Understanding the Latency Benefits of Multi-Cloud Webservice Deployments

Zhe Wu and Harsha V. Madhyastha
University of California, Riverside
{zwu005,harsha}@cs.ucr.edu

ABSTRACT

To minimize user-perceived latencies, webservices are often deployed across multiple geographically distributed data centers. The premise of our work is that webservices deployed across multiple cloud infrastructure services can serve users from more data centers than that possible when using a single cloud service, and hence, offer lower latencies to users.

In this paper, we conduct a comprehensive measurement study to understand the potential latency benefits of deploying webservices across three popular cloud infrastructure services—Amazon EC2, Google Compute Engine (GCE), and Microsoft Azure. We estimate that, as compared to deployments on one of these cloud services, users in up to half the IP address prefixes can have their RTTs reduced by over 20% when a webservice is deployed across the three cloud services. When we dig deeper to understand these latency benefits, we make three significant observations. First, when webservices shift from single-cloud to multi-cloud deployments, a significant fraction of prefixes will see latency benefits simply by being served from a different data center in the same location. This is because routing inefficiencies that exist between a prefix and a nearby data center in one cloud service are absent on the path from the prefix to a nearby data center in a different cloud service. Second, despite the latency improvements that a large fraction of prefixes will perceive, users in several locations (e.g., Argentina and Israel) will continue to incur RTTs greater than 100ms even when webservices span three large-scale cloud services (EC2, GCE, and Azure). Finally, we see that harnessing the latency benefits offered by multi-cloud deployments is likely to be challenging in practice; our measurements show that the data center which offers the lowest latency to a prefix often fluctuates between different cloud services, thus necessitating replication of data.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement techniques; C.2.4 [**Communication Networks**]: Distributed Systems—*Distributed applications*; C.2.5 [**Communication Networks**]: Local and Wide-Area Networks—*Internet*

Keywords

Cloud services, Webservices, Latency

1 Introduction

The task of deploying a low-latency webservice has been simplified by the emergence of numerous cloud infrastructure services. Webservice providers can rent resources from multiple data centers in a cloud service and deploy their webservice in several geographically distributed locations. A webservice can then minimize user-perceived latencies by serving every user from the data center that has the lowest latency to that user.

However, today, webservices are predominantly deployed on a single cloud service. As a result, latencies offered by a webservice are limited by data center locations in the chosen cloud service and by Internet routing between any of those data centers and nearby users. For example, since Microsoft Azure has no data centers in South America, a webservice deployed on Azure has to serve users in Brazil from a data center in Texas, thus incurring RTTs of over 200ms. On the other hand, in the case of Amazon EC2, though users in Greece have a relatively nearby data center in Ireland, they will experience RTTs over 90ms due to circuitous routing to that data center [13].

Therefore, we argue that a webservice deployment that *spans multiple cloud services* can offer lower latencies to its clients, than that possible when using a single cloud service. This is because a multi-cloud webservice deployment has a larger set of data centers to choose from when serving its users. As a result, a webservice can take advantage of the fact that 1) one cloud service may have a data center in a particular region while another may not, or 2) even if multiple cloud services have data centers in a region, the data center in one of those services may have lower latencies to users in that region due to less circuitous routing. Going back to our example above, users in Greece and Brazil will have their latencies reduced to less than 60ms and 25ms, respectively, when a webservice is deployed across both EC2 and Azure.

Webservice deployments spanning multiple cloud services will however come at the expense of greater implementation and management complexity, and depending on the webservice's workload and data replication policy, also potentially incur higher costs than single-cloud deployments; though, note that deploying a webservice is challenging even within a single cloud service if users are served from multiple data centers. While estimating the cost associated with a webservice is best left to the webservice's administrator, in this paper, we focus on understanding the latency benefits that multi-cloud webservice deployments can potentially offer. Our goal is to enable webservice administrators to decide whether the cost and complexity of distributing their applications across multiple cloud services is worth the commensurate latency benefits.

Towards this end, we continually measure latencies for 5 weeks from 265 PlanetLab sites to three popular cloud services—Amazon EC2, Google Compute Engine (GCE), and Microsoft Azure. We use these latency measurements to estimate latencies from data centers in the three cloud services to roughly 90K IP prefixes. We find that, when a webservice is deployed across the three cloud services, users in 20–50% of prefixes will see at least a 20% reduction in RTT as compared to single-cloud deployments. When we dig deeper to understand the sources of these latency benefits, we make the following three observations.

