Enric Pujol, Ingmar Poese, Johannes Zerwas, Georgios Smaragdakis, Anja Feldmann

Steering hyper-giants' traffic at scale

Enric Pujol, Ingmar Poese, Johannes Zerwas, Georgios Smaragdakis, and Anja Feldmann. 2019. Steering Hyper-Giants' Traffic at Scale. In The 15th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '19), December 9-12, 2019, Orlando, FL, USA. ACM, New York, NY, USA, pp. 82–95. https://doi.org/10.1145/3359989.3365430

Steering Hyper-Giants' Traffic at Scale

Enric Pujol
BENOCS
epujol@benocs.com

Ingmar Poese
BENOCS
ipoese@benocs.com

Johannes Zerwas
TU München
johannes.zerwas@tum.de

Georgios Smaragdakis
TU Berlin
georgios@inet.tu-berlin.de

Anja Feldmann
Max Planck Institute for Informatics
anja@mpi-inf.mpg.de

ABSTRACT

Large content providers, known as *hyper-giants*, are responsible for sending the majority of the content traffic to consumers. These *hyper-giants* operate highly distributed infrastructures to cope with the ever-increasing demand for online content. To achieve commercial-grade performance of Web applications, enhanced end-user experience, improved reliability, and scaled network capacity, *hyper-giants* are increasingly interconnecting with eyeball networks at multiple locations. This poses new challenges for *both* (1) the eyeball networks having to perform complex inbound traffic engineering, and (2) *hyper-giants* having to map end-user requests to appropriate servers.

We report on our multi-year experience in designing, building, rolling-out, and operating the first-ever large scale system, the *Flow Director*, which enables automated cooperation between one of the largest eyeball networks and a leading *hyper-giant*. We use empirical data collected at the eyeball network to evaluate its impact over two years of operation. We find very high compliance of the *hyper-giant* to the *Flow Director*'s recommendations, resulting in (1) close to optimal user-server mapping, and (2) 15% reduction of the *hyper-giant*'s traffic overhead on the ISP's long-haul links, i.e., benefits for both parties and end-users alike.

KEYWORDS

CDN-ISP collaboration, traffic engineering, inter-domain, cross-layer, operational experience

1 INTRODUCTION

The phenomenal growth of the Internet has been driven by the ever-growing demand of users to access online content, including video, and social networks [17, 59]. In recent years, large companies, also referred to as *hyper-giants* [44] have been consolidating and increasing their presence on the Internet to serve this demand. Providing Internet-based services at scale with high quality of experience is challenging for several reasons. First, Internet-based services need to account for sudden increases in the demand for popular content, which adds stress to both network links and content servers [37, 74]. Second, provisioning of content servers is difficult, especially when the user demand is volatile. Content servers may be far from the end users, thus, limitations of transport protocols reduce the achievable bandwidth and increase the download time [24]. Finally, the economic model of peering is optimized for revenue increase and cost reduction, not for performance. Data over the Internet does not always follow the optimal path and in many cases it must travel over numerous autonomous networks [45].

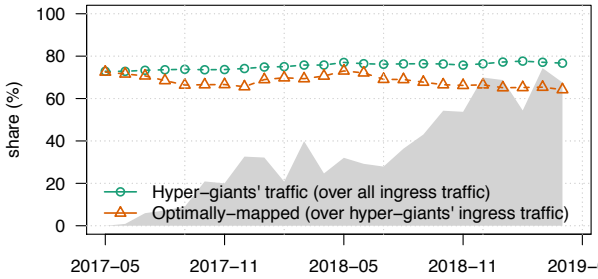


Figure 1: Traffic statistics in a large eyeball network. Gray area illustrates the traffic growth (%) with respect to the first data point (May 2017).

Content delivery networks (CDNs) [26, 45, 51] were introduced to address the aforementioned problems and achieve commercial-grade performance of Internet applications. This can be realized using different architectures [45, 69, 70]. Some of them, e.g., Limelight, deploy servers at data centers that provide good connectivity with a number of networks. Others, e.g., Akamai, rely on a large number of a highly distributed server clusters at multiple data centers deep inside the networks. Large companies that generate and deliver content like Google [24, 32, 77], Facebook [65], Microsoft [10], Amazon [11], and Netflix [50] created their own CDNs. This approach allows them to customize and to optimize content delivery for their needs and assets [36, 45, 60, 68]. Given the scale at which they operate, it can also reduce their costs. Moreover, CDNs establish direct peering relationships [23] with thousands of large and small eyeball networks, and in many cases at multiple locations [3, 23, 28, 29, 33, 70, 75], sometimes even inside ISP networks [13, 24]. However, ISPs have differing policies [20, 47].

In Figure 1, we plot traffic statistics for one of the largest eyeball networks world-wide, with which we established a collaboration, over the last two years. Due to confidentiality, we report the ingress traffic growth (gray area). The reference data point is the total ingress traffic in May 2017. The total ingress traffic increased linearly by around 30% per annum. This is consistent with other reports on the growth of household traffic (compound annual growth rate), e.g., Cisco Systems reports that the household's annual growth after 2017 is about 26% [15]. Labovitz et al. [44] reported that in 2009 the top 130 networks were responsible for 50% of traffic, however since 2016 less than ten networks are responsible for around 90% of many eyeball networks' traffic [7].

Like many other eyeball networks [7, 44], the growth and the majority of traffic for this eyeball network is dominated by *hyper-giants*. We hereby refer to a **hyper-giant** as any organization that meets

the two following conditions: (1) it sends at least 1% of the total traffic delivered to broadband customers in the ISP's network, and (2) publicly identifies itself as a CDN, or is registered as a "content" or "enterprise network" in the public database PeeringDB [46, 54]. [1] The reason for choosing the 1% threshold strives from the long-tail distribution of the share of traffic per organization, i.e., the top-10 *hyper-giants* sum up ~75% of the ingress traffic (dashed lines with circles), while the top-20 ~80% (not shown). We find 1% a good approximation for the *knee of the curve* of our data set. The resulting list of *hyper-giants* is a subset of the list reported by Böttger et al. [6], with the exception of one *hyper-giant* that is clearly recognizable as a CDN by its name and is not registered at PeeringDB.

When we turn our attention to these *hyper-giants*' share of optimally-mapped traffic, i.e., the *mapping compliance*, in Figure 1, we notice a significant reduction. By optimally-mapped traffic, we refer to the traffic that is delivered to the consumer via the closest *hyper-giant*-ISP peering location. A traffic-based *mapping compliance* is more relevant for ISPs than an analogous metric using customer identifiers, e.g., IP addresses, because the former captures the costs for the ISP for delivering content to the end users better than the latter. As shown in Figure 1, the majority of the *hyper-giants*' traffic is optimally mapped, a merit of their mapping components. However, the share dropped from 75% in May 2017 to around 62% in April 2019 despite the continuous joint efforts in upgrading network peering capacity and server infrastructures.

The user-mapping problem for CDNs. Large *hyper-giants* that operate highly distributed CDNs have reported on the problem of assigning end-user requests to the appropriate server. Although the DNS server of an end-user is typically used as a proxy to determine the location of this end-user, many studies have shown that this location is not representative [1, 63, 64]. IP geolocation tools are often insufficient for accurate geolocation at the city-level [27, 38, 58]. Regardless of whether the estimation of the end-user's location is correct, the routing of traffic between server and end-user may be sub-optimal [24, 45]. In this context, Akamai collects measurements on a regular basis and creates topology maps to deal with the problem [48, 51]. Google [42, 77], Facebook [65], and Microsoft [10] are using sophisticated edge architectures and aggregate/de-aggregate prefixes to better map users to their servers. EDNS-Client-Subnet EDNS0 extension (ECS) [9, 18] can also improve user-mapping [12], however it requires large efforts by the DNS server operators (including third party and ISPs) and the *hyper-giants* [71]. Even then, maintaining optimal mapping remains challenging [22]. As our results in Figure 1 show, the user-mapping accuracy has decreased over the last two years, i.e., fewer users are served by the closest server cluster.

Implications of a "bad" user-mapping. The negative externalities of bad mapping are visible to *hyper-giants*, ISPs, and end-users alike. For *hyper-giants*, the lack of optimal mapping may lead to delays and revenue reduction [41] as well as engagement [21, 43]. For ISPs, bad user-mapping translates to (1) higher infrastructure costs, e.g., link capacity upgrades, (2) congestion in the backbone network and thereby higher risk of affecting unrelated third-party traffic (of other users and service providers) [57]. ISPs are also faced with the challenge of having little means of control over which router traffic ingresses into their network, a problem that is known as the inbound traffic engineering problem [34]. Finally, end-users' perceived quality of experience deteriorates [45].

CDN-ISP collaboration. CDN-ISP collaboration has been explored in the past to alleviate the effects of bad mapping, and thereby reaching a win-win situation. One approach is to jointly deploy the servers, either by licensed CDNs [2], or by deploying caches of CDNs within the network [32, 50], and by utilizing micro data centers [25, 72] in locations selected by the ISP. However, many ISPs, especially large ones, object deploying CDN servers in their networks [9, 20, 39, 71]. Other solutions [4, 56, 57, 76] suggest establishing an off-band communication channel between a CDN and an ISP to exchange recommendations, i.e., mappings of user blocks to server clusters. The ALTO/P4P approach [4, 76] shares maps between two parties. The P4P design aims at optimizing P2P data exchanges, while ALTO extends the idea and specifies a protocol that enables networks to exchange maps of their network resources. PaDIS [56, 57], on the other hand, uses a dedicated service within the ISP that collects routing and link utilization information to maintain an up-to-date state of the network activity. The difference between ALTO and PaDIS is that the CDN contacts the PaDIS service when it has to map users to appropriate servers, and ALTO provides an information exchange. To our knowledge, neither one of them has been tested in a real operational environment at scale.

