Lehrstuhl für Intelligente Netze
An-Institut Deutsche Telekom Laboratories

ISP-Aided Neighbour Selection in Peer-to-Peer Systems

vorgelegt von
M.Sc.
Vinay Kumar Aggarwal
aus Indien

Von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr.-Ing. Adam Wolisz
Berichter: Univ.-Prof. Anja Feldmann, Ph.D.
Berichter: Prof. Dr. habil. Odej Kao
Berichter: Prof. Dr. h.c. Hans Albert Aukes

Tag der wissenschaftlichen Aussprache: 27. 11. 2008

Berlin 2008

D 83

Abstract

Peer-to-peer (P2P) systems account for more than half of Internet traffic today, and an increasing number of user applications, e.g., Bittorrent, eDonkey, Joost, Skype, GoogleTalk, and P2P-TV, rely on P2P methodology. P2P systems build overlays at the application layer, independently of Internet routing and ISP topologies. This leads to measurement traffic overhead and routing inefficiencies for P2P users. While P2P applications spur broadband access, they also take customers away from traditional telephones and pose significant traffic engineering challenges for ISPs, thus putting them in a dilemma! Some ISPs have resorted to impeding P2P traffic by bandwidth shaping, though unsuccessfully. Meanwhile, some P2P applications attempt to measure the network latency to potential neighbours, e.g., by ping measurements, to choose high-performance network paths. However, such measures have not addressed the routing conflict between ISPs and P2P systems. Our measurement study and visualization-based analysis finds that the overlay topology of P2P systems is not correlated with the Internet AS topology, and that a larger number of overlay peerings cross AS boundaries multiple times.

In this thesis, we propose a simple, general and unique solution that enables ISPs and P2P systems to collaborate with each other. We propose that an ISP hosts a server, which we call the *oracle*, that helps P2P users choose "good" neighbours. The P2P user sends the list of its potential neighbours to the oracle, which ranks this list of IP addresses based on a number of factors, that an ISP decides individually. For example, the ISP can prefer peers within its network, to prevent traffic from leaving its network. Further, it can choose better bandwidth or lesser delay nodes, or those that are geographically closer (same city, same PoP) within its network. The oracle returns this sorted list to the P2P user, who can then benefit from the knowledge of the ISP and connect to a neighbour recommended by the oracle. This will not only reduce costs and ease routing for ISPs, but will also provide improved performance for P2P users in the sense of higher bandwidth and lesser delay. In this way, ISPs and P2P systems can cooperate so that both of them benefit.

We have conducted a comprehensive analysis of this proposal using graph experiments, testbed implementation, Planetlab deployment, and packet-level simulations on various models of P2P systems. The graph results show that P2P users, on consulting the oracle, are able to keep most of their peerings within the ISP boundaries, without any adverse effects on the overlay graph structural properties. A theoretical analysis of the congestion caused by shorter network paths of P2P links reveals that the congestion in the network is close to the theoretical optimum, while almost all the overlay peerings are formed in accordance with the ISP policies. Through testbed implementation and Planetlab deployment, we show that the ISP-P2P collaboration scheme is feasible with real P2P systems. The experiment results also show that the scalability of P2P systems improve considerably, and there is no adverse effect on the query search phase of the P2P networks. The P2P users are able to locate all available content from nodes at shorter network distances.

Using extensive packet-level simulations, we verify the above results with the Gnutella P2P protocol under churn. We quantify the performance improvements for ISPs and P2P users, using metrics like intra-AS content exchange and content download times. We simulate multiple ISP and P2P topologies, as well as a range of user behaviour characteristics, namely, churn, content availability

and query patterns, using different mathematical distributions. This enables us to study the effects of realistic, best-case, and adverse scenarios on end-user performance. We show that the benefits of our proposed ISP-P2P collaboration scheme hold across a range of user behaviour scenarios and ISP/P2P topologies. The ISPs are able to save costs by keeping a large amount of traffic within their network, perform better traffic engineering, and provide better service to customers. The P2P users benefit from faster content downloads, increased locality of query responses, and improvement in P2P scalability through reduction in overhead traffic.

We extend the ISP-P2P collaboration concept to propose collaboration between multiple-ISPs by exchanging summaries of network information through the respective oracle servers. This will enable P2P and other applications to get estimates of the path properties to potential neighbours/servers both within and outside their ISPs. Using simulation results with very large topologies, we show the benefits of multiple-ISP collaboration by comparing its performance with a bandwidth-based neighbour selection scheme in P2P systems. We also show how this concept can be leveraged to build a global coordinate system, and discuss how it differs from existing coordinate systems. Lastly, we examine the viability of using the oracle service to reduce pollution in P2P file-sharing systems while preserving network locality.

Zusammenfassung

Peer-to-Peer (P2P) Systeme verursachen heutzutage mehr als die Hälfte des Internetverkehrs, und eine wachsende Anzahl von Applikationen, z.B. Bittorrent, eDonkey, Joost, Skype, GoogleTalk und P2P-TV nutzen die P2P-Methodik. P2P-Systeme errichten Overlays auf der Applikationsschicht, unabhängig von Internet-Routing und ISP-Topologien. Dies führt zu zusätzlichen Verkehr aufgrund der Messungen sowie ineffizientes Routing für P2P-Benutzer. Während auf der einen Seite die P2P-Applikationen den Broadband-Access Markt treiben, verringert sich auf der anderen Seite die Anzahl der Nutzer der traditionellen Telefonie. Außerdem verursachen P2P-Applikationen auch ein Traffic-Engineering Problem für die ISPs. In diesem Sinne ist P2P einen Dilemma für die ISPs! Einige ISPs haben reagiert, indem sie P2P-Verkehr durch Bandwidth-Shaping blockieren, dies allerdings wenig erfolgreich. Manche P2P-Applikationen versuchen die Netzwerk-Latenzzeit zu potentiellen Nachbarn zu messen, um die hoch-performanten Netzwerkpfade wählen zu können. Allerdings haben diese Schritte die Routingkonflikte zwischen ISPs und P2P-Systeme nicht lösen können. Unsere Messungsstudie und die Visualisierungs-basierte Analyse haben gezeigt, dass die P2P-Overlay-Topologie nicht mit der Topologie der Autonomous Systems (AS) im Internet korreliert, und dass eine größere Zahl von Overlay-Peerings die ISP-Grenzen mehrmals überschreiten.

In dieser Arbeit stellen wir eine einfache, generische und einzigartige Lösung vor, die es ISPs und P2P-Systemen erlaubt, miteinander zu kooperieren. Wir schlagen vor, dass die ISPs einen Server betreiben, den wir *Orakel* nennen wollen, der P2P-Nutzeren hilft, geignete Nachbarn zu finden. Der P2P-Nutzer schickt eine Liste von potentiellen Nachbarn zum Orakel, der die Liste der IP-Addressen anhand von verschiedenen Parametern sortiert. Zum Beispiel würde ein ISP Nutzer des eigenen Netzes bevorzugen, so dass der P2P-Verkehr nicht nach draußen fließt. Weiterhin kann der ISP Nachbarn mit besserer Netzanbindung oder geringeren Delays bevorzugen, oder diejenigen die geographisch näher sind (z.B. selbe Stadt, selber PoP). Das Orakel gibt die sortierte Liste an den P2P-Nutzer zurück, der sich dann mit dem Nachbarn verbindet, der von dem Orakel empfohlen wurde. Dies führt nicht nur zu reduzierten Kosten und vereinfachtem Routing für die ISPs, sondern führt auch zu verbesserter Performanz für P2P-Nutzer im Sinne von höherer Bandbreite und geringerer Verzögerung. Hierdurch kooperieren P2P-Systeme und ISPs in einer Form von der beide profitieren.

Wir haben eine umfangreiche Analyse von diesem Vorschlag für unterschiedliche Modelle von P2P-Systemen durchgeführt. Hierfür kamen Graph-Experimente, Testbed-Implementierungen, Planetlab-Installationen und Paketebene-Simulationen zum Einsatz. Die Ergebnisse der Graph-Experimente zeigen, dass P2P-Nutzer unter der Verwendung des Orakels in der Lage sind, die meisten Peerings innerhalb der ISP-Grenzen zu halten, ohne die strukturelle Eigenschaften von P2P-Overlays negativ zu beeinflussen. Eine theoretische Analyse der Netzauslastung (Congestion), die durch kürzere Netzwerkpfade von P2P-Links verursacht werden, zeigte, dass die Netzauslastung nahe an dem theoretischen Optimum liegt. Dies resultiert aus der Tatsache, dass nahezu alle Overlay-Peerings in Übereinstimmung mit den ISP-Routing-Policies gebildet wurden. Anhand von Testbed- und Planetlab-Experimenten konnte die Machbarkeit des ISP-P2P Kooperationsschemas mit realen P2P-Systemen nachgewiesen werden. Des weiteren hat das Experiment gezeigt, dass

die Skalierbarkeit von P2P-Systemen sich signifikant verbessert und keine negativen Auswirkungen auf das Antwortverhalten auf Suchanfragen in P2P-Netzwerken resultieren. Die P2P-Nutzer sind so in der Lage die gewünschten Daten auf verfügbaren P2P-Knoten in geringerer Netzwerkdistanz zu finden.

Durch intensive Simulationen auf Paketebene haben wir die oben genannten Ergebnisse unter Verwendung des Gnutella P2P-Protokolls mit Churn-Verhalten verifizieren können. Die Perfomanzverbesserung für ISPs und P2P-Nutzer wurde durch Metriken, wie Intra-AS Datenaustausch und Daten-Downloadzeiten, quantifiziert. Dabei wurden in der Simulation verschiedene mathematische Modelle zur Abbildung von Benutzerverhaltensmustern (z.B. Churn, Datenverfügbarkeit, Suchbegriffe) als auch unterschiedlichen ISP-P2P-Topologien angewendet, um die resultiernden Effekte auf die Endnutzer-Performanz in realistischen, best-case und ungünstigen Szenarien zu studieren. Es zeigte sich, dass sich die Vorteile des vorgeschlagenen ISP-P2P-Kooperationsschemas auf alle simulierten Szenarien auswirken. ISPs sind so in der Lage, Kostenersparnisse zu realisieren, da ein großer Anteil des P2P-Verkehrs innerhalb des eigenen Netzwerks bleibt. Zusätzlich ermöglicht das Konzept ein besseres Traffic-Engineering und bietet dem Kunden eine höhere Servicequalität. Der P2P-Nutzer profitiert von schnelleren Downloads, verbesserten Antwortverhalten auf Suchanfragen sowie einer verbesserten Skalierbarkeit des P2P-Systems durch die Reduktion von Overhead-Traffic.

Wir erweiterten das ISP-P2P-Kooperationskonzept, so dass verschiedene ISPs durch den Austausch von aggregierten Netzinformationen kooperieren können. Dies ermöglicht P2P und anderen Applikationen eine Schätzung der Netzwerkpfad-Eigenschaften zu potentiellen Nachbarn, innerhalb und außerhalb des ISP-Netzes. Mit Hilfe von sehr großen Topologie-Simulationen haben wir die Vorteile der ISP-Kooperation durch den Performanzvergleich mit bandbreite-basierten P2P-Systemen aufgezeigt. Des weiteren zeigen wir auf, wie dieses Konzept zu einem Global Coordinate System ausgebaut werden kann. Letztendlich untersuchten wir die Machbarkeit des Orakel-Services, um die “Pollution” in P2P file-sharing-systemen zu reduzieren, und gleichzeitig Netzwerklokalität aufrecht zu erhalten.

Publications

Parts of this thesis have been published:

Journals

Vinay Aggarwal and Anja Feldmann
Locality-Aware P2P Query Search with ISP Collaboration
Networks and Heterogeneous Media, Vol. 3, Nr. 2, 2008

Vinay Aggarwal, Anja Feldmann, Robert Goerke, Marco Gaertler and Dorothea Wagner
Modelling Overlay-Underlay Correlations Using Visualization
Telektronikk Journal, Nr. 1, 2008

Vinay Aggarwal, Anja Feldmann and Christian Scheideler
Can ISPs and P2P systems co-operate for improved performance?
ACM SIGCOMM Computer Communications Review (CCR), Vol. 37, Nr. 3, 2007

Conferences and Workshops

Vinay Aggarwal and Anja Feldmann
ISP-Aided Neighbor Selection in P2P Systems
Internet Engineering Task Force (IETF) P2P Infrastructure Workshop, Boston, USA, May 2008

Vinay Aggarwal and Anja Feldmann
ISP-Aided Neighbor Selection in P2P Systems
RIPE 56, Berlin, Germany, May 2008

Vinay Aggarwal, Obi Akonjang and Anja Feldmann
Improving User and ISP Experience through ISP-aided P2P Locality
11th IEEE Global Internet (GI'08) Symposium, Phoenix, USA, April 2008

Vinay Aggarwal, Obi Akonjang, Anja Feldmann, Sebastian Mohrs and Rumen Tashev
Reflecting P2P User Behaviour Models in a Simulation Environment
16th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), Toulouse, France, February 2008

Vinay Aggarwal and Anja Feldmann
ISP-aided Biased Query Search for P2P Systems in a Testlab
European Conference on Complex Systems (ECCS), Dresden, Germany, October 2007

Vinay Aggarwal, Anja Feldmann and Roger Karrer
An Internet Coordinate System to Enable Collaboration between ISPs and P2P systems
11th International Conference on Intelligence in Service-Delivery Networks (ICIN), Bordeaux, France, October 2007

Vinay Aggarwal, Anja Feldmann, Marco Gaertler, Robert Goerke and Dorothea Wagner
Analysis of Overlay-Underlay Topology Correlation using Visualization
5th IADIS International WWW/Internet Conference, Murcia, Spain, October 2006

Vinay Aggarwal, Anja Feldmann and Sebastian Mohrs
Implementation of a P2P system within a network simulation framework
European Conference on Complex Systems (ECCS), P2P-Complex Workshop, Paris, France, November 2005

Vinay Aggarwal, Stefan Bender, Anja Feldmann and Arne Wichmann
Methodology for Estimating Network Distances of Gnutella Neighbors
INFORMATIK 2004 - Informatik Verbindet, 34. Jahrestagung der Gesellschaft fuer Informatik (GI), Ulm, Germany, September 2004

News Articles

Heise Netze
“Orakel” soll P2P-Datenverkehr optimieren
http://www.heise.de/newsticker/meldung/107597, 08 May 2008

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Peer-to-Peer (P2P) systems are self-organizing systems of autonomous entities called peers. P2P systems have recently gained a lot of attention in the social, academic and commercial communities. They enable the sharing of computer resources and services by direct exchange between peers. The shared resources can be disk storage space, processing power, or unused bandwidth at the network edges. P2P systems are being increasingly used in a wide variety of applications, e.g., file sharing, distributed storage, content delivery, distributed computing, telephony/chat, and games. Some popular applications running on P2P systems include BitTorrent, Gnutella, eDonkey, KaZaa, Joost, P2P-TV, Chord, Skype, IRC, and SETI@home.

Measurement studies have consistently shown that P2P applications contribute more than half of the total Internet traffic today [18, 106, 50, 86, 46, 102, 128]. While Web and FTP were the dominant Internet protocols in the 1990s, the situation has changed drastically since the advent of P2P applications in 2001. The fraction of P2P traffic in the Internet has been growing steadily since 2001, and has now overshadowed Web, E-mail and other forms of traffic in the Internet [18]. A measurement study as recently as 2007 reports that P2P traffic made up to 70% of the Internet traffic in Germany [46]. Other reports also concur that P2P remains the dominant protocol in the Internet today [102].

The users of P2P systems benefit from the efficient use of resources, reduced infrastructure costs, more freedom, higher scalability, and no single point of failure. However, routing of traffic in P2P systems often causes serious challenges to the Internet Service Providers (ISP). P2P systems form overlays at the application layer, which are virtual networks formed on top of the underlying Internet routing infrastructure. As such, the logical paths and links of an overlay lie on top of the physical paths set up by intra-domain (e.g., OSPF, MPLS, IS-IS) and inter-domain (e.g., BGP) routing protocols running in the Internet underlay. Hence, when the overlay nodes cooperate with each other to route data on behalf of any pair of communicating overlay nodes, the traffic is still carried through the physical Internet routing paths.

It has been shown that overlay routing can enable users access to paths with potentially better performance than those made available by the Internet [6, 94]. However, ISPs use traffic engineering (TE) to provide better routing performance to their customers [7, 27]. This leads to the situation that P2P systems reinvent and re-implement a routing system whose dynamics interact with the dynamics of the Internet routing system [96]. The goals of overlay routing and ISP's traffic engineering are not aligned. An overlay tries to find optimal routing paths between its own peers, while the ISP has to keep in mind the whole network performance, which includes all the underlay as well as the overlay users. This misalignment of goals not only leads to duplication of routing functionality, but also to inefficient routing path oscillations and triangle inequalities.

Put another way, the widespread use of P2P systems has put the ISPs in a dilemma! On the one hand, P2P applications are one of the major reasons cited by Internet users for upgrading their Internet access to broadband, thus increasing ISP revenues [66, 128]. On the other hand, ISPs find that P2P traffic poses a significant traffic engineering challenge [52, 86]. P2P traffic often starves other applications like Web traffic of bandwidth [100], and swamps the ISP network. In some cases, ISPs have resorted to limiting the amount of P2P traffic in their networks by traffic-shaping or blocking it [81]. However, such measures have not been successful [82], and have also led to bad publicity as well as legal problems for the concerned ISPs [117]. Besides, with sufficient computing power available at the end-hosts, P2P protocols also have the option to rely on encryption to circumvent unilateral ISP control.

1.2 Our Contribution

In this thesis, we begin with a measurement study of the Gnutella P2P protocol, and find that its overlay topology is not correlated with the underlying Internet topology. A large number of overlay peerings cross AS boundaries multiple times. A deeper analysis using a unique visualization technique confirms this result, and shows how overlay topologies differ from random networks.

We then propose a simple, general and unique solution that enables ISPs and P2P systems to collaborate with each other. We propose that an ISP hosts a server, which we call the *oracle*, that helps P2P users choose “good” neighbours. Each P2P user has a list of potential neighbours to whom it can connect or download content from. Instead of choosing neighbours independently, we propose that the P2P user sends the list of its potential neighbours to the oracle. The oracle ranks this list of IP addresses based on a number of factors, that an ISP decides individually. For example, the ISP can prefer peers within its network, to prevent traffic from leaving its network. Further, it can choose better bandwidth or lesser delay nodes, or those that are geographically closer (same city, same PoP) within its network. The oracle returns this sorted list to the P2P user, who can then benefit from the knowledge of the ISP and connect to a neighbour recommended by the oracle. This will not only reduce costs and ease routing for the ISPs, but will also provide improved performance for P2P users in the sense of higher bandwidth and lesser delay. In this way, ISPs and P2P systems can cooperate so that both of them benefit.

We conduct a comprehensive analysis of this proposal using graph experiments, testbed implementation, Planetlab deployment, and packet-level simulations on various models of P2P systems. The graph results show that P2P users, on consulting the oracle, are able to keep most of their peerings within the ISP boundaries, without any adverse effects on the overlay graph structural properties like small node degree, small path length, small graph diameter, and graph connectedness. The P2P topology is correlated with the Internet AS topology, with dense subgraphs of peerings local to the AS boundaries.

A theoretical analysis of the congestion caused by shorter network paths of P2P links reveals that the congestion in the network is close to the theoretical optimum. This comes with the advantage that almost all the overlay peerings are formed in accordance with the ISP policies. Through testbed implementation and Planetlab deployment, we show that the ISP-P2P collaboration scheme for neighbour selection in P2P systems is feasible with real P2P systems. The experiment results also show that the scalability of P2P systems improve considerably, and there is no adverse effect on the query search phase of P2P networks. The P2P users are still able to locate all available content from nodes at shorter network distances.

Using extensive packet-level simulations, we verify the above results with the Gnutella P2P protocol under churn. We quantify the performance improvements for ISPs and P2P users, using metrics like intra-AS content exchange and content download times. We simulate multiple ISP and P2P topologies, as well as a range of user behaviour characteristics, namely, churn, content availability and query patterns, using different mathematical distributions. This enables us to study the effects of realistic, best-case and adverse scenarios on end-user performance. We show that the benefits of our proposed ISP-P2P collaboration scheme hold across a range of user behaviour scenarios and ISP/P2P topologies. The ISPs are able to save costs by keeping large amount of traffic within their network, perform better traffic engineering, and provide better service to customers. The P2P users benefit through faster content downloads, increased locality of query responses, and improvement in P2P scalability through reduction in overhead traffic.

We then extend the ISP-P2P collaboration concept to propose collaboration between multiple ISPs by exchanging summaries of network information through the respective oracle servers. This will enable P2P and other applications to get estimates of the path properties to potential neighbours/servers both within and outside their ISPs. Using simulation results with very large topologies, we show the benefits of multiple-ISP collaboration by comparing its performance with a bandwidth-based neighbour selection scheme in P2P systems. We also show how this concept can be leveraged to build a global coordinate system, and discuss its advantages as compared to existing coordinate systems. Lastly, we examine the viability of using the oracle service to reduce pollution in P2P file-sharing systems while preserving network locality.

1.3 Chapter Overview

The thesis is structured as follows.

Chapter 2. We give background information about P2P systems that is helpful for understanding this thesis. We introduce overlays and P2P systems, classification of P2P systems, overlay routing, and its relation with Internet routing. We also introduce the Gnutella P2P protocol and discuss the reasons for using it in our experiments. We then discuss the principal tools used in this thesis to study P2P systems like simulation frameworks, testbed, and Planetlab. This is followed by an overview of the implementation of the Gnutella protocol in SSFNet simulation framework, which is used for extensive packet-level simulations in Chapter 6.

Chapter 3. We begin with a measurement study of the Gnutella P2P network, and find that its overlay topology is not correlated with the Internet AS topology. We develop a visualization technique to study overlay-underlay correlations, and confirm our findings. We also compare the measured overlay topology with a random overlay network.

Chapter 4. We outline the principal proposal of ISP-P2P collaboration through the use of the oracle service. We discuss how the oracle service can be realised, and introduce a model algorithm for peer selection that is beneficial to both ISPs as well as P2P users.

Chapter 5. To evaluate the proposal, we perform experiments on the graph structural properties, e.g., node degree, path length, graph diameter, connectedness, and correlation of ISP-P2P topologies, of a generalized overlay. Using the principle of flow conductance, we compare the congestion caused by localized overlay graphs with the theoretical optimum values. We then make a feasibility analysis of the proposal through experiments in a testbed with a real P2P system, as well as deployment in the Planetlab.

Chapter 6. We use a simulation framework to perform rigorous analysis on various aspects of the ISP-P2P collaboration concept. Using the packet-level simulator SSFNet which supports TCP, we study the effects of various user behaviour characteristics, namely, churn, content availability, and query patterns on end-user experience metrics for ISPs and P2P users, e.g., content localization, download times, query search performance, and P2P scalability. We perform the experiments across a range of ISP and P2P topologies, as well as for multiple models of the user behaviour characteristics, e.g., realistic case, worst-case, and best-case scenarios.

Chapter 7. We extend the oracle concept by proposing collaboration between multiple ISPs, so that P2P users can get estimates of the path properties to potential neighbours both within and outside their ISPs. Using the PeerSim P2P simulator we run very large-scale simulations, and compare the performance of multiple-ISP collaboration with bandwidth-based P2P neighbour selection. We also show how the concept of multiple-ISP collaboration can be leveraged to design a global coordinate system. We discuss which metrics can be provided by the proposed coordinate system, and how it differs from existing coordinate systems.

Chapter 8. We propose another extension to the oracle proposal, to reduce pollution in P2P file-sharing systems. We propose that the oracle consider the reputation of potential neighbours along with their proximity information, while recommending them to a P2P user. Using large-scale simulations, we show the benefits to ISPs as well as P2P users.

Chapter 9. We summarize the contributions of this thesis, and discuss ongoing and future work that is inspired by this thesis.

2 Background

In this chapter, we provide the reader with background knowledge about P2P systems. We introduce overlays and P2P systems, their classifications, how routing works in the Internet, and its relation to P2P routing. We then introduce the Gnutella P2P protocol, and discuss the reasons for using it in many of our experiments. This is followed by an overview of the experimental tools, namely simulation frameworks, testbed, and Planetlab, that are employed in this thesis to study P2P systems. Finally, we report on the implementation of the Gnutella P2P protocol in the packet-level simulation framework SSFNet.

2.1 Overlays and Underlays

In recent times, the design of many real-world applications has changed from a monolithic structure to modular, yet highly customizable services. As an implementation from scratch is usually too time-consuming and expensive, these services are superimposed on an already existing underlay infrastructure as an overlay.

A well-known example arises in logistics. The highways and streets we use everyday constitute a huge transport network. However, traffic in this network is far from structured. In fact, countless companies and institutions rely on this network to accomplish their regular shipping of commodities and services, and by doing so, they cause the traffic on the road network to develop in certain patterns. In technical terms the road network constitutes an *underlay network* while the commodity exchange network of a set of companies implicitly building upon this network forms an *overlay network*.

The overlay network uses the underlay to realize its tasks. However, this abstraction entails a certain trade-off, namely independence versus performance. There is clearly a crucial interdependence between the overlay and the underlay networks. The emergence of overlay networks heavily affects and poses new requirements on the underlay. The major advantage of overlays is that they provide high-level functionality while masking the intrinsic complexity of the underlay structure. However, overlays rely on the underlay to provide them with basic connectivity. Therefore, the intrinsic features of the underlay network determine the efficiency of the overlay.

Another underlay network of prime interest is the Internet, which serves as the workhorse of countless data transfers, e.g, Web, Email, multimedia services, and file-sharing protocols. Almost anytime we use the Internet, we participate in some overlay network that uses the physical Internet (comprised of routers, links, cables, wires) to actually convey the data packets. Interestingly enough, the Internet itself started as an overlay built over the telephone network underlay. Within the Internet, a particular breed of overlays that has received a lot of attention lately are peer-to-peer (P2P) applications [106], which range from file-sharing systems like Gnutella and BitTorrent, to real-time multimedia streaming like P2P-TV, to VoIP phone systems like Skype and GoogleTalk.

2.2 Modeling Overlays and Underlays

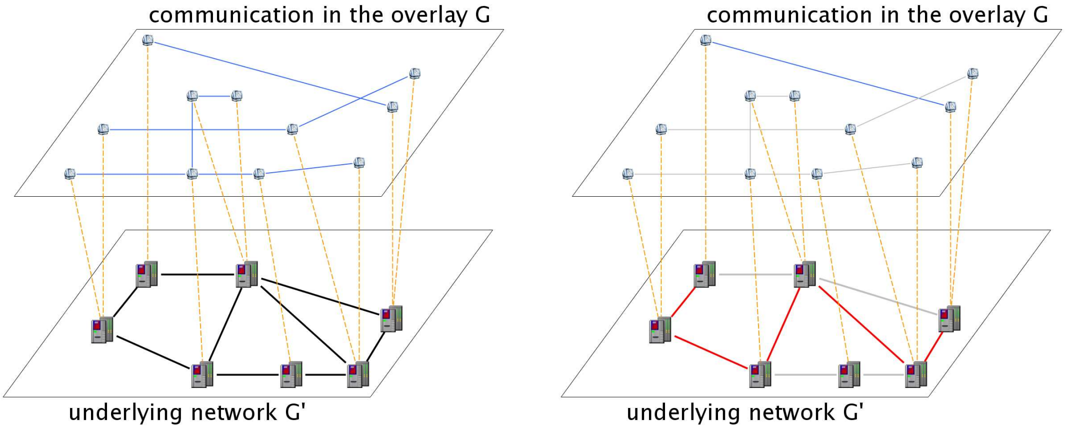
An overlay consists of a network structure that is embedded into another network. More precisely, each node of the overlay is hosted by a node in the underlay, and every edge of the overlay induces at least one path between its end-nodes' hosting nodes in the underlay. The formal definition of an overlay is as follows.

An *overlay* is defined by a four-tuple $\mathscr{O} := (G, G', \phi, \pi)$, where

- $G = (V, E, \omega)$ and $G' = (V', E', \omega')$ are two weighted graphs with $\omega \colon E \to \mathbb{R}$ and $\omega' \colon E' \to \mathbb{R}$, where V and V' denotes the set of vertices, E and E' denote the set of edges, while ω and $omega'$ are the weight functions
- $\phi \colon V \to V'$ is a mapping of the nodes of G to the nodeset of G', and
- $\pi \colon E \to \{p \mid p \text{ is a (un-/directed) path in } G'\}$ is a mapping of edges in G to paths in G' such that $\{\text{source}(\pi(\{u,v\})), \text{target}(\pi(\{u,v\}))\} = \{\phi(u), \phi(v)\}$.

The interpretation of the above definition is that G models the overlay network itself, the graph G' corresponds to the hosting underlay, and the two mappings establish the connection between the two graphs. An example is given in Figure 2.1.

Figure 2.1 Modeling an overlay and its induced underlay $\mathscr{O} := (G, G', \phi, \pi)$. The mapping ϕ is represented by dashed lines between nodes in G and G'.


communication in the overlay G
underlying network G'
communication in the overlay G
underlying network G'

(a) Both networks G and G' with the mapping ϕ.

(b) Highlighting one edge e in G and the corresponding path $\pi(e)$ in G'.

As direct communications in the overlay, which correspond to edges of G, are realized by routing data along certain paths in G', not all parts of the underlay graph are equally important. In order to focus on the relevant parts, we associate an *induced underlay* with an overlay. The induced underlay corresponds to the subgraph of the underlay graph that is required to establish the communication in the overlay graph. It is defined as follows.

Given an overlay $\mathscr{O} := (G = (V, E, \omega), G' = (V', E', \omega'), \phi, \pi)$. The *induced underlay* $\widetilde{\mathscr{O}} := H := (V'', E'', \omega'')$ is a weighted graph, where

- $V'' := \{v \in V' \mid \exists e \in E \colon \pi(e) \text{ contains } v\}$,

- $E'' := \{e' \in E' \mid \exists\, e \in E : \pi(e) \text{ contains } e\}$, and
- $\omega''(e') := \sum_{e \in E} \omega(e) \cdot [e' \text{ contained in } \pi(e)]$.

The weight function ω'' is also called *appearance weight*.

The definition of ω'' is given in the Iverson Notation [54]. The term inside the squared parentheses is a logical statement, the term evaluates to 1 if its value is true, and to 0 otherwise. Note that the defined weight can be interpreted as the load caused by the communication and is thus independent of a weighting in the underlay network.

2.3 Peer-to-Peer Systems

A peer-to-peer (P2P) system is a self-organizing, networked community of equal peers, realized as an overlay on top of the underlying Internet infrastructure. P2P systems are increasingly being used as a convenient means to share resources and content over the Internet. The advent of P2P applications has affected the way content is stored, processed and (re)distributed. The creation, distribution, management and consumption of content is no longer solely controlled by dedicated content providers using centralized servers at the core of the Internet, as is the case with client-server based applications like HTTP and FTP. Rather, in the case of P2P systems, content management is shifting to users at the edge of the Internet. Users generate and manage their own content, and share it with other users across the globe by interacting directly, often without the need of centralized servers. As the user base increases, so too does the volume of generated P2P traffic.

The term *servent* is often used to describe a peer, since it concurrently acts as both a client and a server. To join the community, peers need to locate and establish connections with already active (online) neighbours. This is done by using either a vendor-configured list of peers (for first-time peers), a list of cached peers from previous sessions, or out-of-band, using addresses obtained from other sources such as a Web server. Despite being built independent of the Internet underlay, P2P signalling and content traffic still physically flows via links in the underlay. Most often, neighbouring peers on the overlay are actually physically separated by multiple subnetwork hops or geographical continents at the underlay. The P2P approach is not only revolutionizing the way computers communicate on the Internet, but is also finding widespread acceptance and implementation in a number of areas, principal among them being file sharing, telephony, audio/video media streaming, and discussion forums.

2.4 Classification of P2P Systems

P2P systems can be classified based on their degree of centralization. **Pure** P2P networks are those in which there is no central server or router, and all the peers have equal roles - they act as clients as well as servers. Examples of pure P2P networks are Gnutella and Freenet. In **hybrid** P2P networks, there is a central server that maintains information on peers (such as the resources hosted by them), and responds to requests for such information. The peers in the hybrid P2P networks are responsible for hosting resources and sharing it with other peers that request these resources. The information exchange occurs typically through the central server. Examples of such networks are BitTorrent, Napster, and JXTA.

A P2P overlay network essentially consists of all the participating peers, also known as network nodes, connected by logical links. If a peer knows the location of another peer in the P2P network, there exists a directed edge from the former to the latter node in the overlay. Depending on how the nodes in the overlay are connected to each other, P2P networks can be classified as unstructured or structured.

An **unstructured** P2P network is formed when the overlay links are established arbitrarily. Each peer connects to a set of neighbours randomly when it joins the network. The protocol is simple, but it can keep the nodes highly connected even in the event of major disasters. To search for content, a peer usually floods search queries to all its connected neighbours. As the P2P topology is not related to the location of data, a peer has no idea about where the desired content is available, thus leading sometimes to a “blind search”. This means that queries are simply flooded iteratively until the desired content is found, or the search messages have traversed a certain number of hops. Flooding causes a high amount of signalling traffic in the network, which limits the scalability of such networks as their size grows. To improve scalability, newer versions of such networks use a hierarchical topology, with high performance “supernodes” maintaining the overlay structure by connecting with each other and forwarding only a small number of messages to their shielded nodes. Some networks also use random walks to locate content. Gnutella, FastTrack/KaZaa, and Skype are examples of such networks.

In a **structured** P2P network, nodes are organized in an orderly fashion by employing a globally consistent protocol to ensure that any node can efficiently route a search to another peer that has the desired content in a small number of hops. Such networks are typically based on a distributed hash table (DHT), in which a hashing mechanism is used to assign ownership of each file to a particular node. The structured P2P system provides the interface for storing as well as retrieving the content from the nodes to which it is assigned. Each node typically maintains $O(\log n)$ pointers to other nodes, where n is the number of network nodes. To locate a file, the average number of application-level hops required is $O(\log n)$. Some well known examples of structured P2P networks are Chord, Pastry, Tapestry, and CAN.