Better latency due to better routing. Most of the prefixes that gain from multi-cloud deployments do so because one of the three cloud services has a geographically closer data center to them than the other two services. However, we also observe that a significant fraction of prefixes see latency gains because they have better routing to a nearby data center in one cloud service compared to a nearby data center in another cloud service. For example, we estimate that, when a webservice shifts from a deployment only on Azure to a deployment across EC2 and Azure, 48% of prefixes with

improved latencies are served from a EC2 data center in the same location as their closest Azure data center. This indicates that multi-cloud webservice deployments will be able to offer lower latencies to users even if, in the future, all cloud services were to potentially have data centers in the same locations.

Not everyone wins. Despite the significant latency benefits that multi-cloud webservice deployments can offer, our analysis suggests that users in several countries will continue to suffer RTTs greater than 100ms. The primary causes for high latency to a particular region even with a multi-cloud deployment are: (a) all three cloud services do not have a geographically nearby data center to that region (e.g., Israel), or (b) only one of the three cloud services has a data center near to that region and circuitous routing to that data center inflates latencies (e.g., Argentina). Webservice providers have no control over improving latencies in either case.

Need for data replication. Lastly, we find that, for a large fraction of prefixes, there is no single data center that always yields the lowest latency. Instead, the lowest latency data center for a prefix often fluctuates between two different cloud services. We see that the cause for such fluctuation can be diurnal latency variations in some cases and more long-term variations in other cases. We estimate that, due to these latency variations, storing every user's data in a single data center, as opposed to in the two closest data centers, can inflate the 90^{th} percentile latency by 20% for users in roughly 20% of prefixes.

Implications. Our work has implications for both webservice administrators and cloud service providers. Taking into account the geographic distribution of its users and the characteristics of its workload, a webservice's administrator can use our findings to determine whether she should consider multi-cloud deployments. Our results also show that replication of data across data centers is critical to minimizing user-perceived latencies. For cloud service providers, our study highlights what they need to do in order to compete with other cloud services. The fact that they need to deploy data centers in regions where others have a data center, but they currently lack one, is relatively obvious. However, our work also highlights a subtler issue that cloud services need to pay attention to—we identify the regions in which they need to work with ISPs to improve the Internet's routing between their data centers and nearby users.

2 Background and setting

We consider three popular cloud services in our study—Amazon EC2, Microsoft Azure, and Google Compute Engine (GCE). We provide an overview of these cloud services and describe our envisioned deployment of webservices across these cloud services.

Cloud services. EC2, Azure, and GCE operate on an Infrastructure-as-a-Service (IaaS) model. In each service, customers can rent virtual machines in different data centers, which are referred to as regions. In our study, we consider 8 regions in EC2, 8 regions in Azure, and 3 regions in GCE. Since GCE is still in its infancy and has regions only in the US, we consider latencies to Google App Engine (GAE) as a proxy for latencies to GCE. Though GAE, unlike GCE, operates on a Platform-as-a-Service (PaaS) model, latency measurements to GAE enable us to estimate latencies that GCE will be able to offer once it is available on all of Google's data centers. While Google is known to have data centers in 13 locations [1], which subset of these are used for GAE is unknown.

System setting. Figure 1 shows how we envision webservices being deployed across multiple cloud services. First, a webservice needs to select a set of cloud service regions, i.e., data centers, on which to deploy the application and rent an appropriate number of virtual machines in each data center. The webservice also needs to

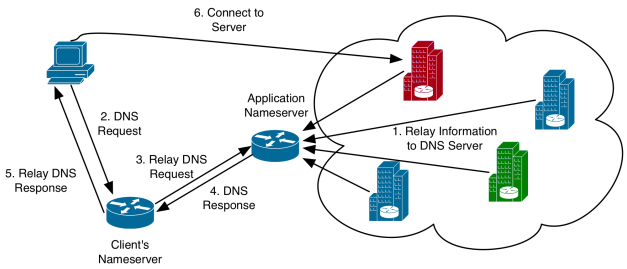
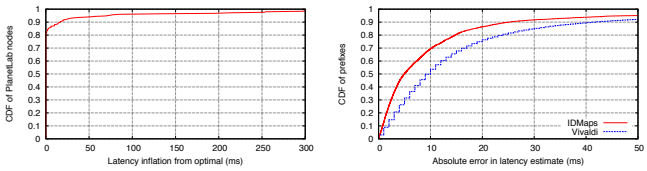


Figure 1: **DNS-based redirection architecture of a distributed webservice. Different data center colors represent regions in different cloud services.**



(a) (b)

