Tradeoffs in CDN Designs for Throughput Oriented Traffic

Minlan Yu* Wenjie Jiang† Haoyuan Li‡ Ion Stoica‡
* University of Southern California † Princeton University ‡ University of California Berkeley

ABSTRACT

Internet delivery infrastructures are traditionally optimized for *low-latency* traffic, such as the Web traffic. However, in recent years we are witnessing a massive growth of *throughput-oriented* applications, such as video streaming. These applications introduce new tradeoffs and design choices for content delivery networks (CDNs). In this paper, we focus on understanding two key design choices: (1) What is the impact of the number of CDN's peering points and server locations on its aggregate throughput and operating costs? (2) How much can ISP-CDNs benefit from using path selection to maximize its aggregate throughput compared to other CDNs who only have control at the edge? Answering these questions is challenging because content distribution involves a complex ecosystem consisting of many parties (clients, CDNs, ISPs) and depends on various settings which differ across places and over time. We introduce a simple model to illustrate and quantify the essential tradeoffs in CDN designs. Using extensive analysis over a variety of network topologies (with varying numbers of CDN peering points and server locations), operating cost models, and client video streaming traces, we observe that: (1) Doubling the number of peering points roughly doubles the aggregate throughput over a wide range of values and network topologies. In contrast, optimal path selection improves the CDN aggregate throughput by less than 70%, and in many cases by as little as a few percents. (2) Keeping the number of peering points constant, but reducing the number of location (data centers) at which the CDN is deployed can significantly reduce operating costs.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Design studies; C.2.1 [**Computer-communication networks**]: Network Architecture and Design—*Distributed networks*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CoNEXT'12, December 10–13, 2012, Nice, France.
Copyright 2012 ACM 978-1-4503-1775-7/12/12 ...$15.00.

General Terms

Design, Performance

Keywords

content distribution networks, throughput, multipath

1. INTRODUCTION

Content delivery networks (CDNs) are serving an ever increasing number of throughput-oriented applications such as video streaming, software, games, and movie downloads. In particular, video streaming represents a significant fraction of today's Internet traffic [1, 2], as many content publishers start to deliver content through the Internet: Netflix has reached 20 million US subscribers, and YouTube streams over 2 billion of videos per day [3]. A recent Cisco report predicts that 90% of the consumer traffic will be video by 2013 [4].

The massive growth of the throughput-oriented applications introduces new design choices for CDNs. CDNs have traditionally been optimized for latency-oriented applications, such as web browsing, rather than throughput-oriented applications. It is important to revisit these CDN designs for the throughput metric. A flow can experience throughput bottleneck at the clients, the servers, or along the network paths in between. Clients have no choice other than upgrading the connection. CDNs often have strong incentives to buy more bandwidths to eliminate throughput bottlenecks. Therefore, the most challenging problem is to avoid throughput bottlenecks inside the network.

Primarily, there are two ways to improve the CDN throughput: (1) use path selection and multipath routing to avoid network bottleneck between a CDN server and a client, and (2) increase the number of peering points of the CDN. ISP-CDNs, such as AT&T and Verizon, can use both these approaches to improve the CDN aggregate throughput [5, 6, 7, 8, 9, 10], as they control both the network and the CDN peering. In contrast, traditional CDNs are left with increasing the number of peering points, as the main choice to improve their throughput. In this context, we consider the following design question: *What are the benefits of optimal path selection–beyond increasing the number of peering points– to improve the aggregate throughput of a CDN)?* In other words, we want to quantify the advantage that the ISP-CDNs may have over traditional CDNs, when it comes to maximizing the throughput.

Another design decision that a CDN faces is choosing the number of locations (data centers) at which to deploy its

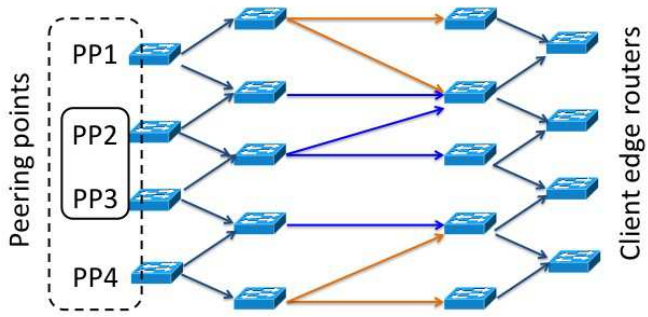


Figure 1: An example comparing the throughput from multipath and more PPs (The capacities of links going out of the PPs are 2. The capacities of all the other links are 1).

servers. For example, Akamai takes a more *distributed* approach by deploying its servers at thousands of locations across the world, while Level3 and Limelight take a more *centralized* approach by building a few data centers, each peering with many ISPs. There are at least two metrics of interest when choosing the number of locations: aggregate throughput and costs. In this paper, we consider the natural question: *How does the number of locations impact the CDN's aggregate throughput and its operating cost?*

Answering the above questions is challenging as content distribution involves a complex ecosystem consisting of many parties (e.g., clients, CDNs, and ISPs), depends on a variety of settings (e.g., topology, routing, business relationships between ISPs and CDNs, operating cost), and evolves rapidly over time. Many of these information are hard to obtain, as they are related to business contracts between different companies. These contracts are difficult to infer even by operators of a single ISP or CDN, letting alone academic researchers. Even if we were able to obtain detailed measurements for the existing CDNs, it would still be difficult to extrapolate the CDN performance for future scenarios. For example, with more peering links between CDNs and ISPs and more investment in increasing edge network capacity [11], it is no longer clear whether the throughput bottleneck will appear at the edge or inside the network.

In this paper we introduce a simple model for understanding CDN design choices on improving throughput. The model is not intended to be a complete representation of reality (e.g., which CDN is better), but instead, is intended to illustrate and quantify the essential tradeoffs in CDN designs in a way that makes it easy to evaluate various scenarios in today's and the future Internet. We focus on modeling two aspects: (1) ISP's ability of path selection inside the network and what path information they expose to CDNs; (2) CDN's choices of server locations and peering points with different operating cost. We drive the model with various settings of both synthetic and Internet network topologies [12, 13], Akamai and Limelight server locations [14], client video streaming demands from Conviva [15], and different types of operating cost. We make two observations:

1. Beyond increasing the number of peering points, optimal path selection has relatively little impact on the throughput. Although path selection and multipath routing are effective in improving the throughout of point-to-point communication, we show that this is not always the case for throughput-oriented video traffic which typically originates at multiple server locations. This is because increasing the number of peering points essentially increases the min-cut size. In contrast, improving path selection only approximates the min-cut size. For example, in Figure 1, the min-cut size from two peering points (PP2,PP3) to four client locations is 4 (the best throughput improvement from multipath is 4/3). In contrast, by doubling the peering points (PP1-PP4), the min-cut size increases to 8. Our evaluations on various settings show that doubling the number of peering points from 10 to 20 can improve the throughput *linearly*, i.e., between 64.1%-157.1%, while optimal path selection can only improve the throughput by 0.7%-69.4% in most cases. This result indicates that CDNs can go a long way to improve their throughput by adding new peering points, while ISP-CDNs can derive relatively little benefits from their ability to optimize the routes.

