

Memo: Towards an Understanding of the Latency and Throughput across Cloud Data Centers

1st Weijia Song

Department of Computer Science

Cornell University

wsong@cornell.edu

I. INTRODUCTION

Cloud storage is a fundamental component of the Cloud infrastructure. Storage reliability is critical because many applications are built around persistent data of various forms. For a major cloud provider, a single data center storage solution thus poses a danger. A data center is more fragile than many developers realize and can suffer damage from power failure [1], [2], accidental fires [3], natural disasters [4], [5], or even unexpected events [6]. Mistakes by administrators that cause damage are rarely discussed, yet are undoubtedly a significant factor as well. Thus for projects in which Failure is Not an Option. [7], [8], cross data-site replication is needed.

Replication data in multiple data centers is a distinct approach to high reliability for Cloud storage and yet is surprisingly poorly supported. One reason is that the relatively high communication latency between data centers, as compared to within a single center, can create a tension between data consistency and system performance. Some systems adopt a bias in favor of performance even at the expense of consistency, by adopting asynchronous replication without addressing consistency aspects [9]–[16]. Those system push the complexity into the application (programs might see inconsistent data), making applications harder to design. Another group of solutions accepts performance loss for strong consistency [17]–[20]. They adopt expensive consensus protocols, such as the ones in the Paxos families [21]–[23]. However, the problem is speed: typical transaction latencies in such systems can be as high as several hundred milliseconds in a cross-country or global deployment.

II. WIDE AREA NETWORK CHALLENGES

Data replication performance is sensitive to network latency and throughput. The speed of light in optical fiber is $\sim 2 \times 10^8 m/s$ or five microseconds per kilometer, which limits the lower bound of network latencies between geographically distributed data centers. Other sources of latency include the network device buffering and operating system overhead. Unlike the high-speed data center network, the communication infrastructure connecting geographically distributed data centers are generally expensive fiber cables maintained with a lot of efforts and shared by multiple tenants [24]. This shared nature constrains the throughput between the data centers.

As the first step to an extensive measurement of wide area network properties, we run measurement in fourteen Amazon

EC2 Regions to understand the latency and throughput in the public cloud, since we believe most of the geographically distributed applications are going to run in public cloud in future.

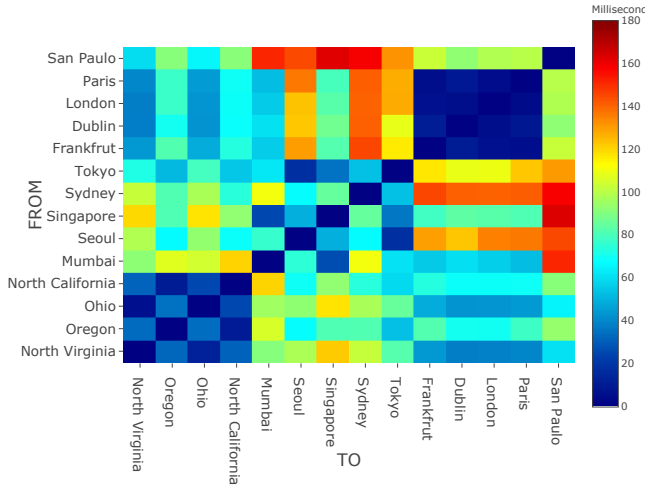
A. Network Latency

We launched fourteen `t2.small` instances in different Amazon EC2 regions and measured the latencies between each pair of instances. Round Trip Time(RTT) measured with `ping` is widely used. We are however more interested in the one-sided latency because an application may experience different performance if messages are passed through different paths with asymmetric links, like delivering packages in a city with heavy commuter traffics.

