Characterization and Analysis of Cloud-to-User Latency: the case of Azure and AWS

Article *in* Computer Networks · November 2020

DOI: 10.1016/j.comnet.2020.107693

CITATION
1

READS
159

6 authors, including:

Fabio Palumbo
University of Naples Federico II
6 PUBLICATIONS **32** CITATIONS
SEE PROFILE

Giuseppe Aceto
University of Naples Federico II
57 PUBLICATIONS **1,871** CITATIONS
SEE PROFILE

Alessio Botta
University of Naples Federico II
83 PUBLICATIONS **3,862** CITATIONS
SEE PROFILE

Domenico Ciuonzo
University of Naples Federico II
110 PUBLICATIONS **2,901** CITATIONS
SEE PROFILE

Characterization and Analysis of Cloud-to-User Latency: the case of Azure and AWS

Fabio Palumbo[a], Giuseppe Aceto[a,b], Alessio Botta[a], Domenico Ciuonzo[a], Valerio Persico[a], Antonio Pescapé[a,b]

[a]*University of Napoli "Federico II", Italy*
[b]*Network Measurement and Monitoring (NM2) s.r.l., Italy*

Abstract

With the growing adoption of cloud infrastructures to deliver a variety of IT services, monitoring cloud network performance has become crucial. However, cloud providers only disclose qualitative information about network performance, at most. This hinders efficient cloud adoption, resulting in uncertainties about the behavior of hosted services, and sub-optimal deployment choices. In this work, we focus on cloud-to-user latency, i.e. the latency of network paths interconnecting datacenters to worldwide-spread cloud users accessing their services. Specifically, we performed a 14-day measurement campaign from 25 vantage points deployed via the Planetlab infrastructure (emulating spatially-spread users) and considering services running in distinct locations on the infrastructures of Amazon Web Services and Microsoft Azure. First, our experimentation allows us to provide an in-depth performance characterization (based on multiple probing methods and fine-grained sampling rate) of such networks as perceived by users spread worldwide, highlighting both spatial and temporal latency trends. Then, our analysis is exploited with design purposes to support both cloud customers and providers with the assessment of cloud-network performance (via badness detection & imputation tools) and the making of deployment decisions (via the evaluation of multi-cloud benefits). The dataset gathered from the campaign is publicly released to foster reproducibility.

Keywords: public-cloud networks; Amazon Web Services; Microsoft Azure; network measurements; network performance.

1. Introduction

The last years have seen increasing adoption of *public clouds* fueled by the remarkable economical and technical benefits they provide.[1] The heterogeneity of applications leveraging such services has resulted in a wide variety of Quality-of-Service (QoS) requirements, in turn surfacing the necessity of fine-grained characterization of cloud performance. Accordingly, a large body of literature has focused on performance analysis (e.g. by analytical models [1]) of cloud computing infrastructures, and on the capability of its computational resources to respond to user requests guaranteeing low response times or providing a certain level of availability.

In this context, both providers and customers showed a growing interest in measurement activities targeting cloud networks, a major—and hard-to-analyze—factor in cloud service performance. Although cloud networks are the workhorse to both operate and capitalize cloud services [2, 3], cloud providers rarely are able or willing to provide guarantees or disclose details on network performance [4]. Therefore, *non-cooperative approaches* [5, 6] have emerged in last years. Being methodologically independent, these can integrate and expand the knowledge base that a provider is able to gather from inside the datacenter, i.e. only leveraging a privileged view on a limited portion of the whole system. Indeed, non-cooperative approaches do not rely on any help from the provider to obtain visibility into the cloud-network performance "building blocks": (*i*) intra-datacenter, (*ii*) inter-datacenter, and (*iii*) cloud-to-user networks. Among these three components, *cloud-to-user* (C2U) network (i.e. the set of paths interconnecting users to the set of pooled resources composing the cloud) is usually beyond direct control of both cloud providers and customers. As a result, C2U network is *harder to be monitored* and accurately predicted when compared to the intra- and the inter-datacenter networks [5, 7]. The performance experienced by the users is also heavily impacted by their location with respect to the cloud resources. This prompted the providers to reduce the network distance between users and cloud servers via geographically-distributed datacenters.

We focus on C2U networks both for its impact on user-perceived service performance and the scarcity of information available. Of the different network metrics impacting cloud performance—whose individual relevance varies with the specific cloud application—the *latency* perceived by users is a critical parameter for several applications that depend on low latency, low latency variation, or both. These include real-time video processing, cloud gaming [8] or ultra-reliable and low-latency communications services in 5G [9]. The importance of this parameter is highlighted by the introduction of edge-cloud architectures, where additional infrastructure is dedicated

Email addresses: fabio.palumbo@unina.it (Fabio Palumbo), giuseppe.aceto@unina.it (Giuseppe Aceto), alessio.botta@unina.it (Alessio Botta), domenico.ciuonzo@unina.it (Domenico Ciuonzo), valerio.persico@unina.it (Valerio Persico), pescape@unina.it (Antonio Pescapé)

[1] https://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html#~mobile-forecast.

to moving computing resources towards the end-users to reduce the overall latency. This novel paradigm is expected to gain importance, and can benefit from monitoring of C2U network performance. Indeed, edge-cloud architecture can be used to *integrate* cloud-based services rather than to replace them, and network paths towards cloud datacenter are of critical importance in this paradigm as well. Several works therefore investigate the coexistence of both paradigms: since the computational resources available in the edge cloud are limited, it is important to evaluate when the latency requirements of a given task can be satisfied by the core cloud alone, and therefore edge resources can be saved. This further motivates the need to monitor latency in the core cloud, even when edge cloud architectures will have a higher deployment, which is not the case yet.

In this paper we investigate the performance of the cloud services of two most popular public-cloud providers, namely *Amazon Web Services (AWS)* and *Microsoft Azure*. These two providers currently add up to $\approx$ 50% of the market share[2] and are often used together for multi-cloud deployments[3] that allow for using more than one cloud service provider for a single application and switching among them based on necessities, e.g. to optimize performance [10]. We have conducted an extensive, 14-day long experimental campaign[4], monitoring the C2U latency for both providers at *high frequency* and with *multiple active methods* (i.e. relying on different functions of the TCP/IP stack and counterparts at cloud side). Our approach is non-cooperative, since it does not require a privileged point of view or privileges inside the provider's infrastructure. Complementarily, to investigate C2U network performance vs. *the geographical position*, we leveraged Planetlab [12] infrastructure *deploying* 25 *Vantage Points (VPs)* to monitor the network latency perceived by cloud users towards cloud services deployed in 8 *distinct datacenters* (4 per provider) within different continents. The study developed from this experimental campaign *represents an unmatched investigation to date, to the best of our knowledge*. Specifically:

- The multiplicity of providers considered, the higher density of VPs, as well as the higher frequency and diversity of measurements (cf. Tab. 1), all enable a *finer characterization* than previous works. We also discuss how these parameters impact the characterization of latency, and why our choices can provide a more comprehensive and accurate one.
- We provide an overall view of how latency varies (over space and time) with the provider, as well as the location of cloud services and users. Then, capitalizing on *grounded statistical evaluation methods*, we (*a*) compare systematically (at fine-grain) C2U performance of the two providers; (*b*) identify anomalous latency degradation ("badness") events and assess their persistence and dependence on a given provider; (*c*) evaluate the impact of different probing methods on both the observed latency values and the outcomes derived from these observations. Regarding (*a*), we leverage rigorous statistical testing, as opposed to previous works [13] mostly relying on simple statistics for comparison (e.g. minimum, median, and percentiles). Conversely, for what concerns (*b*), we build upon the methodology devised in [14] and tailor it to our experimental setup. Finally, with reference to (*c*), we are able to assess whether different probing methods lead to different answers in practical scenarios (e.g. which provider performs the best or how many badness events are identified), as opposed to previous literature [13].
- We take advantage of the knowledge gained via such characterization to design applications useful to both cloud customers and providers, aimed at supporting cloud-network performance assessment (*badness detection & imputation tools*, also in a real-time online fashion as opposed to offline or coarser granularity setups investigated in previous works) and deployment decisions (evaluation of the benefits of *multi-cloud*). These tools, based on non-cooperative measurement approaches, can help providers troubleshoot and monitor their infrastructure, complementing the information they can collect as owners of the infrastructure, for example to locate faults in a more precise manner compared to a purely-passive approach [14].
- We *publicly release the collected dataset* to promote reproducibility and open research. Remarkably, the richness of the aforementioned campaign implies a dataset able to support detailed investigations along different axes (e.g. the impact of the sampling frequency, the latency behaviour on different ports and/or according to different probing methods) and, equally important, matched comparisons between providers.

The rest of the paper is organized as follows. Sec. 2 reviews the literature on network performance of public-cloud providers; Sec. 3 describes the methodology underlying our experimental campaign and the statistical evaluation methods adopted; Sec. 4 discusses its characterization and provides two design-oriented applications; Sec. 5 ends with conclusions and future directions.

2. Background and Related Work

With the increasing popularity of the cloud paradigm and the resulting adoption of cloud infrastructures, its performance evaluation has more and more attracted the interest of the scientific community, which aims at quantifying the trade-offs between cloud benefits and its inherent limitations. While some studies focused on understanding the implications of deploying specific (classes of) applications onto the cloud [15] , a number of works investigated specific aspects of the complex cloud ecosystem that are related to *cloud networks* [4] together with their cost, their evolution, as well as the resulting performance

[2] `https://www.srgresearch.com/articles/leading-cloud-providers-increase-their-market-share-again-third-quarter`.

[3] `https://www.kentik.com/blog/report-multi-cloud-cost-containment-world/`.

[4] A preliminary version of this paper appears as a conference publication in [11].

and impact on user applications, especially latency-sensitive ones [16].

In fact, cloud providers generally do not disclose (neither publicly nor to customers) the proprietary performance-monitoring information about the state of their infrastructures. This drove the development of *non-cooperative* methodologies, challenging the status quo to investigate cloud networks. These methodologies often take advantage of active monitoring approaches and contrast with the more common *cooperative* ones, that use privileged information and "insider views" only available to service providers (or traffic carriers).

Moreover, as cloud deployment is leveraged by applications with diversified goals and requirements, *different portions* of the cloud network may impact the perceived performance. These correspond to the network paths connecting cloud resources to (*i*) resources in the same datacenter, (*ii*) resources in different far-away datacenters, and (*iii*) cloud users (i.e. the *intra-datacenter*, *inter-datacenter*, and *cloud-to-user* network, respectively).

In the following, we provide an overall view about cloud-network latency benchmarking and monitoring, supported by a detailed taxonomy of the corresponding studies that we report in Tab. 1, including this paper for comparison. We focus in more depth on studies that adopt *non-cooperative approaches to evaluate the latency experienced by end-users when connecting to public clouds*.