2. To achieve the same throughput, the highly distributed CDNs incur higher operating cost than the more centralized CDNs We model various types of operating cost functions, such as bandwidth cost at the peering points, power and maintenance costs associated to hosting, and the cost of content replication across data centers. One of our results shows that a CDN with 150 peering points can reduce its operating costs by as much as 69% by deploying its servers at 15 instead of 80 locations. One natural question is then what is the minimum number of locations that a CDN should deploy its servers. To answer this question, we show that in today's Internet, a CDN can *directly* reach over 80% of all IP addresses by deploying its severs only at the top 54 PoP locations.[1] Thus, a CDN doesn't need to deploy its servers at hundreds or thousands of locations to directly reach the majority of clients. Based on these results, it should come as no surprise that most CDNs today choose a more centralized deployment for throughput-oriented traffic.[2]

The remaining of the paper is organized as follows: Section 2 discusses CDN's design goals of improving throughput and reducing operating cost in practice. Section 3 presents our simple model of understanding CDN design choices on path selection and peering point selection. Section 4 compares the effect of path selection and increasing the number of peering points under a variety of settings. Section 5 analyzes the performance and cost tradeoff for more centralized and more distributed CDN designs. Section 6 and 7 discuss related work and conclude the paper.

2. CDN DESIGN GOALS

In this section, we discuss two key goals of the CDN design: (1) improving throughput performance: Some CDNs can only control server and peering point selection at the edge of the network, while others (especially ISP-CDNs) can also control path selection inside the network. (2) Reducing operating cost: Some CDNs (like Akamai) use a large number of locations to improve aggregate throughput, while others (like Limelight, Level 3) take the more centralized ap-

[1]These are the IP addresses owned by the ISPs that peer at the top 54 PoP locations.

[2]Even Akamai is moving towards the more centralized approach for videos.[16].

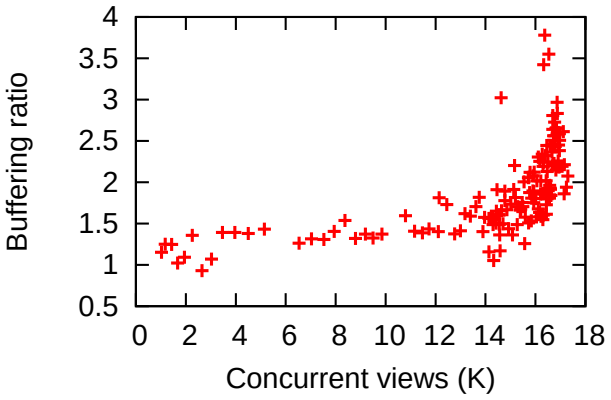


Figure 2: The congestion of a local ISP reduces video quality for viewers from all the CDNs. (The bottleneck is caused by network congestion because viewers with the same CDN server have different performance experiences.)

proach to reduce operating cost by leveraging the economy of scale.

2.1 Improving Throughput

The majority of today's Internet traffic is throughput oriented, including video streaming, software updates, and game and movie downloads. According to Sandvine's report, real-time entertainment applications (including Netflix, YouTube, games, IPTV) alone account for 60% of the Internet traffic in North America [1]. For this traffic (video in particular), throughput is often more important than delay. For example, the start-up time of a video, which is critical for user experience [17], is determined by the time it takes to fill an initial buffer, which depends on throughput. Of course, latency is still important: as most video traffic is streamed over TCP, high round-trip-time can negatively affect the throughput. However, new streaming technologies, such as HTTP chunking [18], allow clients to download multiple chunks in parallel. For these reasons, in this paper we do not consider the impact of latency on throughput.

There are three points at which a flow can experience bottleneck: at the client, at the server, or along the network path. On the client side there is little choice other than upgrading the client device or the connection. On the server side, CDNs often have strong incentives to buy more bandwidth from ISPs and to upgrade the server capacities to match clients' demands, alleviating the server-side bottlenecks.

Recently, the network has emerged as the major bottleneck for video distribution. Netflix has reported that most clients cannot sustain the highest rate (4.8 Mbps) due to ISP limited capacity [19]. Similarly, Akamai has stated that as the video traffic increases, the bottleneck "is no longer likely to be just at the origin data center. It could be at a peering point, or a network's backhaul capacity, or an ISP's upstream connectivity" [20]. Conviva [15], a company that manages video distribution, has seen significant network congestion during flash crowds. Figure 2 shows that the buffering ratio[3] of viewers across all the CDNs inside one ISP during a popular college football game. When there are more than 14K concurrent viewers, there is a clear degradation of the delivery performance caused by network congestion. We expect the network bottleneck to worsen, as the popularity of throughput-oriented applications increases (e.g., people watching online videos for longer time, and at a higher rate). As a result, more traffic "elephants" will start to step on each other, which significantly decreases the benefits of statistical multiplexing and introduces more challenges in bandwidth provisioning.

To improve the throughput and Internet resource utilization, there has been a growing interest in providing multiple paths and better path selection solutions [5, 6, 7, 8, 9, 10]. These designs are moving closer to reality, as many Internet Service Providers (ISPs), such as AT&T and Verizon, move up the revenue chain to deliver *content* by building their own CDNs. These ISP-CDNs are in the best position to take advantage of controlling both path and server selection to improve the network throughput. In contrast, existing CDNs improve throughput by deploying servers at more locations and setting up more peering points to ISPs. Therefore, it is important to compare the impact of path selection, in addition to server and peering point selection, on improving the end-to-end throughput, and understand the tradeoffs between the ISP-CDNs and other CDN designs.

2.2 Reducing CDN operating cost

CDNs incur three types of operating cost: *(1) Management cost:* At each server location, CDNs pay for the electricity, cooling, equipment maintenance, and human resources. *(2) Content replication cost:* the cost of replicating and caching content across different locations. For the content whose popularity follows a long-tail distribution (e.g., YouTube), CDNs often replicate the most popular content at many locations. In contrast, for rare content, CDNs may only use a single server location and redirect traffic to that location. There are both the storage cost of storing extra replicas and the bandwidth cost of redirecting traffic for rarely accessed contents. *(3) Bandwidth cost:* CDNs often pay ISPs for the bandwidth they use at the peering points based on some mutually-agreed billing model (e.g., 95 percentile usage). Since the peering bandwidth cost is not publicly available, in our model, we use publicly available transit prices as an upper bound for peering cost. This is because the alternative to peering is to simply send traffic to an upstream transit provider [21, 22].

One of the main CDN Design choices is where to deploy the servers. Broadly speaking, in this context, there are two designs that are largely guided by the performance (i.e., throughput) and cost tradeoffs:

Highly distributed approach: Traditionally, Akamai has placed its servers at thousands of locations across the world, which ensures that each client can find a nearby server. Moreover, with a large number of server locations, a client has more chances to find a high-throughput path, simply because there are more choices of paths between a client and the server locations. However, these advantages do not come for free, as the management cost increases with

[3]The buffering ratio is the percentage of time a client spend in buffering and is a common metric to indicate video streaming throughput performance.

the number of server locations. Furthermore, the same data item may need to be replicated (cached) to a larger number of locations, increasing the content replication cost.

In terms of bandwidth cost, Akamai traditionally has hosted servers at many ISPs for "free". However, as ISPs start to provide their own CDN services, it is very hard for other CDNs to place servers for free any more. Even Akamai today can only place servers for free for a few small ISPs without many clients, but has to pay those large ISPs for placing the servers. In addition, Akamai is deploying more capacity in large data centers, and thus incurring the peering bandwidth cost as other CDNs. According to the SEC fillings [23], Akamai's bandwidth costs are higher than Limelight's (of course, Akamai carries significantly more traffic, but still it shows that Akamai doesn't get the bandwidth for free).

More centralized approach: CDNs, such as Level3 and Limelight, build large data centers at only a few key locations, and set up peering points from their data centers to a large number of ISPs to increase their reach. Since these CDNs consist of only a few data centers, the cost of data replication and synchronization across data centers is relatively small. In terms of management and bandwidth cost, CDNs with a few larger data centers can benefit from the economy of scale.

3. MODELING CDN DESIGN CHOICES

We propose a simple model to understand CDN design choices. Our model is by no means a thorough representation of the reality. Instead, we design our model to make it both tractable and yet useful in understanding the ecosystem of CDNs and ISPs, as well as quantifying the tradeoffs in CDN design. We focus on modeling two ways of improving the throughput: (1) CDN's choices of server locations and peering points; (2) ISP's ability of path selection inside the network and what path information they expose to CDNs.

