Mitigating Routing Inefficiencies to Cloud-Storage Providers: A Case Study

Soham Sinha*, Di Niu†, Zhi Wang‡, Paul Lu*
*Department of Computing Science, University of Alberta, Canada
{soham, paullu}@ualberta.ca
†Department of Electrical
and Computer Engineering, University of Alberta, Canada
dniu@ualberta.ca
‡Graduate School at Shenzhen, Tsinghua University
wangzhi@sz.tsinghua.edu.cn

***Abstract*—We provide a case study of current inefficiencies in how traffic to well-known cloud-storage providers (e.g., Dropbox, Google Drive, Microsoft OneDrive) can vary significantly in throughput (e.g., a factor of 5 or more) depending on the location of the source and sink of the data. Our case study supplements previous work on resilient overlay networks (RON) and other ideas.**

These inefficiencies exist in the presence of vendor-specific points-of-presence (POP), which try to provide better network performance to the clients. In fact, the existence of special-purpose networks (e.g., national research networks, PlanetLab) and complicated peering relationships between networks, means that performance problems *might* exist in many wide-area networks (WANs).

Our main contribution is to continue the cataloging of network inefficiencies so that practitioners and experimenters are aware of them. But, we also show how simple *routing detours*, can improve throughput by factors of over 3x for client-to-cloud-storage. Although the specific inefficiencies in this paper might be transitory (and we agree with that characterization), WAN bottlenecks due to routing, suboptimal middlebox configuration, and congestion persist as real problems to be cataloged, discussed, and addressed through the use of detours, or data transfer nodes (DTNs), or RONs.

I. Introduction

Non-optimal and inefficient routing on the Internet is known to exist. But with networks evolving over time (e.g., special research networks, like Canada's CANARIE[1]) and different traffic types emerging (e.g., Web vs. video streams vs. cloud-storage data), it is useful to revisit and catalog the known network issues that, in the past, have motivated ideas such as resilient overlay networks (RONs) [1], and data transfer nodes (DTN) [2].

For example, many cloud-storage providers have multiple points-of-presence (POPs) across, say, the United States of America to improve throughput for their clients. However, we found and document how the effective throughput to popular cloud-storage providers can still be far from optimal. For example (Fig. 2), uploading a 100 MB binary file from a University of British Columbia (UBC) PlanetLab node to Google Drive (using Google's application programming interface (API)) takes 87 seconds (s). The same file transferred from a non-PlanetLab node at the University of Alberta (UAlberta) to Google Drive takes 17s. Furthermore, transferring the file from the UBC PlanetLab node to the UAlberta non-PlanetLab node takes 19s, over the CANARIE research network. Together, by using UAlberta as a part of a detour from UBC to Google Drive, the 100 MB file can be transferred in 36s (= 17+19) instead of 87s.

On the one hand, transferring the data from UBC to Google Drive via UAlberta has extra overheads because of the involvement of an additional node. It is also counter-intuitive given the geographic backtracking involved (Fig. 3), and it may just be an unintended artifact of how the PlanetLab testbed is configured, combined with the presence of a high-performance research network between UBC and UAlberta (i.e., CANARIE). On the other hand, the performance benefits are repeatable, and we have found similar inefficiencies elsewhere (Sec. III). We certainly agree that the specific routing between PlanetLab and the world, and the complexities of network peering may have created this transitory problem. However, the emergence of design patterns such as Science DMZ [2] and overlay networks [1], [3] suggest that it may be necessary to explicitly identify and mitigate such inefficiencies for some workloads. The cloud-storage providers create POPs to improve performance, but explicit routing through detours and overlays may still be necessary.

The main contribution of this paper is to catalog and provide a case study of these performance problems. In the past several years, cloud-based storage has become popular due to the convenience, the ability to easily share files between sites and people, and the high availability promised by the cloud-storage providers. However, the speed of uploading and downloading to these cloud-storage providers vary depending on the location of the clients [4]. Therefore, no matter how transitory these performance problems might be, they can have a real impact on many users.

Also, we contribute a simple solution to the performance problem in the form of a *routing detour*, adding an intermediate data transfer node (DTN) to explicitly route the data along a faster overlay network. As a proof of concept, our

[1]http://www.canarie.ca/

implementation of routing detours is shown to improve net transfer-time performance by factors of 3 or more, depending on file size, source of the data, and cloud-storage provider. Admittedly, the scale of the improvement says more about the performance problem than it does about the importance of our contribution. However, we note that routing detours fall within a set of simple ideas that have nonetheless made it into practice. For example, Science DMZ's concept of a DTN improves performance by (in part) bypassing firewall bottlenecks, which are normally configured for non-bulk data transfers [2]. And, overlay networks were conceived (in part) to explicitly control routing on an Internet with decentralized control. Our routing detours are a similar attempt to bypass bottlenecks and explicitly control routing.

In the medium term, the identification of these inefficiencies may encourage cloud-storage providers to add additional POPs or gateways to their internal networks. Universities and institutions with the appropriate means can provide routing detours to use research networks or vendor POPs more effectively, without having to convince external parties (such as Internet Service Providers) to change their routing configuration. And, finally, our group plans to expand the functionality of our routing detours to deal with firewall bottlenecks (like Science DMZ) and to monitor and bypass dynamic bottlenecks on the WAN, as future work.

II. Experimental Set-up

In this section, we describe the design of our idea and experiments. Before going into details of our own system, we provide an overview of the cloud-storage APIs. We have chosen three popular cloud-storage services for our experimentation: Dropbox, Google Drive, Microsoft OneDrive. They all use the OAuth2 [5] authorization standard. They all have similar mechanisms for file transfers, which is the RESTful architecture [6].

Using the RESTful APIs, developers can access certain web-services from the cloud-storage providers. Programming languages which support HTTP requests can be used to access these APIs. For our experiments, we have used Java as our programming language and used the official libraries released by Google and Dropbox. For OneDrive, we used and modified an open-source Java library[2] as OneDrive does not have any official Java client library yet. For our experiments, we focus on the file-transfer operations in the API libraries, i.e., uploading a file and downloading a file from the server.