Monitoring latency in intra-datacenter networks (intra-DC). Intra-DC networks have become increasingly complex, with even limited increases in network latency, loss or non-optimal bandwidth allocation possibly causing a significant performance degradation, in turn affecting both the user's cost and the service provider's revenues [18]. Today these networks present unique challenges due to their scale, traffic volume, and diversity of faults, thus requiring huge effort to debug and troubleshoot as well as tools to monitor metrics at proper granularity and accuracy. Accordingly, research effort has been made allowing the provider to evaluate traffic patterns, packet drops, load imbalance [21], and especially latency [20]. Non-cooperative approaches have been also investigated, with the intent of evaluating perceived network throughput [28, 5, 6], available bandwidth [29], and latency. Concerning latency, `ptpmesh` [19] has been purposely designed to continuously measure the network latency (one-way delay) and packet loss in datacenters. Leveraging the outcomes of this research, a characterization of the provider intra-DC networks for different providers has been also provided [17].

Measuring intra-DC performance faces non-trivial challenges. Indeed, computer and network virtualization, besides responding to scale and efficiency concerns of the providers [30], also introduce bias in the results provided by monitoring tools [29]. In fact, the virtualization layers cause non-negligible delays and introduce variability in the resulting performance, which are even emphasized in the case of sub-ms latency and when proper hardware configuration is not made available by the providers [17]. Moreover, different kinds of intra-DC paths may exist, leading to severe performance discrepancies [5]. Unfortunately, network topology information is usually kept confidential, despite of its great value [31]. Finally, the impact of the management strategies implemented by the provider should be taken into account to understand performance variability [28].

Monitoring latency in cloud WANs (inter-DC and C2U). According to both research trends and latest reports [32], the performance and the QoS of the cloud wide-area networks is gaining growing interest, for what concerns both inter-DC and C2U networks. Top players have made huge investments in specific technologies, cutting-edge solutions to improve availability, manageability efficiency, and performance such as proprietary WANs deployments [7], advanced CDN solutions [33], as well as sophisticated overlay services [34].

Several works focused on investigating the performance of these cutting-edge cloud WANs in terms of throughput [33, 7, 22, 35], availability [36], latency [7, 22, 35, 16], etc. The outcomes of these studies often resulted in non-trivial and unexpected findings, possibly related to the fact that inter-datacenter connections do not always benefit from proprietary links [7].

Concerning *C2U latency*, only two works analyze C2U latency via cooperative approaches [14, 27], to the best of our knowledge. Jin et al. [14] take advantage of data directly supplied by the provider (Azure) in the form of Transmission Control Protocol (TCP) handshake Round Trip Times (RTTs). Based on the information collected in such way, selected active probing is performed to locate possible issues more precisely. Conversely, Bermudez et al. [27] explore AWS traffic characteristics (and also response time) through a passive analysis from a privileged view at the Point of Presence (PoP).

As data at this refined granularity is not publicly available, most of the works focusing on C2U latency either exploit datasets and information not derived from cloud measurement or collect data via *active probing*. For instance, some works apply measurement-oriented approaches to evaluate the deployment of *hypothetical* cloud services in different geographical locations [37]. Others analyze a snapshot of currently available datacenters and conclude that it is sufficient to provide users around the globe with the necessary quality of experience in terms of response times (20–200ms) for interactive, latency-sensitive applications [16]. Differently, Choy et al. [8] evaluate latency from 3 AWS datacenters towards thousand users (selected among active BitTorrent clients), considering endpoints located in the US. The corresponding outcomes highlight the need to expand the edge infrastructure to satisfy the stringent requirements of the cloud gaming scenario, thus emphasizing the necessity of real cloud measurements to investigate the characteristics of current infrastructures. Laghari et al. [25] evaluate RTT values towards endpoints involving ten cloud and service providers (e.g. Salesforce, Facebook, etc.) from two VPs (located in China and Pakistan) but only consider the average response time. Tomanek et al. [13] present a platform for collecting latency measurements from distributed VPs, named *CLAudit* (acronym for Cloud Latency Auditing platform), considering Azure as provider. RTT is measured at different TCP/IP-stack layers by adopting different probing methods. The same authors also present a detection method for suspicious events using the multi-dimensional data collected [26]. CLAudit was

Table 1: Works dealing with latency measurements in cloud networks, including **this paper**.

Net	Paper	Year	Approach	Metric	Providers	Cloud multiplicity	User/VP multiplicity	Multiple probing methods	Probing period	Open Dataset
intra-DC	[17]	2018	NC	OWD	Azure, AWS, GC	2–10 VMs per CR per provider	-	N	Var.	Y
	[18]	2017	NC*	RTT	Azure, AWS, GC	6 CRs, 4VMs per CR	-	N	1 min.	N
	[19]	2017	C; NC†	OWD	AWS, GC	4 CRs, 4VMs per CR	-	Y	1 sec.	N
	[20]	2015	C	RTT	Microsoft DC	5 CRs	-	N	Var.	N
	[21]	2015	C	RTT	Microsoft DC	*2 clusters*	-	N	-	N
inter-DC	[22]	2019	NC	RTT	AWS, Azure	6–8 CRs per provider	-	N	5 min.	N
	[7]	2017	NC	RTT	AWS, Azure	4 CRs per provider	-	Y	5 min.	Y
C2U	[14]	2019	C	RTT	Azure	-	O(100M) clients	N	Var.	N
	[23]	2018	NC	RTT	Azure, AWS	4 CRs	6 VPs	Y	4 min.	Y
	[13]	2016	NC	RTT	Azure	2 CRs	5 VPs	Y	3–4 min.	Y
	[24]	2016	NC	RTT	Azure, AWS	4 CRs	6 VPs	Y	4 min.	Y
	[25]	2016	NC	RTT	10 service provs.	10 hosts overall	2 VPs	N	-	N
	[26]	2015	NC	RTT	Azure	4 CRs	6 VPs	Y	3 min.	Y
	[27]	2013	C‡	RT	AWS	-	1 VP	N	Var.	N
	[8]	2012	NC	RTT	AWS	3 CRs (US only)	≈2.5k US users	N	30 min.	N
	this	**2020**	**NC**	**RTT**	**AWS, Azure**	**4 CRs**	**25 VPs**	**Y**	**1 min.**	**Y**

Legend:

*: Requires access to Time Stamp Counter register, not always available;

†: NC adoption results in higher variability due to virtualization layers;

‡: Passive analyses, traffic captured at the PoP.

Net: intra-DC (intra-datacenter), inter-DC (inter-datacenter); C2U (cloud-to-user);

Approach: NC (non-cooperative), C (cooperative);

Metric: RTT (round-trip time), OWD (one-way delay); RT (response time).

then expanded to additionally collect measurements towards AWS; these data are then leveraged by Mulinka et al. [23] to detect anomalies via unsupervised learning and by Uhlir et al. [24] to evaluate a benchmarking methodology for cloud providers. The latter methodology allows to compare cloud providers through user-defined simple metrics (e.g. mean latency, standard deviation, coefficient of variation). The work, however, does not provide an in-depth evaluation of the methodology, but simply applies it to a restricted scenario. Equally important, data from multiple source points are aggregated, not investigating per-VP (or per-region) results.

From the above analysis, it is evident that all the literature on C2U latency via non-cooperative approaches is mostly based on the data collected via CLAudit platform. However, each work focuses on a peculiar slice of the whole dataset, either considering different providers, number of probe types, period between each measurement, number of VPs and Cloud Regions (CRs). Compared to the works analyzing C2U latency via active probing, our analysis considers (other than the same number of CRs and both AWS/Azure providers) a *higher number of VPs* (i.e. 25 VPs as opposed to only 6 deployed by CLAudit [5]), covering a *larger* geographical area. Secondly, we add to the probing methods considered in [23, 13, 24, 26] HyperText Transfer Protocol (HTTP) and TCP measurements over non-standard ports, thus allowing to investigate the presence of different enforced policies based on the transport-layer port used for communication. Thirdly, we measure latency with a *finer granularity* (1 min.) w.r.t. previous works.

3. Measurement, Evaluation, and Design Methodology

This section describes the experimental procedure adopted to measure C2U-network latency, along with preliminaries needed for its performance assessment and the design of monitoring tools. In detail, we first motivate and describe the considered public-cloud providers, CRs, and geographically-spread VPs emulating cloud users (Sec. 3.1); secondly, we introduce and discuss the probing methods employed (Sec. 3.2); finally, for reproducibility we provide details of the implementation (Sec. 3.3) and of the statistical evaluation methods employed (Sec. 3.4).

3.1. Public Cloud Providers, Cloud Regions (CRs) and Vantage Points (VPs)

Cloud market is currently dominated by few global providers, with *Amazon* and *Azure* being the clear leaders [6] with millions of active customers in hundreds of countries. Both providers are steadily expanding their global infrastructure, based on the continual billion investments in sophisticated technologies. Hence, in this work we considered the IaaS of

[5] Counting the secondary and backup nodes deployed in the platform it reaches a total of 15 VPs, still less than our campaign.

[6] `https://www.gartner.com/doc/reprints?id=1-1CMAPXNO&ct=190709&st=sb`.

these two cloud providers, namely: *EC2* for Amazon and *Virtual Machines* for Azure. Also, to explore *spatial diversity*, we have identified $R = 4$ regions in distinct geographic continents (hereinafter CRs), where both providers have deployed their datacenters: `Ireland` (Europe), `Virginia` (North America), `Sao Paulo` (South America), and `Singapore` (Asia-Pacific).

To deploy the source nodes for our campaigns, we leveraged the open platform *Planetlab* [12] for emulating cloud users spread worldwide. Since its inception, this platform has supported the development of new network services (e.g. distributed storage, network mapping, peer-to-peer systems) and has been used by thousands researchers from both academia and industry. For our measurement campaign, we relied on $V_s = 25$ Planetlab VPs acting as probing sources. VPs have been placed in $R = 4$ *regions* as the CRs according to node availability, with the following distribution: 8 in Asia-Pacific (`AP`); 6 in Europe (`EU`); 10 in North-America (`NA`); 1 in South-America (`SA`).

We highlight that we *empirically* chose the number of VPs and CRs. While these choices are driven by both VP availability and cost of the experimental campaign, we also point out that our campaign has a finer time granularity and a larger geographical scope compared to the state of the art.

3.2. Probing Methods and Sampling Rate

In our experimental campaign, we adopted *active* probing methods, i.e. that inject probing traffic into the network to estimate the latency in terms of RTT. We remark that the measured RTT includes processing time at the end-host, as well as queueing, transmission, and propagation delays along the whole network path. The latter term, depending on the geographical VP–CR distance, imposes a lower-bound on the latency due to physical constraints. Since cloud resources are addressed leveraging numeric IP addresses (rather than symbolic hostnames), DNS resolution has no impact on the estimated RTT in this paper. We instructed each VP to measure the latency perceived by users with different probing methods by means of probing bulks sequentially issued with 1 min sampling rate. This considered rate is *higher* than that adopted in similar works [36, 13] and thus allows a finer-grained analysis (or, alternatively, aggregate analysis with higher statistical significance).

Furthermore, in line with recent related works [13, 24], in our campaign we adopted *multiple* (namely, 4) active probing methods. Precisely, the adopted probing methods (*a*) take advantage of communication mechanisms at different TCP/IP stack levels and (*b*) possibly rely on different counterparts at cloud side (i.e. servers).