3.1 CDNs: Increase peering points at the edge

To improve throughput, CDNs can increase server locations and their peering points at the edge of the network, optimize the selections across these peering points. It is challenging to model existing CDN server locations and peering points, because they depend on business relationships and thus differ across CDNs. Even a single CDN may have to optimize differently for Asia and North America locations, as there are fewer peering links available in Asia. In addition, the CDN architecture evolves rapidly as traditional CDNs peering with more ISPs and many ISP-CDNs also joining the business. Because CDN designs are driven by economic incentives and business relationships, which are difficult to infer even for operators of a single ISP or CDN, letting alone academic researchers. Historical facts also play roles in designs. For example, Akamai already has a wide spread of servers (some are free), while it is challenging for a new CDN to deploy these servers at a low cost.

Modeling peering points: Instead of modeling any specific CDN implementation, we focus on comparing two fundamental types of CDN designs—more centralized and more distributed approaches as discussed in Section 2. The two methods differ in two parameters: the number of server locations at which the CDN deploys servers (n_s), and how many peering points (PP) the CDN sets up at each location to connect to ISPs ($n_p(s)$). First, to study the throughput of different numbers of PPs ($N_s = \sum_{n_s} n_p(s)$), we model PPs in the network independent of the number of servers n_s and their locations. Next, to compare the cost between more centralized and more distributed CDN designs, we fix the total amount of PPs (N_s) and select different number of servers (n_s). We model content replication by introducing a duplication threshold δ. We duplicate top *delta* popular contents at all the locations but place the rest at a random location. We can fully generalize the design choices of various CDNs beyond Akamai and Limelight by using this model.

Optimal peering point selection: To explore the impact of peering points on throughput, we assume CDNs can perform optimal peering point selection that maximizes the throughput of all clients. Assume a CDN has N_s peering points, N_c client locations. Let P_{sc} be the set of paths between peering point s and client c, and let x_p^{sc} be the amount of traffic traversing path $p \in P_{sc}$. We use routing matrix d_{lp} to specify whether l is on path p ($d_{lp} = 1$) or not ($d_{lp} = 0$). Since a client might download more than one content item at a time, we assume arbitrary splitting of client traffic across different server locations x_p^{sc}. We formulate peering point selection as the following optimization problem: the CDN maximizes the throughput (Eq. (1)), given the link capacity constraint (Eq. (2)). The problem can be easily extended to consider peering point selection for different contents by replacing x_p^{sc} to x_p^{scd} where d represents the specific content.

$$\max \quad \sum_{(s,c)} \sum_{p \in P_{sc}} x_p^{sc} \tag{1}$$

$$\text{s.t.} \quad \sum_{(s,c)} \sum_{p \in P_{sc}} x_p^{sc} * d_{lp} \leq Cap_l \tag{2}$$

$$\text{variables} \quad x_p^{sc} >= 0 \tag{3}$$

The problem runs in time $O(|P_{sc}| \times N_c \times N_s)$. In the AS-level Internet topology where N_s = 1K and N_c= 12K, a single simulation on the Internet topology takes about 12 hours. Therefore, to evaluate many settings, we did most of evaluations with smaller topologies and verify the results with the Internet topology.

Client demand constraints: The throughput optimization highly depends on the distribution of client demands across client locations. We start with a simple model that does not have any constraints on client demands to understand the upper bound of the throughput we can get from the network and the CDNs. We then extend the model to understand the impact of client distribution. We set lower and upper bounds on the traffic demand as $Lower_c \leq \sum_s \sum_{p \in P_{sc}} x_p^{sc} \leq Upper_c$ for each client location c. When the network does not have enough bandwidth capacity to serve all the clients, CDNs start to reject clients. We introduce an acceptance ratio a^{sc} ($\in [0, 1]$), and ensure $\sum_{p \in P_{sc}} x_p^{sc} \leq a^{sc} Upper_c$. In fact, the new constraint is the same with the original constraint without rejection, because we can use a new rate variable $\hat{x}_p^{sc}$ to replace x_p^{sc}/a^{sc}. We drive the client demand model with the Conviva traces on the number of video streaming sessions from each client location. Assume a location consist of 100 client sessions with video streaming rates ranging from 100 Kbps to 1 Mbps. In this case, the

Scenarios	CDN design	ISP design	Optimality	CDN's path changes
1. Today: no cooperation (labeled as "1path")	peering point selection	shortest path (e.g., OSPF)	not optimal	none
2. Better contracts: k shortest paths (labeled as "mpath")	pick dedicated paths + peering point selection	provide k shortest paths (e.g., setting MPLS)	sub-optimal	fast
3. ISP-CDN: Joint opt. (labeled as "mcf")	Joint traffic engineering and peering point selection		optimal	slow

Table 1: Interactions of CDNs and ISPs

lower and upper bounds at that location are 10 Mbps and 100 Mbps respectively.

3.2 ISPs: Improve path selection at the core

We model the ISP's path selection process based on how much path information the ISP exposes to the CDN, which leads to different levels of engineering overhead and throughput optimality (Table 1). Our model should capture both ISP-CDNs where both servers and clients are located within the same ISP, and those CDNs that spread servers across many ISPs. We first discuss three ways to implement server (path) selection assuming a single ISP, and then extend the model to include multiple ISPs.

CDN performs peering point selection; ISP uses shortest path. Today, there is no cooperation between CDNs and ISPs (Scenario 1). Given a client, the CDN selects a server, and then uses the shortest path provided by the ISP to deliver the content to the client. To improve the throughput, the CDN has to increase server locations at the edge and optimize peering point selection based on end-to-end performance measurements. The optimization problem for peering point selection is captured by Eq. (1)-(3) with $|P_{sc}| = 1$.

Better CDN and ISP contracts: ISPs expose multiple paths to the CDN. To further improve the throughput, the CDN may negotiate with the ISP the ability to use multiple-paths. The ISP can still make their own traffic engineering decisions, but provide *multiple* shortest paths for each pair of client locations and server peering points (e.g., using MPLS tunneling). The CDN can flexibly split traffic on these paths and pick peering points to optimize throughput (Scenario 2). The more paths the ISP provides to the CDN, the better the CDN can leverage the available bandwidth in the network. On the downside, exposing more paths increases the management overhead of ISPs. To model this setting, we set P_{sc} as k shortest paths [4], and solve the optimization problem for joint selection of peering points and paths captured by Eq. (1)-(3).

ISP-CDNs: Joint optimization of traffic engineering and server selection (Scenario 3). ISP-CDNs have full control on both CDN servers and network paths, and thus perform a joint optimization of server selection and path selection to achieve the *optimal* throughput. To understand the upper bound of the throughput that any multipath approach can achieve, we formulate the joint optimization as a *multi-commodity flow problem (mcf)* as shown in Eq. (4)-(8).

$$\max \quad \sum_{(s,c)} x_{sc} \tag{4}$$

$$\text{s.t.} \quad \sum_{(s,c)} x_{sc} * r_l^{sc} \leq Cap_l, \forall l \tag{5}$$

$$\sum_{l \in in(v)} r_l^{sc} - \sum_{l \in out(v)} r_l^{sc} = I_{v=c} \tag{6}$$

$$(\forall (s,c) \forall v \in V - \{s\}) \tag{7}$$

$$\text{variables} \quad 0 \leq r^{sc} \leq 1, \forall (s,c), \forall l \tag{8}$$

The variable in the problem is r_l^{sc}, denoting the splitting ratio of the traffic from client location c to peering point s at link l. In practice, path selection may happen at a lower frequency than server selection, because it involves route recomputing and network reconfiguration.