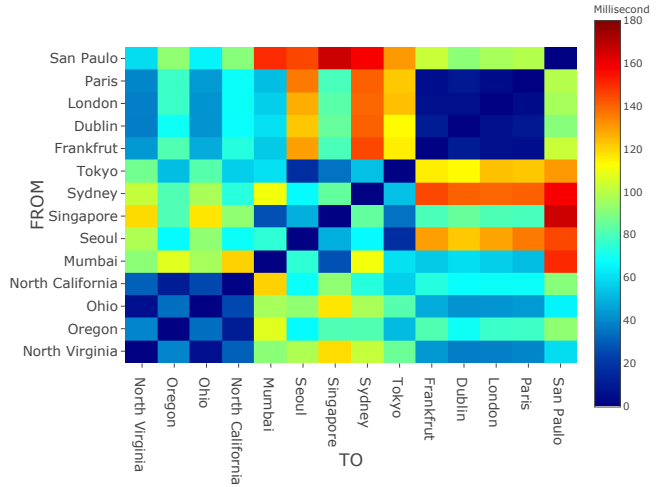
Measuring one-sided latency requires a globally synchronized clock. Although there is no perfect global synchronized clock, we can rely on the timing facilities available in the public Cloud. For example, Amazon EC2 has GPS and atomic clock sources in its regions and allows the instances synchronizing with NTP [25]. A GPS Receiver can synchronize to the satellites within tens of nanoseconds and the time skew between NTP synchronized hosts is about several milliseconds inside a data center [18]. Therefore, we assume a globally synchronized clock with a precision of milliseconds. This precision is enough because the latencies in wide area network are of tens or hundreds of milliseconds.

In figure 1a, we show the one-sided latencies between instances in various regions. We can see that the latencies are consistent with half of the RTT measured by `ping` as shown in figure 1b. The average difference between the latencies in both directions is 1.4 milliseconds, which is in NTP precision range and negligible compared to the WAN latency. We ran the same experiment with instances in both Google Cloud and Amazon EC2 for multiple times and get similar results with some occasional exceptions.

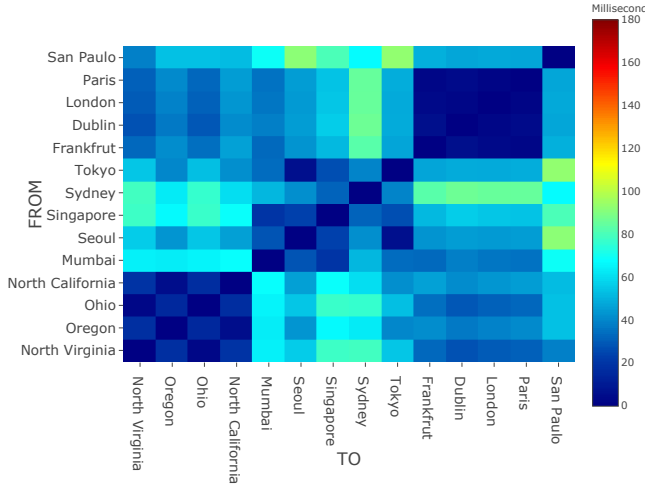
We also tested the latency fluctuation over time between two instances in Amazon EC2 North Virginia and Singapore regions. The latencies are sampled 100 times during a five seconds period every half an hour. The results are shown in figure 1d. At 2.5 hours, some packets take up to six seconds traveling from North Virginia to Singapore. And at 13 hours, some packets take up to 220 milliseconds in the same direction. Except for those cases, the latency is stable around 120 milliseconds in both directions. Therefore, we argue