2.5 Routing in the Internet

To better appreciate the issues associated with the routing of P2P traffic, let us first consider how Internet routing works. At the network layer, the Internet can be viewed as a collection of subnetworks or Autonomous Systems (ASes) that are interconnected together. An AS is a segregated routing domain consisting of a group of routers with independent routing policies under a single administrative control, operated typically by the same ISP or belonging to the same company network. Routers within the same AS run the same routing algorithm and have information about each other. Gateway routers, typically located at the AS boundaries, are responsible for forwarding packets to destinations outside the AS.

Data to be sent across the Internet is broken down into packets, each of which is transmitted independent of the others. The packet formats are defined by the Internet Protocol (IP), which also assigns IP addresses to the source and the destination. Routing through the Internet is done on a per-IP prefix basis and depends on protocols for routing within individual ASes and for routing between ASes.

Border Gateway Protocol (BGP) [87], a policy routing protocol, is the de-facto standard for routing between ASes. It is used to ensure that traffic exchanged between ASes respects the contractual agreements between the ASes [72]. BGP enables each AS to (i) obtain subnet reachability information from neighbouring ASes, (ii) propagate the reachability information to all routers internal to the AS, and (iii) determine good routes to subnets based on the reachability information and AS policy.

An intra-AS routing protocol [40] determines how routing is performed within an AS. Popular examples of such a protocol are Routing Information Protocol (RIP) and Open Shortest Path First (OSPF). RIP is a distance-vector protocol that uses hop count as a cost metric, with each link having unit cost. RIP attempts to find the shortest path from source to destination, and works for small networks. OSPF is a link-state protocol that uses flooding of link-state information and a Dijkstra least-cost path algorithm. With OSPF, a router constructs a topological map of the entire AS, and the router then locally runs Dijkstra's shortest path algorithm to determine a shortest path tree to all subnets within the AS. We thus note that while inter-AS routing is governed by policies, intra-AS routing is based on shortest paths or least costs.

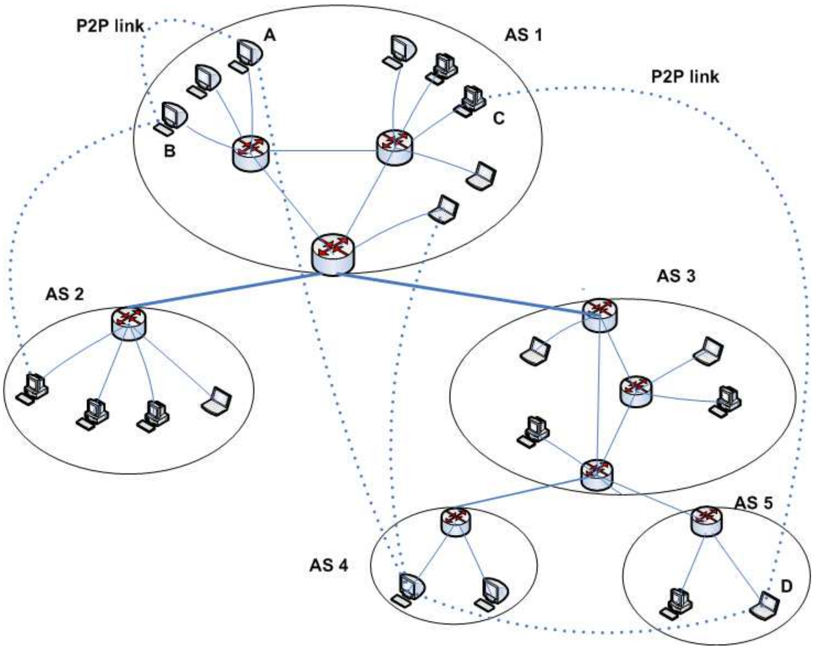
The AS network possesses an implicit hierarchical structure where the ASes can be categorized into three broad categories [30, 72]: (i) backbones, (ii) national, regional or local providers, and (iii) customers. An AS typically buys Internet connectivity from one or more transit providers, which are referred to as upstreams. Such a contractual relationship between ASes is called a customer-provider relationship. This differs from a peering relationship where the link cost is shared by the peering ASes. However, a peering link is only used to exchange traffic between the peers and their customers, no transit traffic should flow though a peering link. An AS that has no upstream provider is called a tier-1 or level-1 AS. All tier-1 ASes peer with each other and build the core of the Internet, while ISPs that do not provide transit services and simple customers, e.g., multi-homed ASes build up the periphery of the Internet AS network. The graph of the ASes, where nodes represent different ASes, and edges correspond to traffic trade agreements between the ASes, provides us with an abstraction of the Internet underlay.

Relation with P2P routing

Routing in P2P systems is in stark contrast to Internet routing. Most P2P systems implement their own routing [6] on top of the Internet by building an overlay network. In most cases routing is no longer done on a per-prefix basis; rather queries are disseminated via flooding [35] or random walks [20] in unstructured P2P networks, or via the routing system of DHT-based P2P networks [107]. Answers can either be sent directly via the underlay routing or through the overlay network by retracing the query path [35]. Since P2P systems choose their neighbours without considering the underlay, traffic along an overlay link often traverses multiple AS or router hops in the Internet.

Figure 2.2 shows a very simple Internet topology, consisting of 5 ASes. Each AS consists of border routers, internal routers, and end system hosts. The dotted lines correspond to overlay links between two nodes in the P2P system, and two peers connected via such a link are considered P2P neighbours. We observe that while the P2P neighbours A and B are located in the same AS, this is not the case for P2P neighbours C and D. Even though the application layer distance between C and D is 1, they are physically located in two different ASes (i.e., AS 1 and 5), which may be as far away as Europe and Australia. The actual path between the P2P neighbours in this case goes through AS 3, crossing multiple AS boundaries and access links, which can account for significant performance penalties in terms of available bandwidth and latency. Hence, the notion of neighbourhood can differ significantly if seen from a P2P node's rather than an ISP's viewpoint. Neither unstructured

Figure 2.2 Lack of correlation between P2P links and the Internet AS topology



nor structured P2P networks take the Internet topology into account when forming neighbourhoods, and are thus not aligned with the Internet topology.

2.6 The Gnutella Protocol

In this thesis, a large part of the experiments have been based on a Gnutella-like unstructured file sharing system. We introduce the Gnutella protocol in this section, and explain the reasons behind choosing it for our experiments.

2.6.1 Introduction

Gnutella is one of the first decentralized P2P file sharing systems, and gives its participants the ability to share and locate resources hosted by other members of the network. It is a P2P system in the sense that, there is no distinction of members into clients and servers, rather, all members are equal and can initiate as well as serve requests.

A participant of the Gnutella network, called a servent or a peer or a node, is a computer system running an implementation of the Gnutella protocol. When launched, a servent searches for other servents in the Gnutella network, to whom it can connect. Each servent may or may not share any resources, and can search for desired resources within the network. While the general notion of resources tends to be multimedia files, the resources can actually be anything from mapping to other resources, cryptographic keys, files of any type, to meta-information on key-able resources.

The servents interact with each other to share information on the resources that they offer, to query for desired resources, and to obtain responses to their queries. Based on the results, a servent decides which resource to obtain from which servent, and then initiates the actual download of the resource.

Gnutella uses several different messages for resource lookup and overlay management. These messages are:

1. **Ping**: It is a simple Hello-like message sent by a servent to actively discover other hosts in the network. It is also used to declare its own presence in the network.

2. **Pong**: It is a response to the `Ping`. A servent includes its own address and some information about the data (like number of files, etc.) that it shares in the network.

3. **Query**: This message is used to search the Gnutella network for desired resources. It is something like a simple question - "Does anybody have this file?"

4. **QueryHit**: It is a response to a `Query`. A servent possessing the requested resource replies with some information about its network connectivity (speed, etc.) to allow the questioner to suitably choose which node to download the resource from.

5. **Push**: A special message to allow a firewalled servent to share data.

On initial startup, a servent must bootstrap and find at least one other peer. Different methods have been used for this, including a pre-existing address list of known peers shipped with the software, using updated websites with lists of known nodes (called GWebCaches), or UDP host caches. The servent searches for additional servents by flooding `Ping` messages to its connected neighbours, which are answered by `Pong` messages. The search queries are flooded to all connected peers using `Query` messages, which are answered by `QueryHit` messages. To limit flooding Gnutella uses TTL (time to live) and message IDs. Messages are generated with a TTL value of 7, and this value is decreased by one with each overlay hop that the message traverses. Messages are discarded when they reach a TTL value of 0. When a node receives a `Query` message, it checks if it has content that satisfies the query search string. If yes, the node generates a `QueryHit` message with a TTL value of the hops value of the corresponding `Query` plus two, lists content files that match the query string in this message, and sends it to the Gnutella node that originated the `Query` message. Each `QueryHit`/`Pong` message traverses the reverse path of the corresponding `Query`/`Ping` message. When a node receives multiple `QueryHit` messages for its search query, it selects one of the nodes randomly, and initiates a direct file download from this node using HTTP. While the negotiation traffic is carried within the set of connected Gnutella nodes, the actual data exchange of resources takes place directly between the relevant servents using HTTP, similar to other P2P protocols like BitTorrent. In other words, the Gnutella network is only used to locate the nodes sharing the desired resources.

Due to scalability issues, the new version of Gnutella (version 0.6) takes advantage of a hierarchical design in which some high-bandwidth and high-performance 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 neighbours, both ultrapeers and leafs. Ultrapeers thus become responsible for routing of messages, thereby shielding leaf nodes and improving the efficiency and scalability of the Gnutella network. 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. For more details on the Gnutella protocol, we refer the reader to [36].

2.6.2 Reasons for Choosing Gnutella

We decided to use Gnutella for experiments due to a number of reasons. At the time of starting this thesis in 2003, it was one of the most popular P2P file sharing systems, with 2 – 3 million users. At least in the period up to 2006, it ranked in the top three P2P systems, with hundreds of thousands of users online simultaneously. Gnutella is an open-source system with a well-known protocol, which has attracted a healthy interest from researchers, e.g., [109, 90, 20, 84, 131, 31]. Hence, its characteristics are well understood, and the existing literature allows us to compare and contrast our experimental results with established behaviour patterns of Gnutella, both in simulation frameworks as well as in the real Internet. Also, some latest developments to its protocol, e.g., the query routing protocol (QRP), dynamic querying (DQ), development of Gnutella2, and Gnutella for mobile users (Symella) [112], have kept Gnutella a reasonably popular protocol.

More importantly, Gnutella represents a P2P system in the true sense, as it has no centralized servers. Hence, it is a good choice to evaluate the P2P methodology. The two-tier topology of the Gnutella network, comprising of ultrapeers (supernodes) and leaf nodes, is similar in concept to other popular P2P protocols, namely, FastTrack/KaZaa, eDonkey/eMule, and Skype. While there are differences regarding (i) the proportion of superpeers among all the nodes, (ii) the rate at which connections between leaves and superpeers change, and (iii) the criteria to decide promotion of leaves to superpeers, the basic topological properties are the same. Also, the content exchange occurs directly between the peers, outside of the P2P network, using the HTTP protocol. This feature is consistent across all file-sharing P2P systems, including BitTorrent.

While BitTorrent is the most popular P2P file-sharing system as of today, it does not provide any search facility. In this sense, it represents more an efficient distribution algorithm for downloading a given file, rather than a P2P network containing a large number of files. As this thesis aims to analyze the effects of localized overlay topologies on the neighbour selection of peers as well as on content search, and not just content exchange, we decided to use the Gnutella protocol for much of our analysis. We view “Gnutella” not as a single project or piece of software, rather as an open source protocol that is used by various P2P systems.

2.7 Tools to study P2P systems

In this section, we discuss the principal tools that are used to study P2P systems and overlay-underlay correlations in this thesis, namely, simulation frameworks, testbed and Planetlab.

2.7.1 Simulations

Simulations are a traditional tool for experimental study in Internet research. While they require one to model the P2P system code and user behaviour, they also enable experimenting with reasonably complex system models and fairly large topologies. It becomes feasible to tune multiple parameters and calibrate their effects on system performance in a simulation environment. One can easily design multiple different topologies, user behaviour models, and other such factors, which play an important role in the P2P system performance. For example, churn has become a major characteristic of most P2P applications. As the pattern of churn varies across different P2P systems, time of day and geographical region, reflecting these characteristics in a Planetlab setting can be a very challenging task. However, this can be achieved in a simulation framework by setting parameters appropriately. The same applies to other P2P characteristics like file-sharing, search strings, neighbour selection, etc. In this way, simulations allow the exploration of complicated scenarios that would otherwise be non-trivial to analyze. Because simulations often use more complex models than those used in the theoretical analysis or feasibility studies, they serve to check if the simplifying assumptions used in the simpler models do not invalidate the results. For a comprehensive discussion on the role (and challenges) of simulations in Internet research, we refer the reader to [26].

In this thesis, we employ three different simulation frameworks, depending on the purpose at hand.

- In Chapter 5, we use the Subjects framework as a graph simulator for a generalized overlay system to compare the graph structural properties of ISP-aided biased overlays with random overlay graphs. This enables us to explore large topologies involving thousands of overlay nodes as long as we focus purely on the graph properties.
- In Chapter 6, we implement the unstructured Gnutella P2P protocol in a packet-level simulator, the SSFNet simulation framework, to experiment with a real P2P system. We simulate the complete routing functionality of the P2P system, along with query search and content exchange, and perform experiments across a broad range of user behaviour patterns and ISP topologies. While we have the advantage of experimenting with an actual P2P system and simulating the packet transmission down to the TCP level of detail, the complexity of the network is limited to about 1000 P2P nodes.
- In Chapters 7 and 8, we focus on the content exchange phase of a generalized file-sharing P2P network to present results on multiple-ISP collaboration as well as reducing pollution in P2P systems. Here, we use the cycle-based application-level PeerSim P2P simulator. This allows us to scale to very large topologies (more than $100,000$ P2P nodes) with reasonably complex network topology models, albeit at a loss of TCP functionality.

A brief overview of these simulation frameworks is given below.

Subjects

The Subjects [95] 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. For a detailed overview of the Subjects environment, we refer the reader to [95].

SSFNet

The Scalable Simulation Framework (SSF) [105] is an open-source, Java-based, discrete-event simulations standard for simulating large and complex networks. 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 (e.g., IP, TCP, UDP, BGP4, and OSPF), network elements (e.g., hosts, routers, links, and LANs), and their various support classes. Link layer and physical layer modeling can be provided in separate components.

Domain Modeling Language (DML) is a public-domain standard for model configuration and attribute specification. It supports extensibility, inheritance and substitution of attributes. SSFNet models are self-configuring, i.e., each SSFNet class instance can autonomously configure and instantiate itself by querying network configuration files written in the DML format. The principal classes used to construct Internet models are organized into two frameworks, SSF.OS (for modeling the host and operating system components, e.g., protocols) and SSF.Net (for modeling network connectivity and configuring nodes and links). For a more comprehensive discussion of SSF, we refer the reader to [105].

PeerSim

Written in Java, PeerSim [76] is composed of two simulation engines: cycle-based and event-driven. The cycle-based engine is simple and more efficient, and allows for scalability. It supports direct communication between the P2P nodes, and allows for a set of protocols to run at each node, e.g., initiating a query and flooding it to neighbours, generating responses to a query, initiating content exchange, etc. The event-based engine supports transport layer simulation and operates on a set of explicitly defined events. It can run cycle-based protocols as well, and provides support for churn. While PeerSim models routers and some functionality of the transport layer, support for TCP is not yet provided. We use a combination of both the engines for our generalized P2P protocol.

2.7.2 Testbed

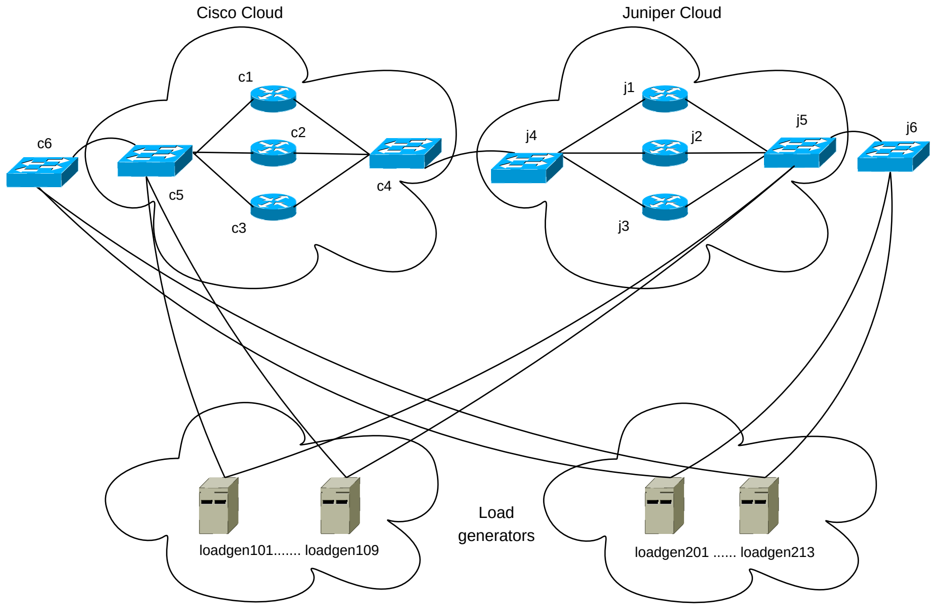
To be able to run actual P2P system code without having to model P2P networks and routing protocols in the experiments, we use a testbed facility. The advantage of using a testbed is that we can experiment with real traffic instead of simulated flows, and can configure network devices like routers, switches, and links to generate a variety of different network scenarios and traffic environments. We have control over the network entities, which enables us to perform a wide range of experiments using real applications, network stacks, and operating systems. Also, we have better control and visibility over the test environment as compared to running the experiments on the Internet, and can additionally eliminate the risk of inadvertently affecting the proper functioning of the Internet due to traffic generated by our experiments. Debugging and developing new applications hence becomes more feasible. However, the scale of experiments in a testbed is typically limited.

Hardware setup of the Testbed

The hardware setup of the testbed consists of the following devices:

- three Cisco 2691XM routers, named `c1,c2,c3`
- three Juniper M7i routers, named `j1,j2,j3`
- one Cisco 3750G24-TS switch, named `c4`
- three Cisco 2950SX-24 switches, named `c5,c6,j6`
- one Cisco 3500XL switch, named `j4`
- one Cisco 3550-12G switch, named `j5`
- nine Opteron-based load generator PCs, named `loadgen101` to `loadgen109`
- 13 Athlon-based load generator PCs, named `loadgen201` to `loadgen213`

Figure 2.3 Hardware layout of the testbed



A graphical representation of the setup is shown in Figure 2.3. The network is divided into four clouds: one cloud containing all the Cisco routers, another cloud containing all the Juniper routers, and two clouds of load-generators. The Cisco cloud and the Juniper cloud are connected to each other by the switches `c4` and `j4` and to the two load generator clouds by the switches `c5`, `j5`, `c6` and `j6`. Host-to-host connections can be set up either directly, using just a switch, or by using routed network links using one or more routers. All the PCs run Linux. We can design and setup different topologies on the testbed hardware, in order to perform realistic experiments with multiple different scenarios. The testbed is used for a feasibility study of our proposal in Chapter 5.

2.7.3 Planetlab

Planetlab [77] is a set of computers available as a testbed for computer networking and distributed systems research. It is organized as a large collection of computers, called nodes, distributed worldwide over the locations of attending research institutions, called sites. These nodes are running a special network-distributed Linux system which uses virtual machines to provide user access. All nodes are controlled by a central manager. Planetlab has more than 800 nodes located at more than 400 sites worldwide, though most of the sites are in North America and Western Europe. Each research project has a virtual machine access to a subset of the nodes for running experiments. A virtual machine is dynamically created and non-permanent, so it may be reset upon host configuration issues. All virtual machines on a Planetlab node have to share the node's limited resources like IP address, memory and disk space.

Running experiments on the Planetlab allows us to install modified P2P clients on machines spread throughout the globe, and observe their interaction with other P2P clients running in the Internet which are not influenced by us. The deployment of biased P2P clients on the Planetlab is discussed in Chapter 5.

2.8 Implementation of a P2P System within a Network Simulation Framework

To enable extensive experimentation with a real P2P system in a controlled environment, we have implemented the Gnutella P2P protocol in a network simulation framework. Due to support for routing as well as application layers, ability to configure different network topologies and user behaviour models, and availability of models for Internet protocols such as IP, TCP, HTTP, BGP, and OSPF, we choose the Scalable Simulation Framework SSFNet [105] for our experiments. In this section, we give an overview of the implementation of Gnutella in SSFNet, and delve into some of the design issues that we faced. We first explain the implementation, and then report on some experiences with SSFNet experiments. The implementation of the Gnutella protocol in SSFNet is used for extensive packet-level simulations in Chapter 6.

2.8.1 Gnutella Implementation

To implement Gnutella, we follow the Gnutella version 0.6 protocol RFC [35]. SSF provides the implementation for the lower layers of the IP stack, on which we “weave” the Gnutella protocol at the application layer. Naturally, a challenging aspect of the task is to fit the Gnutella code onto the SSF code, more specifically, the interaction of the two systems at the TCP layer. Some of the challenges included: in SSFNet, a node needs to tell its communication partner the exact size of the object being transferred. Hence we made changes to the SSFNet socket implementation so that a node can self-compute the size of objects being transferred through it. Also, there was no buffering support built into the sockets, i.e., one could not write any data to a socket unless it was free. So we added support for buffering into the SSFNet sockets.

We first code the Gnutella message header, followed by the message payload types, i.e., the four Gnutella messages `Ping`, `Pong`, `Query` and `QueryHit`. We take care to implement the Gnutella Generic Extension Protocol (GGEP) support for Gnutella messages, as it allows us to add extensions to the messages for experiments later on. While in reality, one IP packet may contain several Gnutella messages, and one Gnutella message may be split up among multiple IP packets, we simplified the implementation by assuming that each IP packet contains only one Gnutella message.

Each network node is assigned an IP address by SSF based on the network topology specified in the DML file. We use IPv4 addresses in our experiments. After implementing the network initialization, bootstrapping, handshaking and querying procedures, we approach the more complex issues like message routing, content search, query matching algorithm, flow control, and user behaviour characteristics at the application layer. Each of these issues is discussed below. We note that all the components of our implementation are in accordance with the Gnutella protocol RFC [35].

Message Routing in the Network

The processing and routing of `Ping`/`Pong` and `Query`/`QueryHit` messages is done as follows. A servent forwards an incoming `Ping`/`Query` message to all of its directly connected servents, except the one from whom it receives the `Ping`/`Query` message. There are some variations to this rule in case of servents using Flow Control, Pong Caching or Ultrapeers capabilities [35]. A servent decrements the TTL and increments the Hops field of the message header before forwarding it. If after decrementing, the TTL equals 0, the message is not forwarded. If a servent receives a

Ping/Query with the same message ID as it has received previously, it discards the message as it is a duplicate.

A Pong or QueryHit message is sent along the reverse path as that of the corresponding Ping or Query message respectively. Every servent implements a forwarding table, where for every Ping or Query message forwarded, a table entry is stored. The table entry uses the message ID as the key, and the servent connection from which the message arrives as the value. When the servent receives a Pong or QueryHit, it looks up its message ID in the forwarding table. If the servent has seen the corresponding Ping or Query message, it will find the message ID in the forwarding table (a Ping or Query and its corresponding Pong or QueryHit have the same message ID respectively). In this case, the servent will forward the Pong or QueryHit to the servent connection stored in the forwarding table. Otherwise, the Pong or QueryHit is not supposed to traverse this path, and is hence removed from the network.

Content Search

We keep a centralized list of all the file names used in the simulation framework in an ASCII file called *shared_resources.txt*. During the initialization phase, all the servents participating in the network are assigned a set of files from this centralized list. To improve the run-time performance of file search operation, we use a HashSet [41], which is a Java class that implements a kind of a set, backed by a hash table. It offers constant time performance for basic operations like adding/removing elements and testing for existence.

For each servent, we compute a HashSet of the file names possessed by it during the initialization phase. When a new Query is generated during simulation, the central manager (which simulates the GWebCache [35] functionality) computes a HashSet of file names contained in *shared_resources.txt* that match the Query. When the Query arrives at any servent, the HashSet of the Query is intersected with the HashSet of the servent. If the servent possesses any files satisfying the Query, this information is passed into the Result Set of the QueryHit message. The result HashSet of each Query is cached at the central manager. Hence, when a new Query is generated with the same search criteria, the resulting HashSet can be reused, thus speeding up the processing chain of the new Query message.

Query Routing Protocol (QRP)

QRP governs how an ultrapeer filters incoming Querys, and forwards them to only those leaf nodes, which are likely to match the Querys. The leaf nodes send a Query Routing Table (QRT) to the ultrapeer, and the ultrapeer makes its decisions by looking up these routing tables. It is important to note that the aim of QRP is to avoid forwarding Querys that cannot match, rather than to forward only those Querys that will match. The protocol operates at two levels: at the leaf node, and at the ultrapeer.

At the leaf node: The following steps are undertaken at the leaf node.

1. We break the resource names into individual words. A word is a consecutive sequence of letters and digits.

2. We hash each word with a hash function as described in [62] and insert a “present” flag in the corresponding hash table slot. The hash table is a big array of bits, and we do not store the key, but only the fact that the key ended up filling some slot. Before hashing, we convert all words

to lower-case, and remove all accents. Besides, we remove all words less than three characters in length.

3. We then remove the trailing one, two and three characters of each word, and re-hash the 3 new words formed in this way, provided their length is greater than 2 characters. This is done with the aim of removing plurals and word-endings like 'ing', 'ed', etc. from words. As an example, consider the file name "Bhajo Gopala.avi". This will give rise to the following hash table entries: bhajo, bhaj, bha, gopala, gopal, gopa, gop, avi.

4. When all the resources of a leaf are hashed, the complete hash table forms the QRT of the leaf. The QRT is optionally compressed, broken into smaller messages, and sent with the normal Gnutella traffic to the ultrapeer, in the form of Route Table Update messages.

The hash table we currently use is 1024 bits in size. This size was found to be sufficient for our current experiments, but can be easily increased on demand. All the leaf nodes thus build their routing tables and send them to their ultrapeers. If the file contents of a leaf node change, the routing table updates are sent to the ultrapeer in the form of Route Table Update messages.

At the ultrapeer: The ultrapeer stores the QRTs of each of its leaf nodes. On receiving any `Query`, the ultrapeer breaks the search string into individual words, and makes a hash table lookup for those individual words in the QRT of each of its leaf nodes. On finding a match, the ultrapeer forwards the `Query` to that particular leaf node.

Flow Control

Flow Control is a mechanism used to regulate the amount of data that passes through a peering connection. The overall scheme has been implemented at the application layer as follows. There are four input queues for each servent connection, corresponding to each Gnutella message type. All incoming messages are queued in their respective queues. Each servent has been assigned a pre-decided output bandwidth of 10 kB/second per peering connection for sending messages. The message queues are processed in FIFO order, prioritized (from most to least) as: `QueryHit`, `Pong`, `Query`, `Ping`. In other words, the `QueryHit` queue is processed first, in FIFO order. All its messages are forwarded one-by-one. Next, the `Pong` queue is taken up, and so on, until all the queues are empty or the output bandwidth of 10 kB/s is fully used up.

To limit excessive data, if the total amount of data in all input queues per connection exceeds 10 kB, all `Query`s which are not originating at the servent itself are dropped. This is done to avoid queuing back potentially large results for these `Query`s when we are already facing a throughput problem.

The HTTP file transfer, which actually takes place outside the Gnutella protocol, is also flow-controlled. It is guaranteed a minimum data flow rate of 10 kB/s per connection irrespective of the number of queued Gnutella messages, while the maximum allowed rate is the available bandwidth. This is done to ensure a minimum bandwidth for the actual data exchange (which is the main purpose of any P2P file-sharing system) even at peak network loads.

User behaviour characteristics

A persistent feature of most P2P systems is *churn*. A peer joins a P2P network when the user starts the application, searches and shares content, and leaves the network when the user closes the application. Such a join-participate-leave cycle is called a *session*. The phenomenon of independent arrival and departure of hundreds of thousands of peers is called churn. As churn can significantly affect various overlay as well as underlay characteristics like scalability, availability, etc., modeling churn appropriately is imperative to P2P simulation studies. Hence, we add support for peers going online and offline in SSFNet. A peer can be set online, i.e., able to participate in the Gnutella network, or offline, with the online/offline session lengths being defined by mathematical distributions.

Another feature of P2P systems is content availability. Most P2P systems are characterized by a large number of peers sharing little or no content at all (commonly termed *free-riders*), while some peers share 80 – 90% of total content available [131]. To be able to reflect content availability realistically, we make the number and type of content files shared by each peer determinable by a mathematical distribution. As mentioned earlier, there is a centralized ASCII file *shared_resources.txt*, where the name, type and size of each file is stored. When a peer goes online, it is allocated resources from this file using a mathematical distribution, which can be tuned at the start of the simulations. In this way, user behaviour characteristics can be determined by defining the appropriate mathematical distributions.

Other features

Support for leaf/ultrapeer nodes and Pong Caching is provided. As the HTTP protocol is already implemented in SSFNet, it is possible to adapt this code for content exchange between nodes. In order to achieve fine-grained control of the simulations, and to be able to calibrate the effects of various factors on overlay and underlay metrics, we make the P2P code highly configurable. The following are some of the parameters that we can easily tune:

- number of peers an ultrapeer/leaf can connect to
- ratio of leafs to ultrapeers at the start of the simulation
- rate of generation of `Ping` or `Pong` messages
- input queue size of messages at each servent and P2P traffic flow control
- online time after which a leaf may become ultrapeer
- number, rate and content of query strings generated by each peer
- number, type and size of content files shared by each servent
- online and offline durations for each servent

The network topology to be simulated is specified with the help of a DML file. SSFNet automatically assigns IP addresses to all host and router interfaces specified in the DML network model. The IP addresses are aggregated in blocks according to the CIDR (Classless Interdomain Routing) recommendations. An explanation of automatic assignment of IP addresses to DML network models by SSFNet is found at [104].

Each simulation run generates a log file, which logs information at different granularities. Not only do we log all the exchanged Gnutella messages, but we also have the ability to log sent, received, error and memory usage messages. This helps to debug as well as to analyze the impact of various events on overlay and underlay performance.

2.8.2 Experiences with SSFNet

We report on the memory consumption of our simulations, as well as scalability and other issues like log file size, etc. To run P2P simulations, we use a Sun Fire X880 machine, with 8 UltraSPARC III Cu processors, and 32 GB RAM. We are able to simulate complex AS topologies, with routers, links, bridges, hosts running P2P software, as well as link and device delays and link bandwidths. We subsample Internet AS topologies as derived from recent measurements [68], and distribute P2P clients within the ASes according to geographical populations or ISP customer information.

As we keep increasing the number of Gnutella servents in the simulations, we realize that the scalability limitations of simulations occur at the underlay network. We realize that given the overhead of P2P protocol computation, using more than 100 routers at the underlay topology leads to runtime degradation because of the computation of the entire message transmission at the TCP level. As routers are running OSPF and BGP protocols and are simulating link delays, we have to restrict ourselves to around 100 routers. As each AS needs at least two routers, we are hence limited to an underlay topology of 50 ASes. Nevertheless, to simulate complex intra-AS topologies with a reasonable number of peers per AS, with representative last-hop bandwidths and link delays, we settle for 25 ASes.

Running multiple simulations for 100, 300, 500, 1000, and 1250 Gnutella servents reveals that P2P characteristics are not affected by the number of peers. However, using more than 1250 peers results in poor run-times. Hence, we concentrate on simulations with $700 - 1000$ Gnutella peers in $16 - 25$ ASes. We examine the memory consumption of the simulations for a $10,000$ seconds simulation run, averaged over 10 runs. The simulations consume 3.1 GB RAM at the start, and end with a consumption of 5.1 GB, increasing linearly. One needs at least 4 GB RAM to start the simulations. The simulations run in real time, i.e., a $10,000$ seconds simulation run completes in about $10,000$ seconds of real time.

The size of the log file ranges with the granularity of logging. With full scale logging of handshake, connection negotiation, `Ping`, `Pong`, `Query`, `QueryHit`, and HTTP file exchange messages, the log file reaches 5.6 GB in size. Using `gzip` compresses the size to 1.4 GB. However, we realized that the bulk of the messages are `Ping`/`Pong`s. When we disable the logging of `Ping`/`Pong` messages (even though SSFNet simulates their transmission in full), the log file reduces in size by a factor of 4. While we loose some of the swarming pattern of messages in the network, we are still able to analyze the more relevant characteristics like peer connectivity, query search and file download patterns of P2P systems using such logs.

We have explained how we have implemented a multi-layered, highly structured, heavily configurable simulation framework for the Gnutella P2P system. The emulation of the underlay topology along with routers, links, hosts, delays, bandwidths, TCP/IP, OSPF and BGP protocols enables us to study the interaction of overlay and underlay routing and the impact of events in one layer on the other layer. We will use this simulation framework in Chapter 6 to study the impact of using the oracle to choose P2P neighbours on P2P routing performance, scalability, overlay graph properties, as well as end-user experience metrics like content download times, content locality, and query search results.

3 Measurements of Overlay-Underlay Correlation

Network applications, such as IRC, MBone, Usenet, etc., route data at the application layer, thus creating overlay networks. A common aspect of these applications is that these overlay networks are controlled by system administrators, who are likely to ensure that neighbourhood choices respect resource limitations to some degree. One can expect that this biases the neighbourhood choices in these applications to respect network proximity. This is in contrast to another class of overlay networks, the popular P2P file-sharing systems. Here system-specific metrics or arbitrary choices govern the neighbourhood selection process. In this chapter we ask the question - how much does the neighbourhood selection process of a typical file-sharing P2P protocol respect the underlying Internet topology, or put differently, how close is a P2P topology to the Internet topology. The answer can help us estimate the (in-)efficiency of using overlays. We investigate this question using Gnutella as our example overlay network.

In Section 3.1, we use a combination of active and passive measurement techniques to crawl the Gnutella network and correlate the overlay peerings to the underlying Internet AS topology. In Section 3.2, we present a visualization-driven analysis technique for evaluating the overlay architecture with respect to the underlay. Our analysis confirms that the Gnutella topology is not correlated with the Internet AS topology, i.e., Gnutella nodes do not bias their neighbour selection process to respect network proximity. However, the Gnutella overlay topology differs in many respects from a randomly generated network.