The probing methods used in our work are *(I)* Internet Control Message Protocol (ICMP), *(II)* TCP, *(III)* HTTP, and *(IV)* HTTP-DB. Before discussing each of them in detail (highlighting advantages and implementation requirements), we remark that our experimental methodology was designed to be *independent* from the specific application running on the top of the cloud infrastructure. Indeed, we mainly focused on the latency measured by using lower-layer protocols such as TCP and ICMP. The sole exception is represented by HTTP that, despite being an application-layer protocol, is employed for RESTful applications and HTTP Adaptive Video Streaming, other than browsing. Therefore our methodology aims at giving a broader view of cloud networks performance, not tied to a specific application.

[I] ICMP probing: This method relies on the `echo request/reply` messages. It operates at the network layer and does not require specific instrumentation or tools at server side (i.e. on the virtual machine running via the IaaS paradigm).

Still, Hu et al. [36] suggested that ICMP probing should be used carefully as it may be unsuitable for measurements involving cloud environments since it can lead to either under- or over-estimating service availability.

[II] TCP probing: Differently from ICMP probing, *TCP probing* exploits `SYN/SYN ACK` messages which provide RTT measurements as perceived by data-transfer protocols (instead of being related to ICMP control messages). Notably, (intermediate) network devices may treat TCP (data) traffic differently than ICMP (control) traffic. It however requires a TCP server running on the cloud host.

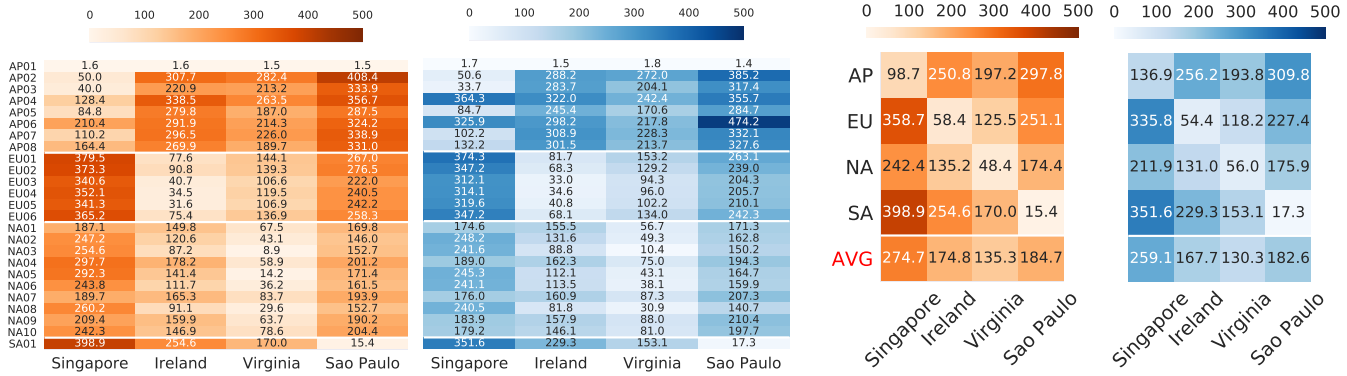
[III] HTTP probing: This method uses HTTP `GET/200 OK` messages. It evaluates the download time of a few-byte resource from the cloud. While the transmission delay is negligible also in this case (due to the small size of the transferred contents), HTTP probing requires a TCP connection to be established, resulting in at least 2×RTT. Also, a processing time on the end-host is implied to serve each request. However, since we request a fixed, small-sized resource from the web server, this processing time is negligible.

[IV] HTTP-DB probing: This method similarly uses `HTTP GET/200 OK` messages as HTTP probing. However, differently from the latter, it relies on a web server that interacts with a database running onto another cloud VM (i.e. an auxiliary server), thus emulating a three-tier application, with latency impacted also by intra-DC contribution (between the web server and the database). In this scenario users have direct interaction with the web-server alone, while the latter is responsible of communicating with external databases, transparently to the final users. Unlike the aforementioned HTTP probe, processing time here has a non-negligible impact, due to the database query. In view of this consideration, the latency in this case is expected to be considerably higher than 2×RTT.

Further, to evaluate the potential impact of *preferential traffic policies* by both cloud and network providers, TCP and HTTP probing (methods **[II]** and **[III]**, respectively) have been tested on *both* well-known (80) and non-standard (54321) destination ports, for a total of 6 probing configurations. No method implements application-level retransmission.

3.3. Reproducibility and Open Research

To summarize, in our campaigns we measured the latency in C2U networks from $V_s = 25$ VPs at 1 min granularity for 14 days. Measurements were run towards cloud datacenters located in $R = 4$ distinct continents and operated by *two* different providers, for a total of 200 measured *(VP, CR) pairs*. Each pair is monitored via 6 probing configurations for AWS, and



(a) AWS, detailed. (b) Azure, detailed. (c) AWS, aggregated. (d) Azure, aggregated.

Figure 1: Average latency [ms] (14-day span, TCP probing method, port 80). (a) and (b) report detailed results at `(VP, CR)` pair granularity for AWS and Azure, respectively. (c) and (d) report results aggregated (average) by VP region for AWS and Azure, respectively. `AVG` reports the CR-average. While in general VPs experience lower latency toward the CR in the same geographic zone, `Virginia` reports the lowest latency from a VP-global perspective.

via 5 configurations for Azure, due to traffic-filtering policies implemented by the provider at the time of collection. Therefore, our dataset results in 1100 distinct time series, with $\approx$ 14k samples each. Further, we highlight that the difference in length among time series is due to different factors. Indeed, all the probing methods considered are subject to different kinds of errors (e.g., connection reset, port unreachable, timeout), possibly originated by the VP, the cloud service, or the network infrastructure connecting them. We remark that errors in the probing process (such as those listed above) result into missing values (`None`) in the collected dataset. Moreover, retransmissions are implemented only for HTTP and HTTP-DB probing (leveraging HTTP all the features of the lower layers in the stack) and not for TCP probing. Regarding the implementation aspects, we employed *HPing3* for ICMP and TCP probing methods (**[I]** and **[II]**, respectively). Differently, we resorted to *HTTPing* for HTTP and HTTP-DB (**[III]** and **[IV]**, respectively) probing methods. Also, we ran *MySQL* database on the auxiliary server for HTTP-DB. Finally, to support open research via reproducibility [38] of our study and fostering further advances on public cloud services assessment, the *dataset is publicly released* at: `http://traffic.comics.unina.it/cloud`.

We remark that when performing active measurements leveraging distributed measurement infrastructures, (*a*) the number and the location of the VPs, (*b*) the probing mechanisms adopted and (*c*) the probing rate are all critical aspects that can potentially impact the collected data. In order to improve the richness of the dataset, in the experimental plan supporting our measurement campaign we have set these aspects so as to improve the data collection available in similar past work.

Beyond the numerical improvement, in order to wisely define the configuration of our campaign, we have selected them guided by both past research experience in the field and empirical evidences (e.g. differential treatment observed for ICMP and TCP packets [36] or the existence of proxies possibly implementing management strategies such as TCP splitting based on L4 port numbers) and opted for distributing the VPs across different continents (avoiding placing nodes in the same cities), according to the node availability granted by infrastructure. Since the number of VPs and their geographical position reflect the distribution of the final users that the VPs are intended to emulate, we claim that a higher number of well-separated VPs brings a more comprehensive and complete characterization.

Also, concerning the sampling rate, our goal was to be able to catch sudden spikes and transient behaviors. Hence we considered 60-second between subsequent measurements. We report that our results exhibited intermittent spikes for several (VP, CR) couples, which do not appear to follow a specific pattern. A lower sampling rate increases the chance of missing these events (that are frequently observed and are possibly related to the bursty nature of Internet and datacenter traffic [30, 19]), therefore leading to an underestimation of latency variability (we recall that low latency variability on short timescales is important for several applications, such as real-time video streaming or cloud gaming [16]).

Considering the mentioned aspects, the proposed experimental campaign results in an unmatched measurement analysis when compared to studies leveraging similar approaches. In these terms, we claim that the above described dataset enables a better characterization of latency over *space* and *time*. The above fine-grained experimental campaign allowed us to obtain and release publicly a dataset whose richness can foster comparisons to investigate the impact of the sampling frequency, the behaviour on different ports and with different probing methods and, last but not least, a matched provider comparison.

3.4. Statistical Evaluation Methods

In order to provide a statistically-sound analysis of the latency gap between the two providers, we use the **Wilcoxon signed-rank test** [39]. It is a *non-parametric* hypothesis test used to compare whether the mean ranks of two populations ($\{x_i\}_{i=1}^{N}$ and $\{y_i\}_{i=1}^{N}$, respectively) differ. The statistic is calculated as follows: (*a*) let $\bar{N} \leq N$ be the number of pairs s.t.

$|y_i - x_i| \neq 0$; (*b*) the non-zero pairs are given a rank $\mathcal{R}_i$ according to the increasing order of $|y_i - x_i|$ (i.e. the smallest $|y_i - x_i|$ gets $\mathcal{R}_i = 1$); (*c*) pairs with the same $|y_i - x_i|$ are given the average of the ranks they span. The statistic is evaluated as:

$$W_{\text{wil}} \triangleq \sum_{i=1}^{\bar{N}} [\text{sign}(y_i - x_i) \cdot \mathcal{R}_i] \tag{1}$$

W_{wil} is then compared to a suitable threshold (defined to enforce the desired p-value). In Sec. 4.3, we use W_{wil} to compare whether statistically-significant different latency values between Azure and AWS time series are observed on a given pair. In affirmative case, the *sign* of W_{wil} is considered to discern which provider experiences lower latency. The use of W_{wil} confers robustness to such comparison (as opposed to Student's t-test), accommodating deviations of the measured latency from Gaussianity and, also, to *outliers* (e.g. very-short events with much-higher latency).

To assess a statistically-significant different *latency-variability* of either provider we adopt **Levene's test** [40]. It is a hypothesis test used to assess the *equality of variances* among K populations. Let N_i be the number of samples of ith population and let $N \triangleq \sum_{i=1}^{K} N_i$. Also, define the score of jth sample within ith group Z_{ij}, as the *unsigned residual* of the mean, the median or the trimmed mean (this choice is arbitrary). Further, define the *per-group* and *overall* score means as $\bar{Z}_i \triangleq \frac{1}{N_i} \sum_{j=1}^{N_i} Z_{ij}$ and $\bar{Z} \triangleq \frac{1}{N} \sum_{i=1}^{K} \sum_{j=1}^{N_i} Z_{ij}$, respectively. The statistic is:

$$W_{\text{lev}} \triangleq \frac{(N-K)}{K-1} \frac{\sum_{i=1}^{K} N_i (\bar{Z}_i - \bar{Z})^2}{\sum_{i=1}^{K} \sum_{j=1}^{N_i} (Z_{ij} - \bar{Z}_i)^2} \tag{2}$$

W_{lev} is then compared to a threshold (defined as to enforce the desired p-value). In Sec. 4.3, we use W_{lev} to assess a statistically-significant lower *latency-variability* of either Azure or AWS time series on a given pair. As for W_{wil}, the use of W_{lev} confers robustness (as opposed to Bartlett's test) in the considered latency-variability comparison.