Figure 2: **(a) Comparison of latency compared to optimal when every PlanetLab node uses the optimal redirection policy of its closest intermediate node. (b) Error in latency estimates to closest EC2 region when using latencies through the closest intermediate node and with a coordinate-based approach.**

set up a set of nameservers which implement the application's redirection policy. In this paper, we consider latency-sensitive webservices, which redirect clients to the lowest latency region. To do so, a webservice provider needs to measure or estimate latencies from each region to every IP address prefix, and then redirect users in a prefix to that region which has the lowest latency to that prefix; webservices typically group clients into IP prefixes and redirect all clients in the same prefix to the same data center. Thereafter, whenever any client attempts to resolve the URL for the webservice, the nameserver that receives the client's DNS request returns the IP address of a virtual machine in the region chosen for that client's prefix. When the client receives this DNS response, it contacts the IP address included in the response to access the webservice.

Note that this design of measuring or estimating latencies from every region to every prefix and then using DNS to redirect every client to the lowest latency region is necessary even when deploying a webservice across multiple regions in a single cloud service. Therefore, apart from the use of a greater number of regions, the additional complexity associated with multi-cloud deployments stems from (a) having to pick the appropriate virtual machine type in each cloud service, and (b) ensuring the portability of the application's implementation on different virtual machine types.

3 Dataset

Next, we describe the measurement dataset that we gathered and the processing we performed on it for use in our analysis.

Measurements. First, we issued HTTP HEAD requests using `curl` to 5 popular websites hosted on GAE for the period of a day. We issued these requests from 265 PlanetLab sites and logged the IP addresses from which GAE served our HEAD requests.

Then, over a 5 week period, we measured latencies every 5 minutes to every GAE IP and to every EC2 and Azure region, again from 265 PlanetLab sites. For EC2 and Azure, we issued HTTP HEAD requests to a webserver that we deployed in each of their regions, and use the TCP connection setup latency on every request (the *time_connect* value reported by `curl`) as a proxy for RTT. In the case of GAE, we issued ICMP pings to all the GAE IPs to measure latencies. We measure latencies to GAE with pings, rather than via HEAD requests, because GAE (being a PaaS platform)

controls the redirection of clients to Google's data centers. We observed several instances wherein the latency to GAE measured from a PlanetLab node was significantly higher than the lowest latency measured from that node to all of GAE's IP addresses. We believe that this is a result of GAE redirecting clients to farther regions in some cases. Therefore, ping-based measurements to GAE help us estimate the latencies that GCE would offer when every client is served from the closest Google data center.

Note that, though the source of our latency measurements differ across Azure, EC2, and GCE (curl versus ping), we measure latencies to EC2 using both `curl` and ping and confirm that RTT measurements made using either method are largely identical.

Estimating latencies to end-hosts. To evaluate the latency benefits that multi-cloud webservice deployments can offer, we first need as input user-perceived latencies both in the case of single cloud and multi-cloud deployments. However, neither do we control a sufficiently large number of end-hosts to measure latencies from them to the cloud services, nor can we measure latencies by issuing pings to end-hosts from GAE and Azure; as mentioned above, we use GAE as a proxy for GCE and since GAE operates a PaaS model, it runs an application's code only to serve HTTP requests, whereas Azure does not permit outgoing/incoming pings from its data centers [2]. Therefore, given the inability to directly measure, we need to *estimate* latencies between arbitrary end-hosts and the regions in these cloud services. We do so by combining latency measurements a) between PlanetLab nodes and cloud services, and b) between PlanetLab nodes and end-hosts, as follows.

We adopt a simple strategy along the lines of that used in IDMaps [10]. From each of the 265 PlanetLab nodes, we measure latencies to over 110K prefixes in which we find responsive IP addresses; these 110K prefixes account for 85% of the end-host prefixes on the Internet [12]. We find that 80% of the 110K prefixes are within 30ms of a PlanetLab node (which roughly corresponds to a distance of 2000 miles); we prune the remaining 20%, which removes most of the prefixes in Africa from consideration since there are no PlanetLab nodes in Africa.

For every prefix, we then use the following two steps to estimate the latency that users in the prefix would incur given a particular deployment of a webservice. For example, for a single cloud deployment on EC2, users in a prefix can be redirected to any one of EC2's regions. Whereas, in the case of a deployment spanning all three cloud services, the redirection options are all of GCE's regions, all of EC2's regions, and all of Azure's regions. In any particular deployment, we consider the latency for a particular prefix as that obtained with the optimal redirection policy, in which users in the prefix are served from that redirection option which offers them the lowest latency.

