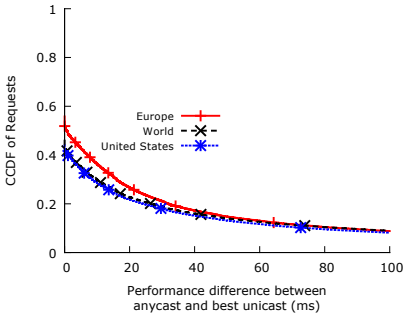


**Figure 3: The fraction of requests where the best of three different unicast front-ends out-performed anycast.**

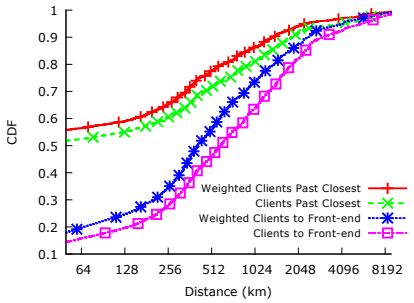


**Figure 4: The distance in kilometers (log scale) between clients and the anycast front-ends they are directed to.**

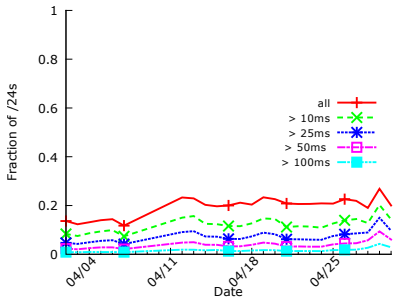


**Figure 5: Daily poor-path prevalence during April 2015 showing what fraction of client /24s see different levels of latency improvement over anycast when directed to their best performing unicast front-end.**

to make suboptimal choices and 2) intradomain routing policies of ISPs select remote peering points with our network.

In one interesting example, a client was roughly the same distance from two border routers announcing the anycast route. Anycast chose to route towards router *A*. However, internally in our network, router *B* is very close to a front-end *C*, whereas router *A* has a longer intradomain route to the nearest front-end, front-end *D*. With anycast, there is no way to communicate [39] this internal topology information in a BGP announcement.

Several other examples included cases where a client is nearby a front-end but the ISP's internal policy chooses to hand off traffic at a distant peering point. Microsoft intradomain policy then directs the client's request to the front-end nearest to the peering point, not to the client. Some examples we observed of this was an ISP carrying traffic from a client in Denver to Phoenix and another carrying traffic from Moscow to Stockholm. In both cases, direct peering was present at each source city.

Intrigued by these sorts of case studies, we sought to understand anycast performance quantitatively. The first question we ask is whether anycast performance is poor simply because it occasionally directs clients to front-ends that are geographically far away, as was the case when clients in Moscow went to Stockholm.

**Does anycast direct clients to nearby front-ends?** In a large CDN with presence in major metro areas around the world, most ISPs will see BGP announcements for front-ends from a number of different locations. If peering among these points is uniform, then the ISP's least cost path from a client to a front-end will often be the geographically closest. Since anycast is not load or latency aware, geographic proximity is a good indicator of expected performance.

Figure 4 shows the distribution of the distance from client to anycast front-end for all clients in one day of production Bing traffic. One line weights clients by query volume. Anycast is shown to perform 5-10% better at all percentiles when accounting for more active clients. We see that about 82% of clients are directed to a front-end within 2000 km while 87% of client volume is within 2000 km.

The second pair of lines in Figure 4, labeled "Past Closest", shows the distribution of the difference between the distance from a client to its closest front-end and the distance from the client to the front-end anycast directs to. About 55% of clients and weighted clients have distance 0, meaning they are directed to the nearest front-end. Further, 75% of clients are directed to a front-end within around 400 km and 90% are within 1375 km of their closest. This supports the idea that, with a dense front-end deployment such as is achievable in North America and Europe, anycast directs most clients to a relatively nearby front-end that should be expected to deliver good performance, even if it is not the closest.

From a geographic view, we found that around 10-15% of /24s are directed to distant front-ends, a likely explanation for poor performance.[1] Next we examine how common these issues are from day-to-day and how long issues with individual networks persist.

**Is anycast performance consistently poor?** We first consider whether significant fractions of clients see consistently poor performance with anycast. At the end of each day, we analyzed all collected client measurements to find prefixes with room for improvement over anycast performance. For each client /24, we calculate the median latency between the prefix and each measured unicast front-end and anycast.