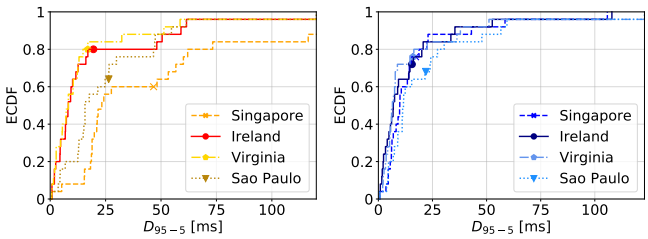
Finally, in what follows, to evaluate **badness events** (i.e. abrupt, but appreciably-persistent latency increments in a time series), we split each `(VP, CR)` time series in "buckets", taking inspiration from [14]. More specifically, we denote the duration of a bucket as T_{bu}, the number of corresponding samples as N_{bu}, and the vector of latency values associated to the bucket as $\mathbf{x}_{bu}$.

Each bucket is marked as *"bad"* if the statistic

$$\lambda(\mathbf{x}_{bu}) > \gamma_{bu}, \tag{3}$$

namely the summarizing function $\lambda(\cdot)$ exceeds the "badness baseline", representing the threshold between bad and normal latency levels. Although many choices could be investigated, we opt for the following: *(i)* we adopt the median as a robust indicator of typical within-bucket latency values, while *(ii)* we resort to the 75^{th} percentile of the time series values for the badness baseline, since it represents a reasonable data-driven threshold for abnormal latency values. Still, we highlight that the principles in this work may be applied whenever a different summarizing function $\lambda(\cdot)$ or (adaptive) baseline γ_{bu} are adopted, including the case of provider-specified (i.e. fixed) values [14].

Finally, we remark that in this work the data-driven calculation of γ_{bu} is performed in *two different fashions*, representative of two different applicative scenarios. On the one hand, in Sec. 4.4 (related to characterization), the badness threshold is obtained in an *offline fashion* from *all the samples* within the series. On the other hand, in Sec. 4.6 we calculate it in an *online fashion* to emulate a realistic setup for the investigation of the proposed badness dashboard. This online calculation involves (*a*) initializing the badness baseline by the observations of the first two days for each `(VP, CR)` pair to learn the normal behaviour and then (*b*) updating γ_{bu} as more recent latency samples are observed while (*c*) discarding all the samples contained in the buckets flagged as bad so far. Finally, as a good trade-off between detecting bad events at fine granularity and a sufficient number of samples to draw out statistically-meaningful outcomes, we choose $T_{bu} = 30$ min (hence $N_{bu} = 48$) in our experimental analysis.



(a) AWS. (b) Azure.

Figure 2: Variability in terms of $D_{95-5} = 95^{th}pctl - 5^{th}pctl$. Markers highlight the average of each distribution.
On average, Virginia shows lower variability, while AWS in Singapore reports a constantly-worse variability compared to the other CRs.

4. Experimental Results

In this section we provide a statistical characterization of our experimental campaign. Specifically, Sec. 4.1 provides an overall performance assessment, whereas Sec. 4.2 performs a pointly AWS-Azure joint analysis. Section 4.3 compares the performance observed for the two providers leveraging the previously introduced statistic methodology. Then, Sec. 4.4 analyzes badness events over space, time and between providers. Sec. 4.5 delves into latency dependence on different probing methods considered.

After that, we capitalize on the previous characterization for design purposes. In detail, Sec. 4.6 derives a dashboard for the online detection and imputation of badness events. Finally, Sec. 4.7 investigates the benefits of multi-cloud deployments.

4.1. Overall view

We first provide a high-level view of the C2U latency considering each `(VP, CR)` pair *separately*. Differently from previous works [13], where minimum and median values are used for

an high-level characterization, we provide a more detailed and comprehensive view of the results. Note that herein we aim at proving a comprehensive view and do not filter out latency samples related to badness events according to the methodology introduced in Sec. 4.4. Figs. 1a and 1b report the average latency (over the 14-day campaign) experienced from each VP when targeting the four CRs for AWS and Azure, respectively, and considering TCP probing (port 80) without loss of generality (the impact of the probing method is investigated in later Sec. 4.5). Also, VPs located in the same geographical region are grouped together. First, results show that latency values (intuitively) grow with the distance between the VP and the CR (lower values are observed for paths connecting VPs and datacenters within the same geographic region). Interestingly, this finding does not always hold in inter-DC networks [7] or with other network metrics in analogous contexts (e.g., network throughput [33]). From these results it is also evident that VPs belonging to the same geographical region can exhibit different behavior, thus showing the benefits of the finer spatial VP displacement provided by our study.

Secondly, Figs. 1c and 1d report the previous results after *aggregating* (by averaging) VP results by geographic zones matched to the four CRs, with last row ("AVG") representing the CR-average. The figures (beyond expected lower values on the main diagonal, corresponding to latencies measured within the same region) show how the `Singapore` CR is the one with highest average latency for both providers, with the VPs in `SA` representing the worst case. Differently, the deployments in `Virginia` offer, for both providers, the *lowest latency on average*, namely considering all the VPs across the world (with VPs in `AP` being the more penalized). Hence, by supposing a cloud customer wants to deploy an application leveraging a single CR (e.g. for budget constraints), and considering potential users scattered around the globe, using `Virginia` datacenters would be the most suitable choice. This result aligns to those about network throughput in [33].

Moving from a time-averaged analysis to a tail-based one, in Figs. 2a and 2b we provide a quantification of the *variability of the latency* over time for AWS and Azure, respectively, as it is experienced by geographically-spread VPs when connecting to the considered CRs. We adopt as the relevant metric the *difference between the* 95^{th} *and the* 5^{th} *percentile* of the latency distribution for each `(VP, CR)` pair (denoted as D_{95-5}). This metric does not focus on latency time-evolution, but simply accounts for the spread between high levels (possibly induced by congestion, suboptimal routing, etc.) and low ones, observed in the 14-day analysis. Adopting the 95^{th} and 5^{th} percentiles (rather than $\max(\cdot)$ and $\min(\cdot)$, respectively) allows to filter-out outliers and provide observations not related to network spot conditions. For conciseness, both figures depict the empirical CDFs (ECDF) of D_{95-5} values corresponding to the four CRs.

This analysis allows to draw the following observations: *(i)* taking into account the per-CR breakdown, D_{95-5} is lower than 25 ms (resp. lower than 50 ms), on median (resp. on average); *(ii)* this notwithstanding, the distribution of D_{95-5} shows long tails, with values higher than 100 ms for both providers; *(iii)* in more detail, `Ireland` and `Virginia` CRs result in lower variability than `Singapore` and `Sao Paulo`, on average, for both providers; however, while this discrepancy is almost negligible for Azure (e.g. +7.5% D_{95-5} for `Singapore`, w.r.t. `Virginia`, on average), this phenomenon is more evident for AWS (e.g. +175% D_{95-5} for `Singapore`, w.r.t. `Virginia`, on average). To conclude, for most of the `(VP, CR)` pairs the variability is limited. However, some cases exist (with evidences of dependence from the CRs and the providers) for which higher D_{95-5} values are observed. Results about tail latency and variability are in line with those found by Tomanek et al. [13]. A detailed analysis has also revealed that part of the discussed variability comes from intermittent spikes that are observed for several `(VP, CR)` pairs, not appearing to follow a specific pattern. We underline that a lower measurement frequency reduces the possibility of capturing those spikes, and therefore we claim that higher sampling frequency contributes to a better characterization.

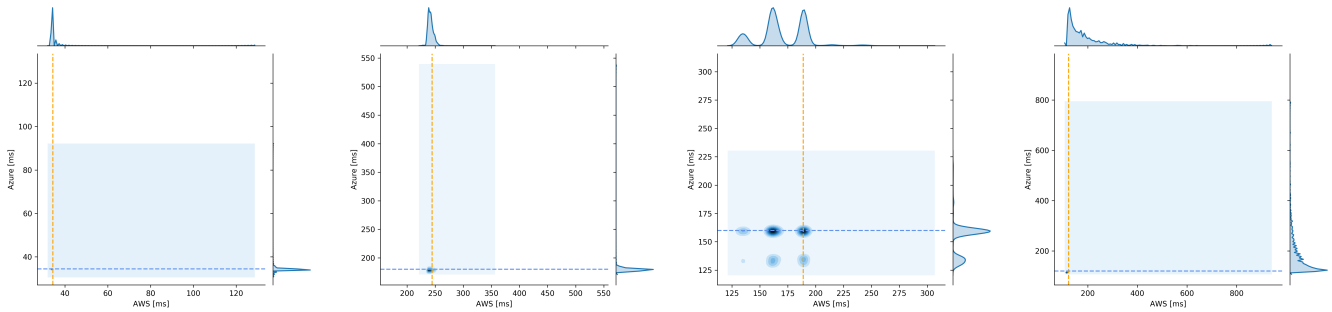
Finally, we highlight that the proposed analysis is not suited to evaluate the trade-off between cloud costs and performance in terms of latency, since differently than for other network performance metrics (e.g. bandwidth [28, 7]) or cloud services (e.g. CDNs [33]), cloud customers are not expected to experience better latency for higher costs for the IaaS under test; indeed, the results of this kind of analysis are dependent on the mutual location of VP and CR, rather than on the specific rented service.

4.2. Joint Analysis

To provide a first comparison of the performance of the two providers at fine-time granularity (1 min) and *from a qualitative standpoint*, in Fig. 3 we report the joint and marginal empirical distributions of latency values for the two providers, for *selected* `(VP, CR)` pairs. Each plot is obtained by means of Kernel Density Estimation with Gaussian kernel, highlighting the frequency of occurrence of quasi-contemporary (less than 1 min apart) probing results towards the two providers (central plots), and the marginal distributions of values related to the single provider (side plots). Dashed lines correspond to the 75^{th} percentile. The scale for density is intentionally omitted as this section aims at a qualitative report (differently than Secs. 4.3 and 4.4). The selected plots exemplify the variety of the joint behaviors observed, corresponding to the *main categories* described hereinafter.

[a] Unimodal-Unimodal, Peaked, Symmetric. This case, depicted in Fig. 3a, (38% of pairs, i.e. the relative majority over the measurement campaign) presents very similar results for both providers with low variance. In such cases in a multi-cloud scenario there would be no reason to prefer any of the providers, and the multi-cloud setup could be motivated by availability guarantees (or reduction of business risks associated with a single provider).

[b] Unimodal-Unimodal, Peaked, Asymmetric. Represented by the case in Fig. 3b (12% of pairs), it is similar to the previous behavior as the results are stable on the observation period for both providers. For this case, instead, the 75^{th} percentiles are consistently and non-negligibly different: a multi-



(a) Unimodal-Unimodal, Peaked, Symmetrical (`EU04, Ireland`).

(b) Unimodal-Unimodal, Peaked, Asymmetrical (`NA10, Singapore`).

(c) Multimodal-Multimodal (`SA01, Virginia`).

(d) Spread-Spread (`EU06, Virginia`).

Figure 3: Joint and marginal empirical distribution (by Kernel Density Estimation) of quasi-contemporary measurements. Dotted lines indicate 75^{th} percentile. Different common behaviors are reported with a representative example case of (`VP,CR`) pair.

cloud scenario would leverage this measured difference to prefer the quick-responding provider.