In this paper, we report on our experience in building, rolling-out, and operating the first-ever large scale system to enable CDN-ISP collaboration inspired by the ALTO/P4P and PaDIS research projects. We deploy our solution in one of the largest eyeball networks in the world with more than 50 million subscribers, and we steer the traffic of one of the largest *hyper-giants* in the world. The contributions of this paper are as follows:

- We present the system architecture and implementation of the *Flow Director*, which enables automatic exchange of recommendations between the ISP and the *hyper-giant*. *Flow Director* collects and analyzes more than 45 billion NetFlow records per day from more than 1000 exporters, correlating them with BGP information from around 600 tables to keep track of the path and ingress points for content that *hyper-giants* injected into the ISP's network and provide recommendations to improve user-mapping.
- We use empirical data collected at the eyeball network to evaluate the performance and impact of *Flow Director* over two years of operation. Our results show that the compliance of the *hyper-giant* to the eyeball network's recommendations is very high. This yields (1) close to optimal user-server mapping, and (2) reduction of this *hyper-giant*'s traffic overhead on the ISP's long-haul links of up to 15%. These observations underscore the benefits of the *Flow Director* for both parties and end-users alike.
- We discuss the challenges of turning an idea that appeared in a research paper to a fully operational system in today's Internet.

2 ISP PROFILE

To understand the mapping challenges and the benefits of collaborations between large eyeball networks and a leading global *hyper-giant*, we deployed the *Flow Director* in a Tier-1 ISP with several tens of millions of broadband and mobile subscribers. This

[1] We opt to use the term *organization* to reflect that a *hyper-giant* can manage multiple Autonomous Systems.

Customers (land & mobile lines)	> 50 million
Daily traffic	> 50 PB
Backbone routers	> 1000 (MPLS)
Links (long-haul / all)	> 500 / > 5000
Points-of-Presence (PoPs)	>10

Table 1: Targeted eyeball ISP statistics.

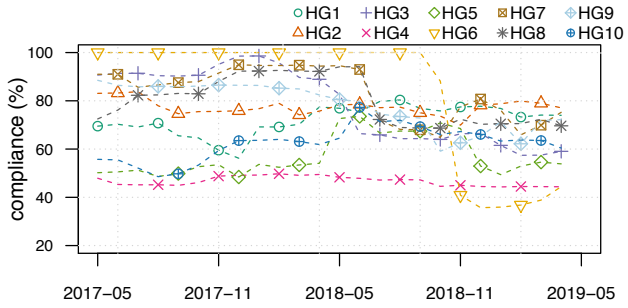


Figure 2: Share of optimally-mapped traffic of top 10 *hyper-giants* over time.

ISP serves more than 50 PB of traffic daily, see Table 1. It no longer allows *hyper-giants* to deploy caches within the ISP. Instead, *hyper-giants* connect mainly via private network interconnects (PNIs) at more than 10 different Points-of-Presence (PoPs) in the ISP's home country and more than 5 international PoPs. The ISP's backbone consists of more than 1, 000 routers out of which several hundred are customer facing, i.e., forwarding traffic to the end-users. The other routers realize inter-PoP connectivity via more than 500 long-haul links. Routing within the ISP's network, e.g., from the ingress point of a *hyper-giant* to an end-user is realized using MPLS and ISIS. The ISP uses both IPv4 as well as IPv6.

For the testing and deployment of *Flow Director*, the ISP granted us access to several data sources including the router inventory along with their geographic locations, IGP feeds, SNMP feeds, BGP and NetFlow data collected at the ISP's border routers from mid-2017 until now. Using router locations along with IGP data allows us to determine path-lengths, and thereby, approximate latency. By combining all of the data sources, we can compute the traffic matrix including how much traffic from which *hyper-giant* to which destination prefix is traversing the network, for details see Section 4. The busy hour for this ISP is at 20:00 local time—the sample we use for daily resp. weekly comparisons.

3 OPTIMAL MAPPING: CHALLENGES

In this section, we address the questions: how optimal are the mappings of various *hyper-giants* for the ISP, and what are the impediments for improving them? Complications include (1) *network topology changes* within the ISP as well as new peering, (2) *routing changes* within the ISP e.g., due to traffic engineering, or at the *hyper-giant* due to mapping changes e.g., to either increase or decrease utilization of server clusters as well as (3) *address-space changes* within the ISP e.g., to reclaim/reassign scarce address space, or at the *hyper-giant* e.g., for better load balancing. In this regard, we show that there are different mechanisms for each operation at various time scales making an optimal mapping difficult to achieve.

3.1 Mapping compliance: Fraction of optimally-mapped traffic

Our first question is about how the share of optimally-mapped traffic per *hyper-giant* changed over time.[2] Hereby, we focus on the top-10 *hyper-giants* in terms of traffic share that have peering agreements with the ISP, most of which exclusively use PNIs. Figure 2 shows how the mapping compliance per *hyper-giant* has developed over the last two years based on monthly averages of the daily busy-hour traffic matrix. To obtain the **mapping compliance**, we calculate the ratio of the *hyper-giant*'s optimally-mapped traffic to its total traffic. Here, **optimal mapping** means that the *hyper-giant* sends traffic to the content consumer via the **best ingress PoP** i.e., the PoP with the shortest path to the consumer. The metric for the cost of a path is a combination of number of hops and physical link distance as agreed with the ISP. While this metric is the best-suited for this particular ISP, *hyper-giants* may not be able to obtain a perfect score e.g., due to overloaded servers, content availability, or limited peering capacity at the best ingress PoP.

We observe that one *hyper-giant* (HG6) drops from 100% to less than 40%. Initially, it only peered at a single location. Once it added additional locations, mapping became relevant, however, it was not calibrated. Other drops also show a correlation with times when the *hyper-giants* added peering locations, which imposes additional mapping challenges. In the case of *hyper-giant* (HG4), we see a fairly stable share of optimally mapped traffic at about 50%. Closer inspection shows that this *hyper-giant* is using round robin load-balancing, which is detrimental for optimal mapping. For other *hyper-giants*, we see fluctuations between 50% and 95%, either increasing or decreasing, which underlines the challenges of deriving and updating optimal mappings. For most *hyper-giants*, the share of optimally-mapped traffic has declined within the observation period.

There are, however, two exceptions: One is the *hyper-giant* that collaborates with the ISP (HG1; blue-green circles), where we can see an increase in the optimally-mapped traffic. The other one (HG2; orange triangles) is a *hyper-giant* that has at times re-adjusted its mapping based on hints from the ISP.

3.2 *Hyper-giant*/ISP connectivity changes

As discussed earlier, adding additional PoPs can impede optimal mapping. Thus, Figures 3 resp. 4 show how often *hyper-giants* add new PoPs or increase their peering capacity. For the latter, we sample the SNMP feeds every 5 minutes and compute the median value per month. For most *hyper-giants*, the number of PoPs as well as the nominal peering capacity is monotonically increasing, thus underlining the continuous investment by the *hyper-giants* and the ISP, both in terms of footprint as well as capacity. While these are planned changes that a *hyper-giant* should be able to account for, we see that this is not always the case as most changes in Figure 2 are correlated with changes in Figures 3 resp. 4.

Six of the *hyper-giants* added peerings in new PoPs, and two increased the number of presences twice (HG3 and HG7). In both cases, they waited for more than 6 months before the second increase. There is one outlier, which reduced its presence (HG7),

[2]Throughout this paper we use the term *mapping compliance* to refer to an *hyper-giant*'s share of optimally-mapped traffic.

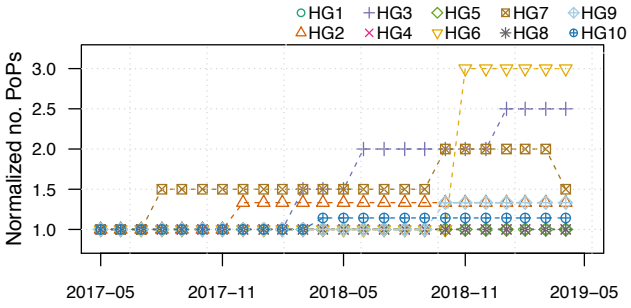


Figure 3: Number of PoPs for the top 10 *hyper-giants* over time (normalized by initial number of PoPs).

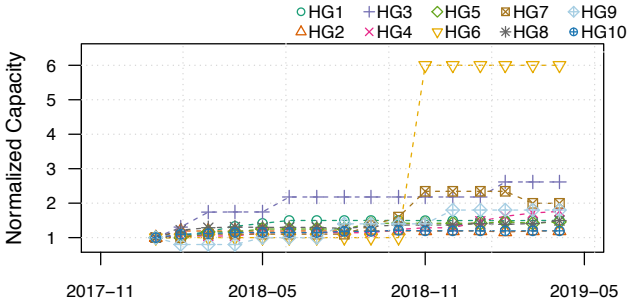


Figure 4: Peering capacity for the top 10 *hyper-giants* over time (normalized by the initial capacity).[3]

however, as expected, its mapping compliance increased. Most *hyper-giants* increased their capacity by at least 50%. Noticeably, HG6 increased its capacity by 500% while also increasing the number of PoPs. In fact, HG6 changed its delivery strategy by utilizing a meta-CDN to rolling out its own serving infrastructure (as in [35]). Overall, most changes occur several weeks or months apart.

3.3 Intra-ISP topology/routing changes

Next, we explore how often the "optimal" ingress PoP varies due to intra-ISP changes. Potential changes include link changes (physical as well as logical–MPLS ones) as well as ISIS weight changes, which change the intra-ISP routing. While such events are common, they may or may not impact the optimal mapping of any particular *hyper-giant*.

To identify times when the optimal mapping changes, we use daily snapshots of the ISP's routing information. While we may miss some short term traffic engineering events, this nevertheless allows us to capture major upgrades. We then calculate the time difference between such events. Note, it cannot be less than 1 day. Figure 5(a) shows a quartile boxplot for each of the top-10 *hyper-giants* using a logarithmic y-axis. For reference, we added support lines for 1 and 2 weeks. Overall, the median time between changes for most *hyper-giants* is in the order of weeks. This means that they may have to adjust their mapping on this time scale. For some of the *hyper-giants*, the time scale is either smaller or larger, which is partly correlated with their number of PoPs. Moreover, some PoPs have experienced more churn than others, which may impose more churn on those *hyper-giants* that are present at these PoPs.

[3]Due to missing data, the X-axis is different in this plot.