3.1 Gnutella Measurement Study

Fully distributed P2P networks, such as Gnutella [35], attracted an enormous interest after Napster, which relied on a central server, was shut down in 2001. Traditionally, networks such as Gnutella are mapped by crawlers [90, 92]. The main component of a crawler is a client which maintains a list of known Gnutella servents. It connects to each servent on this list and uses the `Ping`/`Pong` protocol with large TTLs to discover other Gnutella servents and edges[1] in the Gnutella network. The discovered servents are added to the list of known servents, which are further contacted iteratively to expand the network search. This ultimately results in a snapshot of the network. The crawls, typically lasting a few hours, discovered about $120,000$ [61], $400,000$ [90], and $1,239,487$ [92] servents in the Gnutella network respectively. Overall, the authors of [90] assert that Gnutella's virtual network topology does not match the Internet topology well. Studies like [92] found considerable heterogeneity in Gnutella, and presented evidence of distinct client- or server-like behaviour in servents.

Changes to the Gnutella protocol like Pong Caching, hierarchical topology, termination of connections on frequent pinging, and dynamic querying have vastly improved the scalability of the

[1] A direct peering between two Gnutella servents is referred to as an edge

Gnutella network [61]. Yet at the same time they pose a huge impediment to investigating the structure of Gnutella through simple crawling as used in the previous studies [90, 92], which is based on the original semantic of the Ping/Pong protocol. Thus we first investigate how to overcome these limitations to find neighbours in the Gnutella network, and then explore their network distance in comparison to random node distances.

3.1.1 Methodology for Identifying Edges in a P2P Network

In order to study how close the P2P topology is to the Internet topology we first need to identify a representative set of edges in the P2P network. Then we need to find a comparable set of edges in the Internet and a metric suitable for comparison.

The most obvious way of finding edges in a P2P network is to create some by participating. Yet these are not representative as they are highly biased by the location and the software of the participant. Rather we want to identify edges in the P2P network where neither of the two nodes is controlled by us. We refer to any two nodes connected by an edge as neighbour servents and those not involving a node controlled by us as remote neighbour servents.

Due to the changed semantics of the Ping/Pong protocol [79] the simple crawling approach used in the previous studies is no longer sufficient. As Pongs are cached and due to rapid fluctuations in the Gnutella network[2] one cannot assume that answers to Pings with TTL equal to two (so called crawler pings) contain still active servents. They should, however, have been remote neighbour servents at some point. Note that leaf nodes are no longer reported in Pongs.

To cope with these complications we deploy a combination of active and passive techniques to explore the Gnutella network. Our *passive approach* consists of an ultrapeer based on the GTK-Gnutella [38] software. The goal is to have an ultrapeer that behaves like a normal node in the network, yet worthwhile to connect to. It shares 100 randomly generated music files, totalling 300 MB in size, and maintains 60 simultaneous connections to other servents. To derive various statistics the servent is instrumented to log per-connection information augmented with a packet-level trace. In this way, the passive approach gives us a list of active servents in the Gnutella network.

Our *active approach* consists of multiple Gnutella servents and a manager. The manager controls the servents and supplies each servent with a Gnutella servent address (IP address/port number combination) to connect to, which it obtains from the passive approach. Each servent tries to connect to its assigned servent. Depending on success, connection refusal, connection timeout, or Gnutella error message, the client reports a different result to the manager. Based on this the manager reschedules the servent for retry. If the connection is rejected with a Gnutella error code it is indicative of an active servent that most likely has no open connection slots currently available. If the connection times out, the servent is either inactive or behind a firewall. If the connection is refused, it is either inactive or highly overloaded with connection requests. Accordingly servents that rejected connections are retried faster than those that refused them or did not respond.

The multiple-servent crawler uses Pings with TTL 2 to obtain a list of candidate servents. Since Query results are difficult to cache, we use Querys with TTL value of 2 to obtain a set of remote neighbour servents. This is in contrast with previous crawling approaches which relied on Pings to map the network, and allows us to get around the challenge of network crawling due to Pong Caching. These remote neighbour servents are then contacted actively to further advance the network exploration. This approach allows us to discover edges in the Gnutella network that existed

[2] In our experiments, see Section 3.1.2, the median incoming session duration is 0.98 seconds.

at a very recent point of time. While it is theoretically possible to enhance our strategy to discover "currently" active remote neighbours by connecting to both the servents at the same time and issuing a query with a TTL of 3 (which can only be answered by the crawler servent), the problem with this approach is that connecting to two servents at the same time is problematic due to the restrictions on the neighbourhood size of each servent.

Our *combined active/passive approach* integrates the multiple-servent crawler into the passive ultrapeer. When interacting with other servents, the multiple-servent crawler pretends to be a long-running ultrapeer with an acceptable querying scheme. It processes incoming messages and has a non-intrusive `Ping`/`Pong` behaviour. For example the client issues queries and crawler pings only to those peers that have already responded with a `Pong`, `Ping`s are issued only to those peers that send one themselves, and at the same rate. Such a pragmatic behaviour helps to avoid bans. The client uses `Query` messages with a compiled list of broad catchwords such as *mp3, avi, rar*, which are likely to result in many hits. One can expect the queries to yield only a subset of the neighbours due to the presence of "free-riders" (peers that do not contribute resources to the P2P network) [3].

Early experiments showed that the behaviour of a client can have significant impact on the connection success rate. This has led to several changes that make our client more attractive, e.g., advertising a long online time, ultrapeer handshaking, etc. Also, the client behaviour was made less predictable, e.g., by initializing the timers that issue the `Query` and `Ping` messages with random values within a certain range.

To better understand the limitations of our approach and the behaviour of both client and ultrapeers, we experimented with the prevalent tools in a testbed. The testbed consists of a small Gnutella network with servents based on GTK-Gnutella [38], LimeWire [61], BearShare [14], and Gnucleus [33]. Interestingly only GTK-Gnutella provides a configuration parameter to elevate it to an ultrapeer. We also observed several compatibility issues. For example, while the LimeWire servent allows other servents to establish TCP connections to it, it later rejects the Gnutella handshake with an error message. BearShare also discourages other vendors' servents from connecting to it. We conclude that non-compliance and compatibility issues impose limitations on the success rate of crawling techniques.

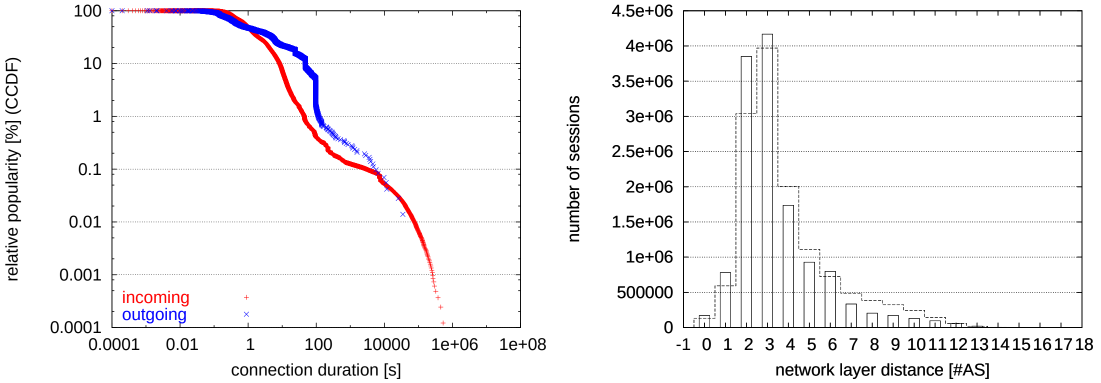
Experiments with the unmodified (passive approach) and the modified ultrapeer (combination of active and passive approaches) confirm that the changes did not alter the characteristics of incoming connections, thus reducing the likelihood of bias in network sampling. Overall this allows us to reach a connection rate well above other known studies (e.g., [24]) during the same time period.

3.1.2 Results

We use the active/passive approach to make a measurement study of the Gnutella network from October 26, 2003 to December 3, 2003. During this time our ultrapeer logs $8,199,643$ sessions of which $8,192,461$ are incoming and $7,182$ are outgoing. The dominance of the incoming connections indicates that our ultrapeer is quite popular, which is likely to reduce the bias in the sampled servents. The crawler discovers $14,101,399$ remote neighbour servents.

Before exploring similarities of the P2P topology with the Internet topology we explore the variability of the Gnutella session durations. Figure 3.1(a) shows the complementary cumulative distribution function (CCDF) of the session duration of the above trace. It is apparent from the plot that most session durations are very short. Indeed, the median duration of incoming and outgoing sessions is 0.98 and 0.74 seconds respectively. Only 5% of the incoming sessions last longer than 12.3 seconds. This implies that edges in the Gnutella network change rapidly. On the one hand this

Figure 3.1 (a) CCDF of session duration distribution in the Gnutella network (b) Comparison of the estimated number of ASes between Gnutella neighbours (solid bars) and random IP addresses (dashed line)



complicates any crawling attempts, on the other hand it affects the expected accuracy and value of any derived map.

Typical metrics for network distance in the Internet are router hop counts and AS distances. Unfortunately, estimating the router hop counts for any two random nodes is non-trivial [103]. While difficult, estimating approximate AS distances is possible. We map IP addresses to their parent ASes using BGP tables from RIPE [89] during the week of October 26, 2003. Using BGP tables and updates we derive an AS topology and the AS relationships [30]. Based on this topology and the heuristic that a customer route is preferred to a peering route over an upstream, we estimate the AS distances.

Figure 3.1(b) (solid bars) shows a histogram of the estimated AS distances of the remote neighbour servents, i.e., Gnutella servents at application layer distance of 1 and 2 from our crawler. We were unable to estimate the distance for $648,059$ sessions and assigned them distance 0. We note that the estimated AS distances for the direct neighbours have a significantly different distribution.

The plot shows that the AS distances span a huge range with some clustering at distance $3-5$. The large values of AS distance as well as their broad range indicates that Gnutella does not bias its neighbour choices to correspond to network proximity. Most of the Gnutella peerings leave the AS boundaries, and indeed, cross multiple AS hops.

To compare the Gnutella network with a random IP network, we generate random peerings by picking end-points at the IP level by randomly choosing two valid IP addresses from the whole IP space. These random IP addresses are then mapped to ASes and the AS distance between them is calculated in the same manner as the Gnutella edges. Figure 3.1(b) (dashed line) shows a histogram of the estimated AS distances of randomly chosen IP addresses. We observe that while the overall shape of AS distances for random and Gnutella peerings is quite similar, there are some differences. This is not surprising as users of P2P file sharing networks need reasonable network connectivity (e.g., broadband) to be able to use the network. Hence, the slight difference between random and Gnutella peerings is to be expected.

3.1.3 Summary

Exploring the Gnutella network topology is limited by the optimizations to the Gnutella protocol as well as the short session durations. Nevertheless we are able to identify a significant number of remote neighbour servents to approximate a representative set of edges in the P2P network. The comparison of Gnutella edges to randomly selected pairs of IP addresses shows that Gnutella peers do not seem to significantly bias their neighbourhood choices towards network proximity. A large number of P2P connections leave the AS boundaries and cross multiple AS hops.

3.2 Using Visualizations to Analyze Overlay-Underlay Correlation

In the previous section, we found that the overlay topology of the Gnutella network is not correlated with the underlying Internet topology. We also found that while the Gnutella overlay topology is similar to a randomly generated network, there are some differences between the two networks. To better understand the similarities as well as the differences between the Gnutella overlay and a random network, we model the overlay-underlay correlations using a unique visualization-driven analysis technique [13]. This technique relies on the concept of cores [12, 97] to analyze the overlay in the context of the underlay. We introduce the visualization technique in Section 3.2.1, and apply this technique to study the correlation of the Gnutella overlay with the Internet AS network, as well as to compare the overlay with a random network in Section 3.2.2.

3.2.1 Analytic Visualization

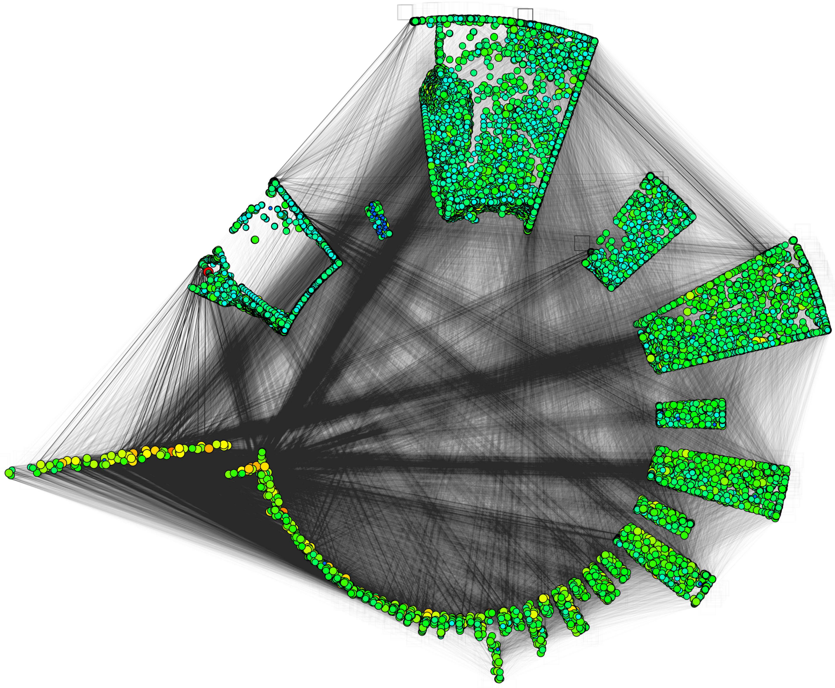
In this section, we describe two visualization techniques that help in the identification of key features in an overlay. Both highlight a given hierarchical decomposition of the network while displaying all nodes and edges. They have been successfully applied to the network of Autonomous Systems (AS), which is an abstraction of the physical Internet, yet are highly flexible and can be easily adjusted to other networks.

We use the concept of cores [12, 97] for the required hierarchical decomposition of the network. Briefly, the k-core of an undirected graph is defined as the unique subgraph obtained by recursively removing all nodes of degree less than k. A node has coreness ℓ, if it belongs to the ℓ-core but not to the (ℓ+1)-core. The ℓ-shell is the collection of all nodes having coreness ℓ. The core of a graph is the non-empty k-core such that the $(k+1)$-core is empty. Generally the core decomposition of a graph results in disconnected sub-graphs, but in the case of the AS network we observe that all k-cores stay connected, which is a good feature regarding network connectivity. Cores have been frequently used for network analysis, e.g., [29, 32].

A visualization technique employing the concept of cores is proposed by Baur, et.al. [13]. Their algorithm lays out a graph incrementally starting from the innermost shell, iteratively adding the lower shells. Their implementation uses core decomposition and a combination of spectral and force-directed layout techniques. A successful application of this visualization technique compares actual AS graphs with generated AS graphs. The obtained layouts clearly reveal structural differences between the networks. The nature of this layout technique is popularly referred to as a *network fingerprint*. Such pseudo-abstract visualizations offer great informative potential by setting analytic characteristics of a network in the context of its structure, revealing numerous traits at a glance.

Another fingerprint drawing technique, that improves upon the above technique and focuses on the connectivity properties of a network decomposition has been presented in [37]. This approach,

Figure 3.2 An example visualization of the core decomposition (segments) of the AS network using LunarVis. Each node represents an AS with size and color reflecting the size of its IP-space. Angular and radial extent of a segment reflect the number of nodes and intra-shell edges respectively. Note the extremely large AS (upper left red node) in the minimum shell.

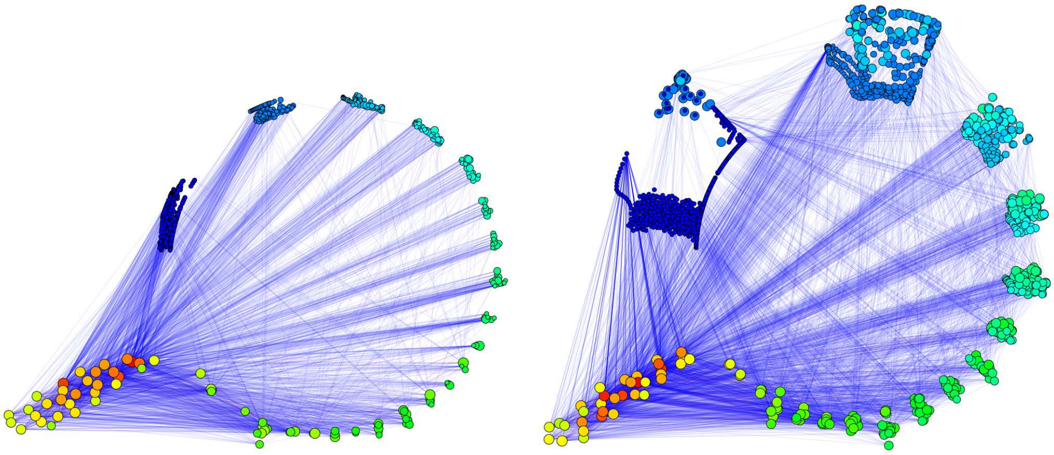


termed *LunarVis*, lays out each set of a decomposition – which are the shells in our case – individually inside the segments of an annulus. The rough layout of LunarVis is defined by analytic properties of the decomposition, allowing the graph structure to determine the details. By virtue of a sophisticated application of force-directed node placement, individual nodes inside annular segments reflect global and local characteristics of adjacency, while the inside of the annulus offers space for the exhibition of the edge distribution. Combined with well-perceivable attributes, such as the size and the color of a node, these layouts offer remarkable readability of the decompositional connectivity and are capable of revealing subtle structural characteristics, see Figure 3.2.

3.2.2 Overlay-Underlay Correlation in the Gnutella network

Using the measurement setup introduced in Section 3.1, we sample the Gnutella network again for one week starting April 14, 2005. The ultrapeer logs 352,396 sessions and the crawler discovers 234,984 remote neighbour servents. For each edge of the Gnutella network we map the IP addresses of the Gnutella peers to their parent ASes using the BGP table dumps offered by Routeviews [91] during the week of April 14, 2005. This results in 2964 unique AS edges involving 754 ASes, after duplicate elimination and ignoring P2P edges inside an AS. For the random graph we pick endpoints at the IP level by randomly choosing two valid IP addresses from the whole IP space. These edges are then mapped to ASes in the same manner as for the Gnutella edges. This results in 4975 unique edges involving 2095 ASes for the random network at the AS graph level. The different sizes

Figure 3.3 Visualization of the core decomposition of the overlay communication networks. Core-shells are drawn into annular segments, with the 1-shell at the upper left. Angular and radial extent of a segment reflect the number of nodes and intra-shell edges respectively. Inside each shell nodes are drawn towards their adjacencies. Colours represent the degree of a node while the size represents their betweenness centrality. Edges are drawn with 10% opacity and range from blue (small weight) to red (large weight).



(a) P2P network

(b) Random network

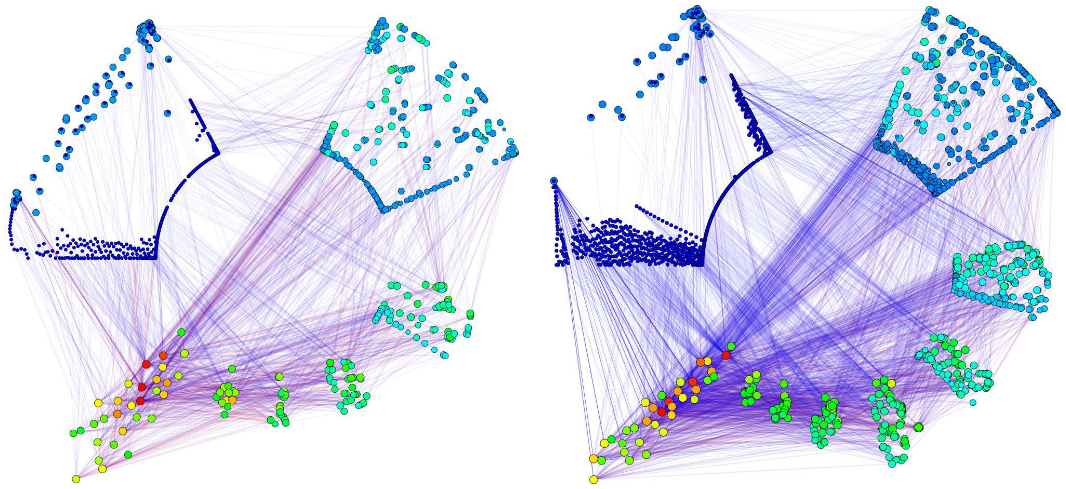
of the graphs are a result of the generation process: we generate the same number of IP pairs for random network as observed in Gnutella, and apply the same mapping technique to both data sets, which abstracts the graph of IPs and direct communication edges to a graph with ASes as nodes and the likely underlay communication path as edges. This way, the characteristics of Gnutella are better reflected than by directly generating a random AS network of the same size as the Gnutella network.

For our analysis, we apply the model and methodology from Section 2.2 as follows. The overlay $\mathscr{O} = (G, G', \phi, \pi)$ uses the direct communication in Gnutella as graph G, while the graph G' corresponds to the hosting Internet, in our case at the AS level. The mapping ϕ corresponds to the IP-to-AS mapping, while π denotes routing in the AS network. Apart from the already introduced induced underlay, we also investigate the network of direct overlay communication, yet abstracted to the level of ASes in order to be comparable to the induced underlay. Note that in a simplified model, where each communication causes uniform costs, the appearance weight in the induced underlay (ω'') corresponds to the total load caused by the overlay routing in the underlay network. As exact traffic measurements on each underlay link are non-trivial, this can be interpreted as an estimate of the actual load on underlay links due to the overlay traffic.

Figure 3.3 shows visualizations of the direct overlay communication of both the Gnutella network and the random network. Employing the LunarVis [37] technique, these visualizations focus on the decompositional properties of the core hierarchy. We point out that max-shells correspond to top-tier ASes, while lower shells denote customer and small ASes.

Numerous observations can be made by comparing the two visualizations. Notice the striking lack of intra-shell edges for all but the maximum shell in the Gnutella network (small radial extent). This is also true for edges between shells, as almost all edges are incident to the maximum shell. This means that almost always at least one communication partner is in the maximum shell, a strongly

Figure 3.4 Visualization of the core decomposition of the induced underlay communication network. These drawing use the same parameters as Figure 3.3.



(a) P2P network (b) Random network

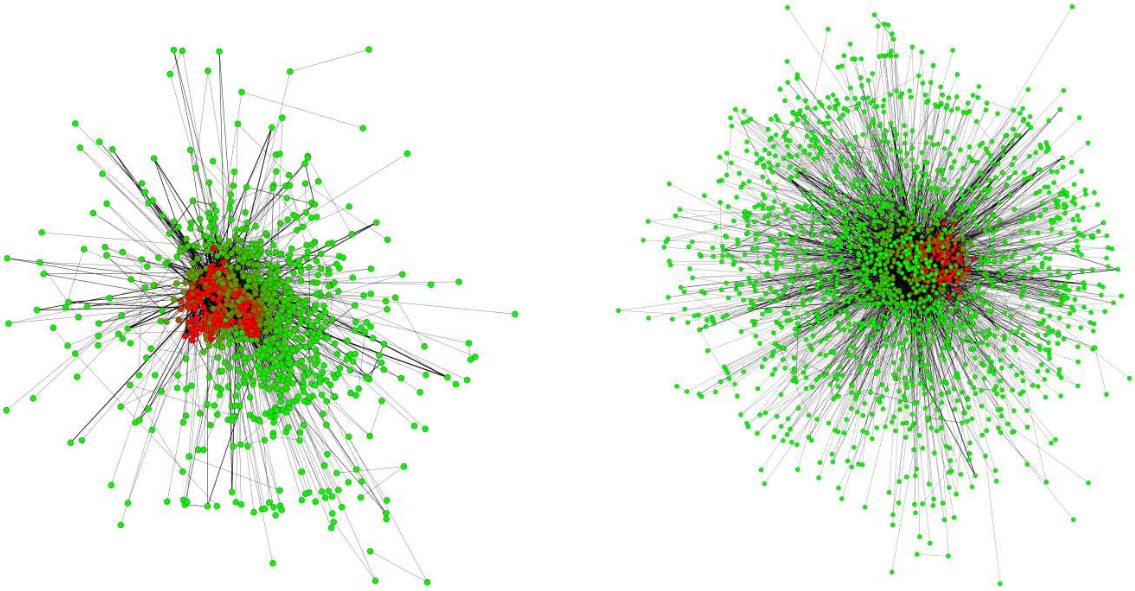
hierarchical pattern that the random network does not exhibit to this degree. Note furthermore that in Gnutella, the betweenness centrality (size of a node) correlates well with coreness, a consequence of the strong and deep core hierarchy, whereas in the random network the two- and even the one-shell already contain nodes with high centrality, indicating that many peerings heavily rely on low-shell ASes. The depth of the Gnutella hierarchy (26 levels) strongly suggests a strongly connected network kernel of ultrapeers, which are of prime importance to the connectivity of the whole P2P network. However, the distribution of degrees (node colors) does not exhibit any unusual traits and no heavy edges are incident to low-shell ASes in either network.

Figure 3.4 visualizes the induced underlay communication of both the Gnutella network and the random network, employing the same technique and parameters as in Figure 3.3. The drawings immediately indicate the much smaller number of ASes and overlay nodes in the Gnutella network. As a consequence, more heavy edges (red) exist and the variance in the appearance weight (edge color) is more pronounced. This is because of the fact that not all the ASes host P2P users (as shown by our measurements in Section 3.1), though this is the case for the random network. Again, the distributions of degrees do not differ significantly.

For a closer comparison, Figure 3.5 shows a top-down view of the visualizations of communication edges in Gnutella and random network. The visualization technique places nodes with dense neighbourhoods (tier-1 and tier-2 ASes) towards the center, and nodes with lesser degrees (tier-3 customer ASes) towards the periphery. We can observe that while both networks have many nodes with large degrees in the center, the random network possesses several nodes with large degree in the periphery. Gnutella, on the other hand, has almost no nodes with large degree in the periphery in both the overlay and the induced underlay models. Moreover, this pattern is more pronounced for Gnutella in the direct overlay communication model (Figure 3.3), while the random network is largely similar in both the models. In other words, it appears that Gnutella peering connections tend to lie in ASes in the core of the Internet where there may be more high-bandwidth links available.

To further confirm our observations, we investigate structural dependencies between the induced underlay communication model and the actual underlay network, by comparing the appearance

Figure 3.5 Comparison of occurring communication in the P2P network and the random network



(a) P2P network

(b) Random network

weight with node-structural properties of the corresponding end-nodes in the original underlay. The node properties degree and coreness have been successfully applied for the extraction of customer-provider relationships as well as visualization [111, 29] due to their ability to reflect the importance of ASes. Therefore we focus on degree and coreness of nodes for our analysis. We systematically compare the appearance weight of an edge with the minimum and the maximum of the degree and the coreness of its end-nodes. Figure 3.6 shows the corresponding plots.

From the plots of the minimum and maximum degree, it is apparent that the appearance weight of an edge and its end-nodes' degrees are not correlated in both the Gnutella and the random network, as no pattern is observable. Also, the distributions are similar as the majority of edges are located in the periphery of the network where the maximum degree of the end-nodes is small. We thus hypothesize that the relation of load in the P2P network and the node degree in the underlying network is the same in both the Gnutella and the random network. In other words, the Gnutella network does not appear to be significantly affected by the node degree of the underlay nodes.

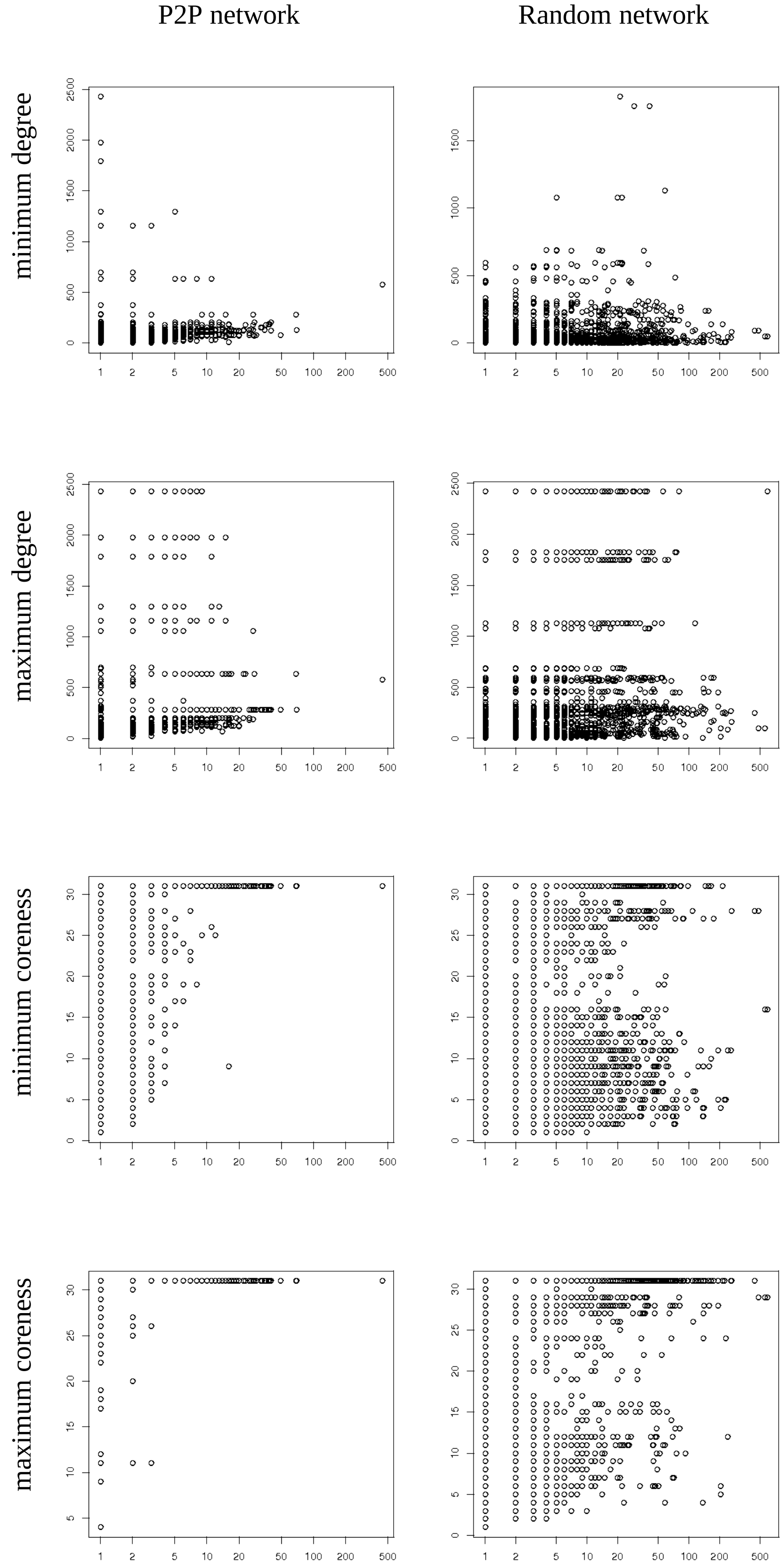
However, when we consider the coreness, interesting observations are revealed. From the graphs of minimum and maximum coreness in Figure 3.6, we can observe that although there is no correlation in either of the two networks, their distributions are different. In the random network the distributions are very uniform, which is a reflection of its random nature. But in the case of Gnutella almost no heavy edge is incident to a node with small coreness, as can be seen in the minimum-coreness diagram. Positively speaking, most edges with large appearance weights are incident to nodes with large minimum coreness. Interpreting coreness as importance of an AS, these Gnutella edges are located in the backbone of the Internet, an important observation. The same diagram for the random network does not yield a similar significant distribution, thus denying a comparable interpretation. For instance, in the random network, there exist edges located in the periphery that are heavily loaded. As an aside, backbone edges need not necessarily be heavily loaded in either network.

All these observations and analysis show that the Gnutella network differs from random networks and there appears to be a slight correlation of Gnutella topology with the Internet underlay, in that the ultrapeers tend to lie in core Internet ASes (typically top-tier ASes) where there is higher prevalence of high-bandwidth connections.

3.3 Conclusion

In this chapter, we perform a measurement study of the Gnutella P2P network. We find that the Gnutella session lengths are very short, and the Gnutella peers do not bias their neighbourhood selection to respect network proximity. As a result, a large number of overlay peerings leave the AS boundaries and often cross multiple AS hops. Using a visualization-driven technique, we transform the overlay graph to a corresponding induced subgraph in the underlay. This is used to compare the overlay graph of the Gnutella network with a randomly generated graph, and to identify several key features of the overlay. We confirm that while the Gnutella topology is not closely correlated with the Internet AS topology, it differs from randomly generated graphs. This is evident from the analysis of the core decomposition of the overlay communications, as well as the comparison of the appearance weights of edges with the coreness of the nodes in the induced underlay. The differences arise because many of the Gnutella peerings lie in top-tier ASes where there is a higher prevalence of high-bandwidth connections.

Figure 3.6 Comparing appearance weight with the minimum and the maximum of the degree and the coreness of the corresponding end-nodes in the Gnutella and the random network. Each data point represents an edge, the *x*-axis denotes the appearance weight and the *y*-axis reflects the degree/coreness of the end-nodes. All axes use logarithmic scales.



4 Proposal: ISP-P2P Cooperation

In this chapter, we introduce the principal proposal of this thesis, which enables ISPs and P2P systems to co-operate with each other in such a way that both of them benefit. We discuss the problem space and the need for a solution in Section 4.1, and propose the oracle solution in Section 4.2. The advantage of using last-hop bandwidth as a metric for the oracle is discussed in Section 4.3, and a pseudo-code for the oracle service is outlined in Section 4.4. With the help of an example, we show how the oracle service can be used by P2P users in Section 4.5, and make some observations on realizing the service in Section 4.6.

4.1 Motivation

We have already discussed that P2P systems are so popular that they contribute more than 50% to the overall network traffic in the Internet [50, 86, 106, 46, 102]. The wide-spread use of such P2P systems has put the ISPs in a dilemma! On the one hand, P2P 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 [66]. On the other hand, ISPs find that P2P traffic poses a significant traffic engineering challenge [52, 86]. P2P traffic often starves other applications like Web traffic of bandwidth [100], 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.