First, in *Step1*, we consider the PlanetLab node to which a prefix has the lowest latency as the *intermediate node* for that prefix, and consider the optimal redirection policy for that prefix to be the same as that for its intermediate node. In other words, in every measurement round in our dataset, we estimate that the redirection option that will offer the lowest latency to a particular prefix is the same as that which has the lowest latency to the prefix's intermediate node. Thereafter, in *Step2*, in any particular round, we estimate the latency from a prefix to its optimal redirection choice in that round as the sum of latencies a) from the prefix to its intermediate node, and b) from the intermediate node to that redirection choice.

We validate the two steps in our latency estimation technique as follows. First, to validate *Step1*, we consider each PlanetLab node in turn and choose an intermediate node for it among all other PlanetLab nodes. We then compare the median latency (across rounds) for that PlanetLab node a) when it is always served from its optimal redirection choice in every round, and b) when it is served in every round from the redirection option that is optimal for its intermediate node. Figure 2(a) shows that the difference between these values is less than 15ms for 90% of PlanetLab nodes. Thus, the minimum latency redirection option for a node is almost always identical to that for its intermediate node.

Second, to validate *Step2*, we pick 10K prefixes at random and measure latencies to them from the seven EC2 regions; we cannot gather such measurements from Azure or GAE. To make these measurements compatible with our HTTP HEAD-based latency measurements between PlanetLab nodes and cloud services, we ping these 10K prefixes every 5 minutes from every PlanetLab node and every EC2 region; these measurements require only 10.6 Kbps of probe traffic per PlanetLab node/EC2 region. For each of the 10K prefixes, we identify the closest EC2 region and estimate the latency from the prefix to that region as the sum of latencies from the prefix to its intermediate PlanetLab node and from that intermediate node to the EC2 region. Figure 2(b) plots the distribution across prefixes of the difference between measured and estimated latencies. We see that the error in latency estimate is less than 15ms for over 80% of prefixes.

Note that the paths on which we validate our latency measurements—between EC2's regions and end-hosts in arbitrary prefixes—do not pass through any academic or research networks. We are able to accurately estimate the latencies along these paths despite the fact that most PlanetLab nodes are in academic institutions. This is because only the paths between PlanetLab nodes traverse academic and research networks. The paths used in our latency estimation—between PlanetLab nodes and cloud service regions and between PlanetLab nodes and arbitrary prefixes—traverse the same commercial ISPs seen on any arbitrary route through the Internet.

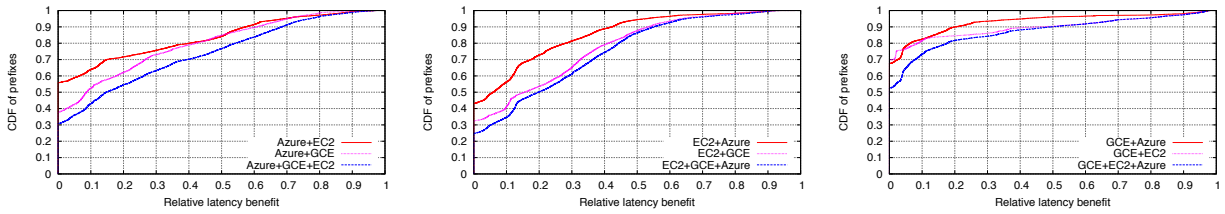
We also considered the use of Vivaldi [9], a state-of-the-art network coordinate system, for estimating latencies. We provided the latencies we measured from PlanetLab nodes to cloud service regions and to the 10K prefixes as input to Vivaldi. However, as seen in Figure 2(b), our IDMaps-based approach outperforms Vivaldi's estimation. We believe that this is due to the relatively widespread distribution of PlanetLab nodes across the globe.

4 Latency benefits of multi-cloud deployments

In this section, we first examine the potential latency benefits of webservice deployments spanning multiple cloud services. We then identify the regions in which users have the most to gain from multi-cloud deployments and the causes for these gains.

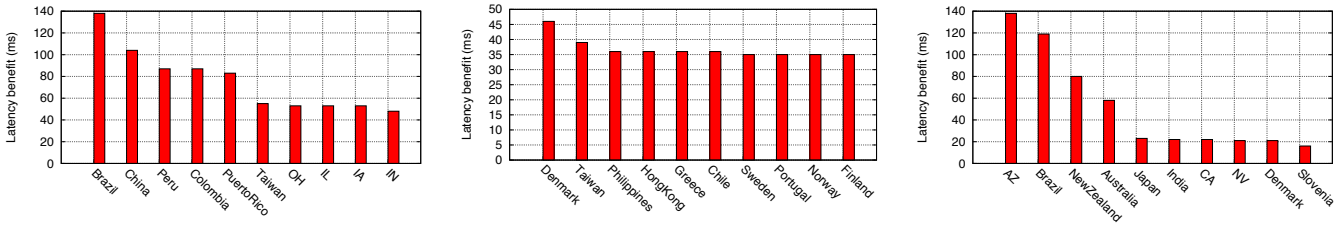
Latency benefits. To estimate latency benefits, we first compute the latency that every prefix would experience in seven scenarios for deploying a webservice: only on EC2, only on Azure, only on GCE, combination of any two of the three cloud services, and across all three cloud services. In each scenario, for every 5 minute measurement round in our dataset, we estimate the latency for every prefix as outlined in the previous section. We then divide our measurement dataset into 5 partitions—one for every week—and in each week, we compute for every prefix the median latency it experiences across all measurement rounds. Since all of our findings were largely identical across weeks, we present results from the middle of the 5 weeks.