Figure 5 shows the prevalence of poor anycast performance each day during April 2015. Each line specifies a particular minimum latency improvement, and the figure shows the fraction of client /24s each day for which some unicast front-end yields at least that improvement over anycast. On average, we find that 19% of prefixes see some performance benefit from going to a specific unicast front-end instead of using anycast. We see 12% of clients with 10ms or more improvement, but only 4% see 50ms or more.

Poor performance is not limited to a few days–it is a daily concern. We next examine whether the same client networks experience recurring poor performance. How long does poor performance persist? Are the problems seen in Figure 5 always due to the same problematic clients?

Figure 6 shows the duration of poor anycast performance during April 2015. For the majority of /24s categorized as having poor-performing paths, those poor-performing paths are short-lived. Around 60% appear for only one day over the month. Around 10% of /24s show poor performance for 5 days or more. These days are not necessarily consecutive. We see that only 5% of /24s see continuous poor performance over 5 days or more.

These results show that while there is a persistent amount of poor anycast performance over time, the majority of problems only last

[1]No geolocation database is perfect. A fraction of very long client-to-front-end distances may be attributable to bad client geolocation data.

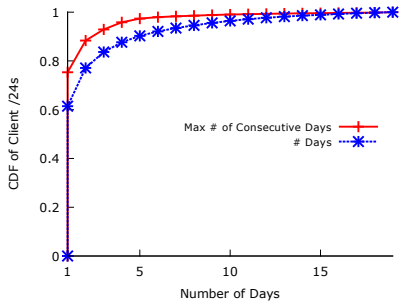


**Figure 6: Poor path duration across April 2015. We consider poor anycast paths to be those with any latency inflation over a unicast front-end.**

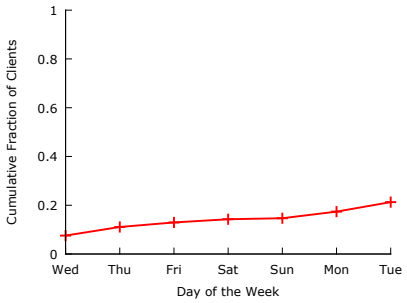


**Figure 7: The cumulative fraction of clients that have changed front-ends at least once by different points in a week**

for a single day. Next we look at how much of poor performance can be attributed to clients frequently switching between good and poor performing front-ends.

**Front-end Affinity:** Recurrent front-end selection changes for user over time may indicate route stability issues which can lead to anycast performance problems. We refer to how "attached" particular clients are to a front-end as *front-end affinity*. In this section, we analyze our passive logs.

Figure 7 shows the cumulative fraction of clients that have switched front-ends at least once by that time of the week. Within the first day, 7% of clients landed on multiple front-ends. An additional 2-4% clients see a front-end change each day until the weekend, where there is very little churn, less than .5%. This could be from network operators not pushing out changes during the weekend unless they have to. From the weekend to the beginning of the week, the amount of churn increases again to 2-4% each day. Across the entire week, 21% of clients landed on multiple front-ends, but the vast majority of clients were stable. We discuss potential solutions to this more at the end of §6. We observe that the number of client front-end switches is slightly higher in a one day snapshot compared to the 1.1-4.7% reported in previous work on DNS instance-switches in anycast root nameservers [20, 33]. A likely contributing factor is that our anycast deployment is around 10 times larger than the number of instances present in K root name server at the time of that work.

Figure 8 shows the change in the client-to-front-end distance when the front-end changes. This shows that when the majority of clients switch front-ends, it is to a nearby front-end. This makes sense given the CDN front-end density in North America and Europe. The median change in distance from front-end switches is 483 km while 83% are within 2000 km.

We saw in this section that most clients show high front-end-affinity, that is, they continue going to the same front-end over time. For the clients that do switch front-ends, there is a long tail of distance between a client and switched pairs of front-ends.

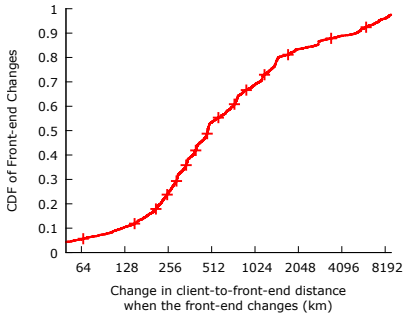


**Figure 8: The distribution of change in client-to-front-end distance (log scale) when when the front-end changes, for the 7% of clients that change front-end throughout a day.**

# 6. ADDRESSING POOR PERFORMANCE

The previous section showed that anycast often achieves good performance, but sometimes suffers significantly compared to unicast beacon measurements. However, the ability for unicast to beat anycast in a single measurement does not guarantee that this performance is predictable enough to be achievable if a system has to return a single unicast front-end to a DNS query. If a particular front-end outperformed anycast in the past for a client, will it still if the system returns that front-end next time? Additionally, because of DNS's design, the system does not know which client it is responding to, and so its response applies either to all clients of an LDNS or all clients in a prefix (if using ECS). Can the system reliably determine front-ends that will perform well for the set of clients?