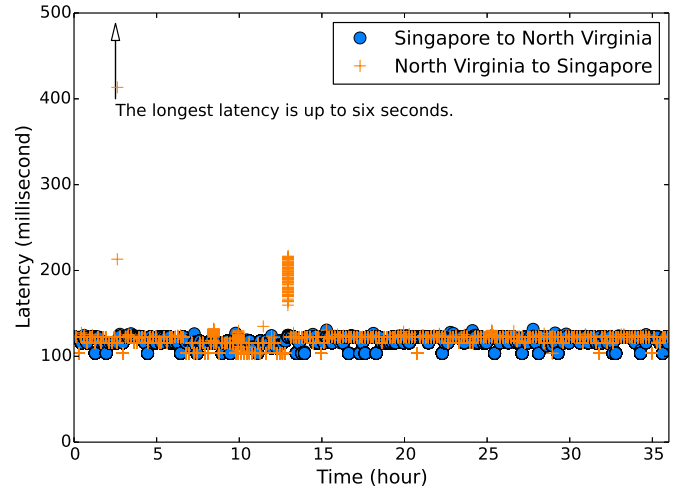
(a) Measured One Way Latency



(b) Half Round Trip Time



(c) Ideal One-Way Latency



(d) Latency over time

Fig. 1: Latency between Amazon EC2 Data Centers

that the network latency between geographically distributed data centers are symmetric. Occasional bursty delays occur presumably due to congestion.

In figure 1c, we also show the ideal latencies between instances in various regions by dividing the great-circle distances by the speed of light in the optical fiber. We found the measured latencies are generally less than two times of the ideal ones. The real distance is a good estimation of the latency between data centers.

B. Throughput

In the same setup in the last section, we use the `iperf` tool to measure the TCP throughput between instances across data centers. Figure 2a shows the average throughput of ten `iperf` runs. Each run lasts for ten seconds. The throughput in both directions roughly agrees with each other showing

less symmetry than network latency. This is because the throughput test is sensitive to the traffic in shared links. As shown in figure 2b, we monitored the throughput between two instances in North Virginia region and Singapore region. Although the average throughput is statistically comparable in both directions, it fluctuates dramatically over time.

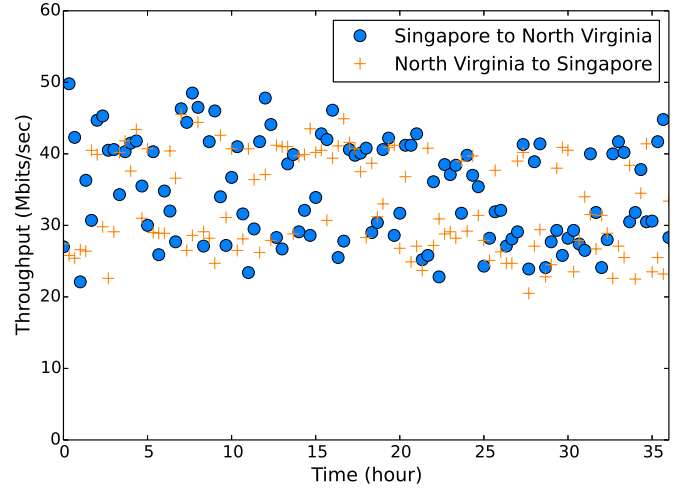
The numbers in figure 2a are throughput values in Mbits per second. We can see the cross-region throughput is magnitudes lower than inside the data center, which limits the speed of data propagation to multiple data centers. An interesting phenomenon is that closer data centers can get higher throughput in between. We are going to explore this aspect with more experiment since our current experiment is far from extensive.

C. Challenges to Data Replication

The latency is challenging with running consensus protocols across data centers, because a process has to block for the

FROM	San Paulo	Paris	London	Dublin	Frankfurt	Tokyo	Sydney	Singapore	Seoul	Mumbai	North California	Ohio	Oregon	North Virginia	TO
	50	40	47	39	27	24	29	28	37	29	25	33	38	607	San Paulo
	65	33	58	32	46	21	41	27	36	131	72	242	895	35	Paris
	57	38	48	38	48	24	32	27	30	118	104	896	298	39	London
	69	61	53	43	55	32	49	35	37	79	465	184	159	45	Dublin
	66	40	41	41	60	28	49	30	40	537	73	170	251	41	Frankfurt
	41	54	38	46	39	65	63	63	374	29	25	31	31	34	Tokyo
	30	39	32	40	30	31	39	832	64	19	18	25	26	24	Sydney
	25	36	26	30	96	34	831	36	96	24	23	29	36	26	Singapore
	37	43	28	45	42	872	54	48	142	22	22	29	30	30	Seoul
	50	38	36	35	503	40	91	40	64	36	34	60	54	29	Mumbai
	74	113	89	607	32	31	38	51	67	27	26	40	40	46	North California
	238	59	460	78	37	32	40	44	51	38	36	62	69	54	Ohio
	62	792	61	132	30	29	45	45	65	25	24	36	35	36	Oregon
	78	54	161	63	44	28	29	37	52	29	31	59	56	57	North Virginia