To construct an overlay topology, unstructured P2P networks usually employ an arbitrary neighbour selection procedure [106]. 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 in Chapter 3 as well as in [51, 90, 125] that P2P traffic often crosses network boundaries multiple times. 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 [5]. Besides, studies have shown that the desired content is often available “in the proximity” of interested users [51, 84]. 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.

Let us once again approach the problem of the overlay-underlay routing clash, by considering routing 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) [40].

P2P systems, on the other hand, setup an overlay topology and implement their own routing [6] 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 [35] or random walks while structured P2P networks often use DHT-based routing systems to locate data [106]. Answers can either be sent directly using the underlay routing [106] or through the overlay network by retracing the query path [35].

Overlay-based approaches serve as a means to circumvent path failures and network congestion. An overlay network forms a virtual network on top of a physical network by deploying a set of overlay nodes above the existing IP routing infrastructure. Overlay nodes cooperate with each other to route packets on behalf of any pair of communicating nodes, forming an overlay network. By routing through the overlay of P2P nodes, P2P systems hope to use paths with better performance than those available via the Internet [6, 94].

But the benefits of redirecting traffic on an alternate 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 [52, 96]. While a routing underlay, e.g., as proposed in [69], can reduce the work duplications of the P2P system, it cannot by itself overcome the interaction problems between the overlay and the underlay. 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. Put together, we identify the following drawbacks:

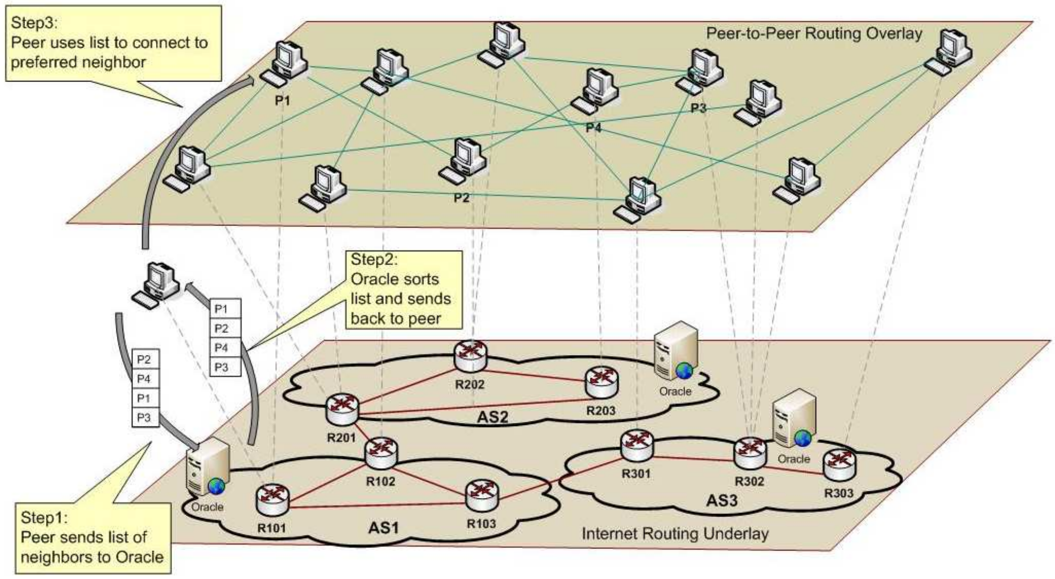
- The ISP has limited ability to manage its traffic and therefore incurs potentially increased costs for its inter-domain traffic, as well as for its inability to do traffic engineering on its internal network.
- The P2P system has limited ability to pick an optimal overlay topology and therefore provide 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.
- Different P2P systems have to measure the path performance independently.

While we do not know of a P2P network that tries to reverse-engineer the Internet topology, there are some proposals that suggest that P2P networks should bias their overlay topology by choosing neighbours that are close in the sense of high throughput or low latency, e.g., [85, 98, 4, 110] or that are within the same AS, e.g., [51, 16]. Others such as the Brocade [130] system propose to build an overlay on top of a structured DHT-based P2P system that exploits knowledge of the underlying network characteristics. Yet another system [100] proposes to use content caching to relieve the tension between ISPs and P2P systems. A recent proposal [126] uses iTrackers as portals of network providers to enable ISP-P2P collaboration.

4.2 The Oracle Service

We, in this thesis, propose and evaluate the feasibility of a simpler solution where ISPs collaborate with P2P systems by offering an **oracle service**. Let us consider how unstructured P2P networks tend to maintain their topologies. New P2P nodes usually retrieve a list of members of the P2P

Figure 4.1 Overview of the ISP-P2P collaboration process


Step3:
Peer uses list to connect to preferred neighbor
Peer-to-Peer Routing Overlay
P1
P3
P4
P2
Step2:
Oracle sorts list and sends back to peer
P1
P2
P4
P3
P2
P4
P1
P3
R202
R203
Oracle
AS2
R201
R102
R301
R302
Oracle
R303
AS3
Oracle
R101
AS1
R103
Internet Routing Underlay
Step1:
Peer sends list of neighbors to Oracle

network either via a well known Web page, a configuration file, or some history mechanism [101, 106]. They then pick some subset of these as possible neighbours either randomly [35] or based on some degree of performance measurement [85]. If the chosen neighbour cannot serve the new node it might redirect the new node by supplying an alternate list of P2P members.

Instead of the P2P node choosing neighbours independently, we propose that the ISP can offer a service, which we call the *oracle*, that ranks the potential neighbours according to certain metrics. This ranking can be seen as the ISP expressing preference for certain P2P neighbours. Possible coarse-grained distance metrics are:

- inside/outside of the AS
- number of AS hops according to the BGP [40] path
- distance to the edge of the AS according to the IGP [40] metric

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

- connection information such as: last-hop bandwidth
- topological and geographical information such as: same point of presence (PoP), same city
- performance information such as: expected delay, available bandwidth
- link congestion (traffic engineering)

This ranking can then be used by the P2P node to select a close-by neighbour although there is no obligation. Figure 4.1 summarizes the operation of the oracle.

The oracle acts like an abstract routing underlay to the overlay network to achieve cross-layer optimization. But as it is a service offered by the ISP, it has direct access to the relevant information and does not have to infer or measure it. For example, an ISP knows whether a customer has a DSL broadband or a modem connection, its geographical location, expected link delay, etc.

The ISPs benefit in multiple ways by offering the oracle service:

- they can now influence the P2P routing decisions via the oracle and thus regain their ability to perform traffic engineering (control the traffic flow)
- by influencing the neighbourhood selection process of the P2P network, they can keep a significant portion of their network traffic localized within their internal network, and hence gain cost advantages by reducing costs for traffic that crosses their network boundary [128]
- the significant amount of measurement traffic that is caused by the P2P users attempting to reverse-engineer network distance (e.g., latency) of potential neighbours is omitted
- due to the ability to better manage their traffic flow, ISPs can provide better service to their customers and ensure fairness for other applications like Web traffic, etc., especially at times of peak demand.

The benefit to P2P nodes of all overlays is also multi-fold:

- they do not have to measure the path performance themselves
- they can take advantage of the knowledge of the ISP
- they can expect improved performance in the sense of low latency and high throughput as bottlenecks [5] can be avoided.

Even when the oracle uses a simple metric like AS distance to rank potential neighbours, this will lead to a large amount of P2P traffic staying within the ISP boundaries. ISPs will save traffic costs, while P2P users will experience lesser delays. That P2P networks benefit by increasing traffic locality has also been shown in [16] for the case of BitTorrent.

A critical issue that arises regarding the use of the oracle is **privacy**. An ISP will be anxious not to reveal its internal network topology. Also, a P2P user that has a high-bandwidth broadband connection may not be willing to answer too many connection requests from other users. Our answer to the privacy concerns is multi-fold. First, the oracle only ranks the list of possible neighbours, it does not provide details of the connectivity information of the potential neighbours. Hence, it does not reveal more information about its network than can anyhow be inferred by reverse-engineering the ISP network via measurements, as in [103, 21]. Second, the ISP does not need to reveal the exact details of the criteria used in sorting the list of neighbours. Finally, an ISP may alter the ranking criteria dynamically. For example, if 100 queries always include the same peer, the ISP may decide to rank this peer lower in the list, to balance its load. As the ability to control/manage its traffic is crucial to the operating costs of every ISP, the benefits accruing from ISP-P2P collaboration will outweigh the potential risks of providing an oracle, namely that the oracle exposes some information about the ISP topology and the network performance.

The oracle service is **application-independent**, as it is available to all overlay networks. One does neither need nor want to use a separate oracle for each P2P network. As an open service, it can be queried by any application and is not limited to P2P file-sharing systems. In fact, the oracle can be used by any application where the users have a choice of more than one destination to connect to. As a consequence, when a user queries the oracle, it does not necessarily imply the user's participation in file sharing systems. The oracle acts as a *peer mapping service*, which helps users of an application to select “good” neighbours. This also mitigates the legal concerns for an ISP, as the ISP is neither engaged in file-sharing activities by providing or caching any content, nor is the ISP providing a service that is solely designed to aid content distribution through P2P systems, irrespective of the nature of the content.

As this service is very generic, the purpose of an oracle query remains unknown to the ISP who operates the oracle or to anyone who has access to the oracle logs. This greatly reduces the usefulness of the logs for a possible legal action, thus protecting the privacy of the user and reducing the risk to the ISP to be implicated in such a legal action.

As an additional precaution to protect their identity, a P2P user could permute, e.g., the last byte of the IP addresses it is interested in or use an anonymization service for querying the oracle. However, such an action would also limit the ability of the ISP to rank the potential neighbours using multiple metrics.

4.3 Using Bandwidth to Select Neighbours

In the previous section, we discussed the benefits of localizing the P2P traffic within the network boundaries of the ISP. In this section, we discuss the benefits for the P2P users when the oracle service uses last-hop bandwidth of P2P users within its AS as a metric for sorting the list of possible neighbours. This will enable the ISP to help querying P2P users select high-performance neighbours. Using this metric is easily 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 [80, 93] for P2P users. We argue that the oracle using the bandwidth information to sort peers is a better alternative than the P2P users choosing their neighbours independently of the oracle, by using latency ping measurements [85] or geolocalization techniques [75].

Advantages over Latency

Using last-hop bandwidth of P2P users as a metric has advantages over neighbour selection using latency measurements [85], as network latency can change quickly [129]. Also, latency is difficult to predict reliably [56, 124], 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 [80] 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 [88] 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 [23], thus making neighbour selection based on last-hop bandwidths a good option.

Advantages over geolocalization techniques

We show in the later chapters that keeping the P2P traffic localized allows users to benefit from the significant geographic and interest-based clustering [25] for P2P content. One may argue in favour of bypassing the ISP's oracle service to utilize geolocalization techniques [75] to choose neighbours. However, we caution that finding a geographically-near neighbour does not necessarily imply that it is in the same ISP network as the querying node, as it is common nowadays to have multiple ISPs operating in the same geographical region. Thus, two overlay nodes having a small geographical distance between them can be separated by a large network distance. Moreover, even the best geolocalization techniques can identify a node to within 22 miles of its actual position [124], 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 neighbours in its network, even within the same city. We thus believe that ISP-aided P2P neighbour selection is a win-win solution for ISPs as well as P2P systems.

4.4 Algorithm for the Oracle Service

We propose the following algorithm that the oracle service can use for sorting the list of possible P2P neighbours of a querying P2P user.

```
Procedure:The Oracle Algorithm
Given a list of peers
  Identify the peers within its own ISP
    sort them according to last-hop bandwidth
    sort same bandwidth peers according to router-level (or PoP) topology
    factors like routing metrics, "available" bandwidth, service class can be considered
  For peers outside its ISP network
    sort peers according to AS-hop distance
    prefer customer or peer ISPs over provider ISPs
    backbone link bandwidths can be considered
Return sorted list of potential neighbours to the querying user
```

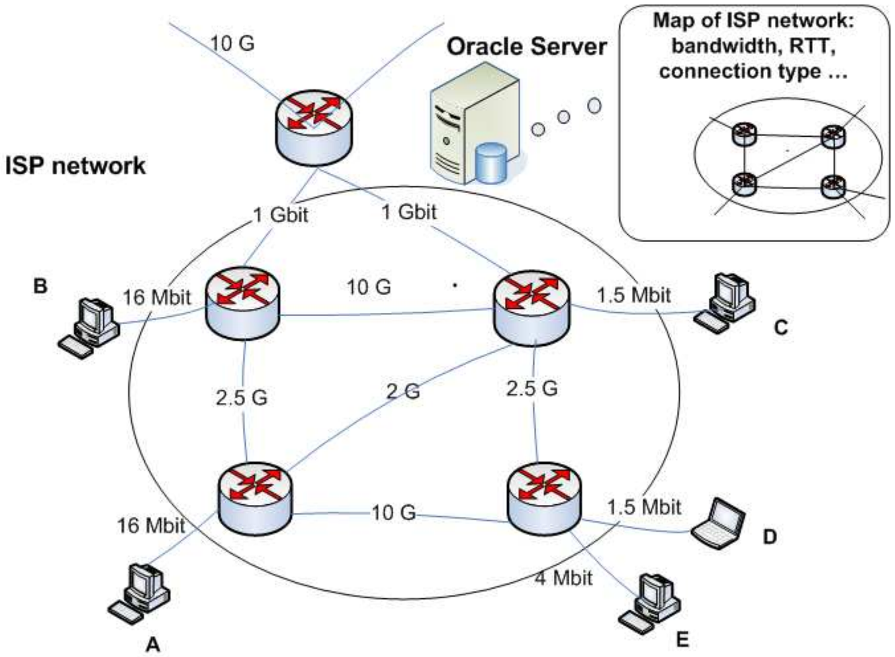
4.5 How the Oracle works

With the help of an example, we show how the P2P users can use the oracle service. Consider the example network shown in Figure 4.2. It shows the simplified internal topology of a hypothetical ISP, with various users A, B, C, D, and E having different last-mile connection bandwidths. The oracle service runs at a publicly known IP address. It has a map of its ISP's entire network topology in the form of a semi-static database, containing information like link bandwidths, router topology, etc.

When user A wishes to connect to another peer for bootstrapping to a P2P network, we assume that it finds B and E as possible candidates through a P2P bootstrapping mechanism. User A queries the oracle server for path properties of B and E. The oracle server knows that B has a last hop bandwidth of 16 Mbit, which is much greater than the 4 Mbit bandwidth of E. Hence, it recommends A to connect to B. The oracle can either rank B ahead of E, or can return a bandwidth classification of B as high, and E as medium. This enables A to connect to a user having a higher bandwidth.

Consider another instance, when E is already connected to both C and D in a P2P network. When E wishes to download a large multimedia file, it queries the oracle about its connected neighbours. The oracle knows that even though both D and C have the same last-hop bandwidth, the user D is topologically and/or geographically closer to E than C. If E downloads the large multimedia file from D instead of C, lesser network resources will be consumed and network congestion will also decrease as a result. Hence, the oracle recommends D over C to the querying node E.

Figure 4.2 Example to show how the oracle service functions



4.6 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 single server or a set of replicated servers within each ISP, that can be queried using a UDP-based protocol, e.g., modeled on *BIND* [15], or run as a Web service. When designed using a protocol similar to BIND, a single packet query can contain up to 350 IP addresses (IPv4) of potential neighbours, which is more than sufficient for contemporary P2P applications. Even with IPv6, about 80 IP addresses will fit into a single packet query, which is still sufficient for current P2P applications. The oracle 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 [83]. This service is capable of satisfying $100,000$ queries using standard PC-hardware [22] in less than one minute. It enables the P2P node to retrieve information about the potential P2P neighbours such as the parent AS and geographic location. This information can then be used by the P2P node to bias its neighbour selection. But purely using the pWhois service only helps to rank potential neighbours based on AS distance. It does not account for last-hop bandwidth of potential neighbours and router- or PoP-level topology. Also, it does not enable cooperation between ISPs and P2P systems. However, the scalability of the pWhoIs service is an encouraging sign to develop a more scalable oracle service, which can rely on powerful computing hardware.

4.7 Evaluation and Analysis

To evaluate the concept of the oracle service, we take the following approach. An overlay or a P2P system that selects neighbours on consulting the ISP-hosted oracle is referred to as a *biased* overlay or a biased P2P system. On the other hand, an overlay or P2P system that does not consult the oracle, and makes neighbourhood selection arbitrarily is termed as a *random* overlay or a random P2P system.

In Chapter 5, we use a generalized overlay system to perform experiments on the graph structural properties like node degree, path length, connectedness, etc., in order to compare random and biased overlays. To analyze if shorter network paths of P2P links lead to increased congestion within an AS network, we use the principle of flow conductance to perform congestion analysis on random and biased overlays. This is followed by a feasibility study of biased neighbour selection in a real P2P system (Gnutella) through testbed implementation as well as Planetlab deployment. In the experiments described in this chapter, the oracle ranks the potential neighbours of an overlay node using AS distance only.

As the feasibility of the proposal has been established at this stage, and the analysis of the graph properties and congestion has led to positive results, we rely in Chapter 6 on extensive packet-level simulations with a real P2P system using more complex network models. First, we validate the graph results from Chapter 5 through SSFNet simulations with Gnutella under churn. We then evaluate the impact of using the oracle on the routing performance of the P2P system using characteristics like P2P scalability, query search performance and localization of content exchange. Drawing upon the insights gained during the tested and Planetlab experiments, especially with regards to P2P content availability, query patterns, and ISP/P2P topologies, we model a range of user behaviour characteristics (churn, content availability, query patterns) as well as multiple ISP/P2P topologies in the simulation environment. We then extend the oracle to also consider the last-hop bandwidth of potential neighbours while sorting them, and use this setup for an intensive study of the effects of P2P user behaviour and ISP/P2P topologies on end-user experience metrics for ISPs as well as P2P users.

In Chapter 7, we propose collaboration between multiple ISPs so that an oracle can give estimates of path properties between any two IP addresses, both within and outside the AS. We also show how a global coordinate system can be built based on multiple-ISP collaboration. Using a very large network topology, we provide experimental results with an application-level P2P simulator. In these experiments, the oracle considers the router-hop count as an additional metric for ranking potential neighbours. We also compare the results of multiple-ISP collaboration with a bandwidth-based P2P neighbour selection scheme.

This will complete the evaluation of the ISP-P2P collaboration proposal made in this thesis. In Chapter 8 we propose an extension to the oracle service that helps to reduce pollution in P2P file-sharing systems, and conclude with a summary of our contributions in Chapter 9.

5 Graph Experiments and Feasibility Study

In this chapter, we evaluate the proposal of ISP-P2P collaboration through the use of the oracle service. We begin by introducing our evaluation methodology in Section 5.1. In Section 5.2, we apply the methodology to evaluate the graph structural properties of biased P2P topologies, and compare them against random neighbour selection. We then undertake a study of the congestion caused by random and localized P2P topologies in Section 5.3. This is followed by a feasibility study of our proposal through experiments in the testbed in Section 5.4, and deployment in the Planetlab in Section 5.5.

5.1 Evaluation Methodology

To overcome the argument that biasing the neighbourhood selection process adversely affects the structural properties of the overlay topology one needs appropriate metrics. In this section, we propose metrics for evaluating the impact of using the oracle on the overlay as well as the underlay topology. These metrics 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. These metrics and the simulation topologies will be the basis for the experiments in this as well as the subsequent chapters.

5.1.1 Metrics

As a basic model for our investigations, we model the AS-graph as a complete directed graph $G = (V,E)$ with a cost function $c : E \rightarrow \mathbb{R}^+$ 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 P, let $AS : P \rightarrow V$ define how the peers are mapped to the ASes and $b : P \rightarrow \mathbb{R}^+$ 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.

Node 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.

Overlay 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 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 [70] 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. [78], and the (theoretically) best construction today is the LAND overlay network [2].

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 $O(\sqrt{\log n})$ [9]. 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 [55]).

Consider a directed network $G = (V,E)$ with edge bandwidths $b : E \rightarrow \mathrm{IR}^+$. 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 on flow conductance.

Summary

We propose measures that are useful for P2P systems and our theoretical 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 will validate through experiments.

5.1.2 Simulation Topologies

The simulation results can be heavily influenced by the topologies used. Hence, we make the basis for our simulations the AS topology of the Internet [65, 68], 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 [89], Routeviews [91], GEANT [44], and Abilene [1]. 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 neighbours in the AS topology graph. 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 neighbours 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 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 [19, 57]. 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 sub-sampling the AS topology derived from the BGP table dumps. Indeed, sensitivity analysis of our results show that if we move more peers from level-1 ASes to level-2 and level-3 ASes, the results improve even more.

5.2 Overlay / Underlay Graph Properties

In this section, we evaluate how the use of the oracle changes the graph properties of the P2P overlay topology. We use the oracle to bias the neighbourhood selection in the overlay to select neighbours within the same AS when possible. For this purpose we use a general graph simulator as it allows us to explore large topologies. We rely on the Subjects simulation environment [95] that is very light-weight, such that we can run experiments on topologies with a large number of ASes, each having many P2P nodes. The Subjects environment has been introduced in Section 2.7.

5.2.1 Simulation setup

In our experiments, the Internet class spawns multiple AS classes in the Subjects environment, and each of the AS classes then spawns a number of overlay node classes. These nodes establish peering connections with each other by exchanging messages (objects), and the relay points serve as an abstraction of the 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.

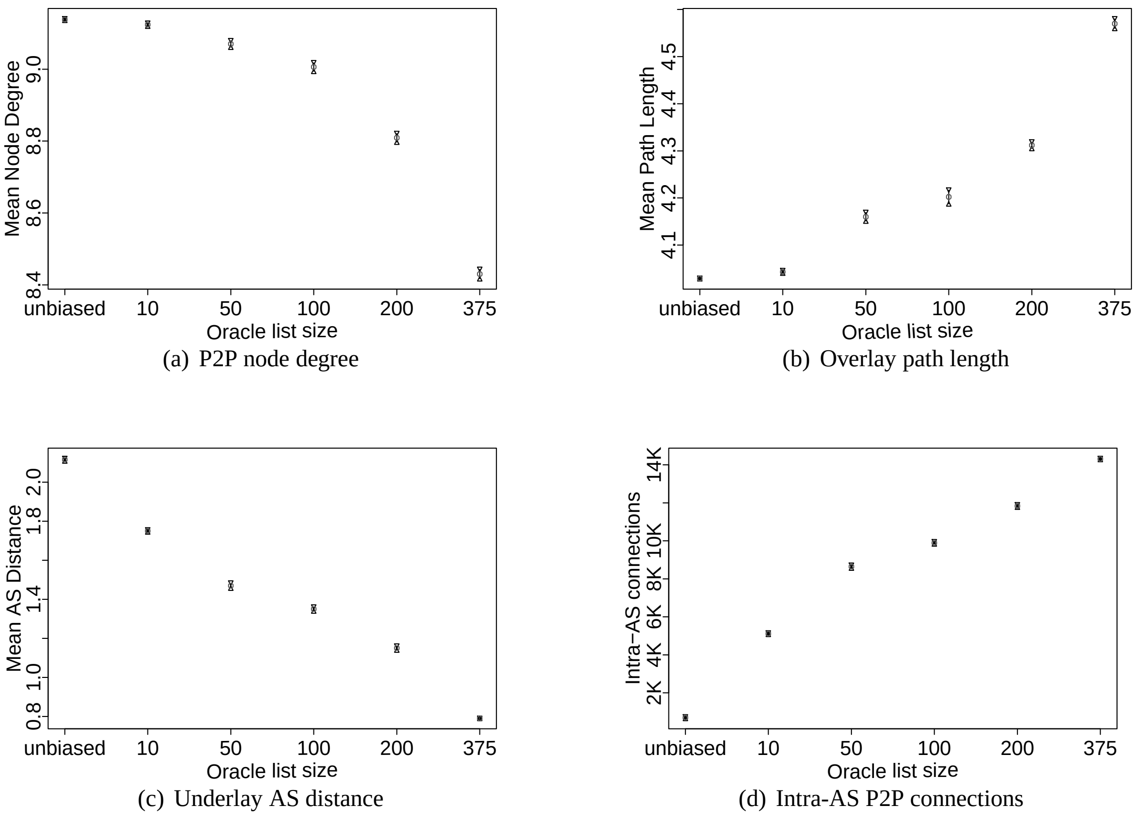
For our evaluation we consider five graphs, each with 300 ASes and 4372 P2P nodes, which results in an average of 14.6 nodes per AS. The topologies are derived by sub-sampling the Internet AS topology as explained in Section 5.1.2. Each graph consists of 4 level-1 ASes, 100 level-2 ASes and 196 level-3 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 to improve our results.

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

- 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 neighbour selection, while the biased case corresponds to a P2P node giving a list of potential neighbours to the oracle, and the oracle helping it pick an optimal neighbour. We simulate the simplest of such oracles where it either chooses a neighbour 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 list of candidates (potential neighbours) of a P2P node, namely, 1, 10, 50, 100, 200, 375. This helps to

Figure 5.1 Comparison of metrics for graph properties with increasing size of oracle list using error plots



(a) P2P node degree

(b) Overlay path length

(c) Underlay AS distance

(d) Intra-AS P2P connections

analyze the effect of the size of the oracle choice list on locality in the overlay. Note, a list length of 1 corresponds to the unbiased case. The candidate list of nodes is filled randomly.

We experiment with establishing from 1000 up to $40,000$ neighbour 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.

To reduce the bias in our experiments, we run 4 experiments for each of the 5 AS graphs where the oracle is used for each neighbour relationship with candidate lists of length 1, 10, 50, 100, 200, 375 - resulting in 120 experiments. The error plots of the results across all the experiments are shown in Figure 5.1.

5.2.2 Results

First, we check whether the overlay graphs remain connected using biased neighbour 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 5.1(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 biased neighbourhood selection increases the diameter and mean path length of the overlay graph, as it prefers “close-by” neighbours. Yet, in all the experiments the hop count diameter of the overlay graph stays at 7 or 8 hops. The AS diameter, which is the maximum over the AS distances between peers, stays constant at 5 hops. Neither does the average path length in the overlay graph increase significantly, see Figure 5.1(b). Therefore we can conclude that the biased neighbourhood selection does not adversely impact the structural properties of the overlay graph.

We find that locality in overlays improves significantly as captured by the average AS-distance of P2P neighbours. Figure 5.1(c) shows how the AS-distance improves with the ability of the P2P node to choose a nearby neighbour. A lower AS-distance should correspond to lower latency. This is also reflected in the number of P2P neighbour connections that stay within each of the ASes, see Figure 5.1(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 6 to 25%, 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.

5.2.3 Summary

With the help of overlay-underlay graph experiments, we have shown that using the oracle to bias the neighbourhood selection of P2P systems does not have any adverse effects on the graph structural properties of the overlay. The principal properties like node degree, mean path length and graph connectedness stay largely unchanged. At the same time, due to choosing neighbours within the same AS when possible, we are able to increase the locality in the P2P topology significantly. Not only do peerings within the same AS increase, there is also a corresponding decrease in the average AS-distance of P2P neighbours, which should correspond to lower latency. The densely connected subgraphs of peerings are now local to the ISPs.

5.3 Congestion Analysis

In this section, we investigate if the localized overlay network maintains its ability to route traffic with low congestion. We initially employ the algorithm in [11] to compute a lower bound for the flow conductance of an overlay graph. Since the run time requirement of our program is $O(n^4)$, we could initially only estimate the flow conductance for small graphs. As such, being able to calculate the conductance of small graphs is not a big problem for the case of unstructured P2P systems. We can calculate the conductance of the graph of superpeers, which is naturally much smaller than the entire overlay connectivity graph comprising both superpeers and the leaf nodes. We initially 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 graph and 0.533

for the 18 node graph (see Section 5.1). 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.

Motivated by the initial positive results, we develop a more efficient approach for analyzing the congestion in random and localized overlay networks. Our improved approach relies on the *augmenting paths* [10] concept. An augmenting path is defined as a path constructed by repeatedly finding a path of positive capacity from a source to a sink and then adding it to the network flow. We first describe our approach, and then present experimental results obtained using this approach.

5.3.1 Approach using Augmenting Paths

Given a set of peers and a set of requests in a general P2P system. Each request can be satisfied by a set of peers, hence, there is a bipartite graph matching from requests to peers, where each request maps to one or more peers. We assume that all the requests are for content of uniform size. The load on a peer is defined as the number of requests it serves. Consider a peering between node *a* and *b*, such that a request made by node *a* is satisfied by node *b*. The peering is defined as *undesirable* for the ISP of *a* if the ISP of *b* is, for example, a provider AS or multiple AS hops away from ISP of *a*. The peering is defined as *desirable* for ISP of *a* if *a* and *b* belong to the same ISP, or if the ISP of *b* is a peering or customer AS of ISP of *a* or has a favourable routing policy to ISP of *a*.

P2P users wish to minimize download time of content, and ISPs wish to minimize interaction with undesirable peers. As the download performance of peers depends upon the load on peers (which implies the congestion in the network), our goal is to achieve a minimization of the maximum load on the P2P system, and a minimization of the number of undesirable peerings. The first goal is the P2P user's interest, while the second goal is the ISP's interest.

When a P2P node finds content available at a set of nodes, it chooses one node randomly and downloads content from it. In other words, given a request that can be satisfied by a set of peers, the request is assigned to a peer randomly chosen from the set of potential peers. A number of requests are generated in the P2P system, which are successively assigned to potential peers randomly.

Strategy: Given this assignment of requests to peers, our goal is to minimize the maximum load as well as the total number of undesirable peerings in the P2P system. For this, we use the concept of augmenting paths [10]. We first run an augmenting paths routine to minimize the maximum load on the system, followed by an augmenting paths routine to minimize the number of undesirable peerings. This will give us the theoretical optimum for the given set of peers and requests, in terms of maximum load and undesirable peerings. We will then calculate the maximum load and the number of undesirable peerings when the oracle helps the peers to establish peerings with potential neighbours. This will enable us to measure the effect of using the oracle on the network congestion, measured in terms of maximum load on the P2P system.

Maximum load minimization: We find the maximum-loaded peer in the system, with load L. We wish to reassign a request from the maximum-loaded peer to another peer with load $<= L-2$, so that the maximum load in the P2P system is reduced by 1. As each request has a set of potential peers that can satisfy it, we examine the requests that are currently assigned to the maximum-loaded peer, and check if they can be reassigned to another peer with load $<= L-2$. This is achieved by use breadth-first-search to investigate if there exists an augmenting path through the peer-to-request and request-to-peer mappings, such that we reach a peer with load $<= L-2$. If such an augmenting path exists, we reassign the request from the original peer to this less-loaded peer, thereby reducing the maximum load on the system by 1. If the breadth-first-search does not reveal such an augmenting path, we augment the search along a path where the peer has load $L-1$. We continue this procedure until no more augmenting paths can be found. At this stage, the maximum load on the system has been minimized.

Undesirable peerings minimization: We apply the same procedure to reduce the number of undesirable peerings in the system. We start with a request that is assigned to an undesirable peer, and can be assigned to a desirable peer. We search for an augmenting path through the peer-to-request and request-to-peer mappings, such that we reach a peer with load $<= L-1$. This constraint ensures that the maximum load in the system is not increased to reduce the undesirable peerings. If an augmenting path is found, we reassign the request from the original peer to the new peer with load $L-1$. In this way, one undesirable peering is replaced by a desirable peering. This procedure is iterated until no more augmenting paths can be found. As this stage, the number of undesirable peerings in the system has been minimized.

We have thus found the theoretically optimum assignment of requests to peers, such that the maximum load as well as the number of undesirable peerings in the system are minimized.

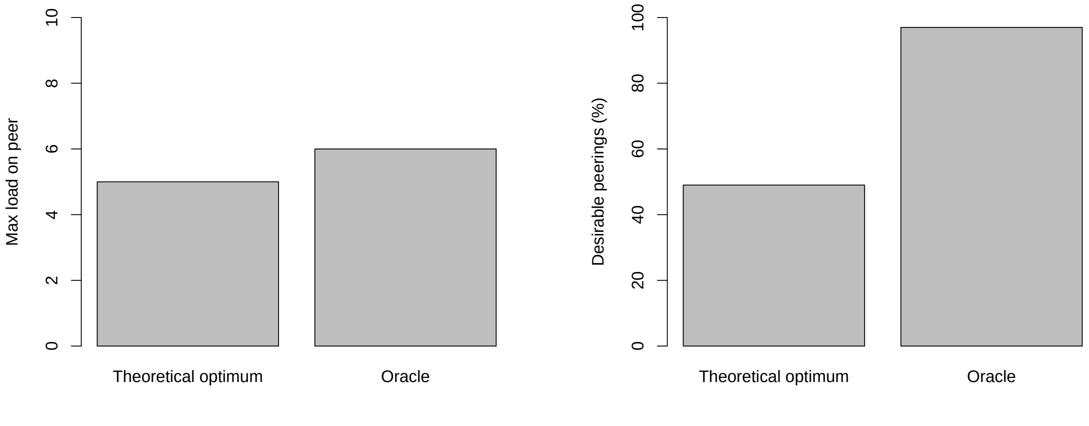
The oracle case: The goal is to compare the obtained theoretical optimum with the oracle-based neighbour selection in P2P systems. For the oracle case, we consider each request as it is generated. From the set of potential peers that can satisfy this particular request, we find the least-loaded desirable peer and the least-loaded undesirable peer. If $load_{desirable\_peer} <= 2 \times load_{undesirable\_peer}$ the request is assigned to the least-loaded desirable peer, else to the least-loaded undesirable peer. When all the requests have been assigned in this manner, we calculate the load on the maximum-loaded peer in the system, as well as the number of desirable peerings in the system. Comparing these values with the theoretical optimum values will enable us to quantify the advantages of using the oracle for neighbour selection in P2P systems. We will also be able to investigate if biased neighbour selection leads to increased congestion in the network.

5.3.2 Results

We implement the described model as a C++ program. The results for a system with $10,000$ peers and $50,000$ peerings are shown in Figure 5.2. We see that the maximum load on the system increases from 5 in the theoretical optimum case to 6 when using the oracle, a very nominal increase. At the same time, the number of desirable peerings increase from 49% in the theoretical optimum case to 97% when using the oracle. Multiple runs of the program with different number of peers and requests give results with a similar magnitude.