Next, we ask how many potential users may be affected by intra-ISP routing changes. We calculate for each *hyper-giant* what fraction of the ISP's announced IPv4 space changed its "optimal" ingress PoP. Hereby, we focus both on relatively short-term changes (1 day) as well as long-term changes, 1 resp. 2 weeks. The latter helps us understand if the changes are persistent or only temporal. Figure 5(b) shows the corresponding quartile boxplot per *hyper-giant* for all three timescales. Typically, each change affects less than 5% of the ISP's address space. There are some outliers that range up to 23%, however, almost all changes affect less than 10% of the IP space. Interestingly, the time offset has an impact. For some *hyper-giants*, the affected address space increases as we move to larger time differences, while it decreases for others. Overall, we do not find a consistent pattern, an important observation for both the *hyper-giant* and the ISP.

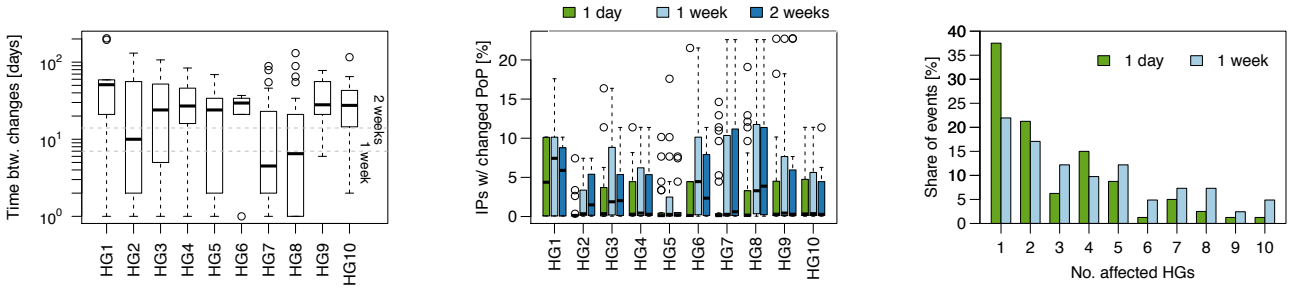
Given that some *hyper-giants* share peering locations, we next ask if a particular routing change affects only a single *hyper-giant* or multiple ones. Figure 5(c) shows the resulting histogram using both the 1 day as well as the 1 week offsets. Most of the changes (> 35% for 1 day and > 20% for 1 week) only affect a single *hyper-giant*, however, a significant share (> 5% for 1 day and > 10% for 1 week) affects 8 or more *hyper-giants*. Short-term changes (1 day) affect a smaller number of *hyper-giants* than more persistent (1 week) changes. These observations show that optimizing for a specific *hyper-giant* may influence others both positively and negatively, which renders routing-based optimizations challenging. Furthermore, when the "optimal" ingress PoP alters due to modifications in the routing or the topology, all affected *hyper-giants* need to adjust their mapping system to leverage this new situation.

3.4 ISP's IP distribution: PoP changes

Other reasons for changes in the "optimal" ingress PoP for an IP are reassignments of end-user IP prefixes to a different PoP e.g., for operational reasons, or shared DHCP pools, e.g., [53, 62]. This is expected given the sparseness of IPv4 address space.

Using daily snapshots of IGP data, we calculate the number of IPv4 /32s resp. IPv6 /56s prefixes -we refer to both as IPs- that are allocated to each PoP (after removing black-holes and accounting for more specifics and prefix delegations). We determine three events: IPs can (1) be newly announced, (2) be no longer announced, or (3) have changed their announcing PoP. We aggregate these events on a daily basis. Figure 6 shows a time-series of the maximum churn (the sum of all three events) within a month for IPv4 and IPv6. We immediately see that (a) there is significant change and (b) the ISP manages both address families differently. IPv6 exhibits more pronounced bursts, while the maximum daily churn of IPv4 is more uniform across time.

Next, we focus on the fraction of PoP changes after some days. In particular, we ask if more than 1 resp. 5% of the ISP's IPs changed their announcing PoP within X days. Figure 7 shows the resulting empirical cumulative density function (ECDF). The IPv4 address space has frequent changes: the likelihood of a 1% change within 14 days is more than 90%. Further investigations suggest that coordinated surges occur mostly on Thursdays, which are



(a) Time between changes in best ingress PoP due to intra-ISP routing.

(b) % of announced ISP IPv4 address space with best ingress PoP changes.

(c) # of affected top-10 *hyper-giants* per intra-ISP routing event affecting best ingress PoP.

Figure 5: Quartile boxplot/histograms for top-10 *hyper-giants* to underline impact of connectivity changes.

then followed by periods without changes (i.e., weekends). A frequent modus operandi is a withdrawal of some IPv4s followed by a re-announcement several weeks later at a different PoP.

3.5 Correlation across *hyper-giants*

Given all of the various challenges for *hyper-giants* to compute optimal mappings on their own, we seek to find correlations in their share of optimally-mapped traffic across the top-10 *hyper-giants* over our two year observation period. Figure 8 is a heatmap of the correlation matrix of the time series displayed in Figure 2, whereby, we grouped the *hyper-giants* into clusters to underscore their correlation patterns. Intuitively, a positive correlation implies that the *hyper-giants*' mapping systems react similarly e.g., to changes in the ISP's IP address space. A negative correlation, on the other hand, may imply that some changes are beneficial for some *hyper-giants* while detrimental for others e.g., traffic engineering within the ISP.

Overall, we see more positive–larger–than negative–and smaller–correlations. The data shows that positive correlations often appear when the *hyper-giants* share PoPs, and negative ones appear when their PoPs sets are different therefore, underlining that PoP diversity and location play a crucial rule in the mapping compliance.

3.6 Takeaways

We find that a significant fraction of the traffic of most *hyper-giants* is not optimally-mapped, whereby decreases in mapping compliance is often correlated with topological changes, e.g., additional PoPs or capacity upgrades, which occur at the time scale of months. Topological changes are, on the other hand, more common events. In this context, intra-ISP routing changes that affect the optimal-mapping occur on a weekly basis and often involve a sizable fraction of the ISP's address space. Moreover, while more than 20% of the changes only impact a single *hyper-giant*, some affect a majority of *hyper-giants* simultaneously. Lastly, this ISP has noticeable churn in its IP to PoP assignment (> 1% change within less than 2 weeks for IPv4) with peaks at 4 and 15% for IPv4/v6.

All of the effects described above hinder *hyper-giants*' pursuit of determining an optimal mapping on their own. While *hyper-giants* traditionally orchestrate sizable active-measurement campaigns to determine which server is closest to a consumer, this is challenging and will often be misleading. Still, a reasonable trade-off between effort and accuracy for such a measurement campaign may be on a daily to weekly basis.

4 *FLOW DIRECTOR* (*FD*)

FD is an ISP service enabling ISP-*hyper-giant* collaboration with the goal of improving the latter's mapping. To this end, *FD* uses extensive network information to compute recommendations to assist the *hyper-giant*'s mapping system. Thus, realizing *FD* within ISPs requires a design that integrates well with ISPs' infrastructure while also being flexible and scalable.

4.1 Design challenges

The main challenge of *Flow Director* (*FD*) is not necessarily the task of computing the recommendations for the *hyper-giant*, but rather collecting and processing the necessary data at scale to be able to compute these recommendations.

Conceptually, to compute a recommendation, *FD* must first determine the cost of delivering traffic from any *hyper-giant* IP to any consumer IP. To do so, *FD* must first determine the ISP's topology from *intra-AS routing protocols* to infer the cost to deliver traffic from PoPs with *hyper-giants*' peerings to the consumers' PoPs and then rank them accordingly. Given that ISPs typically only enable traffic sampling on the ingress routers to avoid monitoring the same packet multiple times and to reduce the overhead, *FD* needs to be able to determine this cost by using *ingress flow data* e.g., NetFlow [14]. There are several steps involved to achieve this. First, *FD* needs to know the distribution of IP addresses behind each peering (*ingress-point detection*). Then, it needs to infer the path over which an edge router sends traffic to a consumer from *intra-*, and *inter-AS routing* data. Since IGP does not carry consumer prefixes, *FD* also requires BGP data to identify the next hop for a consumer IP. This presents a scaling challenge because BGP is a distributed information-hiding protocol [8], and thereby *FD* needs *all* routes from *all* BGP routers. Thus, *FD* has to correlate hundreds of millions of routes from hundreds of BGP sessions with millions of NetFlow records per second, which requires a highly scalable design.

FD works on a diverse set of inputs and needs the capability to (1) tackle different stream volumes, (2) distinguish failures from time lags and gracefully handle them, and (3) handle failures. To provide maximum flexibility *FD* should be deployable anywhere in the ISP: on already existing cloud solutions inside the ISP as well as on generic hardware e.g., to enforce policies regarding network appliances or cloud service independence. Therefore, *FD* has to support public cloud services but cannot depend on their APIs as these may not be supported by bare-metal hardware, or supported

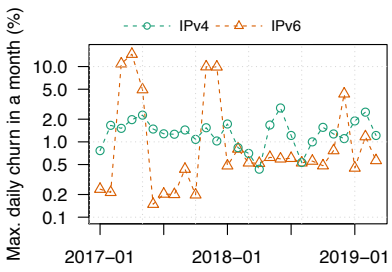


Figure 6: Maximum observed daily churn in customer prefix assignment to PoPs within a month.

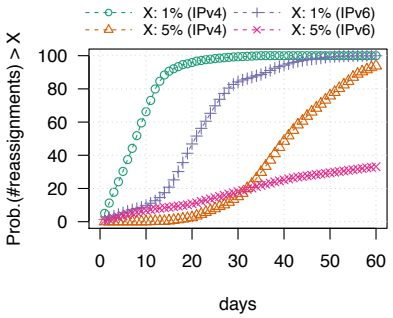


Figure 7: ECDF: % of the customer prefixes with changes in assigned PoP by duration.

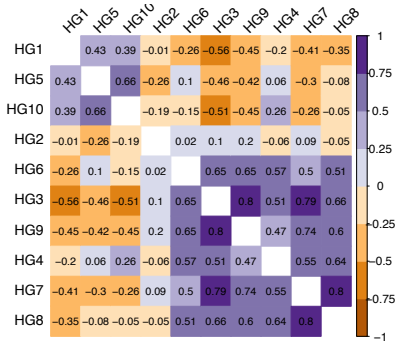


Figure 8: Correlation matrix of optimally-mapped traffic of *hyper-giants* over 2 years.

within ISP-clouds and data centers. We note that the bare-metal deployment use-case is the most constraining one with regards to resource efficiency because it is difficult to scale dynamically. Upgrading or deploying new hardware within an ISP network typically takes weeks, if not months. Moreover, since ISP networks include a diverse set of hardware from multiple vendors with various different interfaces, *FD* should avoid vendor specific implementations to infer and track the network state. Thus, *FD* uses standardized control plane protocols to infer the network state, which all vendors have to support.

