Collaboration Opportunities for Content Providers and Network Infrastructures

Benjamin Frank, Ingmar Poese, Georgios Smaragdakis
Vinay Aggarwal, Anja Feldmann, Steve Uhlig
Bruce Maggs, Fabian Schneider

April 10, 2013

1 Motivation

The Internet is a hugely successful human made artifact that has changed the society fundamentally. Consider the effect a prolonged outage of the Internet would have: (1) Some of the youngsters wouldn't know anymore how to interact with their peers and how to spend their leisure time as they increasingly rely on social networks, online games, YouTube, and other online entertainment offers. (2) Manufacturing would hit a roadblock as the communication path within and between companies increasingly relies on the Internet. (3) Control of critical infrastructures may become a problem as it starts to increasingly rely on the Internet for gathering input data and propagating control information.

In becoming such a hugely successful infrastructure the usage of the Internet and thus the structure of the Internet has also undergone continuous changes. Usage has changed from email and FTP in the early days, to the World Wide Web (WWW) from 1995 to 2000, to Peer-to-Peer (P2P) from 2000 to 2007, back to the WWW since 2007. These changes are in part driven by the Internet users interests as well as how content, including user generated content is made available.

When considering the current application mix and traffic streams in the Internet, the latest buzz is that "Content is King" just as Bill Gates predicted in his essay from 1996. Hereby, the term content has to be seen very broadly and encompasses everything from commercially prepared content, e.g., broadcast and interactive TV, news, and software, to user-generated content, e.g., videos uploaded to YouTube, photos uploaded to Flickr, to interactive activities, e.g., online games. Or to quote Bronfman, the head of a major music producer and distributor: "What would the Internet be without 'content'? It would be a valueless collection of silent machines with gray screens. It would be the electronic equivalent of a marine desert—lovely elements, nice colors, no life. It would be nothing."

The idea of content being the fundamental principle to design future Internet architecture for comes as no surprise. In fact, the idea of Content-Centric Networking (CCN) [107] solely builds upon this principle. However, change takes time, and when hundreds of million of devices are involved, change can only be made slowly. Before such a novel and radically different architecture such as CCN is available or potentially deployable, the Internet in its current state has to cope with the challenge of delivering ever increasing amounts of content to the Internet users.

Accordingly, it appears that solely providing connectivity to end-users is no longer sufficient for Internet Service Providers (ISPs). Yet, connectivity is a crucial ingredient and some authors, e.g., Andrew Odlyzko [171] claim that enabling communication is the main task of the Internet network infrastructure. In his paper "Content is not king" he claims that "Content will have a place on the Internet, possibly a substantial place. However, its place will likely be subordinate to that of business and personal communication".

At this point it is crucial to realize that the producers of content are usually not affiliated with the operators of todays Internet infrastructure. None the less, both types of operation depend on each other. In fact, neither the Internet infrastructure operators nor the content producers can be successful without the other. After all, the content providers want to ensure that their content gets to the Internet users with reasonable performance for which they need to rely on the network infrastructure. Vice versa, the network infrastructure providers have to transport the content and manage the infrastructure to satisfy the demand for content of their subscribers. It is this symbiosis of the two parties that motivates our work on Internet content as well as collaboration between content producers and network operators.

Outline: We start this chapter with a short introduction in Section 2. We then set the stage by providing an overview on todays Internet network infrastructure, background in how Internet service provider perform traffic engineering and review the Domain Name System (DNS)—an essential component of any Web based content delivery architecture—in Section 3. Next, we review current trends in Internet traffic and the application mix as well as traffic dynamics in Section 4 and 5.

We finish the overview with a brief summary on the background of content delivery in Section 6. Here, we assume that the reader is familiar with the basic architecture of the Web. There are excellent text books on this topic, e.g., [124]. Given that there is no universally accepted single system for content delivery we provide a general high level description on how different Content Delivery Infrastructures work. However, since there are also many Peer-to-Peer based content delivery systems we provide a short review of the basic P2P architectures as well. For additional background on P2P we refer the reader to, e.g., [39, 206].

An overview of the current content delivery spectrum is presented in Section 7. Here we discuss various types of Content Delivery Infrastructures (CDIs) which ranges from Web based Content Distribution Networks (CDNs) over Hybrid CDNs to Peer-to-Peer (P2P) systems. Furthermore, in Section 8 we turn to the challenges that each party involved in Internet content delivery faces separately today.

Finally, we turn to the state of the art of collaboration between networks and content providers. Similarly to the challenges, we outline the collaboration incentives for each member of the content delivery landscape in Section 9. Next are the collaboration schemes that have been discussed in research as well as at the Internet Engineering Task Force (IETF) in Section 10. We briefly introduce the well known approaches and summarize their key functions. We then pick two collaboration schemes, namely the P2P Oracle and the Provider-aided Distance Information System (PaDIS) for a case study. In Section 11 we discuss the P2P Oracle with regards to its effect on the P2P system as well as the network operators. Likewise, the second case study discusses the model of the Provider-aided Distance Information System in Section 12, including a large scale analysis based on the real traffic traces. Section 13 outlines a possible future direction for collaboration between content providers and network operators before we conclude this part of the chapter in Section 14.

Summary: This chapter builds upon the students basic knowledge of how the Internet infrastructure operates, i.e., as a network of networks. After reading this chapter the students should have a fundamental understanding about how today's content distribution via the Internet works, what the challenges are, and which opportunities lie ahead. Moreover, the chapter points out how all parties—including the end-users—can benefit from collaboration of ISPs and Content Providers . Indeed, simple almost intuitive means will enable such collaboration.

Contents

1 Motivation **1**

2 Introduction **8**

3 Internet Network Infrastructure **10**
3.1 Traffic Engineering in an AS 11
3.2 Caching Traffic for AS Management 13
3.3 Domain Name System Basics 13

4 Traffic Trends: Overall **17**
4.1 Internet Traffic Characteristics 17
4.1.1 Application Mix 17
4.1.2 Content-types in the Internet 19
4.1.3 Time-of-day Effects 20
4.1.4 Flow Sizes and Durations 21
4.2 Traffic Analysis Tools 22
4.2.1 LIBPCAP and TCPDUMP 22
4.2.2 WIRESHARK 23
4.2.3 BRO 23

5 Traffic Trends: Content Server Diversity **25**
5.1 Residential ISP Traces 25
5.2 Server Diversity and DNS Load Balancing 27
5.3 Server Location Diversity 27
5.3.1 Alternative DNS Resolvers 32
5.4 Impact on Traffic Localization 32
5.5 Summary 33

6 Content Delivery: An Overview **34**
6.1 Terminology 35
6.2 Content Delivery Networks 35
6.3 Peer-to-Peer Networks 39

7 Content Delivery Landscape **42**
7.1 Independent Content Distribution 43
7.2 ISP-operated CDIs 44
7.3 Emerging Trends in CDI Architectures 44
7.3.1 Hybrid Content Distribution 44
7.3.2 Licensed CDNs 45
7.3.3 Application-based CDIs 45
7.3.4 Meta-CDIs 45
7.3.5 CDI Federations 46
7.4 Hybrid P2P-CDN Infrastructures 46

8 Challenges in Content Delivery **47**
8.1 Content Delivery Networks (CDNs) 47
8.2 Peer-to-Peer Networks (P2P) 47
8.3 Internet Service Providers (ISPs) 49
8.4 Summary 49

9 Incentives for Collaboration **51**
9.1 Incentives for CDIs 51
9.2 Incentives for ISPs 52
9.3 Effect on End-users 52

10 Opportunities for Collaboration **53**
10.1 P2P Oracle Service 53
10.2 Proactive Network Provider Participation for P2P (P4P) 54
10.3 Travelocity-based Path Selection 54
10.4 Ono 55
10.5 Provider-aided Distance Information System (PaDIS) 55
10.6 Application-Layer Traffic Optimization (ALTO) 57
10.7 Content Delivery Networks Interconnection (CDNI) 58

11 Use Case for Collaboration: P2P **59**
11.1 Realizing an Oracle Service: 60
11.2 Using Bandwidth to Select Neighbors 60
11.3 Benefit of the Oracle from P2P and ISP Point of View 61
11.3.1 Metrics 61
11.3.2 Simulation Topologies 63
11.3.3 Overlay / Underlay Graph Properties 64
11.4 Simulations with an Actual P2P System 67
11.4.1 Gnutella and SSFNet 67
11.4.2 Simulation Setup 68
11.4.3 Results 69
11.5 Testlab Experiments 73
11.6 Benefit of the Oracle from a Users Point of View 74
11.6.1 Topology Models 74
11.6.2 User Behavior Models 75
11.7 Results—Users Perspective 76
11.7.1 Variation in Topology 76
11.7.2 Variation in User Behavior 77
11.8 Summary 79

12 Use Case for Collaboration: Traffic Engineering **80**
12.1 The CaTE Approach 80
12.2 A Prototype to Support CaTE 81
12.2.1 Architecture 82
12.2.2 Network Monitoring 82
12.2.3 Query Processing 83

12.2.4 Communication Interfaces 83
12.2.5 Privacy and Performance 84
12.2.6 Deployment 84
12.2.7 Operation 85
12.3 Modeling CaTE 85
12.3.1 Traffic Engineering 85
12.3.2 Definition of CaTE 87
12.3.3 CaTE and Traditional TE 87
12.3.4 CaTE and Multipath Routing 88
12.3.5 CaTE and Oscillations 88
12.4 CaTE Algorithms 88
12.4.1 Connection to Restricted Machine Load Balancing 88
12.4.2 Online Algorithm and Competitiveness 90
12.5 Estimating the Benefit of CaTE with Passive Measurements 90
12.5.1 Linear Programming Formulation 91
12.5.2 Approximation Algorithms 91
12.6 Impact of Collaboration: Traffic Engineering 93
12.6.1 CaTE in ISP1 96
12.6.2 CaTE with other Network Metrics 100
12.6.3 CaTE in AT&T and Abilene 100
12.6.4 CaTE and Popular Applications 102
12.6.5 Active Measurements in ISP1 102
12.6.6 Case Study: Netflix in ISP1 103

13 Future of Collaboration **107**
13.1 The New Cloud: Microdatacenters Deep Inside the Network 107
13.1.1 The ISPs Proposal 108
13.1.2 Microdatacenter Specifications 109
13.1.3 Microdatacenter Network Footprint 109
13.2 On-Demand Service Design 110
13.2.1 Microdatacenter Slice 111
13.2.2 On-Demand Service Realization 111
13.2.3 Service Interfaces 112
13.2.4 Billing 112
13.3 On-Demand Service Optimization 113
13.3.1 The Slice Location Problem 113
13.3.2 Assigning Users to Slices 114
13.4 Network Platform as a Service (NetPaaS) 115
13.4.1 Resource Discovery 115
13.4.2 Slice Allocation 116
13.4.3 User-Slice Assignment 118
13.5 Evaluation of NetPaaS 119
13.5.1 Evaluation Framework 120
13.5.2 Evaluation Dataset 120
13.5.3 CDN Evaluation: Extreme Cases 122
13.5.4 CDN Evaluation: Limited Slice Locations 122

13.5.5 CDN Evaluation: Prediction Accuracy 124
13.5.6 Overall Evaluation: Scaled CDN Demands 126

14 Conclusion 127

2 Introduction

Recent traffic studies [87, 128, 183] show that a large fraction of Internet traffic is due to content delivery and is originated by a small number of Content Providers (CPs). Major CPs are highly popular rich media sites like YouTube and Netflix, One-Click Hosters (OCHs), e.g., Uploaded or the newly announced service Mega, as well as Content Delivery Networks (CDNs) such as Akamai or Limelight and hyper-giants, e.g., Google, Yahoo! or Microsoft. Gerber and Doverspike [87] report that a few CPs account for more than half of the traffic of a US-based Tier-1 carrier. Poese et al. [183] report a similar observation from the traffic of a European Tier-1 carrier. Labovitz et al. [128] infer that more than 10% of the total Internet inter-domain traffic originates from Google, and Akamai claims to deliver more than 20% of the total Internet Web traffic [170]. Netflix, offering a high definition video streaming service hosted on CDN infrastructures, is responsible for around 30% of the traffic in North America during peak hours [194].

To cope with the increasing demand for content, CDNs have deployed massively distributed server infrastructures to replicate content and make it accessible from different locations in the Internet [221]. These infrastructures have multiple choices on how and where to place their servers. As described by [136], the main approaches are (1) centralized hosting, (2) data center-based CDNs, (3) cache-based CDNs, and (4) Peer-to-Peer (P2P) networks. Approaches 2 and 3 allow scaling content delivery by distributing the content onto a dedicated infrastructure. This infrastructure can be composed of a few large data centers, a large number of caches, or any combination thereof.

To complicate matters further, some of these infrastructures are entangled with the very infrastructures that provide network connectivity to end-users. For example, one of the largest players in content delivery, Akamai, operates more than 120,000 servers in more than 2,000 locations across nearly 1,150 ISP networks [170, 18]. Google is reported to operate tens of data centers and front-end server clusters worldwide [126, 217, 94]. Microsoft has deployed its CDN infrastructure in 24 locations around the world [157]. Amazon maintains at least 5 large data centers and caches in at least 21 locations around the world [22]. Limelight operates thousands of servers in more than 22 delivery centers and connects directly to 600 networks worldwide [144]. Last but not least, P2P networks rely on a huge number of end-users to store, replicate, and distribute content.

Despite the significant entanglement between the infrastructures that deliver content and the network connectivity fabric, our knowledge of their interactions is largely through the literature on network interconnections, see the recent book by W. Norton [169]. Given the nature of network interconnections, previous work has studied the interactions from an economical perspective [154, 25, 135]. The limited knowledge available about the settlements between networks have led researchers to try to reason about why peering choices are made [43] and what drives the evolution of the Internet [61].

Most of the literature has considered the interactions between content and the network indirectly, i.e., through peerings and traffic measurements. This is the case although recent changes in Internet traffic [87, 128] have shown the importance of content and applications. The observed changes in the traffic, either through direct traffic measurements [74, 75, 222, 128, 9], or through inference [155, 239, 238, 99, 177, 208] have repeatedly shown how volatile traffic can be. With the rise of user-generated content and large shifts of content popularity traffic volatility has become especially relevant.

Handling changes in traffic has been traditionally done through traffic engineering (TE). Initially, traffic engineering was considered as a solution to allow large network operators to optimize the utilization of their network [30]. The vast majority of the traffic engineering literature has therefore focused on traffic engineering inside a single network [72, 80, 232, 29, 120, 81]. In reality, most of the traffic in the Internet is exchanged between different networks [128], and especially directly between data centers and residential ISPs [9]. Organizations that originate a lot of content, e.g., Google, connect directly to a large number of other networks [128], and need to optimize how content leaves their own network. Organizations that provide Internet access to broadband or mobile users typically wish to optimize how the Internet traffic enters their network, as most users still download more content than they upload. In between, the transit ISPs try to balance the load of the traffic exchanged between the networks they inter-connect.

Traditional traffic engineering aims at reducing the likelihood that bottlenecks arise inside a given network, due to a mismatch between the network provisioning and the expected demand. Changes in network provisioning are slow, taking place over time scales of weeks or months. Popular content on the other hand generates bursts in demand over much smaller time scales, e.g., hour or minutes. Today's Internet requires much more reactive network control techniques than those we have today, and these techniques must consider content. A few steps have been made in this direction. Indeed, collaborative approaches [64, 148, 84] have been proposed to help deal with the traffic generated by content delivery networks. Even in the case P2P, portals have been proposed to allow P2P applications and users communicate with ISPs and get an updated view of their networks [233]. In broad terms, all information content providers are missing today to optimize their operation is available to ISPs. Combined with the already proposed schemes for collaboration, it is surprising how little real collaboration is performed in todays Internet.

In this chapter, we are analyzing the operation of content providers as well as network operators. Based on these insights, we show the potential of collaboration. Also, we argue that it is important for every party, i.e., the content producers, the network operators and the end-users, in the content delivery landscape to benefit from the collaboration. Furthermore, we present two systems in depth that have incentives for every party and that can readily be used today.

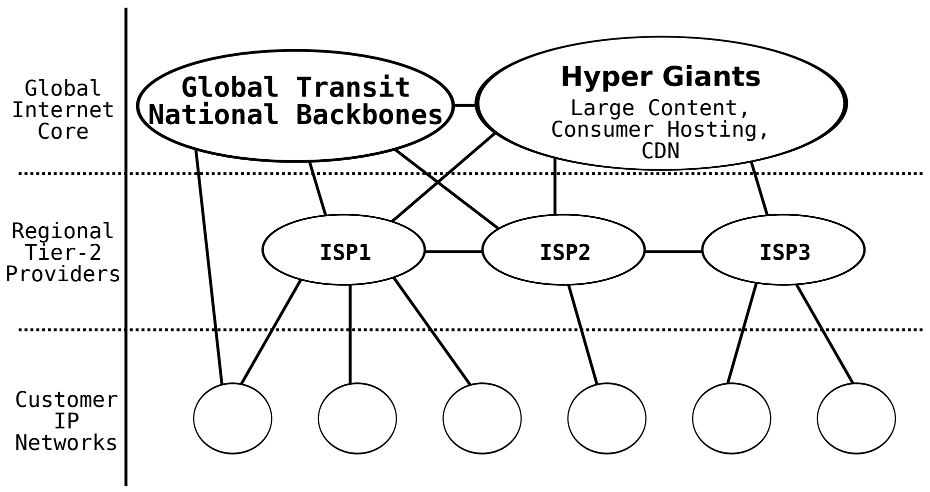

Global Internet Core
Regional Tier-2 Providers
Customer IP Networks
Global Transit National Backbones
Hyper Giants
Large Content, Consumer Hosting, CDN
ISP1
ISP2
ISP3

Figure 1: Layout of the Internet Structure [128]

3 Internet Network Infrastructure

The Internet Network Infrastructure is provided by a set of Internet Service Providers (ISPs). An ISP is, in general terms, an organization that provides access to the Internet for its customers. The Internet is structured by the interconnection of multiple individual networks run by ISPs. However, control of an individual network remains solely with the ISP operating it. Figure 1 shows how the Internet is structured. Here, the ISPs run their own networks. This forces a clear distinction between the individual network that an ISP runs and the global Internet as a network of networks. Also, from this, it can be deduced that nobody has control over the Internet, but instead each ISP has only control over its own network and the direct connections to other networks.

The customers of ISP can be e.g., end-users, hosting facilities, or even other networks. End-users can be connected via a wide range of access technologies such as dial-up-modems, digital subscriber line (DSL), fiber to the home (FTTH) or wireless technologies such as 3G, WiMax or satellite links. If the ISP offers Internet access to end-users via one ore more of such technologies, it is also called an “access ISP”. If other networks use the ISP to reach other networks, the ISP is called a “transit ISP”, as the traffic crosses the ISPs network but neither originates nor terminates in the ISPs network. When the ISP offers other networks connectivity to Internet, that is it allows them to send traffic to the Internet via its own network, the ISP is called an “upstream ISP”. Note that an ISP can have multiple roles at the same time e.g., an upstream ISP is always also a transit ISP.

To be able to interconnect with other networks, an ISP needs to operate an autonomous system (AS). An AS is an administrating entity, generally under the control of one administrative domain, for one or more publicly routable IP prefixes and requires an officially assigned and unique autonomous system number (ASN). Both the

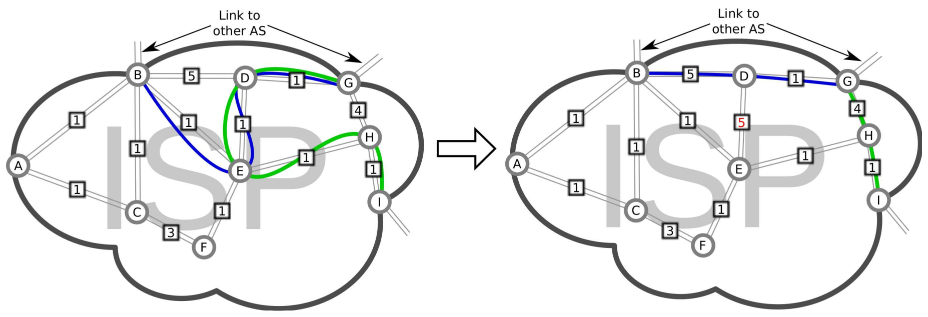


Figure 2: IGP based Traffic Management Example

ASNs and publicly routable IP prefixes are governed by the Internet Assigned Numbers Authority (IANA), which delegates the assignment to the Regional Internet Registires (RIR). Each AS is usually managed by an Interior Gateway Protocol (IGP), e.g., OSPF [161] or ISIS [173]. Since an AS is run centrally by one instance, there is no need for information aggregation or hiding. Thus, each member of an AS can have full topological and operational knowledge of the entire AS. Unless otherwise stated, we assume that one ISP runs exactly one AS.

To interconnect different ASes the Border Gateway Protocol (BGP [190]) is the defacto standard employed and provides the required IP prefix reachability information to make core routing decisions in the Internet. To keep the information transported in the communication scalable throughout the Internet, the entire internal management of the individual AS is abstracted and aggregated. Each AS announces which IP prefixes can be reached via its network and other networks use this information to make routing decision, that is which network path they use to send traffic along towards its destination. For example in the case of an upstream ISP, the ISP would announce the whole public IP space to its customers, while the customers would only announce their own public IP prefixes to the ISP. Furthermore, there are cases when an AS needs to communicate with another AS that it does not have a direct connection to. In this case, the communication has to transit one or more different ASes. Thus, along with with the pure reachability information, the ASN is also transmitted. This allows for loop detection as well as an estimate of how many AS hops away a destination is.

3.1 Traffic Engineering in an AS

The greatest challenge for an ISP is to keep its infrastructure operating efficiently. This is especially hard, since the ISP itself controls neither the behavior, nor the source nor destination of the majority of the traffic it carries. The destination of the traffic is determined by the end-users the ISP sells services to, while the source is usually operated by a Content Delivery Infrastructure (CDI). The behavior is dictated through end-users requesting content, and by the operational choices of the CDI. ISPs today tackle the problem of network operation efficiency by performing Traffic Engineering (TE). In its broadest sense, todays TE encompasses the application of technology and scientific principles to the measurement, characterization, modeling, and control of Internet traffic [30]. Today, traffic engineering reduces to controlling and optimizing the routing function and to steering traffic on an Origin-Destination (OD) flow basis through the network in the most effective way.

Traffic Engineering encompasses multiple steps in order to be performed successfully. First, an ISP needs to record its traffic volume in terms of Origin-Destination flows. This means keeping traffic statistics of how much traffic flows from one router in the network to another. Once the OD flows have been successfully recorded, TE uses this information to simulate the network behavior with different IGP configurations. The goal of these simulations is to find an IGP configuration that spreads the network load as evenly as possible.

Figure 2 shows an example of how an IGP configuration can be used to engineer traffic. The labeled circles represent routers, while the numbers in the squares represent the IGP-weight for the link. For ease of presentation, the weights for each link are set to the same value for both directions. An OD flow, which starts at one router and finishes at another, takes the path through the network that yields the smallest sum over all weights along the path. For example, in the starting configuration of the network (Figure 2 (left)) the flow IG does not take the direct path $I \rightarrow H \rightarrow G$ two, since according to the IGP weights, a more effective path exists. In fact, the path $I \rightarrow H \rightarrow E \rightarrow D \rightarrow G$ has an accumulated weight of 4 instead of 5 (green path). All traffic at router I destined for router G takes this path. Similarly, all traffic that originates from B and goes to G follows the path $B \rightarrow E \rightarrow D \rightarrow G$ (blue path). Also, both paths share links, leading to a possible overload situation. In order to solve this problem, we choose to modify the link weight between the routers D and E. By increasing the weight from 1 to 5 (marked red in the right network), the blue as well as the green paths are shifted to the direct path. The change is shown in Figure 2 (right).

This simple diagram allows for illustrating multiple caveats that IGP based traffic engineering introduces. First, IGP-based traffic engineering affects traffic on an OD-flow basis only. This means that the path from one router to another can be changed, but the traffic on the OD flow cannot be split onto multiple paths. Secondly, the change of one weight can affect multiple OD-flows at the same time. Thus, the weights have to be changed very carefully. In the worst case, it might not be possible to fully separate some OD-flows due to the network layout.

One caveat is not immediately obvious but needs to be taken into account when performing traffic engineering. While the link weights are usually known to all routers, they are propagated by messages that routers exchange. This propagation takes time, which can lead to short-term inconsistencies in the view of a network. We again use Figure 2 for illustrating this. When the link weight is changed as described in the example explained before, routers D and E update their routing. This has an immediate effect on the traffic from B to G. With the update, the shortest path from router E to G is now $E \rightarrow H \rightarrow G$. In accordance, E configures its routing to send all traffic for G through H. However, H has not converged at this point and still uses the old path ($H \rightarrow E \rightarrow D \rightarrow G$). Thus, H still sends all traffic for G towards E. As long as H uses the outdated IGP weight information, all traffic for G that reaches either E or H is sent back and forth between the two routers. This forwarding, on the one hand, likely overloads the link. On the other hand, most traffic that is affected by this will be dropped due to its time-to-live (TTL) running out.

The work of Francois et al. [82] shows that it is possible to gradually change IGP weights by sequentially ordering changes. Accordingly, routing loops like those in the example are avoided. However, these changes still require time during which the network can be in a transient state with overloaded links. Besides the challenges induced by optimizing the IGP, this approach also assumes that traffic is predictable and stable over time. By running simulations based on past traffic aggregates to engineer the routing for the future, it is implicitly assumed that traffic patterns remain similar over a longer period of time.

With the emergence of CDIs, however, traffic has become volatile in terms of its origin. In fact, CDIs can shift massive amounts of traffic in a matter of seconds from one server cluster to another. While this behavior is needed and propagated by CDIs to cope with volatile demand surges, it is in stark contrast to the ISP's traffic engineering, which assumes traffic behavior to be stable for days, weeks or sometimes months.

3.2 Caching Traffic for AS Management

Since traditional link weight based traffic engineering is not able to dynamically handle the volatile traffic CDIs induce, ISPs have also tried to tackle this problem by caching content on proxies. In general, this concept entails that an end-user no longer connects directly to a content server, but is directed to a middle machine, called a proxy, instead. Proxies are scattered throughout the network of an ISP, usually close to end-users. Since hundreds, if not thousands of end-users use the same proxy, the content can be easily cached there. This outlines the idea simply: the proxy is able to store popular content and, once multiple users request it, can serve it directly without burdening the network by connecting to the CDI servers.

However, using proxies comes at a high operational and management cost. This is due to content having become very versatile and highly specific to interest groups. This means that content is not in general popular, but is specific to culture, social group, language, or country. Thus, cacheability of Web content [124, 38, 71, 7, 33] is typically not as good as in other contexts. In addition, with more and more websites being driven by high volume user-generated content, such as YouTube videos, it becomes increasingly difficult for a proxy to cache the right content [13].

By installing proxies, ISPs also become part of the global content delivery infrastructure. But as opposed to CDIs, ISPs are not paid for installing and operating this infrastructure. This leaves ISPs in a situation where they can neither steer the traffic in their network through network configuration nor are they able to reduce the load on the network by caching content at proxies. Furthermore, CDIs do not want ISPs to interfere with their traffic and assignment while the investment and operational burden for ISPs is high and the effectiveness gained by using proxies is limited.

3.3 Domain Name System Basics

The Domain Name System (DNS) plays a major role in todays Internet architecture and is an essential component of any Web based content delivery architecture. DNS relies on a distributed database with a hierarchical structure. The root zone of the DNS

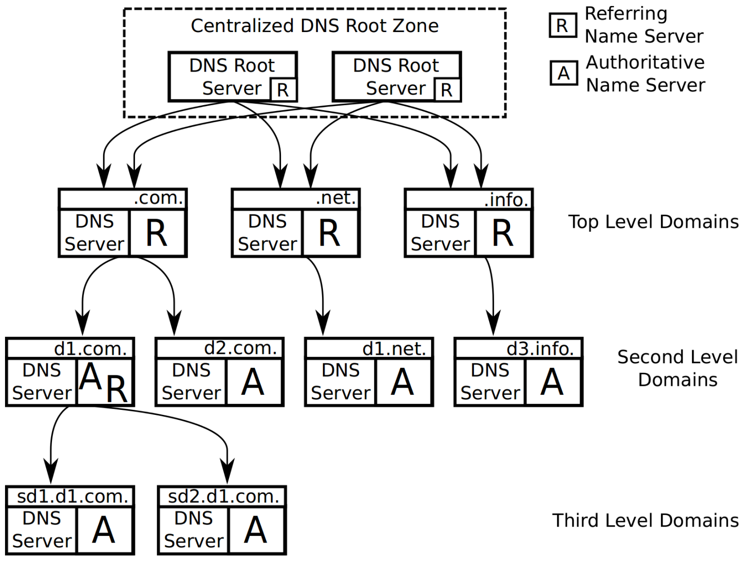


Figure 3: Example DNS hierarchy

system is centrally administered and serves its *zone* information via a collection of *root servers*. The root servers delegate responsibility for specific parts (zones) of the hierarchy to other *name servers*, which may in turn delegate the responsibility to other name servers. At the end, each site is responsible for its own *domain* and maintains its own database containing its information and operates an *authoritative* name server.

The whole DNS database is usually queried by end-hosts using a local name server called *caching resolver*. If this name server receives a query for a domain that it does not know about, it fetches this information from another name server. If the server does not know how to contact the authoritative server for a zone, it will query a root server[1]. The root server will *refer* the resolver to another server that is authoritative for the domain that is immediately below the root and of which the zone is a part. The resolver will then query this server, and so forth, stepping down the tree from the root to the desired zone.

To illustrate this process, Figure 3 show a sample DNS hierarchy. In this case, the root of the DNS name space, denoted with a '.', is hosted on two *DNS root servers*. Both servers are under one administrative control, and both can refer a request to any of the top level domain servers. Here, three domains exist, i.e., *.com*, *.net* and *.info*. Again, these name servers refer to the second level domains. Since the domain name are concatenated together as the hierarchy is traversed, the domains that are now possible are *d1.com.*, *d2.com.*, *d1.net.* and *d3.info.*. At this point, the second level domains d1.net. and d3.info have reached their authoritative resolver. For example, a query to the name server of *.d3* for *www.d3.info* is answered authoritatively from there. Note

[1]The first query can go to some authoritative server below the root if there exists cached information.

that the name servers for the second level domains are operated by independent entities that know nothing of each other. Thus, the database is distributed, while each party is responsible for its own zone. Finally, the name server of *.d1.com.* has a dual role. While it is referring the subdomains *.sd1.d1.com.* and *.sd2.d1.com.* to other name servers, it also answers queries for other names in its name space authoritatively. This means that a query for *www.d1.com.* is directly answered, while a query for *www.sd1.d1.com* is referred to the name server responsible for *.sd1.d1.com*.

For efficiency reasons DNS relies heavily on caching [112, 10]. All information that a name server delivers to a resolver is cached for a duration specified in the time-to-live (TTL) field of the *resource records* (RR). Caching today is usually also performed on end-hosts by the operating system's *stub resolver*, as well as applications, e.g., web browsers.

DNS Today When DNS was introduced in 1983, its sole purpose was to resolve host names into IP addresses in a more scalable fashion than the until then used `hosts` file. Since then a number of features and new uses have found their way into the now omnipresent DNS. In addition to the increasing complexity within the DNS protocol itself [225], new and oftentimes unforeseen (ab)uses have been established. Paul Vixie gives an overview in [226]. The most important points of critique are as follows:

CDN load balancing: Content delivery networks set short TTLs on their DNS answers to allow for short reaction times to shifting loads. Short TTLs impede on cacheability and therefore increase the load on the whole DNS system. In addition, CDNs tailor their reply for the IP address of the requesting resolver using the assumption that the DNS resolver is close to the client originating the request. It has been shown in the past that this assumption is quite often wrong [151, 176, 10, 55].

NXDOMAIN catcher: Some ISPs and third party DNS providers mangle a negative reply with the NXDOMAIN status code into a positive one with the IP address of a search website under the control of the ISP. By hosting advertisements along the search results it is easily possible to increase the profit margin. While this may work to some degree for web browsing, applications relying on proper delivery of NXDOMAIN records, e.g., email, are inevitably hampered.

A third-party ecosystem around DNS has evolved over the last couple of years. Players like OpenDNS, AdvantageDNS, UltraDNS, and most recently Google offer open resolvers to anyone with different feature sets. OpenDNS Basic does NXDOMAIN catching but offers phishing and botnet protection for free. Furthermore, OpenDNS increases the service level for payment between 5 dollars a month up to several thousand dollars per year for business customers. When Google Public DNS entered the market, their highest-valued goals were to "speed up your browsing experience" and to "improve your security". To achieve both targets Google advertises an impressive list of optimizations and fine tuning [96], e.g., prefetching, load balancing with shared cache, validity checking, and nonce prepending. Google Public DNS also refrains from utilizing NXDOMAIN to make profit. From an implementation perspective, most if not all of the third-party resolvers host their DNS servers on multiple sites around the globe and use anycast to guide DNS clients to the nearest resolver.

In this open market space a user annoyed by his ISP's DNS can easily choose for cost-free third-party service. Tools such as namebench [164] might help him in choosing a well-performing one. The irony however is that a user, by choosing a different DNS than the one assigned by his ISP, will most likely undermine the traffic matrix optimizations performed by CDNs and ISPs, and can potentially even lower his quality of experience due to longer download times [10].

4 Traffic Trends: Overall

Before delving into the details of the collaborating opportunities for content providers and infrastructures we embark on giving an overview of typical characteristics of Internet traffic. In Section 4.1 we introduce the mix of applications and content-types as well as traffic patterns. Then, we briefly outline tools commonly used for traffic analysis in Section 4.2.

4.1 Internet Traffic Characteristics

We start by summarizing previous studies on how Internet traffic looks like. We consider four aspects: *(i)* The composition of the application mix, *(ii)* popular content-types, *(iii)* the distribution of traffic over the course of a day, and *(iv)* the distribution of connection sizes.

4.1.1 Application Mix

One constant in the Internet during the last 10 years has been its steady growth by more than 50 % each year [172, 89]. Initially, protocols such as FTP, SMTP, and NNTP were popular. Then, in about 1994, HTTP entered into the picture. Until 2000, P2P protocols such as Napster and Gnutella became popular but were later overtaken by eDonkey and BitTorrent. However, the traffic mix has undergone substantial changes. Therefore, we now revisit previously reported results regarding the application mix of Internet traffic. For this purpose we rely on various studies that report on the application mix between 2007 and 2009 from different vantage points:

- The study by Maier et al. [150], which is based on a subset of the traces studied in Section 12.6. It was presented at IMC '09.
- Two studies by ipoque [201, 200], which report on different regions in the world (Germany and Middle East). These studies are available for download after registration via a Web form.
- The Arbor report [128] on the ATLAS Internet Observatory presented at a recent NANOG[2] meeting.
- The Sandvine report on "Global Broadband Phenomena" [194].

In order to compare the results we have to summarize and unify the traffic categories as each study uses their own nomenclature (see Figure 4). For this purpose we use the following seven categories:

Web All HTTP traffic including One-Click-Hosters (OCHs or Direct Download Providers) but excluding video and audio streaming over HTTP (i.e., Flash-Video).

Streaming All types of streaming in the Internet including streaming over HTTP, RTP, RTSP, RTMP, ShoutCast, etc.

[2]NANOG is the North American Network Operators Group.

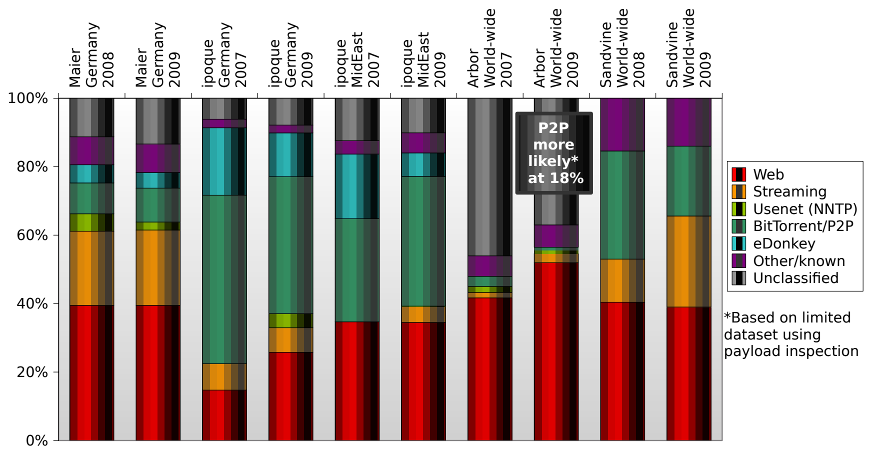


Figure 4: Barplot of the application mix in the Internet (unified categories) for different years, different regions according to several sources [150, 201, 200, 128, 194]. (BitTorrent/P2P contains all P2P except eDonkey.)

Usenet The article reading and posting system that evolved from UUnet and which uses NNTP as protocol.

BitTorrent/P2P The popular P2P-protocol BitTorrent and all other P2P traffic that is not eDonkey. Note, that the P2P traffic that is not BitTorrent or eDonkey only adds a tiny fraction. Moreover, this category represents all P2P traffic if the study no further subdivides P2P traffic. This is the case for Arbor [128] and Sandvine [194]. Note as well, that the Arbor study [128] reports a table with traffic shares, stating 0.95 % for P2P. This table is annotated with the comment that P2P is more likely to account for 18 % based on payload inspection of a limited data subset.

eDonkey Another P2P protocol, if reported.

Other/known Other identified traffic, for details we refer to the corresponding studies.

Unclassified Traffic that has not been classified. Note, that the Sandvine [194] study does not mention unclassified traffic, which either implies a minute fraction or that it is missing in the plot.

Looking at these statistics we find that all studies report a significant fraction of Web traffic. Indeed, Web is dominant ($> 50\,\%$) in most studies, followed by P2P and streaming. It is noteworthy that Usenet is responsible for a non-negligible fraction in several studies. This is surprising and a good example for the importance of revisiting the application mix periodically in order to identify new trends.

In terms of P2P protocol distribution Figure 4 shows that BitTorrent is dominating and the shares of eDonkey are decreasing. Thus, we note that the results of Plissonneau et al. [182] who observed 91 % of the P2P traffic is due to eDonkey in 2004 are no

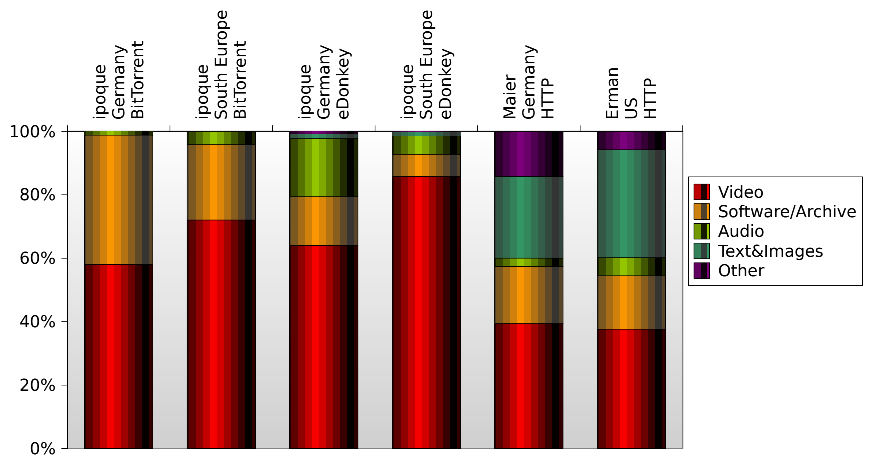


Figure 5: Barplot of content-type popularity in the Internet (unified categories) for different protocols, different regions according to several sources [150, 200, 68].

longer applicable. Indeed, the popularity among P2P protocols swapped in favor of BitTorrent. We can also see a general trend: P2P is declining according to all studies. This is also supported by the results of Anderson [24]. He points out that this decline comes with an increase in video streaming. Moreover, most of the studies pointed out that currently One-Click-Hoster (e.g., Rapidshare or MegaUpload) are as important for file-sharing as P2P systems.

Of course there are also trends that do not impact the application mix, for example Online Social Networks (OSNs) such as Facebook. This is due to the fact that OSNs use HTTP and they do not transport large videos, but profile elements. Nevertheless, OSNs are not unimportant given the huge number of OSN users world-wide.

4.1.2 Content-types in the Internet

Next, we turn to the popularity of content-types in the Internet. Again, we leverage several data sources, namely Maier et al. [150], ipoque [200], and Erman et al. [68]. Once more we unify the categories and present results for contents transferred via BitTorrent, eDonkey, and HTTP. See Figure 5 for a summary.

We see that videos are the most popular content in P2P systems (BitTorrent and eDonkey). Even in HTTP videos account for more traffic than any other category. Although HTTP was designed to transfer Web pages (text, e.g., HTML, XML, CSS, JavaScript, and image files) these contribute less than a third of the total HTTP volume.

Overall, a significant fraction of software and archives is noticeable. According to Maier et al. [150] almost all videos are in flash-video format and are served by video portals such as YouTube. Similarly, almost all archives are served by One-Click-Hosters. This is confirmed by the results of Erman et al. [68].

Shifts in the popularity of content-types can be another indicator of new trends. For example, there have been almost no flash-videos before the breakthrough of YouTube.

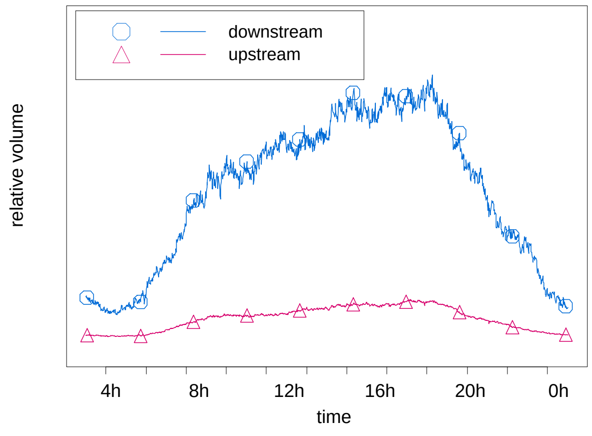


Figure 6: Timeseries of link utilization from Maier et al. [150]

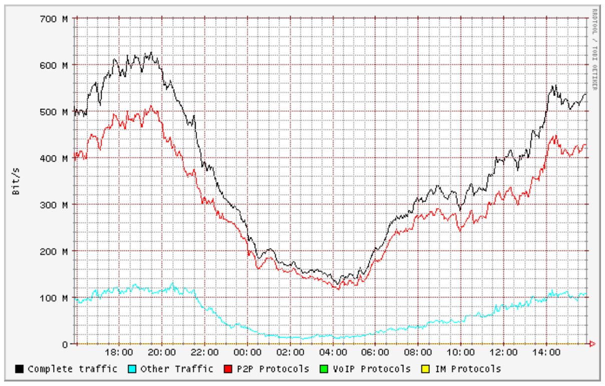


Figure 7: Timeseries of traffic volume from ipoque [201]

4.1.3 Time-of-day Effects

In order to understand when people are active in the Internet we show time-of-day usage plots of link utilization from Maier et al. [150] in Figure 6, and aggregated traffic volume from ipoque [201] and Sandvine [194] in Figure 7 and Figure 8, respectively.

In general, we observe a peak utilization at prime-time around 8 pm and a daily low between 2 am and 4 am. As the data sets of all these studies are primarily collected from residential networks, it not surprising that they all show similar characteristics. The peak usage in the evening hours can easily be explained by the fact that people are usually not at home during business hours. Raising demands just before lunch and in the afternoon may be due to children returning home from school.