The results show that it is possible to assign almost all the requests to desirable peers (i.e., in accordance with the concerns of the ISP), while keeping the congestion in the network close to the theoretical optimum. This also addresses the concern that an ISP may have to invest significantly in its infrastructure if it keeps a large amount of traffic local to its network. As the experiments demonstrate that the congestion due to localized traffic is close to the theoretical optimum, we conclude that an ISP does not necessarily require to upgrade its internal network infrastructure because of increased locality in the P2P traffic. On the other hand, the ISP gains significantly in terms of traffic costs and routing policies when P2P users consult the oracle to choose appropriate

Figure 5.2 Congestion analysis of oracle-aided P2P neighbour selection against theoretical optimum



(a) Maximum load in P2P system

(b) Percentage of desirable peerings

neighbours. The P2P users also benefit from shorter network paths and lesser bottlenecks.

5.4 Feasibility Study in the Testbed

Given the encouraging results on graph properties and congestion analysis of biased generalized overlay graphs in the previous sections, we now evaluate the feasibility of our proposal with a real P2P system, namely Gnutella. For an introduction to the Gnutella P2P system, see Section 2.6.

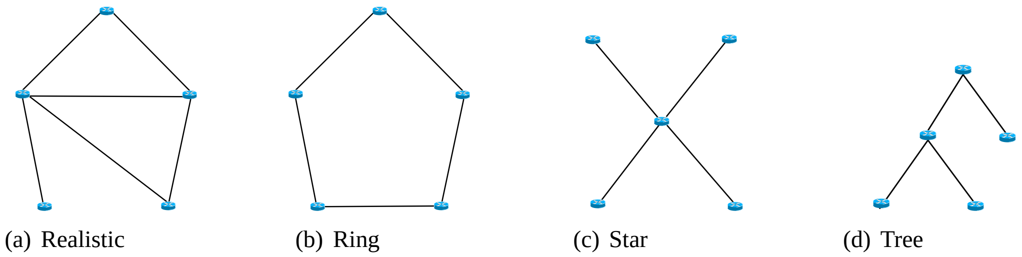
While simulations allow us to experiment with large-sized graphs, containing thousands of P2P nodes, we still have to model the P2P networks and the routing protocols. Hence, in this section, we use a testbed facility to perform the P2P experiments, so that we can run the actual P2P system code, and validate and refine our network models. As we can work directly with real P2P system code, the testbed facility allows us to determine if the oracle-based biased neighbour selection will work with real P2P systems. Once we are sure that the approach is feasible, we can proceed to use a simulation framework to perform rigorous analysis on various aspects of the oracle concept, e.g., to study of the effects of churn, content distribution, complex network topologies, etc., on end-user experience metrics like download times and content localization.

The hardware setup of our testbed has been introduced in Section 2.7.2. In this section, we first explain how we configure various network topologies in the testbed using routers, VLANs and other resources. We then perform experiments with the content search phase of the Gnutella P2P network using different file sharing and query search distributions. This not only serves to evaluate the feasibility of the oracle with a real P2P system, but also to study the impact of the oracle on query search.

5.4.1 Configuration of Topologies in Testbed

We devise topologies with multiple ASes, where each AS hosts multiple machines running P2P applications. As a router can be taken as an abstraction of an AS boundary, and we have 5 routers available (one router was unavailable due to hardware malfunction), we decide to form 5-AS topologies. Each router connects to 3 load-generators, and the memory requirements of the P2P software allow us to run 3 instances of the P2P application on each machine concurrently. This gives us an upper bound of 5-AS topologies, with 15 machines, running 45 P2P clients concurrently. We connect the 5 routers in different ways as shown in Figure 5.3, to arrive at 4 different AS topologies, which we call realistic, ring, star, and tree topologies. Running P2P experiments on different AS topologies enables us to analyze the impact of underlay topologies on P2P locality.

Figure 5.3 AS topologies used in the testbed experiments



(a) Realistic (b) Ring (c) Star (d) Tree

We briefly explain how we configure the testbed hardware to achieve the desired underlay topologies. All the interfaces are first assigned IP addresses using a pre-defined subnet layout structure. Since each router has only two interfaces, one for router-to-router connections and the other for router-to-loadgenerator connections, we have to assign multiple IP addresses to each router interface to create more than one router-to-router connections on a router. This is achieved by using IEEE 802.1Q VLANs.

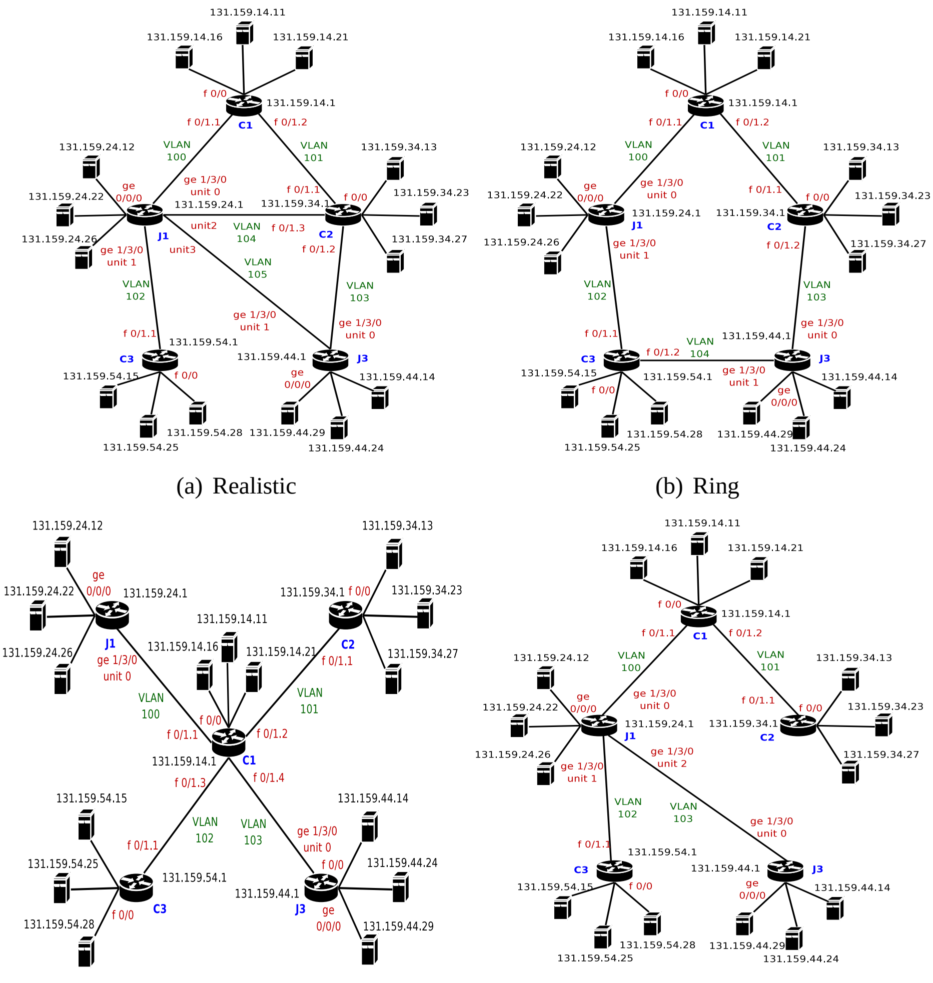
Virtual LAN, commonly known as **VLAN** [118], is a group of devices on one or more LANs that are configured so that they can communicate as if they are attached to the same wire, when in fact they are located on a number of different LAN segments. Because VLANs are based on logical instead of physical connections, they are very flexible for user/host management, bandwidth allocation and resource optimization. By using VLAN-capable hardware devices, it is possible to define more than one Ethernet segment on a port-by-port basis without changing the hardware setup. In the testbed we use the widely used IEEE 802.1Q [43] standard. The router-to-loadgenerator connections are configured using the commands: (i) *ifconfig*: for defining IP addresses on a particular ethernet interface (ii) *route*: for setting up the gateway of routes.

We thus configure each of the four different AS topologies shown in Figure 5.3 such that router-to-router connection is established by VLAN interfaces and each router is connected with 3 loadgenerators, which amounts to a total of 15 load-generators. The final configuration of the testbed devices is shown in Figure 5.4. The configuration details of various topologies can be found in [39, 122].

5.4.2 Testbed Experiments

The first steps consist of installing the Gnutella P2P software on each machine. To be able to install multiple Gnutella servents on each machine, we use the C-based GTK-Gnutella [38] software with a textual interface. By installing three servents each on 15 machines, we have 45 Gnutella servents in our experiments. We designate one servent on each machine to be an ultrapeer, while the other

Figure 5.4 Configuration of testbed devices to achieve the AS topologies



(a) Realistic

(b) Ring

(c) Star

(d) Tree

two are made leaf nodes. This gives us a testbed Gnutella network with 15 ultrapeers and 30 leaf nodes.

Realizing Biased Query Search

A central machine which is connected to all the other load-generators is used to run the oracle. When a Gnutella servent sends a list of IP addresses to the oracle, the oracle sorts this list in the order of, first, servents within the querying servent AS, followed by servents in the AS which is one AS-hop away, followed by servents at increasing AS-hop distance.

To explore the aspect of content search and exchange using an oracle in an actual testbed with real P2P traffic, we employ the following scheme. We first run an experiment with the unmodified Gnutella protocol running on each servent, which does not consult the oracle for neighbourhood selection. We then run another experiment, where each servent (both ultrapeer and leaf node) consults the oracle. To concentrate on content search and exchange, we let each servent communicate with the oracle and send the Query search messages to only those neighbours which are within its AS. Only if there are no neighbours within the same AS, does the servent send the Query to neighbours which belongs to ASes that are least AS-hops away. Hence, a biased Gnutella servent consults the oracle actively during the content search phase. In contrast, servents in unmodified Gnutella flood the Query messages to all their connected neighbours, irrespective of their AS.

As the file sharing pattern of P2P users can impact the content search experiment results, we employ two file sharing schemes:

- **Uniform**: Every servent (both ultrapeer and leaf node) shares 6 unique files, leading to a total of 270 files in the testbed.
- **Variable**: All ultrapeers share 12 files, half the leaf nodes share 6 files each, and the remaining leaf nodes share no files (free-riders). The content of files within any AS is kept the same as in uniform scheme, i.e., within each AS, one leaf shares no files, one leaf shares 6 files, and the ultrapeer shares 12 files.

The aim of the experiments is to compare the impact of the oracle on the content search process of P2P systems. More specifically, we wish to compare the number of QueryHit messages received by each servent with and without consulting the oracle, for uniform and variable file sharing schemes. We let each servent introduce a unique query search string in the network. To better reflect P2P user behaviour, we use query strings that search for content of a particular type (e.g., mp3, rar), as well as those that search for something specific, e.g., a file name [31].

Results

First, we measure the number of Query messages that are relayed in the entire testbed network, and present the results in Tables 5.1 and 5.2. There are only 45 unique Query strings in both cases, but when a Query message is forwarded by a servent to its *n* neighbours, it is counted *n* times. This helps us to quantify the impact of biased neighbour selection on the scalability of the Gnutella network.

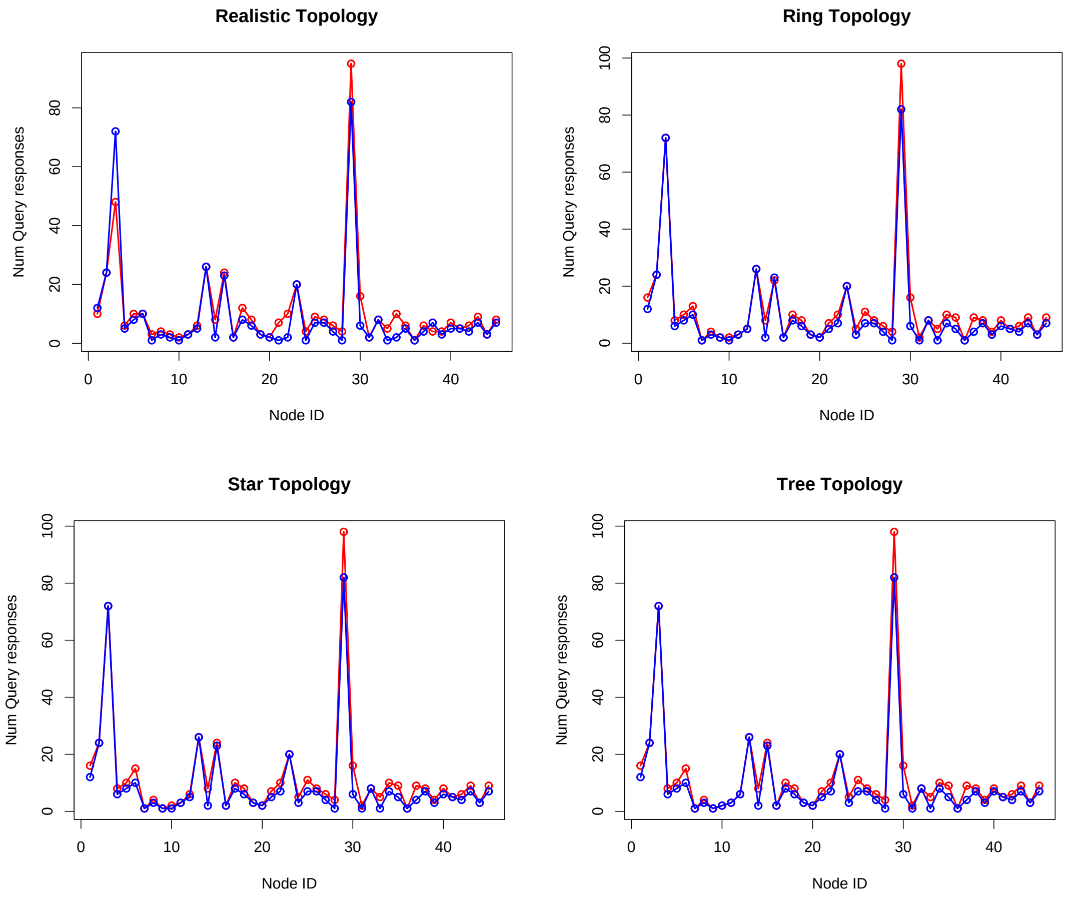
Table 5.1 Total number of query search messages relayed in the network using uniform file sharing

Topology	Unmodified P2P	Biased P2P
Realistic	6604	2473
Ring	6623	2512
Star	6679	2533
Tree	6643	2468

Table 5.2 Total number of query search messages relayed in the network using variable file sharing

Topology	Unmodified P2P	Biased P2P
Realistic	10194	4873
Ring	10939	4834
Star	10902	4863
Tree	10872	4847

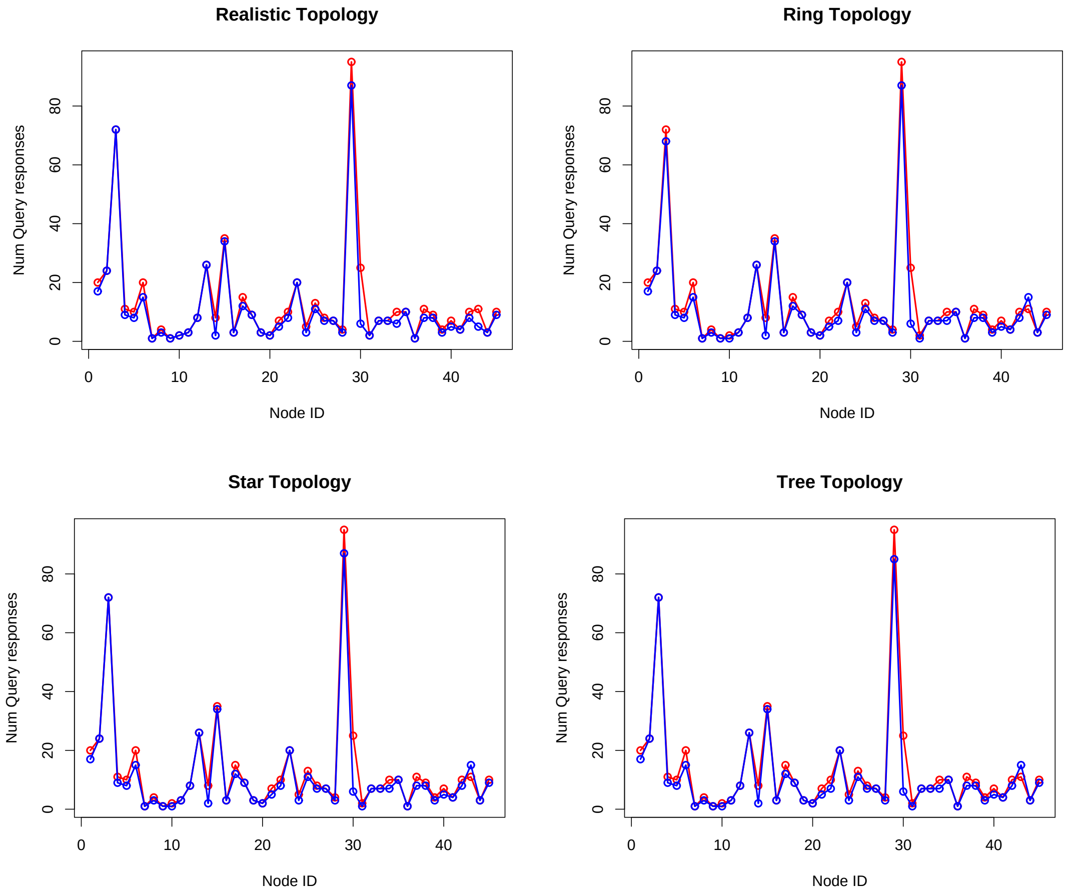
Figure 5.5 Number of query response messages (y-axis) for each P2P node (x-axis) in the four topologies, for uniform file sharing. The red lines are for unmodified P2P, while the blue lines are for the oracle-influenced P2P.



We see that consulting the oracle during content search reduces the number of `Query` messages that are relayed in the network, for both uniform and variable file sharing. The reason for the larger number of messages with variable file sharing is that a `Query` often arrives at a servent which is not sharing any content, and is hence further forwarded to this servent's neighbours, thus generating more negotiation traffic. But even with variable file sharing, forwarding the `Query` messages with the help of the ISP-hosted oracle to nearest neighbours reduces the negotiation traffic by at least 50%. As negotiation traffic for content search forms a significant portion of P2P traffic [31], we conclude that consulting the oracle significantly improves the scalability of such P2P networks.

We now measure the number of `QueryHit` messages received by each Gnutella servent, for the unique query string that it introduces in the network. We compare the number of responses received in unmodified Gnutella experiments with that of oracle-influenced Gnutella experiments. Figure 5.5 shows the results for uniform file sharing, while Figure 5.6 shows the results for variable file sharing. The peaks in the plots correspond to general queries (e.g., mp3, rar) which match a large number of files in the network, while the other values refer to more specific content (e.g., artist or album name). We see that while consulting the oracle during content search often reduces the number of

Figure 5.6 Number of query response messages (y-axis) for each P2P node (x-axis) in the four topologies, for variable file sharing. The red lines are for unmodified P2P, while the blue lines are for the oracle-influenced P2P protocol. The query strings used in these experiments are identical to those used in Figure 5.5.

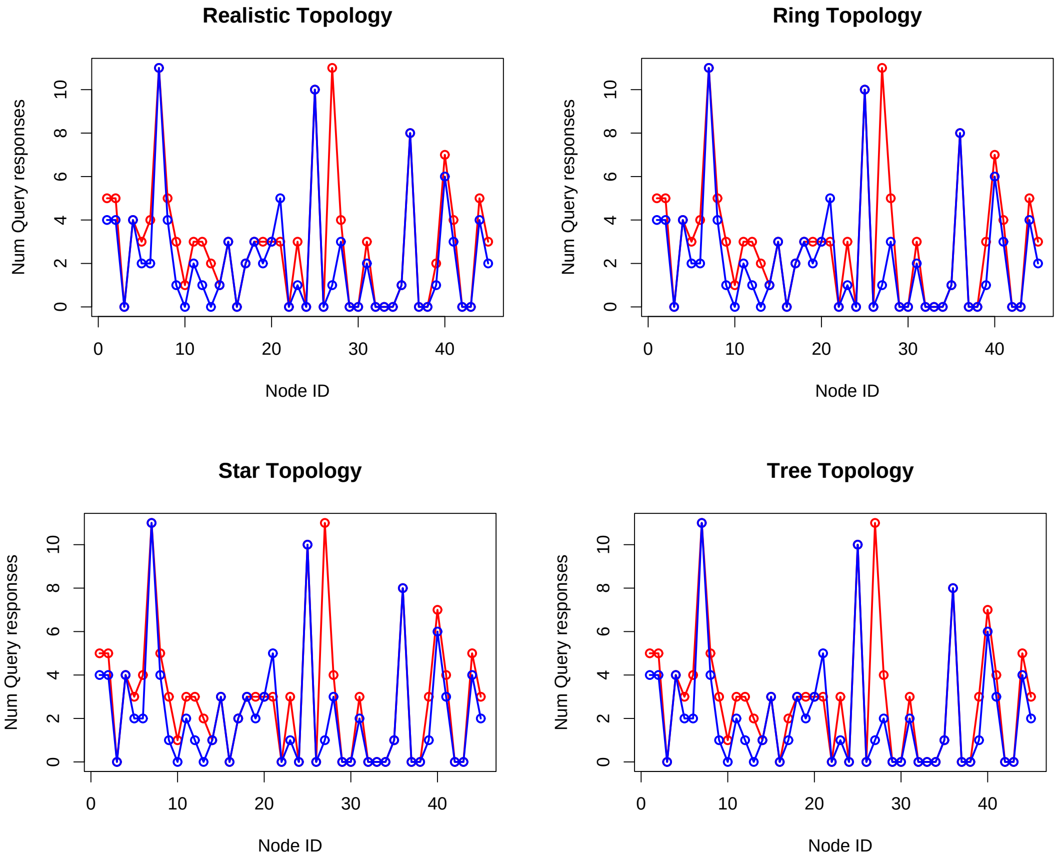


`QueryHit` messages received by a servent, the difference is only nominal. But most importantly, we do not find any case where a `Query` yields a result in unmodified Gnutella, but fails to do so when consulting the oracle.

As the pattern and quality of query strings can also affect results [31], we run another set of experiments by changing the set of query strings in variable file sharing scheme. Here queries have a much lesser chance of finding file content, i.e., they are unlikely to yield a `QueryHit`. This helps to detect whether there are servents that get only a small number of `QueryHits` with unmodified Gnutella, which fail to get any `QueryHits` at all when consulting the oracle.

The results are shown in Figure 5.7. We again see only a nominal reduction in the number of `QueryHit` messages for oracle-influenced Gnutella servents. Besides, we detect only 2 servents (both leaf nodes) out of a total of 45, which did not receive any `QueryHit` when using the oracle, while they received 1 and 2 `QueryHits` respectively with the unmodified Gnutella. However, we find that increasing the TTL of the `Query` message by 1 results in both these queries being successfully resolved as well.

Figure 5.7 Number of query response messages (y-axis) for each P2P node (x-axis) in the four topologies, for variable file sharing. The red lines are for unmodified P2P, while the blue lines are for the oracle-influenced P2P protocol. The query strings used have a much lower chance of successful content search as compared to queries in Figure 5.5.



5.4.3 Summary

We conclude that consulting the oracle does not adversely affect the content search process of P2P networks. P2P nodes are easily able to search and share content, while the scalability of the P2P system improves considerably. The volume of P2P negotiation traffic in the network is reduced by at least 50%, while the content search performance remains comparable. In other words, the overlay-induced traffic in the underlay is reduced by 50%, with no adverse impact on P2P performance. Overall, we find that the P2P system continues to behave as per its protocol, with users able to locate and share content.

Running experiments over different AS topologies as well as different file sharing models imply that the benefits accruing from consulting the oracle for P2P neighbour selection are independent of the underlay topology and P2P user behaviour. The insights gained during the testbed implementation, especially with regards to content availability, query patterns, and AS topologies, prove very useful in modeling larger experiments with simulation frameworks in the subsequent chapters.

5.5 Deployment in the Planetlab

The next stage in the feasibility study of the ISP-P2P collaboration concept is to analyze the interaction of biased P2P nodes with their unbiased peers running over the Internet. For this, we need many computers running a real P2P application, spread throughout the globe, using biased neighbour selection to connect to proximal neighbours, and participating in a real P2P network running in the Internet. The Planetlab infrastructure, introduced in Chapter 2, lends itself well for this purpose. The most popular client software for the Gnutella P2P protocol used in the Internet is LimeWire [109]. Hence, we modify the LimeWire [61] software to use biased neighbour selection, so that the servent only connects to neighbours within its own AS, and install the modified LimeWire servents at multiple Planetlab sites spread throughout the world. This enables us to observe the interaction of biased Gnutella servents with other (unmodified) Gnutella servents running in the Internet. We introduce our experimental setup in Section 5.5.1, followed by some results in Section 5.5.2.

5.5.1 Experimental Setup

To enable biased neighbour selection, we modify the source code of LimeWire servent software. When a LimeWire servent connects to the P2P network, it finds the AS of each potential neighbour, and connects to only those servents that are within its own AS. If a servent does not find a neighbour within its AS, it searches for neighbours in ASes at an AS-hop distance of 1 from its own AS, followed by those at AS-hop distance of 2. As the oracle service is not yet offered by an ISP, the mapping of IP addresses to the parent AS is done using the *whois* [120] service, which we integrate into the LimeWire source code at each node. The AS-hop distance between ASes is calculated using BGP table dumps from RIPE [89] along with data from [22]. Using the *whois* service implies that we can only use the AS distance to rank potential neighbours of a P2P node, and not consider metrics like last-hop bandwidth or router hops. However, as we are interested in analyzing the interaction of biased P2P nodes with their unbiased peers, the *whois* service suffices for this feasibility study. For more details on the experimental setup, we refer the reader to [123].

We run multiple sessions of P2P experiments in the Planetlab, where we start 100 – 120 Planetlab nodes (distributed equally between North America, Europe and Asia) running the modified LimeWire servents, which then connect to the standard Gnutella network. We program our modified servents to send specific queries to the network, and compare the number of responses and other metrics as presented in the next section. We initially run a Planetlab experiment to measure the frequency of routed query reply strings in the Gnutella network, and generate a set of query strings that represent frequent, infrequent and random strings. This enables us to study the response to various kinds of query strings when we use biased neighbour selection in the Gnutella network.

The actual experiment is performed as follows. First, all LimeWire servents in Planetlab connect to the Gnutella P2P network normally, without biased neighbourhood selection. This enables the LimeWire servents running in the Planetlab to establish a reasonable number of peerings throughout the Internet. A query is issued from each Planetlab node every 2 minutes. After 35 minutes, biased neighbour selection is activated, wherein Planetlab nodes drop all peerings which are not within the same AS. Now, the same queries that were sent in the initial 35 minutes, are issued again by the Planetlab nodes every 2 minutes. As the Planetlab nodes are only connected to proximal neighbours at this stage, this enables us to compare the query responses to the same set of queries for unmodified and biased peerings for our Planetlab nodes. A connection list as well as a status and a statistical message is generated every 60 seconds.

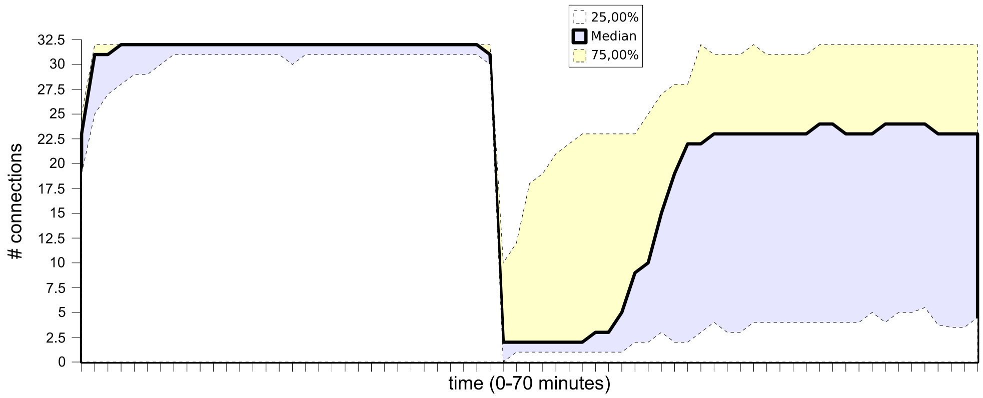
5.5.2 Results

The metrics used for comparing unmodified and biased P2P neighbour selection are the number of ultrapeer connections, AS-hop distance distribution of query replies, the number of query replies, and the total number of messages carried in the overlay.

Ultrapeer connection count:

The default number of ultrapeer connections in Gnutella is 32. Figure 5.8 shows a plot of the median number of connections for each active ultrapeer. We can easily see that when biased neighbour selection is activated, there is an immediate drop in the number of neighbours as all inter-AS peerings are dropped. But the system stabilizes soon, and we reach a median value of 22 connections per node. Though this number is less than the default setting, it is still large enough to guarantee proper functioning of the nodes within the P2P system.

Figure 5.8 Lower quartile, median and upper quartile of number of neighbours maintained by the ultrapeers. X-axis denotes time, oracle is switched on at 35 minutes, y-axis denotes number of neighbours.



AS-hop distance distribution of query replies:

The most important metric is the AS-hop distance of overlay peers from the querying node that satisfy the query request. This is because the actual file download is done using a direct HTTP connection with the file owner. It is thus interesting to see if, while using the oracle we are able to find content at proximal P2P nodes.

Figure 5.9 plots the absolute number of all received query reply messages based on AS-hop distance. To reflect the proportions better, we plot the same data again in Figure 5.10 using a logarithmic y-axis. We clearly see that the larger AS-distance reply messages have been reduced heavily while the number of 1- or 2-hop distance replies stay largely the same. This implies that we reduce the number of messages that cross AS boundaries, while still finding desired content in the proximity of the querying node. This will naturally improve the prospects of the content transfer taking place within the AS boundaries as well. Also, this result is a good indication of an inherent content locality in file-sharing networks, which is due to geographical and linguistic reasons. For

Figure 5.9 AS-hop distance of query responses

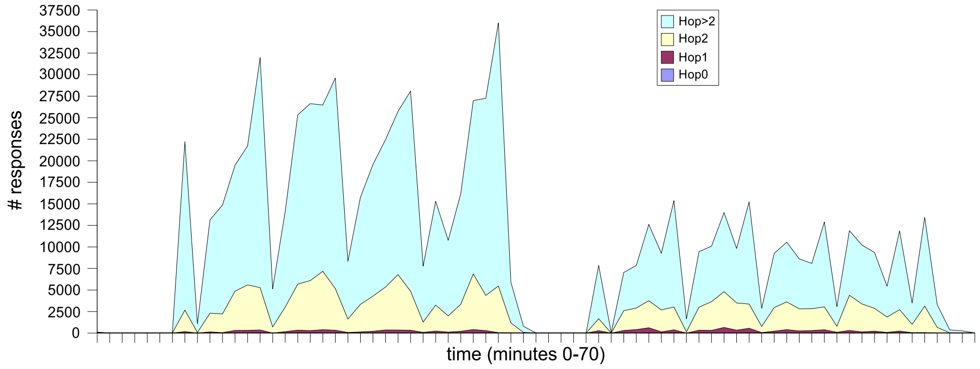
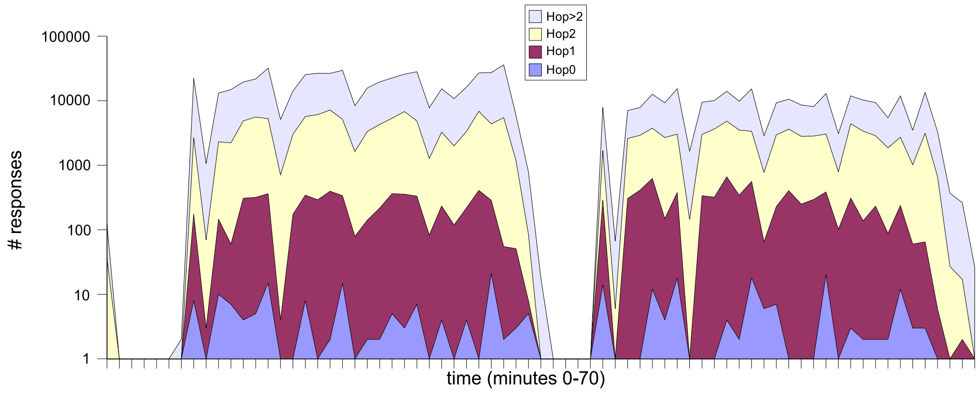


Figure 5.10 AS-hop distance of query responses using logarithmic y-axis



example, German language content is more likely to be found in Germany than in Asia. The ISP-aided biased P2P neighbour selection scheme proposed by us helps to utilize this inherent locality so that both ISPs as well as P2P systems benefit.

Query reply rates:

The rate of received query reply messages is plotted in Figure 5.11. We see that even though biased neighbour selection reduces the number of query replies, we still receive enough responses so as to not affect the proper functioning of the P2P network.