Similarly to the single ISP case, when a CDN connects to multiple ISPs, the CDNs may leverage existing shortest paths (the AS level paths following BGP policies) to maximize throughput (Scenario 1). The CDNs may also get multiple shortest paths from each peering point to ISPs (Scenario 2).

4. QUANTIFY THROUGHPUT BENEFITS

In this section, we compare two approaches to maximize the throughput: (1) increase the number of peering points, and (2) select multiple paths between CDN peering points and client locations to maximize the throughput. We first use max-flow min-cut theorem to illustrate the differences of the two approach. Next, through evaluations on a variety of network and server settings, we conclude that doubling the number of peering points roughly doubles the aggregate throughput over a wide range of values and network topologies. In contrast, optimal path selection improves the CDN aggregate throughput by less than 70%, and in many cases by as little as a few percents. Furthermore, we show that multipath is even less effective with client demand dynamics and long tail distribution of content accesses based on our analysis on Conviva video demands [15].

4.1 Multipath is not necessary for CDNs

Multipath increases throughput significantly for single-source, single-destination applications. However, CDNs' clients can access content from multiple servers via different peering points, which makes multipath less effective compared with increasing the number of peering points. This is illustrated by the max-flow min-cut theorem [24].

We reduce the multi-commodity flow problem formulated in Eq. (4)-(8) to the single source and single sink max flow problem. We assume perfect content replication in servers, such that all the servers and their peering points are treated equally in their abilities to serve clients. The clients are also treated equally with their different demands' lower and

[4] k is an upper bound and there may exist fewer than k shortest paths.

upper bounds. Therefore, we can add a new super source that is connected to all servers, as well as a new super sink. According to the max-flow min-cut theorem, the maximum flow passing from the super source to the super sink is equal to the minimum capacity over all source-sink cuts *(i.e., min-cut size)*. Therefore, the throughput that any multipath solution can achieve is always bounded by the min-cut size (i.e., the throughput from the *mcf* solution).

In contrast, increasing peering points adds new edges (from the new super source to new peering points) to the graph, which potentially grows the min-cut size. As the example shown in Figure 1, when we add two more peering points (PP1, PP4), the min cut size increases from 4 to 8. In contrast, the best throughput multipath can achieve with two peering points (PP1, PP2) is only 4.

Since the optimal multipath (*mcf*) is less effective than increasing peering points at the edge, other path selection solutions perform even worse for CDNs traffic improvement. Therefore, CDNs can go a long way to improve their throughput by just adding new peering points, while ISP-CDNs can derive relatively little benefits from their ability to optimize the routes. Furthermore, as suggested by [25], even using the static routes provided by ISPs, the CDNs can get most of the benefits of path selection by choosing the peering point from which to deliver the content.

To verify this conclusion, we quantify the throughput improvement to compare the optimal multipath solution *mcf* and increasing peering points solutions in the entire CDN design space (with different network topologies, link capacities, and client workloads).

4.2 Evaluation Setup

It is challenging to quantify the benefits of CDN designs because of the complexity of the network, clients, and servers. They may change over time, and are also highly related to business decisions. Instead of studying a point of design by measuring some existing CDNs, we perform an extensive evaluation to understand the effect of *mcf* and more peering points in a variety of settings (Table 2). For each setting, we run the simulation 100 times and take the average throughput.

Network topology and link capacity: The network topology and link capacity are two important factors for the CDN throughput. Network topologies are diverse and keep evolving. For example, the Internet is becoming flatter and peering links keeps increasing[5]. Therefore, it is important to not only study existing Internet topologies but also the other common topologies to fully understand their effect on throughput performance. We evaluate our model under many realistic and synthetic topologies: To understand ISP-CDNs, we take the router-level topologies in several ISPs (e.g., Comcast, Cox) from ITDK data set [12]. We choose the AS-level Internet topology from Cyclops [13].[6] for CDNs spreading across multiple ISPs. To understand the impact of topologies with different levels of connectivity, we use BRITE [27] to generate three types of synthetic topologies: power-law graph, random graph, and the hierarchical graph.[7]

To the best of our knowledge, there is no good understanding of link capacity available inside the networks for CDNs, because it is hard to get precise packet timing (from iplane [28]'s experience [29]), and hard to congest a link inside the Internet and measure the link capacity before ISPs react to congestion. It is even more challenging with the Internet evolving over time (e.g., with growing investments on home access networks). Therefore, the best thing we can do is to follow the normal practice [30] and to evaluate our model under a variety of link capacity distributions. According to previous work on Internet measurement and modeling [31, 32, 28], we choose three representative link capacity distributions (power law, uniform, and pareto) within the range of 100 Mbps to 10 Gbps.[8] We also assign different link capacities inside and between ASes in the hierarchical topology [28, 31] (e.g., with "high peering cap", we increase link capacity between ASes is twice of those inside ASes.)

Peering point and client selection: We get the realistic Akamai and Limelight server IP addresses from PlanetLab measurement collected at Nov. 2010 [14] and map them to the ASes where the servers are located using Quova [33]. Akamai has 1026 ASes while Limelight has 216 ASes. To understand the entire CDN design space, we study two ways of peering point selection in synthetic topologies: First, to compare the throughput gains of increasing PPs and increasing paths (independent of different CDN designs), we pick the peering points (PPs) randomly from the network. Next, to compare the cost of more centralized and more distributed CDNs, we rank the nodes in the network based on their outgoing degrees and pick the high-degree nodes as servers and set peering points from those nodes connecting to the server nodes. Similarly, we pick low-degree nodes to model more distributed CDNs. We choose client locations randomly from a pool of low-degree nodes, because clients typically have only a few access links to their upper tiers.

Path selection: In a single ISP (router-level ISP and synthetic topologies), we can easily select single shortest paths, k shortest paths, and optimal paths. In the AS-level topology, we choose paths based on policies: *(1) Optimal case (mcf):* To understand the throughput upper bound, we view the AS graph as an undirected graph, ignoring all BGP policies. *(2) Single path case (1path):* We model the path selection based on the Gao-Rexford conditions [34] (following Valley free, local preference, shortest path, and the other tie breaking policies). *(3) k shortest paths:* To understand the best possible multipath choices that can be realized by multipath BGP protocols [7, 35], we *only* enforce the valley free policies for k shortest paths.

[5]Fixed orbit reports 144K peering links in 2012 [26].

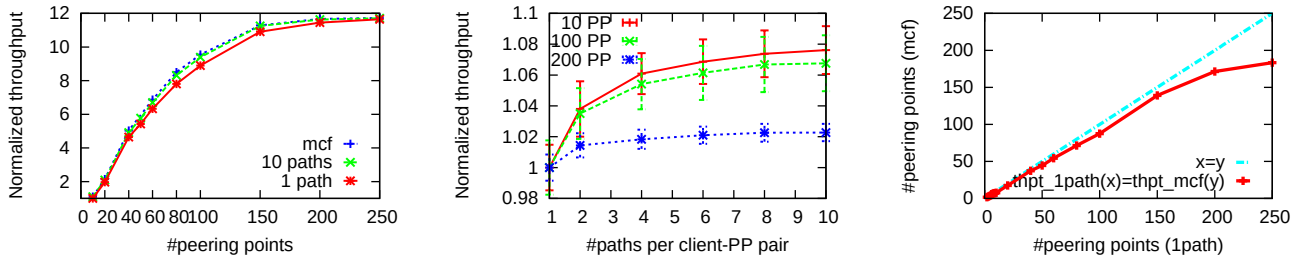
[6]We ignore the links inside an AS to reduce the problem size. The result should not be affected because peering links between ASes are more likely to become the bottleneck than the links inside ASes.

[7]Following the guidelines in BRITE, in the random graph, we use the waxman model and set the parameters $\alpha = 0.2, \beta = 0.15$. In the hierarchical graph, we choose 10 ASes, each AS with an average of 50 nodes. Within the ASes and among the ASes, we create links based on the power law model.