We evaluate to what degree schemes using DNS and ECS can improve performance for clients with poor anycast performance. We evaluate (in emulation based on our real user measurements) a prediction scheme that maps from a client group (clients of an LDNS or clients within an ECS prefix) to its predicted best front-end. It updates its mapping every *prediction interval*, set to one day in our experiment.[2] The scheme chooses to map a client group to the lowest latency front-end across the measurements for that group, picking either the anycast address or one of the unicast front-ends. We evaluate two *prediction metrics* to determine the latency of a front-end, 25th percentile and median latency from that client group to that front-end. We choose lower percentiles, as analysis of client data showed that higher percentiles of latency distributions are very noisy (we omit detailed results due to lack of space). This noise makes prediction difficult, as it can result in overlapping performance between two front-ends. The 25th percentile and median have lower coefficient of variation, indicating less variation and more stability. Our initial evaluation showed that both 25th percentile and median show very similar performance as *prediction metrics*, so we only present results for 25th percentile.

We emulate the performance of such a prediction scheme using our existing beacon measurements. We base the predictions on one day's beacon measurements. For a given client group, we select among the front-ends with 20+ measurements from the clients.

We evaluate the performance of the prediction scheme by comparing against the performance observed in next day's beacon measurements. We compare 50th and 75th anycast performance for the group to 50th and 75th performance for the predicted front-end. The Bing team routinely uses 75% percentile latency as an internal benchmarks for a variety of comparisons. Next, we evaluate prediction using both ECS and LDNS client grouping.

[2]We cannot make predictions at finer timescales, as our sampling rate was limited due to engineering issues.

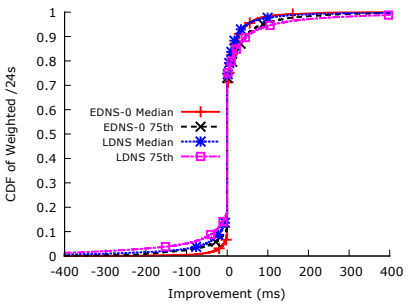


**Figure 9: Improvement over anycast from making LDNS or ECS-based decisions with prediction using 25th percentile prediction metric. Negative x-axis values show where anycast was better than our prediction. Values at 0 show when we predicted anycast was the best performing. Positive x-axis values show our improvement.**

**Prediction using EDNS client-subnet-prefix:** The ECS extension [21] enables precise client redirection by including the client's prefix in a DNS request. Our prediction scheme is straightforward: we consider all beacon measurements for a /24 client network and choose the front-end according to the *prediction metrics*.

The "EDNS-0" lines in Figure 9 depict, as a distribution across clients weighted by query volume, the difference between performance to the predicted front-end (at the 50th and 75th percentile) and the performance to the anycast-routed front-end (at the same percentiles). Most clients see no difference in performance, in most cases because prediction selected the anycast address. For the nearly 40% of queries-weighted prefixes we predict to see improvement over anycast, only 30% see a performance improvement over anycast, while 10% of weighted prefixes see worse performance than they would with anycast.

**LDNS-based prediction:** Traditionally, DNS-based redirection can only make decisions based on a client's LDNS. In this section, we estimate to what degree LDNS granularity can achieve optimal performance when anycast routing sends clients to suboptimal servers. We construct a latency mapping from LDNS to each measured edge by assigning each front-end measurement made by a client to the client's LDNS, which we can identify by joining our DNS and HTTP logs based on the unique hostname for the measurement. We then consider all beacon measurements assigned to an LDNS and select the LDNS's best front-end using the *prediction metrics*. In the page loads in our experiment, public DNS resolvers made up a negligible fraction of total LDNS traffic so their wide user base have an insignificant impact on results.

The "LDNS" lines in Figure 9 show the fraction of /24 client networks that can be improved from using prediction of performance based on an LDNS-based mapping. While we see improvement for around 27% of weighted /24s, we also pay a penalty where our prediction did poorly for around 17% of /24s.