[c] Multimodal-Multimodal. A more complex case (6% of pairs) is exemplified in Fig. 3c, in which each provider exhibits a multimodal latency distribution, with evident peaks showing different but relatively-coherent alternative path conditions. Multimodal distributions for C2U latency were also observed in previous works (e.g. [13]), and we can thus confirm this trend. Similarly to the aforementioned *asymmetric* case, a multi-cloud environment can benefit from the consistent difference of performance between the two providers. Differently, in this case the consistency of measurements has a lower time scale, implying the need for either a finer granularity of monitoring timescale, or a statistical approach, to reap the benefits from C2U network monitoring.

[d] Spread-Spread. Differently from previous cases, in this one (11% of pairs), exemplified in Fig. 3d, the higher variance highlights a less consistent behavior. This suggests the opportunity for a statistical approach to benefit from C2U network monitoring beyond the simple availability enhancement.

[e] Different characteristics. While previous cases regarded similar behaviors for the two providers, the remaining number of pairs (33%) showed different characteristics between them. Combinations with a multimodal vs. unimodal, or peaked vs. wide-variance cases, were observed. For these heterogeneous cases, a multi-cloud environment would allow to prefer one provider to the other in terms of performance based either on long-term stability or short-term improvement.

Overall, the cases in which a behavior difference between the two providers could be exploited to optimize performance amount to 51% of (`VP,CR`) pairs (namely, the cases **[b]**, **[c]**, and **[e]**). This validates the actual possibility for performance optimization in a real-world multi-cloud environment, based on measurements in-the-wild, and motivates the following in-depth quantitative analysis.

4.3. Comparison between Providers

The results provided in the previous section highlight that neither provider always outperforms the other, as the outcomes of the analyses vary with the specific (`VP, CR`) pair. Indeed, from results shown in Fig. 1, we can see that even VPs located in the same geographical region may disagree in terms of which provider reports lower latency, on average. To report a few examples, on average AWS performs slightly worse than Azure for `AP03` towards `Singapore`; however, its latency is far lower considering `AP04` and `AP06` and the `Singapore` CR. Similarly in Europe, where on average `EU01` reports a 4 ms lower latency for AWS towards `Ireland`, while `EU02` reports a 22 ms lower value for Azure. In this sense, we believe that the higher number of VPs employed has led to a more comprehensive characterization, and these results are already an indication that even differences within the same region need to be taken into account when assessing user QoS.

To perform a grounded comparison, we leverage the measured latency for AWS and Azure over time by means of the statistical tests introduced in Sec. 3.4. We highlight that, differently from previous works [24], our analysis *does not rely on heuristics* (e.g. the vector magnitude of the series) to compare the providers, but capitalizes solid hypothesis testing. In fact, we use the outcome of the Wilcoxon signed-rank test (W_{wil}) to assess a statistically-significant difference (the p-value is set to 0.01) in latency time series for any (`VP, CR`) pair. Fig. 4 reports the outcome based on W_{wil} with a per-CR barchart, highlighting with blue (resp. orange) color for how many VPs each provider performed better on the 14-day span. We highlight that our analysis did not highlight cases where the relative comparison was non statistically-significant (i.e. under the considered p-value).

Then, statistically-significant comparisons are broken down by *(i) intra-region* cases (VP and cloud datacenter in the same region, "○" texture) and *(ii) inter-region* cases (VP and cloud datacenter in different regions, "\\" texture). Results show that the *best-performing provider changes with the CR considered*. For instance, for services delivered via `Ireland` and `Virginia` CRs, AWS reports better performance for more VPs (13 and 14 VPs out of 25, respectively). Differently, Azure performs better when targeting `Sao Paulo` and `Singapore` CRs (17 and 19 VPs out of 25, respectively). Also, by limiting the analysis to *intra-region* cases, AWS *always outperforms* Azure, especially in `Virginia` CR (e.g., 9 out of 10 VPs deployed in `NA` expe-

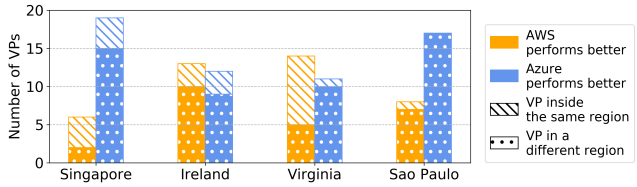


Figure 4: Comparison of the two providers in terms of latency, based on W_{wil} (14-day span, TCP probing method, port 80). Most VPs report better performance for Azure towards `Singapore` and `Sao Paulo`, while towards `Ireland` and `Virginia` there is an opposite trend (although the difference is not significant).

rienced lower latency towards AWS). Lastly, an opposite trend is seen for *inter-region* cases with Azure (except for `Ireland` CR).

Beyond desirable low latency values, a wide range of applications also demand its small variability over time [13, 8, 41, 42]. Hence, we used the *Levene test* (W_{lev}) to assess whether there is a statistically-significant difference (the p-value is set to 0.01) in the latency variability of the two providers for a given `(VP, CR)` pair, i.e. to test the equal-variance hypothesis for the two time series. Hence, Fig. 5 reports the comparison (over the 14-day span) of latency variability expressed as the variance, with a row for each CR and a column for each VP. Precisely, blue (resp. orange) color boxes highlight `(VP, CR)` pairs where Azure (resp. AWS) exhibited a lower variability in latency (based on W_{lev}). Differently, black boxes highlight non statistically-significant difference cases. First, a non-negligible amount of `(VP, CR)` pairs with no significant difference in latency variability between providers is observed, with up to 4 VPs out of 25 toward `Ireland` and `Virginia` CRs, in contrast with Fig. 4. Interestingly, this implies that in a number of cases *lower latency does not imply also reduced variability*. Focusing on statistically-significant comparisons, the result is also in this case influenced by the specific CR, with `Singapore` (resp. `Sao Paulo`) leading to a lower variance for Azure (resp. AWS) in most cases. For other CRs, the comparison is more balanced and depends on the VP. Conversely, for only three VPs (`AP03`, `EU05`, `NA03`) a lower variability towards all the `CRs` is guaranteed by the same provider.

Finally, we have also used the proposed methods to perform a statistical comparison focusing on temporal patterns. In more detail, we have looked for *daily*, *weekly*, and *hourly patterns* (i.e., comparing measurements for the two providers limiting the attention on each of the 14 days of our campaign, comparing data coming from the same day of the week, and considering measurements performed during 4-hour ranges, respectively). For all the `(VP, CR)` pairs where the better performing provider changes over time, none of these analyses (whose details are omitted for brevity) reported evidences of a clear dependence upon the specific time frame at any of the considered scale. These results further witness the need for online monitoring activities to support decisions, that cannot be replaced by simpler time-based assumptions.

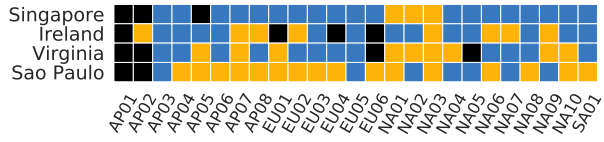
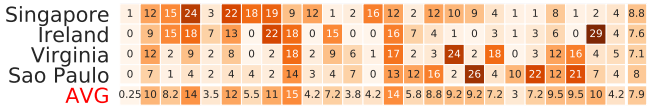
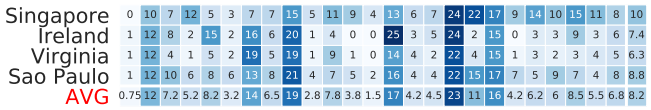


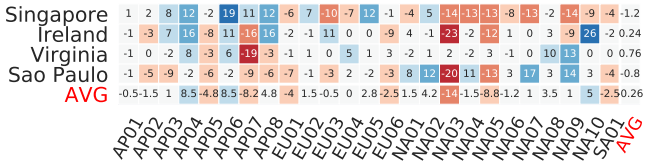
Figure 5: Comparison in terms of latency variability, based on W_{lev} (14-day span, TCP probing method, port 80). Orange and blue color report cases where AWS and Azure show lower variability, respectively. Black color highlights no significant difference between them. Lower variability is experienced from most VPs by Azure (resp. AWS) on `Singapore` (resp. `Sao Paulo`).



(a) AWS, detailed.



(b) Azure, detailed.

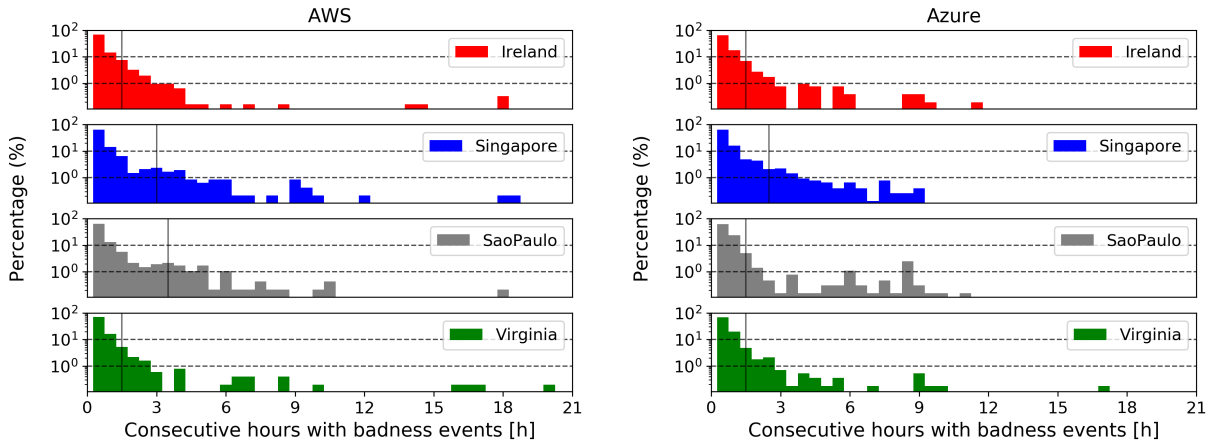


(c) Difference, detailed.

Figure 6: Badness [%] in 30 min buckets (14-day span, TCP probing method, port 80). (a) and (b) report detailed results at `(VP, CR)` pair granularity for AWS and Azure, respectively. Differently, (c) reports the [%] difference heatmap. `AVG` reports either the CR- or VP-average. Average badness may differ depending on the VP or CR. However, no clear patterns emerge.

4.4. Badness Analysis

First, we provide an assessment of badness events as defined in Sec. 3.4 for each `(VP, CR)` pair. We highlight that our methodology for badness events is inspired by Jin et al. [14], although Tomanek et al. [13] have also proposed their methodology for anomaly detection and interpretation. Both methodologies rely on the tuning of some parameters (window size, threshold for detecting events), but we deemed the first one more flexible and suitable for our scenario. To discuss the evaluation, Figs. 6a and 6b report the percentage of bad buckets over the 14-day campaign from each VP when targeting the *four* CRs for AWS and Azure, respectively, and considering TCP probing (port 80). VPs located in the same geographical region are grouped together. Although the overall badness quantities for the two providers are quite similar (i.e. 7.9% vs. 8.2%, see bottom-right entry of the heatmap, "AVG-AVG"), results highlight different peculiar patterns along VP- and/or CR-dimensions. This is also confirmed by a more detailed ECDF-based analysis, omitted for brevity. Such result shows that badness events are mostly related to either the VP or the destination area (or both), rather than to the provider infrastructure.