4.2 System architecture

The overall system architecture of *FD*, see Figure 9, is similar to that of other data transformation systems. A `Core Engine` takes information from the network through a set of *southbound interfaces* called *listeners*, via *Aggregators* and publishes them into its internal network database (*DB*). Each southbound interface is generic, in the sense that it is replaceable without changes to the core. Thus, to adapt *FD* for an ISP that uses ISIS rather than OSPF, only the *listener* responsible for intra-AS routing has to be touched.

For the northbound interfaces (called interfaces), the `Core Engine` uses "consumer" plugins that derive their results based upon the `Core Engine`'s data and that can trigger events to publish them to subscribers. These plugins are generic in the sense that the subscriber can be one or multiple *hyper-giants* or some other system e.g., an ISP internal one.

The `Core Engine` stores a representation of the network and its state in a consumer agnostic model. Internally, it uses a graph representation of the network to store the incoming data and triggers the plugins for further data transformation as needed. For scalability reason *FD* separates the global Internet reachability from the internal network topology. This (1) reduces processing time and (2) enables parallelization by pushing the computation to another process (which may run on a different server).

4.3 Implementation

Figure 10 expands on the generic architecture from Figure 9. Each block refers to a class of processes that are realized either as threads and/or processes that can be distributed across multiple machines. Each process performs data transformations.

4.3.1 *Southbound interfaces.*

Intra-AS routing protocols: We choose to implement one *listener* per protocol, which allows for flexibility when changing to different protocols for the same task i.e., the ISIS logic is encapsulated in the `ISIS listener`. Note, each *listener* runs its own code base and communicates only with the `Aggregator` of the `Core Engine`.

BGP—the inter-AS routing protocol: To use BGP to replicate routing decisions, *FD* needs full BGP information from all eBGP routers. Even route-reflectors are insufficient as they already perform best path selection and, thus, do not forward all routes to their clients. Possible alternative such as, BGP ADD-PATH [73] or BMP [66], are also infeasible. BGP ADD-PATH only forwards a fixed number of alternatives, and BMP is not yet widely deployed and requires every BGP instance on the path to export all routes (including discarded ones). *FD*'s `BGP listener` achieves full visibility by receiving the full FIB of each router—essentially, it is a route-reflector client of every router. The large number of routes per router imposes a huge strain on system memory for any reasonably sized ISP ($> 850k$). Since all tested existing BGP implementations, i.e., Quagga [61] and Bird [19] are not suited for our use case (full FIBs from all neighbors), *FD* includes a custom implementation supporting *cross router route de-duplication* to optimize memory consumption. However, even with this optimization *FD*'s `BGP listener` often uses multiple hundreds of Gigabytes of RAM. Scaling it horizontally is ineffective as the level of route replication decreases unless fast cross machine RAM access is available.

Traffic flows exports: Currently, the best available method for capturing traffic flows at scale on many routers is with traffic sampling protocols such as sFlow [52], NetFlow [14], IPFIX [16], etc. Carrier-grade routers easily generate millions of records per second, which are received at a flow monitor via unordered, unreliable UDP packets. Yet, the *FD*'s `Core Engine` needs a well-formatted, de-duplicated, in-order flow data stream. To solve this problem, we use a pipeline of standalone tools. This tool-chain starts with `uTee` [30], a custom tool that splits the input flow stream into *n* load-balanced streams based on byte count and a flow schema template of `nfacct` [55]. Each `nfacct` instance converts its stream into a standardized, internal format. The resulting stream is pipelined to `deDup`, which (re-)combines multiple flow streams while removing duplicates to avoid double counting-into a single flow stream.

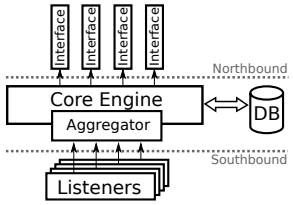


Figure 9: *Flow Director*: High-level system architecture.

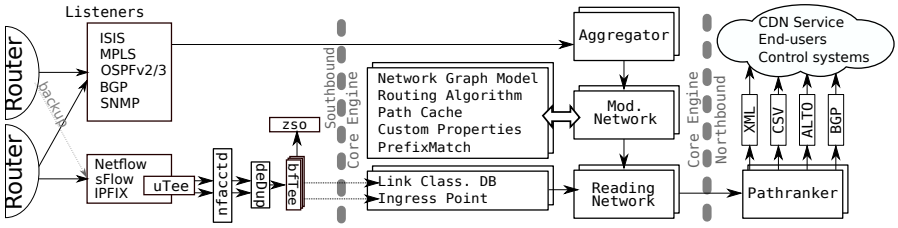


Figure 10: *Flow Director* processing pipeline.

Next, the data is piped into a set of `bfTee` instances [31]. `bfTee` is a reliable, in-order, stream based, lock-free flow duplication tool. We use it to protect *FD* against back pressure e.g., due to time lags or failures, and as an intermediate buffer. Each `bfTee` has two output streams: reliable and unreliable. The reliable one blocks on unsuccessful writes, while the unreliable -but buffered- one discards data when its internal buffer is full. The reliable stream (solid line) ultimately writes to a slightly modified version of `zso`, which is a data rotation tool for disk storage (time based rotation was added). The unreliable streams (dotted lines) are used for sending data to *FD*. In fact, two independent `Core Engine` plugins receive stream duplicates. Due to this setup one process cannot block the other in case of slow processing and/or failures. The remaining `bfTee` instances are used for testing, debugging and research purposes. Due to the implementation, new code can be integrated into the live stream at any time without having any effect on the production system while getting full stream data.

4.3.2 *Core Engine.*

The `Core Engine` is a network database that maintains a directed, weighted -per link direction- (network) graph called `Network Graph`. It distinguishes three types of nodes (router, virtual nodes and broadcast_domain), each identified by a unique ID. In its basic form `Network Graph` merely represents what the IGP of the network supplied, however, more information is needed. This is done by graph annotation using `Custom Properties` supplied by additional *listener* plugins to accommodate a collection of non-routing/topological information. For example, an ISP can use its OSS/BSS system to feed SNMP, Telemetry, or contractual information. At the same time, CDNs can supply metadata like cluster capacities or content availability. Internally, each custom property consists of a data type, attached values, one or more nodes/links, and an aggregation function. The aggregation function is used to aggregate values along the path to calculate its properties e.g., shortest path based on per link distances or less congested path based on link-utilization values. Using `Network Graph`, the `Core Engine` offers various functions, namely, `Routing Algorithm`, which fills the `Path Cache` including all of its pre-computed `Custom Properties`, as well as `prefixMatch` and the `Ingress Point Detection`, which use `Link Classification DB` (details follow).

Lock-free implementation: To allow lock-free access to the network graph database for many processes asynchronously, the `Core Engine` uses two representations: the *Modification* and the *Reading* `Network Graph`. All reads are handled by the *Reading Network*, while all updates (received via the `Aggregator` resp. the flow processing pipeline) are applied to the *Modification Network*. The `Aggregator` is the gatekeeper to the internal databases and triggers updates of the *Reading Network*. Lock-free access is essential as updates are a matter fact in the Internet. By using a *Modification Network*, we batch updates, whereby the minimum batch time is the time to generate a *Reading Network*, which is the "valid one" and is accessible via the northbound interface. Even in the largest deployment it is updated in under a minute when changes trigger a complete recalculation.

`prefixMatch:` The `Core Engine` offers `prefixMatch`, which aggregates routing information into *subnet prefixes*. The subnets are grouped by their attributes (i.e., BGP nextHop, Communities, etc.), enabling massive compression as compared to BGP. Note that `prefixMatch` attaches data to nodes in the topology but it does not affect or re-trigger calculations in `Network Graph` or `Path Cache`.

`Path Cache:` Since path search is time consuming the `Core Engine` uses a `Path Cache` plugin to reduce the overhead of path lookups. The `Core Engine` stores all pre-calculated paths determined via `Routing Algorithm` in the `Path Cache`, along with their `Custom Properties`. These only have to be updated if the IGP weight changes due to the separation of topology within `Network Graph` and Inter-AS routing information via `prefixMatch`. In addition, it uses multiple heuristics to keep paths that do not need to be recalculated from being updated. Note that this plugin chooses the specific IGP flavor by selecting the correct `Routing Algorithm`.

`Link Classification DB (LCDB):` The `LCDB` is initially filled with data from the ISP via a custom interface and then augmented with SNMP data. Moreover, *FD* constantly monitors the flow stream and correlates it with BGP. Once a new link is detected (a fairly frequent event), it is either added manually or via the custom interface. In the end, the `LCDB` maintains all links in one of three defined roles: (1) inter-AS, (2) subscriber or (3) backbone transport link.

`Ingress Point Detection:` To determine a network path for a (potentially) external server, `Core Engine` needs the ingress router ID for every prefix. However, BGP does not offer such information. Thus, the `Core Engine` infers the mapping from the flow stream by, first, using the `Link Classification DB` to filter the flows stream captured on inter-AS interfaces. Then, it pins the flows' source IP addresses to the link ID. To reduce memory, `Ingress Point Detection` aggregates these potentially hundreds of millions of IPs per link ID to prefixes. A full consolidation is done every 5 minutes.