[8]Note that the link capacity is the bandwidth that can be used for a content delivery application. Only the relative values of capacity among links affect our results.

Topology	#nodes	#links	$\frac{Thpt_{mcf}-Thpt_{1path}}{Thpt_{1path}}$ (#PP=10..250)	$\frac{Thpt_{20PP}-Thpt_{10PP}}{Thpt_{10PP}}$ (1path)	$\frac{Thpt_{100PP}-Thpt_{10PP}}{Thpt_{10PP}}$ (1path)
AS-level Internet (Akamai: 1026 ASes)	∼12K	∼120K	18.1%	Akamai/Limelight=205.4%	
AS-level Internet (Limelight: 216 ASes)	∼12K	∼120K	27.2%		
power law (uniform cap dist.)	500	997	0.7%-10.3%	96.8%	789.5%
power law dense (uniform cap dist.)	500	1990	20.2%-32.2%	64.1%	195.8%
power law (exp cap dist.)	500	997	9.2%-15.1%	157.1%	298.8%
power law (pareto cap dist.)	500	997	0.7%-15.1%	149.3%	322.1%
random (uniform cap dist.)	500	1000	2.3%-69.4%	96.4%	659.9%
hierarchy (uniform cap dist.)	500	1020	3.5%-26.4%	120.0%	745.6%
hierarchy (high peering cap)	500	1020	1.9%-31.7%	95.6%	225.7%
AS 13367 Comcast (uniform cap dist.)	382	421	1.8%-11.8%	81.2%	424.7%
AS 12064 Cox (uniform cap dist.)	326	378	110.3-584.0%	337.0%	2755.2%

Table 2: Increasing #PP (Col. 5, 6) is more effective than increasing #paths (Col. 4) in achieving throughput over various topologies.



(a) Increasing #PP (b) Increasing #paths (c) #PP of 1path vs mcf for the same thpt

Figure 3: Multipath has little throughput improvement over increasing #server locations (power law graph)

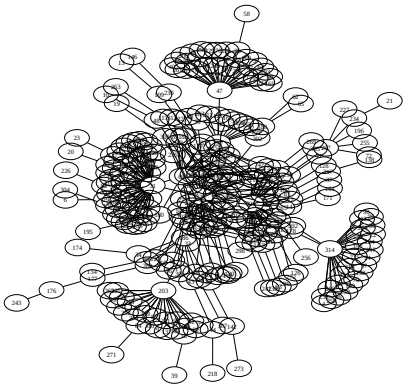


Figure 4: Cox topology

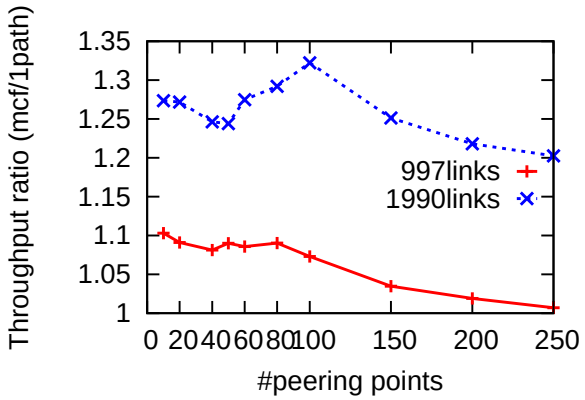


Figure 5: Effect of multipath

4.3 Quantify the throughput improvement

With evaluations on various settings, we show that doubling the number of peering points from 10 to 20 can improve the throughput linearly, i.e., between 64.1%-157.1%, while optimal path selection can only improve the throughput by 0.7%-69.4% in most cases (Table 2). One interesting example is the Cox topology, where both multipath and more PPs have better throughput improvement because there is a few internal routers with high degree.

More peering points also reduce the delay while mulitpath often increases the delay. Moreover, with more peering points, the throughput benefits of multipath decreases in most cases. One exception is the power law dense graph with less than 100 PPs, the benefits of multipath increase slowly with increasing PPs.

Increasing #paths vs increasing #peering points (PPs): In the AS-level Internet topology, the optimal multiple path solution (*mcf*) improves the throughput over the single shortest path (*1path*) by 18.1% on average for Akamai and 27.2% for Limelight. In the power law topology (Figure 3(a)), the throughput increases by 96.8% from 10 to 20 PPs, and by 789.5% from 10 to 100 PPs. However, *mcf* can only improve the throughput by 0.7%-10.3%. As shown in Figure 3(b), with 10 PPs, 10 shortest paths improve the throughput by 7.6% compared with *1path*, approximating the optimal multipath solution (*mcf*).

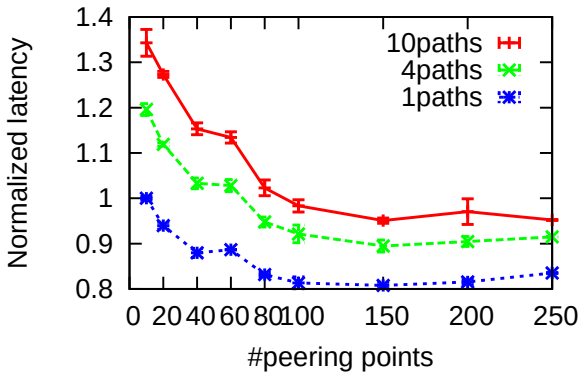


Figure 6: Effect on delay

Figure 3(c) quantifies the number of PPs with *mcf* needed to achieve the same throughput with the number of PPs with *1path*. For example, 100 PPs with *1path* can reach the same throughput with 88 PPs with *mcf*. The gap between the two curves in the figure indicates the benefits of multipath over single path. The gap is small with less than 100 PPs, indicating few benefits from multipath. Beyond 100 PPs, the gap becomes larger. This is because the network capacity is mostly explored and the throughput improvements from both more PPs and *mcf* become small (within 5%). As a result, we need a lot more PPs to match the same throughput improvement when PPs are small.

Effect of network topology: The throughput benefits of more paths and more PPs differ with different link capacities and topologies. For example, the *mcf* has more benefits in the power law dense graph, while increasing PPs works better in the power law graph with fewer links. This is because multipath has more benefits in networks with more path options and higher aggregate link capacity. One interesting topology is the Cox network with a few high-degree internal routers (Figure 4). With a uniform distribution of link capacity, the throughput from *1path* to *mcf* increases by 110.3%-584.0%, which is comparable to double PPs from 10 to 20 (337.0%). This is because even with more PPs, the traffic still have to traverse through the same set of internal routers (i.e., the min-cut size remains almost the same).

Effects of multipath with the increase of #PPs: In general, the more PPs, the less useful is multipath, as shown in Figure 5. This is because more PPs increases the network utilization, leaving less space for multipath to improve throughput. However, for the graph with more links (power law dense), when the number of PPs grows from 10 to 100, the benefits of multipath increase by about 5%. This is because with the growth of PPs, the throughput bottleneck moves from the edge to the network, where multipath can have more benefits.

Effect on delay: In practice, CDNs may also care about delay in addition to the throughput metric. With more PPs, we can improve the throughput and reduce the delay at the same time. In contrast, multipath increases the throughput at the expense of increasing the delay. In Figure 6, we assign the link latency of the realistic topologies based on the geolocation information of the routers, and the latency of synthetic topologies based on the uniform distribution with an average of 50 ms. The propagation delay drops by 20% from 10 to 100 PPs, but remains the same beyond 100 PPs. In contrast, increasing the number of paths from 1 to 10 increases the delay by about 35%.

4.4 Understand the effect of client settings

There are three key client-side settings that may affect the throughput: (1) client location popularity (i.e., some client location has more demands than others); (2) content popularity (especially contents whose accesses follow a long-tail distribution); (3) Client demand changes over time (especially during flash crowds). Considering these settings, we show that the gap of throughput benefits between more PPs and multipath becomes even larger.