Traditionally, users utilize the official client applications of different cloud-storage providers to upload/download their files to/from the cloud servers from/to their local machines. These clients internally use cloud-storage APIs, described above, to connect to the cloud-storage servers. On the other hand, application developers directly use cloud-storage APIs to connect to the cloud-storage servers for their own applications. Therefore, these storage APIs are the underlying layer of connections between the clients and the cloud-storage servers.

[2]https://github.com/tjeerdnet/OneDriveAPI

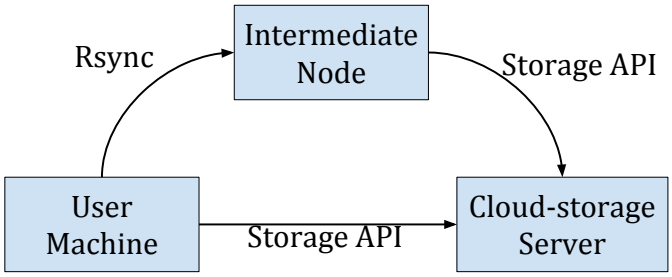


Fig. 1: Schematic Diagram of our experimental design for uploading data to cloud storage

Thus, we selected these APIs to measure the performance of personal cloud-storage services.

Now, we explain our idea on how we propose to transfer files faster for cloud-storages. The idea has been presented schematically in Figure 1. In the figure, we can see the *User Machine* which is the client machine or the source machine from/to where we need to upload/download files to/from the cloud-storage servers. Cloud-storage APIs can directly be used for the file-transfers as described earlier and shown in the figure by the horizontal line between *User Machine* and *Cloud-Storage Server*. However, we have observed that uploading directly to cloud-storages are sometimes slow. This can happen due to various reasons, such as low bandwidth, routing inefficiencies and other unknown bottlenecks. As we explain in our experimental results, the routing inefficiencies can cause significant performance degradation. In those cases, an *Intermediate Node* becomes crucial. An *Intermediate Node* can help a *User Machine* in different ways, for example, by caching or by its higher data-transfer speed. We have observed in our experiments that if we upload a file via certain intermediate node(s), then we can achieve higher throughput (lower time to transfer files) than uploading via a direct route to the cloud-storages. To upload files via the detour routes, we use `rsync`[3] to transfer files between a *User Machine* and an *Intermediate Node* and lastly use cloud-storage APIs to upload files finally to the cloud-storage servers from the *Intermediate Nodes*. The detour has been shown in Figure 1 by the curved lines from the *User Machine* node towards the *Cloud-Storage Server* node via the *Intermediate Node*.

It should be noted that files on the *Intermediate Node(s)* are always deleted before benchmarking, so there is no benefit gained from `rsync`'s ability to send only file deltas. And, since the file contains random data, it is resistant to any compression-based performance artifacts. Lastly, although we currently use `rsync`, it can be replaced with a different file-transfer tool without changing the basic ideas of our system.

We conducted our experiments on different PlanetLab nodes in North America and on our own non-PlanetLab cluster at University of Alberta (UAlberta). Originally, we chose to use PlanetLab to get access to nodes in many geographical

[3]http://linux.die.net/man/1/rsync

locations, and not necessarily to benchmark the performance of transferring from PlanetLab to cloud-storage providers. We realize that our inefficiencies may be very specific to how PlanetLab traffic is routed to the rest of the world, at certain sites. Given different testbeds and different peering relationships between networks, our inefficiencies may or may not even exist. However, PlanetLab *is* a widely used testbed and not just a minor entity. Therefore, we continued with our work.

Our primary metric of performance is file-transfer time, which is directly related to throughput, of course. To compare throughput, we create differently sized binary files (using `dd` Unix utility with `random` data source) and measure the time taken to upload those files. The file sizes are 10, 20, 30, 40, 50, 60 and 100 MB. For each of the measurements, we take the mean of the last five runs among a total of seven runs. One standard deviation has been shown as the error-bar in the figures. We have written very basic programs in Java, using the APIs of the cloud-storage providers to upload/download files. The experiments were conducted in the months of October and November in 2015.

We have chosen the PlanetLab nodes based on their geographical positions. As seen from the previous research, the geographical location of client does have an effect on the performance for different cloud storage [4]. However, the magnitude of the effect is not clear in the previous research. Our experiments clearly shows the effect of geographical locations for the cloud storage. We have chosen one PlanetLab node in Canada at the University of British Columbia, Vancouver (UBC, west coast of North America (NA)). The rest of the PlanetLab nodes are in the United States of America: University of Michigan (UMich, eastern half of NA), Purdue University (eastern half of NA), University of California, LA (UCLA, west coast of NA). We have also identified the locations of data-center servers for Dropbox, Google Drive and Microsoft OneDrive for all of our experiments. They are located at Ashburn, VA (Dropbox), Mountain View, CA (Google Drive) and Seattle, WA (OneDrive). Fig. 3 shows the geographical locations (obtained from `traceroute` and IP Location Founder [7]) of all the tested clients, intermediate nodes, and cloud storage servers in our experiments.

III. Results and Analysis

In this section, we present our results from experiments conducted on PlanetLab nodes spread across multiple locations in North America and explain our findings regarding network speed to three popular personal cloud-storage services including Dropbox, Google Drive and OneDrive. Our experimental results demonstrate that the non-expert intuition that network speed to cloud-storage providers largely depends on the proximity is not always true—sometimes a detour can save transfer time, although the relative benefit of detour may vary for different clients, services, and even file sizes. We present our results for three case studies of uploading files from (A) UBC, (B) Purdue, and (C) UCLA to the mentioned storage services, respectively, and summarize the relative performance