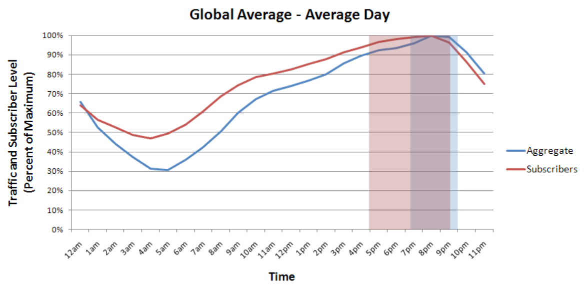


Figure 8: Timeseries of traffic volume from Sandvine [194]

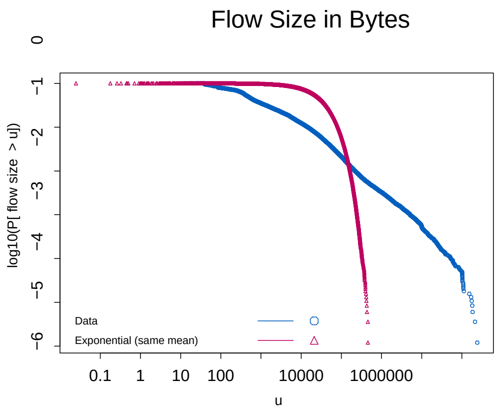


Figure 9: CCDF of flow sizes: Data and exponential distribution with same mean

4.1.4 Flow Sizes and Durations

Finally, we focus on flow characteristics. A flow in this context is the set of packets with the same 5-tuple of source and destination IP, source and destination port, and transport protocol.

A reoccurring observation is that the flow size distribution is consistent with a heavy-tailed distribution [137, 180, 57, 228]. Recent work by Basher et al. [34] compared flow-level characteristics of Web and P2P traffic. They found a mean (median) flow size of 21.5 kB (2.53 kB) for Web and 362.4 kB (1.17 kB) for P2P. This is also reflected in their CCDF [34, Figure 2(a)], where more heavier flows are shown for P2P. Calculating the flow size distribution of all traffic for a data set collected in February 2008 at a major Internet Service Provider we find a mean flow size of 38 kB and a median of 478 Bytes. Figure 9 shows the CCDF of these distributions and an exponential distribution with the same mean.

Basher et al. [34, Table 5] further find that flow durations have a mean (median) of 13 seconds (400 ms) for Web and 123 seconds (24 seconds) for P2P. Again we

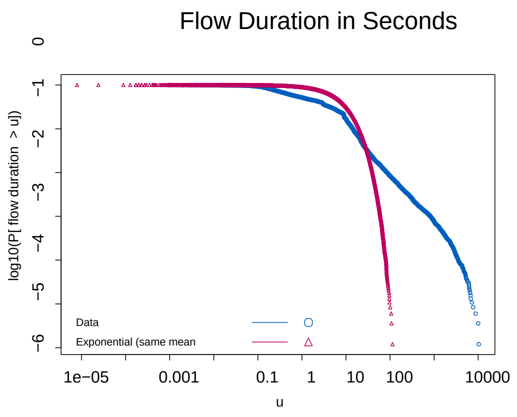


Figure 10: CCDF of flow durations: Data and exponential distribution with same mean

calculate statistics on the data used earlier on for the flow sizes (in February 2008), see Figure 10: Mean flow duration is 9 seconds and median is 1.1 seconds. Both results are in the same order of magnitude. The differences are likely to be caused by the different composition of the traffic that is covered: Web and P2P on the one hand and complete traffic on the other.

4.2 Traffic Analysis Tools

Now, we briefly introduce the traffic analysis tools that we use. All tools are Open-Source and freely available.

4.2.1 LIBPCAP and TCPDUMP

LIBPCAP [106] is a C library for capturing packets, that are transmitted over a link that is attached to the computer running the pcap application. The functions included in LIBPCAP provide a standardized interface for all common (UNIX-based) operating systems, including Linux and FreeBSD. The interface of LIBPCAP can also be used under Windows but the Windows library is called WINPCAP. Many network analysis tools are build on top of LIBPCAP to allow for use on any LIBPCAP supported architecture and operating system.

LIBPCAP defines a packet trace file format and includes functions for reading from such trace files as well as for dumping packets to disk. Therefore LIBPCAP enables both live capture from a network interface and offline analysis from a saved trace file. Furthermore, the pcap trace file format is the de-facto standard for exchanging packet level traces.

One of the most popular tools for network monitoring and network debugging is TCPDUMP (www.tcpdump.org). The command-line tool TCPDUMP uses LIBPCAP for data capture. TCPDUMP itself includes various analyzers for parsing network protocols and producing human readable output summaries per packet. It can also be used to collect traces in pcap format.

4.2.2 WIRESHARK

WIRESHARK [229] is a graphical interface for LIBPCAP with similar capabilities as TCPDUMP. Due to WIRESHARK's information sorting and filtering options it is a very intuitive tool for network debugging. It can read and write pcap format as well as capture data directly from the network. WIRESHARK started as "Ethereal" and was renamed later due to trademark issues.

WIRESHARK's analysis capabilities exceed those of TCPDUMP. For example WIRESHARK can reassemble TCP connections from packet traces and parse the application layer protocols. WIRESHARK also ships some statistical traffic analysis tools, such as conversation lists, response-time analyses, or HTTP request summaries to name a few.

To take advantage of WIRESHARK's extended analysis capabilities via a command-line interface the distribution also includes TSHARK. Moreover, it offers a programmable interface for automated trace editing, converting, or analysis.

4.2.3 BRO

BRO [178] is a network intrusion detection system (NIDS) developed by Vern Paxson [179]. Before attacks can be detected by customizable policy scripts BRO first needs to parse and generate events from the observed network traffic. In order to do so it features a robust TCP reassembly engine and protocol analyzers. These analyzers not only match signatures on the traffic, but try to understand and reconstruct the semantics of the application layer protocol. BRO is organized in several layers (see Figure 11, top to bottom):

- The *policy script interpreter* loads the configuration via policy script files. After parsing the files it instructs the event engine which events it is interested in and which analyzers to load. Once events arrive the interpreter executes the analysis as specified in the policy scripts.

- Depending on the events that the policy layer subscribes to and depending on the analyzers that are loaded the *event engine* compiles the capturing filter and starts to capture packets via LIBPCAP. It parses a packet, determines the connection it belongs to, and hands it to the appropriate analyzer tree for this connection. In case of TCP the first packet instantiates a TCP-reassembly for this connection. Then depending on the TCP payload the correct protocol analyzer is started. The analyzer generates application layer events as they are detected and notifies the policy layer.

- LIBPCAP handles all interactions with the operating system and is used as a uniform interface for packet capturing. Via LIBPCAP BRO can read the network data either live from the network or from a pcap trace file.

The analyzer tree is dynamic. It can have multiple different protocol analyzers examine the same connection. If an analyzer determines that the current data stream

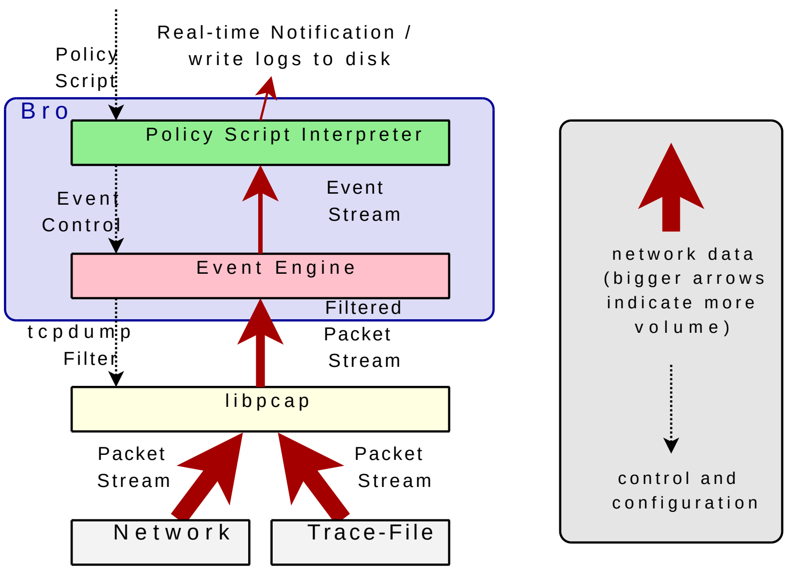


Figure 11: Structural design of BRO

does not comply with the semantics of the protocol that it expects it can detach itself from the tree with a notification. This dynamic protocol detection [66] allows the detection of traffic of certain protocols even if they do not run on default ports, do not match a signature, or are tunneled within another protocol.

Although BRO was designed as an NIDS its protocol analysis capabilities make it a perfect tool for traffic classification and application layer analysis. It ships a whole set of protocol analyzers including an HTTP analyzer that is capable of extracting HTTP requests, responses, and if required also HTTP headers. For a complete list of available analyzers please refer to www.bro-ids.org.

Name	Type	Start date	Dur.	Size	Application Volume
MAR10	packet	04 Mar'10 2am	24 h	>5 TB	> 3 TB HTTP, > 5 GB DNS
HTTP-14d	log file	09 Sep'09 3am	14 d	> 200 GB	corresponds to > 40 TB HTTP
DNS-5d	packet	24 Feb'10 4pm	5 d	>25 GB	> 25 GB DNS

Table 1: Summaries of anonymized traces from a European ISP

5 Traffic Trends: Content Server Diversity

So far we have highlighted that the Web and P2P protocols are responsible for a major share of the Internet traffic. However, we have not yet explored the if all content is equally popular or if a few content providers dominate. This is the goal of this section.

Our evaluation methodology relies on packet level traces from a large European ISP. We analyze them towards identifying CDI infrastructures and their behavior as seen by an ISP. Here, we find that CDIs rely on the domain Name System (DNS) for their operation. Thus, we focus our analysis on the DNS infrastructure in order to find the server deployment, mapping and operational behavior of CDIs. Based on these observations, we develop classification methods to detect CDI infrastructures and perform a first potential analysis on the impact of CDI operation when basic ISP knowledge is available.

5.1 Residential ISP Traces

We base our study on three sets of anonymized packet-level observations of residential DSL connections collected at aggregation points within a large European ISP. Our monitor, using Endace monitoring cards, allows us to observe the traffic of more than 20,000 DSL lines to the Internet. The data anonymization, classification, as well as application protocol specific header extraction and anonymization is performed immediately on the secured measurement infrastructure using the Bro NIDS [179] with dynamic protocol detection (DPD) [66].

We use an anonymized 24 h packet trace collected in March 2010 (MAR10) for detailed analysis of the protocol behavior. For studying longer term trends, we used Bro's online analysis capabilities to collect an anonymized protocol specific trace summary (HTTP-14d) spanning 2 weeks. Additionally, we collected an anonymized 5 day DNS trace (DNS-5d) in February 2010 to achieve a better understanding of how hostnames are resolved by different sites. Due to the amount of traffic at our vantage point and the resource intensive analysis, we gathered the online trace summaries one at a time. 1 summarizes the characteristics of the traces, including their start, duration, size, and protocol volume. It is not possible to determine the exact application mix for the protocol specific traces, as we only focus on the specific protocol. However, we use full traces to cross check the general application mix evolution.

With regards to the application mix, recall Section 4, Maier et al. [150] find that HTTP, BitTorrent, and eDonkey each contribute a significant amount of traffic, see Table 1. In MAR10 HTTP alone contributes almost 60 % of the overall traffic at our vantage point, BitTorrent and eDonkey contribute more than 10 %. Recall that similar protocol distributions have been observed at different times and at other locations of

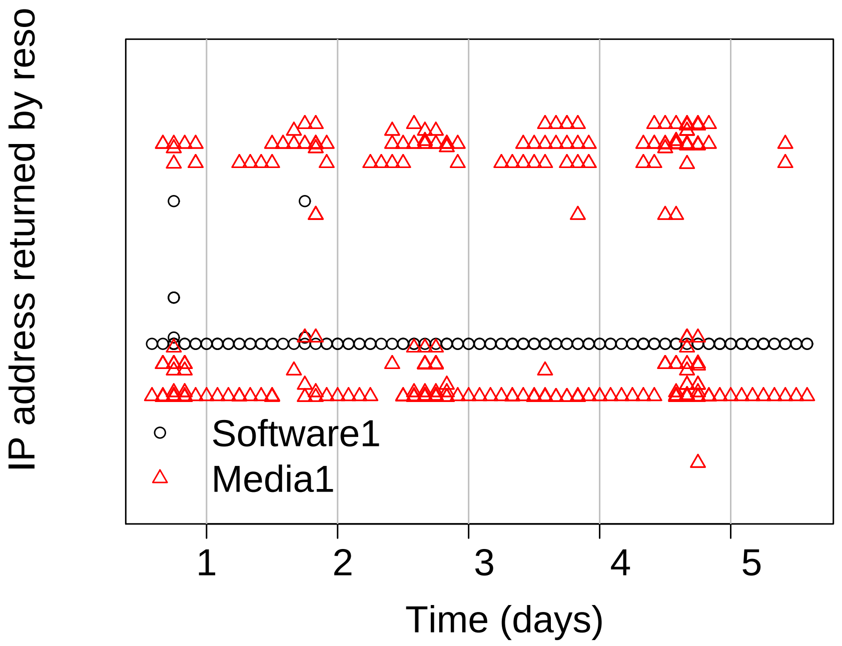


Figure 12: DNS replies for two different sites hosted on a CDI, in two-hour bins

the same ISP, see Figure 4 summarizes the results. Note that almost all streaming is done via the Web on top of HTTP. Therefore, we conclude that currently HTTP is the dominant service and P2P is still responsible for at least 15% of the traffic.

Analyzing HTTP-14d, we find more than 1.2 billion HTTP requests, or 89 million requests per day on average. This is consistent with 95 million requests in 24 hours in MAR10. The advantage of using click stream data from a large set of residential users is their completeness. We are, e.g., not biased by the content offered *(i)* by a Web service, *(ii)* whether sufficient users installed measurement tools such as the alexa.com toolbar, or *(iii)* whether users actually use some kind of Web proxy.

To identify the most popular Web services, we focus on the most popular hosts. As expected, the distribution of host popularity by volume as well as by number of requests is highly skewed and is consistent with a Zipf-like distribution as observed in other studies [150]. The top 10,000 hosts by volume and the top 10,000 hosts by number of requests together result in roughly 17,500 hosts. This indicates that on the one hand, some hosts that are popular by volume may not be popular by number of requests and vice versa. On the other hand, there are some hosts that are popular according to both metrics. The total activity by these hosts accounts for 88.5 % of the overall HTTP volume and more than 84 % of the HTTP requests. Assuming that the HTTP traffic volume accounts for roughly 60 % of the total traffic, similar to the observations made in September 2009 [150, 13] and in MAR10, more than 50 % of the trace's total traffic is captured by these hosts.

5.2 Server Diversity and DNS Load Balancing

To better understand how HTTP requests are handled and assigned to servers, we use DNS-5d to analyze the 20 most heavily queried DNS names to identify typical usage patterns. We consider only the most heavily used resolver. Figure 12 shows two of the typical patterns for two of the DNS names. It also shows how the resolved IP addresses change (y-axis) across time (x-axis) for two hostnames; respectively a software site, labeled Software1, and a media site, labeled Media1. The vertical lines annotate midnight. If two IP addresses are plotted close to each other, this indicates that the longest common prefix of the two addresses is close. We note that the hostname of Software1 is mainly resolved to a single subnet, excepting a few special cases. However, Media1 is load balanced across approximately 16 different sites. For Media1, there appears to be one main site which is almost always available, while the remaining 15 are predominantly used during afternoon and evening peak usage hours.

These results are promising, and show that individual sites do expose a certain degree of server diversity to their users. While our trace (HTTP-14d) includes the queried hostnames, it does not include the resolved IP address, as a HTTP request header contains the hostname but not the IP address of a server. To verify the above behavior and get an up-to-date view of the DNS replies for the hostnames of our trace, we used 3 hosts within the ISP to issue DNS queries to the ISP's DNS resolver for all 17,500 hostnames repeatedly over a fourteen day measurement period starting on Tue Apr 13th 2010. During these two weeks, we received more than 16 million replies. Unless otherwise mentioned, we rely on our active DNS measurements, with augmented statistics concerning volume and requests from HTTP-14d.

5.3 Server Location Diversity

Our analysis of hostnames and their assignment to servers in section 5.2 has shown that content can be served by multiple servers in different locations. In fact, many domains use the service of a *Content Delivery Service* (CDS), which can be seen during the DNS resolution progress: The original domain name is mapped to the domain of a CDS, which then answers requests on behalf of the requested domain name from one of its caches [216]. Almost all CDSs rely on a distributed infrastructure to handle the expected load, load spikes, flash crowds, and special events. Additionally, this introduces needed redundancy and fail over configurations in their services. Among the most studied CDS' are Content Distribution Networks (CDNs), such as Akamai [136, 216, 103], and Content Delivery Platforms (CDPs), such as Google [126] and their YouTube service [41].

The DNS server can choose to return one or more server IP addresses based on the domain name in the request and the IP address of the requesting DNS resolver. For example, it may use a geo-location database [153] to localize the region of the DNS resolver, utilize BGP data to identify the ISP, create a topology map derived via traceroutes, or any combination of these and other topological and geographic localization techniques. A DNS server has, in principle, two methods for load balancing across multiple servers:

MultQuery: Can return multiple IP addresses within a single DNS response

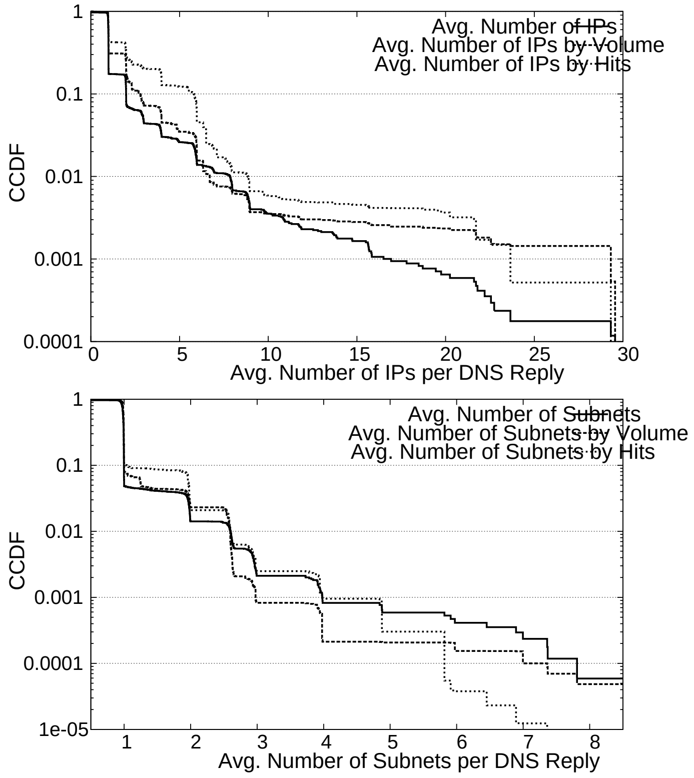


Figure 13: CCDF of mean # of IPs (top) and subnets (bottom) per DNS reply for the ISPs DNS resolver.

CrossQuery: Can return different IP addresses for repeated queries and thus perform DNS redirection.

In our active DNS measurements, we found that often a mixture of MultQuery and CrossQuery is being used in practice. Furthermore, we used the measurement results to *(i)* map hostnames to sets of IP addresses and *(ii)* check the IP address diversity of these sets for a better understanding of server diversity and their location. We achieved this by aggregating the returned IP addresses into subnets based on BGP information obtained from within the ISP. This allows for detailed information about the different locations within the ISP, while giving an aggregated view of subnets reachable via peering links.

Another issue stems from the fact that the IP address returned by the CDS depends on the IP address of the ISP DNS resolver [11, 176, 216]. Due to this, we used the DNS resolver of the ISP of our vantage point as well as external DNS resolvers (see section 5.3.1). The former reflects the experience of most of the clients at our vantage point[3].

[3]We verify using the traces that more than 95 % of the clients use the ISP's DNS resolver as their default one.

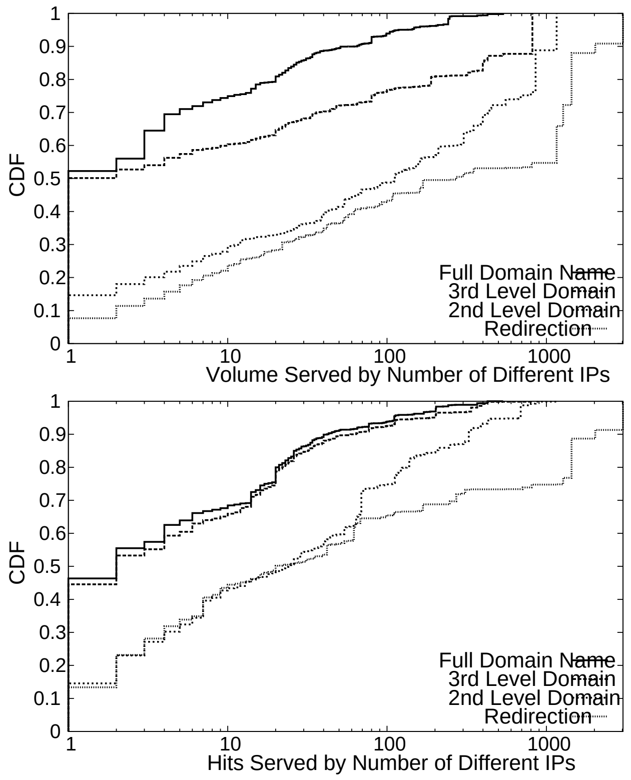
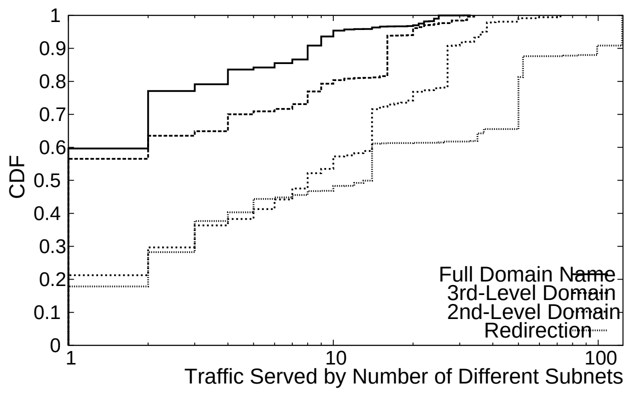


Figure 14: CDF of # of IPs for the ISP DNS resolver normalized by traffic volume (top) and requests (bottom) including aggregation on domain levels. (Logarithmic x-axis.)

The latter lets us discover additional diversity as well as understand the preference of the CDS for this specific ISP.

Prevalence of MultQuery We start our analysis by checking the prevalence of the first form of DNS based load balancing, MultQuery. Figure 13 shows a CCDF plot of the average number of IP addresses (top) and subnets (bottom) per DNS reply. In addition, we included the same data normalized by traffic volume and number of requests.

A first observation is that the number of returned IP addresses per request is rather small. The median is 1, the average is 1.3 and even the 0.9 percentile is 2. We note that even when an answer yields multiple IP addresses, the majority of them are from the same subnet. Therefore, the diversity decreases even further if we aggregate to subnets. From a network perspective, this implies that there is not much choice, neither for the ISP nor for the user, regarding where to download the content from. Both are limited to the information provided by the DNS server. However, when we normalize the hosts by their respective popularity, we see a significant improvement. More than 29% of the volume and 19% of requests have a choice among at least 2 IP addresses.



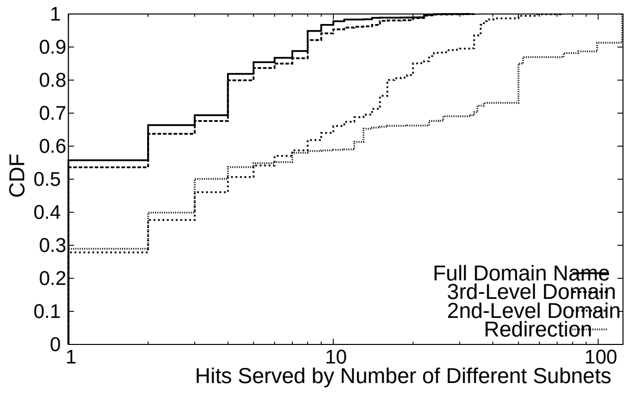


Figure 15: CDF of # of subnets for ISP DNS resolver normalized by traffic volume (top) and by requests (bottom) including aggregation on domain levels. (Logarithmic x-axis.)

Prevalence of CrossQuery Next, we check how prevalent CrossQuery, the second form of DNS based load balancing is. Since CrossQuery returns different IP addresses for repeated queries, its potential contribution to server diversity can only be studied by aggregating across time. The lines labeled `Full Domain Name` in Figures 14 and 15 capture this case.

We find that more than $50\,\%$ of the volume or requests can be served by more than one IP address. similarly, there is choice between at least two subnets over $40\,\%$ of the time across both metrics, see Figure 15. This indicates that there is significant potential for the ISP to bias the location preference of the CDS.

Subdomain Aggregation Since some CDSs only use subdomains as hints about the context of the requested URLs or the requested services, we accumulate the answers further regarding the 2nd and 3rd part of the domain names of the hosts, see Figures 14 and 15 at the respective data series called `3rd Level Domain` and `2nd Level Domain`. For example, we might accumulate the IP addresses from DNS replies for dl1.example.org and dl2.example.org for the statistics on the 2nd level domain, but not

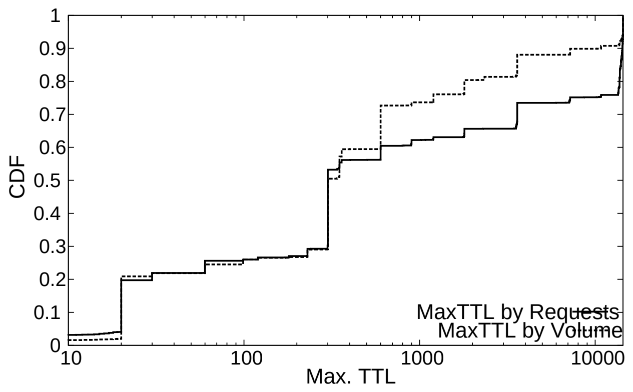


Figure 16: CDF of DNS TTL value by traffic volume and by number of requests.

the third level domain.

This is a feasible approach, since many hosts respond to all requests that belong to a subset of the subnets returned when accumulating by the second-level domain of DNS resolver answer, including recursive requests and redirections. We verify this behavior with active measurements, see Section 13.5. We find that at least two major CDNs, a streaming provider and a One-Click Hoster, serve requested content from servers that match in their second level domain.

We note that the accumulation by third-level domain, and especially by second level domain significantly increases the number of observed subnets per request both normalized by requests as well as by volume. The number of returned subnets further increases when accumulating to the second-level domain of DNS resolver answer. Studying our traces in more detail, we find that this is due to the substantial traffic volume and number of requests that are served by CDNs, some of which are highly distributed within ISPs or located in multihomed datacenters or peer-exchange points.

Infrastructure Redirection Aggregation Taking a closer look at the DNS replies [159], we find that some CDSs use CNAME records to map queried hostname to an A record. These A records show the same pattern as the hostnames in the previous section: the second level domain is identical. Similar to the previous approach, we can aggregated by these A records.

Turning our attention to the implications of the proposed aggregation schemes, we notice the available diversity increases tremendously. More than 50% of the hits and 70% of the bytes can be served by more than 20 servers. With regards to subnets, the diversity decreases slightly. Nevertheless, more than 5 subnets are available for 45 % of the hits and 55% of the bytes.

If we consider aggregation periods in the order of tens of minutes, the numbers do not decrease by much. The reason that most of the diversity is observable even over these short aggregation time periods, is that the typical TTL, see Figure 16, is rather short with a mean of $2,100$ seconds and an median of 300 seconds normalized by volume. When weighted by requests, the mean/median is $4,100$/300 seconds.

	ISP DNS		OpenDNS		GoogleDNS	
Metric	observed	potential	observed	potential	observed	potential
IPs	12.3 %	24.2 %	5.8 %	16.0 %	6.0 %	9.7 %
requests	14.9 %	33.2 %	4.7 %	18.8 %	4.8 %	6.4 %
volume	23.4 %	50.0 %	12.0 %	27.7 %	12.3 %	13.4 %

Table 2: Traffic localization within the network by different DNS resolvers normalized by number of requests and traffic volume together with the potentially available fraction of localized traffic.

5.3.1 Alternative DNS Resolvers

So far we have only considered the effect of content diversity when the ISP DNS resolver is used. To understand how much the DNS load balancing deployed by a CDS is biased by the queried DNS resolver, we repeat the experiment from Section 5.2 using two other DNS resolvers. In particular, we pick the next most popular DNS resolvers found in our traces: GoogleDNS and OpenDNS. Both are third-party resolvers with a global footprint and utilize DNS anycast.

Comparing the results, we find that we attain more IP address diversity and subnet diversity when using the ISP DNS resolver. This is mainly due to the fact that CDSs select the supplied caches based on the source IP address of the querying DNS resolver. Since the CDSs are no longer able to map the request to the AS it originates from, but rather to AS the DNS resolver belongs to, the server selection by the CDS cannot optimize for the location of the DNS client.

5.4 Impact on Traffic Localization

Analyzing the three active DNS measurements from the ISP, OpenDNS as well as Google DNS resolver, we find that a significant part of the requests that could have been in principle served by sources within the ISP are directed towards servers that are outside of the ISP. However, before tackling this issue, we need to understand what fraction of the traffic may be served by IP addresses within the ISP's network and what fraction is served by IP addresses outside of the AS. To this end, we analyze each of the three active DNS traces separately. For each trace, we start by classifying all DNS replies regarding the `redirection` aggregation described in Section 5.3 and account the volume (or hits) evenly to each of the IP addresses. Next, we classify the IP addresses in two groups - inside and outside of the ISP network. Table 2 summarizes the results of this aggregation regarding the traffic and hits that were kept inside the ISP's network in the columns labeled `observed`.

Turning to the results, we find that there is hardly any difference between those clients that use the external DNS resolvers, i.e., GoogleDNS or OpenDNS. Of the returned IP addresses, less than 6 % are within the AS. When weighted by number of requests, this does not change much. However, when normalizing by volume, about 12 % of the traffic stays within the AS. In contrast, clients that use the ISP's DNS resolver fare better: almost a quarter of the traffic volume is served from servers within the AS. Normalized by requests, we see a three fold increase, and normalized by hits or volume, roughly a two fold increase over using external DNS resolvers. Among the reasons for the "bad" performance of external DNS resolvers is that some CDIs may always return IP addresses outside the ISP, despite the fact that many of its servers are deployed within the ISP. The reason behind this is that the CDIs cannot map the DNS resolver to the AS anymore, and thus are unaware of the origin of the request. This explains the substantial difference and highlights on the one hand the effectiveness of the CDI optimization, but also points out its limits. As such, it is not surprising that there are efforts under way within the IETF to include the source IP addresses of the DNS client in the DNS requests [55].

However, one can ask if the CDI utilizes the full potential of traffic localization on an AS level. For this, we check the potential of traffic localization, by changing the volume (or hit) distribution from even to greedy. Thus, as soon as we observe at least one IP address inside the ISP's network, we count all traffic for the entire aggregation to be internal. Table 2 shows the results in the columns labeled `potential` for all three DNS traces. Note the substantial differences. Our results indicate that a gain of more than a factor of two can be achieved. Furthermore, up to 50 % of the traffic can be delivered from servers within the ISP rather than only 23.4 %. This may not only in itself result in a substantial reduction of costs for the ISP, but it also points out the potential of collaboration between CDIs and ISPs. While the increase is noticeable it is nowhere near that of the ISP's DNS resolver. The potential benefit when relying on GoogleDNS is rather small. A deeper study on our results unveils that content served by highly distributed and redundant infrastructures can be localized the most.

5.5 Summary

We find that HTTP is again the dominant traffic source, while the prevalence of P2P traffic decreases. Since most CDSs rely on distributed infrastructure, we not only observe significant server location diversity but also significant path diversity for accessing HTTP based content. Indeed, there is the potential to bias roughly half of the overall traffic by redirecting queries to different content servers.

More precisely, we estimate that around 70 % of the HTTP traffic in a big European ISP can be redirected when taking advantage of the diversity due to MultQuery, CrossQuery and hostname aggregation. Furthermore, we show that current CDS optimizations that approximate the location of end-users based on the location of the local DNS resolvers are more effective than those based on the location of third-party resolvers. Finally, we show that the traffic localization potential within the above mentioned ISP is very high especially when the ISP DNS resolver is utilized.

6 Content Delivery: An Overview

While content may be seen as king not all content is equally popular among users. Indeed, content popularity often follows “Zipf’s law”. If the popularity of elements as function of the rank is consistent with a power-law distribution it is referred to as Zipf’s-like (see [242, 158] and references therein). The rank is determined by the number of occurrence of an element, where a low rank index refers to a popular element. Not surprisingly Zipf’s law does not only apply to the popularity of content but also quite a number of different quantities in Internet traffic, including the popularity of Web pages [38, 203], traffic demands [70, 73, 236, 227, 50], as well as interdomain Web traffic demands [75]. Thus, while some content can be served by a single server most content, namely the popular content, can only be served if it is highly replicated across multiple servers. Thus, one of the main challenges in content delivery is **server selection**. Server selection means identifying a specific server from which the request for content by a user is satisfied.

Content delivery and the network infrastructure interact mostly through content source selection. Here, it does not matter whether the source is a server pushing content through HTTP or from a peer in a P2P network. In the case of HTTP, the domain name system (DNS) is the preferred mechanism for performing source selection. In the case of P2P, peer selection strategies drive where the content is obtained from and how, e.g., when the content is cut into chunks.

To direct users to appropriate servers, CDNs rely extensively on the Domain Name System (DNS). When requesting content, the end user typically only uses a generic hostname, e.g., facebook.com. To map this generic hostname to a server, or more concretely a specific IP address, the user machine contacts a DNS resolver, for the resolution of a domain name. The resolver then asks the authoritative server for the domain. This can be the CDN’s authoritative server, or the the content provider’s authoritative server, which then delegates to the CDN’s authoritative server. The CDN chooses a server based several metrics. Criteria for server selection include the IP address of the end user’s DNS resolver, the availability of the server, the proximity of the server to the resolver, and the monetary cost of delivering the content. Note that the server selection does not know the client IP address or network location, it only knows the IP Address of the DNS resolver the end-user contacted. A recent study [10] showed that sometimes the end user is not close to the resolver. To improve the mapping of end users to servers, the client-IP eDNS extension [55] has been recently proposed.

In P2P systems peers can choose among all other peers to download content from but only if the have the desired content. Thus the problem of getting content in a P2P system is actually two-fold: first the user needs to find the content and once it knows of possible peers it can download the content from, it needs to connect to some of them to get the desired content. In P2P systems the content lookup is realized in many different ways. Some P2P network, called structured P2P, implement a distributed lookup system most often referred to as distributed hash table (DHT). Other P2P systems, called unstructured P2P, like Gnutella, flood search request into the network. Some systems rely on a partial centralized infrastructure to obtain content information. We discuss the different approaches in P2P systems in more detail in section 6.3.

Before we can discuss all the various options on how content delivery can be improved in the current Internet we give a short overview how a typical Content Distribution Network operates.

6.1 Terminology

The following definitions, taken in part taken from the Web Characterization Terminology & Definitions Sheet [132], will serve to clarify the subsequent discussions.

Web site: A collection of interlinked Web objects hosted at the same network location by a set of **origin Web servers**.

Web site publisher, or just **publisher:** A person or corporate body that is the primary claimant to the rewards or benefits resulting from usage of the content of a Web site. A publisher may distribute his content across multiple Web sites. Publishers are also referred to as content providers.

Content delivery network: An alternative infrastructure operated by an independent service provider on which some parts of a Web site can be hosted.

Peer: Participant of a Peer-to-Peer system. All peers are equally privileged in the system, hence every peer can offer and request the services (e.g., content, storage) the P2P system offers from other peers.

6.2 Content Delivery Networks

Recall content is king in the current Internet and content is typically first placed on the Web site of the content producer, the original Web servers. Content Delivery Networks (CDNs) (see, e.g., [104, 63, 86, 35, 110, 125, 196]) are designed to reduce the load on origin servers and at the same time improve performance for the user. Most CDNs have a large set of servers deployed throughout the Internet and cache the content of the original publisher at these servers. Therefore another view of CDNs is that they provide reverse proxy services for content providers, the publishers. In order to take advantage of their distributed infrastructure, requests for data are redirected to the "closest" cache server. Intelligent redirection can reduce network latency and load (and therefore network congestion) improving response time. CDNs differ in their approach to redirecting traffic. Some (such as Akamai [18]), use DNS to translate the hostname of a page request into the IP address of an appropriate server. This translation may consider the location of the client, the location of the server, the connectivity of the client to the server, the load on the server, and other performance and cost based criteria.

An example that shows how the CDN infrastructure is embedded in the Internet architecture is shown in Figure 17. Recall, the Internet is divided into a collection of autonomous systems (ASes). Each AS is managed by an Internet Service Provider (ISP), who operates a backbone network that provides connectivity to clients and to other ISPs. Figure 17 shows four ASes, numbered 1–4, whose backbones consist of three routers each[4] two Web site publishers, home.ex and adserver.ex, and two sets of

[4]Most backbones consist of a larger number of routers, of course.

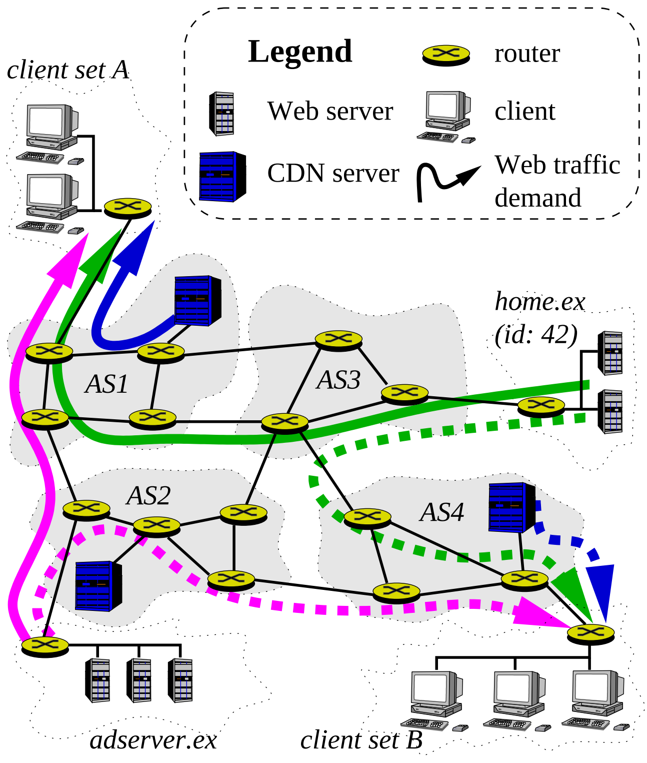


Figure 17: Example of CDN deployment and traffic flows (Web traffic demands).

clients. The publisher home.ex is connected to AS 3 while the publisher adserver.ex is connected to AS 2. A set of clients is connected to AS 1, another to AS 4.

The location of the CDN's servers differ from CDN to CDN and depends on contractual agreements between the CDN and the individual ISPs. In some instances, the CDN servers are deployed within the data centers of the ISP and therefore belong to the same AS, like AS 1, 2, 4 in Figure 17. Clients of the ISP (end users) are typically served by these servers in the same AS. With other ISPs, the CDN may have a private peering agreement that allows the CDN to serve requests from the ISPs clients via a direct connection between the CDN and the AS. The CDN may also co-locate servers with the ISP's clients, e.g., on university campuses. With other ISPs there may be no relationship with the CDN, and the traffic to the ISP's clients is routed via another AS.

Let us consider the steps that are necessary to download the Web page shown in Figure 18. This page consists of one main page located at home.ex/index.htm and four embedded objects. The publisher responsible for home.ex has decided to use the services of a CDN, cdn.ex. One object (ex2.gif) of the sample page is located on the same server as the page itself (index.htm); another object (ex3.gif) is served by a company providing dynamic advertisements, adserver.ex; and objects ex1.gif and ex4.jpg are hosted by the CDN.

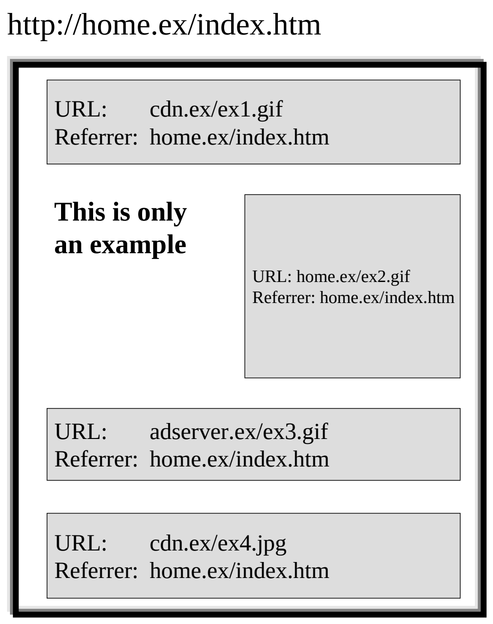


Figure 18: Example Web page with some CDN content.

If a specific client from client set A in Figure 17 accesses the Web page, publisher home.ex serves the bytes for the main page and one embedded object, publisher adserver.ex serves the bytes for the object located on its servers, and the "nearest" CDN server serves the two CDN-located objects—in this case, they will be served from AS 1. In contrast, if a specific client from client set B accesses the page, the two CDN objects are delivered from a different CDN server, namely the one in AS 4. Keep in mind that it is the objective of the CDN to direct the client to a CDN server that is close to the client.

To complete the picture one question remains. How does the CDN choose the "nearest" server to deliver the content from? Todays CDN landscape relies mainly on three techniques to assign end-users to servers.

1. IP-Anycast
2. DNS based redirection
3. HTTP redirection

While all techniques help the CDNs to assign end-users to their servers, all of them have different drawbacks. In the following we will explain how the different techniques work and what those drawbacks are:

IP-Anycast IP Anycast is a routing technique used to send IP packets to the topologically closest member of a group of potential CDN servers. IP Anycast is usually realized by announcing the destination address from multiple locations in a network or on the Internet. Since the same IP address is available at multiple locations, the routing process selects the shortest route for the destination according to its configuration. Simply speaking, each router in a network selects one of the locations the Anycasted IP is announced from based on the used routing metrics (e.g., path length or routing weights) and configures a route towards it. Note that, if a network learns of an Anycasted IP address from different sources, it does not necessarily direct all its traffic to one of locations. Its routing can decide to send packets from region A in the network to location A' while region B gets a route to location B'. This means that the entire server selection of a CDN becomes trivial as it is now a part of the routing process. This means that the CDN looses control of how the users are mapped to the server because the network calculates the routing based on its own metrics. Another issue is that the routing in a network is optimized based on the ISPs criteria which might not be the same as the CDNs or even contrary. Thus the "nearest" server might not be the best one the CDN could offer.

DNS based redirection Today most CDNs rely on the Domain Name System (DNS) to direct users to appropriate servers. When requesting content, the end user typically asks a DNS resolver, e.g., the resolver of its ISP, for the resolution of a domain name. The resolver then asks the authoritative server for the domain. This can be the CDN's authoritative server, or the the content provider's authoritative server, which then delegates to the CDN's authoritative server. At this point the CDN selects the server for this request based on where the request comes from. But the request does not come directly from the end-user but from its DNS resolver! Thus the CDN can only select a server based on the IP address of the end user's DNS resolver. To improve the mapping of end users to servers, the client-IP eDNS extension [55] has been recently proposed. Criteria for server selection include the availability of the server, the proximity of the server to the resolver, and the monetary cost of delivering the content. For proximity estimations the CDNs rely heavily on network measurements [170] and geolocation information [153] to figure out which of their servers is close by and has the best network path performance. A recent study [10] showed that sometimes the end user is not close to the resolver and another study points out that geolocation databases can not be relied upon [185]. Thus the proximity estimations for the "nearest" CDN server highly depend on the quality and precision of network measurements and a proper DNS deployment of the ISPs.