TCP message rates:

We next consider the total number of all received TCP messages by the P2P nodes in Figure 5.12. The TCP messages include, in addition to the query and their reply messages, Ping, Pong, and other connection establishment, maintenance, and connection teardown messages. This enables us to estimate the effect of the oracle on the overall overlay in-band network traffic, i.e., the total

Figure 5.11 Lower quartile, median and upper quartile of number of received query reply messages by the servents, with and without the oracle

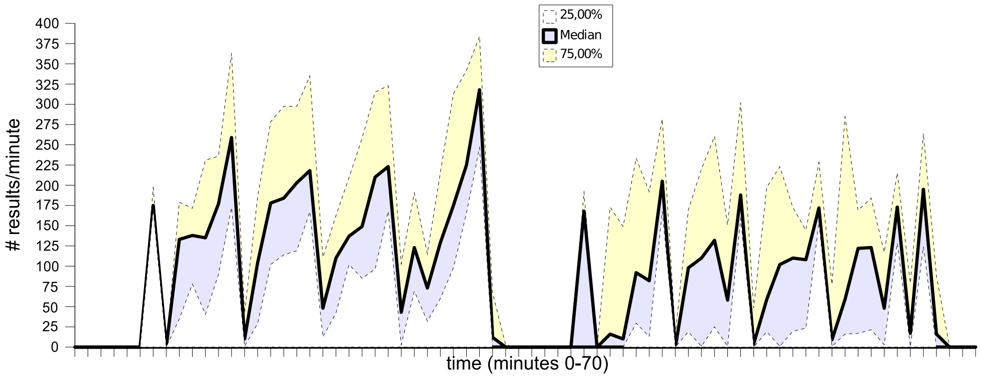
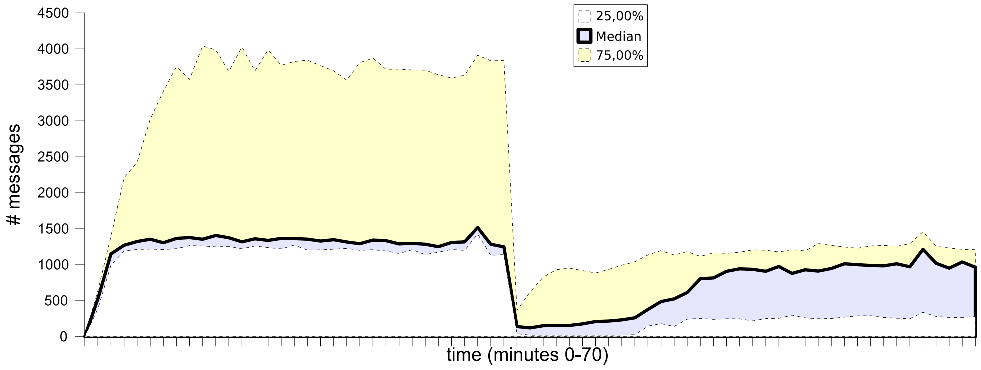


Figure 5.12 Number of TCP messages received by all the servents, with and without the oracle



negotiation traffic that is induced by the overlay in the Internet underlay. Note, that this does not include the actual content transfer. Hence, this is a measure of the reduction in network overhead traffic caused by a P2P system.

It is clearly visible from Figure 5.12 that soon after the oracle is switched on, we see a tremendous drop in the number of TCP messages received by the P2P nodes in Planetlab, a positive result. Despite this welcome reduction in P2P negotiation traffic, there is no adverse effect on P2P performance, as is evident from the adequate number of query responses shown in Figure 5.11.

Through Planetlab deployment and testing of biased P2P neighbourhood selection, we find that the scheme is feasible with real P2P systems in the Internet. The biased P2P nodes interact properly with the nodes of the standard Gnutella network running in the Internet. The P2P nodes form a connected overlay, are able to route messages and search for desired content. The P2P system code lends itself to be modified easily to realize biased neighbour selection. We notice an overall reduction in the P2P negotiation traffic, while the P2P nodes are able to find desired content in their proximity.

We also find that services like *whois* can only enable P2P users to find neighbours within their own AS. They cannot help peers find high-performance neighbours in the sense of high last-hop bandwidth or lesser router hops, which is possible with an ISP-hosted oracle service. Hence, more performance improvement in the sense of end-user metrics like download times, etc. will only be possible by ISP-P2P collaboration, as is shown in the next chapter.

5.6 Conclusion

To evaluate the concept of the P2P nodes consulting an ISP-hosted oracle for neighbourhood selection, we perform experiments with overlay-underlay graphs in a graph simulator, as well as experiments in a testbed and Planetlab. The graph results show that the overlay graphs, on consulting the oracle, are able to increase intra-AS peerings heavily, without any adverse effects on the graph structural properties. Densely connected subgraphs are now local to the ISPs, while only a few peerings leave the AS boundary. This helps to keep the overlay graph connected, as well as to find content which is available outside the AS.

A rigorous theoretical analysis of the congestion caused by shorter network paths of P2P links reveals that the congestion in the network is close to the theoretical optimum. This comes with the added advantage that almost all the P2P links are formed in accordance with the ISP policies. In other words, P2P users experience shorter network paths and lesser bottlenecks, with overall network congestion close to the theoretical optimum. At the same time, ISPs save immense costs by keeping P2P traffic local to their network boundaries, or letting it flow along desirable links outside their network while respecting their routing policies.

Experiments in the testbed and Planetlab with a real P2P system show that the ISP-P2P collaboration concept is feasible, with promising advantages for both ISPs as well as P2P systems. The scalability of P2P systems improves due to a reduction in the overhead traffic in the overlay. A large amount of P2P traffic does not cross the ISP network boundaries, and there is no adverse impact on the query search performance of P2P systems. The insights gained during the testbed and Planetlab experiments prove very useful in designing experiments with larger topologies in a simulation framework.

6 Packet-level Simulations

In the previous chapter, we have presented results on overlay-underlay graph properties, congestion analysis, and a feasibility study through testbed and Planetlab deployments for ISP-P2P collaboration. In this chapter, we perform extensive experiments on a real P2P system in a packet-level simulation framework. The use of a packet-level simulation framework, that supports TCP and message routing to the packet level, allows us to model complex network topologies, including entities like routers, links, etc. and characteristics like bandwidth and delay. The goal is to perform experiments in a controlled setting, to be able to evaluate the impact of various parameters on P2P and ISP performance metrics. The emulation of the underlay topology along with routers, links, hosts, delays, bandwidths, TCP/IP, OSPF and BGP protocols enables us to study the interaction of overlay and underlay routing and the impact of events in one layer on the other layer. Among other things, we model churn and content availability in P2P systems, and experiment with various ISP and P2P topologies. We use packet-level simulations to study the impact of using the oracle to choose P2P neighbours on P2P routing performance, scalability, overlay graph properties, as well as end-user experience metrics like content download times, content locality, and query search results.

In Section 6.1, we validate the graph results from Chapter 5 in a real P2P system under churn, and present results on swarming of queries and their responses, P2P scalability, and content localization. In Section 6.2, we model different user behaviour characteristics, namely churn, content distribution and query strings, as well as different ISP/P2P topologies in the simulation framework, and study their impact on ISP and P2P performance using end-user experience metrics like content download times, and network locality of query responses and desired content.

6.1 Simulations with an Actual P2P System

In Chapter 5 we have seen that the results of biased neighbour 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 the packet-level network simulator SSFNet. We validate the graph results from Chapter 5 in the Gnutella network under churn, and present results on swarming of queries and their responses, P2P scalability, and content localization.

6.1.1 Simulation Setup

The topologies are derived using the same methodology explained in Section 5.1.2. 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, thus distributing the P2P nodes almost equally among level-1, level-2, and level-3 ASes. 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. Each ultrapeer initiates at least 10 connections to other Gnutella nodes itself. It 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. In this section, we take these time periods as uniformly distributed but in Section 6.2, we will use more representative distributions for churn as well as content distribution as recently revealed in studies, e.g., [108].

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 $20-25\%$ of the nodes are off-line and a quarter of the online nodes are functioning as ultrapeers. Note, that all the nodes in our simulations experience churn. This is more aggressive as compared to other studies, e.g., [63], which assume that only half of the nodes in the simulation experience churn, the other half being permanently online.

We run three different experiments with the following parameters for the Gnutella nodes:

- HostCache size = 1000, without oracle
- HostCache size = 100, with oracle for neighbour selection
- HostCache size = 1000, with oracle for neighbour selection

The HostCache [34] is private list of potential neighbours maintained at each node. It is typically populated by Web caches, and content of `Pong` and `QueryHit` messages. In our implementation, each Gnutella node sends the contents of its HostCache to the oracle, which ranks the list of IP addresses according to their proximity from the querying node, and sends the sorted HostCache back to the querying node. The node then establishes a peering connection to the top-most peer in its HostCache. If the connection is unsuccessful (due to the peer being offline or unable to accept incoming connections), the node attempts to connect to the next peer in the list, and so on. When not consulting the oracle, a Gnutella node connects to a peer chosen from its HostCache randomly.

The oracle sorts the list of potential neighbours using the following algorithm: (i) identify nodes within its AS, and place them in the beginning of the list, (ii) for nodes not within its AS, sort them according to AS-hop distance.

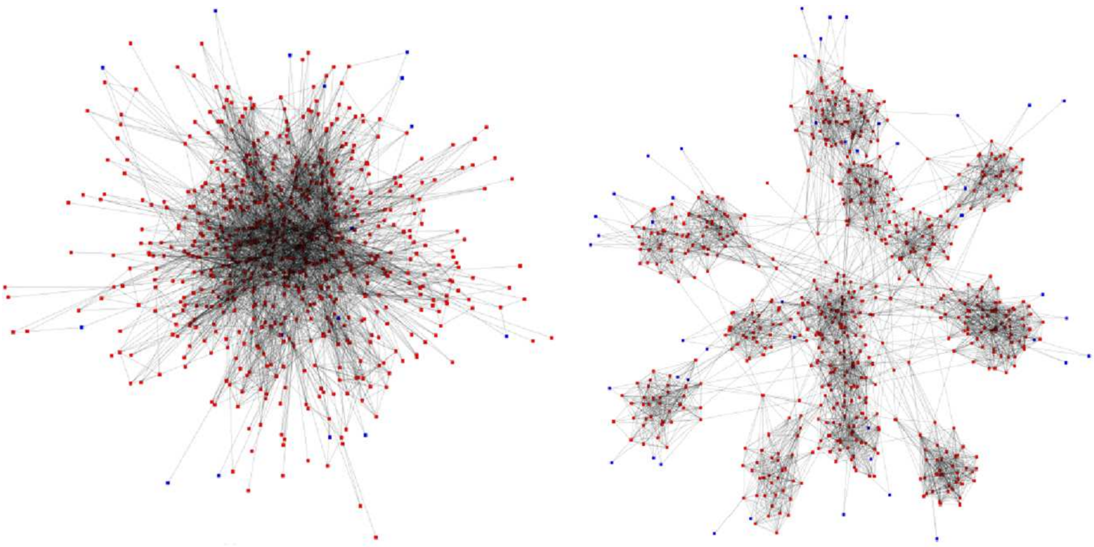
The number of queries in each of the three experiments is the same, and their success rates are also similar. We ran multiple simulations for arbitrary lengths of time and found that the startup phase of the simulation lasts for about 500 seconds. After 5000 seconds of simulation time, the summary statistics do not show significant changes. Therefore we run our simulations for 5000 seconds.

We first analyze the Gnutella overlay graph under churn using the metrics introduced in Section 5.1, followed by an evaluation of metrics such as scalability of the P2P network, number of messages exchanged, swarming pattern of search queries, and localization of content exchange.

6.1.2 Results for Graph Structural Properties

To explore the influence of consulting the oracle on the network topology we visualize the Gnutella overlay topology, for the unbiased case and the biased case with oracle list size 1000. At a particular

Figure 6.1 Visualization of the Overlay Topology



(a) Unbiased P2P

(b) P2P with Oracle

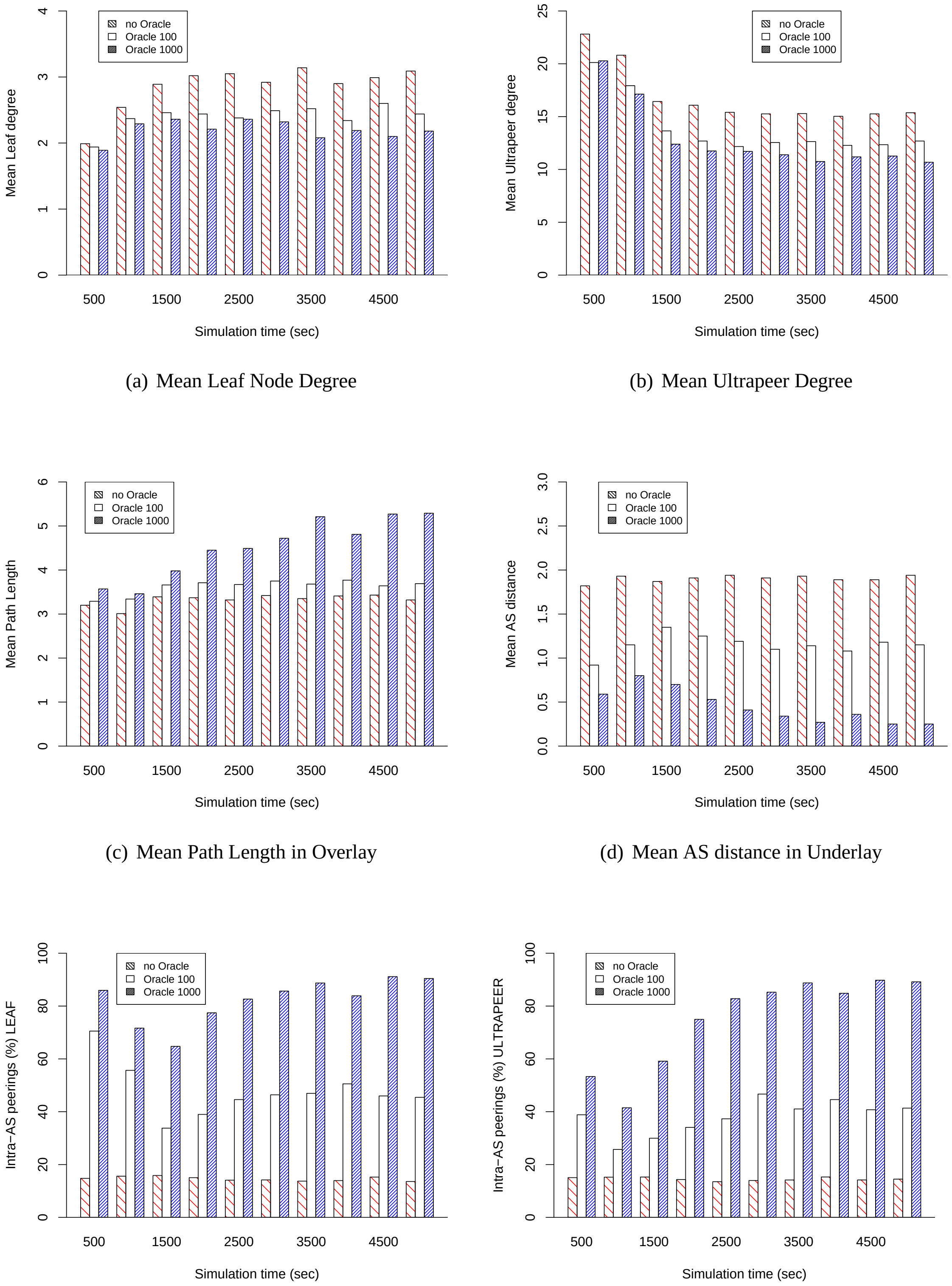
instant in time, we sample the Gnutella overlay topology, display all the online nodes in the graph, and join two nodes with an edge if there exists a Gnutella peering between them at this point of time [113]. Then, using the visualization library yWorks [127], we convert both the graphs into a structured hierarchical format. The resulting graph structures are displayed in Figure 6.1. We can easily observe that the P2P 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 P2P 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 P2P network 10 times during the simulation run, i.e., every 500 seconds. The results are shown in Figure 6.2 on page 64.

Graph connectivity: We begin by checking whether the overlay network graph remains connected using biased neighbour selection. We define the overlay graph at a particular time instant as the graph formed by P2P nodes that are online at that instant, where two nodes are connected by an edge if there exists a P2P connection between them at that instant. We experimentally verify that the overlay graph remains connected at all 10 times where we sample the network, for all three cases. Hence, biased neighbour selection does not affect the connectivity of the overlay network.

Mean Node Degree: Since ultrapeers have a much larger node degree than leaf nodes, we show, in Figure 6.2(a) and (b), how the mean node degree changes over time in a bar plot for all three cases separately for ultrapeers and leaf nodes. This enables us to check if biased neighbour selection affects the structural properties of the overlay network 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

Figure 6.2 Overlay-underlay graph properties in Gnutella SSFNet simulations



(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

15.29 to 10.75, again at 3500 seconds. In other words, despite biasing the neighbour selection via the oracle, the node degree for both leafs and ultrapeers stays within the expected range, and the hierarchical network structure of Gnutella, consisting of high-degree ultrapeers and low-degree leaf nodes, 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 at 4 hops in all the cases.

Mean Overlay Path Length: The average path length in the overlay, shown in Figure 6.2(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 neighbourhood in Gnutella are visible in Figure 6.2(d), which shows the average AS distance (in the underlay) between any two connected overlay 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 is a major relief for ISPs, as they reduce costs heavily for traffic not leaving their domains. Also, traffic that does not leave the network is easier to manage, and it will not encounter inter-ISP bottlenecks [5].

Intra-AS P2P Connections: The above observations on AS distance are further substantiated by the plots in Figure 6.2(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 6.2(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 the 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 6.2(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 Chapter 5. Now we will examine some features related to routing in P2P systems, namely, query search, its impact on scalability of the P2P network, and locality of content exchange. These metrics will help to determine the impact of the oracle on the routing of P2P traffic in the overlay, and its subsequent impact on the Internet underlay. For these results, we concentrate only on comparing the unmodified P2P case with the biased P2P case where cache size is 1000.

6.1.3 Query Search and Network Scalability

The negotiation traffic in many P2P systems like Gnutella represents a large portion of the total P2P traffic [31]. We measure the number of query search messages relayed in the network, using unmodified as well as biased P2P networks. In each case, a total of about 900 unique query messages are generated by different nodes in the network, which are then relayed by the originating nodes to their connected neighbours. The total number of relayed query messages, observed at each time-to-live (TTL) value are shown in Table 6.1. Note that the number of unique messages generated is the same in both cases. However, when a `Ping` or `Query` is generated by a node, and flooded to its `n` neighbours, the message is counted `n` times. Hence, the table shows the total number of query messages carried in the Gnutella overlay.

Table 6.1 Total number of query search messages that are relayed in the network

TTL	Unmodified P2P	Biased P2P
7	19,725	11,149
6	414,718	186,473
5	3,611,604	986,261
4	7,190,754	2,287,036
3	947,035	1,592,910
2	30,653	497,464
1	2,093	74,460
Total	12,216,582	5,635,753

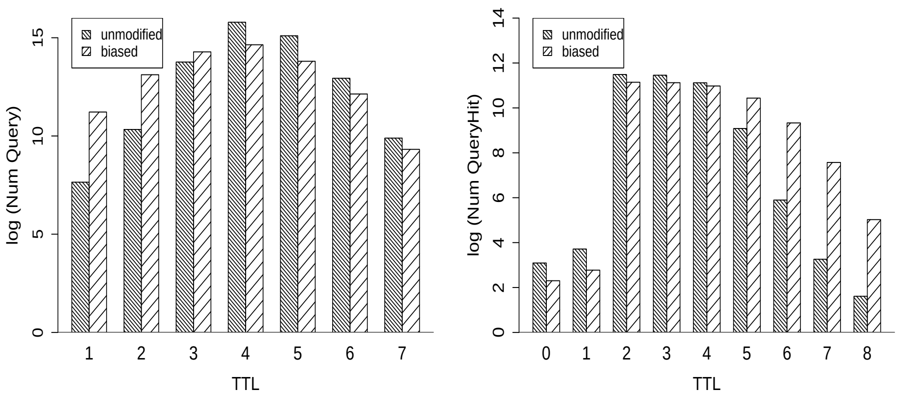
We observe that the number of query messages reduces from 12.2 million in the unmodified P2P network, to 5.6 million messages in the biased P2P network. This is a reduction of 54%. We also observe that consulting the oracle benefits the swarming pattern of query searches. From Table 6.1, we see that not only do the total number of flooded messages go down, rather, the reachability of queries at remote locations of the network increases as well. For example, the biased P2P network shows a much larger number of flooded query messages at TTL values of 1, 2 or 3, thus implying that queries are able to reach more P2P nodes at 5, 6 or 7 overlay hops from the originating node. This implies a more efficient swarming of search queries in the P2P network when nodes consult the oracle while choosing neighbours.

Table 6.2 shows the number of query response messages at different TTL values. We observe that the total number of query responses decreases only by 9.6%, a desirable feature as we naturally do not wish to obtain a lesser number of responses for queries when consulting the oracle. Figure 6.3 displays the logarithm of the number of search queries and their responses for both cases as a bar plot.

Table 6.2 Total number of query search response messages that are relayed in the network

TTL	Unmodified P2P	Biased P2P
8	5	152
7	26	1,941
6	363	11,284
5	8,789	34,031
4	67,381	58,488
3	94,392	67,651
2	97,305	69,003
1	41	16
0	22	10
Total	268,324	242,576

Figure 6.3 Logarithm of the number of query search messages (left) and query response messages (right) that are relayed in the network for unmodified P2P and biased P2P cases



We also measure the impact of using the oracle on the quantity of network discovery traffic, i.e., number of `Ping` and `Pong` messages relayed in the network, see Table 6.3. Once again, we note a reduction in network discovery traffic by 42%, which translates into improved scalability of the P2P system. 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 tables above. Thus information is propagated with lesser message hops, lower delays and reduced network overhead.

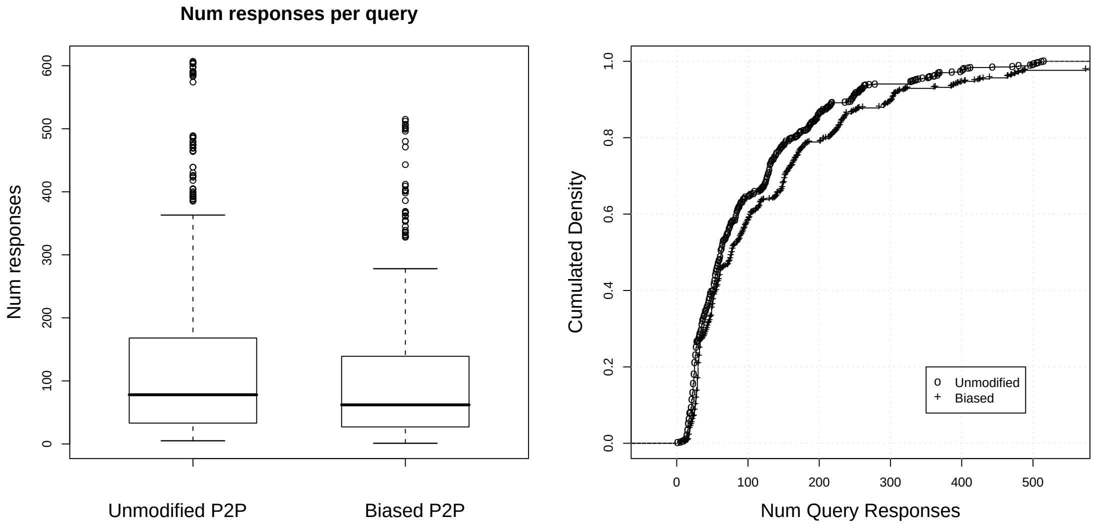
Table 6.3 Total amount of network discovery traffic that is relayed in the network

Message	Unmodified P2P	Biased P2P
Ping	15,323,903	8,986,961
Pong	153,021,689	89,491,751

While it is certainly desirable to improve the scalability of the P2P network, it is even more important to verify that consulting the oracle does not have a negative impact on the content search phase in a P2P network. In other words, it is important to analyze if the number of responses per search query are not adversely affected when P2P nodes bias their neighbourhood selection by consulting the ISP-hosted oracle.

Therefore we now compare the number of unsuccessful queries in unmodified and biased P2P networks. We find that while 24.78% queries do not find any content in the unmodified P2P network, 23.95% queries meet the same fate in the biased P2P network. Hence, we conclude that consulting the oracle does not affect the number of queries that do not find any content in the P2P network.

Figure 6.4 Box plot (left) and CDF plot (right) to compare the number of responses per search query, for unmodified P2P and biased P2P cases.



Finally, we compare the number of responses per query, for all satisfied queries in both the networks, and display it as a box plot [17] and cumulated density function (CDF) plot in Figure 6.4 on page 68. We see that the number of responses per query exhibit similar distributions for both unmodified as well as biased P2P networks. The mean number of responses is 127.7 for the unmodified network, against 102.3 responses for the biased network. The median number of responses is 78 and 62 respectively.

While the average number of responses per query drops slightly while consulting the oracle, we note that the number of queries which do not match any content does not increase. Besides, the swarming pattern of queries improves considerably, the observed query responses are located at lesser overlay hops, and the scalability of the P2P network improves considerably.

6.1.4 Localization of Content Exchange

The negotiation traffic traverses within the set of connected Gnutella nodes, but the actual content exchange happens outside the Gnutella network, using the standard HTTP protocol. When a Gnutella node gets multiple QueryHits for its search query, 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 neighbourhood 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 10.02% in case of biased case. We thus modify the neighbourhood 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 neighbourhood selection, during bootstrapping stage as well as file-exchange stage, leads to significant increase in localization of P2P traffic.

6.1.5 Summary

Using a packet-level simulation framework and a real P2P system, we have shown that the use of the oracle for neighbour selection in P2P systems helps both the P2P system as well as the ISP. The graph structural properties of biased P2P topologies under churn are comparable to random P2P topologies, a positive result. We also notice that the amount of P2P negotiation traffic reduces by about 50%, with no adverse effect on the success rate of the query search. The dense clustering of peerings within an AS ensures efficient swarming of messages in such a way that the reachability of messages to remote locations is not affected, rather, it slightly improves. Lastly, we also find that consulting the oracle again during content exchange leads to increased content being exchanged within the ISP network boundaries.

6.2 Analyzing the Effects of User Behaviour and Topologies

In the previous section, we have demonstrated the benefits of ISP-P2P collaboration using a single topology model and uniform distributions for session lengths and content availability. However, recent measurement studies [108, 131] have indicated that the session lengths of P2P users as well as their content availability is better modeled using heavy-tailed distributions.

In this section, we improve upon our previous work by using more realistic distributions for churn, content availability and query patterns. Also, we study the benefits of our approach on a number of different ISP topologies, as well as different distributions of P2P customers across the various ISPs. So far, we have only considered network locality (nodes within the AS, AS-hop distance) to choose neighbours. Now we will also consider the last-hop bandwidth of potential peers to select appropriate neighbours. We will use additional metrics to characterize the benefits for ISPs and P2P systems, namely, content download times, amount of content exchanged within AS boundaries, network locality of query responses, etc.

To summarize, in this section, we build upon our results from the previous section by

- extending the ISP's oracle to also consider last-hop bandwidth of P2P users while ranking possible neighbours

- studying the impact of different ISP/P2P topologies as well as a broad range of influential user behaviour characteristics, namely content availability, churn, and query patterns, on end-user and ISP experience. This task comprises three stages: (i) design of different ISP and P2P topologies, (ii) design of different user behavioural patterns, namely, content availability, churn, and query patterns, (iii) extensive experimental studies to determine the impact of different topologies and behavioural patterns on end-user experience, a task unaddressed as yet to the best of our knowledge.

The advantages of considering last-hop bandwidth of potential peers for neighbour selection have been discussed in Section 4.3. We introduce the network topology models for ISP/P2P in Section 6.2.1, followed by the user behaviour models in Section 6.2.2. We then present simulation results on variation in topology as well as variation in user behaviour in Section 6.2.3.

6.2.1 Topology Models

In order to study the effects of ISP topologies as well as the distribution of P2P customers across ISPs on P2P locality, we design 5 different AS 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 [59, 16]. As we add support for node bandwidths and more representative distributions for user behaviour in this section, the number of P2P nodes in the simulations becomes limited. We now briefly explain how we design each of the topologies.

Germany: The ISP topology map of Germany has been published in [45]. We take a subset of this map comprising the 12 biggest ISPs in Germany with all their inter-AS connections. The number of broadband (DSL) customers of each of these major ISPs is available at [115]. We thus distribute the 700 P2P nodes according to the proportion of DSL customers to these major ISPs.

USA: For USA, we model several regional providers, one at each of the 25 major US cities, and connect them with peering links using published measurement data from [59, 103]. We distribute the P2P nodes in the 25 ASes according to the ratio of the population of these cities.

World: To model the World topology, we design inter-AS connections as derived from BGP routing information in [68], and distribute P2P nodes based on results in [68, 57]. Each World topology has 1 level-1 AS, 5 level-2 ASes, and 10 level-3 ASes, hence resulting in a 16-AS network. Given these inter-AS connections, we distribute the P2P nodes among the ASes in three different ways. 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 (Germany, USA, and World), and for the World topology, we have 3 different ways of distributing P2P nodes within the ASes. This setup allows us to study the impact of different ISP topologies, as well as different distributions of P2P nodes within the ASes, on ISP/P2P performance.

Network Characteristics: Bearing in mind the memory limitations of a packet-level simulator, and that it is fundamentally difficult to simulate the Internet [26], we model the topologies within SSFNet as follows. Similar to Section 6.1, 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. In this section, we model the last-hop bandwidths of P2P users as follows. The nodes have network interfaces representing typical last-hop DSL and cable modem bandwidths, ranging from 1 Mbps to 16 Mbps. The Germany topology uses typical DSL speeds, while the USA topology uses typical cable modem speeds, due to the prevalence of the two technologies in the respective countries. Level-1 and level-2 ASes have a larger proportion of higher bandwidth customers than the level-3 ASes [93, 59, 115, 57]. For example, in a level-1 AS, 80% of the P2P nodes are assigned 10 and 16 Mbps bandwidths, while in a level-3 AS, 60% of the P2P nodes are assigned $1-4$ Mbps bandwidths. 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 $18-20$ ms [59, 132] for the World topology. The inter-AS delays for the Germany and USA topologies are kept slightly lesser.

6.2.2 User Behaviour Models

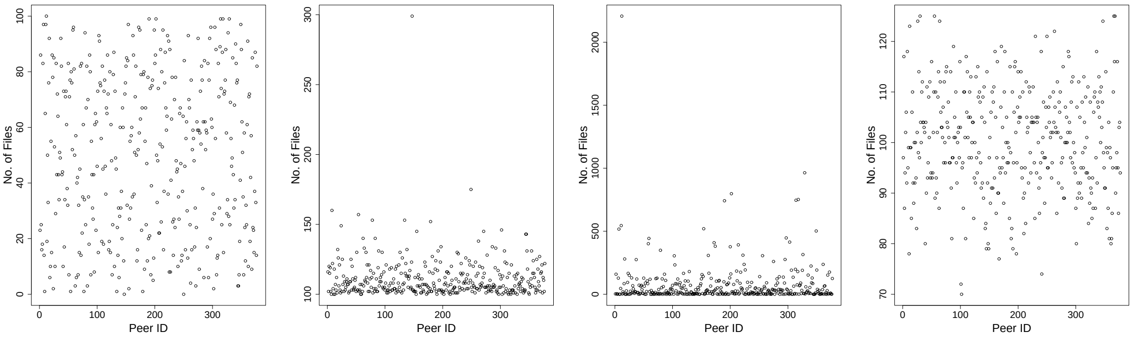
While we have implemented a specific protocol in SSFNet, our goal is to perform experiments that represent a large section of P2P systems in use today. Studies [108, 42] have shown that user behaviour is largely invariant across P2P systems, both structured and unstructured. This means that factors like session lengths, content availability (free-riding), query patterns and search strings are similar across different P2P systems.

We note that user behavioural patterns are in constant transition, although the broad characteristics across different systems are comparable. Hence, we use different distributions to simulate the behavioural patterns, some very close to observed behaviour, 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 behaviour within the limitations of a simulation environment.

Content availability The presence of a large number of free-riders has been confirmed by extensive measurement studies [93, 25, 131, 53]. 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 as shown in Figure 6.5, where the x-axis denotes the P2P nodes, and the y-axis denotes the number of files shared by the peers. While *Weibull case (scale*=42, *shape*=0.5*)* and *Pareto case (k*=100, *alpha*=10*)* represent realistic behaviour (i.e., a 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 [108, 42, 114]. 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 [42]. Hence, we represent online session

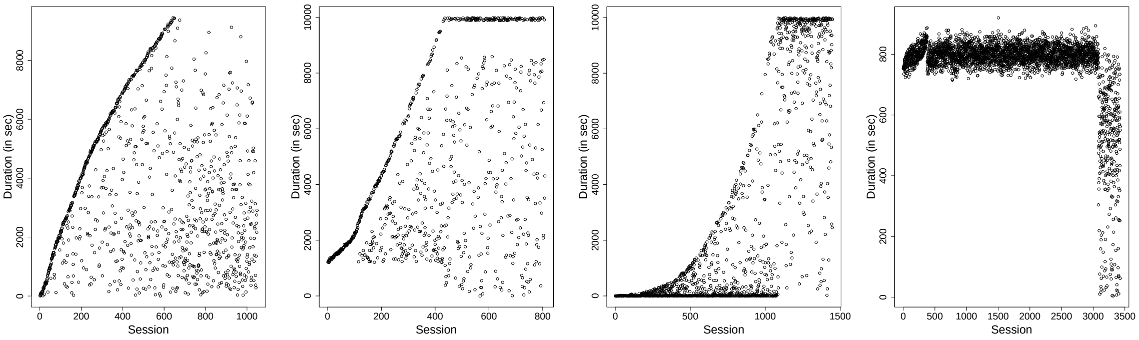
Figure 6.5 Number of files hosted by each peer using different distributions, where x-axis denotes peers and y-axis denotes number of files



(a) Uniform (min=0, max=100)
(b) Pareto (k=100, alpha=10)
(c) Weibull (scale=42, shape=0.5)
(d) Poisson (mean=100)

lengths using different distributions as shown in Figure 6.6, where x-axis denotes the online sessions and the y-axis denotes the duration of the sessions in seconds. The *Pareto case (k*=600, *alpha*=0.5*)* and *Weibull case (scale*=600, *shape*=0.2*)* represent realistic behaviour, *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.

Figure 6.6 Online durations for each P2P session using different distributions, where x-axis denotes sessions and y-axis denotes duration in seconds



(a) Uniform (min=1, max=10000)
(b) Pareto (k=1200, alpha=0.5)
(c) Weibull (scale=1200, shape=0.2)
(d) Poisson (mean=800)

Query strings Most P2P systems are characterized by query search phrases of two kinds [31]: 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 [53, 31]. 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.

6.2.3 Results

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:

- to study the effects of *various topologies* on the above metrics with realistic user behaviour, comparing oracle-aided P2P with unmodified P2P
- to measure the effects of *various user behaviour 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 size 512KB (the typical file piece size used in popular P2P systems) and is exchanged directly between the peers using HTTP. The oracle sorts the candidate list of neighbours based on the following algorithm: (i) identify the nodes within its AS, and sort them using last-hop bandwidth, (ii) for nodes not within its AS, sort them using AS-hop distance.