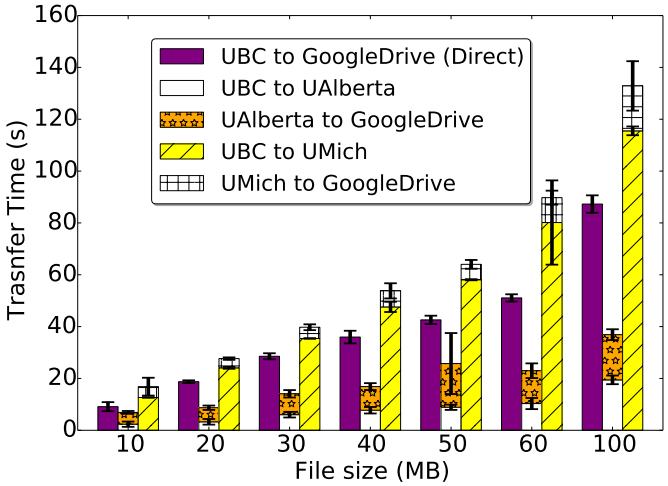


Fig. 2: Upload performance from UBC to Google Drive (direct routes and detours)

of various routes for different file sizes in Table I. We also present Table V which captures the experimental best routes for all of the three client-locations and cloud-storages in geographical maps.

A. UBC: Direct Uploads vs. Detours

First, we present our results measured from the PlanetLab node at the University of British Columbia (UBC). We upload our test files from UBC to Dropbox, Google Drive, and OneDrive, respectively. Our baseline is to use the APIs of these services to upload files directly from UBC to their storage servers. We compare such direct uploads with detoured transfers which route traffic via another intermediate node, as has been described in Sec II. For this experiment, our candidate intermediate nodes include our computing cluster (non-PlanetLab) at the University of Alberta (UAlberta) and a PlanetLab node at the University of Michigan (UMich). We only consider one extra hop in our experiments.

Fig. 2 compares the performance of a direct upload from UBC to Google Drive, versus detoured uploads via UAlberta and UMich. We can see that for all file sizes, the direct upload from UBC is slower than the two-hop indirect route via UAlberta in Edmonton, Alberta, Canada. This is counterintuitive in the sense that one might expect direct uploads to be faster than indirect routes. However, more intuitively, a direct upload from UBC to Google Drive is faster than routing via UMich. Although network speed is the fastest from UMich to Google Drive in these set of experiments, uploads from UBC to UMich are too slow, and thus there is no benefit to using UMich as an intermediate point from UBC to Google Drive. In contrast, UAlberta is a good intermediate node, since the bandwidth is decent both from UBC to UAlberta and from UAlberta to Google Drive. Table II shows the average transfer times from UBC to Google Drive for direct and detour routes. It also includes the relative gain/loss in transfer times for the detoured routes inside the parenthesis, compared to direct routes. We

TABLE I: Summary of the average file transfer times from three client locations to the three cloud-storage providers, using different routes. See also Table V.

Services / Clients	Google Drive	Dropbox	OneDrive
(A) UBC	Fastest: via UAlberta, Fast: Direct, Slowest: via UMich	Fastest: Direct, Fast: via UAlberta, Slowest: via UMich	Fastest: Direct, Fast: UAlberta, Slowest: via UMich
(B) Purdue	Fastest: via UAlberta and via UMich, Slowest: Direct	Fastest: Direct, Slowest: via UAlberta and via UMich [1]	Fastest: Direct, Slowest: via UAlberta and via UMich [2]
(C) UCLA	Fastest: Direct, Fast: via UAlberta, Slowest: via UMich [3]	Fastest: Direct, Fast: via UAlberta, Slowest: via UMich	Fastest: Direct, Fast: via UAlberta, Slowest: via UMich [4]

[1] Exceptions: 40 MB - Slowest: via UMich, Fast: Direct, Fastest: via UAlberta, 60 MB - Slowest: Direct, Fast: via UMich, Fastest: via UAlberta

[2] Exceptions: 10, 60 MB - Slowest: Direct and via UMich, Fastest: via UAlberta, 100 MB - Slowest: Direct, Fast: via UAlberta, Fastest: via UMich

[3] Exceptions: 10, 20 MB - Slowest: via UAlberta, Fast: Direct, Fastest: via UMich, 100 MB - Slowest: via UAlberta, Fast: via UMich, Fastest: Direct

[4] Exceptions: 10 MB - Slowest: via UAlberta, Fast: via UMich, Fastest: Direct

TABLE II: UBC-to-Google Drive Average Transfer Times through different routes

File size (MB)	Direct (s)	via UAlberta (s) [%]	via UMich (s) [%]
10	9.46	6.47 [-31.52%]	15.41 [+62.95%]
20	18.61	8.27 [-55.55%]	27.71 [+48.92%]
30	28.66	13.85 [-51.66%]	39.14 [+36.59%]
40	36.86	17.4 [-52.8%]	51.87 [+40.72%]
50	42.26	19.41 [-54.07%]	63.68 [+50.68%]
60	51.11	21.99 [-56.97%]	80.71 [+57.91%]
100	86.92	35.79 [-58.83]	132.17 [+52.07%]

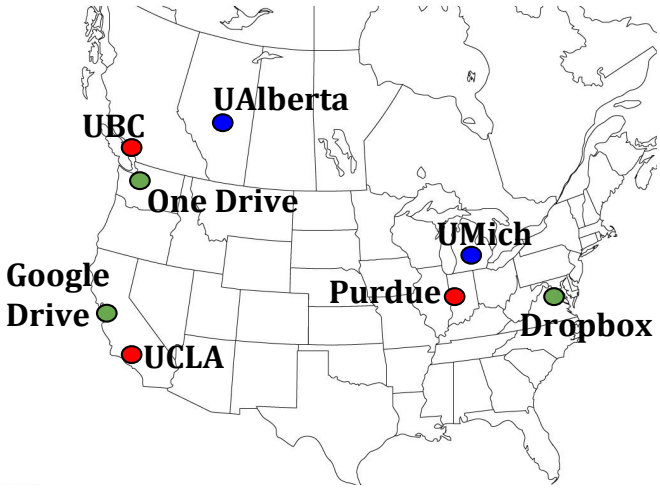


Fig. 3: Locations of clients, intermediate nodes and cloud-storage servers

can see that UAlberta-detour is able to improve the transfer times by more than 50% in most of the cases, while UMich-detour is not able to.