HTTP redirection The Hypertext Transfer Protocol (HTTP) is todays de-facto standard to transport content in the Internet (see section 5). The protocol incorporates a mechanism to redirect users at the application level at least since it was standardized as version 1.0 in 1996 [36]. By sending an appropriate HTTP status code (HTTP status codes 3XX) the web server can tell the connected user that a requested object is available from another URL, which can also point to another server. This allows a CDN to redirect an end-user to another server. Reasons for this might include limited server capacities, poor transfer performance or when another server is closer to the end-user, e.g., a client from the US connecting to a server in Europe although the CDN has servers in the US. The HTTP redirection mechanism has some important benefits over the DNS based approach. First, the CDN directly communicates with the end-user and thus knows the exact destination it sends the traffic to (opposed to the assumption that the DNS resolver is "close"). Yet it still has to estimate the proximity of the end-user using the same methodologies as described in the DNS based case. Second, the CDN already knows which object the end-user requests and can use this information for its decision. It allows a CDN to direct a user towards a server where the content object is already available to improve its cache hit rate. Other important informations includes the size and type of the object. This allows the CDN to optimize the server selection based on the requirements to transfer the object e.g., for delay sensitive ones like streaming video or more throughput oriented ones like huge software patches. Yet this improvement comes at a price as the user has to establish a new connection to another server. This includes another DNS lookup to get the servers IP address as well as the whole TCP setup including performance critical phases like slow start. This can repeat itself multiple times before an appropriate server is found, which delays the object delivery even further.

6.3 Peer-to-Peer Networks

Peer-to-peer (P2P) is a distributed system architecture in which all participants, the so called peers, are equally privileged users of the system. A P2P system forms an overlay network on top of existing communication networks (e.g., the Internet). All participating peers of the P2P system are the nodes of the overlay network graph, while the connections between them are the edges. It is possible to extend this definition of edges in the overlay network graph to all known peers, in contrast to all connected peers. Based on how peers connect to each other and thus build the overlay network, we can classify P2P systems into two basic categories:

Unstructured: The P2P system does not impose any structure on the overlay network. The peers connect to each other in an arbitrary fashion. Most often peers are chosen randomly. Content lookups are flooded to the network (e.g., Gnutella), resulting in limited scalability, or not offered at all (e.g., plain BitTorrent).

Structured: Peers organize themselves following certain criteria and algorithms. The resulting overlay network graphs have specific topologies and properties that usually offer better scalability and faster lookups than unstructured P2P systems (e.g., Kademlia, BitTorrent DHT).

The overlay network is mainly used for indexing content and peer discovery while the actual content is transferred directly between peers. Thus the connection between the individual peers has significant impact on both the direct content transfers as well as the performance of the resulting overlay network. This has been shown in previous studies and multiple solutions have been proposed [233, 45, 216, 17, 21, 167] which are described in detail in section 10.

Applications of P2P systems in content delivery range from time insensitive applications like file sharing, software delivery or patch distribution to very time sensitive ones like streaming TV or on demand video delivery.

Peer-to-Peer systems To construct an overlay topology, unstructured P2P networks usually employ an arbitrary neighbor selection procedure [212]. This can result in a situation where a node in Frankfurt downloads a large content file from a node in Sydney, while the same information may be available at a node in Berlin. While structured P2P systems follow certain rules and algorithms, the information available to them either has to be inferred by measurements [69] or rely on publicly available information such as routing information [193]. Both options are much less precise and up-to-date compared to the information information an ISP has readily at hand. It has been shown that P2P traffic often crosses network boundaries multiple times [14, 114]. This is not necessarily optimal as most network bottlenecks in the Internet are assumed to be either in the access network or on the links between ISPs, but rarely in the backbones of the ISPs [20]. Besides, studies have shown that the desired content is often available "in the proximity" of interested users [114, 189]. This is due to content language and geographical regions of interest. P2P networks benefit from increasing their traffic locality, as shown by Bindal et. al [37] for the case of BitTorrent.

P2P systems usually implement their own routing [23] in the overlay topology. Routing on such an overlay topology is no longer done on a per-prefix basis, but rather on a query or key basis. In unstructured P2P networks, queries are disseminated, e.g., via flooding [91] or random walks, while structured P2P networks often use DHT-based routing systems to locate data [212]. Answers can either be sent directly using the underlay routing [212] or through the overlay network by retracing the query path [91]. By routing through the overlay of P2P nodes, P2P systems hope to use paths with better performance than those available via the Internet native routing [23, 198]. However, the benefits of redirecting traffic on an alternative path, e.g., one with larger available bandwidth or lower delay, are not necessarily obvious. While the performance of the P2P system may temporarily improve, the available bandwidth of the newly chosen path may deteriorate due to the traffic added to this path. The ISP has then to redirect some traffic so that other applications using this path can receive enough bandwidth. In other words, P2P systems reinvent and re-implement a routing system whose dynamics should be able to explicitly interact with the dynamics of native Internet routing [115, 202]. While a routing underlay as proposed by Nakao et al. [163] can reduce the work duplication, it cannot by itself overcome the problems created by the interaction. Consider a situation where a P2P system imposes a lot of traffic load on an ISP network. This may cause the ISP to change some routing metrics and therefore some paths (at the native routing layer) in order to improve its network utilization. This can however cause a change of routes at the application layer by the P2P system, which may again trigger a response by the ISP, and so on.

P2P today The P2P paradigm has been very successful in delivering content to end-users. BitTorrent [53] is the prime example, used mainly for file sharing. Other examples include more time sensitive applications such as video streaming [65, 147, 127]. Despite the varying (and perhaps declining) share of P2P traffic in different regions of the world [150], P2P traffic still constitutes a significant fraction of the total Internet traffic. P2P systems have been shown to scale application capacity well during flash crowds [234]. However, the strength of P2P systems, i.e., anybody can share anything over this technology, also turns out to be a weakness when it comes to content availability. In fact, mostly popular content is available on P2P networks, while older content disappears as users' interest in it declines. In the example of BitTorrent, this leads to torrents missing pieces, in which case a download can never be completed. In case of video streaming, the video might simply no longer be available or the number of available peers is too low to sustain the required video bit-rate, resulting in gaps or stuttering of the video stream.

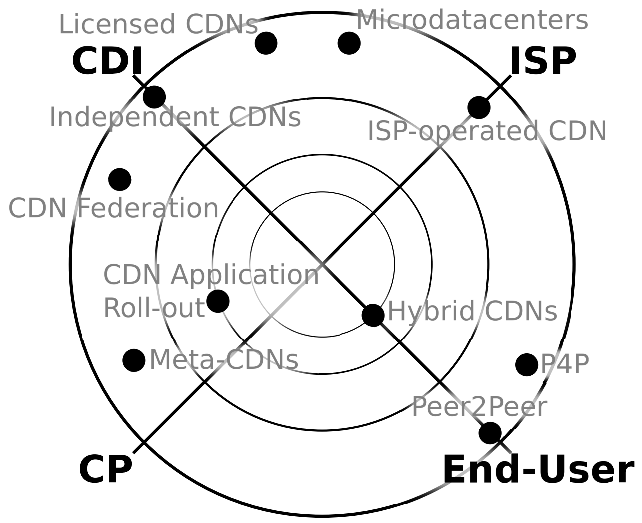


Figure 19: Content Delivery Spectrum

7 Content Delivery Landscape

Internet traffic grows at a rate of approximately 30% per year [49] and is dominated by the delivery of content to end users [9, 87, 128, 183]. To cope with the increasing demand for content, and to support the level of reliability and scalability required by commercial-grade applications, Content Distribution Infrastructures (CDIs) have emerged. In general terms, CDIs are overlays built on top of existing network infrastructures that aim to accelerate the delivery of content to end users. CDIs include, but are not limited to, Content Distribution Networks (CDNs), such as Akamai and Google, Video Streaming Portals (VSP) such as YouTube, One-Click-Hosters (OCH) like Rapidshare and Fileserve. However, a CDI does not necessarily produce the content that it delivers. Thus, we define a Content Producer (CP) as the entity that generates content. In some cases, e.g., Google and YouTube, the CP and CDI can be the same entity. In other instances, for example Akamai and Limelight, the CDI only delivers what a CP pays for.

But not all CDIs are built upon the same philosophy, designs and technology. For example, a CDI can be operated independently by deploying caches in different networks, by renting space in datacenters or by building its own datacenters. Furthermore, some CDIs are operated by ISPs, by Content Producers, or in the case of Peer-to-Peer networks, by self-organized end-users. To summarize the spectrum of CDI solutions, Figure 19 provides an overview of different CDI solutions. They are aligned by their architectures according to which parties are involved.

7.1 Independent Content Distribution

Independent CDIs are usually referred to as Content Delivery Networks (CDNs). They have a strong customer base of content producers and are responsible for delivering the content of their customers to end-users around the world. Today, they are, by traffic volume as well as hosted content, the largest players on the Internet. In general, there are four main components to independent CDN architectures: a server deployment, a strategy for replicating content on servers, a mechanism for directing users to servers, and a system for collecting and processing server logs.

For server deployment, three main approaches exist [136]: centralized, datacenter based and distributed infrastructures:

Central Location: This approach is used by small CDNs, One-Click Hosters, and applications running in public clouds. Centralized hosting takes advantage of (a) the economies of scale that a single location offers [26], (b) the flexibility that multi-homing offers [93], and (c) the connectivity opportunities that IXPs offer [9]. The disadvantages of centralized hosting are the potential for a single point of failure, and the limited ability to ensure low latency to users located in different networks around the world [140].

Datacenter Based: This approach deploys in several large data centers. It again leverages economies of scale while improving reliability and creating a larger foot-print with further reach. However, by utilizing multiple datacenters, new challenges regarding the content distribution, synchronization and delivery arise. For example, the datacenter delivering content to an end-user cannot be statically configured any-more, but the selection needs to take the location of the end-user into account. This approach is used by CDNs such as Limelight, EdgeCast and BitGravity. Many cloud providers also use this approach, including Amazon CloudFront and Microsoft Azure.

Distributed Infrastructures: This approach consists of a highly distributed in-frastructure deployment, potentially deep inside third-party networks. Here, the large number of servers scattered across numerous networks offer high availability and repli-cation of content while being very close to end-users. Furthermore, this type can bal-ance traffic across locations, best react to flash crowds by dynamic server assignments, and deliver content with improved latency. However, with the highly distributed in-frastructures, the challenges of assigning users to the right server location increase many-fold. Also, with deep deployment datacenters are usually not available anymore, leading to the question where to deploy how many servers. Today, Akamai is only one independent CDN that uses this approach on a global scale.

CDNs with more than one location typically follow a pull strategy [170] for content distribution and replication. Thus, content requests can be directed to servers that do not have the required object cached. When a requested object is not at the selected server, neighboring servers in the same cluster or region are asked. If the object is not available at neighboring servers, the origin or root server responsible for the object is contacted to retrieve the content. A requested object that is fetched from a remote server is saved locally and then delivered to the end user. To keep the copies of the object fresh, a TTL value is assigned to it. When the TTL value expires, the object is removed. For scalability reasons, any server of the CDN or within a region can respond to the request of an end user [221].

A special case of the independent CDI category are free CDNs such as Coral [85], which follow a similar architectural design. In these CDNs, server resources are offered by end-users or non-profit organizations.

7.2 ISP-operated CDIs

The potential for generating revenue from content delivery has motivated a number of ISPs to build and operate their own Content Distribution Infrastructures. For example, large ISPs such as AT&T and Verizon have built their own CDNs along the same general architectural principles as independent CDIs. However, due to the limitations arising from being restricted to one network, these CDNs are not deployed in a distributed fashion across multiple networks and thus are not globally operating solutions. To overcome this issue, the CDNi group at the IETF [167] is discussing how to interconnect these CDNs to boost their efficiency and coverage. The content provider are third parties, applications and services offered by the ISP. Other ISPs with large footprints, such as Level3 and Telefonica [130, 131], have also built CDNs in order to efficiently transfer content across the globe and offer improved services to their end users.

7.3 Emerging Trends in CDI Architectures

Economics, especially cost reduction, is the key driving force behind emerging CDI architectures. The content delivery market has become highly competitive. While the demand for content delivery services is rising and the cost of bandwidth is decreasing, the profit margins of storage and processing [26] are dwindling, increasing the pressure on CDIs to reduce costs. At the same time, more parties are entering the market in new ways, looking to capture a slice of the revenue. However, today's traditional CDI deployments lack agility to combat these effects. Contracts for server deployments last for months or years and the available locations are typically limited to datacenters. The time required to install a new server today is in the order of weeks or months. Such timescales are too large to react to changes in demand. CDIs are therefore looking for new ways to expand or shrink their capacity, on demand, and especially at low cost.

7.3.1 Hybrid Content Distribution

In a hybrid CDI, end users download client software that assists with content distribution. As in P2P file-sharing systems, the content is broken into pieces and offered by both other users who have installed the client software as well as by the CDI's servers. The client software contacts dedicated CDI servers, called control plane servers, which schedule which parts of the content are to be downloaded from what peers. Criteria for selecting peers include AS-level proximity as well as the availability of the content. If no close peers are found, or if the download process from other peers significantly slows the content delivery process, the traditional CDI servers take over the content delivery job entirely. Akamai already offers NetSession [8], a hybrid CDI solution for delivering very large files such as software updates at lower cost to its customers. Xunlei [62], an application aggregator with high penetration in China, follows a similar paradigm. It is used to download various types of files including videos, executables, and even emails, and supports popular protocols such as HTTP, FTP, and RTSP. Xunlei maintains its own trackers and servers. A study of hybrid CDIs [102] showed that up to 80% of content delivery traffic can be outsourced from server-based delivery to end users, without significant degradation in total download time.

7.3.2 Licensed CDNs

Licensed CDNs have been proposed to combine the benefits of the large content-provider customer base of an independent CDI with the end user base of an ISP [213]. A licensed CDN is a partnership between an independent CDI and an ISP. The CDI licenses the content delivery software that runs on servers to the ISP while the ISP owns and operates the servers. The servers deliver content to the end users and report logging information back to the CDI. The revenue derived from content producers is then shared between the two parties. Thus, a CDI can expand its footprint deep inside an ISP network without investing in hardware, incurring lower operating costs. The ISP benefits from not having to invest in developing the software for a reliable and scalable content distribution. More importantly, a licensed CDN also alleviates the ISP's need to negotiate directly with content producers, which might be challenging, given an ISPs limited footprint.

7.3.3 Application-based CDIs

Recently, large application and content producers have rolled out their own CDIs, hosted in multiple large data centers. Some popular applications generate so much traffic that the content producers can better amortize delivery costs by doing content distribution themselves. Google is one such example. It has deployed a number of data centers and interconnected them with high speed backbone networks. Google connects its datacenters to a large number of ISPs via IXPs and also via private peering. Google has also launched the Google Global Cache (GGC) [95], which can be installed inside ISP networks. The GGC reduces the transit cost of small ISPs and those that are located in areas with limited connectivity, e.g., Africa. The GGC servers are given for free to the ISPs which install and maintain them. GGC also allows an ISP to advertise through BGP the prefixes of users that each GGC server should serve. As another example, Netflix, which is responsible for around 30% of the traffic in North America at certain times, is also rolling out its own CDI. The Netflix system is called Open Connect Network [166]. Netflix offers an interface where ISPs can advertise, via BGP, their preferences as to which subnets are served by which Open Connect Network servers.

7.3.4 Meta-CDIs

Today, content producers contract with multiple CDIs to deliver their content. To optimize for cost and performance [146], meta-CDIs act as brokers to help with CDI selection. These brokers collect performance metrics from a number of end users and try to estimate the best CDI, based on the server that a user is assigned. To this end, the brokers place a small file on a number of CDIs. Then they embed the request for this file in popular websites' source code, in the form of a javascript. When users visit these sites, they report back statistics based on the servers that each CDI assigned the users. The broker then recommends CDIs for a given source of demand taking also into consideration the cost of delivery. Cedexis is one of these brokers for web browsing. Another broker for video streaming is Conviva [65]. These brokers may compensate when a CDI does not assign a user to the optimal server (which a recent study [183] has shown sometimes occurs) by selecting a different CDI.

7.3.5 CDI Federations

To avoid the cost of providing a global footprint and perhaps to allow for a single negotiating unit with content providers, federations of CDIs have been proposed. In this architecture, smaller CDIs, perhaps operated by ISPs, join together to form a larger federated CDI. A CDI belonging to the federation can replicate content to a partner CDI in a location where it has no footprint. The CDI reduces its transit costs because it only has to send the object once to satisfy the demand for users in that location. Overall, cost may be reduced due to distance-based pricing [223]. The IETF CDNi working group [167] works on CDI federation.

7.4 Hybrid P2P-CDN Infrastructures

Today, BitTorrent is one of the most popular peer-to-peer (P2P) systems and has been adopted in a similar fashion by content distribution network such as Akamai for time insensitive content delivery such as patch delivery or software distribution. Another prominent use case of BitTorrent is the patch distribution of World of Warcraft, an on-line multiplayer game, which patches are in the size of multiple gigabyte. In the realm of video streaming P2P is especially popular in China [62] and numerous services make use of this technology, e.g., BitTorrent Live or Zattoo.com.

8 Challenges in Content Delivery

The challenges that CDIs and P2P systems are faced with are based on the fact that they are unaware of the underlying network infrastructure and its conditions. In the best case, they can try to detect and infer the topology and state of the ISP's network through measurements, but even with large scale measurements, it is a difficult task, especially if accuracy is necessary. Furthermore, when it comes to short-term congestion and/or avoiding network bottlenecks, measurements are of no use. In the following we describe the challenges those systems face in more detail.

8.1 Content Delivery Networks (CDNs)

From the viewpoint of the end-users and ISPs, the redirection schemes employed by existing CDNs have three major limitations:

Network Bottlenecks Despite the traffic flow optimization performed by CDNs, the assignment of end-user requests to servers by CDNs may still result in sub-optimal content delivery performance for the end-users. This is a consequence of the limited information CDNs have about the network conditions between the end-user and their servers. Tracking the ever changing conditions in networks, i.e., through active measurements and end-user reports, incurs an overhead for the CDN without a guarantee of performance improvements for the end-user. Without sufficient information about the network paths between the CDN servers and the end-user, any assignment performed by the CDN may lead to additional load on existing network bottlenecks, or to the creation of new bottlenecks.

User Mis-location DNS requests received by the CDN DNS servers originate from the DNS resolver of the end-user, not from the end-user itself. The assignment is therefore based on the assumption that end-users are close to their DNS resolvers. Recent studies have shown that in many cases this assumption does not hold [151, 10]. As a result, the end-user is mis-located and the server assignment is not optimal. As a response to this issue, DNS extensions have been proposed to include the end-user IP information [55].

Content Delivery Cost Finally, CDNs strive to minimize the overall cost of delivering huge amounts of content to end-users. To that end, their assignment strategy is mainly driven by economic aspects. While a CDN will always try to assign users in such a way that the server can deliver reasonable performance, this can again result in end-users not being directed to the server able to deliver best performance.

8.2 Peer-to-Peer Networks (P2P)

Historically, the wide-spread use of such P2P systems has put ISPs in a dilemma! On the one hand, P2P system applications have resulted in an increase in revenue for ISPs, as they are one of the major reasons cited by Internet users for upgrading their

Internet access to broadband [156]. On the other hand, ISPs find that P2P traffic poses a significant traffic engineering challenge [142, 115]. P2P traffic often starves other applications like Web traffic of bandwidth [205], and swamps the ISP network. This is because most P2P systems rely on application layer routing based on an overlay topology on top of the Internet, which is largely independent of the Internet routing and topology [14].

To construct an overlay topology, unstructured P2P networks usually employ an arbitrary neighbor selection procedure [212]. This can result in a situation where a node in Frankfurt downloads a large content file from a node in Sydney, while the same information may be available at a node in Berlin. It has been shown that P2P traffic often crosses network boundaries multiple times [14, 114]. This is not necessarily optimal as most network bottlenecks in the Internet are assumed to be either in the access network or on the links between ISPs, but not in the backbones of the ISPs [20]. Besides, studies have shown that the desired content is often available "in the proximity" of interested users [114, 189]. This is due to content language and geographical regions of interest. Since a P2P user is primarily interested in finding his desired content quickly with good performance, we believe that increasing the locality of P2P traffic will benefit both ISPs and P2P users.

To better understand the origin of the problem of overlay-underlay routing clash, let us recall how routing works in the Internet and P2P systems. In the Internet, which is a collection of Autonomous Systems (ASes), packets are forwarded along a path on a per-prefix basis. This choice of path via the routing system is limited by the contractual agreements between ASes and the routing policy within the AS (usually shortest path routing based on a fixed per link cost) [100].

P2P systems, on the other hand, setup an overlay topology and implement their own routing [23] in the overlay topology which is no longer done on a per-prefix basis but rather on a query or key basis. In unstructured P2P networks queries are disseminated, e.g., via flooding [91] or random walks while structured P2P networks often use DHT-based routing systems to locate data [212]. Answers can either be sent directly using the underlay routing [212] or through the overlay network by retracing the query path [91]. By routing through the overlay of P2P nodes, P2P systems hope to use paths with better performance than those available via the Internet [23, 198]. But the benefits of redirecting traffic on an alternative path, e.g., one with larger available bandwidth or lower delay, are not necessarily obvious. While the performance of the P2P system may temporarily improve, the available bandwidth of the newly chosen path will deteriorate due to the traffic added to this path. The ISP then has to redirect some traffic so that other applications using this path receive enough bandwidth. In other words, P2P systems reinvent and re-implement a routing system whose dynamics should be able to interact with the dynamics of the Internet routing [115, 202]. While a routing underlay as proposed by Nakao et al. [163] can reduce the work duplications it cannot by itself overcome the interaction problems. Consider a situation where a P2P system imposes a lot of traffic on an ISP network. This may cause the ISP to change some routing metrics and therefore some paths (at the routing layer) in order to improve its network utilization. This can however cause a change of routes (at the application layer) by the P2P system, which may again trigger a response by the ISP, and so on.

8.3 Internet Service Providers (ISPs)

ISPs face several challenges regarding the operation of their network infrastructure. With the emergence of Content, and especially distributed content delivery, be it from CDIs or P2P networks, these operational challenges have increased manifold.

Network Provisioning Provisioning and operation a network means running the infrastructure at its highest efficiency. To ensure this, new cables as well as the peering points with other networks need to be established and/or upgraded. However, with the emergence of CDIs and P2P networks, the network provisioning has become more complicated, since the network loads tend to shift depending on the content that is currently transported while the direct peering might not be effective anymore.

Volatile Content Traffic CDIs and P2P networks strive to optimize their own operational overhead, possibly at the expense of the underlying infrastructure. In terms of CDIs, this means that a CDI chooses the best server based on its own criteria, not knowing what parts of the networks infrastructure is being used. Especially with globally deployed CDIs it becomes increasingly difficult for ISPs to predict what CDI is causing what traffic from where based on past behavior. This has a direct implication on the traffic engineering of the network, as this is usually based on traffic predictions from past network traffic patterns.

Customer Satisfaction Regardless of the increased difficulty with network provisioning and traffic engineering, end-users are demanding more and larger content. This, coupled with the dominant for of flatrates for customer subscriptions, increases the pressure on ISPs to delay network upgrades as long as possible to keep prices competitive. But letting links run full increases packet loss. This, in turn, drastically reduces the quality of experience of the end-users. This, in turn, encourages end-users to switch their subscriptions.

8.4 Summary

In summary, we identify the following drawbacks:

- The ISP has limited ability to manage its traffic and therefore incurs potentially increased costs, e.g., for its interdomain traffic, as well as for its inability to do traffic engineering on its internal network while having to offer competitive subscriptions to its end-users.
- The P2P system has limited ability to pick an optimal overlay topology and therefore provide optimal performance to its users, as it has no prior knowledge of the underlying Internet topology. It therefore has to either disregard or reverse engineer it.
- The Content Delivery Network has limited ability to pick the optimal server and therefore provide optimal performance to its users, as it has to infer the network topology as well as the dynamic network conditions. Moreover, it has limited

knowledge about the location of the user as it only knows the IP address of the DNS resolver.

- The Different systems try to measure the path performance independently.

9 Incentives for Collaboration

ISPs are in a unique position to help CDIs and P2P systems to improve content delivery. Specifically, ISPs have the knowledge about the state of the underlying network topology and the status of individual links that CDIs are lacking. This information not only helps CDIs in their user-to-server mapping, but also reduces the need for CDIs to perform large-scale active measurements and topology discovery [19]. It also enables CDIs to better amortize their existing infrastructure, offer better quality of experience to their users, and plan their infrastructure expansion more efficiently. On the other side, ISPs are not just selflessly giving up their network information. Offering their intimate knowledge of the network to CDIs puts ISPs in the position that they can also actively guide the CDIs. This allows ISPs to gain unprecedented influence on CDI traffic.

The opportunity for ISPs to coordinate with CDIs is technically possible thanks to the decoupling of server selection from content delivery. In general, any end-user requesting content from a CDI first does a mapping request, usually through the Domain Name System (DNS). During this request, the CDI needs to locate the network position of the end-user and assign a server capable of delivering the content, preferably close to the end-user. ISPs have this information ready at their fingertips, but are currently not able to communicate their knowledge to CDIs. Furthermore, ISPs solve the challenge of predicting CDI traffic, which is very difficult due to the lack of information on the CDI mapping strategy regarding the end-users to servers assignment. In order to reap the benefits of the other's knowledge, both parties require incentives to work together.

9.1 Incentives for CDIs

The CDIs' market requires them to enable new applications while reducing their operational costs and improve end-user experience [170]. By cooperating with an ISP, a CDI improves the mapping of end-users to servers, improves in the end-user experience, has accurate and up-to-date knowledge of the networks and thus gains a competitive advantage. This is particularly important for CDIs in light of the commoditization of the content delivery market and the selection offered to end-users, for example through meta-CDNs [65]. The improved mapping also yields better infrastructure amortization and, thanks to cooperation with ISPs, CDIs will no longer have to perform and analyze voluminous measurements in order to infer the network conditions or end-user locations.

To stimulate cooperation, ISPs can operate and provide their network knowledge as a free service to CDIs or even offer discounts on peering or hosting prices, e.g., for early adopters and CDIs willing to cooperate. The loss of peering or hosting revenue is amortized with the benefits of a lower network utilization, reduced investments in network capacity expansion and by taking back some control over traffic within the network. Ma et al. [148] have developed a methodology to estimate the prices in such a cooperative scheme by utilizing the Shapley settlement mechanism. Cooperation can also act as an enabler for CDIs and ISPs to jointly launch new applications in a cost-effective way, for example traffic-intensive applications such as the delivery of high definition video on-demand, or real-time applications such as online games.

9.2 Incentives for ISPs

ISPs are interested in reducing their operational and infrastructure upgrade costs, offering broadband services at competitive prices, and delivering the best end-user experience possible. Due to network congestion during peak hour, ISPs in North America have recently revisited the flat pricing model and some have announced data caps to broadband services. A better management of traffic in their networks allows them to offer higher data caps or even alleviate the need to introduce them. From an ISP perspective, cooperation with a CDI offers the possibility to do global traffic and peering management through an improved awareness of traffic across the whole network. For example, peering agreements with CDIs can offer cooperation in exchange for reduced costs to CDIs. This can be an incentive for CDIs to peer with ISPs, and an additional revenue for an ISP, as such reduced prices can attract additional peering customers. Furthermore, collaboration with CDIs has the potential to reduce the significant overhead due to the handling of customer complaints that often do not stem from the operation of the ISP but the operation of CDIs [46]. Through this, ISPs can identify and mitigate congestion in content delivery, and react to short disturbances caused by an increased demand of content from CDIs by communicating these incidents back directly to the source.

9.3 Effect on End-users

Collaboration between ISPs and CDIs in content delivery empowers end-users to obtain the best possible quality of experience. As such, this creates an incentive for end-users to support the adoption of collaboration by both ISPs and CDIs. For example, an ISP can offer more attractive products, i.e., higher bandwidth or lower prices, since it is able to better manage the traffic inside its network. Also, thanks to better traffic engineering, ISPs can increase data caps on their broadband offers, making the ISP more attractive to end-users. Moreover, CDIs that are willing to jointly deliver content can offer better quality of experience to end-users. This can even be done through premium services offered by the CDI to its customers. For example, CDIs delivering streaming services can offer higher quality videos to end-users thanks to better server assignment and network engineering.

10 Opportunities for Collaboration

As pointed out ISPs are in a unique position to help CDIs and P2P systems to improve content delivery since they have the knowledge about the state of the underlying network topology, the status of individual links, as well as the location of the user. Accordingly, in this section we describe possible ways to enable collaboration, namely, the original Oracle concept proposed by Aggarwal et al [17], P4P proposed by Xie et al. [233], Ono proposed by Choffnes and Bustamante [45], PaDIS proposed by Poese et al. [184]. We then give an overview of the activities within the IETF which have been fulled to some extend by the proposed systems discussed in this section, namely ALTO and CDNi.

10.1 P2P Oracle Service

To overcome the mismatch between the overlay network and underlay routing network Aggarwal et al. [17] describe an oracle service to solve most of the issues related to P2P content delivery. Instead of the P2P node choosing neighbors independently, the ISP can offer a service, the *oracle*, that ranks the potential neighbors according to certain metrics. This ranking can be seen as the ISP expressing preference for certain P2P neighbors. Possible coarse-grained distance metrics are:

- Inside/outside of the AS
- Number of AS hops according to the BGP path [100]
- Distance to the edge of the AS according to the IGP metric [100]

For P2P nodes within the AS the oracle may further rank the nodes according to:

- Geographical information such as: same point of presence (PoP), same city
- Performance information such as: expected delay, bandwidth
- Link congestion (traffic engineering)

This ranking can then be used by the P2P node to select a closeby neighbor although there is no obligation. The benefit to P2P nodes of all overlays is multifold: (1) they do not have to measure the path performance themselves; (2) they can take advantage of the knowledge of the ISP; (3) they can expect improved performance in the sense of low latency and high throughput as bottlenecks [20] can be avoided.

The benefit to the ISPs is that they can influence the neighborhood selection process of the P2P network to, e.g., ensure locality of traffic flows and therefore again have the ability to manage the flow of their traffic. This will also allow them to provide better service to their customers and ensure fairness for other applications like Web traffic, etc. Besides, the ISPs will gain cost advantages, by reducing costs for traffic that leaves their internal network.

The oracle is available to *all* overlay networks. One does neither need nor want to use a separate oracle for each P2P network. Furthermore, as an open service, it can be queried by any application and is not limited to file-sharing systems. Hence, querying the oracle does not necessarily imply participation in file sharing systems. This should limit the desirability of the oracle logs to, e.g., the music industry. Moreover the P2P system could permute, e.g., the last byte of the IP addresses it is interested in or use an anonymization service for querying the oracle.

10.2 Proactive Network Provider Participation for P2P (P4P)

The "Proactive Network Provider Participation for P2P" is another approach to enable cooperative network information sharing between the network provider and applications. The P4P architecture [233] consists of a control plane, a management plane and a data plane whereas the data plane is optional and not further specified. The management plane objective is to monitor the behavior of the control plane. For the control plane P4P introduces iTrackers as portals operated by network providers that divides the traffic control responsibilities between providers and applications by exposing certain interfaces to the applications. Possible interfaces can be e.g., *policy*, *p4p-distance* or *capability*. The *policy* interface allows applications to obtain information about network policies such as coarse-grained time-of-day link usage policies or near-congestion and heavy-usage thresholds. The *capability* interface exports network capabilities to applications, e.g., cache locations or service classes while the *p4p-distance* interface allows the application to query costs and distance between peers according to networks. To calculate costs and distance between peer P4P introduces the notion of *p-distance*. Each iTracker maintains an internal map of the network in an internal representation of nodes and edges. A node represents an aggregated set of clients and different nodes can be at different aggregation levels. The p-distance between two nodes is computed from the network path and routing information available to the iTracker, e.g., OSPF weights and BGP preferences, financial costs for traffic on the used links or congestion levels. It is up to the network providers which information is used to calculate the p-distance thus the relevance of its value differs from network to network, even for the same application. The applications can use the p-distance to optimize the connections between peers while allowing the network operators to improve the impact that the traffic has on its network.

10.3 Travelocity-based Path Selection

In another approach to optimize the issue of proper peer selection Su et al. propose "techniques for inferring and exploiting the network measurements performed by CDNs for the purpose of locating and utilizing quality Internet paths without performing extensive path probing or monitoring" [216]. The underlying assumption for this approach is that the redirection times of Akamai highly correlated with network latencies on the path between the client and the selected edge server. This information can be leveraged by P2P systems to identify intermediate nodes for connections in the P2P overlay network. A prerequisite for this technique is that the overlay network is able to map a subset of its nodes to Akamai edge servers. As the number of peers in a P2P system is usually several orders of magnitude larger than the number of Akamai edge servers, finding hosts that are mapped to the same edge server should not be difficult. An additional benefit for network operators (ISPs) stems from the fact that Akamai deploys its edge servers deep inside their networks. Clients mapped to the same severs are more likely to be inside the same or neighboring networks thus reducing inter AS traffic. To find a high quality path between two nodes in the overlay network both nodes perform a race to determine the lowest delay path: the direct path between them or the path via a third node mapped to the same Akamai edge server. Once the race determined a "winning" path, the nodes start using this (overlay-)path for further communication.

10.4 Ono

Using the techniques from travelocity based path selection Choffnes and Bustamante present the design and evaluation of a system to improve the peer selection in P2P systems. Their system, called *Ono*, "recycles network views gathered at low cost from content distribution networks to drive biased neighbor selection without any path monitoring or probing" [45] and is implemented as a plugin for the Azureus BitTorrent client. Based on their observations that CDN redirection is driven primarily by latency [216], they formulate the following hypothesis: Peers that exhibit similar redirection behavior of the same CDN are most likely close to each other, probably even in the same AS. For this each peer performs periodic DNS lookups on popular CDN names and calculates how close other peers are by determining the cosine similarity with their lookups. To share the lookup among the peers they use either direct communication between Ono enabled peers or via distributed storage solutions e.g., DHT-based. They show that their system design scales well over 100.000 users and peers using this information are expected to generate less inter-domain traffic for the AS they're located in while improving performance all without sacrificing the robustness of the P2P system. On the downside Ono relies on the precision of the measurements that the CDNs perform and that their assignment strategy is actually bases mainly on delay. Should the CDNs change their strategy in that regard Ono might yield wrong input for the biased peer selection.

10.5 Provider-aided Distance Information System (PaDIS)

Given the trends regarding server resources and increasing user demand, content delivery systems have to address fundamental problems. One is end-user to server assignment problem, i.e., how to assign users to the appropriate servers. The key enabler for addressing this problem is *informed user-server assignment* or in short *user-server assignment*. It allows a CDN to receive recommendations from a network operator, i.e., a server ranking based on performance criteria mutually agreed upon by the ISP and CDN. The CDN can utilize these recommendations when making its final decision regarding end-user to server assignments. This enabler takes full advantage of server and path diversity, which a CDN has difficulty exploring on its own. Moreover, it allows the coordination of CDNs, content providers, and ISPs at the scale of seconds or even per request. Any type of CDN can benefit from this enabler including ISP-operated CDNs. The advantage of our enabler in comparison with other ISP-CDN [64, 109] and ISP-P2P [233] cooperation schemes is that no routing changes are needed.

In [184] Poese et al. propose a "Provider-aided Information Systems (PaDIS)", that realizes this enabler within an ISP. The main tasks of this system are:

- maintain an up-to-date annotated map of the ISP network and its properties as well as the state of the ISP-operated servers within the network.
- provide recommendation on where servers can be located to better satisfy the demand by the CDN and ISP traffic engineering goals,
- to assist the CDN in user-server assignment by creating preference rankings based on the current network conditions.

The goal of the system is to fully utilize the available server and path diversity as well as ISP-maintained resources within the network, while keeping the overhead for both the CDN and the ISP as small as possible. While the main focus of the PaDIS system is ISP-CDN collaboration, it is also suitable for P2P systems, as it is, simply put, a system that creates a preference based ranking of IPs for a given source. To make use of this service the P2P application could send a list of potential peer candidates to the PaDIS service and use the answer to make an informed decision to which peers to connect to. The PaDIS system comprises of the following components:
Network Monitoring The Network Monitoring component gathers information about the topology and the state of the network to maintain an up-to-date view of the network. The Topology Information component gathers detailed information about the network topology, i.e., routers and links, annotations such as link utilization, router load as well as topological changes. An Interior Gateway Protocol (IGP) listener provides up-to-date information about routers and links. Additional information, e.g., link utilization and other metrics can be retrieved via SNMP from the routers or an SNMP aggregator. The Routing Information uses routing information to calculate the paths that traffic takes through the network. Finding the path of egress traffic can be done by using a Border Gateway Protocol (BGP) listener. Ingress points of traffic into the ISP network can be found by utilizing Netflow data. This allows for complete forward and reverse path mapping inside the ISP. In total, this allows for a complete path map between any two points in the ISP network. The Network Map Database processes the information collected by the Topology and Routing Information components to build an annotated map of the ISP network. While it builds one map of the network, it keeps the information acquired from the other two components in separate data structures. The Topology Information is stored as a weighted directed graph, while the prefix information is stored in a Patricia trie [160]. This separation ensures that changes in prefix assignment learned via BGP do not directly affect the routing in the annotated network map. To further improve performance, the path properties for all paths are pre-calculated. This allows for constant lookup speed independent of path length and network topology. Having ISP-centric information ready for fast access in a database ensures timely responses and high query throughput.
Informed User-Server Assignment Component When the CDN sends a request for user-server assignment to PaDIS, the request is handled by the the Query Processor. The request from the CDN specifies the end-user and a list of candidate CDN servers. First, the Query Processor maps each source-destination (server to end-user) pair to a path in the network. Note that the end-user is seen through its DNS resolver, often the ISPs DNS resolver [10], unless both ISP and CDN support the EDNS0 Client Subnet Extension [54, 174]. The properties of the path are then retrieved from the Network Map Database. Next, the pairs are run individually through the Location Ranker sub-

component to get a preference value. Finally, the list is sorted by preference values, the values stripped from the list, and the list is sent back to the CDN. The ISP Location Ranker computes the preference value for individual source-destination pairs based on the path properties and an appropriate function. The function depends on the goal specified by the CDN, such as a performance goal, as well as an operational one, such as a traffic engineering objective. Note that PaDIS is not limited to a single optimization function per CDN.

10.6 Application-Layer Traffic Optimization (ALTO)

The research into P2P traffic localization has led the IETF to form a working group for "Application Layer Traffic Optimization (ALTO)". The goal of the working group is to develop Internet standards that offer "better-than-random" peer selection by providing information about the underlying network to the Application. The charter of the ALTO WG [152] notes the goal of designing a query-response protocol that the applications can query for an optimized peer selection strategy:

"The Working Group will design and specify an Application-Layer Traffic Optimization (ALTO) service that will provide applications with in- formation to perform better-than-random initial peer selection. ALTO services may take different approaches at balancing factors such as maximum bandwidth, minimum cross-domain traffic, lowest cost to the user, etc. The WG will consider the needs of BitTorrent, tracker-less P2P, and other applications, such as content delivery networks (CDN) and mirror selection" [152]

The result of the IETF WG so far consists of a protocol draft [21] that describes the necessary information and communication an ALTO compliant server offers. While the draft describes multiple different service not all of them are mandatory for ALTO compliance. ALTO offers multiple services to the Applications querying it, most notably are the Endpoint Cost Service and the Map service. The Endpoint Cost Service allows the Application the query the ALTO server for costs and rankings based on endpoints (usually IP subnets) and use that information for an optimized peer selection process or to pick the most suitable server of a CDI. The Network Map service makes use of the fact that most endpoints are in fact rather close to each other and thus can be aggregated into a single entity. The resulting set of entities is then called an ALTO Network Map. The definition of proximity in that case depends on the aggregation level, in one Map endpoints in the same IP subnet may be considered close while in another all subnets attached to the same Point of Presence (PoP) are close. In contrast to the Endpoint Cost Service the ALTO Network Map is suitable when more Endpoints need to be considered and offers better scalability, especially when coupled with caching techniques. Although the ALTO WG statement is more P2P centric, the service is also suitable to improve the connection to CDN servers.

10.7 Content Delivery Networks Interconnection (CDNI)

To cope with the increased demand for content CDNs scale up their infrastructure and more and more big content providers deploy their own CDN solutions. The need to further increase capacity and footprint of CDN solutions has led to a need for interconnecting standalone CDNs so they can interoperate and form an open and transparent infrastructure for content delivery. To facilitate open content delivery infrastructures the IETF has formed the *CDNI* working group to create a standardized and open specification for CDN Interconnection. RFC 6707 [168] outlines the problem area for the CDNI working group and RFC 6770 [168] focuses on use cases for CDN Interconnection. Goal of the working group is "to allow the interconnection of separately administered CDNs in support of the end-to-end delivery of content from CSPs through multiple CDNs and ultimately to end users (via their respective User Agents)" and aims to deliver "a targeted, deployable solution in a short timeframe (18-24 months) as needed by the industry". So far the WG has produced the above mentioned RFCs and drafts for the CDNI framework, the CDNI requirements as well as specification drafts for the used protocol and logging mechanisms [4].

11 Use Case for Collaboration: P2P

Recall, P2P systems are self-organizing systems of autonomous entities, called peers, that cooperate for common goals. These common goals range from sharing of resources, e.g., music and video files, processing power, or storage space [212] to collaborative routing as in Skype and P2P-TV. A fundamental characteristic of these systems is their distributed topologies and resources.

Advantages of P2P systems include elimination of bottlenecks and single points-of-failure within the system, increased processing power, high availability/redundancy, and little or no dependence on any particular central entity. However, P2P systems are plagued by some fundamental issues, such as overlay/underlay topological and routing mismatch [202], inefficiencies in locating and retrieving resources, and scalability and performance issues caused by uncontrolled traffic swamps [212].

Several of these drawbacks can be addressed by collaboration between the P2P overlay and the Internet routing underlay. To overcome these limits each ISP can offer the “oracle” service as introduced in Section 10.1 to the P2P users which explicitly helps P2P users to choose “good” neighbors. The P2P user can supply its ISP’s oracle with a list of possible P2P neighbors, during bootstrapping and/or content exchange. The ISP’s oracle then returns a ranked list to the querying user, according to its preference (e.g., AS-hop distance) and knowledge of the ISP topology and traffic volume, while at the same time keeping the interest of the P2P user in mind. We show that in principle, P2P systems as well as the ISPs profit from the use of the oracle even when only considering the AS-distance for ranking nodes [17], because the overlay topology is now localized and respects the underlying Internet topology, and the P2P user profits from the ISP’s knowledge.

We show, in Section 11.3.3, relying on graph based simulations and measured Internet topologies, that the resulting P2P overlays maintain their graph properties like small diameter, small mean path length and node degree, but the densely connected subgraphs are now local to the ISPs. To study the impact of biased neighbor selection on a real P2P network that implements its own routing, we run extensive simulations of the Gnutella protocol in Section 11.4. These experiments help us to evaluate the effect of churn in P2P systems, and to study the impact of oracle on scalability and traffic content localization. We find that the Gnutella topologies maintain their graph properties, the ISP now has the ability to influence the overlay topology, and the scalability and network performance of Gnutella improves considerably. Then, in Section 11.5, we show that a modified version of Gnutella when used in a testbed can indeed take advantage of the oracle service. Moreover, using an oracle explicitly avoids caching content, contrary to [205], thus absolving ISPs of potential legal issues. While there are alternate proposals to localize P2P traffic, e.g., [37][114][69], the oracle proposal [17] is simple, scalable, applicable to all overlays, and promotes active collaboration between ISPs and P2P systems.

In addition, we study the impact of different ISP/P2P topologies and user behavior patterns on end-user performance and extend the ISP’s oracle to also consider last-hop bandwidth of P2P users while ranking possible neighbors. To this end we (i) discuss the advantages of bandwidth-based neighbor selection over latency- and geography-based options, (ii) design of different ISP and P2P topologies, (iii) design of different user behavioral patterns, namely, content availability, churn, and query patterns, (iv) extensive experimental studies to determine the impact of different topologies and behavioral patterns on end-user experience.