To underline the importance of tracking ingress points, we show in Figure 11 the churn in detected IPv4 prefixes per ISP PoP for the top-10 *hyper-giants* per 15 minute time bins. While the majority of the prefixes are stable, the churn of $\sim$ 200 prefixes is significant enough to potentially harm the *hyper-giants*' mapping. When looking at subnet sizes, see Figure 12, it is obvious that the driving

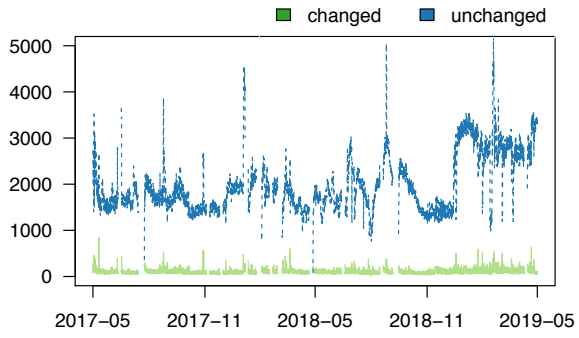


Figure 11: Timeline: 15 min. PoP level churn rate for IPv4 addresses identified by `Ingress Point Detection`.

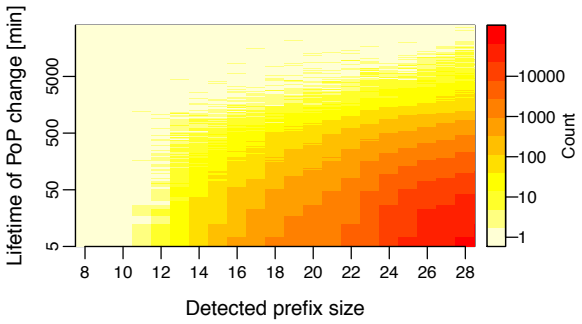


Figure 12: Heatmap: PoP changes vs. subnet sizes across two years.

force for the huge churn are small subnets. Still, even large subnets experience significant churn. Overall, this analysis underlines that ingress points constantly change e.g., due to changes in the *hyper-giants* mapping, server maintenance, BGP route changes, ISP IGP changes, etc. The `Ingress Point Detection` plugin enables their detection in almost real-time, which then enables re-assignment and re-routing of ingress traffic within minutes.

4.3.3 *Northbound interface – hyper-giant interface(s).*

The final part of the system is the northbound interface and, in particular, the interface to the *hyper-giants*. This is fed by the *Reading Network* to allow lock-free read access by any number of processes. The `Path Ranker` computes the "optimal" mapping from every ingress point for every internal subnet by taking advantage of the `Path Cache` to minimize path calculation. Hereby, the optimal function is agreed by the ISP and the *hyper-giant*, even though *Flow Director* only provides a recommendation to the latter. "Optimal" can differ per *hyper-giant* and e.g., involve any combination of hop count, physical distance, network distance, or other custom link properties. In the extreme case of the *hyper-giant* ignoring *FD*'s recommendation completely, the status-quo of no co-operation is maintained. This makes the *Flow Director* useless, but has no effect on any traffic either. The other side of the coin is that the *hyper-giant* follows the recommendation blindly. While this can be optimal for the ISP, it could potentially create a resource problem for the *hyper-giant*, if *Flow Director* does not have information about the *hyper-giant*'s capacity and content availability. To counteract this problem, the *hyper-giant* can supply this information to *FD*'s `Custom Properties` via its northbound interface. This would turn the *Flow Director* into a centralized and intermediate repository of information about the *hyper-giant* and ISP. Finally, `Path Ranker` can communicate its recommendation using the *hyper-giants*' interface of choice. Currently implemented interfaces include:

`ALTO-based interface` [40]: ALTO, at its core, defines two different types of mapping information. First, it creates the network map that defines clusters of network position identifiers (PIDs) e.g., routers, prefixes, etc. Attached to each network map are one or more cost maps, which define the pair-wise cost between each PID pair. In *FD* terms, this results in a general network map that segments the ISP's network, and one cost map per *hyper-giant* derived via `Path Ranker`. Note that not all PID combinations are needed by *hyper-giant* e.g., ISP internal connections. To reduce space, the cost map omits these PID combinations. In case a *hyper-giant* has different classes of content, multiple custom cost maps can be supplied, effectively splitting the *hyper-giant* into several parts. Finally, one important extension among ALTO's multiple extensions is the Service Side Events extension (SSE), which enables subscriptions to an ALTO network/cost-map for a secure push-based notification service implemented over a *RESTful* interface. Note that *Flow Director* only supplies the network and the cost map to the *hyper-giant*, while keeping topological and measurement information out of the maps as necessary.

`BGP-based interfaces` (e.g., [5, 49, 75]): BGP is a protocol designed to provide network-layer reachability information (NLRI). While operators can annotate network prefixes via BGP communities by design, BGP is not designed to associate a network prefix to another network prefix *directly*. However, BGP communities are still often used to associate or group prefixes together to realize this mapping. *Flow Director* can interface with a *hyper-giant* over out-of- or in-band BGP sessions. For example, in a BGP out-of-band session the *hyper-giant* can announce the prefixes of its servers, together with a cluster identifier encoded in the BGP communities via BGP. After receiving this information, *FD* announces back for each cluster ID the ISP's prefixes with a BGP-community with the server cluster ID encoded in the upper 16 bits and the ranking value for that cluster ID in the lower 16 bits. The case of an in-band session works similarly, with the added tricky part of possible collisions in the BGP-community values. In fact, the space for encoding mapping information is halved.[4] Furthermore, the *Flow Director* requires additional information from both parties, i.e., which communities are in use. These can be incorporated into the `Custom Properties` via a southbound interface that needs to be customly implemented.

`Customized interfaces`: The last scenario includes *hyper-giants* not offering an automated interaction interface. *FD* supports multiple output formats such as JSON/XML/CSV, which can be then forwarded to the relevant parties via file uploads, e-mail, etc.

4.4 Data problems and failovers

Whenever one operates a large scale system with multiple different data sources, problems occur, and things break. These can often

[4] BGP communities are 32 bit values.

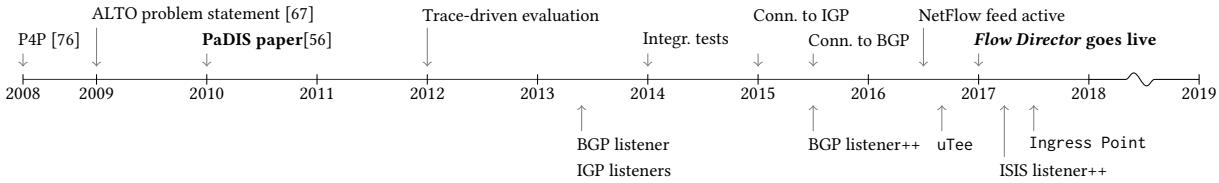


Figure 13: Timeline: From a research idea to a fully operational CDN-ISP collaboration. Top: Project management and infrastructure roll-out events. Bottom: *FD*'s development milestones and main overhauls (++).

be detected using cross correlations and fixed by initiating appropriate actions. For example, configuring BGP is tedious because each neighbor needs to be explicitly configured. With hundreds of routers, this is not only error-prone but also entails significant overhead. *FD* automates such tasks whenever possible. For example, when a new node is detected in the `Network Graph`, it can be set to automatically configure it as BGP peer with its loopback IP. In the same manner, connection aborts are distinguished from planned shut-downs.[5] *FD* monitors such events using a rule based system with appropriate thresholds to keep the network state up to date. Hereby, fast detection of errors and their resolution benefit the ability to correlate data- and control-plane information in real-time.

It is possible to run multiple `Core Engine` processes, e.g., for redundancy. In this case, each *listener*, except for the NetFlow one, connects to all `Core Engine` processes independently. For NetFlow (due to the volume of its data stream) we are using a floating IP that is assigned to all `Core Engines`. The IP is announced via the `IGP listener` and by choosing the metric appropriately it is possible to realize fail overs, load balancing, etc.

4.5 Milestones and lessons learned

To reach the successful first deployment of the *Flow Director*, its implementation underwent several iterations and extensions to gain enough scalability and flexibility such that it can accommodate the rich number of network protocols and data streams existing in large ISP networks. Figure 13 illustrates this process. While the initial system design and idea remain unaltered, from 2013 until 2017 we had to devise mechanisms to circumvent constraints and limitations that we did not anticipate. For example, shortly after establishing the first BGP connection with an edge router, we realized its memory footprint was too large. The `BGP listener` underwent a major overhaul to be able to scale up the number of routers to the order of hundreds. Likewise, while the first `ISIS listener` was *silent* for security reasons (LSPs announcements were disabled), we had to iterate its code basis to be able to support redundancy once we instantiated a second *FD*. Similarly, along with the need to scale up the NetFlow processing pipeline and thereby building a custom load-balancing infrastructure, we had to devise several data-sanity checks for NetFlow data, as it cannot be completely "trusted". For example, during cache flushes, reboots or when line-cards are replaced, updated or reconfigured, the resulting NetFlow timestamps might be in the future (up to several months) or in the past (we saw packets from every decade since 1970). Furthermore, even in "normal" operation, timestamping is skewed due to cache evicts and missing/faulty NTP synchronization. Another aspect that we did not anticipate is inconsistencies among inventories. These are usually manually maintained and thus prone to errors. Such inconsistencies are, in fact, the motivation behind the `LCDB`, which enables the `Ingress Point Detection` (another component we did not anticipate as well but is crucial for the correct operation of the *Flow Director*).

[5] A router shutting down withdraws its IGP information prior to shutdown. A router going into maintenance should set itself to overload, telling the IGP not to use it in its path calculation anymore. In contrast, a random connection abort does neither of these.

5 OPERATIONAL EXPERIENCE

Although the system was initially proposed within a research project in 2010, and first feasibility studies showed significant potential benefits (Figure 13: 2010–2014), it was not until 2015 before it was deployed in an ISP. It also required two additional years for *FD* to become fully operational and to get a contractual agreement with a *hyper-giant*.[6]

In July 2017, a major *hyper-giant* with geographic diverse peering connectivity and the ISP started their formal cooperation with the joint goal of shortening the paths from the ingress router of the ISP to its customers (a function of the hops and geographical distance). Since Spring 2018 the collaboration of the ISP and the *hyper-giant*'s mapping system is fully automated. The anticipated benefit for the *hyper-giant* is *reduced delivery times* hence, improved QoE metrics and, on the other hand, for the ISP is *reduced long-haul traffic load*, hence, reduced cost. It is important to note that *FD* does not force a *hyper-giant* to deliver traffic from the recommended server cluster. It can ignore the recommendation and instead serve content from the best cluster–according to its own metrics. In fact, the cooperating *hyper-giant* sometimes ignores *FD*'s recommendations, if its mapping system anticipates congestion for traffic crossing the recommended ingress points.