Let us take a closer look at the geographical locations of UBC, UAlberta and Google Server. Through `traceroute` to Google Drive from UBC (Fig. 5) and from UAlberta (Fig. 6), we can see that all three direct and indirect upload modes lead to the same Google Drive server located in Mountain View, CA, USA, based on IP Geolocation services [7]. From Fig. 3, we can see that going from UBC to Google Drive in Mountain View via UAlberta is a significant *geographical* detour. Yet, it has a higher bandwidth than the direct route. This observation implies that geographical proximity or a direct route may not necessarily indicate higher upload bandwidth than detoured transfers. Such an observation also brings about the opportunities to increase network speed by judiciously choosing an intermediate routing node in an overlay network for cloud storage services.

Moreover, even if the routes to be compared are both direct routes, Fig. 2 reveals that geographic proximity may not be positively correlated with the network speed. For example, although UBC is located closer to the same Google Drive server in Mountain View than UAlberta is, UBC has a slower upload speed to Google Drive. It is worth noting that the outgoing bandwidth at UBC is not really the bottleneck here, which can easily be seen from the file transfer time between UBC and UAlberta in Fig. 2.

Continuing the analysis based on `traceroute` (Figures 5 and 6), both network routes cross the middlebox `vncv1rtr2.canarie.ca (199.212.24.1)` once. However, the traffic from UBC is routed through the *pacificwave*[4] network link to the Google Drive server from that middlebox, whereas the traffic from UAlberta is directly

[4]http://pacificwave.net/

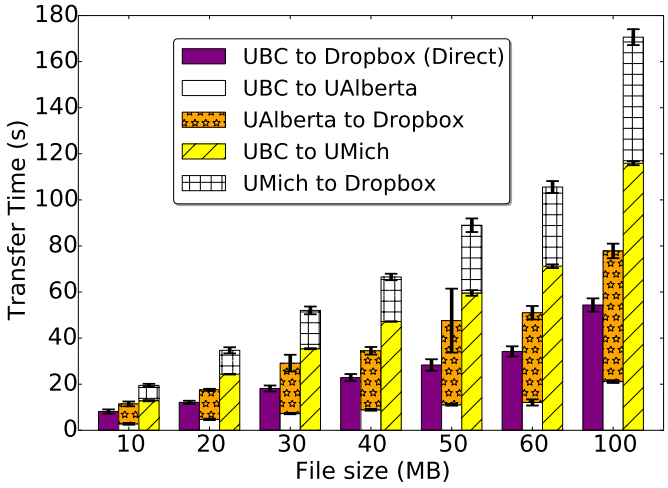


Fig. 4: Upload performance from UBC to Dropbox (direct routes and detours)

```
traceroute to www.googleapis.com (216.58.216.138)
 1  142.103.2.253 (142.103.2.253)
 2  a0-a1.net.ubc.ca (142.103.78.250)
 3  anguborder-a0.net.ubc.ca (137.82.123.137)
 4  345-IX-cr1-UBCAb.vncv1.BC.net (134.87.0.58)
 5  vncv1rtr2.canarie.ca (199.212.24.64)
 6  google-1-lo-std-707.sttlwa.pacificwave.net (207.231.242.20)
 7  209.85.249.32 (209.85.249.32)
 8  216.239.51.159 (216.239.51.159)
 9  sea15s01-in-f138.1e100.net (216.58.216.138)
```

Fig. 5: UBC to Google Drive Server Traceroute

transferred to the Google Drive server through an unknown hop (denoted by $\star\star\star$).

These observations confirm that routing inefficiencies do exist and the network bandwidth between two nodes on the Internet is not explicitly dependent on the proximity between them. Since we have *the least* control over how packets are routed in the Internet, we may leverage the opportunities to route through other intermediate nodes on an overlay network to mitigate routing inefficiencies and increase network speed.

However, intermediate node(s) must be carefully chosen to increase the transfer performance to cloud-storage providers. The performance gain observed for one service may not appear in another service. Fig. 4 plots the performance of file uploads from UBC to another cloud storage service, namely Dropbox. We can see here that direct upload outperforms both indirect routes via UAlberta and UMich. Thus, in this case, choosing UAlberta or UMich as an intermediate node could be a worse choice and is unlikely to yield performance gain. Similar results are observed for OneDrive, and hence, are not plotted here for brevity.

B. Purdue: Choice of Detoured Node

We now discuss the experimental results from Purdue University, with the same candidate intermediate nodes UMich and UAlberta, for all three cloud storage providers tested before. Purdue is located in the eastern part of North America, which is geographically far away from our previous point of

```
traceroute to www.googleapis.com (216.58.216.138)
 1  ww-fw.cs.ualberta.ca (129.128.184.254)
 2  * * *
 3  172.26.244.22 (172.26.244.22)
 4  172.26.244.17 (172.26.244.17)
 5  core1-sc.backbone.ualberta.ca (129.128.0.10)
 6  gsb-asr-core1.backbone.ualberta.ca (129.128.0.21)
 7  uofa-p-1-edm.cybera.ca (199.116.233.66)
 8  edmn1rtr2.canarie.ca (199.212.24.68)
 9  vncv1rtr2.canarie.ca (199.212.24.1)
10  * * *
11  209.85.249.32 (209.85.249.32)
12  216.239.51.159 (216.239.51.159)
13  sea15s01-in-f138.1e100.net (216.58.216.138)
```