Our findings show that in contrast to the unmodified P2P system, the ISP-aided localized P2P system shows consistent improvements in the observed end-user experience, measured in terms of content download times, network locality of query responses and desired content, and quality of query responses. A significantly large portion of P2P traffic remains local to the ISP network, and ISPs notice a substantial reduction in overall P2P traffic. This can lead to immense cost savings for the ISPs [40]. The oracle consistently shows performance gains even across different topologies under a broad range of user behavior scenarios.

11.1 Realizing an Oracle Service:

It may seem rather challenging to build such an oracle in a scalable manner, but much more complicated services, e.g., DNS, already exist. The oracle service can be realized as a set of replicated servers within each ISP that can be queried using a UDP-based protocol or run as a Web service. It can rely on a semi-static database with the ISP's prefix and topology information. Updating this information should not impose any major overhead on the ISP.

While the oracle service is not yet offered by the ISPs, P2P nodes have the chance of using a simple service to gain some of the oracle benefits already using the "pWhois" service [187]. This service is capable of satisfying 100,000 queries using standard PC-hardware [58] in less than one minute. It enables the P2P node to retrieve information about possible P2P neighbors such as the AS and some geographic information. This information can then be used by the P2P node to bias its neighbor selection. But purely using the "pWhois" service only helps the P2P system. It does not enable the ISP to express its preference and therefore does not enable cooperation.

11.2 Using Bandwidth to Select Neighbors

We propose to extend the ISP-hosted oracle [17] to go beyond using only network proximity (nodes within the AS, AS-hop distance) to keep traffic within its network. It should also use last-hop bandwidth of P2P users within its network to help querying P2P users select high-performance neighbors. This is possible as the ISP knows its customers' last-hop bandwidth and hence does not have to measure it, yet this metric is difficult and traffic-intensive to reverse engineer accurately [197][186] for the P2P users.

Moreover, this has advantages over neighbor selection using latency measurements [69] as network latency can change quickly [237]. Also, latency is difficult to predict reliably [134][230], especially in the face of newer breed of Internet applications characterized by large data content and high churn. While we agree that similar arguments hold to some degree regarding estimating available last-hop bandwidth [186] as well, we argue that utilizing the ISP knowledge via the oracle helps to (i) improve accuracy (ii) mitigate ISP's concerns about traffic management and respect for routing policies (iii) reduce the excessive traffic swarm [191] that results from frequent pinging of the network to deduce latency and/or available bandwidth. Besides, latency between Internet hosts is dominated by the cable/DSL bandwidths at the last-mile connections [60], thus making neighbor selection based on last-hop bandwidths a good option.

We show that keeping the P2P traffic localized allows users to benefit from the significant geographic and interest-based clustering [133] for audio/video P2P content. One may argue in favor of bypassing the ISP's oracle service to utilize geolocalization techniques [175] to choose neighbors. However, we caution that even the best such techniques can identify a node to within 22 miles of its actual position [230], hence making differentiation of nodes even within the same city difficult. On the other hand, the ISP being aware of the minute details of its PoP-level backbone topology, can easily use this information to better rank the querying node's neighbors, even within the same city. Indeed, oracles from multiple ISPs can collaborate to build a global coordinate system [15]. We thus believe that ISP-aided P2P neighbor selection is a win-win solution for ISPs as well as P2P systems.

11.3 Benefit of the Oracle from P2P and ISP Point of View

In this section, we first propose metrics for evaluating the effectiveness of the idea of using an oracle which can also be used to characterize overlay-underlay graphs in general. Then we describe how we derive representative topologies for our simulations from the Internet AS topology.

11.3.1 Metrics

As a basic model for our investigations, we model the AS-graph as a complete bi-directed graph $G = (V, E)$ with a cost function $c : E \to \mathbb{R}^+$ associated with the edges. Every node represents an AS, and for every pair (u, v), let $c(u, v)$ denote the overall cost of routing a message from AS u to AS v (which depends on the routing policies of the ASes such a message may traverse).

Given a set of peers[5] P, let $AS : P \to V$ define how the peers are mapped to the ASes and $b : P \to \mathbb{R}^+$ denotes the bandwidth of the Internet connections of the peers. The overlay network formed by the peers is given as a directed graph $H = (P, F)$ in which every edge $(p, q) \in F$ has a cost of $c(AS(p), AS(q))$. The graph H can be characterized using several metrics.

Degree:
The *degree* of a peer is defined as the number of its outgoing connections. Ideally, every peer should have a large number of connections to other peers within its AS so as to favor communication within the AS, while connections to other ASes should be limited to avoid high communication costs and high update costs as peers enter/leave the network.

Hop count diameter:
Another parameter that should be small is the hop count diameter of the overlay graph H. The hop count diameter D of H is the maximum over all pairs $p, q \in P$ of the minimum length of a path (in terms of number of edges) from p to q in H. It is

[5]In this section a peer refers to a node of the P2P network and not to a BGP peer.

well-known that any graph of n nodes and degree d has a hop count diameter of at least $\log_{d-1} n$, and that dynamic overlay networks such as variants of the de Bruijn graph [165] can get very close to this lower bound, a very nice property. However, even though the hop count diameter may be small, the AS diameter (i.e., the distance between two P2P nodes when taking the underlying AS-graph G with cost function c into account) can be very large.

AS diameter:
The AS diameter of H is defined as the maximum over all pairs $p, q \in P$ of the minimum cost of a path from p to q in P, where the cost of a path is defined as the sum of the cost of its edges. Ideally, we would like both the hop count diameter and the AS diameter to be as small as possible. Research in this direction was pioneered by Plaxton et al. [181], and the (theoretically) best construction today is the LAND overlay network [6].

Surprisingly, the best AS diameter achievable when avoiding many P2P connections to other ASes can be better than the best AS diameter achievable when all P2P connections go to other ASes. Consider the simple scenario in which the cost of a P2P edge within the same AS is 0 and that between two different ASes is 1. Let the maximum degree of a peer be d. In scenario 1, we require all edges of a peer to leave its AS, and in scenario 2, we only allow one edge of a peer to leave its AS. In scenario 1, the best possible AS diameter is $\log_{d-1} n$ (see our comments above). However, in scenario 2 one can achieve an AS diameter of just $\log_{d-2}(n/(d-1))$. For this, organize the peers into cliques of size $d-1$ within the ASes (we assume that the number of peers in each AS is a multiple of $d-1$). We can then view each clique as a node of degree $d-1$. It is possible to connect these nodes with a graph of diameter close to $\log_{d-2}(n/(d-1))$, giving the result above.

Flow conductance:
Having a small hop count diameter and AS diameter is not enough to ensure high network performance. A tree, for example, can have very low hop count and AS diameter. Yet, it is certainly not a good P2P network, since one single faulty peer is sufficient to cut the network in half. Ideally, we would like to have a network that is well-connected so that it can withstand many faults and can route traffic with low congestion. A standard measure for this has been the expansion of a network. However, it seems that the expansion of a network cannot be approximated well. The best known algorithm can only guarantee an approximation ratio of $O(\sqrt{\log n})$ [27]. Therefore, we propose an alternative measure here that we call the *flow conductance* of a network (which is related to the flow number proposed in [122]).

Consider a directed network $G = (V, E)$ with edge bandwidths $b : E \to \mathbb{R}^+$. If $E(v)$ is the set of edges leaving v then for every node $v \in V$, let $b(v) = \sum_{e \in E(v)} b(e)$. Furthermore, for any subset $U \subseteq V$ let $b(U) = \sum_{v \in U} b(v)$. Next we consider the concurrent multicommodity flow problem M_0 with demands $d_{v,w} = b(v) \cdot b(w)/b(V)$ for every pair v, w of nodes. That is, we consider the heavy-traffic scenario in which each node aims at injecting a flow into the system that is equal to its edge bandwidth, and the destinations of the flows are weighted according to their bandwidth. The *flow conductance C* measures how well the network can handle this scenario, or more formally, the flow conductance is equal to the inverse of the largest value of λ so that there is a feasible multicommodity flow solution for the demands $\lambda d_{v,w}$ in G. It is easy to show that for any network G, $0 \leq \lambda \leq 1$, and the larger λ is, the better is the network. As an example, for uniform link bandwidths the flow conductance of the $n \times n$-mesh is $\Theta(1/n)$ and the flow conductance of the hypercube of dimension n is $\Theta(1/\log n)$.

Interestingly, one can significantly lower the number of inter-AS edges without losing much on the flow conductance. Suppose we have m peers with bandwidth b that can have a maximum degree of d. Consider a class of networks $G(n)$ of degree d and size n with monotonically increasing flow conductance $C(n)$. Connecting the m peers by $G(m)$ gives a network with flow conductance $C(m)$. Suppose now that every peer can establish only one inter-AS edge with bandwidth $b/2$, and the remaining bandwidth can be used for intra-AS edges. In this case, let us organize the peers into cliques of size $d-1$ within the ASes (we assumed that the number of peers in each AS is a multiple of $d-1$) and interconnect the cliques so that they form $G(m/(d-1))$. Then it is not difficult to see that the resulting network has a flow conductance of $2C(m/(d-1))$. Hence, compared to arbitrary networks we lose a factor of at most two.

Summary:
We propose measures that are useful for P2P systems and our results demonstrate that it is possible to have a highly local topology with an AS diameter and a flow conductance that is comparable to the best non-local topologies. Hence, worst-case communication scenarios can be handled by local topologies (i.e., topologies with many intra-AS connections) essentially as well as by non-local topologies. In addition, we expect local topologies to be far better cost-wise for serving P2P traffic in practice than non-local topologies, which we aim to validate through experiments.

11.3.2 Simulation Topologies

The simulation results can be heavily influenced by the topologies used. Hence, we make the basis for our simulations the current AS topology of the Internet [162, 149], as it can be derived from the BGP routing information. We use BGP data from more than $1,300$ BGP observation points including those provided by RIPE NCC, Routeviews, GEANT, and Abilene. This includes data from more than 700 ASes as on November 13, 2005. Our dataset contains routes with $4,730,222$ different AS-paths between $3,271,351$ different AS-pairs. We derive an AS-level topology from the AS-paths. If two ASes are next to each other on a path, we assume that they have an agreement to exchange data and are therefore neighbors. We are able to identify $58,903$ such edges. We identify level-1 providers by starting with a small list of providers that are known to be level-1. An AS is added to the list of level-1 providers if the resulting AS-subgraph between level-1 providers is complete, that is, we derive the AS-subgraph to be the largest clique of ASes including our seed ASes. This results in the following 10 ASes being referred to as level-1 providers: 174, 209, 701, 1239, 2914, 3356, 3549, 3561, 5511, 7018. While this list may not be complete, all found ASes are well-known level-1 providers. There are $7,994$ ASes that are neighbors of a `level-1` provider, which we refer to as `level-2`. All other $13,174$ ASes are grouped together into the class `level-3`. We thus identify $21,178$ ASes in all.

As it is not known how many P2P nodes are in each AS, and we may want to study smaller subsets to be able to compute the complex graph properties in reasonable time, we randomly subsample the AS-topology by keeping all level-1 ASes and their interconnections, and selecting a fraction of the level-2 and level-3 ASes while keeping their proportion the same as in the original data. Hereby, we first select the level-2 ASes and keep their interconnections. Only then do we select the level-3 ASes from among the ASes that are reachable in our subgraph.

Most level-1 ASes traditionally are expected to serve more customers than level-2 and level-3 ASes [141, 42]. At the same time there are more level-3 than level-2 than level-1 ASes. Thus we distribute the P2P clients among the ASes in the following ad-hoc manner: a P2P node has equal probability to pick an AS from each level. This results in a $1/3 : 1/3 : 1/3$ split of the nodes among the AS levels. This way a level-1 AS serves many more P2P nodes than a level-3 AS. All the topologies used in our experiments have been derived in this manner by randomly subsampling the AS topology derived from the BGP table dumps. Indeed, sensitivity analysis of our results show that if we move more peers to level-2, level-3 ASes the results improve even more.

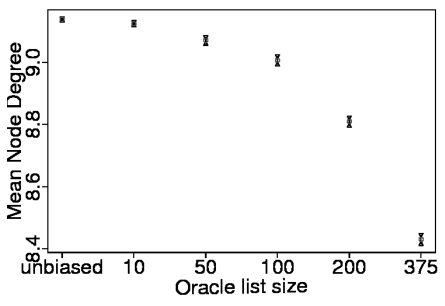
11.3.3 Overlay / Underlay Graph Properties

In this section, we first evaluate how the use of the oracle changes the graph properties of the P2P overlay topology. Later, in Sections 11.4 and 11.5 we explore the interactions of the two routing systems, the impact of churn on the topology, and the benefits of the oracle for satisfying queries. For this purpose we in this section use a general graph simulator as it allows us to explore large topologies. Namely, we rely on a simulation environment, the Subjects environment [199], that is very light-weight, such that we can run experiments on large topologies with many P2P nodes.

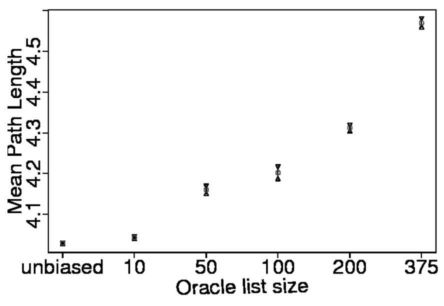
The Subjects environment is developed for the design of highly robust distributed systems and provides us with support for operations on general overlay graphs. It is based on C++ and consists of three basic types of entities: subjects, objects, and relay points. Subjects are the base class for processes (that are used to emulate nodes in the overlay network), objects are the base class for messages exchanged between subjects, and relay points are used by the subjects in order to establish connections to each other so that objects can be exchanged. In our experiments, the Internet class spawns multiple AS classes, each of which then spawns a number of overlay node classes. These nodes then establish peering connections with each other by exchanging messages (objects), and the relay points serve as an abstraction of network ports. The way these entities are set up ensures that subjects have a firm control on who can send information to them so that the consent and control principle can be strictly enforced.

For our evaluation we consider five graphs, each with 300 ASes and $4,372$ P2P nodes, which results in an average of 14.6 nodes per AS. Each graph consists of 4 level-1 ASes, 100 level-2 ASes and 196 level-3 ASes. We place 375 nodes within each level-1 AS, 15 nodes within each level-2 AS, and 7 nodes within each level-3 AS. Increasing the number of nodes in the level-2, level-3 ASes only helps our case.

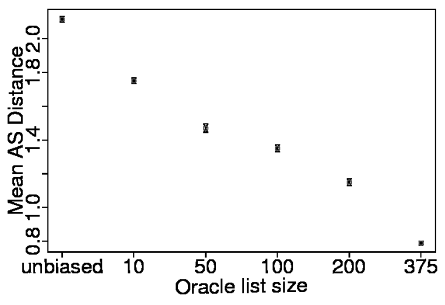
We establish P2P neighbor relationships by randomly picking one of the P2P nodes and let it establish a neighborship either



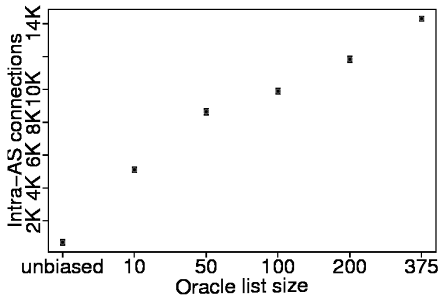
(a) P2P node degree



(b) Overlay path length



(c) Underlay AS distance



(d) Intra-AS P2P connections

Figure 20: Comparison of metrics with increasing size of Oracle list (Error plots).

unbiased: to a single randomly chosen P2P node or
biased: to one from a list of candidates.

The unbiased case corresponds to a P2P protocol with arbitrary neighbor selection, while the biased case corresponds to a P2P node giving a list of potential neighbors to the oracle, and the oracle helping it pick an optimal neighbor. We simulate the simplest of such oracles where it either chooses a neighbor within the querying node's AS if such a one is available, or a node from the nearest AS (considering AS hop distance). We experiment with different sizes of the oracle's choice list.

We experimented with establishing from 1000 up to $40,000$ neighbor relationships in total. Given that for random graphs, the threshold for the number of edges to ensure connectivity is $\log n/2$ times the number n of nodes, it is not surprising that we need roughly $18,000$ edges to ensure that the simulated graph is connected. Increasing the number of edges beyond this number does not change the graph properties noticeably. Accordingly, we concentrate on results for $20,000$ peerings.

We run four experiments for each of the five AS graphs where the oracle is used for each neighbor relationship with candidate lists of length 1, 10, 50, 100, 200, 375, resulting in 120 experiments. Note that a list length of 1 corresponds to the unbiased case. The results we obtained are as follows.

Structural properties:
First, we check whether the overlay graphs remain connected using biased neighbor selection. In principle it is possible that due to a heavy bias, the graph disintegrates into disconnected components which are themselves well connected. We experimentally verify that all resulting graphs remain connected, thereby not impacting the reachability of the overlay graph.

The next question is if the mean degree of the P2P nodes changes. We find that the mean degree value of 9.138 of an unbiased graph changes to 8.8 in biased graphs with list size 200, see Figure 20(a). The small change in node degree implies that we do not affect the structural properties of the overlay graph seriously.

One may expect that our biased neighborhood selection increases the diameter and mean path length, as it prefers "closeby" neighbors. Yet, in all experiments the hop count diameter of the overlay graph stays at 7 or 8 hops and the AS diameter of the underlying AS graph stays at 5 hops. Neither does the average path length in the overlay graph increase significantly, see Figure 20(b). Therefore we can conclude that the biased neighborhood selection does not negatively impact the structural properties of the overlay graph.

Locality in topology:
We find that locality in overlays improves significantly as captured by the average AS-distance of P2P neighbors. Figure 20(c) shows how the AS-distance improves with the ability of the P2P node to choose a nearby neighbor. A lower AS-distance should correspond to lower latency. This is also reflected in the number of P2P neighbor connections that stay within each of the ASes, see Figure 20(d). Without consulting the oracle, only 4% of the edges are local to any of the ASes. The use of the oracle increases locality by a factor of 7 from 697 to 5088 (in a total of $20,000$ peerings), even with a rather short candidate list of length 10. With a candidate list of length 200, more than half of the edges, 59%, stay within the AS. We find that the effects are even more pronounced for smaller networks. This demonstrates how much the oracle increases the ability of the AS to keep traffic within its network and with a refined oracle to better manage the P2P traffic. These results also indicate the benefit to the user, as traffic within the AS is less likely to encounter network bottlenecks than inter-AS traffic.

Flow conductance:
The remaining question is if the network maintains its ability to route traffic with low congestion. Since the run time requirements of our algorithm for computing a lower bound for the flow conductance of a graph is $O(n^4)$, we can currently only estimate the flow conductance for small graphs[6]. Being able to calculate the conductance of smaller graphs only is not a big problem as in case of Gnutella [91], we can calculate the conductance of the graph of ultrapeers, which is naturally much smaller than the entire Gnutella connectivity graph. We construct unbiased as well as biased graphs with 10 nodes and 21 edges, respectively 18 nodes and 51 edges. Both graphs are generated on a topology with 6 ASes.

The expected flow conductance of the unbiased graphs is 0.505 for the 10 node

[6]Meanwhile, we have found a way to reduce the complexity to $O(n^2 logn)$ and work on computing the conductance of larger graphs is continuing.

graph and 0.533 for the 18 node graph (see Section 11.3). We experimentally verify that both unbiased graphs support a conductance of at least 0.5. Also, we find that the penalty for the two biased graphs is less than a factor of 2. The 10 node biased graph supports a flow conductance of at least 0.3, and the 18 node graph, of at least 0.25. We furthermore observe that subgraphs of the biased graphs support a higher flow conductance which indicates that the connectivity within the ASes is good. This will likely result in a performance boost if the desired content can be located within the proximity of the interested user. The locality of biased graphs increases to 50% (for 10 nodes), respectively 80% (for 18 nodes) compared to 20% in the unbiased graphs.

11.4 Simulations with an Actual P2P System

In the previous section, we have seen that the results of biased neighbor selection on the graph properties of a generalized overlay network as well as its correlation to the underlay graph are promising. We now explore how a real P2P file sharing system benefits from using the oracle using a packet level network simulator [211]. For this purpose, we choose Gnutella, an unstructured P2P file sharing system. In the following we first give an overview of the Gnutella protocol, then discuss how we realize it within the simulation framework, and then discuss the simulation setup and our results.

11.4.1 Gnutella and SSFNet

Gnutella [91] is a popular file-sharing network with about 2 million users [189, 215]. Moreover it is an open-source system, which has attracted a healthy interest from researchers, e.g., [215, 88, 214]. The Gnutella network is comprised of agents called servents, who can initiate as well as serve requests for resources. When launched, a servent searches for other peers to connect to by sending Hello-like `Ping` messages. The `Ping` messages are answered by `Pong` messages, which contain address and shared resource information. The search queries are flooded within the Gnutella network using `Query` messages, and answered by `QueryHit` messages. To limit flooding Gnutella uses TTL (time to live) and message IDs. Each answer message (`QueryHit`/`Pong`) traverses the reverse path of the corresponding trigger message. While the negotiation traffic is carried within the set of connected Gnutella nodes, the actual data exchange of resources takes place outside the Gnutella network, using the HTTP protocol. Due to scalability problems, new versions of Gnutella take advantage of a hierarchical design in which some servents are elevated to ultrapeers, while others become leaf nodes. Each leaf node connects to a small number of ultrapeers, while each ultrapeer maintains a large number of neighbors, both ultrapeers and leafs. To further improve performance and to discourage abuse, the `Ping`/`Pong` protocol underwent semantic changes. Answers to `Ping`s are cached (Pong caching) and too frequent `Ping`s or repeated `Query`s may cause termination of connection.

We have coded the Gnutella protocol within the packet level network simulator SSFNet [16]. The Scalable Simulation Framework (SSF) [211] is an open standard for simulating large and complex networks. Written in Java, it supports discrete-event simulations. SSF Network models (SSFNet) are Java models of different network entities, built to achieve realistic multi-protocol, multi-domain Internet modeling and simulation at and above the IP packet level of detail. These entities include Internet protocols like IP, TCP, UDP, BGP and OSPF, network elements like hosts, routers, links, and LANs, and their various support classes. The network topologies are defined using the Domain Modeling Language (DML), and the SSFNet class instances autonomously configure and instantiate themselves by querying these DML configuration files. The coding for the lower layers of the IP stack is thus provided by SSFNet, while we implement the Gnutella protocol as an SSFNet application [16].

We modify the neighbor selection procedure of Gnutella to take advantage of the oracle [218]. Normally, when a Gnutella node connects to the network, it gets a list of popular Gnutella node addresses in its Hostcache [90], which is a locally maintained Gnutella hosts list, typically containing a few hundred IP addresses. The node chooses a random subset of the Hostcache, and initiates Gnutella peerings with these selected nodes. We modify this procedure so that the Gnutella node sends the contents of its Hostcache (list of IP addresses) to the oracle, which then picks a node within the querying node's AS if it exists, or a random node otherwise. The node then establishes a Gnutella peering with this oracle-preferred node. This way, we influence the neighborhood selection of Gnutella network, to choose a peer within the AS if it exists. Moreover when a Gnutella node receives query results for its search requests, it again consults the oracle to select the nearest node from whom it then downloads the file content.

11.4.2 Simulation Setup

The topologies are derived using the methodology explained in Section 11.3.2. The network consists of a total of 25 ASes and 1000 nodes. More specifically it consists of 1 level-1 AS, 8 level-2 ASes and 16 level-3 ASes. We place 360 nodes within the level-1 AS, 40 nodes within each level-2 AS, and 20 nodes within each level-3 AS. Within each AS, all the nodes are connected in a star topology to an intra-AS router. Each node in level-1 AS has a 1 Gbit network interface, each node in level-2 AS has a 100 Mbit network interface, while each node in level-3 AS has a 10 Mbit network interface. The links between level-l and level-2 ASes have a delay of 2 ms, while the links between level-2 and level-3 ASes have a delay of 10 ms. Each AS has 2 routers, one for the intra-AS node connections, and one for the inter-AS connections between different ASes. Thus, we have a topology with 25 ASes, 50 routers and 1000 nodes running the Gnutella protocol.

Each leaf node can have between 2 to 4 connections to ultrapeers, while each ultrapeer initiates at least 10 connections to other Gnutella nodes itself, and stops accepting incoming connections from other nodes, once it is connected to 45 nodes, be they leafs or ultrapeers. Each node shares between 0 and 100 files, uniformly distributed.

To take churn in P2P systems into account, each node remains online for a minimum of 1 and a maximum of 1500 seconds. Once a node goes off-line, it may become online again after a time period between 1 to 300 seconds. For a start, we take these time periods as uniformly distributed but we are in the process of migrating to more precise distributions, as recently revealed in [214]. Furthermore, a leaf node must be online for at least 600 seconds before it can serve as an ultrapeer. At any given point of time in our simulations, we find that $20 - 25\%$ nodes are off-line[7], and a quarter of the online nodes are functioning as ultrapeers.

We ran multiple simulations for arbitrary lengths of time and found that the startup phase of the simulation lasts for about 500 seconds. After 5000 seconds of simulation time, the summary statistics do not show significant changes. Therefore we run our simulations for 5000 seconds.

11.4.3 Results

We first analyze the Gnutella network graph according to the metrics explained in Section 11.3, followed by an evaluation of some Gnutella specific metrics like scalability of network, number of messages exchanged, localization of file content exchange and visualization of topology.

We run three different experiments on five different topology instances with roughly the same number of search queries and the following parameters for the Gnutella nodes:

- Cache size = 1000, without oracle
- Cache size = 100, with oracle for neighbor selection
- Cache size = 1000, with oracle for neighbor selection

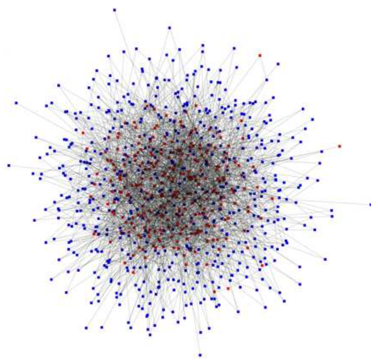
Note that in our implementation, each Gnutella node sends the contents of its Host-cache to the oracle, which ranks the list of IP addresses according to proximity from the querying node. In other words, the above three cases correspond to experiments with oracle list size of 1, 100, and 1000 respectively. The success rates of the search queries are similar.

To explore the influence of consulting the oracle on the network topology we visualize, in Figure 21 [218], the Gnutella overlay topology, for the unbiased case and the biased case with oracle list size 1000. At a particular instant in time, we sample the Gnutella overlay topology, display all the online nodes in the graph, and join two nodes with an edge if there exists a Gnutella peering between them at this point of time. Then, using the visualization library yWorks [235], we convert both the graphs into a structured hierarchical format. The resulting graph structures are displayed in Figure 21. We can easily observe that the Gnutella topology in the biased case is well correlated with the Internet AS topology, where the nodes within an AS form a dense cluster, with only a few connections going to nodes in other ASes. This is in stark contrast to the unbiased Gnutella graph, where no such property can be observed.

To analyze how churn influences the metrics such as node degree, path length, diameter and number of intra-AS peerings, we sample the Gnutella network 10 times during the simulation run, i.e., every 500 seconds. The results are shown in Figure 22. Multiple runs of the above experiments, using different world topologies yield similar results.

Graph connectivity:
We begin by checking whether the Gnutella network graph remains connected using

[7]This is more aggressive as compared to other studies, e.g., [145] which assume that only half the nodes churn.


(a) Unbiased Gnutella

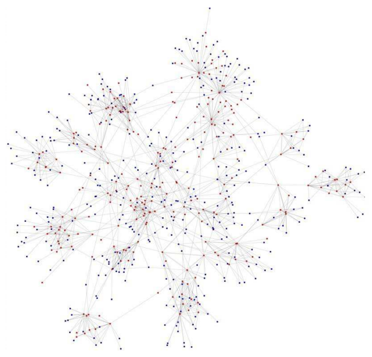
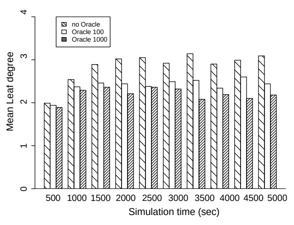
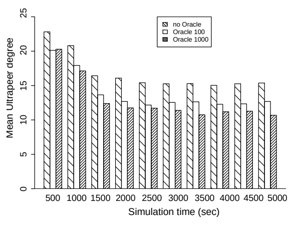

(b) Gnutella with Oracle

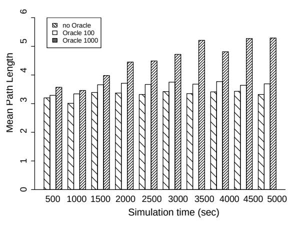
Figure 21: Visualization of Gnutella overlay topology



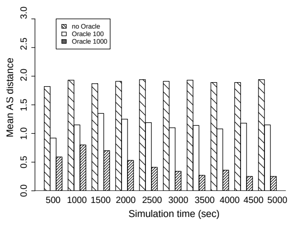
(a) Mean Leaf Node Degree



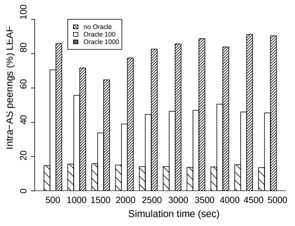
(b) Mean Ultrapeer Degree



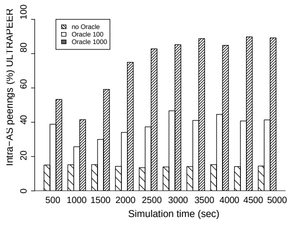
(c) Mean Path Length in Overlay



(d) Mean AS distance in Underlay



(e) Intra-AS peerings (%) for Leaf nodes



(f) Intra-AS peerings (%) for Ultrapeers

Figure 22: Metrics for Gnutella simulations

biased neighbor selection. We define the Gnutella network graph at a particular time instant as the graph formed by nodes that are online at that instant, where two nodes are connected by an edge if there exists a Gnutella connection between them at that instant. We experimentally verify that the Gnutella network remains connected at all 10 times where we sample the network, for all three cases. Hence, biased neighbor selection does not affect the connectivity of Gnutella network.

Mean Node Degree:

Since ultrapeers have a much larger node degree than leaf nodes, we show, in Figure 22(a) and (b), how the mean node degree changes over time in a barplot for all three cases separately for ultrapeers and leaf nodes. This enables us to check if a biased neighbor selection affects the structural properties of Gnutella adversely. We observe that the mean node degree for leafs decreases only slightly, across time, with a maximum decrease from 3.14 to 2.08 at 3500 seconds. The same is the case for ultrapeers, where the maximum decrease is from 15.29 to 10.75, again at 3500 seconds. In other words, despite biasing the neighbor selection via the oracle, the node degree for both leafs and ultrapeers stays within the expected range, and the network structure of Gnutella remains unchanged.

Graph diameter:
The diameter of the overlay graph, which is $5-7$ hops in the unbiased case, increases to $6-8$ hops with a oracle size of 100, only a nominal increase. Using an oracle with list size of 1000 results in a diameter between $7-12$ hops, with an average of 9.2. The AS diameter of the underlay graph remains is 4 hops in all cases.

Mean Overlay path length:
The average path length in the Gnutella overlay, shown in Figure 22(c), while registering an increase, does not change significantly. The maximum increase occurs at 3500 seconds, from 3.35 in the unbiased case to 5.21 hops in the biased case with oracle list size of 1000.

Mean AS distance:
The benefits of using an oracle for biasing the neighborhood in Gnutella are visible in Figure 22(d), which shows the average AS distance (in the underlay) between any two connected Gnutella nodes. The AS distance is obtained as follows. We map each Gnutella node's IP address to its parent AS, and for each overlay edge, we find the network distance in AS hops between the two end-nodes. We observe that the least amount of decrease in the average AS distance occurs from 1.93 to 0.8 at 1000 seconds, and the maximum decrease from 1.94 to 0.25 happens at 5000 seconds. Given that the AS diameter remains constant at 4 hops, the average decrease of 1.45 in the AS distance is significant. Besides, as the average AS distance in the case of oracle list size of 1000 is 0.45, a value less than 1, it implies that most of the Gnutella peerings are indeed within the ASes, i.e., they are not crossing AS boundaries. This can be a major relief for ISPs, as they do not incur any additional cost for traffic within their domains. Also traffic that does not leave the network is easier to manage. Moreover, P2P traffic will not encounter inter-ISP bottlenecks.

Intra-AS P2P connections:
The above observations on AS distance are even better understood from the plots in Figure 22(e) and (f), where we show the total number of intra-AS P2P connections in the Gnutella network as a percentage of the total number of intra- and inter-AS P2P connections, for both leafs and ultrapeers.

In Figure 22(e), we observe that in the case of leaf nodes, taking the average over the 10 time points, the percentage of intra-AS P2P connections increases from 14.6% in unbiased case to 47.88% in the case of oracle with list size 100. For oracle with list size 1000, we note an average of 82.22% intra-AS P2P connections.

In Figure 22(f), we observe similar results for ultrapeers. The percentage of intra-AS P2P connections increases from an average value of 14.54% in the unbiased case to 38.04% in the case of oracle with list size 100, and further to 74.95% in case of oracle with list size 1000.

The percentage increase in intra-AS P2P connections is larger for leaf nodes as compared to ultrapeers, a welcome development. One needs a certain number of inter-AS connections, to maintain network connectivity and to be able to search for file content that may not be available within an AS. However, as leaf nodes typically have poor connectivity to the Internet, and have lower uptimes, it is reasonable to have leaf nodes keep most of their peerings within their AS, while allowing the ultrapeers to have slightly more connections outside their ASes.

Overall, we observe that the results for the metrics comparison in Gnutella simulations are in conformity with the graph-based simulation results in Section 11.3.3.

Scalability of Gnutella:
In order to quantify the impact of biased neighborhood selection on the scalability of the Gnutella network, we measure the number of Gnutella messages generated in the entire network, for all the three cases. The negotiation traffic in many P2P systems like Gnutella represents a large portion of the total P2P traffic [88]. In Table 3, we show the number of each type of Gnutella message (`Ping`, `Pong`, `Query` and `QueryHit`) generated during the entire simulation run. Note that the number of unique messages generated is about the same in all the three cases. However, when a `Ping` or `Query` is generated by a node, and flooded to its `n` neighbors, the message is counted `n` times. Hence, the table shows the total number of messages exchanged between Gnutella nodes.

As we can observe, the number of `Ping` messages decreases from 7.6 million in the unbiased case to 4 million in the case of oracle with list size 1000. Even more significant is the reduction of `Pong` messages, from 75.5 million to 39 million messages. The `Query` and `QueryHit` messages also register similar improvements. This reduction of `Ping/Pong` messages by a factor of 2, and search queries by a factor of almost 3 proves that the scalability of Gnutella network improves significantly with biased neighborhood selection.

The reason for this reduction in message volume is as follows. Even though the node degrees are largely unchanged, the oracle helps in building an efficient overlay topology. As the nodes form a dense cluster within an AS with very few inter-AS connections, caching of messages ensures that messages are flooded within sub-networks very efficiently, by traversing lesser overlay hops, which is reflected in Table 3. Thus information is propagated with lesser message hops, lower delays and reduced network overhead.

Localization of content exchange:
The negotiation traffic traverses within the set of connected Gnutella nodes, but the actual file content exchange happens outside the Gnutella network, using the standard HTTP protocol. When a Gnutella node gets multiple `QueryHits` for its search query,

Gnutella Message Type	**Unbiased Gnutella**	**Biased, cache 100**	**Biased, cache 1000**
Ping	7.6M	6.1M	4.0M
Pong	75.5M	59.0M	39.1M
Query	6.3M	4.0M	2.3M
QueryHit	3.5M	2.9M	1.9M

Table 3: Number of exchanged Gnutella message types

it chooses a node randomly and initiates an HTTP session with it to download the desired file content. Since the file content is often bulky, it is prudent to localize this traffic as well, as it relates directly to user experience. In the above experiments, we use the oracle to bias only the neighborhood selection. In other words, when a node comes online, it consults the oracle and sends connection requests to an oracle-recommended node selected from its Hostcache. However, while choosing a node from the `QueryHits`, it so far did not consult the oracle. We now analyze how much of the file content exchange remains local in this case and how much one can gain if one consults the oracle again at this stage.

We observe that the intra-AS file exchange, which is 6.5% in the unbiased case, improves slightly to 7.3% in case of oracle with list size 100, and to 10.02% in case of oracle with list size 1000.

We then further modify the neighborhood selection, so that a node consults the oracle again at the file-exchange stage, with the list of nodes from whom it gets the `QueryHits`. After this change, we notice that 40.57% of the file transfers now occur within an AS. In other words, 34% of file content, which is otherwise available at a node within the querying node's AS, was previously downloaded from a node outside the querying node's AS.

This leads us to conclude that consulting the oracle for neighborhood selection, during bootstrapping stage as well as file-exchange stage, leads to significant increase in localization of P2P traffic.

11.5 Testlab Experiments

After extensive simulations on general overlay graphs and Gnutella system, we now confirm these results by modifying P2P clients, namely Gnutella, to take advantage of the oracle service in a controlled setting, a Testlab.

Using 5 routers, 6 switches, and 15 computers, we configure four different 5-AS topologies: ring, star, tree and random mesh. Each router is connected to 3 machines, and each machine runs 3 instances of Gnutella software, where one is an ultrapeer and the other two are leaf nodes. Thus, we have a network of 45 Gnutella nodes, each running the GTK-Gnutella software [98]. A router is taken as an abstraction of an AS boundary.

We modify the source code of the Gnutella nodes, so that when a node wishes to join the network, it sends the contents of its Hostcache to the oracle. The Hostcache of each node is filled with a random subset of the network nodes' IP addresses. The oracle is a central machine accessible to all Gnutella nodes, and running the oracle's neighbor selection algorithm. When it gets a list of IP addresses from a node, it ranks the list according to AS hops distance. Hence, the Gnutella node joins another node within its AS if such a node is present in its Hostcache, else it joins a node from the nearest AS.

We experiment with two schemes of file distribution. In the uniform scheme, each node shares 6 files each. In the variable scheme, each ultrapeer shares 12 files, half the leaf nodes shares 6 files each, and the remaining leaf nodes share no content. We thus have 270 unique files with real content.

We run two sets of experiments: unbiased Gnutella and Gnutella using oracle. We generate 45 unique search strings, one for each node, and allow each node to flood its search query in the network. Each node searches for the same query string in both the experiments. We then calculate the total number of `Query` and `QueryHit` messages exchanged in the network and analyze whether biased neighbor selection leads to any unsuccessful content search which was otherwise successful in unbiased Gnutella. We experimentally verify that all `Query`s that are satisfied in unbiased Gnutella network, are also satisfied in the biased Gnutella network. We find, as predicted by the simulations, that with biased neighbor selection, the number of `Query` and `QueryHit` messages decreases (60% reduction in `Query`, 12% reduction in `QueryHit`) and that the messages tend to stay within the ASes.

11.6 Benefit of the Oracle from a Users Point of View

To be able to study user behavior and its impact on P2P system performance, we again use the Java-based, discrete-event network simulator SSFNet [211][143]. Recall, that we use an application layer P2P protocol [16] similar to Gnutella, which relies on flooding connectivity pings and search queries to locate content [92]. Each node floods query search messages to all its connected peers, which iteratively forward it to their connected peers, until the query reaches a peer that possesses the searched content. Such a peer then sends a query response to the originating peer, retracing the overlay path traversed by the corresponding query. Each message carries a TTL (time to live) and message ID tag. To improve scalability, nodes are classified in a two-level hierarchy, with high-performance ultrapeer nodes maintaining the overlay structure by connecting with each other and forwarding only the relevant messages to a small number of shielded leaf nodes. File exchange is carried out between the peers directly using HTTP, similar to most other P2P file sharing systems. Recent developments like efficient querying, Gnutella2, and support for mobile users have helped keep the number of Gnutella users around $1 - 2$ million [240].

11.6.1 Topology Models

For the next set of simulations we extend the set of considered topologies: Germany, USA, World1, World2 and World3. Each topology consists of 700 P2P nodes distributed within various ASes (recall the memory limitations of packet-level simulators [143][37]). We take a subset of the AS topology of Germany as published in [105], and distribute the P2P nodes in each of the 12 ISPs according to the actual number of DSL customers of these major ISPs [220]. For USA, we model several regional providers at most of the major cities, and connect them with peering links [143][209], distributing the P2P nodes in the 25 ASes according to the ratio of the population of the cities. To model the World topologies, we design inter-AS connections as derived from BGP routing information in [162], and distribute P2P nodes based on results in [162][141]. Each World topology has 1 level-1 AS, 5 level-2 ASes, and 10 level-3 ASes, hence resulting in a 16-AS network. The number of P2P nodes assigned to (level-1,level-2,level-3) ASes are as follows: World1: $10, 46, 46$; World2: $355, 23, 23$; World3: $50, 46, 42$. We thus have 3 different topologies, and for the World topology, 3 different ways of distributing P2P nodes within the ASes.

Bearing in mind the memory limitations and that it is fundamentally difficult to simulate the Internet [79], we model the topologies within SSFNet as follows. Each AS has 2 routers, one for intra-AS node connections, and one for the inter-AS connections between ASes. Within each AS, all the nodes are connected in a star topology to the intra-AS router. The nodes have network interfaces representing typical last-hop DSL and cable modem bandwidths, ranging from 1 Mbps to 16 Mbps, and top-tier ASes have a larger proportion of higher bandwidth customers than the lower-tier ASes [197][143][220][141]. The links between level-1 and level-2 ASes have a delay of $4 - 6$ ms, while links between level-2 and level-3 ASes have a delay of $14 - 18$ ms [143][241].

11.6.2 User Behavior Models

While we have implemented a specific protocol in SSFNet, our goal is to perform experiments that represent a large section of P2P systems in use today. Studies [214][101] have shown that user behavior is largely invariant across P2P systems, both structured and unstructured. This means that factors like session lengths, content availability (free-riding), query patterns and search strings are similar across different P2P systems.

We note that user behavioral patterns are in constant transition, although the broad characteristics across different systems are comparable. Hence, we use different distributions to simulate the behavioral patterns, some very close to observed behavior, e.g., Weibull distributions, some that serve as a comparison standard, and some that reflect worst-case or utopian scenarios, e.g., exponential or uniform distributions. We derive the parameters for each P2P user characteristic via careful *sensitivity analysis*, by exploring multiple parameters for each distribution, until we achieve a representation that reflects observed user behavior within the limitations of a simulation environment.

Content availability: Extensive measurement studies [197][133][240][119] have confirmed the presence of a large number of free-riders in P2P systems. The distribution of the number of files shared by each peer appears to be heavy-tailed, though there is no agreement on the exact parameters. Hence, we take different models to represent file distribution. While Weibull (scale=42,shape=0.5) and Pareto (k=100,alpha=10) cases represent realistic behavior (i.e., large number of free-riders), the Uniform case (min=0,max=100) is used as a comparison base, and the Poisson case (mean=50) represents a scenario where every peer shares a constant number of files.

Session lengths: Churn in P2P systems has attracted much attention from researchers [214][101][219]. Again, while most studies agree that online session length is a heavy tailed distribution, different P2P systems have been shown to fit different distributions (or different parameters of the same distribution) at different times of measurement [101]. Hence, we represent online session lengths using different distributions where Pareto (k=600,alpha=0.5) and Weibull (scale=600,shape=0.2) cases represent realistic behavior, Uniform case (min=1,max=600) is used as a comparison base, and Poisson case (mean=300) represents the scenario where almost every peer has a constant online duration.

Query strings: Most P2P systems are characterized by query search phrases of two kinds [88]: constant phrases that aim to find content of a particular type, e.g., mp3, rap, dvd; and volatile phrases that search for a specific content, e.g. artist or album name. Query popularity distributions and load across time and region are reported in [119][88]. We reflect this by using 45% constant phrases and 45% volatile phrases for query strings. The rest 10% query strings are chosen such that they do not match any content in the network. Besides, 20% of all queries match only 1 or 2 content files. This enables us to analyze the effect of P2P locality on content search.