(a) Throughput (Mbits/sec)



(b) Throughput over Time

Fig. 2: Throughput between Amazon EC2 Data Centers

message from the other side. Prior to the introduction of the Paxos suite of protocols, the consensus was generally realized by total-ordering the commands applied to some form of replicated state machine, atomic broadcasts [21]. Systems of this kind dynamically track membership, and in the event of node failure, a view reconfiguration is triggered to remove the failed node and restart execution with only healthy members. Paxos protocols avoid this reconfiguration by instead using a quorum-overlap approach. The result is that a Paxos protocol can avoid reconfiguration for any failure that is expected to be transient in duration. However, Paxos requires two round trips of one-to-many and many-to-one message passing to decide a value [22]. To solve this problem, previous work suggested that when running a stream of Paxos protocols, it would be useful to explore piggybacking the prepare phase in last accept phase [26], or electing a stable leader to avoid the prepare phase for multiple log entries [19]. Fast Paxos goes further by dynamically adjusting the size of the quorum so that decisions can be made in one round trip [27]. Along with other optimizations like batching and separate data and control planes, today's consensus protocols are generally considered practical for use within a single data center. But geo-replicated replication is still a challenge.

Some applications like Facebook and Instagram can work without strong consistency, where data operations do not block until it reaches the remote data center. However, a replica may wait until all dependencies arrived before exposes an update to clients. Data archiving applications can work with only eventual consistency. In geo-replication setup, those applications require that the network bandwidth be able to catch up with their throughput.

Fortunately, WAN optimizations are available to explore now. BBR, the new TCP congestion control algorithm, allows fast adaption to an optimal point of latency and through-

put [28]. Besides BBR compatibility and security, a new transport layer protocol called QUIC optimized the connection latency with fewer round-trips [29]. Introducing relays can reduce the network latency in WAN [30]. We are going to explore those directions in future work.

REFERENCES

- [1] S. Sanders, "January 28th incident report," <https://bit.ly/1KYel5e>, accessed 2016-11-17.
- [2] S. Carey, "Delta meltdown reflects problems with aging technology," <https://on.wsj.com/2JaL2D7>, accessed 2018-04-01.
- [3] R. Lawler, "Samsung data center fire causes outage, errors on smart tvs and phones," <https://engt.co/2EcNQf8>, accessed 2018-04-01.
- [4] A. McNevin, "Hurricane sandy takes out manhattan data centers," <https://bit.ly/2oNIArj>, accessed 2018-04-01.
- [5] K. Porter, "Can your data center survive an earthquake? how to know your risk," <https://bit.ly/2Grs11o>, accessed 2018-04-01.
- [6] S. C. Margi Murphy, "11 data center disasters," <https://bit.ly/2Ec7esY>, accessed 2018-04-01.
- [7] C. Hayes, "Data centres: failure is not an option," <https://bit.ly/2HIVsyv>, accessed 2018-04-01.
- [8] R. Miller, "Failure is not an option: Lessons learned from disasters," <https://bit.ly/2pXBNNH>, accessed 2018-04-01.
- [9] D. B. Terry, M. M. Theimer, K. Petersen, A. J. Demers, M. J. Spreitzer, and C. H. Hauser, *Managing update conflicts in Bayou, a weakly connected replicated storage system*. ACM, 1995, vol. 29, no. 5.
- [10] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Voshall, and W. Vogels, "Dynamo: amazon's highly available key-value store," in *ACM SIGOPS operating systems review*, vol. 41, no. 6. ACM, 2007, pp. 205–220.
- [11] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski, "Conflict-free replicated data types," in *Symposium on Self-Stabilizing Systems*. Springer, 2011, pp. 386–400.
- [12] Y. Sovran, R. Power, M. K. Aguilera, and J. Li, "Transactional storage for geo-replicated systems," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. ACM, 2011, pp. 385–400.
- [13] W. Lloyd, M. J. Freedman, M. Kaminsky, and D. G. Andersen, "Don't settle for eventual: scalable causal consistency for wide-area storage with cops," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. ACM, 2011, pp. 401–416.
- [14] C. Li, D. Porto, A. Clement, J. Gehrke, N. M. Preguiça, and R. Rodrigues, "Making geo-replicated systems fast as possible, consistent when necessary," in *OSDI*, vol. 12, 2012, pp. 265–278.

