipath routing in the Internet. *IEEE/ACM Transactions on Networking*, 19(3):830–840, June 2011.

[8] Cisco. Cisco global cloud index: Forecast and methodology, 2014-2019. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.html, 2016.

[9] D. Emma, A. Pescape, and G. Ventre. Analysis and experimentation of an open distributed platform for synthetic traffic generation. In *Distributed Computing Systems, 2004. FTDCS 2004. Proceedings. 10th IEEE International Workshop on Future Trends of*, pages 277–283, May 2004.

[10] Y. Feng, B. Li, and B. Li. Jetway: minimizing costs on inter-datacenter video traffic. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 259–268. ACM, 2012.

[11] B. Fink and R. Scott. nuttcp, v6.1.2. http://www.nuttcp.net/.

[12] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica. 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.

[13] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. The cost of a cloud: research problems in data center networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68–73, 2008.

[14] K. LaCurts, S. Deng, A. Goyal, and H. Balakrishnan. Choreo: Network-aware task placement for cloud applications. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 191–204, New York, NY, USA, 2013. ACM.

[15] A. Li, X. Yang, S. Kandula, and M. Zhang. Cloudcmp: Comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 1–14, New York, NY, USA.

[16] V. Persico, P. Marchetta, A. Botta, and A. Pescapé. Measuring network throughput in the cloud: the case of amazon ec2. *Elsevier, Computer Networks, Special Issue on Cloud Networking and Communications II*.

[17] V. Persico, P. Marchetta, A. Botta, and A. Pescapé. On network throughput variability in microsoft azure cloud. In *GLOBECOM, 2015 Proceedings IEEE*, 2015.

[18] C. Raiciu, M. Ionescu, and D. Niculescu. Opening up black box networks with cloudtalk. In *Proc. 4th USENIX Conference on Hot Topics in Cloud Computing*, page 6, 2012.

[19] J. Schad, J. Dittrich, and J.-A. Quiané-Ruiz. Runtime measurements in the cloud: Observing, analyzing, and reducing variance. *Proc. VLDB Endow.*, 3(1-2):460–471, Sept. 2010.