11.7 Results—Users Perspective

Next, we use the following metrics to judge end-user as well as ISP experience: number of responses that each Query generates, the AS distance and overlay hop count of Query-responses, time taken to download a single file, amount of exchanged content that remains within ISP network boundaries, and total reduction in P2P negotiation traffic. We perform two sets of experiments: (i) to study the effects of *various topologies* on the above metrics with realistic user behavior, comparing oracle-aided P2P with unmodified P2P, (ii) to measure the effects of *various user behavior patterns* on the above metrics for oracle-aided P2P. All the results are based on experiments with $10,000$ successful queries that result in $10,000$ file transfers. Each file is of 512KB size and is exchanged directly between the peers using HTTP.

11.7.1 Variation in Topology

For each topology model, we run two experiments, one with unmodified(U) P2P, another with oracle-aided and therefore a biased(B) P2P. In the unmodified case, P2P nodes go online, connect to random neighbors, search for content and exchange files, without consulting the ISP's oracle at any stage. In the biased case, P2P nodes consult the oracle while bootstrapping, as well as when downloading files. The bootstrapping phase is used to connect to proximal neighbors, hence setting up a localized P2P topology that is correlated with the Internet AS topology [17]. Nodes search for a specific content by flooding queries. On finding it at a set of nodes, they again consult the oracle to choose the best node for downloading. The oracle sorts the candidate list of neighbors based on the location of the nodes within the AS, last-hop bandwidth, and AS-hop distance. We model content availability and online session lengths by Weibull distributions (realistic behavior). The results for all 5 topologies are shown in Figure 23.

Content exchange: The most important metric for the end-user is the time taken to download content. As shown in Figure 23(a), the download time per 512KB file decreases by $1-3$ seconds (a reduction of $16-34\%$) for all 5 topologies, when P2P users consult the ISP-hosted oracle to choose proximal neighbors. This is because the download times are dominated by the bottleneck last-hop bandwidths [60]. Moreover we notice that changing the inter-AS delays does not have a significant effect on file download times, which confirms the result. From the ISP point of view, the amount of file content that remains within the ISP network boundaries more than doubles for the biased P2P case, see Figure 23(b). This can result in direct cost savings for the ISP, estimated to be in the order of $1 billion world-wide [40].

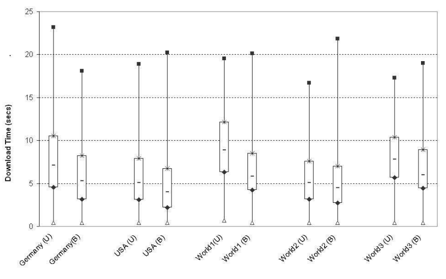
Query Search: Figure 23(c) shows that there is no adverse effect on the query search phase of P2P systems when nodes actively consult the oracle. We actually notice an increase in the number of query responses per query for the biased P2P case, which is a result of a more efficient swarming of queries (and their responses) within the localized P2P topology. A closer examination reveals that for the same number of unique queries, the negotiation traffic in the overlay resulting from flooding and forwarding of queries and their responses decreases by $17-40\%$ in the biased P2P topologies. Despite this welcome reduction in P2P traffic, there is no adverse effect, as the number of responses per query actually increases. This implies that a significantly smaller number of duplicate messages are carried in the overlay, thus improving the scalability of P2P systems and reducing the traffic in the ISP network.

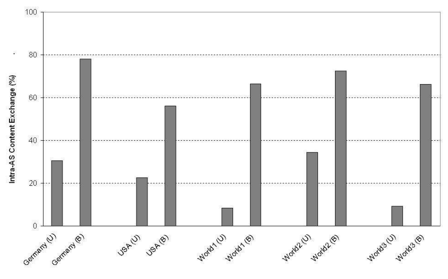
The number of queries that fail to find any content remains the same for unmodified as well as biased P2P. This means that even for the case of queries which match only 1 or 2 content files located somewhere in the network, the efficient swarming of queries in the localized topology ensures that queries find such content. Besides, the query responses now come more often from peers that are located within the same AS as the originating query, see Figure 23(d). This naturally leads to a decrease in the average AS distance of query responses per query for the biased P2P case.

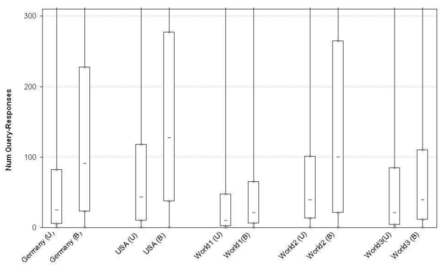
P2P topology: An investigation of the graph topological properties of biased overlay graphs reveals that localized P2P graphs maintain the nice graph properties typical of random overlays, namely, small node degree, small graph diameter, small mean path length and connectedness. The average node degree decreases from 22 for unmodified P2P to 18 for biased P2P, graph diameter remains constant at $6-7$ hops, and the mean overlay path length increases nominally from 2.5 hops for unmodified P2P to 3 hops for biased P2P. Importantly, despite heavy node churn, the overlay graph remains connected. Even if a sub-graph gets temporarily disconnected, P2P nodes quickly re-establish peerings and form a connected topology. This is also reflected in the good query responses, as discussed above.

11.7.2 Variation in User Behavior

Now that the benefits of ISP-aided P2P locality have been established across various topology models, we analyze the effects of user behavior on the above metrics. This helps to reveal the effect of aggressive node churn on graph connectivity and query responses. We also study scenarios when very few nodes serving most of the files in the P2P network go offline, and observe their impact on network performance. In


(a) File download time - box plot


(b) Amount of intra-AS file exchange - bar plot


(c) Number of Query-responses per Query - box plot

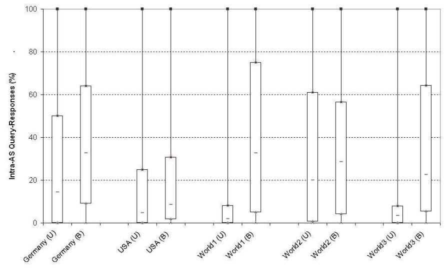

(d) Amount of intra-AS Query-responses per Query - box plot

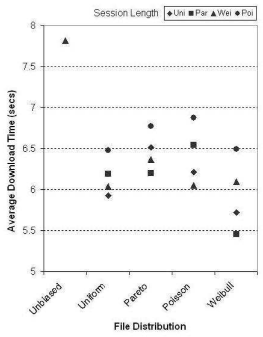
Figure 23: Plots comparing unmodified(U) and biased(B) P2P neighbor selection across 10K queries and 10K file transfers, for 5 topologies

other words, we determine if biased P2P maintains its benefits across different possible scenarios.

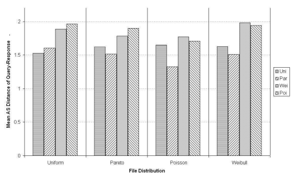
As explained in Section 11.6.2, we model content availability as well as session lengths as Uniform, Weibull, Pareto and Poisson distributions, thus giving us 16 possible combinations of the two characteristics. Hence, we run 16 different experiments for the biased P2P case for each topology. We focus on the World3 topology as the P2P nodes are nearly evenly distributed in each of the 16 ASes, thus minimizing the effect of topology on the metrics.

We see that across the 16 combinations of content availability and online session lengths for the biased P2P case, even though the median file download time (see Figure 24(a)) varies from $5.5 - 7$ seconds, it still remains below 7.8 seconds for unmodified P2P. The results for mean AS distance of query responses are similar, see Figure 24(b). The mean overlay hop count of query responses remains steady as shown in Figure 24(c), an important result for mobile applications where an increase in the overlay hop count can lead to performance degradations due to processing overhead. The amount of content exchanged within the ISP network boundaries ranges from $60-80\%$, more than double that for unmodified P2P. The success rate of queries remains the same, while the number of responses to queries remains consistently higher than unmodified P2P.

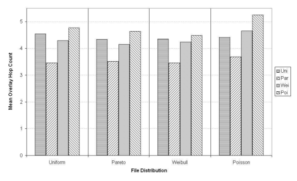
This shows that ISP-aided P2P neighbor selection maintains its benefits across dif-



(a) Median file download time



(b) Mean AS distance of Query-responses



(c) Overlay hop count of Query-responses

Figure 24: Effect of user behavior (content availability and session length) patterns for World3 topology. X-axis denotes file distribution models, and symbols denote online session length models: Uniform, Pareto, Weibull and Poisson.

ferent user behavior patterns. Even the presence of a large number of free-riders, or a large number of peers who have very short online durations does not adversely affect localized P2P topologies. The inherent dynamic of P2P systems ensures that the overlay graph remains connected, and is able to keep the node degrees as well as the graph diameter small.

11.8 Summary

In this section, we evaluate the Oracle concept for P2P networks using representative ISP/P2P topology models and user behavior characteristics in a simulation framework. Through extensive experiments, we show that both P2P users and ISPs benefit from ISP-aided P2P locality, measured in terms of improved content download times, increased network locality of query responses and desired content, and overall reduction in P2P traffic. The advantages hold across different ISP/P2P topologies under a broad range of user behavior scenarios.

12 Use Case for Collaboration: Traffic Engineering

The growth of demand for content and the resulting deployment of content delivery infrastructures pose new challenges to CPs and to ISPs. For CPs, the cost of deploying and maintaining such a massive infrastructure has significantly increased during the last years [188] and the revenue from delivering traffic to end-users has decreased due to the intense competition. Furthermore, CPs struggle to engineer and manage their infrastructures, replicate content based on end-user demand, and assign users to appropriate servers.

The latter is challenging as end-user to server assignment is based on inaccurate end-user location information [151, 55], and inferring the network conditions within an ISP without direct information from the network is difficult. Moreover, due to highly distributed server deployment and adaptive server assignment, the traffic injected by CPs is volatile. For example, if one of its locations is overloaded, a CP will re-assign end-users to other locations, resulting in large traffic shifts in the ISP network within minutes. Current traffic engineering by ISP networks adapts the routing and operates on time scales of several hours, and is therefore too slow to react to rapid traffic changes caused by CPs.

The pressure for cost reduction and customer satisfaction that both CPs and ISPs are confronted with, coupled with the opportunity that distributed server infrastructures offer, motivate us to propose a new tool in the traffic engineering landscape. We introduce *Content-aware Traffic Engineering* (CaTE). CaTE leverages the location diversity offered by CPs and, through this, it allows to adapt to traffic demand shifts. In fact, CaTE relies on the observation that by selecting an appropriate server among those available to deliver the content, the path of the traffic in the network can be influenced in a desired way. Figure 25 illustrates the basic concept of CaTE. The content requested by the client is in principle available from three servers (A, B, and C) in the network. However, the client only connects to one of the network locations. Today, the decision of where the client will connect to is solely done by the CP and is partially based on measurements and/or inference of network information and end-user location. With CaTE the decision on end-user to server assignment can be done jointly between the CP and ISP.

12.1 The CaTE Approach

CaTE complements existing traffic engineering solutions [21, 64, 109, 204, 231, 233] by focusing on traffic demands rather than routing. Let **y** be the vector of traffic counts on links and **x** the vector of traffic counts in origin-destination (OD) flows in the ISP network. Then **y**=A**x**, where A is the routing matrix. $A_{ij} = 1$ if the OD flow i traverses link j and 0 otherwise. Traditional traffic engineering is the process of adjusting A, given the OD flows **x**, so as to influence the link traffic **y** in a desirable way. In CaTE, we revisit traffic engineering by focusing on traffic demands rather than routing changes:

Definition 1: Content-aware Traffic Engineering (CaTE) is the process of adjusting the traffic demand vector **x**, given a routing matrix A, so as to change the link traffic **y**.

CaTE offers additional traffic engineering capabilities to both ISPs and CDNs to

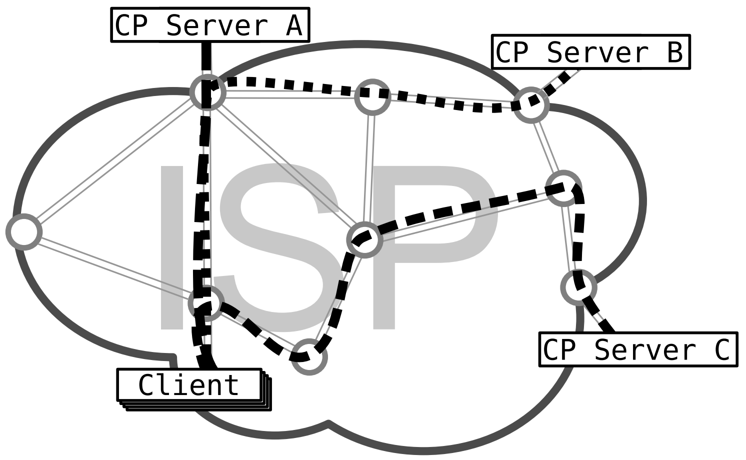


Figure 25: By choosing a CP server for a client with the help of CaTE, traffic engineering goals and accurate end-user server assignment become possible.

better manage the volatility of content demand in small time scales. Traditional traffic engineering [21, 64, 109, 204, 231, 233] relies on changes of routing weights that take place in the time scale of hours [81]. On the contrary, in CaTE (as we will show in Section 12.2.7), the redirection of end-users to servers can take place per request or within the TTL of a DNS query that is typically tens of seconds in large CDNs [183]. Thanks to the online recommendations by ISP networks, CDNs gain the ability to better assign end-users to servers and better amortize the deployment and maintenance cost of their infrastructure. Network bottlenecks are also circumvented and thus the ISP operation is improved. Furthermore, the burden of measuring and inferring network topology, and the state of the network, both challenging problems, is removed from the CDNs. Moreover, in [83, Sections 4 and 5] we show that the online CaTE decisions on the end-user to server assignment leads to optimal traffic assignment within the network under a number of different metrics. The advantage is that now the problem of assigning traffic to links reduces to a fractional solution (on the contrary, assigning routing weights to links is NP-hard). In short, all involved parties, including the end-users, benefit from CaTE, creating a win-win situation for everyone.

12.2 A Prototype to Support CaTE

CaTE relies on a close collaboration between CDN and ISP in small time scales (seconds or per request). To achieve this goal, network information has to be collected and processed by the ISP. Candidate CDN servers have to be communicated to the ISP and ranked based on a commonly agreed criteria, e.g., to optimize the delay between the end-user and the CDN server. Today, there is no system to support the above operations. This motivate us to design, implement and evaluate a novel and scalable system that can support CaTE. In this section we describe the architecture and deployment of our working prototype to enable CaTE. We start by presenting our prototype in Section 12.2.1. We then comment on its operation and deployment within the ISP, its interaction with a CDN, and its performance that is beyond the state-of-the-art [21].

12.2.1 Architecture

The CaTE system is installed in an ISP and interacts with the existing CDN server selector. The main tasks of the CaTE system are to: (1) maintain an up-to-date annotated map of the ISP network and its properties, (2) produce preference rankings based on the paths between end-users and candidate servers, and (3) communicate with the CDN server selection system to influence the assignment of end-user to servers. To this end, we propose an architecture that comprises a *Network Monitoring* component, a *Query Processing* component and a *communication interface* between an ISP and a CDN. For an overview of the architecture see Figure 26.

12.2.2 Network Monitoring

The network monitoring component gathers information about the topology and the state of the network from several sources to maintain an up-to-date view of the network. The network monitoring component consists of the following subcomponents:

The **Topology Information** component gathers detailed information about the basic network topology, i.e., routers and links, as well as annotations such as link utilization, router load, and topological changes. An Interior Gateway Protocol (IGP) listener provides up-to-date link-state (i.e., IS-IS, OSPF) information. Information about routers and links is retrieved, thus, the network topology can be extracted. The nominal link delay, i.e., the latency on a link without queuing, can be found through the link length and physical technology. The link utilization and other metrics can be retrieved via SNMP from the routers or an SNMP aggregator.

The **Connectivity Information** component uses routing information to calculate the paths that traffic takes through the network. Finding the path of egress traffic can be done by using a Border Gateway Protocol (BGP) listener. Ingress points of traffic into the ISP network can be found by utilizing Netflow data. This allows for complete forward and reverse path mapping inside the ISP. Furthermore, the system can map customers as well as CDN infrastructures into the network map by finding the routers that announce the address space associated with them. In total, this allows for a complete path map between any two points in the ISP network. Finally, our system has access to an uplink database that provides information about the connectivity statistics of end-users.

The **Network Map Database** component processes the information collected by the *Topology* and *Connectivity Information* components to build an annotated network map of the ISP network tailored towards fast lookup on path properties. It uses a layer of indirection to keep the more volatile information learned from BGP separate from the slower changing topological information. This allows address space to be quickly reassigned without any reprocessing of routing or path information. It also enables pre-calculation of path properties for all paths that yields a constant database lookup complexity independent of path length and network architecture. If topology changes, e.g., IGP weights change or a link fails, the *Topology Information* component immediately updates the database which only recalculates the properties of the affected

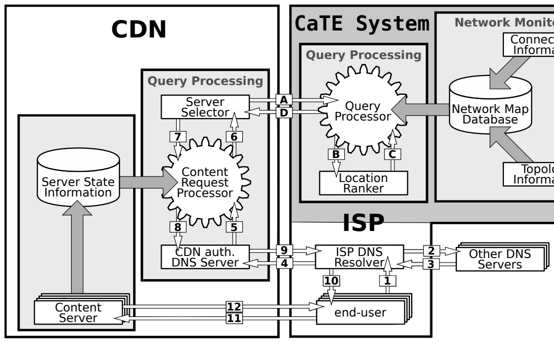


Figure 26: CaTE System architecture and flow of messages.

paths. Having ISP-centric information ready for fast access in a database ensures timely responses and high query throughput.

12.2.3 Query Processing

The **Query Processing** component receives a description of a request for content from the CDN, which specifies the end-user making the request and a list of candidate CDN servers. It then uses information from the *Network Map Database* and a selected ranking function to rank the candidate servers. This component consists of the following subcomponents:

The **Query Processor** receives the query from the CDN. First, the query processor maps each source-destination (server to end-user) pair to a path in the network. In most cases, the end-user is seen as the ISP DNS resolver, unless both ISP and CDN support the client IP eDNS extension [55]. Once the path is found, the properties of the path are retrieved. Next, the pairs are run individually through the location ranker subcomponent (see below) to get a preference value. Finally, the list is sorted by preference values, the values are stripped from the list, and it is sent back to the CDN.

The **Location Ranker** component computes the preference value for individual source-destination pairs based on the source-destination path properties and an appropriate function. Which function to use depends on (a) the CDN, (b) what metrics the CDN asked for and (c) the optimization goal of the ISP. The preference value for each source-destination pair is then handed back to the Query Processor. Multiple such optimization functions being defined upon the collaboration agreement and subsequently selected individually in each ranking request. For example, a function might be the minimization of end-user and server delay. In Section 13.5 we evaluate CaTE with multiple ranking functions for different optimization goals.

12.2.4 Communication Interfaces

When a CDN receives a content request, the *Server Selector* needs to choose a content server to fulfill this request. We propose that the server selector sends the list of eligible content servers along with the source of the query and an optimization goal to the ISP's CaTE system to obtain additional guidance about the underlying network. If the guidance is at granularity of a single DNS request, we propose a DNS-like protocol using UDP to prevent extra overhead for connection management. If the granularity is at a coarser level, i.e. seconds or even minutes, we rely on TCP.

12.2.5 Privacy and Performance

During the exchange of messages, none of the parties is revealing any sensitive operational information. CDNs only reveal the candidate servers that can respond to a given request without any additional operational information (e.g., CDN server load, cost of delivery or any reason why a server is chosen). The set of candidate servers can be updated per request or within a TTL that is typically in the order of a tens of seconds in popular CDNs [183]. On the other side, the ISP does not reveal any operational information or the preference weights it uses for the ranking. In fact, the ISP only re-orders a list of candidate servers provided by the CDN. This approach differs significantly from [21, 233], where partial or complete ISP network information, routing weights, or ranking scores are publicly available. We argue that an important aspect to improve content delivery is to rely on up-to-date information during server selection of the CDN. This also eliminates the need of CDNs to perform active measurements to infer the conditions within the ISP that can add overhead to CDN operation and may be inaccurate. With CaTE, the final decision is still made by the CDN, yet it is augmented with up-to-date network guidance from the ISP.

To improve the performance of our system, we do not rely on XML-based network maps as proposed in [21], but on light protocols that are close to DNS in design. This design choice is important as topology information in large networks (in the order of multiple MBytes). Transferring this information periodically to many end-users is likely to be challenging. In a single instance of our system, we manage to reply to up to $90,000$ queries/sec when 50 candidate servers supplied by the CDN. At this level, the performance of our system is comparable to that of current DNS servers, such as BIND. However, the number of replies drops to around $15,000$ per second when considering 350 candidate servers. The additional response time when our system is used is around 1 ms when the number of candidate servers is 50 and around 4 ms when considering 350 candidate servers. This overhead is small compared to the DNS resolution time [10]. The performance was achieved on a commodity dual-quad core server with 32 GB of RAM and 1GBit Ethernet interfaces. Furthermore, running additional servers does not require any synchronization between them. Thus, multiple servers can be located in different places inside the network (see Section 12.2.6).

12.2.6 Deployment

Deploying the system inside the ISP network does not require any change in the network configuration or ISP DNS operation. Our system solely relies on protocol listeners and access to ISP network information. Moreover, no installation of special software is required by end-users. The CaTE system adds minimal overhead to ISPs and CDNs. It only requires the installation of a server in both sides to facilitate communication between them.

Typically, an ISP operates a number of DNS resolvers to better balance the load of DNS requests and to locate DNS servers closer to end-users. To this end, we envision that the ISP's CaTE servers can be co-located with DNS resolvers in order to scale in the same fashion as DNS. CaTE servers can also be located close to peering points in order to reduce the latency between the CDN and an instance of the system. Synchronization of multiple CaTE instances is not necessary as they are aware of the state of the same network. We concluded that this is the best deployment strategy, other possible deployment strategies we have considered are presented in [83].

12.2.7 Operation

We now describe the operation of our working prototype and its interaction with the CDN. In Figure 26 we illustrate the basic system architecture to support CaTE including the flow of information when the CaTE system is used. When a DNS request is submitted by an end-user to the ISP DNS resolvers *(1)* there are a number of recursive steps *(2)* until the authoritative DNS server is found *(3)*. Then, the ISP DNS resolver contacts the authoritative DNS server *(4)*. There, the request is handed to the content request processor operated by the CDN query processing component *(5)*. The content request processor has access to full information about the status of the CDN. Based on the operational status of the CDN servers, the server selection system [170] is responsible for choosing eligible content servers *(6)*. In the end, a preference list of content servers is generated. At this point, the CDN server selector sends the list of eligible content servers *(A)* along with user information, such as the IP of the DNS resolvers or client and an optimization metric to ISP. The query processor of the ISP system ranks the list using the location ranker *(B)*. After all the elements have been processed, the query processor has an annotated list with preferences for the ISP *(C)*. The query processor sorts the list by the preference values, strips the values and sends the list back to the CDN *(D)*. The CDN server selector incorporates the feedback, selects the best content server(s) and hand them back to the content request processor *(7)*. Then, the answer travels the path back to the client, i.e. from the CDN's authoritative DNS server *(8)* via the ISP DNS resolver *(9)* to the end-user *(10)*. Finally, the end-user contacts the selected server *(11)* and downloads the content *(12)*.

12.3 Modeling CaTE

Next, we formalize CaTE and discuss how it relates to traditional traffic engineering and multipath routing.

12.3.1 Traffic Engineering

We model the network as a directed graph $G(V, E)$ where V is the set of nodes and E is the set of links. An origin-destination (OD) flow f_{od} consists of all traffic entering the network at a given point $o \in V$ (origin) and exiting the network at some point $d \in V$ (destination). The traffic on a link is the superposition of all OD flows that traverse the link.

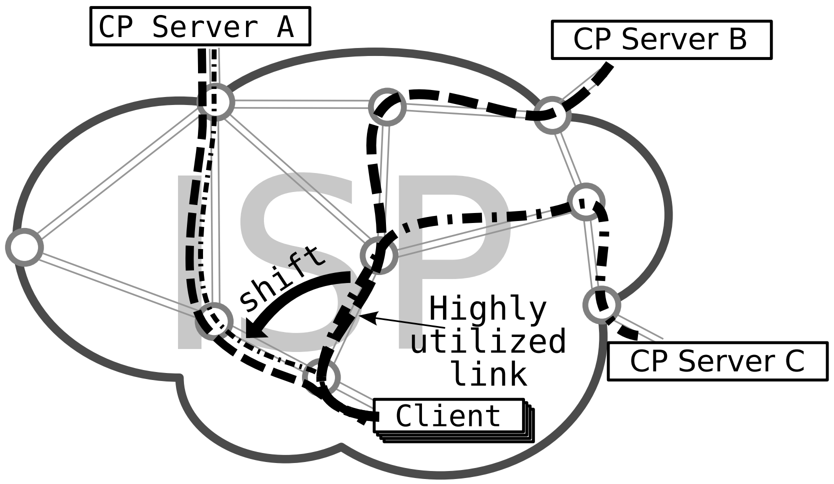


Figure 27: Content-aware Traffic Engineering Process

The relationship between link and OD flow traffic is expressed by the routing matrix A. The matrix A has size $|E| \times |V|^2$. Each element of matrix A has a boolean value. $A_{ml} = 1$ if OD flow m traverses link l, and 0 otherwise. The routing matrix A can be derived from routing protocols, e.g., OSPF, ISIS, BGP. Typically, A is very sparse since each OD flow traverses only a very small number of links. Let **y** be a vector of size $|E|$ with traffic counts on links and **x** a vector of size $|V|^2$ with traffic counts in OD flows, then **y**=A**x**. Note, **x** is the vector representation of the traffic matrix.

Traditional Traffic Engineering: In its broadest sense, traffic engineering encompasses the application of technology and scientific principles to the measurement, characterization, modeling, and control of Internet traffic [30]. Traditionally, traffic engineering reduces to controlling and optimizing the routing function and to steering traffic through the network in the most effective way. Translated into the above matrix form, traffic engineering is the process of adjusting A, given the OD flows **x**, so as to influence the link traffic **y** in a desirable way, as coined in [129]. The above definition assumes that the OD flow vector **x** is known. For instance, direct observations can be obtained, e.g., with Netflow data [48, 74].

Terminology: We denote as *flow* an OD flow between two routers in the network. We call a flow *splittable* if arbitrarily small pieces of the flow can be assigned to other flows. This is not to be confused with end-to-end sessions, i.e., TCP connections, which are *un-splittable*. The assumption that flows are splittable is reasonable, as the percentage of traffic of a single end-to-end session is small compared to that of a flow between routers. Let C be the set of nominal capacities of the links in the network G. We denote as *link utilization* the fraction of the link capacity that is used by flows. We denote as *flow utilization* the maximum link utilization among all links that a flow traverses. We introduce the terms of *traffic consumer* and *traffic producer* which refer to the aggregated demand of users attached to a router, and the CPs that are responsible for the traffic respectively. We refer to the different alternatives from which content can be supplied by a given CP as *network locations* that host servers.

12.3.2 Definition of CaTE

We revisit traffic engineering by focusing on the traffic demands rather than changing the routing.

Definition 1: Content-aware Traffic Engineering(CaTE) is the process of adjusting the traffic demand vector $\mathbf{x}$, given a routing matrix A, so as to change the link traffic $\mathbf{y}$.

Not all the traffic can be adjusted arbitrarily. Only traffic for which location diversity is available can be adjusted by CaTE. Therefore, $\mathbf{x}$=$\mathbf{x}_r$+$\mathbf{x}_s$ where $\mathbf{x}_r$ denotes the content demands that can be adjusted and $\mathbf{x}_s$ denotes the content demands that can not be adjusted as there is only a single location in the network where the content can be downloaded from. The amount of traffic that can be adjusted depends on the diversity of locations from which the content can be obtained. We can rewrite the relation between traffic counts on links and traffic counts in flows as follows: $\mathbf{y}$=$A(\mathbf{x}_s + \mathbf{x}_r)$. CaTE adjusts the traffic on each link of the network by adjusting the content demands $\mathbf{x}_r$: $\mathbf{y}_r$=$A\mathbf{x}_r$. Applying CaTE means adjusting the content demand to satisfy a traffic engineering goal.

Definition 2: Optimal Traffic Matrix is the new traffic matrix, $\mathbf{x}^*$, after applying CaTE, given a network topology G, a routing matrix A and an initial traffic matrix $\mathbf{x}$.

Figure 27 illustrates the CaTE process. A content consumer requests content that three different servers can deliver. Let us assume that, without CaTE, the CP redirects the clients to servers B and C. Unfortunately, the resulting traffic crosses a highly-utilized link. With CaTE, content can also be downloaded from server A, thus, the traffic within the network is better balanced as the highly utilized link is circumvented.

Minimizing the maximum utilization across all links in a network is a popular traffic engineering goal [80, 81, 136]. It potentially improves the quality of experience and postpones the need for capacity increase. CaTE mitigates bottlenecks and minimizes the maximum link utilization by re-assigning parts of the traffic traversing heavily loaded paths. Thus it redirects traffic to other, less utilized paths. Later in this chapter, we will elaborate in Section 12.6, different metrics such as path length or network delay can also be used in CaTE.

12.3.3 CaTE and Traditional TE

CaTE is complementary to routing-based traffic engineering as it does not modify the routing. Routing-based traffic engineering adjusts routing weights to adapt to traffic matrix changes. To avoid micro-loops during IGP convergence [82], it is common practice to only adjust a small number of routing weights [81]. To limit the number of changes in routing weights, routing-based traffic engineering relies on traffic matrices computed over long time periods and offline estimation of the routing weights. Therefore, routing-based traffic engineering operates on time scales of hours, which can be too slow to react to rapid change of traffic demands. CaTE complements routing-based traffic engineering and can influence flows at shorter time scales by assigning clients to servers on a per request basis. Thus, CaTE influences the traffic within a network online in a fine-grained fashion.

12.3.4 CaTE and Multipath Routing

Multipath routing helps end-hosts to increase and control their upload capacity [116]. It can be used to minimize transit costs [93]. Multipath also enables ASes to dynamically distribute the load inside networks in the presence of volatile and hard to predict traffic demand changes [74, 67, 113, 76]. This is a significant advantage, as routing-based traffic engineering can be too slow to react to phenomena such as flash crowds. Multipath takes advantage of the diversity of paths to better distribute traffic.

CaTE also leverages the path diversity, and can be advantageously combined with multipath to further improve traffic engineering and end-user performance. One of the advantages of CaTE is its limited investments in hardware deployed within an ISP. It can be realized with no change to routers, contrary to some of the previous multipath proposals [113, 67, 76]. The overhead of CaTE is also limited as no state about individual TCP connections needs to be maintained, contrary to multipath [113, 67, 76]. In contrast to [67, 113], CaTE is not restricted to MPLS-like solutions and is easily deployable in todays networks.

12.3.5 CaTE and Oscillations

Theoretical results [78, 77] have shown that load balancing algorithms can take advantage of multipath while provably avoiding traffic oscillations. In addition, their convergence is fast. Building on these theoretical results, Fischer et al. proposed REPLEX [76], a dynamic traffic engineering algorithm that exploits the fact that there are multiple paths to a destination. It dynamically changes the traffic load routed on each path. Extensive simulations show that REPLEX leads to fast convergence, without oscillations, even when there is lag between consecutive updates about the state of the network. CaTE is derived from the same principles and thus inherits all the above-mentioned desired properties.

12.4 CaTE Algorithms

In this section we propose algorithms to realize CaTE, in the context of an ISP. A key observation is that CaTE can be reduced to the restricted machine load balancing problem [32] for which optimal online algorithms are available. The benefit of the CaTE online algorithm can be estimated either by reporting results from field tests within an ISP or by using trace-driven simulations. Typically, in operational networks only aggregated monitoring data is available. To estimate the benefit that CaTE offers to an ISP, we present offline algorithms that uses traffic demands and server diversity over time extracted from those statistics as input.

12.4.1 Connection to Restricted Machine Load Balancing

Given a set of CPs and their network location diversity, we consider the problem of re-assigning the flows that correspond to demands of content consumers to the CPs in such a way that a specific traffic engineering goal is achieved. Given that sub-flows between end-systems and content provider servers can be re-distributed only to a subset of the network paths, we show that the solution of the optimal traffic matrix problem

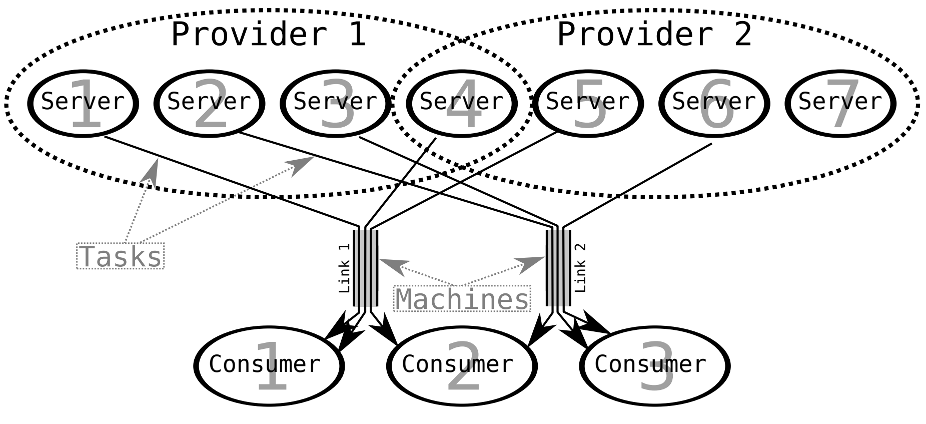


Figure 28: CaTE and Restricted Machine Load Balancing.

corresponds to solving the *restricted machine load balancing problem* [32]. In the restricted machine load balancing problem, a sequence of tasks is arriving, where each task can be executed by a subset of all the available machines. The goal is to assign each task upon arrival to one of the machines that can execute it so that the total load is minimized. Note, contrary to the case of multipath where paths between only one source-destination pair are utilized, CaTE can utilize any eligible path between any candidate source and destination of traffic.

For ease of presentation let us assume that the traffic engineering goal is to minimize the maximum link utilization in the network [80, 81]. Let us consider three consumers where each one wants to download one unit of content from two different content providers, see Figure 28. Given that different servers can deliver the content on behalf of the two providers, the problem consists in assigning consumers to servers in such a way that their demands are satisfied while minimizing the maximum link utilization in the network. Thus, the problem is the restricted machine load balancing one where tasks are the demands satisfied by the servers and machines are the bottleneck links that are traversed when a path, out of all eligible server-consumer paths, is selected. Figure 28 shows one of the possible solutions to this problem, where consumer 1 is assigned to servers 1 and 4, consumer 2 to servers 5 and 2, and consumer 3 to servers 3 and 6. Note that the machine load refers to the utilization of the bottleneck links of eligible paths, denoted as link 1 and 2.

To be consistent with our terminology, we define the *restricted flow load balancing problem*. Let J be the set of the consumers in the network, K be the set of content producers, and I be the set of servers for a given content provider, i.e., the set of locations where a request can be satisfied. Note, this set is offered by the CP in order to satisfy its own objectives and can change over time. We denote as M_{jk} the set of flows that can deliver content for a given content producer k to consumer j.

Definition 3: Restricted Flow Load Balancing Problem is the problem of finding a feasible assignment of flows such that a traffic engineering goal is achieved, given a set of sub-flows $\{f_{ijk}\}$ from all eligible servers $i \in I$ of a given content provider $k \in K$ to a consumer $j \in J$, and a set of eligible residual flows f_{ij}^{-k}, $i \in M_{jk}$ (after removing the traffic of the above mentioned sub-flows).

Despite some similarities, the nature of our problem differs from the multi-commodity flow and bin packing. In the multi-commodity flow problem [31], the demand between source and destination pairs is given while in our problem the assignment of demands is part of the solution. In the bin packing problem [52], the objective is to minimize the number of bins, i.e., number of flows in our setting, even if this means deviating from the given traffic engineering goal. Note, in the restricted flow load balancing problem any eligible path from a candidate source to the destination can be used, contrary to the multipath problem where only equal-cost paths can be used.

12.4.2 Online Algorithm and Competitiveness

We next turn to the design of online algorithms. It has been shown that in the online restricted machine load balancing problem, the greedy algorithm that schedules a permanent task to an eligible processor having the least load is exactly optimal [32], i.e., it is the best that can be found, achieving a competitive ratio of $\lceil \log_2 n \rceil + 1$, where n is the number of machines. If tasks are splittable then the greedy algorithm is 1-competitive, i.e., it yields the same performance as an offline optimal algorithm. The greedy algorithm is an online one, thus it converges to the optimal solution immediately without oscillations.

In the restricted flow load balancing problem, the set M_{jk} can be obtained from the set of candidate servers that can deliver content when utilizing CaTE as described in Section 12.2.6. The online assignment of users to servers per request, which minimizes the overall load, leads to an optimal assignment of sessions within sub-flows. In our case, flows are splittable since the content corresponding to each content request is negligible compared to the overall traffic traversing a link. Note, the end-to-end TCP connections are not splittable. Thus, the following online algorithm is optimal:
Algorithm 1. Online Greedy Server Selection. Upon the arrival of a content user request, assign the user to the server that can deliver the content, out of all the servers offered by the CP, such that the traffic engineering goal is achieved.

12.5 Estimating the Benefit of CaTE with Passive Measurements

Before applying CaTE in real operational networks, it is important to understand the potential benefits that it can bring in a given context. For example, the operator of an ISP network would like to know in advance what are the gains when applying CaTE, as well as being able to answer what-if scenarios, when applying CaTE to traffic delivered by different CPs. Operators of CPs would also like to quantify the benefits by participating in CaTE before collaborating with an ISP. In most operational networks, aggregated statistics and passive measurements are collected to support operational decisions. Therefore, we provide a framework that allows a simulation-driven evaluation of CaTE. To that end, we present offline algorithms that can take as input passive measurements and evaluate the potential gain when applying CaTE in different scenarios We propose a linear programming formulation as well as greedy approximation algorithms to speed-up the process of estimating the gain when using CaTE.

12.5.1 Linear Programming Formulation

To estimate the potential improvement of CaTE we formulate the Restricted Flow Load Balancing problem (see Section 12.4.1) as a Linear Program (LP) with restrictions on the variable values. Variables f_{ijk} correspond to flows that can be influenced. Setting $f_{ijk} = 0$ indicates that consumer j cannot download the content from server i of a content provider k. For each consumer j we require that its demand d_{jk} for content provider k is satisfied, i.e., we require $\sum_{i \in M_{jk}} f_{ijk} = d_{jk}$. The utilization on a flow f_{ij} is expressed as $f_{ij} = \sum_k f_{ijk}$.

We use the objective function to encode the traffic engineering goal. For ease of presentation we use as objective function the minimization of the maximum link utilization. Let T_e be the set of flows f_{ij} that traverse a link $e \in E$. The link utilization of a link $e \in E$ is expressed as $L_e = \sum_{T_e} f_{ij}$. Let variable L correspond to the maximum link utilization. We use the inequality $\sum_{T_e} f_{ij} \leq L$ for all links. This results in the following LP problem:

$$\begin{array}{ll}
\texttt{min}\, L & \\
\sum_i f_{ijk} = d_{jk}, & \forall\, j \in J,\ k \in K \\
\sum_{T_e} f_{ijk} \leq L, & \forall\, j \in J,\ i \in I,\ k \in K,\ e \in E \\
0 \leq f_{ijk} \leq d_{jk}, & \forall\, j \in J,\ i \in M_{jk},\ k \in K \\
f_{ijk} = 0, & \forall\, j \in J,\ i \notin M_{jk},\ k \in K
\end{array}$$

The solution of the above LP provides a fractional assignment of flows under the assumption that flows are splittable and thus can be solved in polynomial time [117]. The solution is the optimal flow assignment, f^*_{ijk}, that corresponds to the optimal traffic matrix $\mathbf{x}^*$. If flows are not splittable, or the sub-flows are discretized, then the integer programming formulation has to be solved. In this case the Restricted Flow Load Balancing problem is NP-hard and a polynomial time rounding algorithm that approximates the assignment within a factor of 2 exists [138].

12.5.2 Approximation Algorithms

Since it is a common practice for operators to study multiple scenarios to quantify the effect of changes in traffic matrices over periods that spans multiple weeks or months, solutions based on LP may be too slow. It might be also too slow to estimate the gain of CaTE when applying it to an arbitrary combination of CPs. To that end, we turn our attention to the design of fast approximation algorithms. Simple greedy algorithms for load balancing problems [97] are among the best known. Accordingly, we propose a greedy algorithm for our problem which starts with the largest flow first.

Algorithm 3: Greedy-Sort-Flow. Sort sub-flows in decreasing order based on volume and re-assign them in this order to any other eligible flow which, after assigning the

Algorithm 2: Iterative Greedy-Sort-Flow.

```
INPUT: $I$, $J$, $K$, $\{f_{ijk}\}$, $\{M_{jk}\}$, $A$.
OUTPUT: $\{f^*_{ijk}\}$.

Initialization:
1. Sort $k \in K$ by decreasing volume: $\sum_i \sum_j f_{ijk}$.
2. Sort $j \in J$ by decreasing volume: $\sum_i f_{ijk}$ for all $k \in K$.

Iteration:
Until no sub-flow is re-assigned or the maximum number of
iterations has been reached.
  ▷ Pick unprocessed $k \in K$ in descending order.
    ▷ Pick unprocessed $j \in J$ in descending order.
      ▷ Re-assign $f_{ijk}$ in $f_{ij}^{-k}$, $i \in M_{jk}$ s.t. the engineering
         goal is achieved.
```

sub-flow f_{ijk}, will yield the desire traffic engineering goal.

Assignment in sorted order has been shown to significantly improve the approximation ratio and the convergence speed [59, 97]. Recent studies [87, 128, 183] show that a small number of content providers are responsible for a large fraction of the traffic. Therefore it is expected that the algorithm yields results close to the optimal ones. To further improve the accuracy of the proposed approximation algorithm, we design an *iterative* version of the algorithm, presented in Algorithm 2, that converges to the optimal solution. Indeed, a small number of iterations, typically one, suffice to provide a stable assignment of flows.

As we elaborate in Section 12.6, we performed a number of simulations using real operational traces, and different sets of CPs. Our evaluation show that the performance of the iterative greedy algorithm presented in Algorithm 2 yields results very close to this obtained with LP, but in significantly shorter time.

12.6 Impact of Collaboration: Traffic Engineering

In this section, we quantify the potential of CaTE with different traffic engineering goals in mind. We evaluate CaTE with operational data from three different networks. For the first network, we rely on content demands built from observed traffic of a European Tier-1 ISP. The other two networks, namely AT&T and Abilene, allow us to evaluate the impact of the ISP topology structure.

Experimental Setting

To evaluate CaTE, an understanding of the studied ISP network is necessary, including its topological properties and their implications on the flow of traffic. Indeed, the topological properties of the ISP network influence the availability of disjoint paths, which are key to benefit from the load-balancing ability of CaTE. Because CaTE influences traffic aggregates inside the ISP network at the granularity of requests directed to CPs, fine-grained traffic statistics are necessary. Traffic counts per-OD flow, often used in the literature, are too coarse an input for CaTE.

Data from a Large European ISP To build fine-grained traffic demands, we rely on anonymized packet-level traces of residential DSL connections from a large European Tier-1 ISP, henceforth called *ISP1*. For ISP1, we have the complete annotated router-level topology including the router locations as well as all public and private peerings. ISP1 contains more than 650 routers and 30 peering points all over the world.

We collect a 10 days long trace starting on May 7, 2010. Our monitor, using Endace monitoring cards [51], allows us to observe the traffic of more than $20,000$ DSL lines to the Internet. We capture HTTP and DNS traffic using the Bro intrusion detection system [179]. We observe 720 million DNS messages as well as more than 1 billion HTTP requests involving about 1.4 million unique hostnames, representing more than 35 TBytes of data. With regards to the application mix, more than 65% of the traffic volume is due to HTTP. Other popular applications that contribute to the overall traffic volume are NNTP, BitTorrent, and eDonkey.