- [15] W. Lloyd, M. J. Freedman, M. Kaminsky, and D. G. Andersen, "Stronger semantics for low-latency geo-replicated storage," in *NSDI*, vol. 13, 2013, pp. 313–328.
- [16] Z. Wu, M. Butkiewicz, D. Perkins, E. Katz-Bassett, and H. V. Madhyastha, "Spanstore: Cost-effective geo-replicated storage spanning multiple cloud services," in *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*. ACM, 2013, pp. 292–308.
- [17] J. Baker, C. Bond, J. C. Corbett, J. Furman, A. Khorlin, J. Larson, J.-M. Leon, Y. Li, A. Lloyd, and V. Yushprakh, "Megastore: Providing scalable, highly available storage for interactive services," in *CIDR*, vol. 11, 2011, pp. 223–234.
- [18] J. C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, J. J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild *et al.*, "Spanner: Googles globally distributed database," *ACM Transactions on Computer Systems (TOCS)*, vol. 31, no. 3, p. 8, 2013.
- [19] T. Kraska, G. Pang, M. J. Franklin, S. Madden, and A. Fekete, "Mdcc: Multi-data center consistency," in *Proceedings of the 8th ACM European Conference on Computer Systems*. ACM, 2013, pp. 113–126.
- [20] S. Mu, L. Nelson, W. Lloyd, and J. Li, "Consolidating concurrency control and consensus for commits under conflicts," in *OSDI*, 2016, pp. 517–532.
- [21] K. P. Birman and T. A. Joseph, "Reliable communication in the presence of failures," *ACM Transactions on Computer Systems (TOCS)*, vol. 5, no. 1, pp. 47–76, 1987.
- [22] L. Lamport, "The part-time parliament," *ACM Transactions on Computer Systems (TOCS)*, vol. 16, no. 2, pp. 133–169, 1998.
- [23] F. P. Junqueira, B. C. Reed, and M. Serafini, "Zab: High-performance broadcast for primary-backup systems," in *Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on*. IEEE, 2011, pp. 245–256.
- [24] Unknown, "Submarine cable map," <https://www.submarinecablemap.com/>, accessed 2018-04-06.
- [25] R. Hunt, "Keeping time with amazon time sync service," <https://amzn.to/2zCG8K7>, accessed 2018-04-06.
- [26] T. D. Chandra, R. Griesemer, and J. Redstone, "Paxos made live: an engineering perspective," in *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*. ACM, 2007, pp. 398–407.
- [27] L. Lamport, "Fast paxos," *Distributed Computing*, vol. 19, no. 2, pp. 79–103, 2006.
- [28] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control," *Queue*, vol. 14, no. 5, p. 50, 2016.
- [29] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar *et al.*, "The quic transport protocol: Design and internet-scale deployment," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 183–196.
- [30] A. Babay, E. Wagner, M. Dinitz, and Y. Amir, "Timely, reliable, and cost-effective internet transport service using dissemination graphs," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017, pp. 1–12.