Our results demonstrate that traditional and recent DNS techniques can improve performance for many of the clients who experience suboptimal anycast routing. We are also considering a hybrid approach that combines anycast with DNS-based redirection. The key idea is to use DNS-based redirection for a small subset of poor performing clients, while leaving others to anycast. Such a hybrid approach may outperform DNS redirection for clients not well represented by their LDNS, and it may be more scalable.

# 7. RELATED WORK

Most closely related to our work is from Alzoubi et al. [9, 8]. They describe a load-aware anycast CDN architecture where ingress routes from a CDN to a large ISP are managed by an ISP's centralized route controller. Unlike our work, they do not examine the end-to-end application performance comparison between DNS redirection and anycast. Follow up work focuses on handling anycast TCP session disruption due to BGP path changes [7]. Our work is also closely related to FastRoute [23], a system for load balancing within an anycast CDN, but it does not address performance issues around redirection. There has been a good deal of work on improving and evaluating general CDN performance [37, 24, 36, 6, 35, 25]. The majority of previous work on anycast performance has focused on DNS. There has been significant attention to anycast DNS from the network operations community [13, 15, 14, 28, 19, 12, 20] but less so for TCP and anycast [31]. Sarat et al. examined the performance impact of anycast on DNS across different anycast configurations [38]. Fan et al. [22] present new methods to identify and characterize anycast nodes. There are several pieces of work describing deployment of anycast services [30, 10, 11, 26].

Akamai recently published a study on DNS-based redirection [17]. The authors showed that the majority of clients are nearby their LDNS, enabling DNS-based redirection to perform well. However, they also show that a significant number of clients are far from their LDNS, and that some LDNS serve clients spread over large geographic regions. The paper describes Akamai's adoption of ECS-based redirection for clients of public DNS resolvers, showing impressive performance improvements for these clients versus LDNS-based redirection. However, public resolvers only make up a small fraction of global DNS traffic. Clients using their ISPs' LDNS cannot benefit unless the ISPs enable ECS and the CDN supports ECS requests from the LDNS. Since anycast works well for many clients, we see benefit in a hybrid approach that chooses whether to use DNS redirection or anycast based on measurements of which works better for the LDNS and whether the LDNS supports ECS.

# 8. CONCLUSION

In this paper we studied the performance of a large anycast-based CDN, and evaluated whether it could be improved by using a centralized, DNS-based solution. We found that anycast usually performs well despite the lack of precise control, but that it directs $\approx$ 20% of clients to a suboptimal front-end. We demonstrated that a simple prediction scheme may allow DNS redirection to improve performance for some of the clients that see poor anycast performance.

## Acknowledgements

We gratefully acknowledge Nick Holt and Daniel Gicklhorn for their support of this work. Matt Calder and Ethan Katz-Bassett were partially supported by the U.S. National Science Foundation grant numbers CNS-1351100 and CNS-1413978.

# 9. REFERENCES

[1] CloudFlare. `https://www.cloudflare.com/`.
[2] RIPE Atlas. `https://atlas.ripe.net/`.
[3] USC CDN Coverage. `http://usc-nsl.github.io/cdn-coverage`.
[4] V. K. Adhikari, Y. Guo, F. Hao, V. Hilt, and Z.-L. Zhang. Tale of Three CDNs: An Active Measurement Study of Hulu and its CDNs. In *IEEE Global Internet Symposium '12*.
[5] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing DNS Resolvers in the Wild. In *IMC '10*.
[6] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Web Content Cartography. In *IMC '11*.

[7] Z. Al-Qudah, S. Lee, M. Rabinovich, O. Spatscheck, and J. Van der Merwe. Anycast-aware Transport for Content Delivery Networks. In *WWW '09*.

[8] H. A. Alzoubi, S. Lee, M. Rabinovich, O. Spatscheck, and J. Van Der Merwe. A Practical Architecture for an Anycast CDN. *ACM Transactions on the Web (TWEB) '11*.

[9] H. A. Alzoubi, S. Lee, M. Rabinovich, O. Spatscheck, and J. Van der Merwe. Anycast CDNs Revisited. In *WWW '08*.

[10] H. Ballani and P. Francis. Towards a Global IP Anycast Service. In *SIGCOMM '05*.

[11] H. Ballani, P. Francis, and S. Ratnasamy. A Measurement-based Deployment Proposal for IP Anycast. In *IMC '06*.