A large fraction of the traffic in the Internet is due to large CPs, including CDNs, hyper-giants, and OCHs, as reported in earlier studies [87, 128, 183]. In Figure 29, we plot the cumulative fraction of HTTP traffic volume as a function of the CPs that originate the traffic. We define a CP as a organizational unit where all servers from the distributed infrastructure serve the same content, such as Akamai or Google. We rank the CPs by decreasing traffic volume observed in our trace. Note that the x-axis uses a logarithmic scale. The top 10 CPs are responsible for around 40% of the HTTP traffic volume and the top 100 CPs for close to 70% of the HTTP traffic volume. The marginal increase of traffic is diminishing when increasing the number of CPs. This shows that collaborating directly with a small number of large CPs, can yield significant savings.

In Figure 30 we plot the traffic of the top 1, 10, 100 CPs by volume as well as the total traffic over time normalized to the peak traffic in our dataset. For illustrative purposes, we show the evolution across the first 60 hours of our trace. A strong diurnal pattern of traffic activity is observed. We again observe that a small number of CPs are responsible for about half of the traffic. Similar observations are made for the rest of the trace.

Understanding the Location Diversity of CPs

To achieve traffic engineering goals, it is crucial to also understand the location

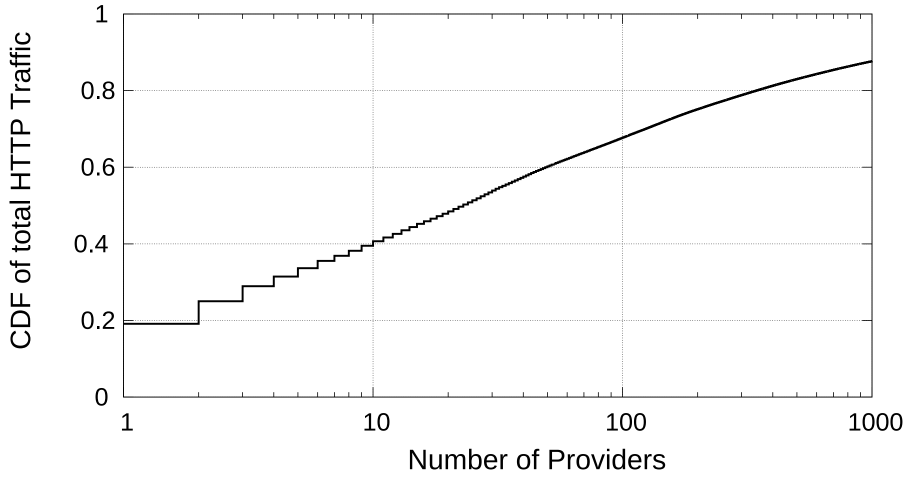


Figure 29: CDF of traffic volume of CPs in ISP1.

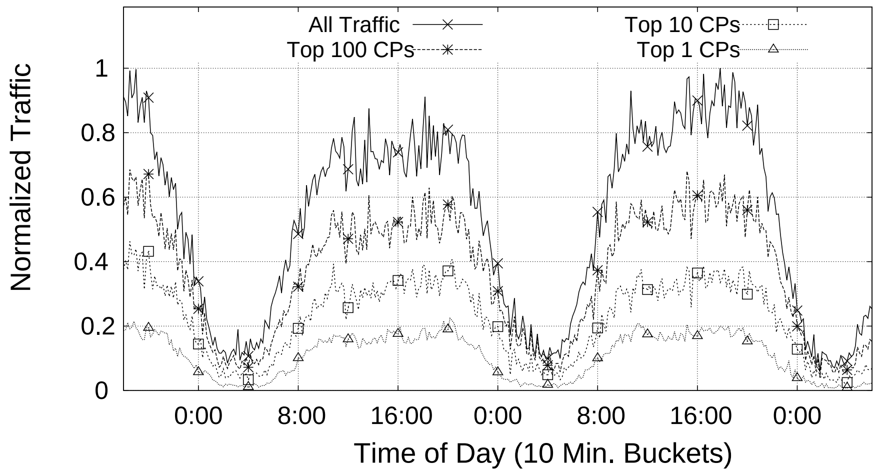


Figure 30: Normalized traffic for top CPs by volume in ISP1.

diversity of the top CPs, as CaTE relies on the fact that the same content is available at multiple locations. Traffic originated from multiple network locations by a given CP is seen by CaTE as a single atomic traffic aggregate to be engineered. Furthermore, as routing in the Internet works per prefix, we assume that the granularity of subnets is the finest at which CaTE should engineer the traffic demand. Thus, we differentiate candidate locations of CPs by their subnets and quantify the location diversity of CPs through the number of subnets from which content can be obtained.

We examine the amount of location diversity offered by CPs based on traces from ISP1. To identify the subnets of individual CPs, we rely on a similar methodology to the one from Poese et al. [183]. Our granularity is comparable to their ”infrastructure redirection aggregation”. Figure 31 shows the cumulative fraction of HTTP traffic as a

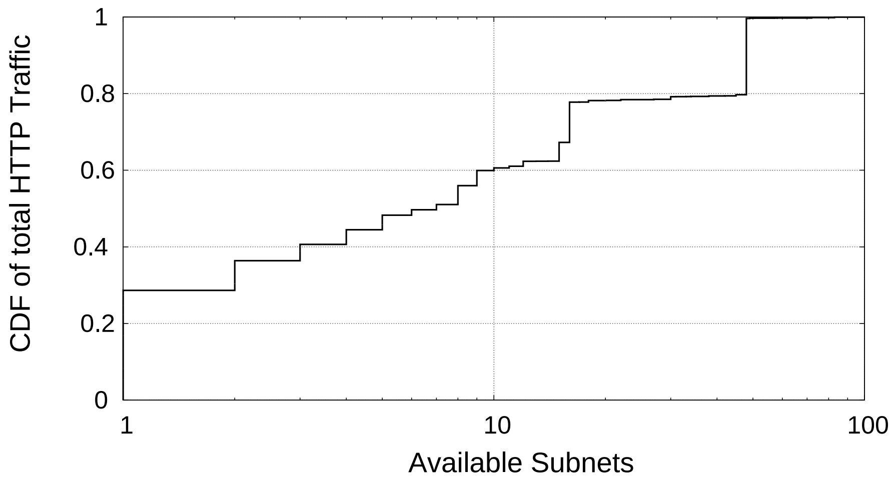


Figure 31: Subnet diversity from which content is available.

function of the number of subnets (logarithmic scale) from which a given content can be obtained, over the entire 10 days of the trace. We observe that more than 50% of the HTTP traffic can be delivered from at least 8 different subnets, and more than 60% of the HTTP traffic from more than 3 locations. These results confirm the observations made in [183].

Dynamics in Location Diversity

So far the location diversity of CPs has been evaluated irrespective of time. To complement the finding, we turn our attention to the location diversity exposed by CPs at small time-scales, i.e., in the order of minutes. To this end, we split the original trace into 10 minutes bins. Figure 32 shows the evolution of the number of exposed subnets of five of the top 10 CPs by volume. Note that the diversity exposed by some CPs exhibits explicit time of day patterns, while others do not. This can be due to the structural setup or the type of content served by the CP. The exposed location diversity patterns, i.e., flat or diurnal, are representative for all CPs with a major traffic share in our trace. We conclude that a significant location diversity is exposed by popular CPs at any point in time, and is quite extensive during the peak hour.

Content Demand Generation

The location diversity is not a mere observation about CPs deployment. It requires to revisit the mapping between a given content demand and the realized traffic matrix. Given the location diversity for content, multiple traffic matrices can be realized from a given content demand. The standard view of the OD flows therefore provides an incomplete picture of the options available for CaTE.

As an input for CaTE, we introduce an abstraction of the demand that reflects the available location diversity. We rely on the notion of *potential vectors*, that were denoted as x_r in Section 12.3.2. To generate the potential vector for a given CP, the amount of traffic this CP originates as well as the potential ingress points need to be known. Combining all potential vectors and x_s, we synthesize a network-wide content demand matrix for each time bin, by scaling the traffic demand to match the network

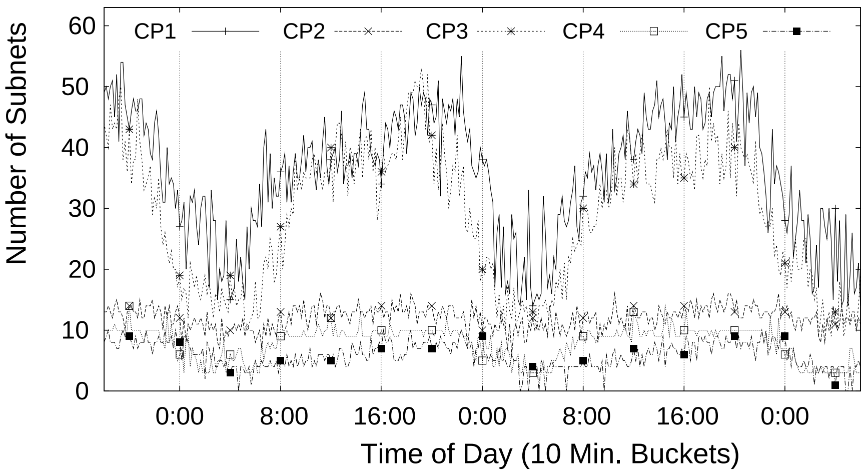


Figure 32: Evolution over time of number of subnets for selected CPs in the top 10 CPs.

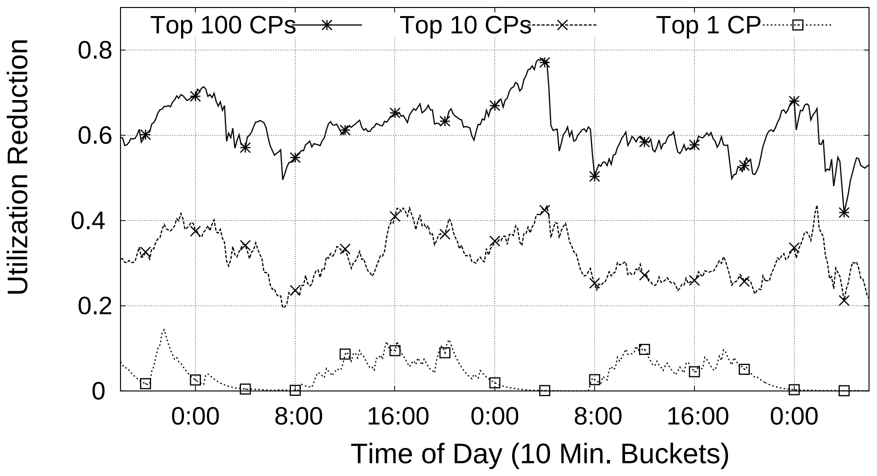
utilization of ISP1. For our evaluation, we use the series of content demand matrices over a period of 10 days. The content demands are based exclusively on the HTTP traffic of our trace.

12.6.1 CaTE in ISP1

To quantify the benefits of CaTE, we first consider one of the most popular traffic engineering goals, namely minimizing the maximum utilization of the links in the network [80, 81]. The rationale is that by minimizing the maximum link utilization, network bottlenecks are reduced, in turn limiting queuing delays, improving the quality of experience and postponing the need for increased network capacity.

With CaTE, an ISP can collaborate with any CP. It is up to the ISP to select the set of CPs that are the most important to establish collaboration with. Since a significant fraction of the traffic originates from a small number of CPs, we consider the most popular CPs by volume to evaluate CaTE. In the following, we perform a sensitivity study where we quantify the benefits of CaTE when restricting its use to the top 1, 10 and 100 CPs by volume. All other traffic remains unaffected by CaTE. For all experiments, we use the Algorithm 2 from Appendix 12.5.2.

Effect on Maximum Link Utilization. Figure 33 (top) shows the reduction of the maximum link utilization over a period of 2 days when considering the top 1, 10 and 100 CPs. Once again, we normalized the absolute link utilization by the maximal one. The largest gain in maximum link utilization reduction is up to 15%, 40% and 70% respectively. We observe large fluctuations of the gains which are due to variations in traffic (see Figure 31) and location diversity (Figure 32) throughout the day. The largest gains are obtained during peak times, when there is more traffic and the highest location diversity is available. This is also when congestion is at its peak and CaTE is most needed. Our results show that CaTE is able to react to diurnal changes in traffic



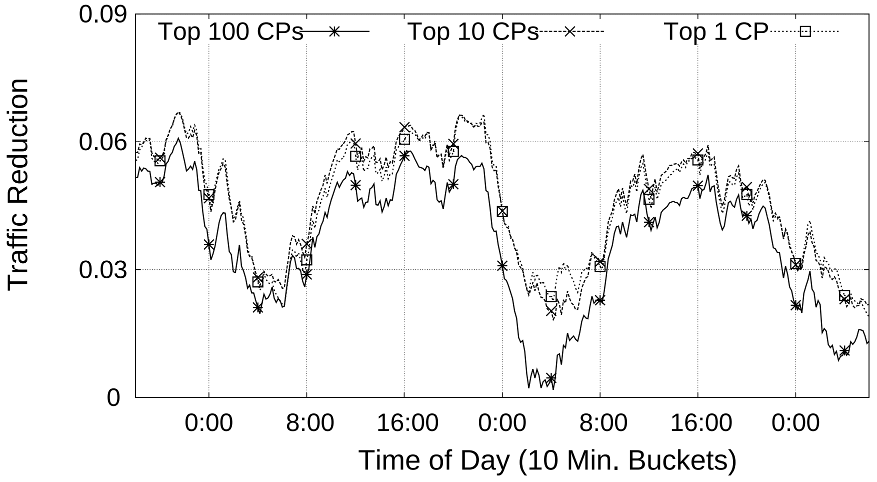


Figure 33: Maximum link utilization reduction (top) and total traffic reduction (bottom) with CaTE for the top CPs.

volume and utilizes the available location diversity.
Effect on Network-wide Traffic. Although optimizing for link utilization, CaTE reduces the overall traffic that flows through the network, see Figure 33 (bottom). This is due to CaTE choosing the shortest path when multiple ones with the same utilization are available, thus, as a side effect, content is fetched from closer locations and therefore traverses less links. With CaTE, the gains in overall traffic reduction are up to 6% and follows a clear diurnal pattern. It is worth noticing that just with the top 10 CPs, the total traffic reduction is very close to the one when considering the top 100 CPs, indicating that CaTE only needs to be implemented with the major players. Also, an ISP that is able to reduce the overall traffic inside its network is more competitive as it can serve more end-users with the same infrastructure, delay additional investments in

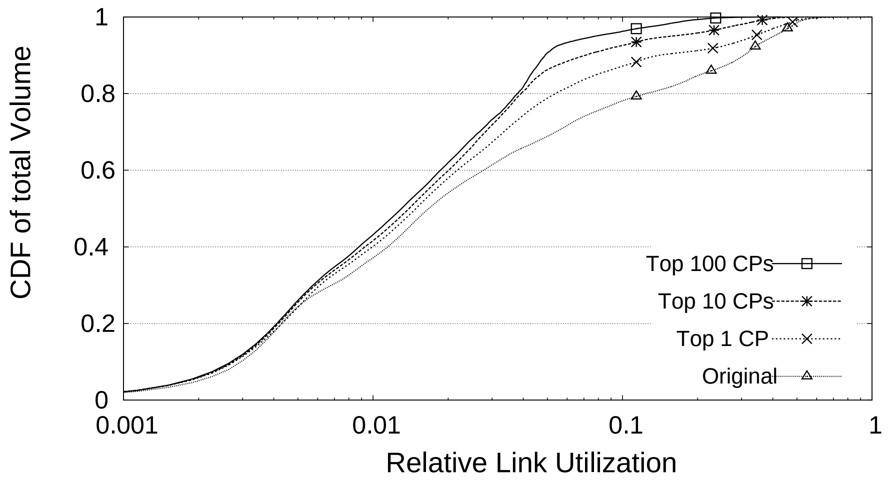


Figure 34: Improvements in link utilization with CaTE.

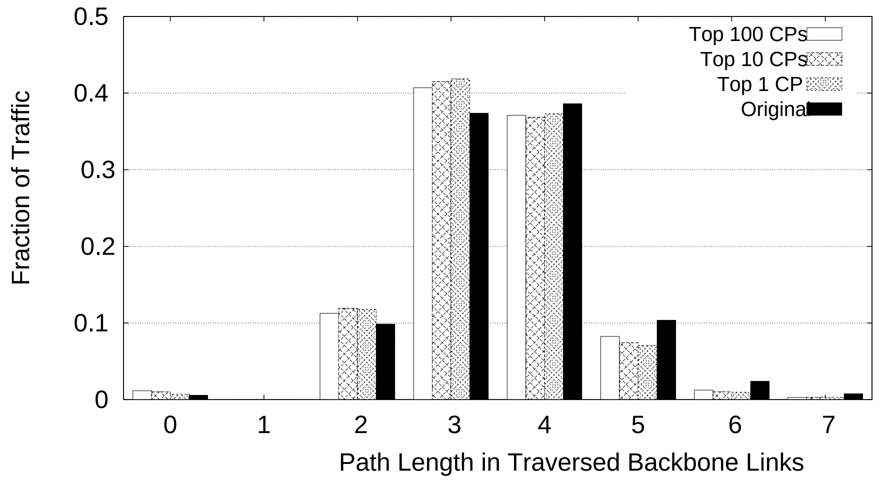


Figure 35: Backbone path length count with CaTE.

capacity upgrades and improve end-user satisfaction.

Effect on Distribution of Link Utilization. Reducing the maximum link utilization shifts traffic away from congested links. However, it should not be done at the expense of creating congestion on other highly utilized links. In Figure 34 we plot the CDF of traffic volume in ISP1 across all link utilizations, normalized by the maximum one when considering sets of the top CPs by volume. The results show that CaTE shifts the traffic away from highly utilized links to low utilized ones.

Effect on Traffic Path Length. Our results in Figure 33 (bottom) show a reduction in the overall traffic in ISP1, which can be attributed to an overall reduction of the path length. Path length reduction is an important metric for ISPs for the dimensioning of the network as well as the reduction of operational costs. To quantify this reduction in

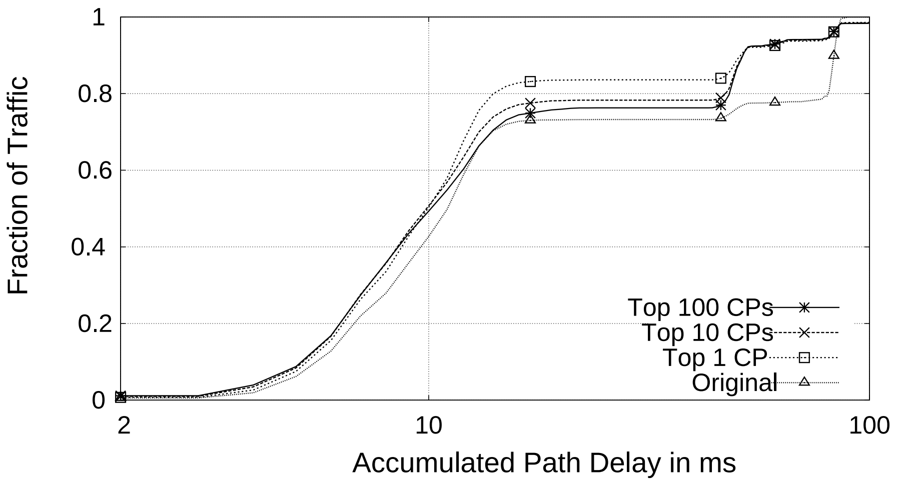


Figure 36: Improvement in path delay (in ms) with CaTE.

terms of the path length inside ISP1, Figure 35 shows the relative traffic across different path lengths inside the network. CaTE redirects the traffic towards paths with the same or even shorter length than the ones used without CaTE, only in the rare case where a longer paths yields a lower utilization, CaTE can choose a longer one. Note that there is no traffic for backbone path length equal to 1 due to the network design of ISP1. We conclude that applying CaTE to a small number of CPs yields major improvements in terms of path length.

Effect on Path Delay. Although the objective of minimizing maximum link utilization is not directly related to the reduction of path delay, the achieved reduction in path length directly affects the path delay. Figure 36 shows the accumulated path delay for the traffic that flows within ISP1, when applying CaTE. The reported numbers for the backbone path delay are relatively modest compared to the values for the access part of the network [150]. However, improving the access delay requires significant investments as it can be done mostly through changes in the access technology, e.g., from copper to fiber. When considering the end-to-end delay, the delay of the path outside the ISP's network also needs to be considered. As content infrastructures are located close to peering points [128, 126, 12], e.g., IXPs or private peerings, the delays are expected to be relatively small, especially for popular CPs. Estimating the impact of CaTE on the end-to-end performance for every application is very challenging, due to the many factors that influence flow performance, especially network bottlenecks outside the considered ISP. In Appendix 12.6.5 we show the results from active measurements conducted in the case of traffic-heavy applications, confirming the significant improvements in end-to-end delay as well as download time that can be achieved thanks to CaTE.

Summary. Our evaluation shows that CaTE yields encouraging results, even when only a few large CPs are collaborating with an ISP. In fact, even metrics that are not directly related to the optimization function of CaTE are improved. Besides significant improvements for the operation of ISP networks, the end-users are expected to also benefit from these gains. This can be attributed to the decrease of delay as well as the reduced link utilization.

12.6.2 CaTE with other Network Metrics

So far we have evaluated CaTE with one traffic engineering objective, namely, the minimization of maximum link utilization. CaTE allows ISPs and CPs to to optimize for other network metrics such as path length or path delay. To this end, we quantify the effects of CaTE when using path length and delay and compare it with the results presented in Section 12.6.1. We focus on the top 10 CPs as our results show that most of the benefits from CaTE can be achieved with this rather low number of CPs. Similar observations are made when applying CaTE to the top 1 and 100 CPs.

In Figure 37 (top) we plot the total traffic reduction when applying CaTE to the top 10 CPs with different optimization goals. The first observation is that when the network metric is path length, the total traffic reduction is the highest, with up to 15%. The total traffic reduction when optimizing for path length are close to the one achieved when the metric is delay. Optimizing for other metrics provides the expected result: the optimized metric is significantly improved, but at the cost of not optimizing other metrics as much. For example, optimizing for link utilization diminishes the benefits from path length (Figure 38 top) and vice-versa (Figure 37 bottom). Still, significant improvements can be achieved even when optimizing for another network metric and we encountered no case of significant deterioration in on of the network metrics throughout our experiments, see Figure 37 and Figure 38.

12.6.3 CaTE in AT&T and Abilene

To quantify the potential benefits of CaTE in networks with different topological structures than ISP1, we repeat our experiments for two other ISPs: AT&T and Abilene.

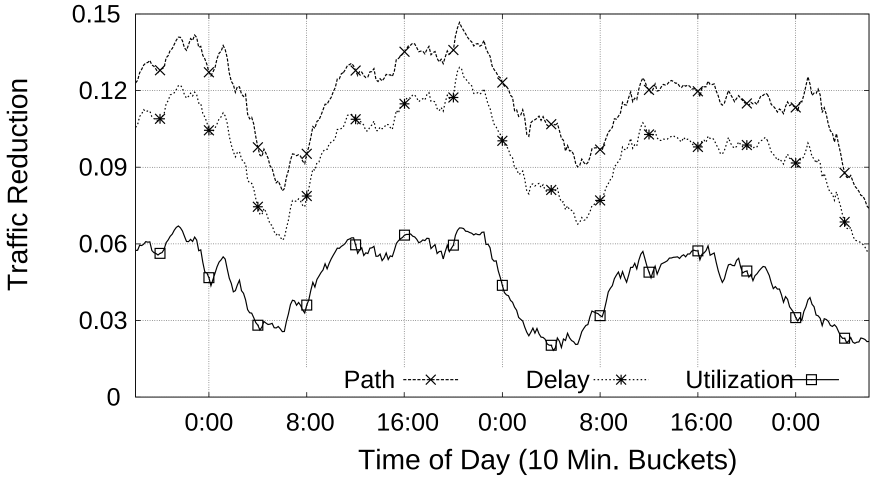
AT&T is one of the largest commercial networks. We use the topology for the US backbone of AT&T as measured by the Rocketfuel project [210, 207]. Given that no publicly available traffic demands exist for AT&T, we rely on the gravity model [192] to generate several traffic demand matrices as in ISP1.

Abilene is the academic network in the US. We use the Abilene topology and traffic demands covering a 6 month period that are both publicly available.[8]

The topology of both networks differ significantly from the one of ISP1. In AT&T, many smaller nodes within a geographical area are aggregated into a larger one. Abilene has few but large and well connected nodes with a high degree of peerings. For the application mix we rely on recent measurements in AT&T [87] and for server diversity we rely on measurements of users in these networks [12].

Figure 39 shows the cumulative fraction of normalized link utilizations for AT&T and Abilene with different optimization goals. As already done in ISP, only the Top 10 CPs are considered for CaTE, while all other traffic stays unaffected. For AT&T the benefit for the maximum link utilization is about 36% when the network is optimized for minimizing the maximum link utilization, while the median reduction in

[8]http://userweb.cs.utexas.edu/~yzhang/research/AbileneTM/



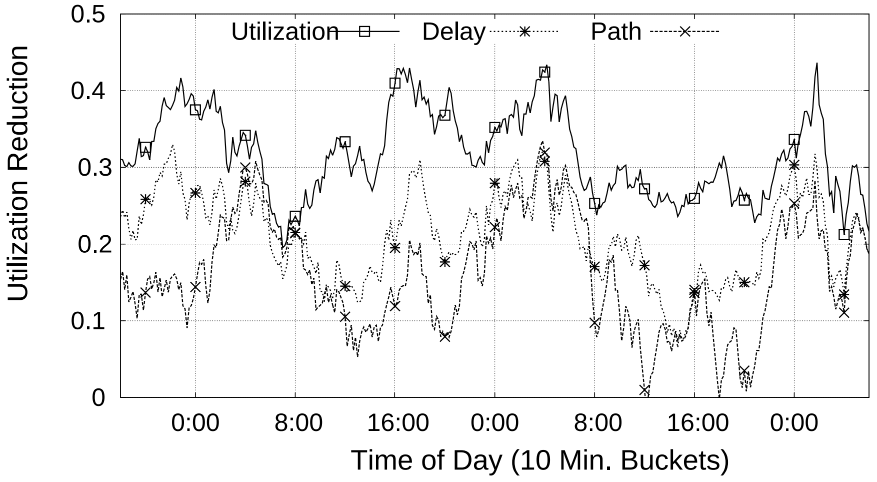


Figure 37: Total traffic (top) and maximum link utilization (bottom) reduction with CaTE and different network metrics.

terms of network-wide traffic is about 3.7%. When other optimizations are used, the benefits of CaTE regarding the link utilization minimization are approximately 12% for path length and delay. However, when looking at the median traffic reduction of these metrics, the traffic is reduced by 5.4% when path length is used, while delay achieves a reduction of 5%. In the Abilene network benefits of CaTE are more significant: 45% reduction in the maximum link utilization and 18% for network-wide traffic when CaTE optimizes for link utilization. When targeting the other two metrics, i.e., path length and delay, the results show that CaTE does not reduce the maximum link utilization. In fact, the maximum link utilizations stays constant. This is due to the structure of the network and the fact that the content is available closer, but at the cost of keeping the high utilization on some of the links. However, when looking at the median traffic reduction, both metrics manage to reduce the traffic by over 24%. These results show that CaTE is capable of targeting different optimization goals in different network structures and is able to optimize for different metrics.

It is worth noting that for AT&T 40% of the links have a normalized link utilization less than 10% while the remaining link utilizations are distributed almost linear. This distribution fits the structural observations made for the AT&T network: many links from smaller nodes are aggregated into larger ones. This also explains why the benefits for AT&T are smaller, since such a structure reduces the path diversity. Turning our attention to Abilene, we attribute the higher reduction of maximum link utilization and network-wide traffic to the non-hierarchical structure of the network and a higher ratio of peering locations. Applying CaTE to both AT&T and Abilene networks where the network metric is delay or path length shows similar behavior of CaTE as it does in ISP1.

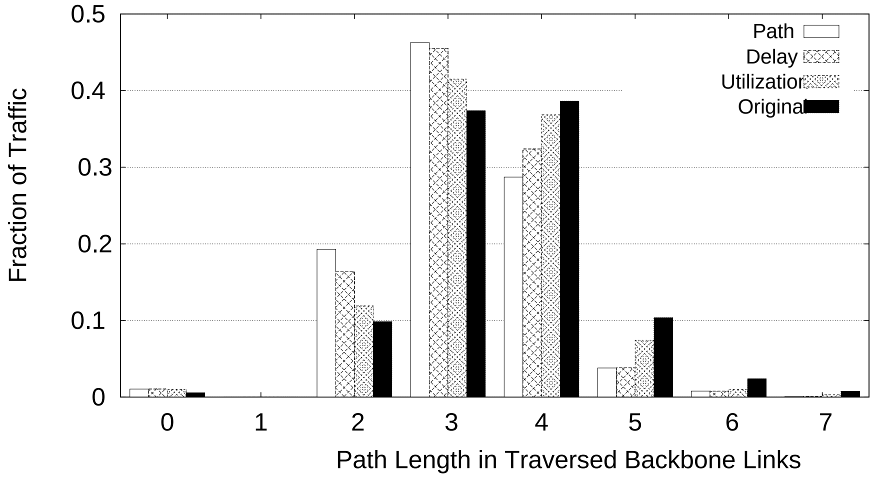
12.6.4 CaTE and Popular Applications

Today, the launch of new content hosted on CPs such as high definition video or others that share flash-crowd characteristics, is not done in coordination with ISPs. This is challenging to ISPs that have to deal with rapid shifts of traffic volume as currently deployed traffic engineering tools are too slow to react to rapid demand changes. Furthermore, the end-user experience for popular applications is far from optimal as application designers have limited means to optimize the end-to-end delivery of content [126]. Both ISPs and applications would benefit from the improved traffic engineering capabilities of CaTE. We believe that CaTE can act as an enabler for ISP-application collaboration.

For example, Netflix, a very popular application that delivers high quality videos to end-users, relies on commercial CDNs such as Level3 and Limelight to improve the content delivery. Today, Netflix is only available in North and Latin America. However, Netflix has announced that it will be launching its services in Europe early 2012. To quantify the effect of Netflix coming to Europe, we use our simulation to estimate the effect on ISP1. We run a series of experiments, assuming that the traffic of the CPs hosting Netflix will increase 20-fold. Our results show that with CaTE, the total HTTP traffic volume is reduced by up to 8% and the utilization of the most utilized link by 60%.

12.6.5 Active Measurements in ISP1

The CaTE evaluation in Section 12.6.1 does not allow us to argue about end-user performance, as it is based on simulations. To this end, we complement our previous network-wide simulations with active measurements. Over a period of one week, we repeatedly downloaded a 60MB object from one of the major CPs. This CP is an OCH distributed across 12 locations. The downloads were performed every two hours, from each of the 12 locations. Additionally, mapping requests were issued every 200ms to find out the dynamics in the server assignment of this CP. Figure 40 shows the distribution of total download times when the CP assigns end-users to its servers (”original”) and compares it to the download time that would be observed if CaTE had been used.



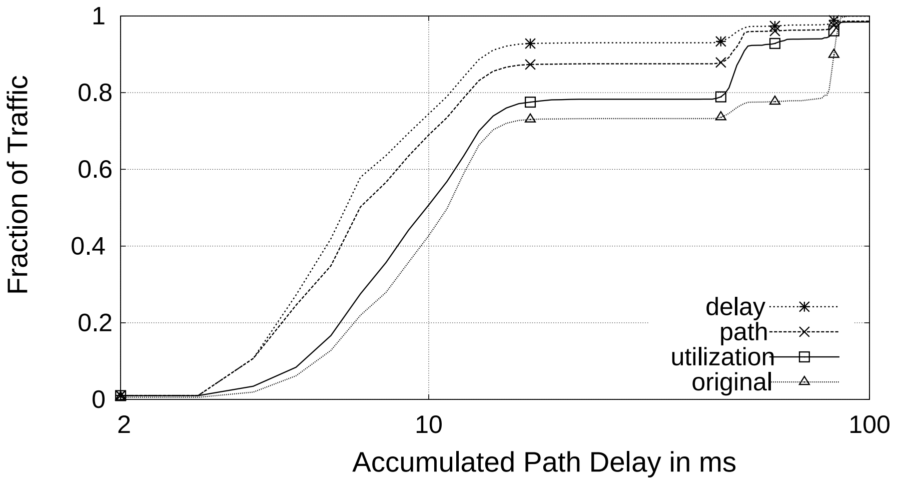
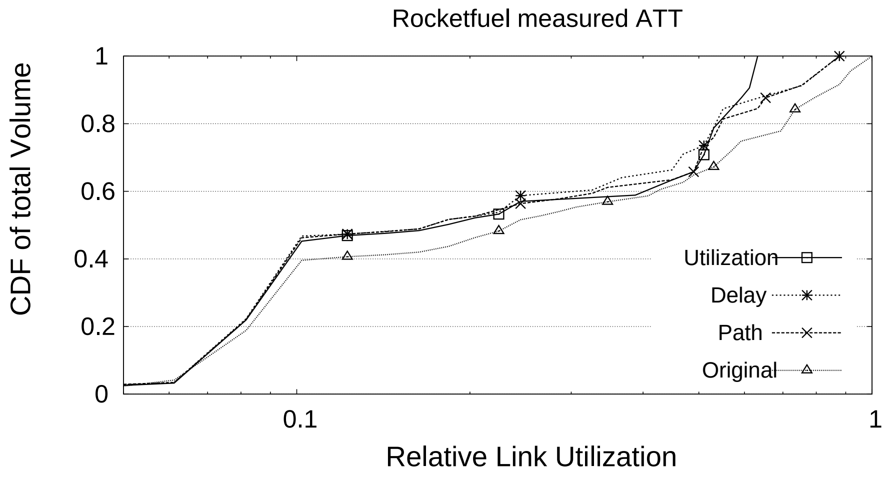


Figure 38: Backbone path length (top) and accumulated path delay (bottom) with CaTE and different network metrics.

We observe that more than 50% of the downloads do not show a significant difference. This happens when congestion is low, e.g., during non-peak hours. For 20% of the downloads, we observe a significant difference in the download times, mainly during peak hours. This confirms our observation that CaTE is most beneficial during peak hours.

12.6.6 Case Study: Netflix in ISP1

Netflix, a very popular application that delivers high quality videos to end-users, relies on commercial CDNs such as Level3 and Limelight to improve the content delivery. Today, Netflix is available in North and Latin America, and is announced to arrive in



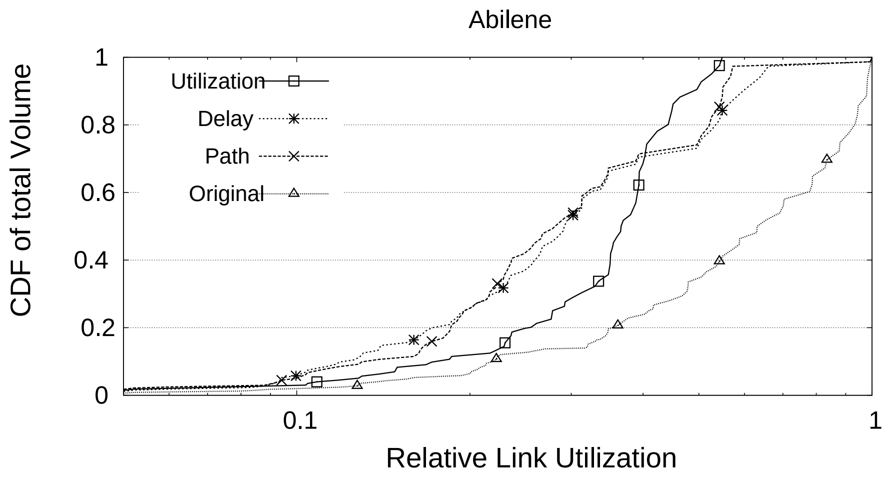


Figure 39: Link utilization improvements after applying when using CaTE in AT&T and Abilene.

the UK soon. Recent studies show that Netflix is responsible for more than 30% of the peak downstream traffic in large ISPs [195]. Consider the scenario where Netflix is launching its service in the large European ISP1 we described in Section 12.6. If the launch happens overnight, ISP1 would have to deal with a huge amount of highly variable traffic, which would have significant implications on the operation of ISP1. With CaTE, the traffic of Netflix can be spread across the ingress points of ISP1. This will limit the negative consequences imposed by additional traffic for the CP delivering Netflix as well as for ISP1 and thus avoids a deteriorated end-user experience.

To quantify the effect of Netflix being deployed in Europe, we simulate the launch of Netflix in ISP1, assuming that the CP currently hosting Netflix increases its traffic 20-fold, while keeping the distribution of the requests. Next, we generate a new set of

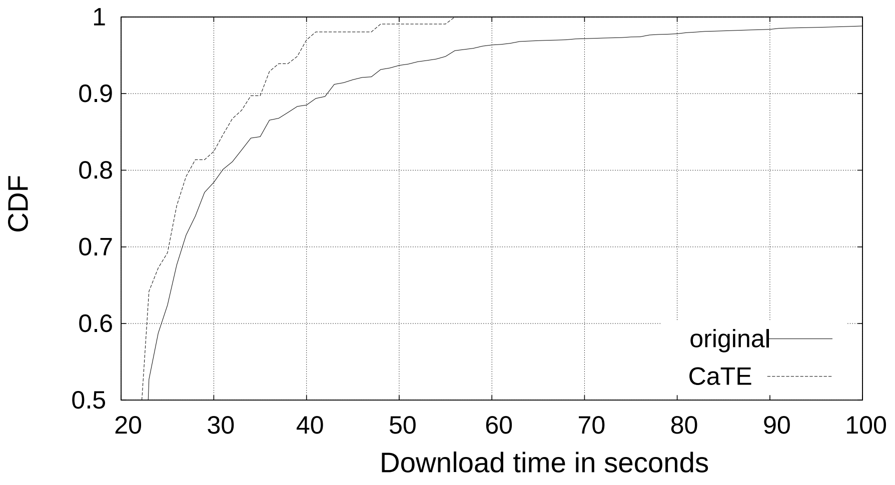
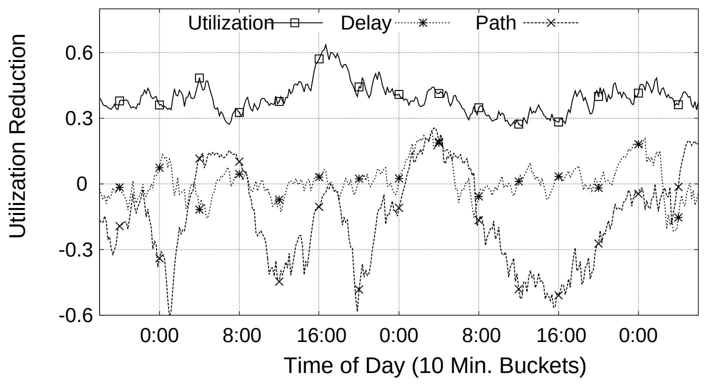


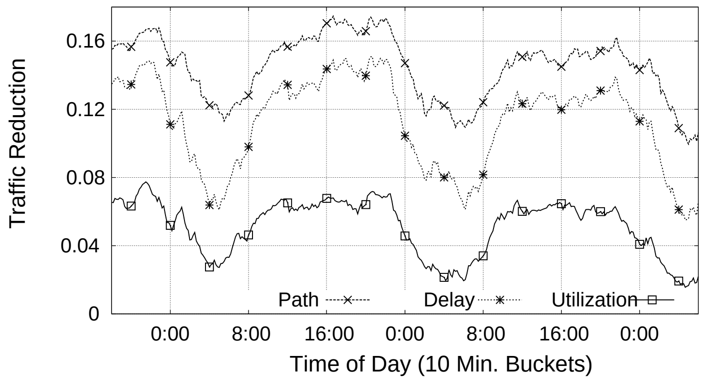
Figure 40: Distribution of download times of a CP.

traffic demands for CaTE accordingly. We consider the the top 10 CPs by volume for CaTE, and show the benefits when optimizing for different metrics.

Our results show that with CaTE, the utilization of the most utilized link can be reduced by up to 60% (see top of Figure 41), the total HTTP traffic volume can be reduced by 15% (see middle of Figure 41) and traffic can be shifted towards shorter paths inside the network of ISP1 (bottom of Figure 41). However, when considering all metrics, we observe that not all metrics can be optimized to their full extend at the same time. For example, a reduction of traffic in the order of 15% would actually increase the utilization on the highest loaded link by 60%. This indicates that the optimization function employed by CaTE needs to be carefully chosen to target the most important metrics when deploying CaTE inside a network. Nonetheless, if minimizing the maximum link utilization is chosen as the optimization function for CaTE, benefits in all metrics can be observed.

Internet applications such as Netflix are in a position to negotiate how they should be deployed in order to improve end-user experience and not disturb the operation of ISPs. CaTE can be used to identify the best peering points between the CPs that deliver Netflix traffic and the ISPs that receive its traffic. In addition, ISPs might offer better peering prices if the CPs hosting Netflix are willing to provide a higher diversity in the locations from which the traffic can be obtained. This would lead to a win-win situation where Netflix can offer better service to its users, the CPs achieve reduced pricing on their peering agreements, and ISPs can compensate the reduced peering revenue through more efficient operations.





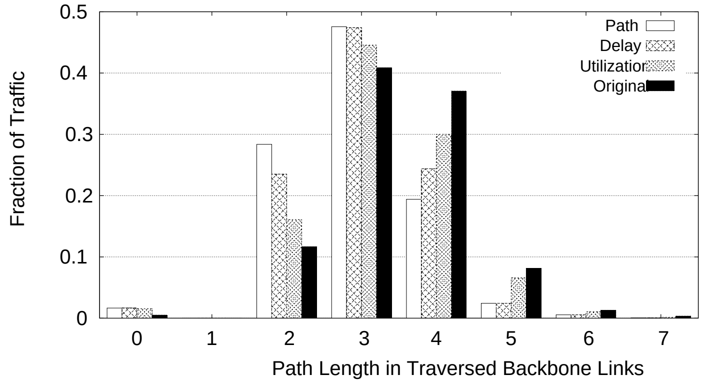


Figure 41: Projection of reduction in link utilization (top), reduction in overall network traffic (middle) and fraction of volume by path length (bottom) if Netflix is launched in ISP1.

13 Future of Collaboration

PaDIS and CaTE are designed to enable cooperation between CDI and ISPs for the already deployed servers. Recent advances in virtualization offer CDNs additional degree of freedom to the CDN to scale-up or shrink the footprint on demand and thus allows it to deliver content from additional locations inside the network. This can be done either by jointly deploy and operate new servers with the ISPs. In this section formally introduce the design of on-demand services motivated by the recent announcement of major ISPs to support generic hardware-network appliances, also referred to as microdatacenters, and offer them to application, service, and content providers. We also provide the design, implementation, and evaluation of a system to orchestrate the deployment of on-demand service inside microdatacenters by utilizing the view of the ISP for the network and additional computation and storage resources inside the network.

13.1 The New Cloud: Microdatacenters Deep Inside the Network

Applications are increasingly relying on direct interactions with end-users and are very sensitive to delay [136]. Indeed, transaction delay is critical for online businesses [121]. Network delay and loss are important contributors. Today large-scale service deployments are restricted by limited locations in the network, e.g., datacenters, private peering locations, or IXPs. These locations are not necessarily ideal [140]. We point out that *selection of service location is critical and currently not flexible enough*. Services should be located close to, in terms of network distance, the clients. Since client demands are volatile and change across time, SPs need agility [47]. They can improve their service quality by quickly allocating, de-allocating, and migrating resources on-demand where and when they are needed. Indeed, since delay and packet loss are among the critical metrics it may be necessary to install the service deep inside the network. Thus, current cloud services do not suffice. This approach is taken by many ISPs for IPTV services. However, this option is not yet available for non-ISP service providers.