While the ISP system is prepared to interface with any willing *hyper-giant*, at this time, the northbound interface is used by only one *hyper-giant*. Still, we have used *FD* to manually improve the mapping of two other *hyper-giants*, recall Section 3, to incentivize them. In this section, we report on the existing fully-automated cooperation and discuss their benefits.

5.1 *Flow Director* deployment

The deployment of *FD* within the ISP is summarized in Table 2. We use two physically separated bare-metal servers each with 16 cores (64 hyper-threads) and more than 700 GB of memory to accommodate unforeseen peaks. Each server receives live ISIS and BGP routing feeds from all routers, which implies more than 600 BGP

[6] Even during this time the ISP benefited from the initial *FD* deployments via its analytic capabilities.

Number of machines	2
Cores/threads/hyper-threads	16/32/64 (Xeon E5)
RAM	> 700 GB
Number or IPv4/IPv6 routes	~850 k / ~680 k
NetFlow records	> 45 billion per day
NetFlow rate (peak)	> 1.2 Gbps
BGP peers	> 600
Num. cooperating *hyper-giants*	1
% *steerable* over all ingress traffic	> 10%

Table 2: *Flow Director* deployment.

peers with an average number of roughly 850 k routes. The two servers are configured for fail-over automation, whereby both receive the routing information but only one handles flow data. The latter processes more than 45 billion NetFlow records per day at a peak rate which exceeds 1.2 Gbps. Currently, both live systems use roughly 200 GB of memory (mainly for the BGP listeners), and their average CPU utilization is less than 45 resp. 5 out of the 64 hyper-threads. The load difference is due to the processing of flow data. Although both servers are ready to receive SNMP data to detect backbone bottlenecks and incorporate into the `Path Ranker`, the ISP does not deem it necessary for this period as its backbone is sufficiently over-provisioned.

5.2 Impact of *Flow Director*

Figure 14 illustrates how the percentage of optimally-mapped traffic for the *hyper-giant* evolved over the last two years. In particular, the plot shows the mapping compliance, recall Section 3.1, as well as the "*steerable*" traffic. Traffic is considered "steerable" by the ISP if the *hyper-giant*'s mapping decision receives a recommendation from the *Flow Director*, i.e., if their system accepts recommendations for that content. Note, steerable does not imply that the *hyper-giant* follows the recommendation.

When the collaboration started, the optimally-mapped traffic of that *hyper-giant* was around 70% with a declining trend, motivating the test of *FD*. At the beginning, the fraction of steerable traffic was rather small, yet, it quickly increased to 40%. This led to an increase in mapping compliance. In December 2017, this fraction dropped drastically when the *hyper-giant* and the ISP engaged in a test to evaluate the benefits of EDNS (similar to [12]). An investigation revealed that a misconfiguration caused the *hyper-giants*' mapping system to default to a state where it neither used the ISPs recommendations nor the information it used to rely on prior to the testing. This misconfiguration coincided with the holiday season, which may explain its rather long duration.

Once the misconfiguration was removed, the fraction of optimally-mapped traffic increased. In addition, the *hyper-giant* increased the fraction of steerable traffic, which further increased its mapping compliance. After a transient phase, the compliance stayed in the range of 75–84%, which is significantly larger than that of most other considered *hyper-giants*. Note, this *hyper-giant* maintains a large fraction of optimally-mapped traffic, despite having the largest number of PoPs, contributes a significant fraction of the overall ingress traffic (> 10%), and continuously increases its peering capacity.

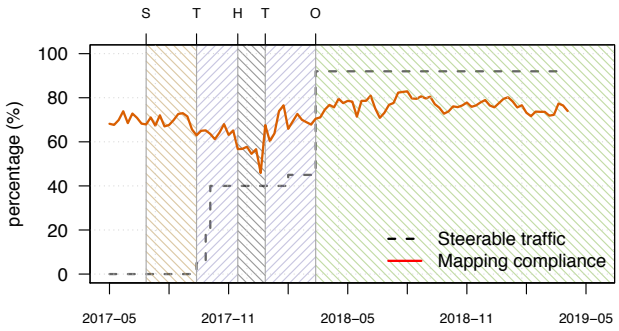


Figure 14: Timeline: Impact of CDN-ISP collaboration on share of optimally-mapped traffic annotated with events: Start (S/yellow), initial testing (T/blue), temporary hold (H/gray), operational (O/green).

5.3 ISP KPI: Long-haul traffic reduction

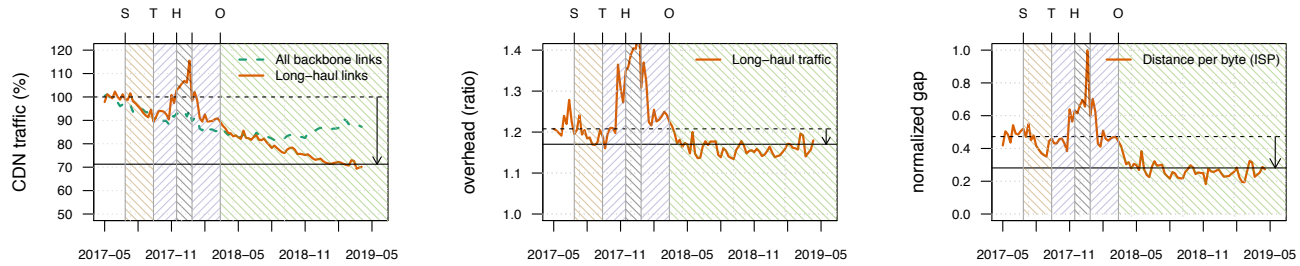
Next, we evaluate if *FD* meets the expectations of the ISP, namely, if it reduces its long-haul traffic load – the ISP's Key Performance Indicator (KPI). A naïve way of evaluating this KPI is to sum the *hyper-giant*'s traffic share for each long-haul's link over time. However, this is infeasible since gathering flow records for all backbone links does not scale. Moreover, transit traffic needs to be removed. We tackle this problem by reusing *FD*'s unique capabilities to combine flow data from the ISP's edge routers with routing data to infer its forwarding path and determine which links are long-haul vs. local within a PoP using the geographic locations of the routers. Then, we filter this traffic for destinations in the ISP's network and sources from this *hyper-giant*. However, this data is influenced by various trends, seasonal patterns, and other artifacts. Thus, we normalize traffic to account for traffic growth and customer migration as follows:

Ingress-traffic trends: We eliminate seasonal trends by normalizing the volume of ingress traffic within a time period to a constant, keeping the relative traffic share of each edge router unaltered.

Customer migration: Over the past two years, the ISP has migrated users to Broadband Network Gateways (BNGs), which increases the hop count by one. We mitigate this artifact by ignoring BNG links.

Figure 15(a) shows the benefit of *FD* on the *hyper-giant*'s long-haul traffic using May 2017 as reference. Thus, the value for May 2017 is 100%. When the collaboration started, we see a decrease. However, around December 2017 we see a significant increase. This coincides with the *hyper-giant*'s misconfiguration of its mapping system after the ISP's EDNS test. Once *FD* is fully utilized, the fraction of long-haul link traffic has a strong declining trend. The plot also shows the *hyper-giant*'s traffic share for all backbone links (again normalized). Note, the backbone traffic declined overall but has started increasing in 2018. This is not that surprising given that a reduction in long-haul traffic is often at the expense of increased intra-PoP traffic. Overall, the "relative" decline in long-haul traffic of more than 30% is impressive.

Naturally, the ISP has increased its network capacity and its footprint within the last two years by not only increasing capacity of existing links but also deploying new physical or logical links. This



(a) Impact of CDN-ISP collaboration on the *hyper-giant*'s long-haul/backbone traffic.

(b) Long-haul traffic overhead ratio between the actual and "ISP-optimal" mapping .

(c) Gap between the actual and the "ISP-optimal" *distance·byte* (relative to obs. worst case).

Figure 15: Timelines annotated with cooperation events: Start (S/yellow), initial testing (T/blue), temporary hold (H/gray), operational (O/green). Horizontal lines for metric average for May 2017 (top) and March 2019 (bottom).

will also either decrease or increase the path length. Unfortunately, we do not have a direct way to separate the impact of these upgrades from the benefits of the cooperation via *FD*. Moreover, there are reasons why the benefits of the activation of *FD* are not always realizable and are not immediately visible, including:

Content availability: Some content is only hosted on a subset of the *hyper-giants* infrastructure.

Processes: The *hyper-giant* may have had to update its internal processes and its mapping system to automatically consider *FD*'s recommendations.

Server resp. network load: The *hyper-giants* resource/ cost optimization may favor different server clusters.

Thus, we cannot expect that the *hyper-giant* always follows the ISP's recommendation to realize an "ISP-optimal" mapping. To understand these operational limits, we compute the ratio between the actual load on the long-haul links and one assuming the *hyper-giant* would follow all recommendations. This approach removes the effect of topological changes and thereby, reveals the actual benefits of *FD*.

Overall, we see, in Figure 15(b), that before the introduction of *FD* the gap was increasing. Moreover, during the time of the misconfiguration in December 2017 the gap was sizable. Once *FD* was fully operational, the overhead reduced to 1.17 for most months. Indeed, we still see a decreasing trend indicating that *FD* is beneficial for both the *hyper-giant* and the ISP.

Next, we explore why a *hyper-giant* may not follow *FD*'s recommendations. In particular, we focus on whether or nor the mapping compliance ratio decreases under high traffic load. The motivation comes from how the *hyper-giant*'s other resource/cost constraints may dominate their motivation to reduce latency. Accordingly, we plot for each hour in the month of February 2019 the fraction of traffic that is following the *FD*'s recommendation vs. the *hyper-giants* traffic volume (normalized by its peak hourly traffic volume within the month). Figure 16 shows the resulting scatter-plot overlayed with a heatmap to highlight the center of gravity. The plot shows that for most hours the compliance ratio is rather large, between 80% and 90%, which matches previous observations. However, for peak hours, the compliance ratio decreases but typically still exceeds 70%. Even for the worst hour the compliance is above 60%.