Fig. 6: UAlberta to Google Drive Server Traceroute

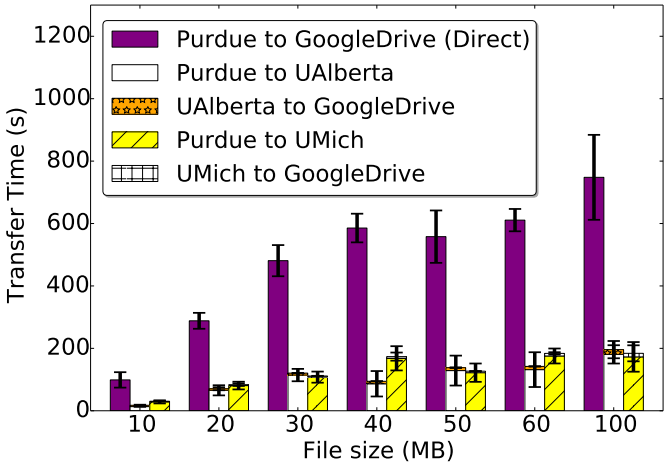


Fig. 7: Upload performance from Purdue to Google Drive (direct routes and detours)

experiment, UBC. However, similar to UBC for Google Drive, a detour for Purdue results in faster uploads (Fig. 7). And, both the detours (i.e., through either UAlberta and UMich) are faster than a direct upload, which is different from the UBC case for Google Drive. And, there is no performance-based reason to prefer a detour through UAlberta to that through UMich or vice versa, since their transfer times are comparable. Table III shows the average transfer times and the relative gain of both the detours (with respect to the direct route) in parenthesis.

Choosing the best intermediate node for a detour is a multi-dimensional problem. The optimal detour depends on the location of the client/user, the available detour nodes, the cloud-service provider, and possibly even the size of the file (discussed below). For example, as discussed above, UBC-to-Google Drive via UMich is not beneficial compared to direct upload, but Purdue-to-Google Drive via UMich is better than the direct upload. Therefore, the location of the client is one of the parameters for the choice of detour node. At this time, our case study only identifies the best detour, but we have not implemented an automatic detour selection algorithm. We now point out how the size of a file can have an effect on the transfer times.

TABLE III: Purdue-to-Google Drive Average Transfer Times through different routes

File size (MB)	Direct (s)	via UAlberta (s) [%]	via UMich (s) [%]
10	98.89	17.57 [-82.24%]	30.59 [-69.07%]
20	288.23	70.55 [-75.52%]	83.62 [-70.99%]
30	480.95	120.69 [-74.91%]	111.37 [-76.84%]
40	585.54	94.43 [-83.87%]	173.53 [-70.36%]
50	557.9	138.03 [-75.26%]	126.82 [-77.27%]
60	610.88	142.15 [-76.73%]	183.85 [-69.9%]
100	748.03	195.88 [-73.81]	184.07 [-75.39%]

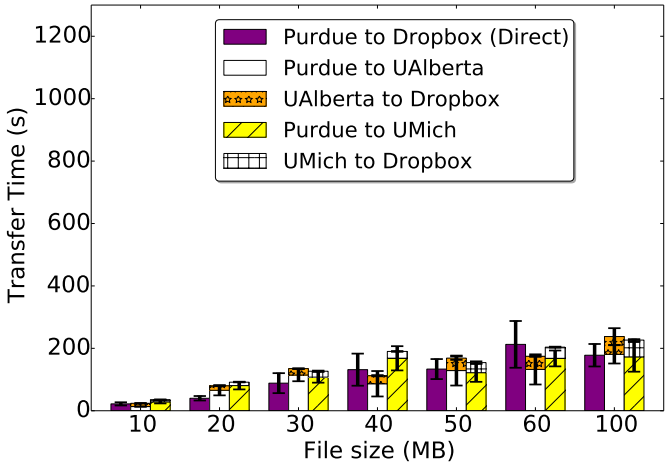


Fig. 8: Upload performance from Purdue to Dropbox (direct routes and detours)

The upload behavior becomes more complicated when we analyze file uploads from Purdue to Dropbox and OneDrive, as shown in Fig. 8 and Fig. 9, respectively, in the sense that the relative performance of direct uploads vs. detoured uploads *is dependent on file sizes*. From Fig. 8, we can see that detoured transfers via intermediate nodes are generally no better than direct uploads, except for 40 MB and 60 MB files. For 40 MB files, detoured uploads via UAlberta outperform direct uploads, although a detour via UMich does not. For 60 MB files, both detoured uploads via UAlberta and UMich outperform direct uploads.

Similarly, for file uploads from Purdue to OneDrive, as shown in Fig. 9, the relative gain from detoured transfers also varies as file sizes change. More evidently, for the case of OneDrive, detoured transfers via intermediate nodes can bring more benefits for larger files. As a result, in such scenarios where the relative performance of different routes also depends on the size of the file to be transferred, it is tricky to decide between the direct route and detours, and choose the best intermediate node for a high speed detoured transfer.

Furthermore, we have not only evaluated the relative performance of different routes by the mean upload time taken, but have also recorded the variations of the time measurements

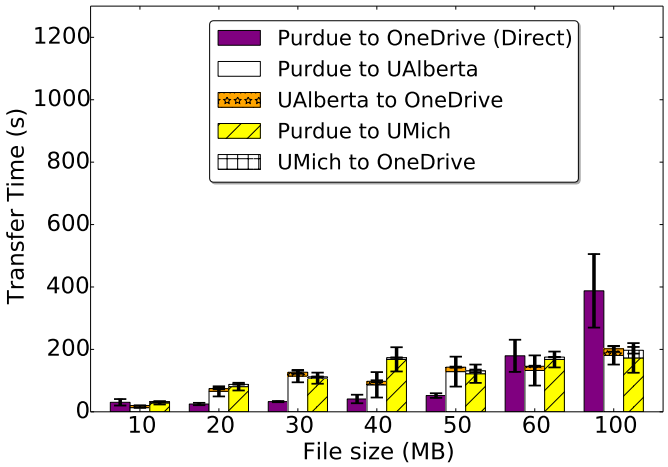


Fig. 9: Upload performance from Purdue to OneDrive (direct routes and detours)

TABLE IV: The mean and standard deviation of upload times (in seconds) from Purdue. Same data as in Figures 8 and 9, but quantitatively for 60 MB and 100 MB only.