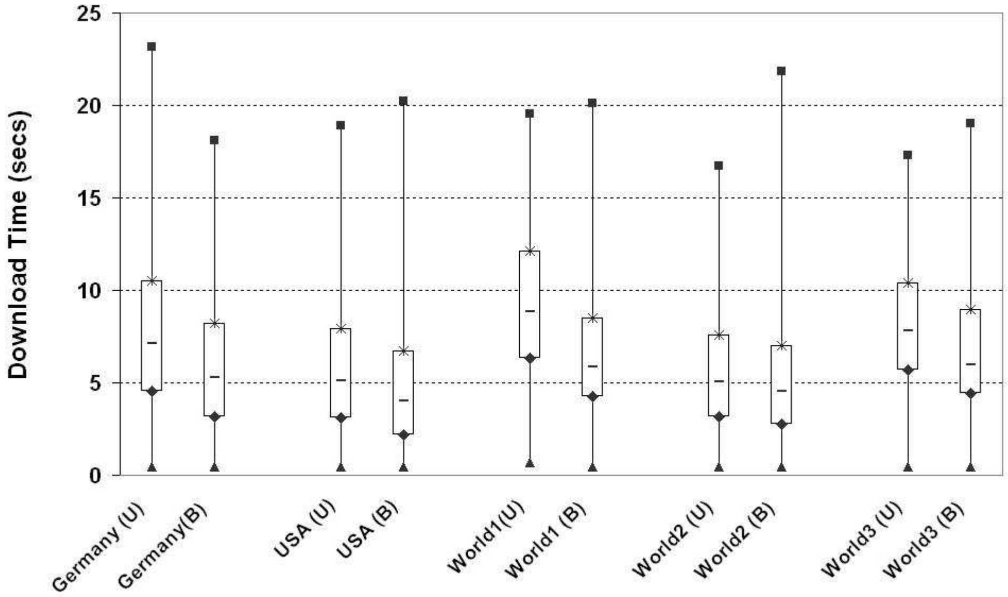
Variation in Topology

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 neighbours, 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 neighbours, hence setting up a localized P2P topology that is correlated with the Internet AS topology. 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. We model content availability and online session lengths by Weibull distributions (realistic behaviour). The results for all 5 topologies are discussed below.

Content exchange: The most important metric for the end-user is the time taken to download content. As shown in Figure 6.7 with the help of a box plot [17], 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 neighbours. We also notice that changing the inter-AS delays does not have a significant effect on file download times. Moreover, additional simulations confirm that exchanging content with a high-bandwidth peer in another AS is consistently faster than a low-bandwidth peer in the same AS. This confirms that file download times are dominated by last-hop bandwidths [23].

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 6.8. This can result in direct cost savings for the ISP, estimated to be in the order of $1 billion world-wide [18, 128]. We note that the improvements for the biased P2P system are more pronounced in World1 and World3, as the peers are more evenly distributed across the ASes in these topologies.

Figure 6.7 Comparison of file download times using a box plot for unmodified(U) and biased(B) P2P neighbour selection across 10K file transfers for 5 topologies



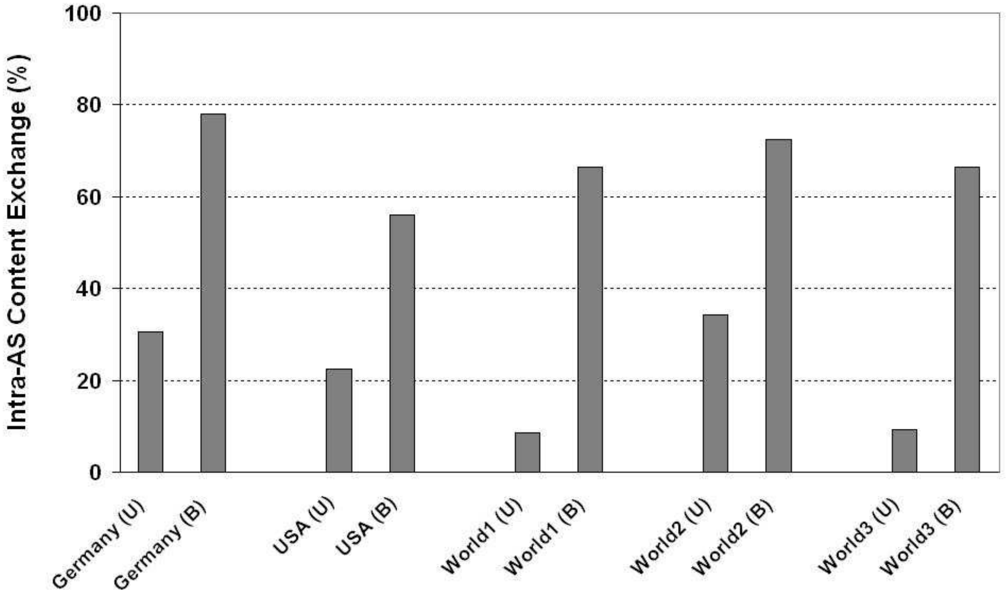
We conclude that while the ISP benefits from AS-distance based neighbour selection, the benefits to P2P users accrue mainly from last-hop bandwidth based selection, thus underscoring the need for both metrics in the oracle.

Query Search: Figure 6.9(a) 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 due to a more efficient swarming of the queries (and their responses) within the localized P2P topology, see Section 6.1.3. A closer examination reveals that for the same number of unique queries, the negotiation traffic in the overlay, which is emanating from flooding and forwarding of queries and their responses, decreases by about 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 is 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 the unmodified as well as the biased P2P system. 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 the queries find such content. Besides, the query responses more often come from peers that are located within the same AS as the originating query, see Figure 6.9(b). 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 which are typical of random

Figure 6.8 Comparison of amount of intra-AS content exchange for unmodified(U) and biased(B) P2P neighbour selection across 10K file transfers for 5 topologies



overlays, namely, small node degree, small graph diameter, small mean path length and connectedness, even under heavy node churn. The average node degree, shown in Figure 6.9(c), changes only slightly, from 18 for unmodified P2P to 16 for biased P2P. The graph diameter is found to remain constant at 6 – 7 hops, and the mean overlay path length between all pairs of overlay nodes increases only nominally from 2.5 hops for unmodified P2P to 3.3 hops for biased P2P, see Figure 6.9(d). In other words, the graph structural properties of the overlay are not affected adversely when consulting the oracle even under churn. 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.

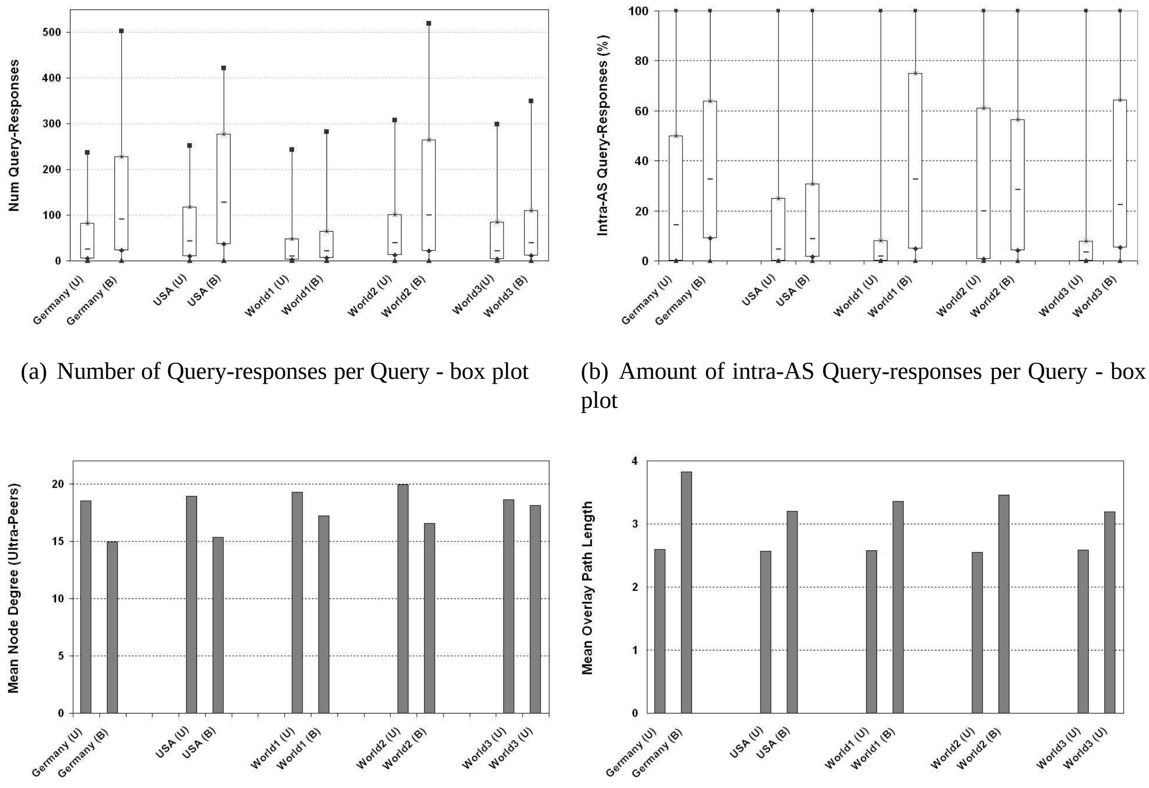
Variation in User Behaviour

Now that the benefits of ISP-aided P2P locality have been established across various topology models, we analyze the effects of user behaviour 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 a small number of nodes serve most of the files in the P2P network and go offline, to observe their impact on network performance. In other words, we determine if biased P2P maintains its benefits across different scenarios.

As explained in Section 6.2.2, we model content availability as well as session lengths as Uniform, Pareto, Weibull and Poisson distributions, thus giving us 16 possible combinations for the two characteristics. Hence, we run 16 different experiments for the biased P2P case for each topology. In this section, 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 all the 16 combinations of content availability and online session lengths, the

Figure 6.9 Query search and graph properties metrics comparing unmodified(U) and biased(B) P2P neighbour selection across 10K queries and 10K file transfers, for 5 topologies



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

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

(c) Mean node degree of overlay nodes - bar plot

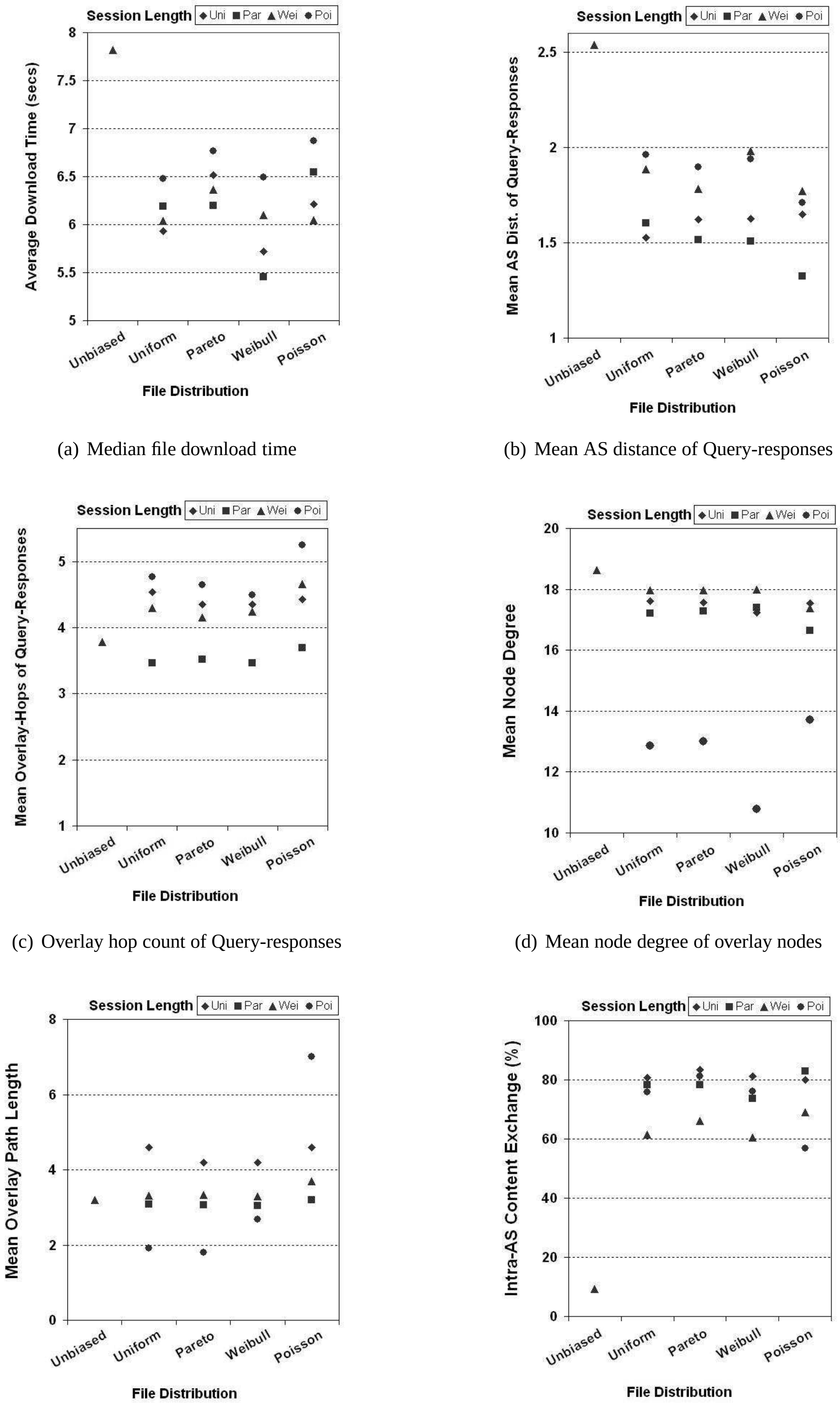
(d) Mean overlay path length - bar plot

biased P2P topologies maintain their benefits for the P2P users as well as the ISPs. Consider the median file download time in Figure 6.10(a). Even though its value varies from 5.5 – 7 seconds for biased P2P, it still remains below 7.8 seconds for unmodified P2P. The results for the mean AS distance of query responses are similar. In Figure 6.10(b), we witness a noticeable reduction in the number of AS hops between peers that send a query and peers that satisfy the query. Also, the mean overlay hop count of query responses in biased P2P cases remains comparable to that of unbiased P2P, as shown in Figure 6.10(c). This result has positive ramifications for mobile applications, where an increase in the overlay hop count can lead to performance degradations due to processing overhead at each additional node encountered in the path. The success rate of queries remains the same, while the number of responses to queries remains consistently higher than that with the unmodified P2P system.

With regards to the graph properties, the node degree in Figure 6.10(d) remains largely unchanged, except for the case of Poisson session lengths. The results for the mean overlay path length between all pairs of nodes are also similar, see Figure 6.10(e). Although the graph properties are negatively affected by the Poisson session length distribution, we note that this distribution is not observed in real P2P systems, hence we can ignore this case.

Analyzing the benefits to the ISPs, we notice in Figure 6.10(f) that the amount of exchanged content that remains within the ISP network boundaries across all the tested scenarios ranges from

Figure 6.10 Effect of user behaviour (content availability and session length) patterns on end-user experience for World3 topology. X-axis denotes file distribution models, and symbols denote online session length models: Uniform, Pareto, Weibull and Poisson.



(a) Median file download time

(b) Mean AS distance of Query-responses

(c) Overlay hop count of Query-responses

(d) Mean node degree of overlay nodes

(e) Mean Overlay path length

(f) Amount of intra-AS file exchange

60 – 80%, significantly more than the 10% value observed in the case of unmodified P2P. This convincingly shows that ISP-aided P2P neighbour selection maintains its benefits across different user behaviour 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 maintains its nice graph structural properties, while ISPs as well as P2P systems benefit from co-operation.

Visualization of Overlay Topology

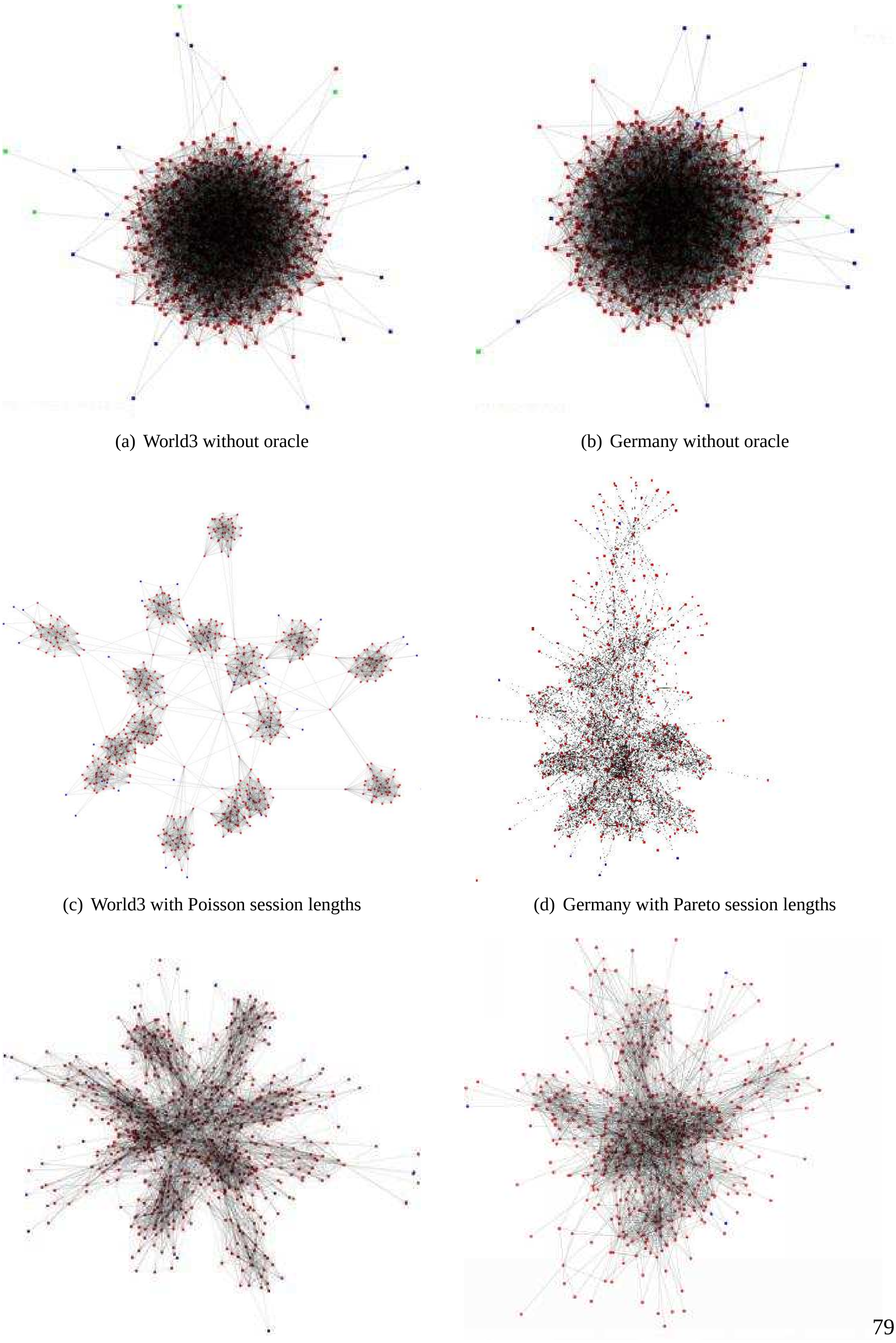
We show 6 different snapshots of the overlay topology to give a better intuition on our results. Figure 6.11 on page 79 shows the World3 and Germany topologies under different scenarios, namely, without oracle, with oracle using Poisson session lengths, and with oracle using heavy-tailed session lengths. The visualizations have been produced with the yWorks [127] software using the same technique used for Figure 6.1.

We can easily see that the overlay topology without the oracle looks the same for different AS structures, as it is not correlated with the Internet topology and there is no noticeable pattern. Our results in the previous section have already shown that Poisson session lengths are the most punishing on the overlay structural properties. However, we see that despite being non-robust, there *is* a correlation of overlay topology with the Internet topology in the case of World3 topology with Poisson session lengths, see Figure 6.11(c). More importantly, the graph remains connected. We once again note that the Poisson session length is not observed in real P2P systems, we use it here only to make worst-case comparison study. With Weibull session lengths (which is the observed behaviour in most P2P systems), the overlay topology is nicely correlated with the Internet AS topology. A large number of peerings stay within the AS boundaries, forming dense sub-graphs local to the ASes, while a good number of peerings cross the AS boundaries which keeps the overlay structure robust against churn.

6.2.4 Summary

In this section, we design representative ISP/P2P topology models and user behaviour characteristics in a simulation framework, and study their impact on ISP-aided bandwidth-based localized neighbour selection for P2P users. Through extensive experiments, we show that both P2P users and ISPs benefit from collaboration, measured in terms of improved content download times, increased network locality of query responses and desired content, and overall reduction in P2P traffic. While ISPs benefit from a simple AS distance-based neighbour selection, P2P users benefit mainly by peering with nodes possessing higher last-hop bandwidth links. The advantages of ISP-P2P collaboration hold across different ISP/P2P topologies under a broad range of user behaviour scenarios.

Figure 6.11 Visualization of overlay topology under different scenarios



(a) World3 without oracle

(b) Germany without oracle

(c) World3 with Poisson session lengths

(d) Germany with Pareto session lengths

(e) World3 with Weibull session lengths

(f) Germany with Weibull session lengths

7 Multiple-ISP Collaboration

We have already shown that by consulting the oracle, P2P users are able to pick appropriate neighbours, both for forming an efficient overlay topology as well as for downloading content, in a way that both ISPs and P2P systems benefit. We now extend the oracle concept to propose collaboration between multiple ISPs, so that P2P and other applications can get estimates of the path properties to potential neighbours/servers, both within and outside their ISPs. In this chapter, we discuss how multiple-ISP collaboration can be achieved, and present simulation results to show its benefits. We also explain how this concept can be used to design a global coordinate system, and discuss how our proposed coordinate system differs from existing ones.

7.1 Proposal

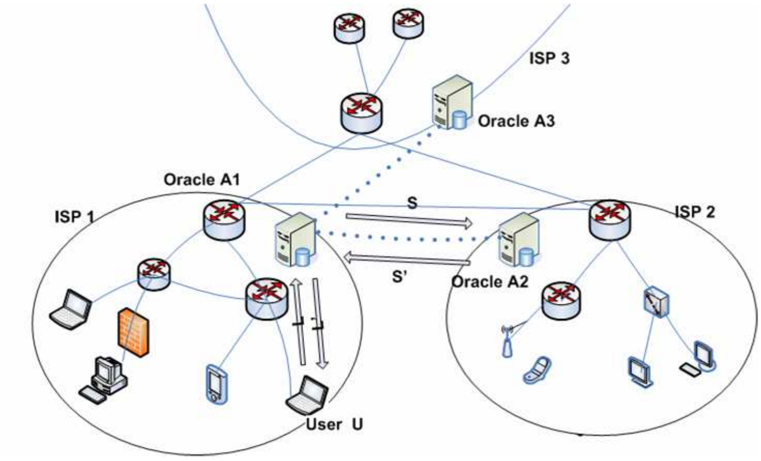
In the previous chapters, we have introduced and evaluated the concept that each ISP hosts an oracle server. The oracle has access to an up-to-date map of the ISP network. For example, it knows the bandwidths of the links within the network, the connectivity characteristics of the users, and the estimated RTT. On getting a list of potential neighbours from a P2P user, the oracle uses this network information to sort the nodes within its network. So far, nodes that are not belonging to its own AS are only sorted by AS distance as the ISPs do not have information about the internal network of other ISPs. We now propose that oracles from different ISPs collaborate with each other to exchange summaries of their network information, which will enable them to sort even those nodes that are outside their network.

The motivation for this is that an ISP knows not only its own network, but also its routing policies to neighbouring ISPs. Furthermore, each ISP has information about which of its neighbouring ISPs are customer, peer or provider ASes [72]. As an ISP routes traffic to and receives traffic from other ASes, it also has BGP [40] path information to other ASes, and a fairly good estimate about the IP address ranges of their customers. Besides, an ISP is also aware of the capacities and other characteristics of inter-domain backbone links, at least in its neighbourhood. We propose to use this vast information available at each ISP to extend the oracle service, such that the oracle servers from different ISPs can collaborate with each other, see Figure 7.1.

Collaboration between ISPs

When an oracle receives a list of candidate IP addresses from a user within its network, it can rank the nodes within its network using its network information, as described in Chapter 4. For nodes that do not belong to its own network, it first segregates them according to their parent ASes. Nodes that belong to ASes in its immediate neighbourhood can be further classified as belonging to customers, peers or provider ISPs. As each ISP has to pay for traffic going to upstream provider networks, it has an interest in preferring nodes from customer and peer links, depending on its individual AS-level routing policy. Hence, for instance, customer and peer-ISP users can be higher ranked than provider ISP users. If the queried IP addresses do not belong to ISPs in its immediate neighbourhood, an ISP

Figure 7.1 Communication between oracles of different ISPs to achieve multiple-ISP collaboration



can use AS-hop distance (the number of AS-hops on the chosen BGP route for the IP address), or BGP routing policy [40] (preferred AS paths, point-of-exit for traffic, etc.) for ranking nodes outside its network. For instance, as discussed in Chapter 4, nodes belonging to ASes with lesser AS-hop counts will be ranked higher as compared to nodes belonging to ASes farther away. The level of granularity for ranking the list of nodes can be decided by each ISP independently.

To further fine-tune the list of nodes, the oracle contacts the oracle server from the neighbouring ISP, and sends it the list of users that it wants to rank. The neighbouring ISP oracle can use its own network information to rank the nodes that belong to its network, and return the sorted list back to the querying oracle. The oracle can then combine this ranked list of nodes with its own network information, and thus be able to estimate the path properties to potential neighbours, both within and outside its network. We illustrate this with the help of an example below.

Example

Consider the scenario in Figure 7.1. ISP1 is connected by a peering link to ISP2, and by an upstream link to ISP3. When oracle A1 gets a list L of candidate IPs from a user U, it sorts the list of IPs within ISP1 on its own, using a semi-static database containing information about its network. As ISP1 prefers to route traffic to ISP2 instead of ISP3, it estimates the subset S of the list of IPs which belong to ISP2, and sends it to the corresponding oracle server A2. The oracle A2, on receiving this list S of IPs which belongs to its network, can easily rank this list based on metrics like available link capacity, estimated delay, geographical location, etc., as described in Chapter 4. It then sends the ranked list S' back to A1, which A1 incorporates into its final ranked list L' to be returned to the user U. Depending on the level of fine-tuning desired, A1 can even contact multiple neighbouring oracles, e.g., A3.

Thus, each ISP, using a combination of ISP-P2P collaboration (for the individual network), and ISP-ISP collaboration (for multiple networks) can provide estimates of path properties to potential neighbours both within and outside its network. This allows a P2P user to pick the “best” neighbour in terms of network connectivity and ISP routing policy, even if the potential neighbour is not within its own network. This has the added advantage that the routing policies of the ISPs are also taken into account when forming neighbourhoods. Note, that each ISP is free to rank the list of IPs according to its own criteria and has full control over how much information it wishes to reveal.

7.2 Global Coordinate System

In this section, we demonstrate how the oracle concept and the collaboration between oracle servers from multiple ISPs can be leveraged to design a global coordinate system. A coordinate system maps the IP address of a peer into an n-dimensional coordinate space. The coordinate distance between two nodes in that space reflects the network distance between them in the Internet, which is typically defined as the RTT propagation and transmission latency.

Coordinate systems have been proposed previously [28, 71, 23, 60, 99], however, we argue that the way such coordinate systems are built are not very efficient and suitable. Existing coordinate systems measure the RTT and map the distances into a low dimensional Euclidean system like the Cartesian coordinates [28], or a non-Euclidean one, e.g., hyperbolic, spherical, or toroidal [99]. Unfortunately, the actively measured RTTs are far from accurate and may change quickly over time [124]. Moreover, up to now, they cannot offer available bandwidth or capacity estimates.

We therefore propose an alternate way for building a coordinate system: namely again by collaboration among ISPs. We have previously discussed that ISPs have detailed information about the connectivity of peers that are located within their domain: their bandwidth, their usage patterns, etc. Moreover, ISPs also decide and implement their routing policy, and are thus aware of the routing paths within their network and to other ISPs. By using this already available ISP information and exchanging summaries of it among ISPs, a coordinate system can therefore be built that does not require active measurements. We argue that this coordinate system is more accurate, is capable of addressing additional metrics and can provide the information quicker to a querying node than current coordinate systems.

7.2.1 Oracle as a Coordinate System for a Single ISP

We describe our approach for building a coordinate system for a single ISP, based on collaboration between an ISP and P2P or other user applications running within the ISP network. The basic task of a coordinate system is: given two IP addresses return an estimate of the network distance (usually defined in terms of RTT) between them. Our main insight is that ISPs either have or gather the most relevant as well as accurate information about the connectivity of hosts that are located in the ISP's domain, where the term connectivity includes information such as physical bandwidth to the last hop (modem, DSL, VDSL, etc.), latency statistics, geographical location and customer service class including different quality classifications, such as gold, silver, or normal customer. Moreover, each ISP decides the routing policy for transmitting traffic within its network, using intra-domain routing protocols like OSPF, IS-IS, and RIP. In other words, an ISP is already in possession of the information that other coordinate systems have to infer including link capacity, service classes, available bandwidth, estimated delay/RTT, etc. Hence, given two IP addresses *within* its network, an ISP can determine or estimate a summary of the basic path characteristics of the network path between them.

Recall the example demonstrating the use of the oracle in Section 4.5 on page 38. We can see that the proposed oracle service already provides an abstraction of a coordinate system. In the terminology of current coordinate systems, each ISP - represented by its oracle server - is the pendant to a landmark. However, instead of measuring distances between different landmarks and between landmarks and peers as is the case in existing coordinate systems [28, 71, 23, 60, 99], each ISP's oracle stores connectivity information to build a coordinate system. Compared to existing coordinate systems, it has a number of advantages. First, the knowledge of the oracle goes far beyond knowing the distance between peers in terms of RTT only. With its knowledge about link capacities, available bandwidth, geographical location, etc., it can also answer questions such as "which peer has the best bandwidth to me?" Even combinations of multiple metrics are possible.

7.2.2 A Global Coordinate System through Multiple-ISP Collaboration

We have seen that given two IP addresses within its network, an ISP's oracle can serve as a coordinate system and return an estimate of the path properties between the two IPs that can take into account multiple metrics like bandwidth, delay, geographical or topological proximity, etc. The same concept can be extended to also return an estimate of the path properties between two IP addresses that are not within the same ISP network. In such a case, oracle servers from different ISPs can collaborate with each other, as explained in Section 7.1.

When a user U sends a request to his ISP's oracle A to find the path property to another IP address U', the oracle A finds the parent ISP of U', and contacts its oracle server A'. The oracle A', being aware of its own network, can easily return a classification of the path property to U'. In this way, ISPs can collaborate with each other to estimate the path properties between any two IP addresses in the Internet.

We believe that a P2P application can use the coordinate system in two ways. First, it may use the system to get an estimate of the network path properties between any two nodes in the system, e.g., low, medium, or high bandwidth. Second, a peer may submit its own address and a list of potential neighbour peers, and ask the coordinate system to sort the list in increasing order of their distance to itself. Using these functions, an overlay topology of a P2P system can be built that reflects the real distances in the physical topology. In particular, nodes should only be neighbours in a P2P system if their distance in the coordinate system is small.

7.3 Related Work in Coordinate Systems

In recent years, research on Internet coordinate systems has received much attention. Most of the coordinate systems proposed so far including [28, 71, 23, 60, 99] rely on a set of landmarks or on peer-to-peer technology. The current set of coordinate systems mainly attempt to map hosts into synthetic coordinates in some coordinate space (Euclidean [28], spherical [99], hyperbolic [99] being some examples) such that the distance between two hosts' synthetic coordinates reflects or estimates the actual round-trip-time (RTT) between them in the real Internet. While this approach may serve well for some applications like Web servers or content distribution systems (CDNs) which do not experience high churn, its leads to performance degradation in the face of newer brands of file sharing and CDNs which are characterized by high user churn [108], pollution in content, and malicious activity on the part of the users.

Coordinate systems are vulnerable to malicious users who lie about their locations [49, 48]. It has also been shown that coordinate systems are several orders of magnitude slower than direct ping measurements made by individual peers, often taking several tens of seconds or even minutes to converge [49, 48]. This is clearly unacceptable given the high amount of churn in P2P systems [108], and the small online durations of the participating peers (see Chapter 3). Lua et.al. [64] have shown that the accuracy metrics used by these coordinate systems are not accurate enough. More recently, Ledlie et.al. [56] have shown that while the performance of the coordinate systems reaches expected levels on Planetlab nodes and simulation environments, the performance degrades significantly when deployed in the real Internet.

Recent studies [124] have also shown the limitations of using RTT as a metric for coordinate systems. The existing coordinate systems predict network distance as a sum of RTT propagation and transmission delay, which they assume to be a fairly stable characteristic between Internet hosts. However, RTT is dependant on network load, which is heavily influenced by factors like churn and bursts in user activity in P2P and CDN systems. As such systems are dominated by peers who have very short uptimes [108], assuming RTT to be a stable metric is not a sound assumption. Besides, small RTT does not always correspond to peers being well connected in terms of bandwidth [124].

Discussion

Compared to the above systems, our proposed coordinate system does away entirely with the complex mathematical computation process of mapping a node's location in the Internet to a point in the mathematical coordinate space. As the node location, its connection information, and the network routing policy is known to the oracle, the need for Internet measurements [88] and parameter estimations is heavily reduced, thus reducing network overhead and increasing scalability. Also, our system is not based solely on RTT. Rather, the network distance between two nodes reflects not only the RTT propagation and transmission delay, but also factors like:

- path capacity and available bandwidth
- better paths which may or may not correspond to least RTT, but do offer better bandwidth and lower packet loss rates
- respect for AS relationships, BGP-based policy routing and other routing metrics like point-of-exit of AS, next-hop AS, multi-exit discriminator (MED), etc.