Currently, most services and networks are run by independent entities with different and often conflicting objectives. Such *uncoordinated service and network operation has no winner.* Lack of information about the other entity leads to suboptimal performance and resource allocation for both the SP and the ISP. For example, SPs implement sophisticated methods to infer network conditions to improve perceived end-user experience [170], e.g., active measurements within the ISPs. Yet, the information gleaned from these measurements is already available with far greater precision to the ISP. On the other hand, ISPs continuously upgrade their infrastructures without being able to efficiently engineer the SP traffic flows [183]. Today, cooperation and/or partnership between providers is limited to, e.g., peering or lately service provider interconnection. This level of cooperation is too narrow to reduce operational costs, improve end-user experience, circumvent bottlenecks, handle flash crowds, and adapt to changing network conditions and user demands. This has led to discussions on how to improve communication, e.g., within the IETF ALTO and CDNi working groups.

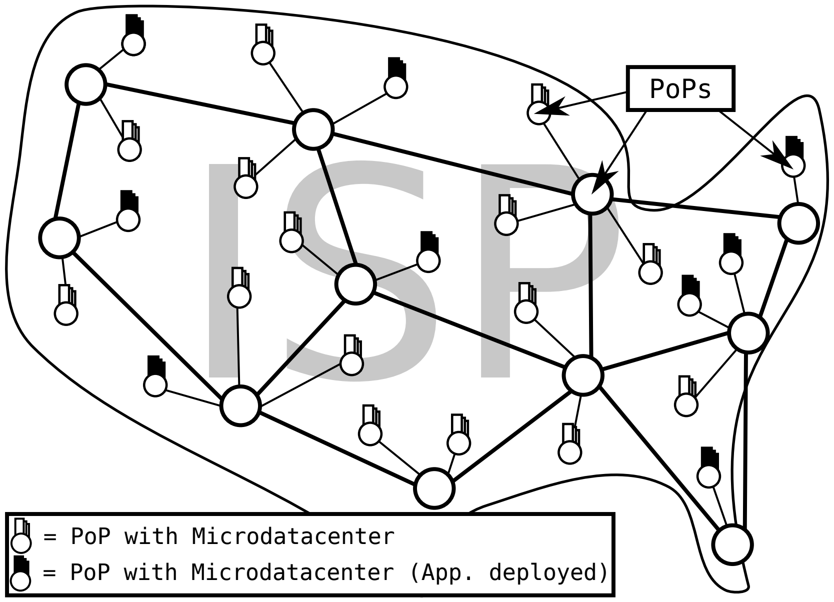

PoPs
ISP
= PoP with Microdatacenter
= PoP with Microdatacenter (App. deployed)

Figure 42: Microdatacenters in an ISP with NetPaaS enabled

13.1.1 The ISPs Proposal

To overcome the above mentioned obstacles in service deployment and operation major ISPs, including AT&T, Deutsche Telekom, and Telefonica, have proposed the use of cloud resources consisting of general purpose appliances that are co-located at network aggregation points inside the ISP and are in the process of being deployed to extend service provider infrastructures. With the convergence of computing, storage, and communications, the acceptance of cloud services, and the ever increasing demand for popular services, ISPs are moving towards deploying general-purpose computing and storage infrastructures in their points of presences (PoPs). Henceforth, we refer to these as *microdatacenters*. The description of the functionality of these microdatacenters is provided in a white paper that appeared in the SDN and OpenFlow World Congress in October 2012 and signed by 13 of the largest ISPs [5]. Microdatacenters can be also the technical solution needed to materialize recent alliances of major CDNs, such as Akamai with large ISPs in the area of content delivery [1, 2, 3].

Figure 42 illustrates the basic idea. The ISP can offer *slices* within their microdatacenters that can be leased by the SP—using our proposed mechanism—based on their needs. This approach leverages recent advances in virtualization technology, and flexible billing models, such as pay-as-you-go, to provide cost-efficient and scalable service deployment. This enables unprecedented flexibility. Moreover, the diversity of available service locations within the network can be used to improve end-user experience and makes it possible to launch even more demanding applications, such as interactive ones. On-demand service enables SPs to rely on a fixed infrastructure deployment for their baseline operation and then scale it up by dynamically allocating resources closer to end-users. It also lowers the burden of entrance in the service market for smaller SPs—at one extreme they only use the on-demand service. Moreover, it is an enabler for a marketplace for Internet resources.

13.1.2 Microdatacenter Specifications

Microdatacenters consist of one or more racks of off-the-shelf hardware deployed in general purpose rack space at network aggregation points. State-of-the-art solutions have been proposed by the VMware/Cisco/EMC VCE consortium [224], and are also offered by other vendors, such as NetApp and Dell. These solutions are general-purpose and provide a shared infrastructure for a large range of applications. They typically consist of two basic components: hardware and management software.

Hardware: Typical microdatacenters include *storage*, *computing*, *memory*, and *network access* components. Storage consists of tens of Terabytes with an ultra fast controller providing I/O throughput in the order of hundreds of Gbps. The storage component is connected to the Internet through multiple Gbps interfaces and to the computing component with Gigabit Ethernet switches. Typically, a rack includes up to 40 physical multi-core blade servers as well as two routers and two switches in mesh configuration for redundancy and load balancing.

Management software: Each vendor offers a set of management tools not only for administering the components but also to create resource slices and to delegate the operation of the slices to external entities. This can be done per-server or via hardware supported virtualization[9]. The management software is also responsible for storage allocation and handling network resources, including IP address space. In addition, the management tools come with a monitoring interface that allows the ISP to monitor the utilization of the overall microdatacenter as well as the information for each slice that can be shared with the external entity.

Thus, it is possible for an ISP to allocate resource slices consisting of computing, storage, memory, and network access in a microdatacenter and then delegate the operation of the slice to a SP. This is what we refer to as the *ISPs cloud service* which is realized via resource slices in microdatacenters throughout the ISPs infrastructure.

13.1.3 Microdatacenter Network Footprint

Most ISPs' networks consist of an access network to provide Internet access to DSL and/or cable customers, as well as an aggregation network for business and/or VPN customers. Routers at this level are often referred to as *edge routers*. The access and aggregation networks are then connected to the ISP's backbone which consists of *core routers*. *Border routers* are core routers that are used to connect either to other networks or to co-location centers. Opportunities to deploy microdatacenters exist at each level: edge, core, or border router locations.

[9]For example, para-virtualization [56] presents the VM with an abstraction that is similar but not identical to the underlying hardware.

The advantage of deploying service infrastructure only at the core router locations is that there are a few large and well established locations. This is also a disadvantage as location diversity is limited. Location diversity is highest at the edge router locations. However, it might not be possible to deploy a microdatacenter immediately, i.e., due to limited space and/or power at the facilities, or due to cost. These locations, however, minimize the distance to the customers. Border router locations are often a subset of core routers, hence they inherit the same advantages and disadvantages.

The SPs advantage of using an ISP cloud service vs. a public cloud service is the chance to minimize the distance to the end-user. Moreover, on-demand service deployment allows the service provider to control the location of the slices and ensures that there are no major network bottlenecks. Through cooperation a good mapping of user demands to resource slices can be ensured.

13.2 On-Demand Service Design

An *on-demand service* is a service of the ISP (see Figure 42) that enables service providers to use a hosting infrastructure that scales or shrinks according to end-user demands, so as to minimize its capital expenditures and operating costs, as well as the distance between its hosting infrastructure and the source of the demand. Moreover, it offers an interface that enables the SP to map user requests to appropriate slices in order to maximize slice utilization and minimize the distance between the end-user and the slices.

Today's services have no choice but to use a scalable design, e.g., one that scales with the number of users. A scalable design is necessary for a service to avoid being a victim of its own success, to be capable of handling flash-crowds [111], and to compensate for the limitations of current hardware. The motivation of SPs include (a) reducing infrastructure costs, (b) properly scaling and dimensioning the service (which is difficult to do before launching), and (c) improving the end-user experience, which can be a key factor that determines the success or failure of a service [121].

Definition 1: ISP On-Demand Service. The ISP on-demand service is a service offered by the ISP and uses as its base unit of resource allocation the notion of a microdatacenter *slice*. It is the ISP's task to allocate/de-allocate the slices since it operates the microdatacenter. The SP requests slices based on its clients demand. When the slice is allocated to the SP, the service can be installed on the slice. From that point on, the SP fully controls the operation of the service installed in the microdatacenter. Negotiation about slices are done via the *on-demand service interface* through which SP demands are matched to the ISPs resources. How to map demands to resources in an efficient manner is the task of the ISP and part of the *on-demand service realization*. In addition, the interface allows for access to the billing information. Moreover, the *on-demand service interface* enables the mapping of user requests to appropriate slices.

The above mentioned use of microdatacenters is in-line with the available primitives of private and public clouds operated in large-scale datacenters, e.g., [22, 157].

13.2.1 Microdatacenter Slice

Based on our description of microdatacenters in the previous sections we define a slice as follows.

Definition 2: Slice. The slice of a microdatacenter is a set of physical or virtualized resources of a specific capacity, for each of the resources of the microdatacenter. The slice is delegated to the service provider that can install and operate its service using the resources of the slice.

A slice can be a 1-core server with 2 GB RAM, 30 GB storage, a 1 Gbps Internet access bandwidth, 2 public IPs—an actual physical resource. Alternatively, it can be a VServer with 2GB and 1 Gbps Internet access bandwidth, 1 public IP, and a pre-installed OS—a virtual machine of a specific type. With the current management and virtualization tools available from microdatacenter vendors, it is possible to allocate/deallocate slices on-demand in with unprecedented degree of freedom, e.g., [26] and references within.

13.2.2 On-Demand Service Realization

Based on the above specification of on-demand service, the ISP has to implement two functions to offer its *on-demand service*: *mapping of service provider demands to slices* and *assigning users to slices*.

Note, the time scales at which these two services are expected to be used differ significantly. The first one allows the service provider to flexibly allocate and de-allocate its slices based on its forecast of demands, in those locations where it wants them. We foresee that requests for slices are not issued individually but rather collectively on a time scale of tens of minutes or hours.

The SP provides the ISP with a set of demands for slice resources, predicted demand locations, desired slice locations, as well as optimization criteria. The ISP then has to map the demands to its microdatacenter resources. We expect that the major degree of freedom that the ISP uses to jointly optimize performance is the desired slice location. We refer to this optimization problem as the `slice location` problem. If the desired slice locations are fully specified or the predicted demand locations are missing, the `slice location` problem becomes trivial and the ISP only grants or denies the request.

At the second time scale, the ISP can help the SP in assigning users to slices. Since the service is offered at multiple locations, a good assignment of users to slices impacts not only the load on the network but also the network delay and packet loss, which are key contributors to the user experience. Jointly optimizing this mapping is therefore of interest to both the ISP and the service provider. The CDI can query the ISP for each request on where to map it, based on the current set of slice assignments and service loads. The ISP then uses its network information to propose possible slices. We refer to this problem as the `user-slice assignment` problem. We postpone the formal definition and possible solution approaches of the `slice location` and `user-slice assignment` problems for the following section.

Another degree of freedom on-demand service offers to the CDI is auto-scaling. While it is quite feasible to dimension applications, flash-crowds or device failures are hard to predict. To this end, a CDI may allow on-demand service to create replicas if its monitoring indicates that the capacity of the service at a given location is or will be exceeded. To realize this service, the ISP needs to constantly monitor resource availability and if necessary migrate or suggest the creation of additional slices. Moreover, it has to allow the CDI to monitor the utilization of its slices.

13.2.3 Service Interfaces

The ISP offers four interfaces to the service providers:

Resource discovery: Using this interface the CDI requests information about resources, e.g., about available locations for slices and if in principle slices are available at those locations at what price.

Slice allocation: Using this interface the CDI requests slice allocation within a certain cost limit.

User-slice assignment: Using this interface the CDI requests recommendations for user demand to slice mapping.

Monitoring and billing: Using this interface the CDI monitors the status and cost of its slices.

In Section 13.4 we give specific examples of how these service interfaces can be used by a CDI and ISP to cooperate in order to improve their services.

13.2.4 Billing

It is important for the CDI to minimize and track the cost of its use of on-demand service. Depending on the scale of the services, the service provider has to pay the usual price or negotiate bilateral agreements with the ISP. Using the resource discovery interface, it estimates the cost of slice allocation at possible locations. Using the slice allocation interface, it can bound the total cost of the request.

We expect that the billing of a slice allocated via on-demand service follows that of large-scale datacenters. This means that there is an installation cost and a usage cost. The installation cost applies to a single slice in a microdatacenter and is charged only once or over long time intervals, e.g., hours, and is fixed. The installation cost typically increases if additional licenses have to be leased, e.g., software licenses. The installation cost can depend on the location of the microdatacenter that hosts the slice or the time-of-day.

The usage cost follows a pay-as-you-go billing model and charges for the usage of different resources assigned to a slice. The billing among different resources in the same slice can be quite diverse. The slice can use expensive resources such as bandwidth or cheaper ones such as CPU.

For example, a slice may have a \$0.01 per hour installation cost and a usage cost that depends on its use of various resources, e.g., \$0.02 per real CPU usage per hour, \$0.001 per GByte stored per hour, and \$0.001 per Gbps outgoing traffic per hour. If the slice is idle, then only the installation cost is charged. Note, that if the slice is used for a short period within the allocation time, e.g., a few minutes, then the charge may apply to the minimum billing granularity.

To minimize the cost of deploying an on-demand service, the service provider can change its total slice demands as well as its slice specifications dynamically. Moreover, it can relax the slice specifications to reduce overall cost of its service deployment.

13.3 On-Demand Service Optimization

In this section we propose solutions for the `slice location` and `user-slice assignment` optimization problems. We show that the `user-slice assignment` problem reduces to the facility location problem and propose heuristics based on linear programming and local search.

13.3.1 The Slice Location Problem

Let a directed graph $G = (V, E)$ represent the ISP network given by a router set V and a set of links E. Let $L \subseteq V$ be the set of locations in the network where the ISP operates microdatacenters. Let S be the set of available slices at these locations that can potentially host the service. Let c_{ij} be the delay between a slice $s_i \in S$ and clients attached to v_j. Also, let u_i, f_i and r_i denote the slice capacity, the installation cost and the the unit price for resource utilization of slice s_i respectively. Finally, let $d(v_j)$ denote the service demand of users attached to v_j. Let us now formally define the `slice location` problem:

Definition 3: Slice Location Problem (SL). Given a set of available slices S with associated installation and usage cost, and a set of demands $d(v_j)$, $\forall v_j \in V$, select a subset of slices $F \subseteq S$ so as to minimize the total cost of installing and operating the slices, as well as offering the service close to the client demand.

SL is the capacitated facility location problem (CFL) [123], where the facilities correspond to the slices and can be co-located. Moreover, to model the cost of operating a slice, the distance between a slice and clients is increased linearly by the unit price for the usage of a resource in this slice (p_i). We focus on the CFL with *splittable demands*, which allows demand to be allocated to more than one facility. This is a reasonable assumption as requests from different users that are attached to the same router can be served by different slices. This allows better utilization of the microdatacenter resources and thus reduces usage cost. The total number of slices that are allocated are part of the solution of the optimization problem. If k is an upper bound on the number of slices that a service can install then SL is the capacitated k-median problem with splittable demands. We refer to this version of the `slice location` problem as k-`slice location`.

Both the capacitated facility location and the k-median problem are NP-hard [123] and therefore both versions of the `slice location` problem are as well. Thus, we propose heuristics based on linear programming and local search.

Linear Programming Formulation. We formulate SL as an integer linear program, and relax the integrality constraints to obtain a linear program (LP):

$$\texttt{min} \qquad \sum_i f_i y_i + \sum_j \sum_i r_i x_{ij} d(v_j) c_{ij}$$

$$
\begin{aligned}
\texttt{s.t.} \qquad & \sum_i x_{ij} \geq 1 && \forall j \\
& x_{ij} \leq y_i && \forall i, j \\
& \sum_j x_{ij} d(v_j) \leq u_i y_i && \forall i \\
& y_i \leq 1 && \forall i \\
& x_{ij}, y_i \geq 0 && \forall i, j.
\end{aligned}
$$

Variable y_i is boolean and indicates if a slice s_i is selected ($y_i = 1$) or not. Variable x_{ij} indicates the fraction of demand $d(v_j)$ that is assigned to slice s_i. The first constraint states that the demand has to be satisfied and the second one that the demand can be assigned only to selected slices. The third constraint states that the total demand served by a slice can not exceed the slice capacity. The fourth constraint states that a slice can be served only once and the last one that no negative fraction of demand can be assigned or no unavailable slice can be assigned. To solve the k-`slice location`, one more constraint must be added: $\sum_i y_i \leq k$.

The solution of the above LP can be found in polynomial time [117]. A number of techniques have been proposed to find a solution faster and include rounding [44, 139] and primal-dual methods [108]. The above LP can be extended to tackle the resource requests of multiple CDIs at the same time.

Fast Heuristic: Local Search. The LP solution may be too slow for on-demand service as its runtime scales with the number of microdatacenters and CDIs. Therefore, we consider alternative heuristics. The best heuristic for the facility location and k-median problems is *local search* [118]. In the setting of the `slice location` problem, the local search heuristic starts with an initial feasible allocation of slices. Then, it incrementally improves the solution either by evaluating neighboring solutions, e.g., by adding, removing, or swapping one or more slices. Once the local search finds a stable set of slices it has found its local optimum. For the `slice location` problem with splittable demands, a local search heuristic that permits adding, dropping, or swapping one slice has been shown to give good approximations [28].

13.3.2 Assigning Users to Slices

Once slices have been selected, users have to be mapped to slices. We propose that the CDI utilizes recommendations from the ISP to enhance the end-user experience. For this, the CDI identifies the slices that can satisfy the requested demand and then ranks them based on its own criteria. Let $S' \subseteq S$ be the set of possible CDI slices and $\{x_i\}$ and $\{y_i\}$, $i \in [1, |S'|]$ a ranking of the slices from the viewpoint of the CDI and ISP, respectively. The CDI then assigns the user to the slice that minimizes the rankings of the CDI and the ISP. We formally define the joint optimization problem of assigning service users to slices hosted in ISP microdatacenters as `user-slice assignment` problem:

Definition 4: User-Slice Assignment Problem. Given a new demand request d_r from an end-user that originates at v_j, and a set of slices $S' \subseteq S$ that can satisfy the request, assign the end-user to the slice $s_i \in S'$ that optimizes the CDI criteria, e.g., minimum cost, while considering the ISP's recommendations.

This problem corresponds to the online restricted machine load balancing problem (RMLB). The RMLB considers a sequence of tasks each of which can be executed on a subset of the available machines. The goal is to assign each task upon arrival to one of the machines that can execute it in such a way that the total load is minimized. In our setting, machines correspond to slices and tasks to requests. While in principle requests can be assigned to all slices, the partnership agreement between CDI and ISP may specify how strictly the slices recommendations of the ISP have to be followed. For the RMLB, the greedy algorithm that assigns users to highly ranked slices by the ISP with the least load, has been shown to be optimal [32]. Thus, the heuristic outlined above is expected to converge to reasonable solutions.

13.4 Network Platform as a Service (NetPaaS)

Next, we discuss the prototype system that has been proposed to materialize the On-demand service, Network Platform as a Service (NetPaaS). NetPaaS leverages the view of PaDIS and also utilize the knowledge about the status of the microdatacenters within the network. NetPaaS is also able to map the requests of CDIs to available microdatacenters to better match the demand with the resources inside the network. The granularity at which they are exchanged via the service interface. We also outline several possible protocols for the service interfaces. We focus on resource discovery, slice allocation, and user-slice assignment. We do not discuss monitoring and billing because they can be realized today using techniques similar to those in use by current public clouds, e.g., [26].

Recall that our assumption that the time scales at which the two principle components of on-demand service operate are different. On the one hand, resource discovery and slice allocation are expected to be done on time scales of tens of minutes, hours, or even days. On the other hand, user-slice assignment potentially happens on a per user request basis. Accordingly, the protocols differ. We propose to use out-of-band protocols for the first two service interfaces and in-band protocols for the third one.

13.4.1 Resource Discovery

The purpose of resource discovery is to provide the CDI with the ability to gather information about the resources offered by the ISP. Accordingly, we have two message types: CDI_Discovery_Request and ISP_Discovery_Response.

CDI_Discovery_Request: Is issued either without and with arguments. In the first case the response is the set of resources that are offered. In the second case the responds contains details about the resources named in the argument.

ISP_Discovery_Response: List of available resource or details about the resources specified in the argument.

So far we have not outlined at what granularity and specificity the resources are requested. This depends on the agreements between the CDI and the ISP. For example, the ISP may have no problem revealing its microdatacenter locations to a major CDI.

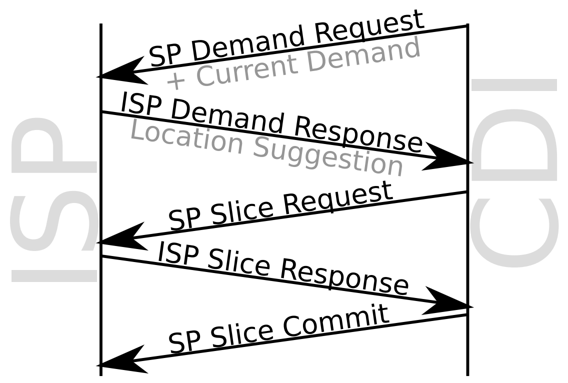


Figure 43: Slice allocation message exchange.

However, it may not want to share this information with an untrusted CDI that wants to run a single slice. For the latter, the region in which the microdatacenter is located might well suffice.

With regards to granularity, the ISP can specify which type of servers it is offering in each microdatacenter region, as is common for public cloud services [22], unless another agreement is in place that enables access to more specific information. With regards to the availability and/or price, the ISP can either return a base price, including installation and usage cost, to indicate that resources are available or offer an auction-based system. In the latter case, the discovery request returns information about past prices.

13.4.2 Slice Allocation

Slice allocation enables the CDI and ISP to cooperate for allocating slices in microdatacenters close to the end-user that are able to satisfy the demands. We envision five message types: CDI_Demand_Request, ISP_Demand_Response, CDI_ Slice_Request, ISP_Slice_Response, CDI_Slice_Commit. The first two message types enable the cooperation between the CDI and the ISP to allocate slices at appropriate microdatacenter by utilizing network information, see Figure 43. The last three messages enable a three way handshake between the CDI and the ISP to verify that the ISP is able to provide a specific slice for the CDI.

CDI_Demand_Request: Is submitted by the CDI to the ISP and contains a summary of the hardware resources the CDI wants together with optimization criteria, constraints, and a demand forecast, e.g., per region or per network prefix. Possible optimization criteria are to minimize network distance or the cost. The constrains include: number of locations, minimum resources per slice, etc.

ISP_Demand_Response: The ISP returns a set of proposed slice configurations and their price. It computes these by solving the `slice location` problem.

CDI_Slice_Request: The CDI either selects, based on its criteria, a set of proposed slices as returned by the ISP_Demand_Response, or it completes a specification of a slice set request using information it retrieved via the resource discovery interface. In addition, the request contains a maximum cost.

ISP_Slice_Response: Upon receiving a CDI_Slice_Request, the ISP checks if it can offer the set of slices at the requested cost. This involves solving another version of the `slice location` problem. If possible, the ISP returns the price otherwise it declines the request. At this step, the ISP reserves the requested resources to guarantee their availability. Note, the ISP does not have to return precise slice definitions, e.g., instead of returning that slice x should be located in microdatacenter y attached to router z it only returns slice x should be located in region xyz.

CDI_Slice_Commit: This step confirms CDI_Slice_Requests. Upon receiving the commit from the CDI, the ISP allocates the slices and delegates their control to the CDI.

Now we discuss different ways in which a CDI and an ISP can cooperate using the above messages. These ways differ in which information is shared and with whom.

Minimum information exchange: The CDI uses the information from the ISP_Demand_Response for queries via CDI_Demand_Request with a uniform distributed demand vector. The responses include slice candidates with servers having specified hardware profiles and in specific regions. Then, the SP scales the suggested slices according to its demand locations and uses the CDI_Slice_Request message to check if the ISP can offer it and at what price. Once it has found a satisfactory configuration it can use the CDI_Slice_Commit message to request the necessary slices.

Partnership 1: The CDI uses CDI_Demand_Request with a scaled demand CDI selects one of these and uses the ISP_Slice_Request message so that the ISP can reserve the resources. Upon successful reservation, the CDI_Slice_Commit message confirms the allocation of the slices.

Partnership 2: The CDI uses the SP_Demand_Request without a demand vector but with specific resource requests. The ISP response specifies candidate microdatacenters with available resources. Then, the SP uses its version of the `slice location` problem to find possible slice sets at a subset of these locations. Then, the SP uses the ISP_Slice_Request message to see if the ISP can offer it and at what price. Once it finds a satisfactory configuration it uses the SP_Slice_Commit message to allocate the slices.

The first scenario corresponds to the minimum information that has to be exchanged in order to reach a consensus on the locations and specification of the slices. The latter two summarize different forms of possible cooperations that can be agreed upon in bilateral agreements.

So far, we have assumed that there are no preallocated slices. However, this is typically not the case, and the actual task is to augment a preexisting set of slices in such a way as to best serve the predicted demand for the next time period. To enable this, another argument to each message request can be provided, indicating a set of already allocated resources and a penalty value for deallocating slices in one location and allocating them in another. This penalty is needed as part of the optimization problem. Basically, it indicates up to which point it is preferable to keep a suboptimal

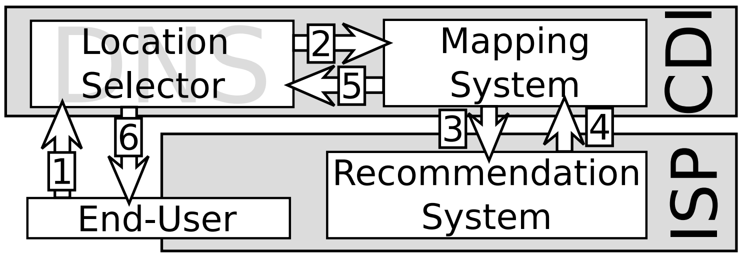


Figure 44: User-slice assignment schematic.

location because of stability of resource allocation vs. when to migrate the service to a new location. Based on the returned information, the SP has the option of either moving slices using VM migration or to de-allocate and allocate new slices.

The ISP microdatacenter can offer VM migration and/or consolidation[10] with keeping the IP addresses only within the same microdatacenter location. Across microdatacenters it may only offer migration with tunnels which requires the SP to temporarily operate two slices at both locations. However, the old one is a good candidate for consolidation so that it is possible to reduce the allocated resources to a minimum within a microdatacenter once all new requests are served by the newly allocated slices. Thus, if an ISP offers service consolidation, one option for SPs that want to use diverse sets of microdatacenters is to always keep a minimal slice active at each location and expand or shrink it according to the demand.

13.4.3 User-Slice Assignment

The purpose of the user-slice assignment interface is to enable small time scale interactions between the SP and the ISP, to ensure that end-user demand is mapped to the appropriate slice. Therefore, the interface has to be integrated into the process used by the SP to map user requests to SP servers. Next, we first review how this is currently realized using DNS. Then, we discuss options on how the user-slice assignment interface can be integrated, see Figure 44.

SP: User request to SP server mapping. Before an end-user issues a request for the CDI service, e.g., downloading some content or watching a video, it issues a DNS request to map the hostname to an IP address of the server that hosts the service. This DNS request is sent to the ISPs DNS resolver or an alternative DNS resolver. This resolver contacts the authoritative DNS server of the CDI service, since caching is typically restricted by small TTL's. The authoritative DNS server uses a CDI service, the CDI mapping system, to select a CDI server from which to satisfy the future requests of this end-user. The CDI mapping system performs a CDI specific optimization. This optimization may consider the load on the CDI servers, the network location of the CDI server as well as the requesting DNS server, the price of network access at the CDI server, etc. Note, the CDI's authoritative DNS name servers usually does not have

[10]Here, consolidation corresponds to moving multiple VMs with minimal resource requirements to the same physical machine to keep a base service.

access to the IP address of the end-user as the request is forwarded via the DNS resolver unless the eDNS [55] standard is used. With eDNS, the client IP address or the IP prefix can be added to the DNS request sent by the DNS resolver to the authoritative DNS name server. In addition, the CDI can use HTTP redirection to further load balance within its infrastructure.

User-Slice Assignment: Option 1. Considering the process outlined above, one possible way to use the user-slice assignment interface is within the optimization that the CDI mapping system performs. For this case, we envision two message types: CDI_User-Slice_Assign_Request and ISP_User-Slice_Assign_Response which correspond to steps 3 and 4 in Figure 44.

CDI_User-Slice_Assign_Request: Issued by the CDI's DNS server to the ISP. It contains the client IP address as well as slice locations within or outside of the ISP.

ISP_User-Slice_Assign_Response: The ISP responses with a ranking of the slice locations.

The previous two messages enable the CDI to consider information from the ISP, conveyed via the ranking, for its mapping. This is equivalent to the functionality and protocols proposed by the IETF ALTO working group. However, we envision a more light weight implementation. We propose to not rely on XML for encoding each request if the granularity of the requests is on a per connection level. If the CDI uses a coarser granularity such as subnet or region, the efficiency of the message exchange is even less critical.

User-Slice Assignment: Option 2. Another way to integrate the user-slice assignment interface within the above process is by proactively sharing information between the CDI and the ISP. For this case, we again envision two message types: ISP_User-Slice_Assign_Proposal and CDI_User-Slice_Assign_Ack.

ISP_User-Slice_Assign_Proposal: Is sent by the ISP to the CDI mapping system. It contains a set of IP prefixes each with an associated ranking of the different microdatacenter locations.

CDI_User-Slice_Assign_Ack: The CDI either acknowledges or rejects the proposal.

This again enables the CDI to include information from the ISP, conveyed via the ranking, in its mapping. For example, one can use BGP to send such messages—a mechanism Akamai already utilizes to aid in mapping users to its clusters [170].

13.5 Evaluation of NetPaaS

In this section we quantify the benefits of using *on-demand service* with NetPaaS to extend the deployment of the largest CDN inside a tier-1 ISP with cooperation. After describing our evaluation framework, we explore various scenarios, ranging from better utilizing the existing footprint to using all possible network locations. We also explore the impact of perfect knowledge of future demands vs. forecasting future demands.

13.5.1 Evaluation Framework

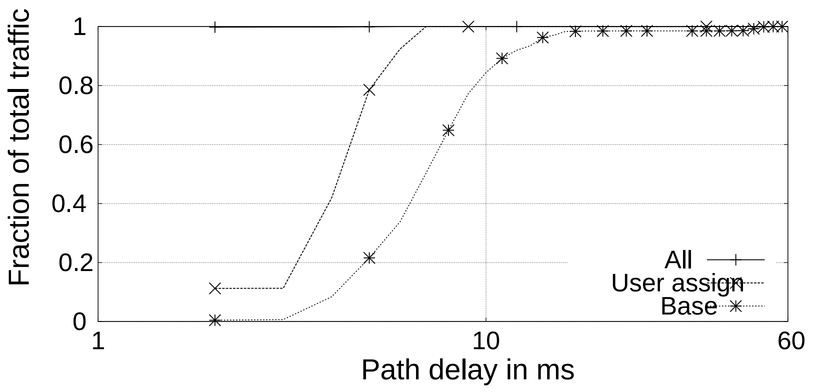
To evaluate the potential benefit of our scenario—CDN deployment using on-demand service—we rely on a simulator. Our simulator takes as input (i) the annotated topology and routing information for the considered ISP, (ii) the traffic demands of the considered CDN, (iii) the traffic demands of additional SPs, (iv) possible locations for slices inside the ISP network, and (v) traffic matrices representing the traffic inside the ISP without the load imposed by the CDNs–the background traffic. Based on this input, the simulator computes the resulting network loads inside the ISP network by computing solutions to the `slice location` and `user-slice assignment` problems. Multiple metrics are computed for the ISP network, including path delays within the ISP network, maximum link utilization, and hop counts. For the CDN/SPs we compute the network statistics for their subset of the traffic as well as the load imposed on each CDN/SP cluster and the network delay within the ISP topology for each cluster. Using an appropriate pricing model, the CDN/SP can estimate its economic benefits.

13.5.2 Evaluation Dataset

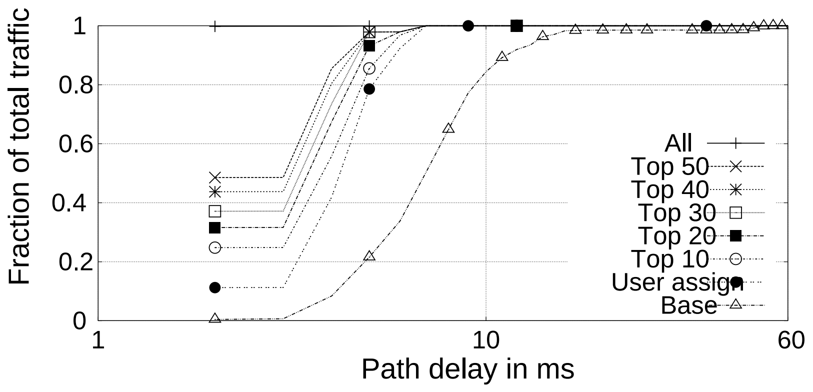
As in Section 8, we rely on two datasets, one from the CDN and the other from a tier-1 ISP. However, contrary to Section 8, in this section we require fine-grained information about ISP traffic.

As already mentioned, we know the complete annotated router-level topology of our tier-1 ISP, including the router locations as well as all public and private peerings. To derive the ISP's fine-grained traffic demands necessary to create the background traffic in our simulator, we combine information from the CDN with information from anonymized packet-level traces of residential DSL connections from the tier-1 ISP. We use a 10 day long trace starting on May 7, 2010. Our monitor, using Endace monitoring cards [51], allows us to observe the traffic of more than $20,000$ DSL lines to the Internet. We capture HTTP and DNS traffic using the Bro intrusion detection system [179]. We observe 720 million DNS messages as well as more than 1 billion HTTP requests involving about 1.4 million unique hostnames, representing more than 35 TBytes of data. Regarding the application mix, more than 65% of the traffic volume is due to HTTP. Other popular applications that contribute to the overall traffic volume are NNTP, BitTorrent, and eDonkey. As has been observed before for other traces, a large fraction of the traffic is due to a small number of large SPs, including the considered CDN, hyper-giants, and one-click-hosters [128, 87, 150].

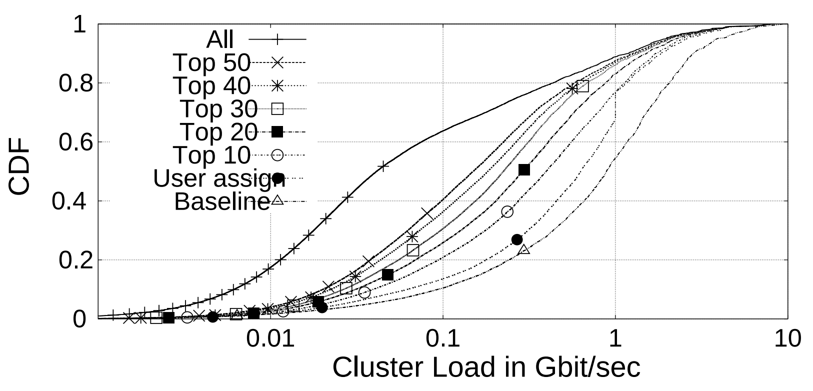
To derive the background traffic matrix—on an origin-destination flow granularity—we compute from the DSL traces (on a 10-minute time bin granularity) the demands for this location in the ISP network. This is then scaled according to the load imposed by users of the CDN to the other locations in the ISP network. For the additional SPs, we first identify their infrastructure locations using the infrastructure aggregation approach as proposed in [183] and then derive the traffic demands for them in a similar manner as the background traffic matrix. For the CDN traffic demands we use the ones from the traces aggregated to 10-minute time intervals.



(a) CDN: Path delay.



(b) CDN: Path delay.



(c) CDN: Slice/cluster utilization.

Figure 45: CDFs of CDN path delay as well as corresponding CDN slice/cluster utilization (logarithmic x-axis).

13.5.3 CDN Evaluation: Extreme Cases

In order to understand the impact of the two step approach consisting of `slice location` and `user-slice assignment`, we start our evaluation of the performance of on-demand service with a simple scenario. This scenario does not rely on any optimizations. Instead, it corresponds to the results from Section 8 which describe the CDN's current deployment and mapping of end-users to clusters. We refer to this as `Base`. The next scenario takes advantage of the `user-slice assignment` solution, and is referred to as `User-assign`. The third assumes that all locations in the ISP network offer slices, referred to as `All`. The resulting improvements in terms of one-way path delay inside the ISP for the CDN traffic are shown in Figure 45a. Note, the simulations use the links as seen by the IS-IS protocol from the ISP topology, which does not account for the delay of the last hop. We presume in our simulations a last hop delay of 1ms.

To put this in context, we perform a detailed analysis of the CDN connection log. More specifically, we examine the RTTs for the first TCP handshake as well as the averaged smoothed RTTs over the duration of the connection. Figure 46a shows the CDF of the RTTs for connections served within the PoP as compared to those served from other network locations. We find that the RTTs are significantly shorter for connections within the PoP. Therefore we conclude that the shorter path delays that the CDN can achieve with on-demand service translate to an improved end-user experience.

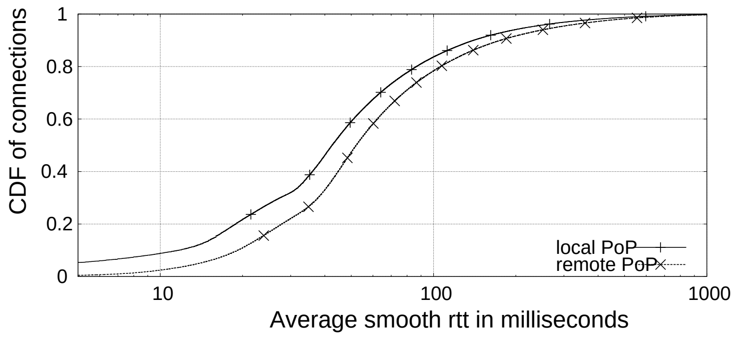
With regards to `Base` we find that the CDN is able to keep the backbone delays short on its own. More than 80% of the traffic does not experience a delay higher than 10ms. However, shorter delays can be achieved, as we can see from the `User-assign` and `All` results. When making on-demand service resources available at all locations, without resource constraints that limit the ability of `user-slice assignment`, almost all traffic is mapped to the CDN slices closest to the end-user.

Even if we do not allow `slice location` to use additional locations within the ISP network, `user-slice assignment` can be used to achieve a redistribution of the client requests to minimize the delay inside the ISP, presumably improving the end-user performance. We observe from Figure 45a that more than 10% of the traffic can be kept locally within the same slice location and more than 80% of the traffic can be served with a path delay of less than 5ms.

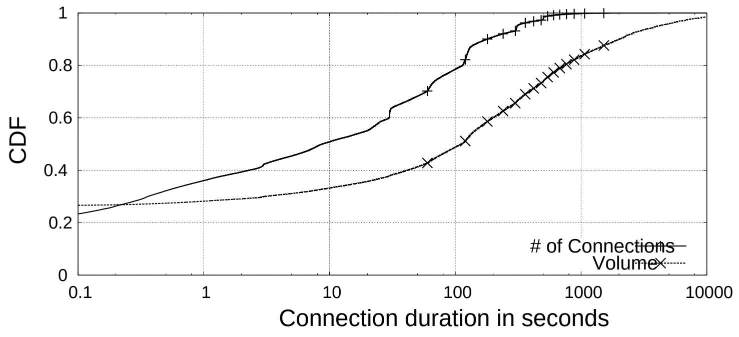
Figure 45c includes the CDFs of the differences in the slice/cluster utilizations between the three simulation runs. We see that `User-assign` shifts some of the load from the highly utilized slices/clusters to the other slices/clusters. When making more slice locations available in the ISP network, we observe a much more even spread of the loads.

13.5.4 CDN Evaluation: Limited Slice Locations

Now, the CDN uses the flexibility that `slice location` provides by allocating slices at microdatacenter locations with the top demand, as seen in the traces. We run simulations for additional 10, 20, 30, 40, 50 locations. In this case, the CDN is no longer constrained in terms of where its resources are located within the ISP. Figure 45b shows the CDF of the resulting one-way path delays for the CDN traffic in



(a) CDN: Avg. smoothed RTT: local/remote.

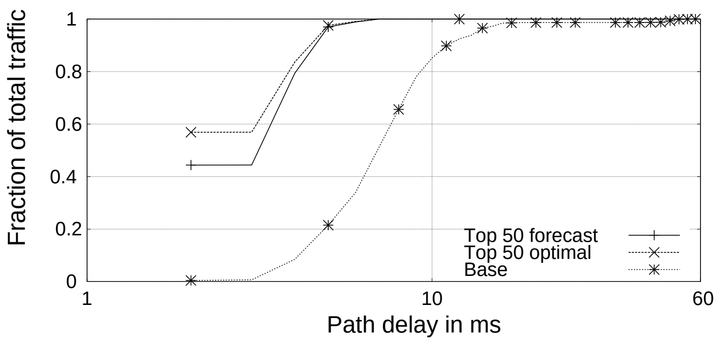


(b) CDN: Connection durations.

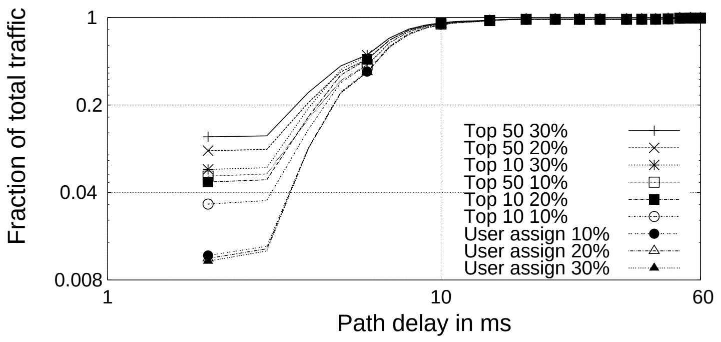
Figure 46: CDFs of trace RTT and connection durations.

addition to the above discussed extreme cases. We see that `user-slice assignment` is able to use the additional network locations – drastically reducing the path delay. The resulting CDFs sample the space between the curves from `User-assign` and `All`. With 50 locations, roughly 50% of the traffic benefits from the minimum possible backbone network delay. The marginal benefit of adding 10 additional network locations gets smaller. The first ten add more than 10% while the last ten less than 4%.

Figure 45c also includes the CDFs of the slice/cluster utilizations for the simulation runs with limited slice locations. The corresponding curves lie in between the extremes. As more locations are added, more traffic stays local, and thus the load on the slices are better balanced. With only 12.5% of the slice locations, we are able to achieve roughly half the benefit of going from `User-assign` to `All`. We point out that with contracts that only charge for slices that are utilized and do not differentiate between slice locations, there is no reason a priori to limit slice locations.



(a) CDN: Path delay based on predictions (log x).



(b) All: Path delay for scaled CDN demands (log-log).

Figure 47: CDFs of path delays.

13.5.5 CDN Evaluation: Prediction Accuracy

In the previous section we have assumed that we statically pick additional slice locations based on the full knowledge of the CDNs future demands. However, this information is typically not available. In addition, the CDN may want to further increase its flexibility by using slices at location x on day 1 but location y at day 2. However, this is only possible if the CDN is able to use new locations in a short time frame. For this to be feasible, the typical connection duration of the deployed service must be small enough. VM migration between network locations or slice consolidation within the location (see Section 13.4.2) must be done quickly to continue servicing the existing requests while moving the service closer to the end-users.

Figure 46b shows the CDF of the CDNs connection durations weighted by the number of connections as well as the traffic volume within the connection. We see that most connections are short. Indeed, almost 80% last less than 100 seconds. Indeed, within 10 minutes (600 seconds), more than 80% of the Bytes are served. Thus, it seems possible for the CDN to issue a new `slice location` request on time scales upward of 10 minutes.