Figures 3(a), 3(b), and 3(c) show the estimated latency benefits that multi-cloud webservice deployments will offer, as compared to deployments solely on Azure, EC2, or GCE; relative latency benefit is the latency benefit offered by a multi-cloud deployment as a fraction of the absolute latency seen with a single cloud deployment. We see that multi-cloud deployments can offer significant latency gains, especially as compared to deployments solely on



(a) (b) (c)

Figure 3: **Relative improvement in RTTs offered by multi-cloud deployments, as compared to deployments only on (a) Azure, (b) EC2, or (c) GCE.**



(a) (b) (c)

Figure 4: **Median latency improvements expected in top 10 regions in which users benefit the most when webservices are deployed across all cloud services as opposed to solely on (a) Azure, (b) EC2, or (c) GCE.**

Azure or on EC2. Our latency estimates show that, as compared to single-cloud deployments, latencies to 20–50% of prefixes can be reduced by over 20% by having webservices span all three cloud services. A 20% latency gain is significant because even the simple operation of loading a single web page requires several RTTs of communication between a client and the webserver [7].

In addition, in all three single-cloud deployments, we see that combining two cloud services yields most of the latency gains that multi-cloud deployments can offer. Expanding deployments from two to three cloud services only marginally improves the latency benefits. In the case of Azure-only, EC2-only, and GCE-only deployments, expanding the deployment to GCE in the first two cases and to EC2 in the third case accounts for most of the increased geographical diversity and the routing inefficiency fixes that multi-cloud deployments have to offer.

Note that, though the latency gains seen in Figure 3 may be affected by errors in our latency estimates, we observe similar (even slightly higher) gains when considering only PlanetLab nodes as clients (not shown). Since we have measured latencies from all PlanetLab nodes to all cloud services, in this case, we can compute the latency benefits offered by multi-cloud deployments without relying on estimated latencies.

Biggest gainers. Since the latency benefits offered by multi-cloud deployments significantly vary across prefixes, we next identify the specific regions in which end-hosts have the most to gain if webservices were spread across cloud services. To do so, we determine the region (at the granularity of state within the US, and at the granularity of country elsewhere) for every prefix in our dataset by combining information from MaxMind [4] and IP2Location [3]; we prune any prefix for which we find that the measured latency to the prefix violates speed-of-light constraints.

Figure 4 shows the top 10 regions that would experience the highest latency improvements when webservices shift from single-cloud deployments to deployments spanning EC2, Azure, and GCE. Here, we compute the latency gain for a prefix as the difference in latency estimates for that prefix when using one cloud service and when using all three cloud services. In every region, we compute the median of this latency gain across all prefixes in that region. We see that the biggest beneficiaries have latency improvements of over 100ms in the case of GCE and Azure, while several regions see more than 30ms reduction in latency in the case of EC2. As mentioned before, we consider these latency gains to be significant since the simple operation of loading a web page can involve tens of RTTs of interaction between a client and a server [7].

Reasons for latency gains. As expected, the most common reason for reduced latency when using multiple cloud services is that one cloud service has a data center in a particular region while others do not. For example, users in Brazil would see latency improvements of over 100ms for webservices currently deployed on GCE or Azure. This is because EC2 has a data center in Brazil, whereas GCE and Azure do not. Similarly, for webservices currently deployed only on EC2, users in Hong Kong would have significant latency gains because Google has a data center in Hong Kong.

However, we also find that a significant fraction of latency gains are due to the ability of multi-cloud deployments to correct for routing inefficiencies. Due to inefficient routing, several regions have high latencies to a particular cloud service in spite of the presence of a nearby data center in that cloud service. Multi-cloud webservice deployments can improve latencies in such cases because routing from the same region may be more efficient to nearby data centers in other cloud services.