[12] P. Barber, M. Larson, and M. Kosters. Traffic Source Analysis of the J Root Anycast Instances. NANOG 39. February, '07.

[13] P. Barber, M. Larson, M. Kosters, and P. Toscano. Life and Times of J-ROOT. NANOG 32. October, '04.

[14] P. Boothe and R. Bush. Anycast Measurements Used To Highlight Routing Instabilities. NANOG 35. October, '05.

[15] P. Boothe and R. Bush. DNS Anycast Stability. *19th APNIC, '05*.

[16] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan. Mapping the Expansion of Google's Serving Infrastructure. In *IMC '13*.

[17] F. Cheng, R. K. Sitaraman, and M. Torres. End-user mapping: Next Generation Request Routing for Content Delivery. In *SIGCOMM '15*.

[18] Y. Chiu, B. Schlinker, A. B. Radhakrishnan, E. Katz-Bassett, and R. Govindan. Are We One Hop Away from a Better Internet? In *IMC '15*.

[19] L. Coletti. Effects of Anycast on K-root Performance. NANOG 37. June, '06.

[20] L. Colitti, E. Romijn, H. Uijterwaal, and A. Robachevsky. Evaluating the Effects of Anycast on DNS Root Name Servers. *RIPE document RIPE-393, '06*.

[21] C. Contavalli, W. van der Gaast, D. Lawrence, and W. Kumari. Client Subnet in DNS Requests. IETF Draft draft-vandergaast-edns-client-subnet-02, July 2015.

[22] X. Fan, J. Heidemann, and R. Govindan. Evaluating Anycast in the Domain Name System. In *INFOCOM '13*.

[23] A. Flavel, P. Mani, D. Maltz, N. Holt, J. Liu, Y. Chen, and O. Surmachev. FastRoute: A Scalable Load-Aware Anycast Routing Architecture for Modern CDNs. In *NSDI '15*.

[24] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, and R. Weber. Pushing CDN-ISP Collaboration to the Limit. *SIGCOMM CCR '14*.

[25] M. J. Freedman, E. Freudenthal, and D. Mazieres. Democratizing Content Publication with Coral. In *NSDI '04*.

[26] M. J. Freedman, K. Lakshminarayanan, and D. Mazières. OASIS: Anycast for Any Service. In *NSDI '06*.

[27] M. J. Freedman, M. Vutukuru, N. Feamster, and H. Balakrishnan. Geographic Locality of IP Prefixes. In *IMC '05*.

[28] J. Hiebert, P. Boothe, R. Bush, and L. Lynch. Determining the Cause and Frequency of Routing Instability with Anycast. In *AINTEC '06*.

[29] A. Jain, J. Mann, Z. Wang, and A. Quach. W3C Resource Timing Working Draft. http://www.w3.org/TR/resource-timing/, July 2015.

[30] D. Katabi and J. Wroclawski. A Framework For Scalable Global IP-anycast (GIA). *SIGCOMM CCR '00*.

[31] M. Levine, B. Lyon, and T. Underwood. Operation Experience with TCP and Anycast. NANOG 37. June, '06.

[32] W. Li, R. K. Mok, R. K. Chang, and W. W. Fok. Appraising the Delay Accuracy In Browser-based Network Measurement. In *IMC '13*.

[33] Z. Liu, B. Huffaker, M. Fomenkov, N. Brownlee, et al. Two Days in the Life of the DNS Anycast Root Servers. In *PAM '07*.

[34] Z. M. Mao, C. D. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang. A Precise and Efficient Evaluation of the Proximity Between Web Clients and Their Local DNS Servers. In *USENIX ATC '02*.

[35] E. Nygren, R. K. Sitaraman, and J. Sun. The Akamai Network: A Platform for High-performance Internet Applications. *SIGOPS '10*.

[36] J. S. Otto, M. A. Sánchez, J. P. Rula, and F. E. Bustamante. Content Delivery and the Natural Evolution of DNS: Remote DNS Trends, Performance Issues and Alternative Solutions. In *IMC '12*.

[37] I. Poese, B. Frank, B. Ager, G. Smaragdakis, S. Uhlig, and A. Feldmann. Improving Content Delivery with PaDIS. *Internet Computing, IEEE '12*.

[38] S. Sarat, V. Pappas, and A. Terzis. On the Use of Anycast in DNS. In *ICCCN '06*.

[39] N. Spring, R. Mahajan, and T. Anderson. The Causes of Path Inflation. In *SIGCOMM '03*.