Next, we ask how well the CDN is able to predict its resource needs. Overall,

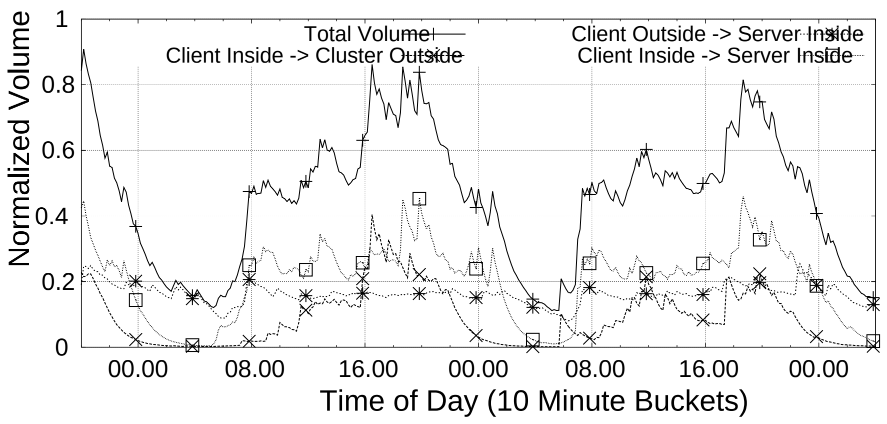


Figure 48: Moving average ($\alpha = 0.85$) of normalized CDN traffic by origin/destination position across 2 days.

	Total		ISP only	
Server	Inside	Outside	Inside	Outside
Users Inside	45.6%	16.8%	73%	27%
Users Outside	37.6%	N/A	N/A	N/A

Table 4: Fraction of CDN traffic inside ISP.

the time series of the CDN demand follows the typical time of day and day of week pattern, e.g., Figure 48. Figure 48 illustrates that the percentages from Table 4 are merely averages over time. During peak times, the percentage of the total volume from CDN servers external to the ISP to clients inside the ISP can reach up to 40%. This indicates that there is a significant fraction of volume from end-users within the ISP that is served from servers outside the ISP even though there is in principle capacity available within the ISP. This capacity is being used by end-users from outside the ISP. The total traffic observed includes requests from end-users of the ISP to CDN clusters inside and outside the ISP, as well as requests from end-users outside the ISP that are served by CDN clusters inside the ISP. We find that 45.6% of the total CDN traffic is served by clusters within the ISP to end-users in the ISP. However, 16.8% of the total traffic is served to end-users in the ISP by clusters outside the ISP. When looking only at clients inside the ISP, 73% of the traffic is served from within the ISP, while 27% is fetched from outside the ISP. In addition, we note that 37.6% of the total traffic served by clusters inside the ISP goes to end-users outside the ISP. It is interesting that some of the clusters inside the ISP serve external requests while requests from within the ISP are routed to servers outside the ISP.

However, when broken down on a per location basis, we notice a lot of variability and burstiness, e.g., due to localized flash crowds effects. A perfect prediction cannot be expected, and it is up to the CDN to deal with such effects. Therefore, the CDN has to over-provision its resources. We examine how well the demand can be predicted based on the previous 10 minutes, the previous hour, the previous six hours, as well as the previous day. We find that 10 minutes are too small a time scale for prediction purposes. One hour is not bad, but six hours are even better. Figure 47a shows the achieved path delay inside the ISP network as compared to the `base` scenario and an optimal placement of the slices assuming full knowledge of the demands in the next six hours. We see that with an optimal placement, one could keep almost 60% of the traffic local. Moreover, the prediction is good enough to localize more than 45% of the traffic with 50 microdatacenters.

13.5.6 Overall Evaluation: Scaled CDN Demands

Next, we examine the influence of on-demand service on the overall traffic pattern of the ISP. We note, that the overall impact is biased by the amount of traffic that can be influenced. If only 5% of the traffic can be influenced, the overall impact is relatively small. With 20%, the impact can be substantial. To see how the benefits of on-demand service scales as more and more SPs are taking advantage of it, we scale the demands of the CDN to account for 10%, 20%, 30% of the total traffic within the ISP.

The CDFs of the path delay are shown in Figure 47. Note the logarithmic scale also on the y-axis. We find that we are able to almost keep the maximum percentage of traffic local within the PoP. With 10%/20% CDN traffic we see 5%/8% within the PoP for 50 slice locations. (10%, 20% is the maximum reachable by placing slices at all network locations.)

Overall, we find a reduction in mean/median path delay inside the ISP network by 28.7%/28.5% for traffic of the SPs that can only take advantage of `user-slice assignment`. The CDN can take advantage of both `slice location` and `user-slice assignment`. It sees a reduction in path delay by 60.1%/42.8% with 50 locations. Put together this results in a reduction in the mean/median path delay of 15.7%/12.5% for all traffic of the ISP for the 20% CDN traffic case.

14 Conclusion

People value the Internet for the content and the applications it makes available [107]. For example, the demand for online entertainment and web browsing has exceeded 70% of the peak downstream traffic in the United States [194]. Recent traffic studies [87, 128, 183] show that a large fraction of Internet traffic is originated by a small number of Content Distribution Infrastructures (CDIs). Major CDIs are highly popular rich media sites such as YouTube and Netflix, One-Click Hosters (OCHs), e.g., RapidShare, as well as Content Delivery Networks (CDN) such as Akamai and hyper-giants, e.g., Google or Yahoo!. Gerber and Doverspike [87] report that a few CDIs account for more than half of the traffic of a US-based Tier-1 carrier.

To cope with the increasing demand for content, CDIs deploy massively distributed infrastructures [136] to replicate content and make it accessible from different locations in the Internet [221, 12]. Not all CDIs are built upon the same philosophy, designs and technology. For example, a CDI can be operated independently by deploying caches in different networks, by renting space in datacenters or by building its own datacenters. Furthermore, some CDIs are operated by ISPs, by Content Producers, or in the case of Peer-to-Peer networks, by self-organized end-users. Accordingly, we give an overview of the spectrum of CDI solutions.

CDIs often struggle in mapping users to the best server, regardless of whether the best server is the closest, the one with the most capacity or the lowest delay. The inability for CDIs to map end-users to the right server stems from the fact that CDIs have limited visibility into ISP networks as well as their setup, operation and current state. Thus, in this book chapter, we propose to tackle the challenges that CDIs as well as ISPs face as an opportunity: *to collaborate*. We point out the opportunities and incentives for all parties—CDIs, ISPs as well as end-users—to get involved. This may ultimately lead to major changes in the way that content is distributed across the Internet.

Accordingly, we review the proposed enablers and building blocks for collaboration ranging from the P2P oracle service, P4P, Ono, PaDIS, to the IETF activities [17, 233, 45, 183, 152, 167]. To illustrate the benefits of collaboration between applications and networks, we provide two use-cases: P2P and Traffic Engineering. The main take away is that substantial benefits for all involved parties are obtainable.

Among the upcoming trends are vitalization and the Cloud. These trends offer new ways of collaborative deployment of content delivery if combined with the proposed enablers for collaboration. Accordingly, we propose Network Platform as a Service (NetPaaS), which allows CDIs and ISP to cooperate not only on the user assignment, but also to dynamically deploy and remove servers and thus scale their infrastructure on demand.

We believe that collaboration as well as NetPaaS can play a significant role in the future content delivery ecosystem. However, most of the collaboration enablers have not yet been deployed in the wild, and therefore only the future will answer if the Internet takes advantage of the opportunities.

List of Figures

1 Layout of the Internet Structure [128] 10
2 IGP based Traffic Management Example 11
3 Example DNS hierarchy 14
4 Barplot of the application mix in the Internet 18
5 Barplot of content-type popularity in the Internet 19
6 Timeseries of link utilization from Maier et al. 20
7 Timeseries of traffic volume from ipoque 20
8 Timeseries of traffic volume from Sandvine 21
9 CCDF of flow sizes: Data and exponential distribution with same mean 21
10 CCDF of flow durations: Data and exponential distribution with same mean 22
11 Structural design of BRO 24
12 DNS replies for two different sites hosted on a CDI, in two-hour bins . 26
13 CCDF of mean # of IPs (top) and subnets (bottom) per DNS reply for the ISPs DNS resolver. 28
14 CDF of # of IPs for the ISP DNS resolver normalized by traffic volume (top) and requests (bottom) including aggregation on domain levels. (Logarithmic x-axis.) 29
15 CDF of # of subnets for ISP DNS resolver normalized by traffic volume (top) and by requests (bottom) including aggregation on domain levels. (Logarithmic x-axis.) 30
16 CDF of DNS TTL value by traffic volume and by number of requests. 31
17 Example of CDN deployment and traffic flows (Web traffic demands). 36
18 Example Web page with some CDN content. 37
19 Content Delivery Spectrum 42
20 Comparison of metrics with increasing size of Oracle list (Error plots). 65
21 Visualization of Gnutella overlay topology 70
22 Metrics for Gnutella simulations 70
23 Plots comparing unmodified(U) and biased(B) P2P neighbor selection across 10K queries and 10K file transfers, for 5 topologies 78
24 Effect of user behavior (content availability and session length) patterns for World3 topology. X-axis denotes file distribution models, and symbols denote online session length models: Uniform, Pareto, Weibull and Poisson. 79
25 By choosing a CP server for a client with the help of CaTE, traffic engineering goals and accurate end-user server assignment become possible. 81
26 CaTE System architecture and flow of messages. 83
27 Content-aware Traffic Engineering Process 86
28 CaTE and Restricted Machine Load Balancing. 89
29 CDF of traffic volume of CPs in ISP1. 94
30 Normalized traffic for top CPs by volume in ISP1. 94
31 Subnet diversity from which content is available. 95
32 Evolution over time of number of subnets for selected CPs in the top 10 CPs. 96

33 Maximum link utilization reduction (top) and total traffic reduction (bottom) with CaTE for the top CPs. 97
34 Improvements in link utilization with CaTE. 98
35 Backbone path length count with CaTE. 98
36 Improvement in path delay (in ms) with CaTE. 99
37 Total traffic (top) and maximum link utilization (bottom) reduction with CaTE and different network metrics. 101
38 Backbone path length (top) and accumulated path delay (bottom) with CaTE and different network metrics. 103
39 Link utilization improvements after applying when using CaTE in AT&T and Abilene. 104
40 Distribution of download times of a CP. 105
41 Projection of reduction in link utilization (top), reduction in overall network traffic (middle) and fraction of volume by path length (bottom) if Netflix is launched in ISP1. 106
42 Microdatacenters in an ISP with NetPaaS enabled 108
43 Slice allocation message exchange. 116
44 User-slice assignment schematic. 118
45 CDFs of CDN path delay as well as corresponding CDN slice/cluster utilization (logarithmic x-axis). 121
46 CDFs of trace RTT and connection durations. 123
47 CDFs of path delays. 124
48 Moving average ($\alpha = 0.85$) of normalized CDN traffic by origin/destination position across 2 days. 125

List of Tables

1 Summaries of anonymized traces from a European ISP 25
2 Traffic localization within the network by different DNS resolvers normalized by number of requests and traffic volume together with the potentially available fraction of localized traffic. 32
3 Number of exchanged Gnutella message types 73
4 Fraction of CDN traffic inside ISP. 125

References

[1] Akamai and AT&T Forge Global Strategic Alliance to Provide Content Delivery Network Solutions. http://www.akamai.com/html/about/press/releases/2012/press_120612.html.

[2] Orange and Akamai form Content Delivery Strategic Alliance. http://www.akamai.com/html/about/press/releases/2012/press_112012_1.html.

[3] Swisscom and Akamai Enter Into a Strategic Partnership. http://www.akamai.com/html/about/press/releases/2013/press_031413.html.

[4] Content delivery networks interconnection (cdni). http://datatracker.ietf.org/wg/cdni/charter/, 2011.

[5] Network Functions Virtualisation. SDN and OpenFlow World Congress, Oct 2012.

[6] I. Abraham, D. Malkhi, and O. Dobzinski. LAND: stretch $(1+\epsilon)$ locality-aware networks for DHTs. In *SODA*, 2004.

[7] M. Abrams, C. R. Standridge, G. Abdulla, S. Williams, and E. A. Fox. Caching proxies: limitations and potentials. In *WWW Conference*, 1995.

[8] P. Aditya, M. Zhao, Y. Lin, A. Haeberlen, P. Druschel, B. Maggs, and B. Wishon. Reliable Client Accounting for Hybrid Content-Distribution Networks. In *Proc. NSDI*, 2012.

[9] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger. Anatomy of a Large European IXP. In *Proc. SIGCOMM*, 2012.

[10] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing DNS Resolvers in the Wild. In *Proc. IMC*, 2010.

[11] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing DNS Resolvers in the Wild. In *Proc. IMC*, 2010.

[12] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Web Content Cartography. In *Proc. IMC*, 2011.

[13] B. Ager, F. Schneider, J. Kim, and A. Feldmann. Revisiting Cacheability in Times of User Generated Content. In *Proc. IEEE Global Internet*, 2010.

[14] V. Aggarwal, S. Bender, A. Feldmann, and A. Wichmann. Methodology for Estimating Network Distances of Gnutella Neighbors. In *GI Jahrestagung - Informatik 2004*, 2004.

[15] V. Aggarwal, A. Feldmann, and R. Karrer. An Internet Coordinate System to Enable Collaboration between ISPs and P2P Systems. In *ICIN*, 2007.

[16] V. Aggarwal, A. Feldmann, and S. Mohrs. Implementation of a P2P system within a network simulation framework. In *ECCS P2P-Complex Workshop*, 2005.

[17] V. Aggarwal, A. Feldmann, and C. Scheideler. Can ISPs and P2P systems co-operate for improved performance? *ACM SIGCOMM Computer Communications Review (CCR)*, 37(3):2940, July 2007.

[18] Akamai Facts & Figures. http://www.akamai.com/html/about/facts_figures.html.

[19] Akamai Inc. SureRoute. www.akamai.com/dl/feature_sheets/fs_edgesuite_sureroute.pdf.

[20] A. Akella, S. Seshan, and A. Shaikh. An Empirical Evaluation of Wide-Area Internet Bottlenecks. In *ACM IMC*, 2003.

[21] R. Alimi, R. Penno, and Y. Yang. ALTO Protocol. draft-ietf-alto-protocol-08, May 2011.

[22] AMAZON Web Services. http://aws.amazon.com.

[23] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris. Resilient Overlay Networks. In *SOSP*, 2001.

[24] N. Anderson. P2P traffic drops as streaming video grows in popularity. http://arstechnica.com/old/content/2008/09/p2p-traffic-drops-as-streaming-video-grows-in-\popularity.ars, 2008.

[25] A. Arlandis and E. Baranes. Interactions between network operators, content producers and internet intermediaries: Empirical implications of network neutrality. *Intereconomics*, 46(2):98–105, 2011.

[26] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the Clouds: A Berkeley View of Cloud Computing. UC Berkeley Technical Report EECS-2009-28, 2009.

[27] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC*, 2004.

[28] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local Search Heuristics for k-Median and Facility Location Problems. *SIAM J. on Computing*, 2004.

[29] P. Aukia, M. Kodialam, P. Koppol, T. Lakshman, H. Sarin, and B. Suter. RATES: A server for MPLS traffic engineering. *IEEE Network Magazine*, 2000.

[30] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and Principles of Internet Traffic Engineering. RFC3272.

[31] B. Awerbuch and F. Leighton. Multicommodity Flows: A Survey of Recent Research. In *Proc. ISAAC*, 1993.

[32] Y. Azar, J. Naor, and R. Rom. The competitiveness of on-line assignments. *J. Algorithms*, 1995.

[33] P. Barford, A. Bestavros, A. Bradley, and M. E. Crovella. Changes in Web Client Access Patterns: Characteristics and Caching Implications. *World Wide Web*, 1999.

[34] N. Basher, A. Mahanti, A. Mahanti, C. Williamson, and M. Arlitt˙ A comparative analysis of web and peer-to-peer traffic. In *Proc. WWW*, pages 287–296, 2008.

[35] L. Bent and G. Voelker. Whole Page Performance. In *Proc. Int. Workshop on Web Content Caching and Distribution*, 2002.

[36] T. Berners-Lee, R. Fielding, and H. Frystyk. Hypertext Transfer Protocol – HTTP/1.0. RFC1945, 1996.

[37] R. Bindal, P. Cao, W. Chan, J. Medved, G. Suwala, T. Bates, and A. Zhang. Improving Traffic Locality in BitTorrent via Biased Neighbor Selection. In *Proc. IEEE ICDCS*, 2006.

[38] L. Breslau, P. Cao, L. Fan, G. Philips, and S. Shenker. Web caching and Zipf-like distributions: Evidence and implications. In *Proc. INFOCOM*, pages 126–134, 1999.

[39] J. Buford, H. Yu, and E. K. Lua. *P2P Networking and Applications*. Morgan Kaufmann Series in Networking, 2008.

[40] CacheLogic: The True Picture of P2P Filesharing. http://www.cachelogic.com/.

[41] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. Analyzing the Video Popularity Characteristics of Large-scale User Generated Content Systems. *IEEE/ACM Trans. Netw.*, 17(5):1357–1370, 2009.

[42] H. Chang, S. Jamin, Z. Mao, and W. Willinger. An Empirical Approach to Modeling Inter-AS Traffic Matrices. In *IMC*, 2005.

[43] H. Chang, S. Jamin, and W. Willinger. To Peer ot not to Peer: Modeling the Evolution of the Internet's AS-level Topology. In *Proc. INFOCOM*, 2006.

[44] M. Charikar, S. Guha, D. Shmoys, and E. Tardos. A Constant Factor Approximation Algorithm for the k-median Problem. In *Proc. STOC*, 1999.

[45] D. R. Choffnes and F. E. Bustamante. Taming the Torrent: a Practical Approach to Reducing Cross-ISP Traffic in Peer-to-peer Systems. In *Proc. ACM SIGCOMM*, 2008.

[46] B. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. Controlling Data in the Cloud: Outsourcing Computation withour Outsourcing Control. In *Proc. ACM workshop on Cloud computing security,*, 2009.

[47] K. Church, A. Greenberg, and J. Hamilton. On Delivering Embarrasingly Distributed Cloud Services. In *Proc. HotNets*, 2008.

[48] Cisco. NetFlow services and applications. White paper: http://www.cisco.com/warp/public/732/netflow.

[49] CISCO Global Visual Networking and Cloud Index. Forecast and Methodology, 2011-2016. http://www.cisco.com.

[50] K. C. Claffy and N. Brownlee. Understanding Internet traffic streams: Dragonflies and Tortoises. *IEEE Trans. Communications*, 2002.

[51] J. Cleary, S. Donnelly, I. Graham, A. McGregor, and M. Pearson. Design Principles for Accurate Passive Measurement. In *Proc. PAM*, 2000.

[52] J. E. Coffman, M. Garey, and D. Johnson. Approximation Algorithms for Bin Packing: A Survey. In *Approximation algorithms for NP-hard problems*, 1997.

[53] B. Cohen. Incentives Build Robustness in BitTorrent. In *P2PEcon Workshop*, 2003.

[54] C. Contavalli, W. van der Gaast, S. Leach, and E. Lewis. Client subnet in DNS requests. draft-vandergaast-edns-client-subnet-01.

[55] C. Contavalli, W. van der Gaast, S. Leach, and D. Rodden. Client IP Information in DNS Requests. IETF draft, work in progress, draft-vandergaast-edns-suspend-ip-01.txt, 2011.

[56] S. Crosby and D. Brown. The Virtualization Reality. *Queue*, 2006.

[57] M. Crovella and A. Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. In *Proc. SIGMETRICS*, 1996.

[58] Cymru Whois. http://www.cymru.com/BGP/asnlookup.html.

[59] A. Czumaj, C. Riley, and C. Scheideler. Perfectly Balanced Allocation. In *RANDOM-APPROX*, 2003.

[60] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A Decentralized Network Coordinate System. In *SIGCOMM*, 2004.

[61] A. Dhamdhere and C. Dovrolis. Ten years in the evolution of the Internet ecosystem. In *Proc. IMC*, 2008.

[62] P. Dhungel, K. W. Ross, M. Steiner, Y. Tian, and X. Hei. Xunlei: Peer-Assisted Download Acceleration on a Massive Scale. In *Proc. PAM*, 2012.

[63] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl. Globally distributed content delivery. *IEEE Internet Computing*, 2002.

[64] D. DiPalantino and R. Johari. Traffic Engineering versus Content Distribution: A Game-theoretic Perspective. In *Proc. INFOCOM*, 2009.

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

[66] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer. Dynamic Application-Layer Protocol Analysis for Network Intrusion Detection. In *Proc. USENIX Security Symp.*, 2006.

[67] A. Elwalid, C. Jin, S. Low, and I. Widjaja. MATE : MPLS adaptive traffic engineering. In *Proc. INFOCOM*, 2001.

[68] J. Erman, A. Gerber, M. Hajiaghayi, D. Pei, and O. Spatscheck. Network-aware forward caching. In *Proc. WWW*, 2009.

[69] R. et.al. Topologically aware overlay construction and server selection. In *INFOCOM*, 2002.

[70] W. Fang and L. Peterson. Inter-AS Traffic Patterns and their Implications. In *Proc. IEEE Global Internet*, 1999.

[71] A. Feldmann, R. Caceres, F. Douglis, G. Glass, and M. Rabinovich. Performance of Web Proxy Caching in Heterogeneous Bandwidth Environments. *INFOCOM*, 1999.

[72] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford. NetScope: Traffic Engineering for IP Networks. *IEEE Network Magazine*, 2000.

[73] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True. Deriving Traffic Demands for Operational IP Networks: Methodology and Experience. In *Proc. SIGCOMM*, 2000.

[74] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True. Deriving Traffic Demands for Operational IP Networks: Methodology and Experience. *IEEE/ACM Trans. Netw.*, 2001.

[75] A. Feldmann, N. Kammenhuber, O. Maennel, B. Maggs, R. D. Prisco, and R. Sundaram. A Methodology for Estimating Interdomain Web Traffic Demands. In *Proc. IMC*, 2004.

[76] S. Fischer, N. Kammenhuber, and A. Feldmann. REPLEX: Dynamic Traffic Engineering based on Wardrop Routing Policies. In *Proc. CoNEXT*, 2006.

[77] S. Fischer, H. Räcke, and B. Vöcking. Fast Convergence to Wardrop Equilibria by Adaptive Sampling Methods. In *Proc. STOC*, 2006.

[78] S. Fischer and B. Vöcking. Adaptive Routing with Stale Information. In *Proc. PODC*, 2005.

[79] S. Floyd and V. Paxson. Difficulties in Simulating the Internet. In *IEEE/ACM Transactions on Networking, 9(4)*, 2001.

[80] B. Fortz and M. Thorup. Internet Traffic Engineering by Optimizing OSPF Weights. In *Proc. INFOCOM*, 2000.

[81] B. Fortz and M. Thorup. Optimizing OSPF/IS-IS Weights in a Changing World. *IEEE J. Sel. Areas in Commun.*, 2002.

[82] P. Francois and O. Bonaventure. Avoiding transient loops during the convergence of link-state routing protocols. *IEEE/ACM Trans. Netw.*, 2007.

[83] B. Frank, I. Poese, G. Smaragdakis, S. Uhlig, and A. Feldmann. Content-aware Traffic Engineering. *CoRR arXiv*, 1202.1464, 2012.

[84] B. Frank, I. Poese, G. Smaragdakis, S. Uhlig, and A. Feldmann. Enabling content-aware traffic engineering. *ACM CCR*, 42(4):21–28, 2012.

[85] M. J. Freedman. Experiences with CoralCDN: A Five-Year Operational View. In *Proc. NSDI*, 2010.

[86] S. Gadde, J. Chase, and M. Rabinovich. Web caching and content distribution: a view from the interior. *Computer Communications*, 2001.

[87] A. Gerber and R. Doverspike. Traffic Types and Growth in Backbone Networks. In *OFC/NFOEC*, 2011.

[88] A. Gish, Y. Shavitt, and T. Tankel. Geographical Statistics and Characteristics of P2P Query Strings. In *IPTPS*, 2007.

[89] Global Internet Geography. TeleGeography research. http://www.telegeography.com/product-info/gb/download/executive-summary.pdf, 2009.

[90] Gnutella Hostcache. http://www.the-gdf.org/index.php?title=The_Local_Hostcache.

[91] Gnutella v0.6 RFC. http://www.the-gdf.org.

[92] Gnutella wikipedia. http://en.wikipedia.org/wiki/Gnutella.

[93] D. Goldenberg, L. Qiuy, H. Xie, Y. Yang, and Y. Zhang. Optimizing Cost and Performance for Multihoming. In *Proc. SIGCOMM*, 2004.

[94] Google Datacenters. http://www.google.com/about/datacenters/.

[95] Google Global Cache. http://ggcadmin.google.com/ggc.

[96] Google Public DNS. https://developers.google.com/speed/public-dns/.

[97] R. Graham. Bounds on Multiprocessing Timing Anomalies. *SIAM J. Applied Math.*, 1969.

[98] GTK-Gnutella. http://www.gtk-gnutella.com/.

[99] A. Gunnar, M. Johansson, and T. Telkamp. Traffic Matrix Estimation on a Large IP Backbone: A Comparison on Real Data. In *Proc. IMC*, 2004.

[100] S. Halabi. *Internet Routing Architectures*. Cisco Press, 2000.

[101] K. Ho, J. Wu, and J. Sum. On the Session Lifetime Distribution of Gnutella. In *International Journal of Parallel, Emergent and Distributed Systems*, 2007.

[102] C. Huang, J. Li, A. Wang, and K. W. Ross. Understanding Hybrid CDN-P2P: Why Limelight Needs its Own Red Swoosh. In *NOSSDAV*, 2008.

[103] C. Huang, A. Wang, J. Li, and K. Ross. Measuring and Evaluating Large-scale CDNs. In *Proc. IMC*, 2008.

[104] S. Hull. *Content Delivery Networks: Web Switching for Security, Availability, and Speed*. McGraw-Hill, 2002.

[105] Germany Internet. http://www.internet-sicherheit.de/internet-deutschland.html.

[106] V. Jacobson, C. Leres, and S. McCanne. libpcap and tcpdump, 2009. http://www.tcpdump.org.

[107] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard. Networking Named Content. In *Proc. CoNEXT*, 2009.

[108] K. Jain and V. Vazirani. Primal-Dual Approximation Algorithms for Metric Facility Location and k-Median Problems. In *Proc. FOCS*, 1999.

[109] W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang. Cooperative Content Distribution and Traffic Engineering in an ISP Network. In *Proc. SIGMETRICS*, 2009.

[110] K. Johnson, J. Carr, M. Day, and M. Kaashoek. The Measured Performance of Content Distribution Networks. In *Proc. Int. Workshop on Web Caching and Content Delivery*, 2000.

[111] J. Jung, B. Krishnamurthy, and M. Rabinovich. Flash crowds and denial of service attacks: characterization and implications for CDNs and web sites. In *Proc. WWW*, 2002.

[112] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS Performance and the Effectiveness of Caching. *IEEE/ACM Trans. Netw.*, 10(5):589–603, 2002.

[113] S. Kandula, D. Katabi, B. Davie, and A. Charny. Walking the Tightrope: Responsive Yet Stable Traffic Engineering. In *Proc. SIGCOMM*, 2005.

[114] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should ISPs fear Peer-Assisted Content Distribution? In *IMC*, 2005.

[115] R. Keralapura, N. Taft, C. Chuah, and G. Iannaccone. Can ISPs Take the Heat from Overlay Networks? In *HotNets*, 2004.

[116] P. Key, L. Massoulié, and D. Towsley. Path Selection and Multipath Congestion Control. *Commun. of ACM*, 2011.

[117] L. Khachiyan. A Polynomial Time Algorithm for Linear Programming. *Dokl. Akad. Nauk SSSR*, 1979.

[118] J. Kleinberg and E. Tardos. *Algorithm Design*. Addison Wesley, 2005.

[119] A. Klemm, C. Lindemann, M. K. Vernon, and O. P. Waldhorst. Characterizing the Query Behavior in P2P File Sharing Systems. In *IMC*, 2004.

[120] M. Kodialam and T. V. Lakshman. Minimum Interference Routing with Applications to MPLS Traffic Engineering. In *Proc. INFOCOM*, 2000.

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

[122] P. Kolman and C. Scheideler. Improved bounds for the unsplittable flow problem. In *SODA*, 2002.

[123] M. Korupolu, C. Plaxton, and R. Rajaraman. Analysis of a Local Search Heuristic for Facility Location Problems. *J. Algorithms*, 37:146–188, 2000.

[124] B. Krishnamurthy and J. Rexford. *Web protocols and practice: HTTP/1.1, Networking protocols, caching, and traffic meas urement*. Addison-Wesley, 2001.

[125] B. Krishnamurthy, C. Wills, and Y. Zhang. On the Use and Performance of Content Distribution Networks. In *Proc. ACM IMW*, 2001.

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

[127] S. S. Krishnan and R. K. Sitaraman. Video Stream Quality Impacts Viewer Behavior: Inferring Causality using Quasi-Experimental Designs. In *Proc. IMC*, 2012.

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

[129] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. Kolaczyk, and N. Taft. Structural Analysis of Network Traffic Flows. In *Proc. SIGMETRICS*, 2004.

[130] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez. Inter-Datacenter Bulk transfers with NetStitcher. In *Proc. SIGCOMM*, 2011.

[131] N. Laoutaris, G. Smaragdakis, P. Rodriguez, and R. Sundaram. Delay Tolerant Bulk Data Transfers on the Internet. In *Proc. SIGMETRICS*, 2009.

[132] B. Lavoie and H. Nielsen. Web Characterization Terminology & Definitions Sheet. http://www.w3c.org/1999/05/WCA-terms/.

[133] F. Le Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massoulié. Clustering in P2P File Sharing Workloads. In *IPTPS*, 2004.

[134] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In *NSDI*, 2007.

[135] J. Lee, Y. Yi, S. Chong, and Y. Jin. On the Interaction between Content-oriented Traffic Scheduling and Revenue Sharing among Providers. In *Proc. of IEEE INFOCOM Workshop on Smart Data Pricing*, 2013.

[136] T. Leighton. Improving Performance on the Internet. *Commun. of ACM*, 2009.

[137] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of Ethernet traffic. In *Proc. SIGCOMM*, pages 183–193, 1993.

[138] J. Lenstra, D. Shmoys, and E. Tardos. Approximation Algorithms for Scheduling Unrelated Parallel Machines. *Math. Program.*, 1990.

[139] R. Levi, D. Shmoys, and C. Swamy. LP-based Approximation Algorithms for Capacitated Facility Location. In *Proc. SODA*, 2004.

[140] A. Li, X. Yang, S. Kandula, and M. Zhang. CloudCmp: Comparing Public Cloud Providers. In *Proc. IMC*, 2010.

[141] L. Li, D. Alderson, W. Willinger, and J. Doyle. A First-Principles Approach to Understanding the Internet's Router-level Topology. In *SIGCOMM*, 2004.

[142] Light Reading. Controlling P2P Traffic. http://www.lightreading.com/document.asp?site=lightreading&doc_id=44435&page_number=3.

[143] M. Liljenstam, J. Liu, and D. Nicol. Development of an Internet Backbone Topology for Large-Scale Network Simulations. In *Winter Simulation Conference*, 2003.

[144] Limelight Networks. http://www.limelight.com/technology/.

[145] P. Linga, I. Gupta, and K. Birman. A Churn-Resistant P2P Web Caching System. In *SSRS*, 2003.

[146] H. H. Liu, Y. Wang, Y. Yang, H. Wang, and C. Tian. Optimizing Cost and Performance for Content Multihoming. In *Proc. SIGCOMM*, 2012.

[147] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A Case for a Coordinated Internet-Scale Video Control Plane. In *Proc. SIGCOMM*, 2012.

[148] R. T. B. Ma, D. M. Chiu, J. C. S. Lui, V. Misra, and D. Rubenstein. On Cooperative Settlement Between Content, Transit, and Eyeball Internet Service Providers. *IEEE/ACM Trans. Netw.*, 2011.

[149] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat. Systematic Topology Analysis and Generation Using Degree Correlations. In *SIGCOMM*, 2006.

[150] G. Maier, A. Feldmann, V. Paxson, and M. Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *Proc. IMC*, 2009.

[151] Z. Mao, C. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang. A Precise and Efficient Evaluation of the Proximity Between Web Clients and Their Local DNS Servers. In *Proc. USENIX ATC*, 2002.

[152] E. Marocco and V. Gurbani. Application-layer traffic optimization, 2008.

[153] MaxMind LLC. http://www.maxmind.com.

[154] L. W. McKnight and J. P. B. (Eds). *Internet Economics*. MIT Press, 1998.

[155] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic Matrix Estimation: Existing Techniques and New Directions. In *Proc. SIGCOMM*, 2002.

[156] T. Mennecke. DSL Broadband Providers Perform Balancing Act. http://www.slyck.com/news.php?story=973.

[157] Microsoft Azure. http://www.windowsazure.com.

[158] M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 2003.

[159] P. Mockapetris. Domain Names - Implementation and Specification. RFC 1035, Nov 1987.

[160] D. R. Morrison. Practical Algorithm to Retrieve Information Coded in Alphanumeric. *J. of the ACM*, 1968.

[161] J. Moy. OSPF Version 2. RFC 2328, IETF, 1998.

[162] W. Mühlbauer, A. Feldmann, O. Maennel, M. Roughan, and S. Uhlig. Building an AS-Topology Model that Captures Route Diversity. In *SIGCOMM*, 2006.

[163] A. Nakao, L. Peterson, and A. Bavier. A Routing Underlay for Overlay Networks. In *Proc. ACM SIGCOMM*, 2003.

[164] Namebench. http://code.google.com/p/namebench/.

[165] M. Naor and U. Wieder. Novel architectures for P2P applications: the continuous-discrete approach. In *SPAA*, 2003.

[166] Announcing the Netflix Open Connect Network. http://blog.netflix.com/2012/06/announcing-netflix-open-connect-network.html.

[167] B. Niven-Jenkins, F. L. Faucheur, and N. Bitar. Content Distribution Network Interconnection (CDNI) Problem Statement. IETF draft, work in progress, draft-ietf-cdni-problem-statement-03.txt, 2012.

[168] B. Niven-Jenkins, F. L. Faucheur, and N. Bitar. Content distribution network interconnection (cdni) problem statement. RFC6707, 2012.

[169] W. Norton. The Internet Peering Playbook: Connecting to the Core of the Internet. DrPeering Press, 2012.

[170] E. Nygren, R. K. Sitaraman, and J. Sun. The Akamai Network: A Platform for High-performance Internet Applications. *SIGOPS Oper. Syst. Rev.*, 2010.

[171] A. Odlyzko. Content is not king. *First Monday*, 6(2), 2001.

[172] A. M. Odlyzko. Internet traffic growth: Sources and implications. http://www.dtc.umn.edu/mints/home.php, 2003.

[173] D. Oran. OSI IS-IS Intra-domain Routing Protocol. RFC 1142, IETF, 1990.

[174] J. S. Otto, M. A. Sánchez, J. P. Rula, and F. E. Bustamante. Content Delivery and the Natural Evolution of DNS - Remote DNS Trends, Performance Issues and Alternative Solutions. In *Proc. IMC*, 2012.

[175] V. Padmanabhan and L. Subramanian. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *SIGCOMM*, 2001.

[176] J. Pan, Y. Hou, and B. Li. An Overview of DNS-based Server Selections in Content Distribution Networks. *Com. Networks*, 2003.

[177] K. Papagiannaki, N. Taft, and A. Lakhina. A Distributed Approach to Measure IP Traffic Matrices. In *Proc. IMC*, 2004.

[178] V. Paxson. Bro intrusion detection system. http://www.bro.org.

[179] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks*, 31(23-24):2435–2463, 1999.

[180] V. Paxson and S. Floyd. Wide area traffic: The failure of Poisson modeling. *IEEE/ACM Trans. Networking*, 3(3):226–244, 1995.

[181] G. Plaxton, R. Rajaraman, and A. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *SPAA*, 1997.

[182] L. Plissonneau, J.-L. Costeux, and P. Brown. Analysis of Peer-to-Peer Traffic on ADSL. In *Proc. PAM*, 2005.

[183] I. Poese, B. Frank, B. Ager, G. Smaragdakis, and A. Feldmann. Improving Content Delivery using Provider-Aided Distance Information. In *Proc. IMC*, 2010.

[184] I. Poese, B. Frank, B. Ager, G. Smaragdakis, S. Uhlig, and A. Feldmann. Improving Content Delivery with PaDIS. *IEEE Internet Computing*, 2012.

[185] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye. IP Geolocation Databases: Unreliable? *ACM CCR*, 41, April 2011.

[186] R. Prasad, M. Murray, C. Dovrolis, and K. Claffy. Bandwidth estimation: metrics, measurement techniques and tools. In *IEEE Network*, 2003.

[187] pWhoIs. http://pwhois.org.

[188] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs. Cutting the Electric Bill for Internet-scale Systems. In *Proc. SIGCOMM*, 2009.

[189] A. Rasti, D. Stutzbach, and R. Rejaie. On the Long-term Evolution of the Two-Tier Gnutella Overlay. In *Global Internet*, 2006.

[190] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271, IETF, 2006.

[191] S. Rewaskar and J. Kaur. Testing the Scalability of Overlay Routing Infrastructures. In *PAM*, 2004.

[192] M. Roughan. Simplifying the Synthesis of Internet Traffic Matrices. *ACM CCR*, 2005.

[193] University of Oregon Routeviews Project. http://www.routeviews.org/.

[194] Sandvine Inc. Global Broadband Phenomena. Research Report http://www.sandvine.com/news/global_broadband_trends.asp.

[195] Sandvine Inc. Global Broadband Phenomena. Research Report http://www.sandvine.com/news/global_broadband_trends.asp, 2011.

[196] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy. An analysis of Internet content delivery systems. In *proc. OSDI*, 2002.

[197] S. Saroiu, P. Gummadi, and S. Gribble. A Measurement Study of P2P File Sharing Systems. In *Technical Report UW-CSE-01-06-02, Dept of CS, Washington University*, 2002.

[198] S. Savage, A. Collins, and E. Hoffman. The End-to-End Effects of Internet Path Selection. In *SIGCOMM*, 1999.

[199] C. Scheideler. Towards a paradigm for robust distributed algorithms and data structures. In *HNI Symposium on New Trends in Parallel and Distributed Computing*, 2006.

[200] H. Schulze and K. Mochalski. Internet study 2007-2009. http://www.ipoque.com/resources/internet-studies/.

[201] H. Schulze and K. Mochalski. Internet study 2007. http://www.ipoque.com/resources/internet-studies/ (need to register), 2007.

[202] S. Seetharaman and M. Ammar. On the Interaction between Dynamic Routing in the Native and Overlay Layers. In *INFOCOM*, 2006.

[203] D. N. Serpanos, G. Karakostas, and W. H. Wolf. Effective Caching of Web Objects using Zipf's Law. In *ICME*, 2000.

[204] A. Sharma, A. Mishra, V. Kumar, and A. Venkataramani. Beyond MLU: An application-centric comparison of traffic engineering schemes. In *Proc. INFOCOM*, 2011.

[205] G. Shen, Y. Wang, Y. Xiong, B. Y. Zhao, and Z.-L. Zhang. HPTP: Relieving the Tension between ISPs and P2P. In *IPTPS*, 2007.

[206] X. Shen, H. Yu, J. Buford, and M. Akon. *Handbook of peer-to-peer networking*, volume 1. Springer Heidelberg, 2010.

[207] R. Sherwood, A. Bender, and N. Spring. DisCarte: A Disjunctive Internet Cartographer. In *Proc. SIGCOMM*, 2008.

[208] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot. Traffic Matrices: Balancing Measurements, Inference and Modeling. In *Proc. SIGMETRICS*, 2005.

[209] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *SIGCOMM*, 2002.

[210] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP Topologies with Rocketfuel. *IEEE/ACM Trans. Netw.*, 2004.

[211] SSFNet. http://www.ssfnet.org.

[212] R. Steinmetz and K. Wehrle. *P2P Systems and Applications*. Springer Lecture Notes in CS, 2005.

[213] Streamingmedia Blog. http://blog.streamingmedia.com/the_business_of_online_vi/2011/06.

[214] D. Stutzbach and R. Rejaie. Understanding Churn in P2P Networks. In *IMC*, 2006.

[215] D. Stutzbach, R. Rejaie, and S. Sen. Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems. In *ACM IMC*, 2005.

[216] A. Su, D. Choffnes, A. Kuzmanovic, and F. Bustamante. Drafting behind Akamai (travelocity-based detouring). In *Proc. SIGCOMM*, 2006.

[217] M. Tariq, A. Zeitoun, V. Valancius, N. Feamster, and M. Ammar. Answering What-if Deployment and Configuration Questions with Wise. In *Proc. SIGCOMM*, 2009.

[218] R. Tashev. Experimenting with Neighbour Discovery Schemes for P2P Networks in a Simulation Framework. In *Master thesis, Dept of CS, TU Munich*, 2006.

[219] J. Tian and Y. Dai. Understanding the Dynamic of P2P Systems. In *IPTPS*, 2007.

[220] TK Fachbegriffe. http://www.tk-fachbegriffe.de/index.php?id2=2400&a=320.

[221] S. Triukose, Z. Al-Qudah, and M. Rabinovich. Content Delivery Networks: Protection or Threat? In *ESORICS*, 2009.

[222] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon. Providing Public Intradomain Traffic Matrices to the Research Community. *ACM CCR*, 2006.

[223] V. Valancius, C. Lumezanu, N. Feamster, R. Johari, and V. V. Vazirani. How Many Tiers? Pricing in the Internet Transit Market. In *Proc. SIGCOMM*, 2011.

[224] Virtual Computing Environment Consortium. http://www.vce.com.

[225] P. Vixie. DNS Complexity. *ACM Queue*, 5(3):24–29, 2007.

[226] P. Vixie. What DNS is Not. *Commun. of ACM*, 2009.

[227] J. Wallerich, H. Dreger, A. Feldmann, B. Krishnamurthy, and W. Willinger. A methodology for studying persistency aspects of Internet flows. *ACM CCR*, 2005.

[228] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Trans. Networking*, 5(1), 1997.

[229] Wireshark Foundation. http://www.wireshark.org, 2009.

[230] B. Wong, I. Stoyanov, and E. Sirer. Octant: A Comprehensive Framework for Geolocalization of Internet Hosts. In *NSDI*, 2007.

[231] P. Xia, S.-H. G. Chan, M. Chiang, G. Shui, H. Zhang, L. Wen, and Z. Yan. Distributed Joint Optimization of Traffic Engineering and Server Selection. In *IEEE Packet Video Workshop*, 2010.

[232] X. Xiao, A. Hannan, B. Bailey, and L. Ni. Traffic Engineering with MPLS in the Internet. *IEEE Network Magazine*, 2000.

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

[234] X. Yang and G. de Veciana. Service Capacity of Peer to Peer Networks. In *Proc. INFOCOM*, 2004.

[235] yWorks. http://www.yworks.com.

[236] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker. On the Characteristics and Origins of Internet Flow Rates. In *Proc. SIGCOMM*, 2002.

[237] Y. Zhang and N. Duffield. On the Constancy of Internet Path Properties. In *Internet Measurements Workshop*, 2001.

[238] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg. Fast Accurate Computation of Large-scale IP Traffic Matrices from Link Loads. In *Proc. SIGMETRICS*, 2003.

[239] Y. Zhang, M. Roughan, C. Lund, and D. Donoho. An Information-theoretic Approach to Traffic Matrix Estimation. In *Proc. SIGCOMM*, 2003.

[240] S. Zhao, D. Stutzbach, and R. Rejaie. Characterizing Files in the Modern Gnutella Network. In *MMCN*, 2006.

[241] H. Zheng, E. Lua, M. Pias, and T. Griffin. Internet Routing Policies and Round-Trip-Times. In *PAM*, 2005.

[242] G. Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.