For example, even though EC2 has a data center in Ireland, the latency to this data center from the PlanetLab nodes in Greece is over 90ms. We suspect that this is due to circuitous routing through the Internet [13] from Greece to the EC2 data center in Ireland. In contrast, Azure too has a data center in Ireland and the latency to it from Greece is below 60ms. As a result, we expect a latency improvement of over 30ms for users in Greece when webservices span EC2 and Azure. Similarly, we estimate that users in Taiwan can expect latency gains of over 50ms from webservices currently deployed only on Azure, because routing is circuitous from Taiwan to Azure's data center in Hong Kong but not to Google's data center in Taiwan.

To quantify the fraction of latency gains offered by multi-cloud deployments that can be attributed to routing inefficiencies, we perform the following analysis. We compare a deployment only on Azure with a deployment spanning EC2 and Azure; we exclude GCE from this analysis since we do not know the locations of Google's data centers. In this case, we consider the prefixes for whom the optimal redirection option in a EC2+Azure deployment

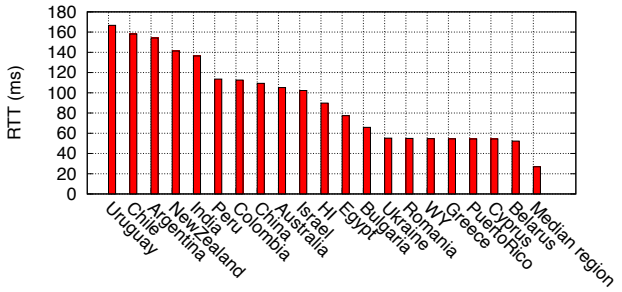
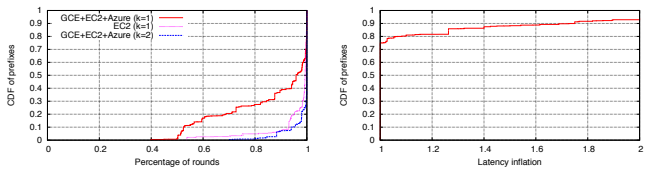


Figure 5: The 20 worst regions with respect to the latencies from webservice deployments spanning EC2, GCE, and Azure.



(a) (b)

Figure 6: (a) Fraction of rounds accounted for by top k redirection options with EC2-only and GCE+EC2+Azure deployments, and (b) overhead in 90^{th} percentile latency when storing every user's data only in the closest data center.

is a data center in EC2, i.e., the data center to which they are redirected in an EC2+Azure deployment differs from that in the Azure-only case. We find that, for 48% of these prefixes that see latency gains by combining EC2 and Azure, the EC2 data center to which they are redirected in a EC2+Azure deployment is in the same region as the Azure data center they would be redirected to in an Azure-only deployment. Furthermore, prefixes that get redirected to an EC2 data center in the same region as their lowest latency Azure data center account for 37% of all prefixes that see at least a 20% latency reduction from combining EC2 and Azure.

These results have the following implication. In the future, competition between these major cloud services may potentially drive all of them to have data centers in the same set of locations. If this were the case, there would be no latency gains due to the first reason we identify above—one cloud service has a data center in a certain location, whereas others do not. However, due to the significant latency gains obtained by correcting for sub-optimal routing, we expect that webservices can use multi-cloud deployments to reduce latencies for a significant fraction of their users even if all cloud services had co-located data centers.

5 Challenges in harnessing latency benefits

Having focused thus far on the positives of deploying webservices across multiple cloud services, we next highlight a couple of challenges in reducing user-perceived latencies in this manner.

Lack of control to improve poor latencies. Though we expect significant latency benefits for users in several regions, we find that several other regions that experience high latencies with single-cloud webservice deployments will continue to suffer from the same problem even when webservices span EC2, Azure, and GCE. Figure 5 shows that the worst 20 regions (ranked based on the estimated latencies from EC2+Azure+GCE deployments) will all experience latencies over 50ms [1]; in comparison, the latency for the median region is less than 25ms.

[1]Note that EC2 recently announced a new region in Sydney. So, we expect that the latencies to Australia and New Zealand would decrease.

The most common reason for this lack of latency improvement is that the closest data center to a region is distant, even after combining cloud services. Chile, India, and Israel are examples of such regions. In other regions, in spite of the presence of a nearby data center in one of the cloud services, circuitous Internet routing to that data center causes high latencies to those regions even when combining cloud services. For example, though EC2 has a data center in Brazil, the latency to it is over 100ms from the PlanetLab node in Argentina. Since neither Azure nor GCE has a nearby data center to correct this routing inefficiency, we see that users in Argentina will experience high latencies irrespective of whether webservices use one or multiple cloud services.