(a) Persistence of badness events for AWS.

(b) Persistence of badness events for Azure.

Figure 7: Persistence of badness events for both providers. Each empirical PMF shows the number of consecutive buckets reporting a badness event considering the whole 14-day measurements campaign and all VPs ($V_s = 25$) towards a specific CR. For both providers, badness events tend to be short-lived (less than 1.5 hours), save from few event cases lasting more than 15 hours, especially for AWS (a).

Table 2: Empirical conditional probability [%] of badness events.

(a) AWS.

	$X =$	
	B_{AZ}	$\neg B_{AZ}$
$Pr(B_{AWS}\|X)$	11.45	8.24
$Pr(\neg B_{AWS}\|X)$	88.55	91.76

(b) Azure.

	$X =$	
	B_{AWS}	$\neg B_{AWS}$
$Pr(B_{AZ}\|X)$	11.83	8.51
$Pr(\neg B_{AZ}\|X)$	88.17	91.49

By aggregating in terms of CRs (i.e. in a per-VP view), for example, higher badness percentages are observed from `EU01` and `EU06` for *both* providers toward all the four CRs. Differently, an analogous behaviour is seen for `NA03` (having the highest overall average percentage toward all the CRs) and `NA05` for the sole Azure. Differently, by aggregating in terms of VPs (i.e. in a per-CR view), we observe a slightly different behavior for different CRs, with the same relative ranking observed for *both* providers (`Singapore` → `Sao Paulo` → `Ireland` → `Virginia`). Interestingly, the *highest* badness percentage is experienced by `NA10` when targeting AWS `Ireland` CR; however, such result is *scattered* as it corresponds to neither aggregations over VPs nor over CRs.

Finally, a direct comparison of badness percentage between the two providers, on a `(VP, CR)` basis, is given in Fig. 6c, where the *difference heatmap* is depicted. Results highlight no structured pattern, with relative badness performance depending on the specific `(VP, CR)` pair. For example, the highest badness increase incurred by Azure (w.r.t. AWS) is observed for `NA10` and `AP06` toward `Ireland` and `Singapore` CRs, respectively. Differently, the highest badness increase incurred by AWS (w.r.t. Azure) is observed for `NA03` toward both `Ireland` and `Sao Paulo`.

In order to evaluate the severity of these badness events, we analyze how much badness events are *correlated over time*, i.e. we analyze the incidence of consecutive badness events in latency time series. To this end, in Figs. 7a and 7b, we report the empirical probability mass function (EPMF) of the *badness persistence* (namely, the duration of a badness event in multiples of $T_{bu} = 30$ min buckets), for AWS and Azure, respectively. For each figure we show *four* EPMFs, corresponding to the considered CRs, obtained by aggregating badness persistence samples over the VPs. Results show that the EPMF of bad buckets persistence is qualitatively similar when varying CR and the provider, showing a (reasonable) decreasing trend with the increasing persistence. Most of badness events ($\approx 60\%$) last for one single bucket (i.e. they are bounded in a 30 min interval) and the 95^{th} percentile is always constrained within 3 h, with the sole exception of `Sao Paulo` for AWS, being ≈ 4 h. Similar findings about the short-lived duration of anomalous events were also discussed in previous works [13, 14]. Interestingly, sporadic badness events with high persistence ($\in (12, 20)$ h) appear for some specific cases involving all AWS CRs and Azure `Virginia` CR.

Finally, we analyze how much badness events are *correlated between providers*, i.e. we analyze the incidence of a badness event of one provider on the other considering the same `(VP, CR)` pair and the same time span. Taking into account a given `(VP, CR)` pair, we denote with B_{AWS} and B_{AZ} the binary (i.e. $\in \{0, 1\}$) badness event for AWS and Azure, respectively, for the same 30 min time span for this pair. Then, we consider the two *empirical conditional probabilities* $\Pr(B_{AZ}|B_{AWS})$ and $\Pr(B_{AWS}|B_{AZ})$, to investigate how much a badness event experienced by one provider reflects into a similar badness event for the other provider.

The results in Tabs. 2a and 2b report these conditional probabilities averaged over all the `(VP, CR)` pairs: when a badness event is observed for one provider, the probability that a badness event is observed also for the other is lower than 12%; Differently, when a badness event is not observed for one provider, the probability that a badness event is observed for the other one is $\approx 8\%$. Being these two close values, we can conclude

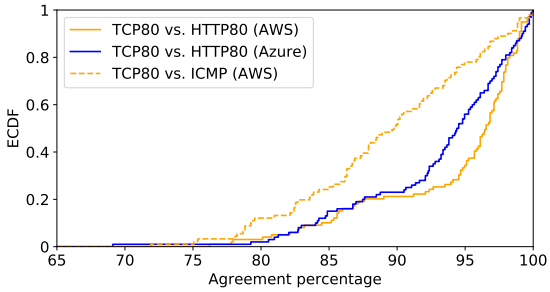


Figure 8: ECDF of agreement [%] in detecting badness events between probing-methods pairs. Each value of the ECDF reports the agreement related to a (VP, CR) pair, considering all the related time series. For all the cases, the agreement is above 67%. Still, the median agreement between TCP and ICMP is the lowest.

that B_{AWS} and B_{AZ} are *loosely correlated*.

4.5. Impact of probing methods

As discussed in Sec. 3.2, C2U latency can be measured via a plethora of active-probing methods, which possibly require different configurations at server side. Our results report that probing methods adopting mechanisms implemented at different levels of the TCP/IP stack may lead to different latency estimates. On the other hand, the analyses that leverage these measures (e.g., provider comparison as in Sec. 4.3 or badness analysis as in Sec. 4.4) provide the same outcomes in most of the cases, independently of the probing method. Details are provided in the following.

First, we aim at assessing whether *distinct probing methods, evaluating the latency for the same (VP, CR) pair at the same time, highlight any of them reporting systematically-lower latency values.* The Wilcoxon signed-rank test W_{wil}, used to compare 14-day time series, reports the following outcomes. Considering the latency comparison between (*a*) TCP 80 vs. 54321, (*b*) HTTP 80 vs. 54321, (*c*) HTTP vs HTTP_DB and (*d*) ICMP vs TCP 80 for any (VP, CR) pair and both providers (when applicable), we observed statistically-significant differences between the considered time series. Still, while for (*c*) this discrepancy corresponds to a *systematic* relative order between the probing methods (i.e. a higher latency was found for HTTP_DB, with $\approx$ 9 ms higher latency than HTTP averaging all over the pairs, due to the latency of the intra-DC network and the processing time at the auxiliary server), in the other cases ((*a*),(*b*), and (*d*)) there is no clear pattern stemming out from the analysis: while there is always a probing method measuring a consistently-higher latency for each (VP, CR) pair, such method is not always the same.

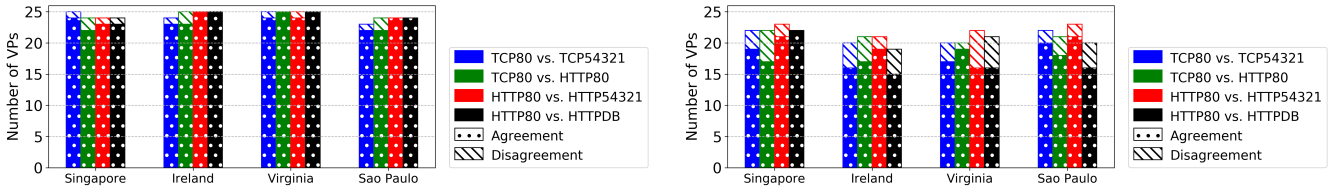
While investigating the root cause of these discrepancies is out of the scope of this work, results suggest that these aspects should be taken into consideration when designing non-cooperative methodologies for monitoring public-cloud networks. Indeed, in our case different probing methods returned measurements that differ up to 198 ms, on average. For instance, this is evident for the AP01 VP, where the TCP proxy presence along the path towards the cloud causes the monitored latency to be heavily underestimated when using TCP probing with port 80. Indeed, referring to the results for this VP, latency measured using TCP over port 80 consistently reported RTT values around 1 ms, irrespective of the CR—thus highlighting the presence of a network proxy interfering with the measurement process. On the other hand, measures leveraging the non-standard port appeared more realistic, being variable depending on the destination region and compatible with expected VP-to-CR RTTs. Moreover, HTTP did not report relevant differences between the two ports. This can be explained considering that, in this case, latency measures the response time of a specific page, which is sent from the web-server at the intended destination.

Concerning ICMP, our results are in line with what observed in [36] for service availability measurements: though ICMP is widely adopted—as it does not require particular instrumentation at the targeted cloud node—its results can differ from latency experienced by upper-layer protocols, possibly leading to either an underestimated or (more often) an overestimated observed value.

Secondly, we *evaluate whether different probing methods agree in identifying badness events (cf. Sec. 3.4), and to what extent*. The results of this analysis are shown in Fig. 8, where the ECDF of percentage agreement in detecting badness events between probing methods is shown.[7] In general, we observe an agreement $\in (78, 97)$ %. Differently, higher levels of agreement between TCP and HTTP are visible (mostly not depending on the specific provider) with an average agreement around 93–94%. Concerning the agreement between TCP and ICMP, the average agreement is around 82%. Interestingly, looking at detailed agreement percentages (i.e. per (VP, CR) pairs, not shown for brevity), the level of agreement appears to (slightly) vary with the VP but it is not impacted by the specific CR (with the exception of a few cases related to TCP vs. ICMP comparison).

Finally, we investigate *what is the impact of the probing method adopted in defining which provider achieves better performance.* The results are shown in Fig. 9, reporting the number of agreements/disagreements (between probing-method pairs) of the outcomes provided by the Wilcoxon signed-rank test (Fig. 9a) and Levene test (Fig. 9b), aggregated by CR. The former test is used as an intermediate outcome to compare the punctual latency of providers (for each probing method) on each (VP, CR) pair as in Sec. 4.3. Similarly, the latter test is employed to compare the standard deviation (viz. latency variability) of the two providers. We highlight that some bars do not sum to 25 VPs for each CR, because in some instances the test itself returns that the discrepancy between providers is not statistically-significant. Notably, we can observe that the disagreement cases are limited to 13/400 (i.e. $\approx$ 3%) in Fig. 9a. Differently, in Fig. 9b, a larger number of cases report a non-statistically significant difference, and more disagreements appear (50/400 cases, corresponding to 12.5%); the highest num-

[7] Note that the analysis takes into account the agreement between TCP and HTTP (port 80) for both providers while the one between TCP and ICMP only for AWS, based on measure availability. Results for non-standard ports are omitted for brevity.



(a) No. of (dis)agreements **considering punctual latency**. Disagreements are observed only in about 3% of the cases.

(b) No. of (dis)agreements **in terms of standard deviation**. We observe more disagreements (12.5% of the cases) compared to (*a*).

Figure 9: No. of (dis)agreements between probing methods pairs about best provider across different protocols in the four CRs, considering (*a*) punctual latency or (*b*) latency variability as the relevant metric.

ber of disagreements (13 cases) is reported between TCP80 and HTTP80 and between HTTP80 and HTTP_DB.

To summarize, the above analyses witness that, notwithstanding the observed discrepancies in the observed values, the outcomes of both badness analysis and provider comparison are not impacted by changing the probing method.