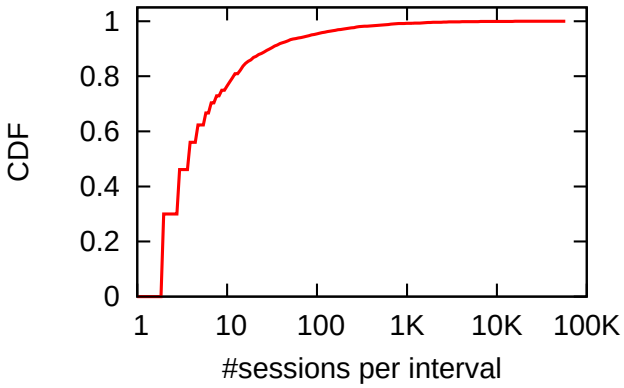
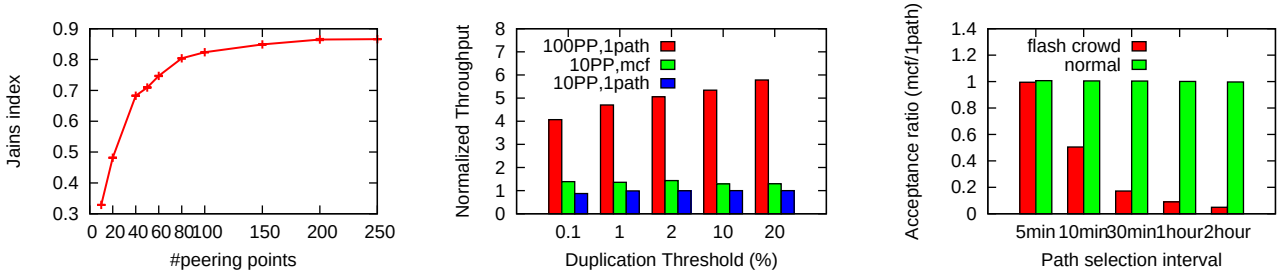


Figure 8: Client sessions distribution across ASes

We study three sets of session-level traces from Conviva collected between Dec. 2011 and April. 2012: the *normal trace*, the *long-tail trace* with a long-tail content distribution, and the *flash-crowd trace* containing flash crowds during a popular local event. We calculate the number of sessions from each AS based on the client IP addresses every five-minute interval. The clients of each trace spread over 12K ASes. Figure 8 shows the CDF of the average # sessions per interval at each AS for the normal trace. The normal trace and the long-tail trace contain an average 340K sessions per interval and remain stable across intervals. The flash crowd trace contains an average of 450K sessions and a peak of 630K sessions per interval. We set the upper and lower bounds of the sending rate per session as 300 kbps and 3 Mbps respectively. We time the rate bound with the number of sessions to get the throughput demand bounds at each location $Lower_c$ and $Upper_c$. For those synthetic topologies, we map these throughput demands randomly to client locations.

Fairness among client locations: One concern is whether the throughput gains of more PPs come from a few popular client locations. We evaluate the normal trace in the power-law topology, where client locations have diverse demands. The throughput improves by 494.9% from 10 PPs to 100 PPs, but only by 0.3% with *mcf*. This is consistent with the results in Table 2 without client constraints. We also calculate the Jain's fairness index across the throughput of all the clients in Figure 7(a). When there are more than 20 PPs, the throughput is relatively equal among clients (The



(a) Fairness across client locations (b) Long-tail distribution of content (c) Flash crowds

Figure 7: Effect of client-side settings

index is above 0.5). This is because there is a large amount of clients (200 out of 500 nodes) evenly distributed across the network. These clients have equal chances to use the network bandwidth, as long as their access links are not bottlenecked. The broad spread of CDN clients across the Internet is true in practice because of the growth of video streaming traffic as observed from our analysis on Conviva traces.

The long-tail distribution of contents: The throughput performance is highly related to content replication decisions, which in turn depend on content popularity. We use the long-tail trace to understand the effect of content popularity. We set a replication threshold δ based on server capacity. We replicate top δ popular contents across all the server locations, but randomly pick a single server to store the other contents. As shown in Figure 7(b), with different levels of replications (δ=0.1%-20%), more PPs always improve throughput more than *mcf*. When there are fewer replications (δ=0.1%), the throughput improvement of *mcf* increases, and the improvement of more PPs decreases. This is because without replication the content delivery is closer to the single-source traffic.

Flash crowds: It takes different time intervals to adjust peering point selection and path selection to client demand changes. We compare two cases: (1) the traditional CDNs that can only select peering points with a single path between each client-PP pair. (2) the ISP-CDNs that can perform joint server and path selection. It takes longer time to switch paths because it requires path computation, network reconfiguration, and routing propagation. Therefore, we fix the server/peering point selection interval as 5 minutes for both cases, but set the path selection intervals ranging from 5 minutes to 2 hours for ISP-CDNs. During the periods when ISP-CDNs only perform server selection, we keep the routing decisions based on the last time when a joint optimization is performed and kept unchanged throughput the entire interval.[9] We use equal splitting across k shortest paths to approximate the optimal path selection (calculated by the multi-commodity flow problem), as we observe a small gap between the two solutions with a large k (e.g., k=10) according to our previous evaluations (e.g., Figure 3 (a) and (b)). We use the normal and flash-crowd traces in the power law graph with 50 PPs.

[9]However, it might not able to learn the routing information sometimes, for instance, a client may not appear (i.e., zero demand) at time t, but have a large demand at time $t+1$. The routing decision for this client at time t is arbitrary, and hence can be quite sub-optimal at time $t+1$.

Since the client demands change significantly during the flash crowd event, it is sometimes impossible to satisfy all the clients if the demand is very high. To understand the potential of delivering flash crowd events, we define the *acceptance ratio* (a^{sc} defined in Section 3) as the fraction of demands the CDN can deliver. Figure 7(c) compares the acceptance ratios with different path selection intervals with the single path case with PP selection only. As expected, when the path selection interval is the same with PP selection, *mcf* only has 0.7% improvement in acceptance ratio than *1path* case. Worse still, during the flash crowd event, when the path selection interval increases, the acceptance ratio drops significantly. For example, with 10 minute interval, the ratio drops by 50% from *1path*'s ratio; with 1 hour interval, the ratio is only 8% of that of *1path* case. This is because when the traffic changes rapidly during flash crowds but the ISP-CDN cannot adjust its path selection quickly, the routing decision at one time is far from optimal for the traffic at the following time intervals. In contrast, the CDN who takes a simpler approach of fast and optimal peering selection at the edge can achieve a much higher acceptance ratio.