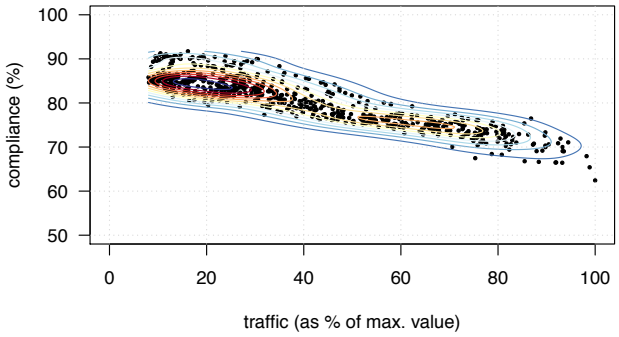


Figure 16: Scatter-plot with heatmap overlay: Compliance ratio vs. *hyper-giants* traffic normalized by peak traffic for February 2019.

5.4 *Hyper-giant* KPI: Latency reduction

The *hyper-giant*'s main incentive to use *Flow Director* is to improve latency. While this *hyper-giant* could measure the effects itself, it lacks the visibility of *FD*, e.g., choice of routes within the ISP. Thus, we approach this problem in the same manner as before: for each day we compute the *distance per byte* for the actual and the optimal mapping.[7] We then compute the gap by taking the difference between both, and then normalize it with the maximum observed gap over the two years. Figure 15(c) shows the results. We also added two support lines for the mean gap in May 2017 vs. March 2019. This plot again highlights that as the mapping compliance increases the gap closes, which will improve the QoE metrics of the *hyper-giant* and ISP and underline the benefits of *FD*. This observation is also supported by the *hyper-giant*'s own private measurements, which show a reduction of round-trip times (RTT) since the activation of *FD*.

5.5 What-if analysis

Given the benefits of the *hyper-giant*-ISP collaboration for a single *hyper-giant*, we next investigate the potential benefits if all other considered *hyper-giants* would also use *FD* recommendations. Using data from March 2019, Figure 17 shows the theoretically possible

[7] We are aware that distance is not the only factor affecting latency, but serves well as a proxy for the latency in optimal conditions i.e., uncongested networks, which is the case for the targeted ISP.

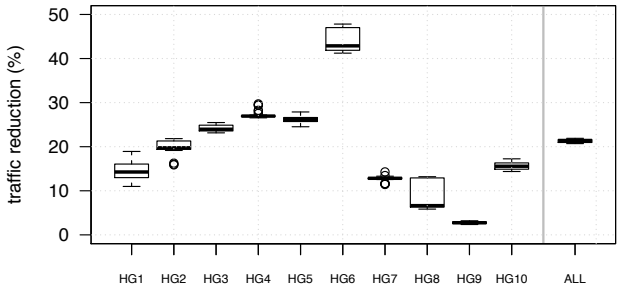


Figure 17: Quartile boxplot for the ratio of traffic under optimal mapping conditions vs. observed traffic.

ISP's long-haul traffic reduction, which with more than 20% is very sizable (assuming no resource constraints on the *hyper-giant* side).

We see that the potential long-haul traffic reduction varies from *hyper-giant* to *hyper-giant* since *hyper-giants* interconnect with the ISP at different PoPs and their traffic matrices differ. For some *hyper-giants* traffic can be reduced by 40%, e.g., for HG6. For other *hyper-giants* e.g., HG9, the benefit is much less, even though its mapping compliance is less than 80%. The reason for this counter-intuitive result is a consequence of the currently used optimization function (optimize for hop-count and distance). For example, the optimization potential is small for consumers that are located in-between two of a *hyper-giant*'s ingress PoPs even when the mapping is sub-optimal. The specific peering configuration of HG9 together with the mapping function in use fosters this effect.

However, the choice of optimization function for *FD* is flexible as long as it is computable using network information. For the initial deployment we focused on a function that provides (a) stability over time, (b) simplicity of evaluating the cooperation, and (c) avoids high-frequency changes.

6 SUMMARY AND OUTLOOK

In this paper, we report on our experience in building, rolling-out, and operating *Flow Director* (*FD*), the first-ever *ISP-hyper-giant* collaboration system. For two years, *FD* has been enabling one of the largest eyeball networks world-wide to steer the large traffic volumes from one of the most distributed *hyper-giants*: it provides recommendations to this *hyper-giant*'s mapping system in real-time, i.e., which *hyper-giant*'s server cluster should serve a consumer IP prefix given an optimization function defined by the *ISP*.

To motivate this system deployment, we investigate to which degree *hyper-giants* are able or rather unable to find optimal mappings for this ISP. Challenges include changes in (1) the *hyper-giants*/ISP connectivity, (2) the ISP's routing/topology, and (3) the geographic distribution of the ISP's IP address space. These often lead to mapping compliance of $\leq 64\%$, opening a large space for improvements via the *Flow Director*.

The goal behind the deployment of *FD* is to reduce traffic on –costly– long-haul links. Thus, the system uses an optimization function that is a combination of path length and distance. *FD* helps to decrease the *hyper-giant* traffic on these links by 30% by recommending appropriate ingress points to the *hyper-giant* and by steering traffic to shorter ISP links. Moreover, the gap between actual and "ISP-optimal" long-haul traffic has also decreased to roughly 1.15 (15% reduction of the overhead). For the *hyper-giant*, the incentive to interface with *FD* is latency reduction. Using a topology-agnostic KPI: the distance-per-byte's gap to reach consumers decreased by almost 40%. Moreover, we find a strong negative correlation between traffic demand and mapping compliance metric. Namely: available resources and cost factors external to the *FD* affect its overall efficiency.

Realizing *FD* within the *ISP* was, at times, a frustrating experience as we had to overcome many challenges, including (1) policy enforcement e.g., clearance for the necessary input data, (2) changing processes within the ISP e.g., obtaining up to date topology information, (3) interfacing with proprietary solutions, and (4) handling system failures and data problems. Moreover, having a principle agreement for collaboration between an *ISP* and a *hyper-giant* does not mean immediate deployment. Nevertheless, *FD* has now been in operation for more than two years, and it has outperformed our expectations. If the system were to be used by all top-10 *hyper-giants*, the traffic on long-haul links would further reduce to less than 80%.

In the future, we plan to further improve *FD* by (1) adding other optimization functions, e.g., to reduce max. utilization, (2) taking advantage of its analytic capabilities e.g., to assess ISPs on the suitability of a new peering location, and (3) interfacing with ISPs' routers to optimize egress traffic.

ACKNOWLEDGMENTS

We thank all past and current employees from BENOCS for making possible this paper, as well as Jason Moreau and Scott Roche (Akamai) for their valuable input. This work and its dissemination efforts were partially supported by the European Research Council (ERC) grant ResolutioNet (ERC-StG-679158).

REFERENCES

[1] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. 2010. Comparing DNS resolvers in the wild. In *ACM IMC 2010*. 15–21.
[2] Akamai. 2019. https://www.akamai.com/us/en/resources/cdn-solutions.jsp. (2019).
[3] Akamai. 2019. Akamai Facts and Figures. https://www.akamai.com/us/en/about/facts-figures.jsp. (2019).
[4] R. Alimi, R. Penno, and Y. Yang. 2011. ALTO Protocol. IETF RFC 7285. (2011).
[5] Amazon. 2019. https://docs.aws.amazon.com/directconnect/latest/UserGuide/routing-and-bgp.html. (2019).
[6] T. Böttger, F. Cuadrado, and S. Uhlig. 2018. Looking for Hypergiants in PeeringDB. *ACM CCR* 48, 3 (2018).
[7] C. Labovitz. 2019. Internet Traffic 2009-2019. APRICOT 2019. (2019).
[8] M. Caesar and J. Rexford. 2005. BGP Routing Policies in ISP networks. *IEEE network* 19, 6 (2005), 5–11.
[9] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan. 2013. Mapping the Expansion of Google's Serving Infrastructure. In *ACM IMC*. 313–326.
[10] M. Calder, M. Schroder, R. Gao, R. Stewart, J. Padhye, R. Mahajan, G. Ananthanarayanan, and E. Katz-Bassett. 2018. Odin: Microsoft's Scalable Fault-Tolerant CDN Measurement System. In *Proc. USENIX NSDI 2018*. 501–517.
[11] Amazon CloudFront CDN. 2019. https://aws.amazon.com/cloudfront/. (2019).
[12] F. Chen, R. K. Sitaraman, and M. Torres. 2015. End-User Mapping: Next Generation Request Routing for Content Delivery. In *Proc. ACM SIGCOMM 2015*.
[13] Y. C. Chiu, B. Schlinker, A. B. Radhakrishnan, E. Katz-Bassett, and R. Govindan. 2015. Are We One Hop Away from a Better Internet?. In *ACM IMC*.
[14] Cisco. 2012. Introduction to Cisco IOS NetFlow - A Technical Overview. https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html. (2012).
[15] Cisco. 2019. Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html. (2019).

[16] B. Claise, B. Trammell, and P. Aitken. 2013. RFC 7011: Specification of the IPFIX Protocol for the Exchange of Flow Information. (2013).

[17] D. D. Clark, J. Wroclawski, K. Sollins, and R. Braden. 2002. Tussle in Cyberspace: Defining Tomorrow's Internet. In *Proc. ACM SIGCOMM 2002*. 347–356.

[18] C. Contavalli, W. van der Gaast, D. Lawrence, and W. Kumari. 2016. Client Subnet in DNS Queries. IETF RFC 7871. (2016).

[19] CZ.NIC Labs. 2019. The BIRD Internet Routing Daemon. http://bird.network.cz. (2019).

[20] A. Dhamdhere, D. D. Clark, A. Gamero-Garrido, M. Luckie, R. K. P. Mok, G. Akiwate, K. Gogia, V. Bajpai, A. C. Snoeren, and kc claffy. 2018. Inferring Persistent Interdomain Congestion. In *Proc. ACM SIGCOMM*.

[21] F. Dobrian, A. Awan, D. Joseph, A. Ganjam, J. Zhan, V. Sekar, I. Stoica, and H. Zhang. 2011. Understanding the Impact of Video Quality on User Engagement. In *Proc. ACM SIGCOMM 2011*. 362–373.

[22] X. Fan, E. Katz-Bassett, and J. Heidemann. 2015. Assessing affinity between users and CDN sites. In *TMA*.