4.6. Online badness detection and imputation

Leveraging the rationale behind the results of Sec. 4.4, hereinafter we design and analyze the outcome of the *online* detection and imputation of badness events. Here we take advantage of the online calculation of the badness threshold as detailed in Sec. 3.4, whose resulting information is used to compose what we call *badness dashboard* which highlights badness events and allows to correlate them based on the `(VP, CR)` pairs. This tool can effectively aid the providers, giving a basic tool to enable *badness imputation*, i.e. to provide early warnings of root causes when performance degradation is observed. Although we have limited the badness analysis to one protocol (TCP on standard port), providers can also benefit of multi-protocol measurements to precisely pinpoint the cause of degradation. We stress that our non-cooperative approach complements server-side information that can be collected by providers. By integrating measurements about user-perceived Quality of Service and information collected directly on the infrastructure, providers obtain a global view about the C2U network.

In the following, we detail *two different views* provided by the proposed dashboard shown in Figs. 10a and 10b, where the blue and orange vertical bars report badness events (i.e. bad buckets) for (Azure and AWS) single paths. Differently, the bottom box with black bars shows a summary view obtained by averaging their info, i.e. performing a "decision fusion". In the same box, a dashed line is reported, representing the outcome of a "majority voting" imputation statistic. Note that this dashboard is intended to provide early hints to troubleshoot cloud networks in case of performance degradation and not to perform definitive root-cause analysis, as additional context information should be integrated to enrich the provided view [14].

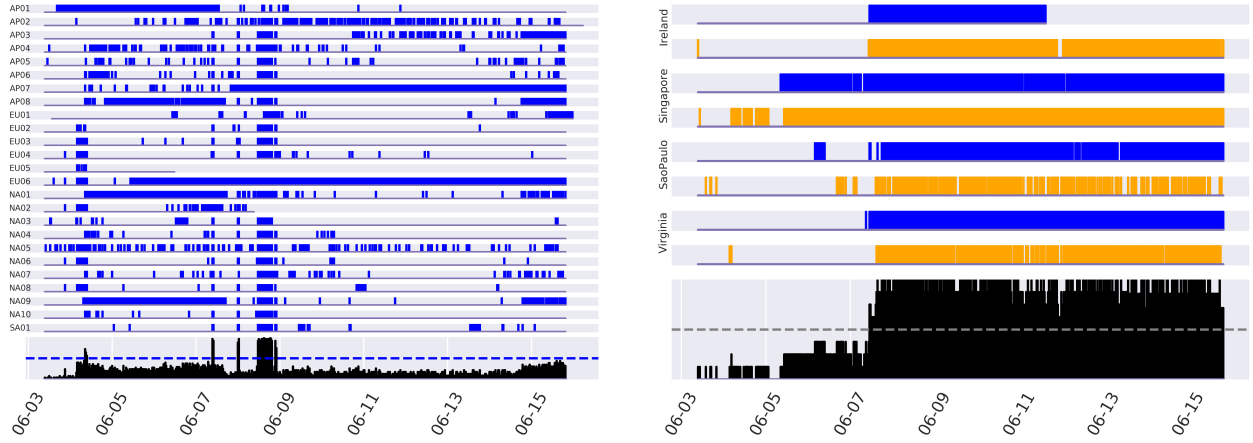
In Fig. 10a the dashboard highlights the correlation of badness events related to the `(VP, CR)` paths from all the $V_s = 25$ sources towards the Azure `Sao Paulo` CR. According to the selected view, high values (spikes) in the black box report badness events observed by (almost) all the distributed VPs (e.g. on 7–9th June) and witness degradation events that can be ascribed on the specific provider CR (e.g., possibly due to overhead on either the specific datacenter or on the related access network). On the other hand, Fig. 10b reports a VP-based view, showing the badness events of the paths from `EU06` towards the four CRs of both providers. The high (and long-lasting) values in the summary view witness degradation involving all the paths departing from this VP, i.e. for which neither provider can be blamed. That is, in this case the access network of the user is supposed to be the cause of the performance issue.

The above results show that the dashboard provides an effective real-time view onto C2U network performance as perceived from distributed VPs and towards multiple CRs and providers. Also, the effectiveness of the proposed badness (i.e. its accuracy in highlighting root causes) is expected to increase in case of capillary VP deployments (e.g., deploying VPs at higher density in specific ISPs).

Finally, the outcome provided by the dashboard (regardless the specific root cause of the performance degradation) can be leveraged to evaluate the effectiveness of performing specific actions to improve the performance, e.g., the effectiveness of multi-cloud solutions (discussed in Sec. 4.7).

4.7. Evaluating the benefits of multi-cloud deployments

Multi-cloud architectures (based on the concomitant use of services of two or more cloud providers) are increasingly adopted by enterprises, so as to exploit the flexibility deriving from multiple cloud offerings, thus achieving cost reduction and increased reliability[3]. Hereinafter we focus on the potential gains customers could achieve when adopting multi-cloud architectures in terms of *improved network performance* [10]. To this goal we evaluate the upper-bound of the C2U latency reduction w.r.t. *two baseline* cases: (*i*) the adoption of a single cloud provider for all the users (in this case we consider the adoption of the provider with better performance, on average, on a global scale, i.e. Azure according to previously shown results), denoted with L_{single}; (*ii*) the adoption of the best-performing provider on a `(VP, CR)` basis (i.e. for each `(VP, CR)` pair we consider to *statically* adopt the provider with better performance on average, based on previously-discussed results, denoted with L_{best}). These two baselines are compared to the ideal performance obtained with a multi-cloud architecture, i.e. at each instant in time the user is served by the provider



(a) Online badness events from all the 25 VPs towards Azure in `Sao Paulo` (blue lines). On July 8th, all VPs report badness events towards this CR, highlighting a possible issue in the datacenter or in its proximity.

(b) Online badness events from VP `EU05` towards the 4 CRs for both Azure (blue lines) and AWS (orange lines). Since July 8th, this VP reports badness events towards all CRs and for both providers, highlighting a possible problem in the VP or in its proximity.

Figure 10: Different views for online badness events. The last row reports the *average* of all the above bars. Dashed lines are set at 0.5, providing a visual majority-voting check.

reporting the best performance (say L_{MC}). Notably, this ideal case is representative of an architecture either (*a*) leveraging a system predicting which provider offers better performance at each time, or (*b*) duplicating the resources and properly managing the redundancy.

The results when considering the two baselines, focusing on TCP probing (port 80), are reported in Figs. 11a and 11b. The former reports for each `(VP, CR)` the relative improvement with respect to L_{single}, (i.e. $\frac{L_{\text{single}}-L_{\text{MC}}}{L_{\text{single}}} \times 100$, while the latter the relative improvement with respect to L_{best}, (i.e. $\frac{L_{\text{best}}-L_{\text{MC}}}{L_{\text{best}}} \times 100$). Our results show how multi-cloud deployments achieve better performance both when compared to the locally-better provider (performance improves more than 5% in 7% of the cases and up to 21.3%, cf. Fig. 11b) and when compared against a deployment relying on the provider performing better on a global scale (performance improves more than 5% in 29% of the cases and up to 70.8%, cf. Fig. 11a).

5. Conclusions and Future Directions

This work assessed C2U latency performance for the leading providers AWS and Azure, due to the poor visibility both customers and providers may suffer in these networks. To this goal, we used a variety of probing methods and geographically distributed VPs and CRs. Overall, these results can be helpful to both *cloud customers* serving final users (for deployment choices according to requirements and costs) and *providers* (to locate problems or bottlenecks their infrastructure could be subjected to). The adopted approach and methods led to the following *take-home messages*.

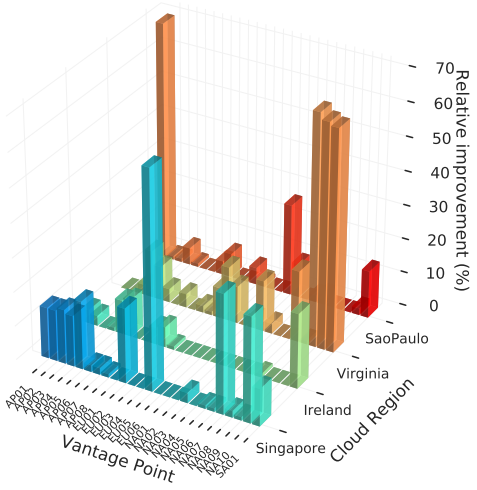
As expected, we found that globally-distributed users experience **better C2U latency for intra-region paths, on average** (with a few significant exceptions). Concerning the **latency variability**, we observed that low values are experienced in most cases, although we identified a few cases with higher variability for some `(VP, CR)` pairs. Also, analyzing at 1 min time scale the **joint (AWS-Azure) behavior** of latency distributions, we found that for 51/100 `(VP, CR)` pairs the behavior difference between the two providers could be exploited to optimize performance, paving the way to multi-cloud optimization.

Then, aiming at an in-depth characterization, we **compared the performance** of the two providers for each `(VP, CR)` pair. Best performance was achieved by AWS and Azure in 50% CR cases each, with intra- and inter-region cases giving clearer (but opposite) outcomes. Differently, analyzing variability over time, we found that usually lower latency does not imply lower variability.

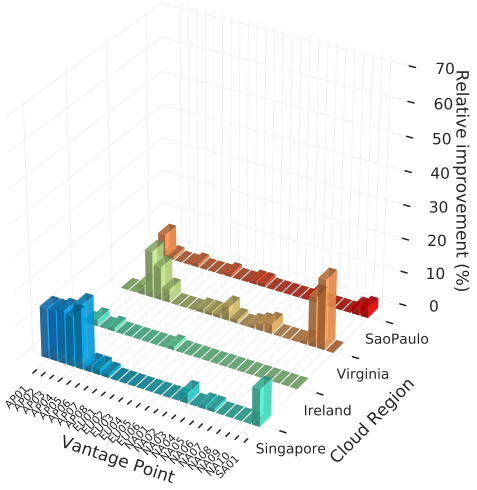
Concerning **badness analysis**, results have highlighted that the frequency of such events is quite similar for both providers (and weakly-dependent between them), but different peculiar patterns along VP- and/or CR-dimensions are present. This result shows that badness events are mostly related to either the VP or the destination area (or both), rather than to the infrastructure provider. Focusing on the persistence of badness events, such occurrences have been observed to have a decreasing trend with the duration (60% of events last for 30 min), with sporadic long events appearing for all AWS CRs and the sole Azure `Virginia`.

Concerning the adoption of **different probing methods**, our results have witnessed that, despite some discrepancies in the observed latency values, they marginally impact the outcomes of both badness analysis and provider comparison.

Finally, our experimental characterization has been exploited to support cloud network design. In detail, we implemented a prototype for the **online detection and imputation (to CRs, VPs, or provider) of badness events**, whose appeal was highlighted with an aggregated-view analysis. Also, non-negligible latency gains were guaranteed in 7% of the cases with an ideal



(a) Relative improvement over L_{single}.



(b) Relative improvement over L_{best}.