Thus, in several areas, the latencies that webservices can offer to their users are limited until either 1) one of the cloud services sets up a new data center nearby, or 2) ISPs and cloud services work together to reduce the inefficiencies of Internet routing. Figure 5 shows that these problems affect users in several regions—South America, Middle East, Europe, and some parts of Asia. Webservice providers have limited control over addressing these issues.

Latency fluctuation. In the optimal redirection policy that we have considered so far, in each measurement round in our dataset, a webservice serves users in a prefix from that redirection option which has the lowest latency in that round to the prefix. For every prefix, we compute the fraction of measurement rounds for which it was served from each redirection option and sort the redirection options based on this fraction. Figure 6(a) shows the distribution across prefixes of the fraction of rounds for which each prefix was served from its top k redirection options. From the line *"GCE+EC2+Azure (k=1)"*, we see that though most prefixes were assigned to one redirection option—the region closest to them—for almost all measurement rounds, this is not the case for a large fraction of prefixes. For 28% of prefixes, their most common redirection option accounted for less than 80% of the measurement rounds.

This indicates that, due to the churn in the Internet's routes and latencies, it is often the case that the optimal redirection choice for a prefix changes across rounds. The line for *"GCE+EC2+Azure (k=2)"* shows that the two most common redirection options account for over 95% of measurement rounds for 90% of prefixes, which is similar to the distribution in the EC2-only case with $k = 1$. Thus, the fluctuation of the optimal redirection choice across two data centers is specific to multi-cloud deployments; indeed, in the multi-cloud case, the two most common redirection options are in different cloud services for 96% of prefixes. This is to be expected since data centers within a cloud service are spread across the globe, but there can be data centers from multiple cloud services in the same region.

We find that these fluctuations in the lowest latency redirection option for a prefix are due to three main reasons. First, for many prefixes, latencies to the best and second best redirection options are largely identical and minor latency variations can alter the option that yields the lowest latency in any particular round; Figure 7(a) shows an example. On the other hand, there are several cloud service regions to which we see a distinct diurnal variation in latencies; this pattern is particularly dominant for Google's data centers in Europe as seen in Figure 7(b). In such cases, the lowest latency redirection option varies based on the time of day. Finally, we also observe cases wherein we see more long-term variations in latencies, e.g., Figure 7(c). We believe that such cases are due to changes in routing configurations that either introduce or fix circuitous routing. Of the prefixes for which the best redirection option accounts for less than 80% of the measurement rounds in our dataset, we find that 32%, 46%, and 22% of the prefixes, respectively, can be attributed to the three above-mentioned reasons.

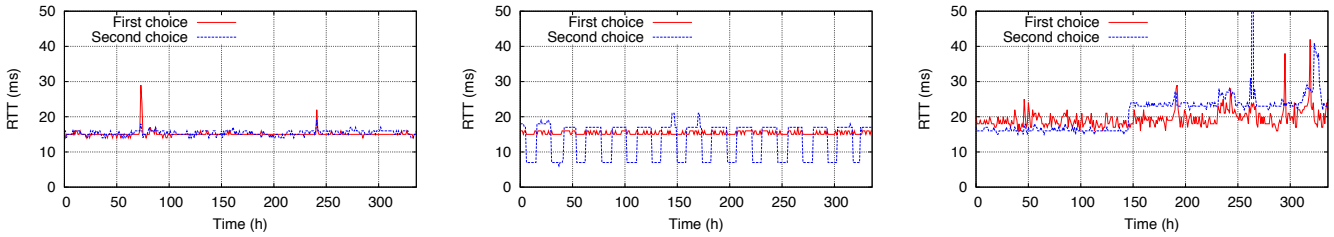


Figure 7: **Examples of latency fluctuations that result in the lowest latency data center to a prefix changing over time.**

Our observation that the cloud service to which a prefix is redirected often changes over time has the following implication for the design of webservices. Consider a user who may be served from data center A at some times and from another data center B at other times. The simplest design for a webservice would be to store data uploaded by the user in one of these data centers, say A, and thus avoid the complications of replicating data. However, as we see in Figure 6(b), if every user's data were stored only in the data center closest to the user, the 90^{th} percentile latency experienced by that user will increase by 20% for users in 20% of prefixes. This is because, when the user is redirected to the alternate data center B, the webservice incurs the overhead of fetching the user's data from A. Thus, to obtain the optimal latency benefits offered by multi-cloud deployments, it is critical that webservices replicate a user's data asynchronously between the two data centers—typically in different cloud services—which offer the lowest latencies to the user.