5. CENTRALIZED & DISTRIBUTED CDNS

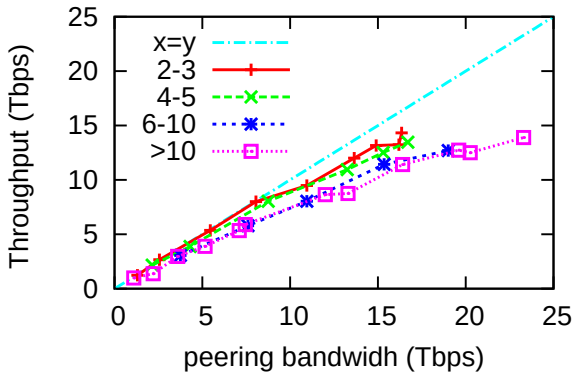


Figure 9: Comparing CDN throughput with different number of server locations (power law graph)

There are two ways to increase peering points to improve throughput: the more distributed CDNs (e.g., Akamai) increase the number of server locations to increase peering

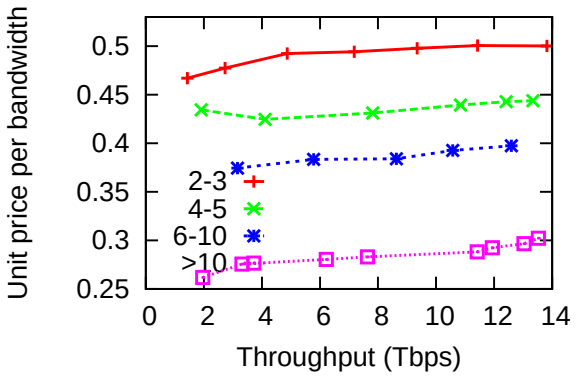


Figure 10: Comparing CDN cost (Polynomial cost) with different number of server locations (power law graph)

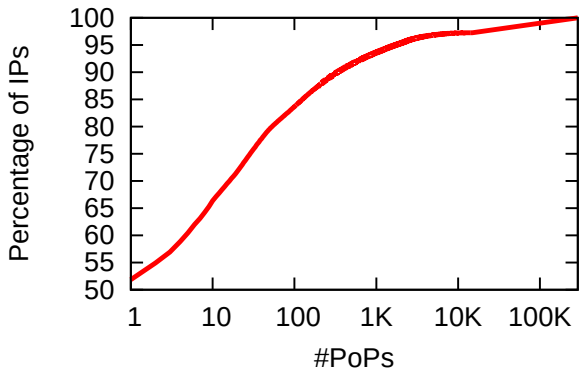


Figure 11: Percentage of IP addresses top x PoPs can reach

points; the more centralized CDNs (e.g., Limelight) only deploy servers at a few locations but set up many peering points at each location. Our evaluation of various types of cost functions shows that given a total of 150 peering points, more centralized CDNs achieve 22.5% less throughput than more distributed CDNs, but can reduce the operating cost per bit of bandwidth by as much as 69%. We also analyze the Internet topology to understand the minimum number of server locations a CDN should have in practice. Our results are consistent with the fact that most CDNs today choose a more centralized deployment for throughput-oriented traffic.

5.1 Comparing the throughput

To understand the general tradeoff of CDN design rather than looking into specific CDNs, we study different approaches of picking server locations based on the peering points (PPs) at each location. We classify the potential server locations into four categories: the locations with 2-3, 4-5, 6-10, and >10 PPs at each location. We fix the total number of peering points, and then pick more server locations with 2-3 PPs per location to model more distributed CDNs. We can also pick fewer locations >10 PPs each to model more centralized CDNs, or pick the other server locations with 4-5, 6-10 PPs to model the CDNs in between. For example, to get 150 total peering points, the more distributed CDN has 80 server locations with 2-3 PPs each, while the more centralized CDN has 15 server locations with >10 PPs each.

Cost function	Avg unit price				
	2-3	**4-5**	**6-10**	**>10**	$\frac{P_{>10}}{P_{2-3}}$
Bw: Poly. bw^{α},($\alpha = -\log 2$)	49.0	43.6	38.6	28.5	58.2%
Bw: Amazon [36]	5.4	5.1	4.3	3.1	57.4%
Bw: OC3 [37]	89.4	59.8	44.3	31.9	35.7%
Mngt: $c \cdot e^{a \cdot bw}$,$a < 0$	30.3	26.7	25.4	25.0	82.5%
Elec: $c \cdot \log(bw + a)$,$c < 0$	64.5	57.4	49.6	28.5	44.2%
Linear $c \cdot bw + a$,$c < 0$	89.5	84.6	76.6	27.5	30.7%
Per-PP bw: $(bw/n_p)^{\alpha}$	64.6	68.6	72.3	82.4	1.3

Table 3: Cost comparison of centralized and distributed solutions. (Each cost function value is normalized into [0,100]. Power-law graph)

Given the same number of peering points with the same aggregate peering bandwidth, different types of CDNs have different throughput (Figure 9). More distributed CDNs with more locations ("2-3") can better explore the available bandwidth inside the network and thus provide better throughput than more centralized CDNs with fewer locations (">10"). For example, when there are around 150 peering points with a total of about 78 Tbps peering bandwidth, the more distributed solution ("2-3") reaches 8.0 Tbps, while the more centralized solution (">10") reaches 6.2 Tbps.

5.2 Comparing per-bandwidth cost

Although more distributed CDNs can achieve a better throughput, they often incur more operating cost. By evaluating different types of cost functions, we conclude that to achieve the same throughput, more centralized CDNs are cheaper to build than more distributed CDNs.

We model the operating cost per *unit traffic* for a server location as $f(bw)$, where bw is the traffic volume at this location. In general, $f(bw)$ is a decreasing function of the used bandwidth. We introduce a family of cost functions based on sensitivity of unit price with respect to throughput as shown in Table 3: The polynomial cost function[10], step functions (Amazon [36] and OC3[37]) to model bandwidth cost, exponential function for management cost, log function for electricity cost, and the linear cost function. The content replication cost can be modeled as one or a combination of the above functions depending on CDN designs.

More centralized CDNs incur less operating cost than more distributed CDNs: As shown in Table 3, the more centralized CDNs consistently achieve the lowest price per bit of bandwidth, across all cost types and different values of aggregate throughput. For example, in Figure 10, fixing the total number 150 PPs, the unit price of more centralized solution (15 locations with > 10 PPs each) is only 58.3% of the more distributed solution (80 locations with 2-3 PPs each). This is because with a few server locations, more centralized CDNs already achieve good throughput at each location and hence attain a low price per bit of bandwidth. In contrast, due to the low traffic volume at each location, more distributed CDNs ("2-3") the highest unit price.

The cost advantage of more distributed CDNs is independent of CDN sizes and aggregate throughput. We have

[10]We get the function based on the transit bandwidth cost—1$ at 1G, 0.5$ at 10G, and 0.25$ at 100G [15].

consistent observations with different parameter choices in all the cost functions listed in Table 3, and a linear combination of these functions. The results also hold for multiple paths because multipath has limited effect on throughput.

More centralized CDNs have a higher price variation w.r.t. throughput: We also observe that the unit price for more centralized CDNs grows faster with the throughput increase than more distributed CDNs. For example, in Figure **??**(b), the unit price grows by 7.1% for more distributed CDNs ("2-3") from 2 Tbps to 14 Tbps, but grows by 15.4% for more centralized CDNs (>10). With the linear cost function, the price increases by 300% from 2 Tbps to 14 Tbps for the more centralized CDNs. This is because more centralized CDNs cannot keep up with the throughput growth with more peering points (i.e., achieves less throughput with the same total PPs), and thus incur a faster unit price growth. Note that even with faster growth, when we push the throughput close to the network capacity, the unit price for more centralized CDNs is still lower than more distributed CDNs.

More distributed CDNs have a lower unit price when CDNs pay bandwidth cost for each peering point separately: At each server location, CDNs may pay for peering bandwidth purchased from different ISPs separately. We model the per-PP bandwidth cost as $f(bw/n_p) = (bw/n_p)^{\alpha}$, where n_p is the number of peering points a server location has. Opposite to other cost functions, here more centralized CDNs have higher unit price than more distributed CDNs (Table 3). This is because given the same number of PPs, more distributed CDNs have a higher per-PP traffic volume than more central CDNs and thus incur a better unit price.

5.3 How many server locations in practice?

Based on the tradeoffs between throughput and cost, the more centralized a CDN is, the less cost it takes to achieve a given throughput. However, in practice, we cannot just have a single server location because it is almost impossible to find a single best position to peer with all the ISPs. One natural question is then what is the minimum number of locations that a CDN should deploy its servers. We analyze the PoP-level Internet topology of about 12K ASes with 177K PoPs, and the IP addresses owned by each AS [28]. Figure 11 shows that the top 54 PoPs already have peering links to those ISPs who own over 80% of the IP addresses. This means a CDN does not need to deploy its servers at hundreds or thousands of locations to directly reach the majority of clients.