File-size (MB)	Type	Mean (s)	Standard deviation
100	Dropbox (Direct)	177.89	36.03
	Dropbox (via UAlberta)	237.78	56.1
	Dropbox (via UMich)	226.43	50.48
	OneDrive (Direct)	387.66	117.81
	OneDrive (via UAlberta)	201.90	38.65
	OneDrive (via UMich)	197.21	58.19
60	Dropbox (Direct)	212.66	74.92
	Dropbox (via UAlberta)	174.54	50.16
	Dropbox (via UMich)	203.78	26.93
	OneDrive (Direct)	179.44	51.49
	OneDrive (via UAlberta)	145.93	50.12
	OneDrive (via UMich)	175.37	26.09

on each route. For example, in Fig. 8, with 40 MB files, the error bar (representing one standard deviation) is quite large for direct uploads, with its lower end overlapping with the time for a detour via UAlberta, making it harder to decide whether direct uploads or detoured uploads are faster. Generally, a route (whether being direct or detoured) with both a lower mean transfer time and a lower transfer time variance is more desirable.

To understand the statistical significance of the measured performance and possible overlapping of detoured and direct transfer times, Table IV summarizes the average transfer times from Purdue along with their standard deviations for Dropbox and OneDrive, respectively, for 100 MB and 60 MB files. If we consider the mean timings for both the direct and detour routes and add/subtract 1 standard deviation to/from it, we can see that their higher and lower end of readings overlap. It can be clarified more by one example. We can take a look at the measurements for uploading a 100 MB files for Dropbox.

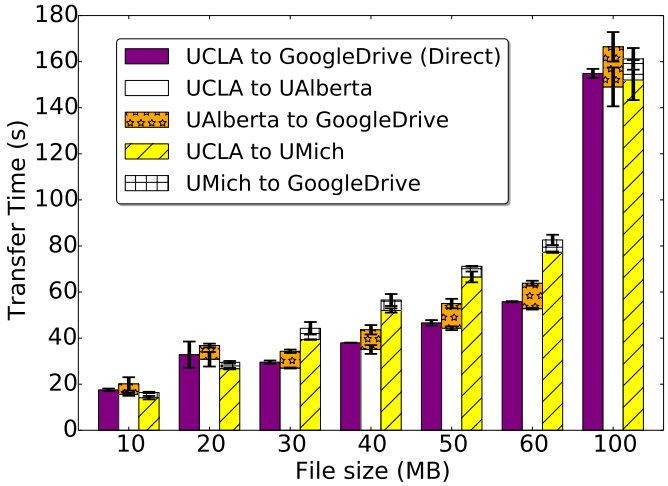


Fig. 10: Upload performance from UCLA to Google Drive (direct routes and detours)

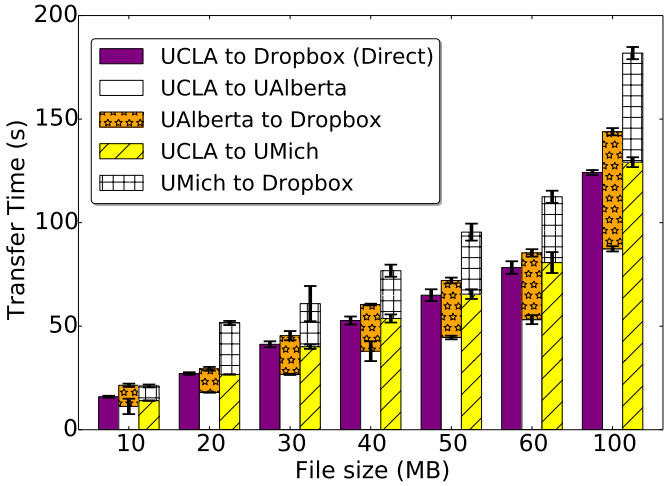


Fig. 11: Upload performance from UCLA to Dropbox (direct routes and detours)

Adding +1 standard deviation to the mean value of the upload times for Dropbox (Direct) gives (177.89 + 36.03) = 213.92s, which marks the higher end of the error bar of upload time. However, subtracting 1 standard deviation from the mean value of the detoured upload times for Dropbox gives (237.78 - 56.1) = 181.68s for UAlberta-detour, and (226.43 - 50.48) = 175.95s for UMich-detour. Both 181.68s and 175.95s for detours represent the lower end of their respective error bars for the upload times measurements. Although the mean upload time for direct route (177.89s) is smaller than the mean upload times for both the detoured routes (237.78s, 226.43s), the higher end of direct routes (213.92s) becomes greater than the lower ends of the detoured routes (181.68s, 175.95s). Hence, it clearly exemplifies an overlap between the direct and detoured transfer times. It is also evident in Fig. 8. Similar observations can be made for other cases in Table IV. Because of this significant overlap, we may not choose to rely on any detours in these types of scenarios. The direct routes can be better than the other choices in these cases due to the unsure benefits of the detours.

C. UCLA: Where Detours Do Not Help

It is worth noting that in all the above experiments, the performance benefit is not due to the network configurations on specific PlanetLab nodes, as the outgoing bandwidth of the above tested PlanetLab nodes did not form the bottleneck in the corresponding experiments. However, we need to emphasize that it is harder for the detoured transfer to improve the throughput where the network bottleneck is the last mile bandwidth of the end host being tested. The experimental results of uploading files from UCLA to Google Drive and Dropbox have illustrated this phenomenon. It turns out that the PlanetLab node at UCLA has limited bandwidth. Fig. 10 shows the performance of uploading files from UCLA to Google Drive. We can see that the file transfers from UCLA to all other locations including the Google Drive server, UAlberta, etc., take a long time. In this case, there is little chance to reduce transfer time by identifying faster detours, since the network bottleneck is (we speculate) UCLA's outgoing bandwidth from that PlanetLab node. Similar observations are made for Dropbox and OneDrive. We plot the results for Dropbox in Fig. 11 and skipped the results of OneDrive for brevity.