Figure 11: Gains achievable with multi-cloud deployments w.r.t. the *globally-better provider* (a) and the *locally-better provider* (b). Up to 70% (resp. 10%) relative improvement can be achieved over baseline in (a) (resp. baseline in (b)).

multi-cloud deployment, with a relative improvement up to 21.3%, also w.r.t. the (*locally-better*) single-cloud deployment.

Future works will include capitalization of collected data to develop prediction techniques, leveraging the spatial diversity of our dataset to possibly combine different probing methods and/or different VPs to enhance performance. This task enables a proactive management of the cloud infrastructure, and can optimize the probing process itself, for instance using adaptive probing techniques (e.g. via reinforcement learning) to have a less-invasive and cheaper probing process. Evaluation of edge-cloud infrastructures already deployed by the considered cloud providers from the latency perspective is also an interesting next step.

Acknowledgments

This work is partially funded by the Italian Research Program "PON AIM *Attraction and International Mobility*, Azione I.2 Linea 1, *Mobilità dei Ricercatori*" (Codice proposta attività AIM1878982-2 CUP E56C19000330005).

References

[1] E. Ataie, R. Entezari-Maleki, L. Rashidi, K. S. Trivedi, D. Ardagna, A. Movaghar, Hierarchical stochastic models for performance, availability, and power consumption analysis of iaas clouds, IEEE Transactions on Cloud Computing 7 (2019) 1039–1056. doi:10.1109/TCC.2017.2760836.

[2] M. Kwon, Z. Dou, W. Heinzelman, T. Soyata, H. Ba, J. Shi, Use of network latency profiling and redundancy for cloud server selection, in: IEEE CLOUD, 2014, pp. 826–832. doi:10.1109/CLOUD.2014.114.

[3] M. Menzel, R. Ranjan, CloudGenius: decision support for web server cloud migration, in: Proceedings of the 21st International Conference on World Wide Web, 2012, p. 979988. doi:10.1145/2187836.2187967.

[4] J. C. Mogul, L. Popa, What we talk about when we talk about cloud network performance, ACM SIGCOMM Computer Communication Review 42 (2012) 44–48. doi:10.1145/2378956.2378964.

[5] V. Persico, P. Marchetta, A. Botta, A. Pescapé, Measuring network throughput in the cloud: The case of Amazon EC2, Computer Networks 93 (2015) 408–422. doi:https://doi.org/10.1016/j.comnet.2015.09.037.

[6] A. Li, X. Yang, S. Kandula, M. Zhang, Cloudcmp: comparing public cloud providers, in: ACM IMC, 2010, pp. 1–14. doi:10.1145/1879141.1879143.

[7] V. Persico, A. Botta, P. Marchetta, A. Montieri, A. Pescapé, On the performance of the wide-area networks interconnecting public-cloud datacenters around the globe, Computer Networks 112 (2017) 67–83. doi:10.1016/j.comnet.2016.10.013.

[8] S. Choy, B. Wong, G. Simon, C. Rosenberg, The brewing storm in cloud gaming: A measurement study on cloud to end-user latency, in: IEEE/ACM NetGames, 2012, pp. 1–6. doi:10.1109/NetGames.2012.6404024.

[9] A. Ksentini, P. A. Frangoudis, P. C. Amogh, D. Nikaein, Providing low latency guarantees for slicing-ready 5G systems via two-level MAC scheduling, IEEE Network 32 (2018) 116–123. doi:10.1109/MNET.2018.1800005.

[10] Z. Wu, H. V. Madhyastha, Understanding the latency benefits of multi-cloud webservice deployments, ACM SIGCOMM Computer Communication Review 43 (2013) 13–20. doi:10.1145/2479957.2479960.

[11] F. Palumbo, G. Aceto, A. Botta, D. Ciuonzo, V. Persico, A. Pescapé, Characterizing cloud-to-user latency as perceived by AWS and Azure users spread over the globe, in: IEEE GLOBECOM, 2019, pp. 1–6. doi:10.1109/GLOBECOM38437.2019.9013343.

[12] A. C. Bavier et al., Operating systems support for planetary-scale network services, in: USENIX NSDI, 2004, pp. 19–19. doi:10.5555/1251175.1251194.

[13] O. Tomanek, P. Mulinka, L. Kencl, Multidimensional cloud latency monitoring and evaluation, Computer Networks 107 (2016) 104–120. doi:https://doi.org/10.1016/j.comnet.2016.06.011.

[14] Y. Jin, S. Renganathan, G. Ananthanarayanan, J. Jiang, V. N. Padmanabhan, M. Schroder, M. Calder, A. Krishnamurthy, Zooming in on wide-area latencies to a global cloud provider, in: ACM SIGCOMM, 2019, pp. 104–116. doi:10.1145/3341302.3342073.

[15] R. Tudoran, A. Costan, G. Antoniu, L. Bougé, A performance evaluation of Azure and Nimbus clouds for scientific applications, in: Proceedings of the 2nd International Workshop on Cloud Computing Platforms, 2012, pp. 1–6. doi:10.1145/2168697.2168701.

[16] D. Griffin, T. K. Phan, E. Maini, M. Rio, P. Simoens, On the feasibility of using current data centre infrastructure for latency-sensitive applications, IEEE Trans. Cloud Comput. (2018). doi:10.1109/TCC.2018.2822271.

[17] D. A. Popescu, A. W. Moore, A first look at data center network condition through the eyes of PTPmesh, in: IEEE TMA'18, 2018, pp. 1–8. doi:10.23919/TMA.2018.8506493.

[18] D. A. Popescu, N. Zilberman, A. W. Moore, Characterizing the impact of network latency on cloud-based applications performance, Technical Report UCAM-CL-TR-914, Univ. of Cambridge, 2017. doi:https://doi.org/10.17863/CAM.17588.

[19] D. A. Popescu, A. W. Moore, PTPmesh: Data center network latency measurements using PTP, in: IEEE MASCOTS'17, 2017, pp. 73–79. doi:10.1109/MASCOTS.2017.30.

[20] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. Maltz, Z. Liu, V. Wang, B. Pang, H. Chen, et al., Pingmesh: A large-scale system for data center network latency measurement and analysis, ACM SIGCOMM Computer Communication Review 45 (2015) 139–152. doi:10.1145/2829988.2787496.

[21] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, D. Maltz, L. Yuan, M. Zhang, B. Y. Zhao, et al., Packet-level telemetry in large datacenter networks, ACM SIGCOMM Computer Communication Review 45 (2015) 479–491. doi:10.1145/2785956.2787483.

[22] J. L. Garca-Dorado, S. G. Rao, Cost-aware multi data-center bulk transfers in the cloud from a customer-side perspective, IEEE Trans. Cloud Comput. 7 (2019) 34–47. doi:10.1109/TCC.2015.2469666.

[23] P. Mulinka, P. Casas, L. Kencl, Hi-Clust: Unsupervised analysis of cloud latency measurements through hierarchical clustering, in: IEEE CloudNet, 2018, pp. 1–7. doi:10.1109/CloudNet.2018.8549558.

[24] V. Uhlir, O. Tomanek, L. Kencl, Latency-based benchmarking of cloud service providers, in: IEEE/ACM UCC, 2016, pp. 263–268. doi:10.1145/2996890.3007870.

[25] A. A. Laghari, H. He, M. Shafiq, A. Khan, Assessing effect of cloud distance on end user's Quality of Experience (QoE), in: IEEE ICCC'16, 2016, pp. 500–505. doi:10.1109/CompComm.2016.7924751.

[26] P. Mulinka, L. Kencl, Learning from Cloud latency measurements, in: IEEE ICCW, 2015, pp. 1895–1901. doi:10.1109/ICCW.2015.7247457.

[27] I. Bermudez, S. Traverso, M. Mellia, M. Munafò, Exploring the cloud from passive measurements: The amazon AWS case, in: IEEE INFOCOM, 2013, pp. 230–234. doi:10.1109/INFCOM.2013.6566769.

[28] V. Persico, P. Marchetta, A. Botta, A. Pescapé, On network throughput variability in Microsoft Azure cloud, in: IEEE GLOBECOM'15, 2015, pp. 1–6. doi:10.1109/GLOCOM.2014.7416997.

[29] P. Ha, L. Xu, Available bandwidth estimation in public clouds, in: IEEE INFOCOM WKSHPS, 2018, pp. 238–243. doi:10.1109/INFCOMW.2018.8407010.

[30] C. Guo et al., Pingmesh: A Large-Scale System for Data Center Network Latency Measurement and Analysis, ACM SIGCOMM Computer Communication Review (2015) 139–152. doi:10.1145/2829988.2787496.

[31] C. Raiciu, M. Ionescu, D. Niculescu, Opening up black box networks with CloudTalk, in: ACM HotCloud, 2012, p. 6. doi:10.5555/2342763.2342769.

[32] Cloud Performance Benchmark, Technical Report, ThousandEyes, 2019.

[33] V. Persico, A. Montieri, A. Pescapé, On the network performance of Amazon S3 cloud-storage service, in: IEEE Cloudnet, 2016, pp. 113–118. doi:10.1109/CloudNet.2016.16.

[34] D. Chou, T. Xu, K. Veeraraghavan, A. Newell, S. Margulis, L. Xiao, P. M. Ruiz, J. Meza, K. Ha, S. Padmanabha, et al., Taiji: managing global user traffic for large-scale internet services at the edge, in: ACM SOSP'19, 2019, pp. 430–446. doi:10.1145/3341301.3359655.

[35] B. Karacali, J. M. Tracey, P. G. Crumley, C. Basso, Assessing cloud network performance, in: 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1–7. doi:10.1109/ICC.2018.8422770.

[36] Z. Hu et al., The need for end-to-end evaluation of cloud availability, in: International Conference on Passive and Active Network Measurement, 2014, pp. 119–130. doi:https://doi.org/10.1007/978-3-319-04918-2_12.

[37] Y. A. Wang, C. Huang, J. Li, K. W. Ross, Estimating the performance of hypothetical cloud service deployments: A measurement-based approach, in: IEEE INFOCOM, 2011, pp. 2372–2380. doi:10.1109/INFCOM.2011.5935057.

[38] V. Bajpai, A. Brunstrom, A. Feldmann, W. Kellerer, A. Pras, H. Schulzrinne, G. Smaragdakis, M. Wählisch, K. Wehrle, The Dagstuhl beginners guide to reproducibility for experimental networking research, ACM SIGCOMM Computer Communication Review 49 (2019) 24–30. doi:10.1145/3314212.3314217.

[39] F. Wilcoxon, S. Katti, R. A. Wilcox, Critical values and probability levels for the Wilcoxon rank sum test and the Wilcoxon signed rank test, Selected tables in mathematical statistics 1 (1970) 171–259.

[40] J. L. Gastwirth, Y. R. Gel, W. Miao, The impact of Levene's test of equality of variances on statistical theory and practice, Statistical Science (2009) 343–360.

[41] K. Lee, D. Chu, E. Cuervo, J. Kopf, Y. Degtyarev, S. Grizan, A. Wolman, J. Flinn, Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming, in: Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services, 2015, pp. 151–165. doi:10.1145/2742647.2742656.

[42] A. Ali-Eldin, J. Westin, B. Wang, P. Sharma, P. Shenoy, Spotweb: Running latency-sensitive distributed web services on transient cloud servers, in: Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing, 2019, pp. 1–12. doi:10.1145/3307681.3325397.