6. RELATED WORK

Understanding CDN performance from measurement: Many works take a *measurement-based* approach to understand CDN performance and the CDN-ISP relations from the *delay* perspective [38, 39, 40, 41]. Unfortunately, using measurement studies alone is not sufficient to understand CDN design choices. For example, [40, 41] uses a combination of web-browser clients, SpeedTest [42] servers located near Akamai servers, and active probing to measure the delay and throughput provided by both of Akamai and Limelight. However, despite these careful measurements, the study provides little information about the routes used to deliver content to various clients, the overall throughput, and the costs incurred by CDNs [43]. Instead of studying an existing CDN, we choose a simple model to understand the fundamental design choices of CDNs in the design space, and simulate the model using video streaming traces and a diversity of network topologies (with varying CDN peering points and server locations).

Better ISP support for network performance: There have been many works on improving path selection and providing multiple paths to improve network performance [5, 6, 7, 8, 9, 10]. While better path selection does provide better performance and more reliability for single-source, single destination applications, our evaluations show that even optimal multipath solution has limited impact for CDNs with many peering points and massive clients.

The papers [44, 45] show the benefits of multihoming in improving delay and throughput compared to overlay routing. We reach a more general conclusion that more peering points at the edge can improve the throughput much more than multiple paths inside the network across a wide variety of network topologies and settings. We also study the right number of CDN server locations considering the throughput and cost tradeoffs.

7. CONCLUSION

It is crucial to revisit the design space for CDNs, with more throughput-oriented applications (e.g., video steaming). We provide a simple model of essential CDN designs and perform extensive evaluations on a variety of network topologies (with varying numbers of CDN peering points and server locations), operating cost models, and client video streaming traces. Our results show that adding new peering points helps CDNs improve the throughput most. On the other hand, ISP-CDNs could not benefit much from their ability to optimize the routes. In addition, CDNs should choose the more centralized CDN design with many peering points, because it requires lower operating cost than the more distributed approach to achieve the same throughput.

8. ACKNOWLEDGMENT

We thank our shepherd Timothy G. Griffin, the anonymous reviewers, John Chuang, Ali Ghodsi, Xin Jin, Srinivas Narayana, Peng Sun, and Vytautas Valancius, for their comments on earlier versions of this paper. We especially thank Dahai Xu for improving the scalability of our optimization code and David Choffnes for providing us the locations of Akamai and Limelight servers from their PlanetLab measurement.

9. REFERENCES

[1] Sandvine, "Global Internet phenomena report." Technical report, 2011.

[2] G. Maier, A. Feldmann, V. Paxson, and M. Allman, "On dominant characteristics of residential broadband Internet traffic," in *IMC*, 2009.

[3] `http://youtube-global.blogspot.com/2010/05/at-five-years-two-billion-views-per-day.html`.

[4] E. Schonfeld, "Cisco: By 2013 video will be 90 percent of all consumer ip traffic and 64 percent of mobile." `http://tinyurl.com/nw8jxg`, 2009.

[5] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley, "Design, implementation and evaluation of congestion control for multipath TCP," in *NSDI*, 2011.

[6] D. Xu, M. Chiang, and J. Rexford, "PEFT: Link-state routing with hop-by-hop forwarding can achieve optimal traffic engineering," in *IEEE INFOCOM*, 2008.

[7] W. Xu and J. Rexford, "MIRO: Multi-path interdomain routing," in *SIGCOMM*, 2006.

[8] M. Motiwala, M. Elmore, N. Feamster, and S. Vempala, "Path splicing," in *SIGCOMM*, 2008.

[9] "Routescience." `http://support.avaya.com/css/Products/P0345`.

[10] "Cisco optimized edge routing (oer)." `http://www.cisco.com/en/US/products/ps6628/products_ios_protocol_option_home.html`, 2010.

[11] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "The flattening internet topology: Natural evolution, unsightly barnacles or contrived collapse?," in *PAM*, 2008.

[12] "The internet topology data kit, caida." `http://www.caida.org/data/active/internet-topology-data-kit`.

[13] Y.-J. Chi, R. Oliveira, and L. Zhang, "Cyclops: The internet as-level observatory.," in *CCR*, 2008.

[14] A. J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante, "Drafting behind Akamai (travelocity-based detouring)," in *SIGCOMM*, 2006.

[15] `www.conviva.com`.

[16] "Private conversations with akamai operators.."

[17] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," in *SIGCOMM*, 2011.

[18] "Adobe. http dynamic streaming." `http://www.adobe.com/products/httpdynamicstreaming`.

[19] `http://techblog.netflix.com/2011/01/netflix-performance-on-top-isp-networks.html`.

[20] E. Nygren, R. K. Sitaraman, and J. Sun, "The Akamai network: A platform for high-performance internet applications," in *ACM SIGOPS Operating Systems Review*, 2010.

[21] D. Clark, W. Lehr, and S. Bauer, "Interconnection in the internet: The policy challenge," *Research Conference on Communication, Information and Internet Policy*, 2011.

[22] `http://drpeering.net/AskDrPeering/blog/articles/Peering_vs_Transit___The_Business_Case_for_Peering.html`.

[23] `http://www.akamai.com/dl/investors/10k_2010_1.pdf`.

[24] T. Leighton and S. Rao, "Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms," in *Journal of the ACM*, 1999.

[25] M. Caesar, M. Casado, T. Koponen, J. Rexford, and S. Shenker, "Dynamic route computation considered harmful," in *CCR*, 2010.

[26] `http://www.fixedorbit.com/stats.htm`.

[27] `www.cs.bu.edu/brite`.

[28] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: An information plane for distributed services," in *OSDI*, 2006.

[29] "Private conversation with iplane group.."

[30] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker, "On selfish routing in internet-like environments," in *SIGCOMM*, 2003.

[31] L. Li, D. Alderson, W. Willinger, and J. Doyle, "A first-principles approach to understanding the internet's router-level topology," in *SIGCOMM*, 2004.

[32] T. Hirayama, S. Arakawa, S. Hosoki, and M. Murata, "Models of link capacity distribution in isp's router-level topology," in *International Journal of Computer Networks and Communications (IJCNC)*, 2011.

[33] `www.quova.com`.

[34] L. Gao and J. Rexford, "Stable internet routing without global coordination," in *TON*, 2001.

[35] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet routing," in *SIGCOMM*, 2009.

[36] `http://aws.amazon.com/ec2/\#pricing`.

[37] D. K. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang, "Optimizing cost and performance for multihoming," in *SIGCOMM*, 2004.

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

[39] V. K. Adhikari, S. Jain, and Z.-L. Zhang, "YouTube traffic dynamics and its interplay with a tier-1 ISP: An ISP perspective," in *IMC*, 2010.

[40] C. Huang, Y. A. Wang, J. Li, and K. W. Ross, "Measuring and evaluating large-scale CDNs," in *Microsoft Research Technical Report MSR-TR-2008-106*, 2008.

[41] Y. A. Wang, C. Huang, J. Li, and K. W. Ross, "Estimating the performance of hypothetical cloud service deployments: A measurement-based approach," in *IEEE INFOCOM*, 2011.

[42] `www.speedtest.net`.

[43] "Akamai and limelight say testing methods not accurate in microsoft research paper." `http://blog.streamingmedia.com/the_business_of_online_vi/2008/10/akamai-responds.html`.

[44] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman, "A measurement-based analysis of multihoming," in *SIGCOMM*, 2003.

[45] A. Akella, J. Pang, B. Maggs, S. Seshan, and A. Shaikh, "A comparison of overlay routing and multihoming route control," in *SIGCOMM*, 2004.