6 Related work

Evaluating benefits of cloud deployments. With the explosive growth of cloud computing, there has been some work recently to answer the question—when to use cloud services? Answers to this question have largely been restricted to examining how and when to migrate applications from the application provider's data center to a cloud service [11, 15, 16]. In this paper, we examine the benefits that application providers can obtain by taking the use of cloud services to the next level—spreading webservices across multiple cloud infrastructure services.

Choosing among cloud services. Recently, there have been some efforts at measuring and characterizing the performance offered by various cloud services [14, 5]. The focus in these efforts has been either to enable application providers to choose between cloud services (with the application being deployed on one of them) or to monitor an application deployed on a cloud service. The focus of our study instead is to understand the latency benefits of deploying webservices *across multiple cloud services*.

Using multiple cloud services. RACS [6] proposes replicating data across multiple cloud providers to obtain similar benefits as RAID disk arrays. AppScale [8] focuses on enabling portability of applications across cloud services. In contrast to AppScale, we envision application deployments spanning multiple cloud services, and in contrast to RACS, we explore the latency benefits from multi-cloud deployments.

Trading off cost and performance. Zhang et al. [17] showed how a service provider can trade off cost versus performance by jointly optimizing network routing and client redirection. In contrast, webservices deployed on cloud services have little control over Internet routing. Moreover, since costs of running a webservice are crucially dependent on its workload and implementation architecture, the computation of costs are best left to webservice administrators. Our goal instead is to evaluate the potential latency gains of multi-cloud deployments, narrow down on the regions in which users would benefit the most, and highlight the challenges in harnessing these latency benefits.

7 Conclusions

To the best of our knowledge, we are the first to recognize the opportunity for aggressively minimizing user-perceived latencies by deploying webservices across multiple cloud services. With the aid of measurements over 5 weeks from 265 PlanetLab sites to three popular cloud services, we demonstrated that webservices that span multiple cloud services can reduce latencies by over 20% for users in up to 50% of prefixes. Furthermore, our analysis highlighted the regions in which users have the most to gain. Finally, we showed that users in several regions will experience high latencies even if webservices take advantage of multiple cloud services, and that multi-cloud deployments will necessarily have to replicate data to optimize user-perceived latencies.

8 References

[1] Google data centers locations. `http://www.google.com/about/datacenters/locations/index.html`.

[2] ICMP, outbound ping on Azure VM. `http://social.msdn.microsoft.com/Forums/en-US/WAVirtualMachinesforWindows/thread/e9e53e84-a978-46f5-a657-f31da7e4bbe1`.

[3] IP2Location. `http://www.ip2location.com/`.

[4] MaxMind. `http://www.maxmind.com/`.

[5] VMware vFabric Hyperic. `http://www.vmware.com/products/datacenter-virtualization/vfabric-hyperic/`.

[6] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon. RACS: A case for cloud storage diversity. In *SOCC*, 2010.

[7] M. Al-Fares, K. Elmeleegy, B. Reed, and I. Gashinsky. Overclocking the Yahoo! CDN for faster web page loads. In *IMC*, 2011.

[8] N. Chohan, C. Bunch, S. Pang, C. Krintz, N. Mostafa, S. Soman, and R. Wolski. AppScale: Scalable and open AppEngine application development and deployment. In *CloudComp*, 2009.

[9] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A decentralized network coordinate system. 2004.

[10] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. IDMaps: A global Internet host distance estimation service. *IEEE/ACM Transactions on Networking*, 2001.

[11] M. Hajjat, X. Sun, Y.-W. E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani. Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud. In *SIGCOMM*, 2010.

[12] E. Katz-Bassett, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson. Studying black holes in the Internet with Hubble. In *NSDI*, 2008.

[13] R. Krishnan, H. V. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao. Moving beyond end-to-end path information to optimize CDN performance. In *IMC*, 2009.

[14] A. Li, X. Yang, S. Kandula, and M. Zhang. CloudCmp: Comparing public cloud providers. In *IMC*, 2010.

[15] B. C. Tak, B. Urgaonkar, and A. Sivasubramaniam. To move or not to move: The economics of cloud computing. In *HotCloud*, 2011.

[16] T. Wood, E. Cecchet, K. Ramakrishnan, P. Shenoy, J. van der Merwe, and A. Venkataramani. Disaster recovery as a cloud service: Economic benefits & deployment challenges. In *HotCloud*, 2010.

[17] Z. Zhang, M. Zhang, A. Greenberg, Y. C. Hu, R. Mahajan, and B. Christian. Optimizing cost and performance in online service provider networks. In *NSDI*, 2010.