D. Discussion

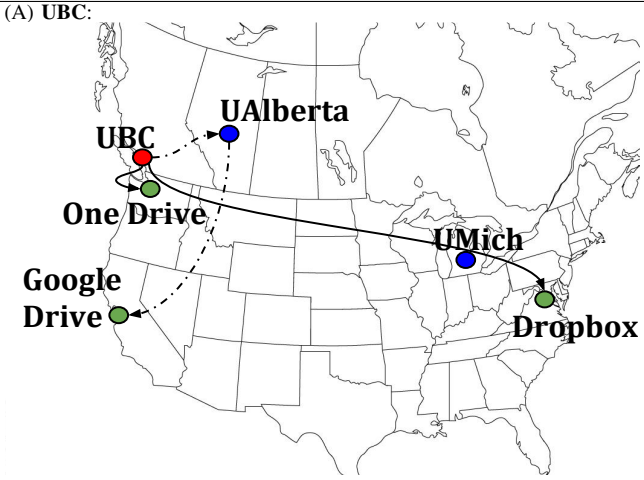
Ideally, we would like to be able to determine the root causes for each of the routing inefficiencies that we have identified. However, the goal of this paper is mainly to catalog and ameliorate the performance problems found. It has been suggested that routing table monitoring systems such as RouteViews (http://www.routeviews.org) might assist in our understanding. Certainly, RouteViews is more sophisticated than our current use of `traceroute`. But, we wanted to understand the potential performance benefits first, as motivation for further investigation.

For future work, systems like RouteViews and dynamic network monitoring tools can be used as important input for a full-fledged overlay network that moves data efficiently to cloud-storage providers, based on routing data and dynamic network conditions (e.g., current congestion points).

To expand the discussion even further: the root cause of performance problems on such a dynamic, multi-administrative domain, and heterogeneous entity such as the Internet can be hard to prove to one's satisfaction. For example, our simple `traceroute` probing (Section III-A) showed that Google Drive traffic from UAlberta took a very similar path between Vancouver and Seattle as traffic from PlanetLab UBC to Google Drive, except for one hop (i.e., pacificwave vs. other). We can see the difference in route, but so what? Is it a physical bandwidth difference at that hop? Is it a rate-limited bandwidth issue? Is it purely a PlanetLab artifact? Is it a software configuration issue affecting only that one hop?

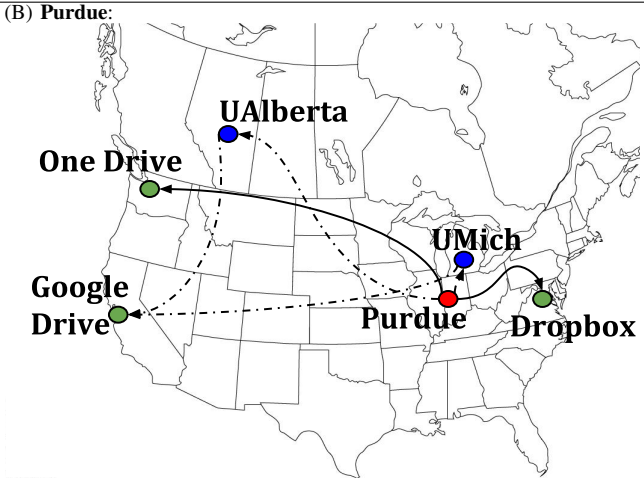
TABLE V: Geographical summary of fastest routes for three client-locations and cloud-storage services. See also Table I. [Direct: solid; Detour: dashed-dotted]

(A) **UBC**:



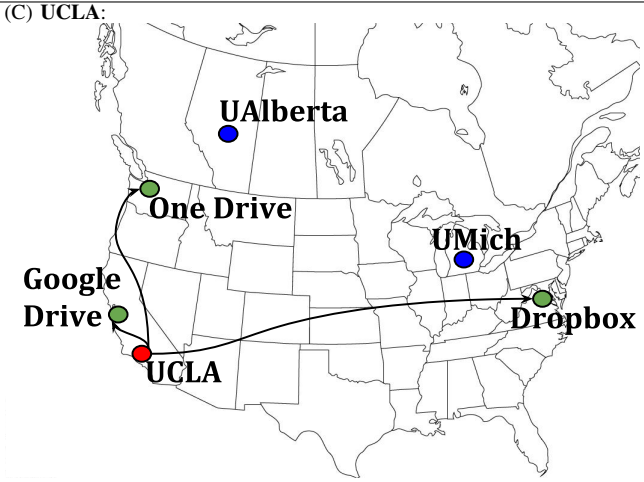
Uploading to Google Drive via a detour to UAlberta (dotted) is faster. Uploading directly to Dropbox and OneDrive are faster than taking any detours.

(B) **Purdue**:



Uploading to Google Drive via a detour to UAlberta or UMich is faster than taking direct route. Uploading directly to Dropbox and OneDrive are faster than taking any detours.

(C) **UCLA**:



Uploading to any cloud-storage services through direct routes is faster than taking any detours

Even if we could inspect pacificwave (which is outside of our control), what would we systematically test first? For this case study, we decided that fixing the performance problem was our first priority, working from top-down.

IV. Other Related Work

The performance of cloud-storage providers has been studied in the past from the client-side point-of-view [4], [8], [9], [10], [11], [12]. Most of the previous work focus on specific details of the client applications of cloud-storage providers and improvements concerning these applications. Li *et. al.* examined the data synchronization traffic for the client applications which can help improve the applications for cloud-storage providers [11]. Drago *et. al.* characterize network traffic to Dropbox by doing passive measurements from different locations in Europe [8]. This work bolsters the case that it is important to address cloud-storage bottlenecks whenever possible. Drago *et. al.* extend their work and benchmark popular cloud-storage providers [4]. Both Li and Drago reveal how Google Drive has benefited from having their datacenters being closer to many of their clients. But, our work shows that geographical proximity can be thwarted by routing inefficiencies.

Our work is different from various attempts to use multiple network paths simultaneously (e.g., Begen, Altunbasak, and Ergun [16]) to improve latency, loss, or other properties. Routing detours pick a single path, known for improved performance to a particular cloud-storage provider. Future use of multiple paths would require changes to the provider's API.

The main concept behind our proposed system is using the idea of overlay networking [1], [13]. This concept has been applied to many kinds of systems such as media streaming [14], [15], peer-to-peer network schemes [17], [18], [19].