[23] P. Faratin, D. D. Clark, S. Bauer, W. Lehr, P. Gilmore, and A. Berger. 2008. The Growing Complexity of Internet Interconnection. *Communications and Strategies* 72 (2008), 51.

[24] T. Flach, N. Dukkipati, A. Terzis, B. Raghavan, N. Cardwell, Y. Cheng, A. Jain, S. Hao, E. Katz-Bassett, and R. Govindan. 2013. Reducing Web Latency: the Virtue of Gentle Aggression. In *Proc. ACM SIGCOMM*.

[25] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, and R. Weber. 2013. Pushing CDN-ISP Collaboration to the Limit. *ACM CCR* 43, 3 (2013), 34–44.

[26] M. J. Freedman. 2010. Experiences with CoralCDN: A Five-Year Operational View. In *Proc. USENIX NSDI 2010*. 95–110.

[27] M. J. Freedman, M. Vutukuru, N. Feamster, and H. Balakrishnan. 2005. Geographic Locality of IP Prefixes. In *ACM IMC 2005*. 13–13.

[28] V. Giotsas, M. Luckie, B. Huffaker, and k. claffy. 2014. Inferring Complex AS Relationships. In *ACM IMC 2014*. 23–30.

[29] V. Giotsas, G. Smaragdakis, B. Huffaker, M. Luckie, and k. claffy. 2015. Mapping Peering Interconnections at the Facility Level. In *Proc. ACM CoNEXT 2015*. 37.

[30] Benocs GmbH. 2019. https://github.com/Benocs/utee. (2019).

[31] Benocs GmbH. 2019. https://github.com/Benocs/bftee. (2019).

[32] Google. 2019. Google Global Cache. https://peering.google.com/#/options/google-global-cache. (2019).

[33] Google. 2019. Google Peering. https://peering.google.com. (2019).

[34] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett. 2014. SDX: A Software Defined Internet Exchange. In *Proc. ACM SIGCOMM 2014*.

[35] O. Hohlfeld, J. Rüth, K. Wolsing, and T. Zimmermann. 2018. Characterizing a Meta-CDN. In *PAM*.

[36] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Holzle, S. Stuart, and A. Vahdat. 2013. B4: Experience with a Globally-Deployed Software Defined WAN. *ACM CCR* 43 (2013), 3–14.

[37] J. Jung, B. Krishnamurthy, and M. Rabinovich. 2002. Flash Crowds and Denial of Service Attacks. In *Proc. ACM WWW 2002*. 293–304.

[38] E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe. 2006. Towards IP geolocation using delay and topology measurements. In *ACM IMC 2006*. 71–84.

[39] kc claffy, D. D. Clark, S. Bauer, and A. Dhamdhere. 2016. Policy challenges in mapping Internet interdomain congestion. In *TPRC*.

[40] S. Kiesel, W. Roome, R. Woundy, S. Previdi, S. Shalunov, R. Alimi, R. Penno, and Y. R. Yang. 2014. Application-Layer Traffic Optimization (ALTO) Protocol. IETF RFC 7285. (2014).

[41] R. Kohavi, R. M. Henne, and D. Sommerfield. 2007. Practical Guide to Controlled Experiments on the Web: Listen to Your Customers not to the HiPPO. In *Proc. ACM KDD 2007*. 959–967.

[42] R. Krishnan, H. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao. 2009. Moving Beyond End-to-end Path Information to Optimize CDN Performance. In *ACM IMC 2009*. 190–201.

[43] S. S. Krishnan and R. K. Sitaraman. 2012. Video Stream Quality Impacts Viewer Behavior: Inferring Causality using Quasi-Experimental Designs. In *ACM IMC 2012*. 2001–2014.

[44] C. Labovitz, S. Lekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. 2010. Internet Inter-Domain Traffic. In *Proc. ACM SIGCOMM*.

[45] T. Leighton. 2009. Improving Performance on the Internet. *Comm. of the ACM* 52, 2 (2009), 44–51.

[46] A. Lodhi, N. Larson, A. Dhamdhere, C. Dovrolis, and K. Claffy. 2014. Using PeeringDB to Understand the Peering Ecosystem. *ACM CCR* 44, 2 (2014).

[47] M. Luckie, A. Dhamdhere, D. Clark, B. Huffaker, and kc claffy. 2014. Challenges in Inferring Internet Interdomain Congestion. In *ACM IMC 2014*. 15–22.

[48] B. M. Maggs and R. K. Sitaraman. 2015. Algorithmic Nuggets in Content Delivery. *ACM CCR* 45, 3 (2015), 52–66.

[49] Netflix. 2019. https://openconnect.netflix.com/en/deployment-guide/network-configuration/#routing-and-content-steering-via-bgp. (2019).

[50] Netflix. 2019. Netflix Open Connect. https://signup.netflix.com/openconnect. (2019).

[51] E. Nygren, R. K. Sitaraman, and J. Sun. 2010. The Akamai Network: A Platform for High-performance Internet Applications. *SIGOPS Oper. Syst. Rev.* 44, 3 (2010), 2–19.

[52] P. Phaal and M. Lavine. 2004. sFlow version 5. https://sflow.org/sflow_version_5.txt. (2004).

[53] R. Padmanabhan, A. Dhamdhere, E. Aben, kc. Claffy, and N. Spring. 2016. Reasons Dynamic Addresses Change. In *ACM IMC 2016*. 183–198.

[54] PeeringDB. 2019. PeeringDB. https://www.peeringdb.com. (2019).

[55] pmacct. 2019. . http://www.pmacct.net/. (2019).

[56] I. Poese, B. Frank, B. Ager, G. Smaragdakis, and A. Feldmann. 2010. Improving Content Delivery using Provider-aided Distance Information. In *ACM IMC 2010*. 22–34.

[57] I. Poese, B. Frank, G. Smaragdakis, S. Uhlig, A. Feldmann, and B. Maggs. 2012. Enabling Content-aware Traffic Engineering. *ACM CCR* 42, 5 (2012), 21–28.

[58] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye. 2011. IP Geolocation Databases: Unreliable? *ACM CCR* 41, 2 (2011), 53–56.

[59] L. Popa, A. Ghodsi, and I. Stoica. 2010. HTTP as the Narrow Waist of the Future Internet. In *Proc. ACM SIGCOMM HotNets*. 1–6.

[60] E. Pujol, P. Richter, B. Chandrasekaran, G. Smaragdakis, A. Feldmann, B. Maggs, and K. C. Ng. 2014. Back-Office Web Traffic on The Internet. In *ACM IMC 2014*. 257–270.

[61] Quagga community. 2019. Quagga Routing Suite. http://www.nongnu.org/quagga/. (2019).

[62] P. Richter, G. Smaragdakis, D. Plonka, and A. Berger. 2016. Beyond Counting: New Perspectives on the Active IPv4 Address Space. In *ACM IMC*.

[63] J. P. Rula and F. E. Bustamante. 2014. Behind the Curtain – Cellular DNS and Content Replica Selection. In *ACM IMC 2014*. 59–72.

[64] M A. Sánchez, J. .S. Otto, Z. S. Bischof, D. R. Choffnes, F. E. Bustamante, B. Krishnamurthy, and W. Willinger. 2013. Dasu: Pushing Experiments to the Internet's Edge. In *Proc. USENIX NSDI 2013*. 487–499.

[65] B. Schlinker, H. Kim, T. Cui, E. Katz-Bassett, H. V. Madhyastha, I. Cunha, J. Quinn, S. Hasan, P. Lapukhov, and H. Zeng. 2017. Engineering Egress with Edge Fabric: Steering Oceans of Content to the World. In *Proc. ACM SIGCOMM 2017*. 418–431.

[66] J. Scudder, R. Fernando, and S. Stuart. 2016. BGP Monitoring Protocol (BMP). IETF RFC 7854. (2016).

[67] J. Seedorf and E. Burger. 2009. Application-Layer Traffic Optimization (ALTO) Problem Statement. IETF RFC 5693. (2009).

[68] R. Singh, M. Ghobadi, K. T. Foerster, M. Filer, and P. Gill. 2018. RADWAN: Rate Adaptive Wide Area Network. In *Proc. ACM SIGCOMM 2018*. 547–560.

[69] R. K. Sitaraman, M. Kasbekar, W. Lichtenstein, and M. Jain. 2014. *Overlay Networks: An Akamai Perspective*. John Wiley & Sons.

[70] V. Stocker, G. Smaragdakis, W. Lehr, and S. Bauer. 2017. The Growing Complexity of Content Delivery Networks: Challenges and Implications for the Internet Ecosystem. *Telecommunications Policy* 41, 10 (2017), 1003–1016.

[71] F. Streibelt, J. Boettger, N. Chatzis, G. Smaragdakis, and A. Feldmann. 2013. Exploring EDNS-Client-Subnet Adopters in your Free Time. In *ACM IMC 2013*. 305–312.

[72] V. Valancius, N. Laoutaris, L. Massoulie, C. Diot, and P. Rodriguez. 2009. Greening the Internet with Nano Data Centers. In *Proc. ACM CoNEXT 2009*. 37–48.

[73] D. Walton, A. Retana, E. Chen, and J. Scudder. 2016. Advertisement of Multiple Paths in BGP. IETF RFC 7911. (2016).

[74] P. Wendell and M. J. Freedman. 2011. Going Viral: Flash Crowds in an Open CDN. In *ACM IMC 2011*. 549–558.

[75] F. Wohlfart, N. Chatzis, C. Dabanoglu, G. Carle, and W. Willinger. 2018. Leveraging Interconnections for Performance: The Serving Infrastructure of a Large CDN. In *Proc. ACM SIGCOMM 2018*. 206–220.

[76] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz. 2008. P4P: Provider Portal for Applications. In *Proc. ACM SIGCOMM 2008*.

[77] K-K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain, V. Lin, C. Rice, B. Rogan, A. Singh, B. Tanaka, M. Verma, P. Sood, M. Tariq, M. Tierney, D. Trumic, V. Valancius, C. Ying, M. Kallahalla, B. Koley, and A. Vahdat. 2017. Taking the Edge off with Espresso: Scale, Reliability and Programmability for Global Internet Peering. In *Proc. ACM SIGCOMM 2017*. 432–445.