Previous work have shown that Triangle Equality Violation (TIV) exists in the Internet [20], [21]. However, most of them report this for latency (in round trip time) [20], [3], [22] and propose solutions exploiting this phenomenon as a feature of the Internet. In our work, we discover that due to routing inefficiencies present in the Internet, we can improve the bandwidth of a particular type of network traffic (cloud-storage traffic) when exploiting TIV.

V. Concluding Remarks

As a cautionary tale and case study, we have identified some inefficiencies in how traffic to well-known cloud-storage providers can vary significantly by a factor of 5 or more, depending on the location of the source and sink of the data. Using simple *routing detours*, we show how the client-to-cloud-storage throughput can be improved by factors of over 3x, even with the detour overheads. Although the specific inefficiencies in this paper might be transitory, they do affect a growing class of traffic of data, namely traffic between clients and cloud-storage providers. As part of the larger discussion of data transfer nodes (DTNs) and overlay networks, our routing detours are simple and effective for a contemporary use case.

As future work, our group plans to expand the functionality of our routing detours to deal with other bottlenecks (e.g., firewalls, like Science DMZ) and to monitor and bypass dynamic bottlenecks on the wide-area network.

VI. Acknowledgements

Thank you to Cybera (www.cybera.ca), CANARIE (www.canarie.ca), the Natural Sciences and Engineering Research Council of Canada (NSERC) (www.nserc.ca), and PlanetLab (www.planet-lab.org) for their support of this research.

References

[1] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris, "Resilient Overlay Networks," in *18th Symposium on Operating System Principles (SOSP)*, Banff, Canada, October 2001, pp. 131–145.

[2] E. Dart, L. Rotman, B. Tierney, M. Hester, and J. Zurawski, "The science dmz: A network design pattern for data-intensive science," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '13. New York, NY, USA: ACM, 2013, pp. 85:1–85:10. [Online]. Available: http://doi.acm.org/10.1145/2503210.2503245

[3] R. Kawahara, E. K. Lua, M. Uchida, S. Kamei, and H. Yoshino, "On the quality of triangle inequality violation aware routing overlay architecture," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 2761–2765.

[4] I. Drago, E. Bocchi, M. Mellia, H. Slatman, and A. Pras, "Benchmarking personal cloud storage," in *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*. ACM Press, 2013, pp. 205–212.

[5] IETF, "The OAuth 2.0 Authorization Framework," https://tools.ietf.org/html/rfc6749, Last accessed: 06-Mar-2016.

[6] R. T. Fielding and R. N. Taylor, "Principled design of the modern web architecture," in *Proceedings of the 22Nd International Conference on Software Engineering*, ser. ICSE '00. New York, NY, USA: ACM, 2000, pp. 407–416. [Online]. Available: http://doi.acm.org/10.1145/337180.337228

[7] IP Location Finder, https://www.iplocation.net/, Last accessed: 06-Mar-2016.

[8] I. Drago, M. Mellia, M. M. Munafo, A. Sperotto, R. Sadre, and A. Pras, "Inside Dropbox: Understanding Personal Cloud Storage Services," in *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, ser. IMC '12. New York, NY, USA: ACM, 2012, pp. 481–494.

[9] A. Bergen, Y. Coady, and R. McGeer, "Client bandwidth: The forgotten metric of online storage providers," in *Communications, Computers and Signal Processing (PacRim), 2011 IEEE Pacific Rim Conference on*, Aug 2011, pp. 543–548.

[10] E. Bocchi, M. Mellia, and S. Sarni, "Cloud storage service benchmarking: Methodologies and experimentations," in *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, Oct 2014, pp. 395–400.

[11] Z. Li, C. Jin, T. Xu, C. Wilson, Y. Liu, L. Cheng, Y. Liu, Y. Dai, and Z.-L. Zhang, "Towards network-level efficiency for cloud storage services," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14. New York, NY, USA: ACM, 2014, pp. 115–128. [Online]. Available: http://doi.acm.org/10.1145/2663716.2663747

[12] L. Yin and G. Cao, "Supporting cooperative caching in ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 1, pp. 77–89, 2006. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1542018

[13] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr., "Overcast: Reliable multicasting with on overlay network," in *Proceedings of the 4th Conference on Symposium on Operating System Design & Implementation - Volume 4*, ser. OSDI'00. Berkeley, CA, USA: USENIX Association, 2000, pp. 14–14. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251229.1251243

[14] D. Tran, K. Hua, and T. Do, "Zigzag: an efficient peer-to-peer scheme for media streaming," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 2, March 2003, pp. 1283–1292 vol.2.

[15] X. Zhang, J. Liu, B. Li, and T. Yum, "Coolstreaming/donet: a data-driven overlay network for peer-to-peer live media streaming," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, March 2005, pp. 2102–2111 vol. 3.

[16] A. C. Begen, Y. Altunbasak, and O. Ergun, "Multi-path selection for multiple description encoded video streaming," in *Communications, 2003. ICC'03. IEEE International Conference on*, vol. 3. IEEE, 2003, pp. 1583–1589. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1203869

[17] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '01. New York, NY, USA: ACM, 2001, pp. 161–172. [Online]. Available: http://doi.acm.org/10.1145/383059.383072

[18] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '03. New York, NY, USA: ACM, 2003, pp. 407–418. [Online]. Available: http://doi.acm.org/10.1145/863955.864000

[19] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys Tutorials, IEEE*, vol. 7, no. 2, pp. 72–93, Second 2005.

[20] H. Zheng, E. K. Lua, M. Pias, and T. G. Griffin, "Internet routing policies and round-trip-times," in *Passive and Active Network Measurement*. Springer, 2005, pp. 236–250.

[21] B. Zhang, T. Ng, A. Nandi, R. H. Riedi, P. Druschel, and G. Wang, "Measurement-based analysis, modeling, and synthesis of the internet delay space," *Networking, IEEE/ACM Transactions on*, vol. 18, no. 1, pp. 229–242, Feb 2010.

[22] G. Wang, B. Zhang, and T. Ng, "Towards network triangle inequality violation aware distributed systems," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. ACM, 2007, pp. 175–188.