We have already shown the oracle system is resistant to churn in P2P applications. As the oracle does not need to ask the nodes for their location but already knows them authoritatively, the susceptibility of the proposed coordinate system to malicious nodes lying about their location is also reduced, a major improvement over existing coordinate systems. Moreover ISPs can tailor their answers to regain control over their traffic.

7.4 Experiments

We now present some experimental results to evaluate the performance benefits when the oracle servers from multiple ISPs collaborate with each other to recommend neighbours to P2P users. We concentrate on the content exchange phase of a generalized P2P file sharing system. The experiments are performed using the PeerSim [76] P2P simulator, which has been introduced in Section 2.7.

7.4.1 Network Model

We use a topology with $116,000$ P2P nodes distributed in 200 ASes. In this topology model, 10 level-1 ASes which are completely connected with each other. Each level-1 AS is connected to 5 level-2 ASes, which in turn are each connected to 2 – 3 level-3 ASes. We model 10 intra-AS routers in level-1 ASes, 4 intra-AS routers in level-2 ASes, and 2 intra-AS routers in level-3 ASes. Besides, each AS has 1 – 2 routers for inter-AS connections, thus giving a topology with 840 routers spread in 200 ASes. We note that using a large topology with multiple routers per AS allows us to study the effect of using router-hop count as a factor in the oracle node-ranking algorithm. Each intra-AS router is connected to 200 P2P nodes. The nodes have a last-hop bandwidth ranging from 1 – 16 Mbps, with top-level ASes having a larger proportion of high-bandwidth peers, see Section 6.2.1. A sub-network of one level-1 AS with all its level-2 and level-3 AS connections, along with the corresponding link delays is shown in Figure 7.2. We model a node-to-oracle communication with a delay of 20 ms, to account for the processing time of the oracle.

7.4.2 Realizing Multiple-ISP Collaboration

When a P2P node `n` goes online, it searches for a specific content and finds it available at a set of nodes `L`. When not using the oracle, the node `n` selects a node randomly from the list `L`, and initiates the file exchange process with it. When consulting the oracle, the node `n` sends the list `L` to the oracle `A` of his ISP. The list `L` is sorted by the oracle `A` using the following algorithm, based on the discussions in Section 4.4 and Section 7.1.

Algorithm

The oracle `A` identifies the nodes in `L` that belong to its ISP, and sorts them using last-hop bandwidth. The same-bandwidth nodes are further sorted using router-hop distance from the querying node `n`

For nodes not within its ISP, `A` segregates them according to their ISPs into sublists. It then sends the various sublists `S` of nodes to their respective ISP's oracle.

The respective ISP oracles sort their sublist `S` using the last-hop bandwidth of the nodes, and send the sorted list `S'` back to the oracle `A`. The oracle `A` combines all the sorted lists into a final list `L'`, with the peers in its own AS at the top of the list, followed by nodes at AS-hop distance 1, followed by nodes at AS-hop distance 2, and so on. In this way, nodes outside the AS are sorted by AS-hop distance, which are further sorted by their last-hop bandwidths.

The final sorted list is sent back to the querying node `n`. The node `n` connects to the first member of the list `L'`. If the first node is offline, it tries the next member, and so on until it finds a peer from whom it can download the desired content.

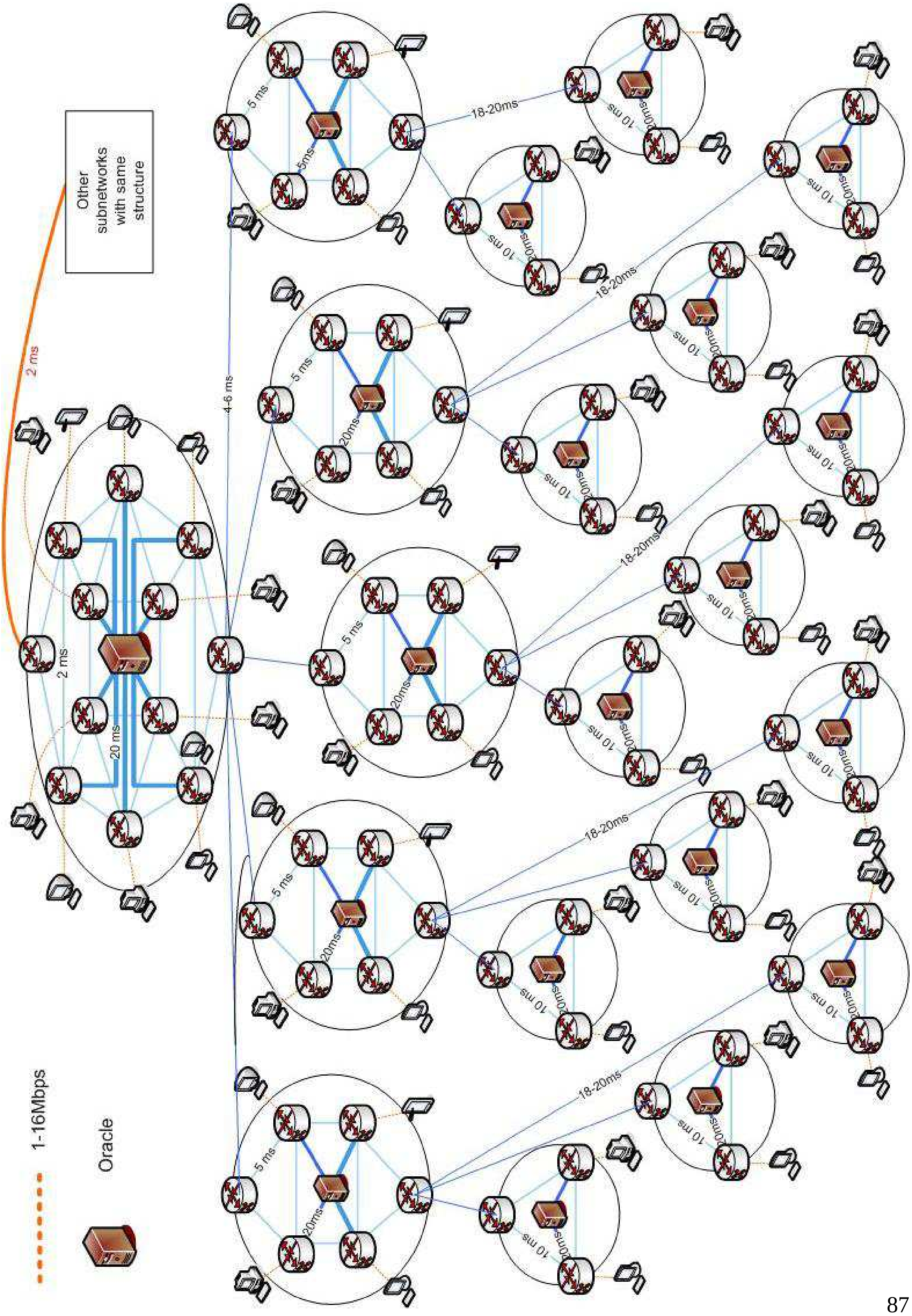
7.4.3 Results

Given this simulation setup, we run 3 sets of experiments:

- P2P nodes do not consult the oracle and choose neighbours randomly
- P2P nodes consult the oracle, which sorts the list of possible neighbours in collaboration with other ISPs
- P2P nodes choose the peer with the highest last-hop bandwidth for content exchange

The last case is used to compare multiple-ISP collaboration with a bandwidth-based neighbour selection done by the P2P users themselves without taking the ISP or network topology into account. In other words, this case simulates a P2P protocol that does not collaborate with the ISP, and picks

Figure 7.2 Network model showing a sub-network of one level-1 AS with all its level-2 and level-3 AS connections, along with intra-AS router and node topologies and link delays. In PeerSim simulations, ten such subnetworks are connected through level-1 AS interconnections.



up a neighbour by reverse-engineering the last-hop bandwidth of possible neigbhours, and picking the best one irrespective of its parent AS.

All the results are based on $100,000$ downloads of files, each of which is 1 MB in size. The online/offline behaviour of P2P nodes, as well as content availability at each node is modeled using Weibull distributions as explained in Section 6.2.2. To compare the three cases, we use the following metrics. For each pair of P2P nodes exchanging content with each other, we calculate the number of AS-hops, number of router-hops, and the router-to-router latency between them. We also calculate the amount of content exchanged that remains within the ISP, and the amount of time taken to download each 1 MB file. The bandwidth distribution of neighbours is also calculated. We plot these metrics across $100,000$ file download instances in Figure 7.3 on page 90. Running another set of experiments with a different distribution of P2P nodes within the ASes yields similar results.

In Figure 7.3(a) we plot the AS distance between P2P nodes that exchange content with each other. We can immediately see that the AS distance reduces significantly with multiple-ISP collaboration, implying that most of the content exchange takes place within the ISP network boundaries. Figure 7.3(b) shows that this is indeed the case. With multiple-ISP collaboration, 60% of the content exchange takes place within the ISP boundaries, as compared to around 10% for the case when P2P nodes choose neighbours randomly or pick neighbours having the highest last-hop bandwidth.

As the oracle uses the router-hop distance as one of the factors to choose “good” neighbours, we plot the router-hop distance between P2P neighbours in Figure 7.3(c). We see that with multiple-ISP collaboration, the router-hop count between P2P neighbours reduces significantly. This implies that even within the ISP network, a large amount of content traverses lesser router hops, hence using lesser network resources as compared to random or bandwidth-based P2P neighbour selection.

Figure 7.3(d) shows the distribution of last-hop bandwidths of peers from which content is downloaded. We observe that using multiple-ISP collaboration helps to pick more peers with higher bandwidths, i.e., 50%, as compared to the random case (20%). However, bandwidth-based neighbour selection results in all (100%) of the peers having the highest last-hop bandwidth. While this is a desirable result for P2P users, this does not lead to a balanced distribution of content among P2P nodes, as all the content is served exclusively by 16 Mbit peers. Also, network resource utilization is not optimal, as is evident from AS- and router-hop count between neighbours, see Figure 7.3(a) and (c), as well as the amount of content that remains within network boundaries, see Figure 7.3(b). Considering these metrics, we see that multiple-ISP collaboration gives more balanced results, which are beneficial to both ISPs as well as P2P users, and not to only one of them.

Figure 7.3(e) further emphasizes this observation, where we plot the estimated download times for the $100,000$ files that are exchanged between the peers. Not only does multiple-ISP collaboration results in faster downloads as compared to random neighbour selection, it is also very close in performance to the bandwidth-based neighbour selection. The reason for this is that even though bandwidth-based P2P nodes pick all 16 Mbit neighbours, many of these neighbours are at a greater AS- and router-hop count from the querying peer, and thus experience more link delays as compared to the oracle-recommended neighbours.

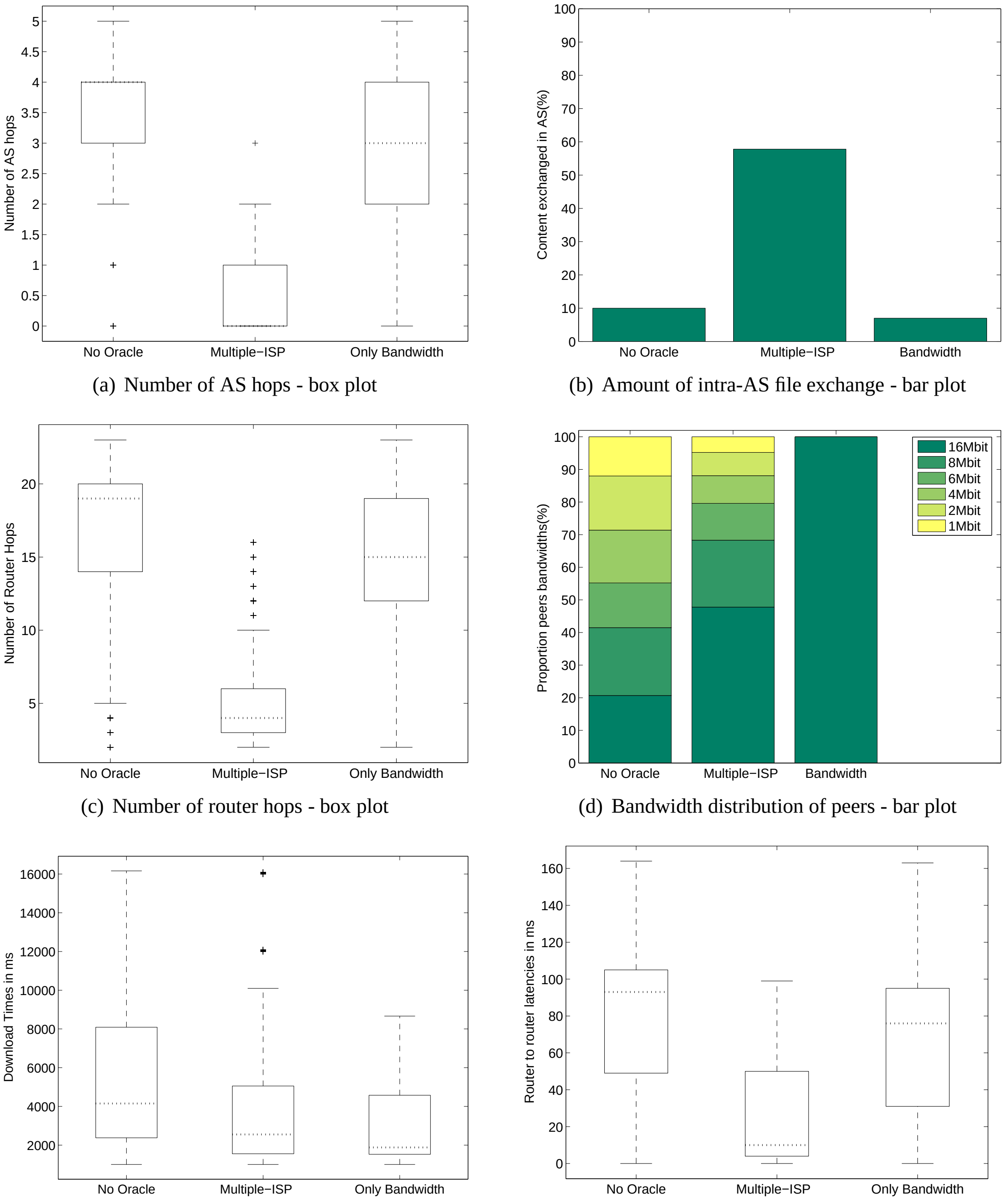
In Figure 7.3(f), we plot the router-to-router latency between P2P neighbours. In this case, we find the PoP-level router of both the P2P nodes of a particular P2P link, and calculate the latency of the network path between the two routers. The plots show that multiple-ISP collaboration reduces the network latency significantly as compared to both random P2P as well as bandwidth-based P2P neighbour selection.

We thus show that multiple-ISP collaboration benefits both ISPs as well as P2P applications. The ISPs are able to keep a large amount of traffic local to their networks, and are able to use their network resources more efficiently. The P2P users get near-optimal performance in terms of download times and latency. Also, multiple-ISP collaboration achieves balanced distribution of content among the P2P nodes, in the sense that the high-bandwidth peers are not indirectly penalized by having to serve too much content to other peers.

7.5 Summary

We have proposed and evaluated the concept of collaboration between user applications and ISPs on the one hand, and between different ISPs on the other hand. Using large-scale simulations, we have shown that multiple-ISP collaboration benefits both P2P users as well as ISPs. While P2P users get near optimal download performance, ISPs save immense costs and are able to use their network resources in an efficient and balanced manner. We have also demonstrated how the multiple-ISP collaboration concept can be leveraged to build a global coordinate system. The proposed coordinate system provides accurate network distance information, using not only RTT, but also other important metrics like path capacity, available bandwidth, customer service class, AS relationships and routing policies, without the need for reverse-engineering the Internet by large scale measurements. The system is scalable, resistant to churn, and less susceptible to malicious nodes. The coordinate system can be used by all kinds of user applications, which need some estimation of network properties to choose appropriate neighbours.

Figure 7.3 Comparing multiple-ISP collaboration oracle P2P (Multiple-ISP) with unbiased P2P (No Oracle) and a bandwidth-only based P2P (Bandwidth)



(a) Number of AS hops - box plot

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

(c) Number of router hops - box plot

(d) Bandwidth distribution of peers - bar plot

(e) Estimated content download time - box plot

(f) Router-to-router latency - box plot

8 Reducing Pollution in P2P Systems

In this chapter, we examine the viability of using the oracle to reduce pollution in P2P systems, while at the same time, improving locality. We propose that the oracle uses the proximity as well as the trust information of potential neighbours of a P2P user while ranking them. With the help of large-scale simulations, we show that there are performance gains for both ISPs as well as P2P users, when the oracle helps P2P users choose neighbours which are proximal and have a good reputation. We discuss the problem posed by P2P pollution in Section 8.1, introduce our proposal in Section 8.2, and provide experimental results in Section 8.3.

8.1 Introduction

As the popularity of and amount of traffic in P2P file sharing systems has increased in the last years, so has the pollution of content in such systems. *Pollution* is a kind of attack on P2P file sharing systems, when bogus content is added to popular files in the system. Pollution manifests in various forms, principal among them being content pollution and metadata pollution.

- *Content pollution*: The polluting party modifies the content of a file, e.g., by shuffling bytes, adding messages, or simply inserting white noise.
- *Metadata pollution*: In this case, the content of a file is changed so that it does not match the title any more. When a user downloads the file, he gets content that is completely different from what he expects.

The sources of pollution are intentional as well as unintentional. Sometimes the music industry employs companies to deliberately insert polluted instances of popular music songs in the P2P file-sharing systems [121]. The aim is that P2P users will get frustrated by wasting bandwidth and not finding the desired content, and thus abandon the use of such systems. Such pollution is termed *intentional*. On the other hand, a user accidentally picking up noise while recording a song and putting it up for sharing in a P2P system is an example of *unintentional* pollution.

Studies have shown that pollution can constitute up to 50% of content in popular P2P systems [58]. Even systems like BitTorrent are susceptible to pollution [67]. While polluted content can normally be detected through user inspection after it has been downloaded, the bandwidths of the peers have already been wasted in this case. Besides, if the polluted content contains viruses or trojans, the security implications can be more grave.

Various systems based on trust and reputation have been proposed to reduce the download of polluted content in P2P systems, e.g., Credence [119], P2PRep [8], etc. *Trust* implies the subjective probability with which a peer assesses that other peers will treat it fairly during content exchange. In other words, the amount of trust a peer A has in another peer B, normally reflected by a score, is a measure of the confidence that A has in B, that B will provide the content which A expects it to. *Reputation* of a peer is a reflection of its trustworthiness as seen by the whole community of peers. An overview of various trust- and reputation-based techniques, as well as attacks on them is found in [133].

8.2 Proposal

We propose that the oracle uses the proximity as well as the trust information of the potential neighbours of a P2P user while ranking them. The advantage will be a reduction in the amount of polluted content exchanged in the network, which will not only improve customer satisfaction, but also save the ISP's network resources from unwanted (polluted) traffic. While the benefits of using network proximity to select neighbours have been well researched in the previous chapters, we will now run experiments with the oracle using a combination of proximity and trust to recommend neighbours to a P2P user. This will enable us to investigate the viability of such a scheme, and the performance gains to ISPs as well as P2P users.

Most trust-based schemes rely on querying other peers about the reputation of potential neighbours through voting/polling [8], or build a reputation score based on a statistical measure of the reliability of a peer's past behaviour [119]. We assume that the oracle maintains a centralized list of nodes within its ISP with their trust scores, in addition to their network connectivity information. Maintaining the list at the oracle should lead to a reduction in overhead, and reduced susceptibility to attacks like sybil attack, unfair rating, front peers or collusion. This will be possible because ISP has detailed and authentic knowledge on peers within its network, and does not need to verify it through polling or other mechanisms, which makes other trust-based systems susceptible to such attacks [133].

We agree that this opens the argument that the oracle is no more limited to being a network mapping service, rather it is directly colluding with P2P file sharing systems. However, we also note that using trust as a factor in sorting potential neighbours will allow the ISP to better collude its node-sorting algorithm, and will help to reduce divulgence of network information. Also, it promotes customer service, leading to better customer satisfaction.

The exact implementation details of achieving centralization of trust scores, or the legal argument of the ISP hosting a service directly facilitating P2P file transfer is outside the scope of this work. In this chapter, we restrict ourselves to examining the feasibility of combining trust with proximity for neighbour selection in P2P systems. We foresee the trust-proximity service as a separate add-on service component of the peer mapping oracle service. This can be a viable service offering at least for legal P2P applications that rely on a notion of trust for proper functioning.

8.3 Experiments and Results

To run representative experiments, we use the network model and experimental setup from Section 7.4, where we have $116,000$ P2P nodes distributed in 200 ASes, with Weibull distributions for content availability and churn. We use $50,000$ content files in our experiments, of which 50% are polluted, while the rest are genuine. Each P2P node starts with a trust value of 50, on a scale of 100. When a node comes online, it starts with the trust value from its last session. For each good (genuine) download, the trust value of the peer providing the content increases by 3, while for each bad (polluted) download, the trust value of the peer decreases by 2. This is done to give a benefit of doubt to P2P users who may provide polluted content unknowingly to other users, as they themselves might have received the polluted content from another peer without realizing it. The oracle knows the network connectivity (last-hop bandwidth, router hops, AS distance) as well as the trust score of each peer within its network.

When a P2P user sends the oracle a list of potential neighbours, the oracle sorts the list once based on the trust scores of the peers, and once based on proximity. It then combines the two sorted lists into one list, depending on the weightage given to trust and proximity. We run 6 sets of experiments, varying the weightage or importance that the oracle gives to trust and proximity while sorting the list of neighbours, as listed below. P2P users consult the oracle to choose neighbours in all the sets of experiments except the first one.

- No oracle: P2P users do not consult the oracle. They choose a neighbour for file download randomly, without considering trust or network proximity.
- 100% trust: The oracle sorts the list of potential neighbours based only on trust. Proximity is not considered.
- 70% trust - 30% proximity: The oracle gives 70% weightage to trust and 30% weightage to proximity.
- 50% trust - 50% proximity
- 30% trust - 70% proximity
- 100% proximity: The oracle sorts the potential neighbours based only on proximity, trust is not considered.

The experiments help to evaluate the performance gains that accrue to ISPs as well as to P2P users, when the oracle considers trust and proximity to rank potential neighbours. Also, the experiments help to determine how much weightage needs to be given to trust to reduce pollution in the P2P network.

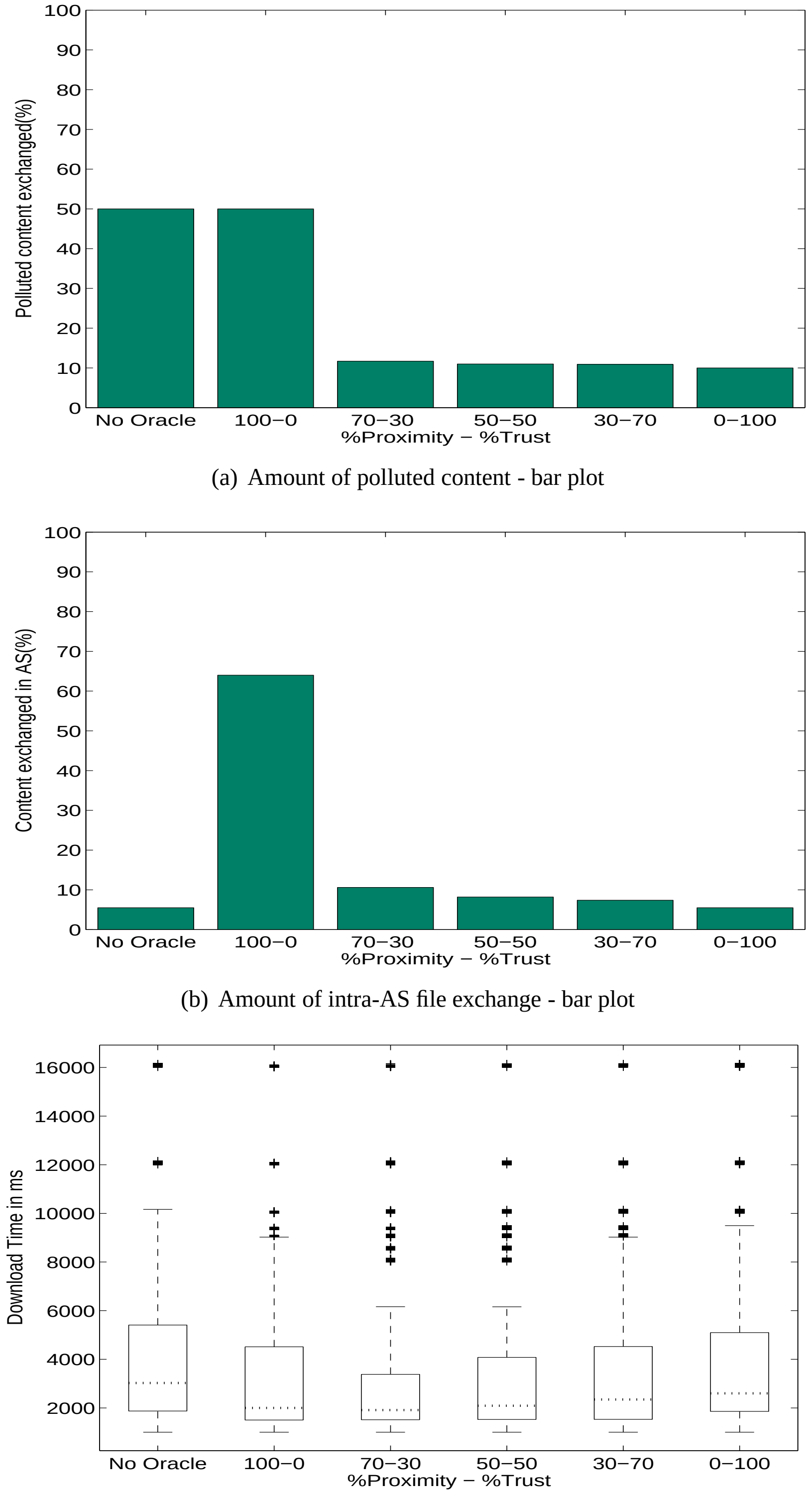
In Figure 8.1, we show the results for $100,000$ file downloads using the following metrics: amount of polluted content exchanged, amount of downloads within an AS, and the download time per file. We see in Figure 8.1(a) that even with 30% weightage given to trust, there is a significant reduction in the amount of polluted content exchanged in the network. The amount of polluted content reduces from 50% in the no oracle or 100% proximity case, to 10% when only 30% weightage is given to trust. The amount of content exchanged within the AS boundaries increases slightly, see Figure 8.1(b). We also notice a reduction in the time taken to download content, see Figure 8.1(c), as compared to the no oracle case. With increasing weightage given to trust, the improvement in results is marginal.

Overall, we see that for content locality and download times, the benefits of using trust along with proximity to rank the potential neighbours of a P2P user are not as significant as when the oracle considers proximity only. However, due to a significant reduction in the amount of polluted content exchanged in the P2P network, both ISPs and P2P users benefit.

8.4 Summary

We have shown that when the oracle uses trust as a factor in sorting the potential neighbours of a P2P user, the amount of polluted content in a P2P network reduces significantly. Even with 30% weightage given to trust (and 70% to proximity) by the ISP's oracle to sort the potential neighbours of a P2P user, the benefits to ISPs as well as P2P systems are clearly evident. The amount of content exchanged that remains within the AS boundaries increases, and the download time for files decreases. At the same time, there is a significant reduction in the polluted content exchanged in the P2P network. This not only improves customer satisfaction, but also saves the ISP's network resources from a significant amount of polluted (and hence unwanted) content.

Figure 8.1 Effects of combining trust and proximity for P2P neighbour selection



(a) Amount of polluted content - bar plot

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

(c) Estimated content download time - box plot

9 Conclusions and Future Work

In this chapter, we summarize the contributions of this thesis, and discuss some of the ongoing and future work that is inspired by this thesis.

9.1 Conclusions

We begin with a measurement study of the Gnutella P2P protocol, and find that its overlay topology is not correlated with the underlying Internet topology. The session length durations in Gnutella are very short, and a large number of overlay peerings cross AS boundaries multiple times. A deeper analysis using a unique visualization technique confirms this result, and shows that overlay topologies differ from random networks as many overlay peerings lie in top-tier ASes where there is a higher prevalence of high-bandwidth connections.

We then propose a simple, general and unique solution that enables ISPs and P2P systems to collaborate with each other. We propose that an ISP hosts a server, which we call the *oracle*, that helps P2P users choose "good" neighbours. Each P2P user has a list of potential neighbours to whom it can connect or download content from. Instead of choosing neighbours independently, we propose that the P2P user sends the list of its potential neighbours to the oracle. The oracle ranks this list of IP addresses based on a number of factors, that an ISP decides individually. For example, the ISP can prefer peers within its network, to prevent traffic from leaving its network. Further, it can choose better bandwidth or lesser delay nodes, or those that are geographically closer (same city, same PoP) within its network. The oracle returns this sorted list to the P2P user, who can then benefit from the knowledge of the ISP and connect to a neighbour recommended by the oracle. This will not only reduce costs and ease routing for ISPs, but will also provide improved performance for P2P users in the sense of higher bandwidth and lesser delay. In this way, ISPs and P2P systems can cooperate so that both of them benefit.

We conduct a comprehensive analysis of this proposal using graph experiments, testbed implementation, Planetlab deployment, and packet-level simulations on various models of P2P systems. The graph results show that P2P users, on consulting the oracle, are able to keep most of their peerings within the ISP boundaries, without any adverse effects on the overlay graph structural properties like small node degree, small path length, small graph diameter, and graph connectedness. The P2P topology is correlated with the Internet AS topology, with dense subgraphs of peerings local to the AS boundaries.

A theoretical analysis of the congestion caused by shorter network paths of P2P links reveals that the congestion in the network is close to the theoretical optimum. This comes with the advantage that almost all the overlay peerings are formed in accordance with the ISP policies. This means that P2P users experience shorter network paths and lesser bottlenecks, with overall network congestion close to the theoretical optimum. At the same time, ISPs save immense costs by keeping P2P traffic local to their networks, or letting it flow along desirable links outside their network while respecting their routing policies.

Through testbed implementation and Planetlab deployment, we show that the ISP-P2P collaboration scheme for neighbour selection in P2P systems is feasible with real P2P systems. We experiment with various underlay topologies as well as uniform and variable content availability in the testbed. The experiment results show that the scalability of P2P systems improve considerably due to a reduction in the overhead traffic by 50%, and there is no adverse effect on the query search phase of P2P networks. The testbed experiments also demonstrate that the source code of P2P protocols can be easily modified to enable ISP-P2P collaboration. Using the Planetlab infrastructure, we show that biased P2P nodes (which consult the oracle for neighbour selection) interact well with unbiased P2P nodes running in the Internet. Furthermore, we verify all the testbed results in the Internet, and show that the P2P users, when consulting the oracle, are able to locate all available content from P2P nodes at shorter network distances.

Having gained insights from the testbed and Planetlab experiments, we implement the Gnutella P2P protocol in the SSFNet packet-level simulation framework, and perform a rigorous analysis of the various aspects of the ISP-P2P collaboration concept using extensive simulations. First, we verify the graph structural properties results with the Gnutella P2P protocol under churn. We then quantify the performance improvements for ISPs and P2P users, using metrics like intra-AS content exchange and content download times. We simulate multiple ISP and P2P topologies, as well as a range of user behaviour characteristics, namely, churn, content availability and query patterns, using different mathematical distributions. This enables us to study the effects of realistic, best-case and adverse scenarios on end-user performance. We show that the benefits of our proposed ISP-P2P collaboration scheme hold across a range of user behaviour scenarios and ISP/P2P topologies. The ISPs are able to save costs by keeping large amount of traffic within their network, perform better traffic engineering, and provide better service to customers. The P2P users benefit through faster content downloads, increased locality of query responses, and improvement in P2P scalability through reduction in overhead traffic. We also show that while the ISPs benefit by keeping traffic within their networks, improvement in download times for P2P users comes about when the oracle considers the last-hop bandwidth of potential neighbours while sorting them.

We extend the ISP-P2P collaboration concept to propose collaboration between multiple ISPs by exchanging summaries of network information through the respective oracle servers. This will enable P2P and other applications to get estimates of the path properties to potential neighbours/servers both within and outside their ISPs. Using simulation results with very large topologies, we show the benefits of multiple-ISP collaboration by comparing its performance with a bandwidth-based neighbour selection scheme in P2P systems. The results demonstrate that multiple-ISP collaboration achieves near-optimal performance in terms of content locality, network latency and download times. The scheme leads to a balanced distribution of content among P2P nodes in the sense that the high-bandwidth peers are not indirectly penalized by having to serve too much content to other peers. The ISPs are also able to use their network resources in an efficient and balanced manner. We further show how the multiple-ISP collaboration concept can be leveraged to build a global coordinate system, and discuss its advantages as compared to existing coordinate systems.

Lastly, we examine the viability of using the oracle service to reduce pollution in P2P file-sharing systems while preserving network locality. We propose that the oracle uses the proximity as well as the trust information of the potential neighbours of a P2P user while ranking them. Experiment results show that the major advantage is a reduction in the amount of polluted content exchanged in the network, which not only improves customer satisfaction, but also saves the ISP's network resources from unwanted (polluted) traffic. We also notice an increase in the content locality, as well as a reduction of content download times.

To sum up, this thesis proposes collaboration between ISPs and P2P systems so that both of them benefit, and makes a thorough analysis of the proposal and its extensions as outlined above.

9.2 Ongoing and Future Work

The Internet Engineering Task Force (IETF) is considering our proposal along with the P4P proposal [126] for adoption as a standard to enable ISP-P2P collaboration. We were invited to present our work at the IETF P2P Infrastructure Workshop at MIT, Boston, USA on 28 May 2008. We wish to pursue the standardisation effort and enter into close collaboration with the P4P group to achieve a common standard for ISP-P2P collaboration.

A prototype of the oracle service is being implemented based on the BIND protocol. It will rely on real ISP network information of a major telecommunications network provider and interact with popular P2P applications like BitTorrent, eDonkey and Gnutella. We will modify the source code of BitTorrent and eDonkey clients so that they can communicate with the oracle service and choose neighbours based on the oracle's recommendation. The oracle service software as well as the software patches for the P2P clients will be publicly available. A homepage has been made for this project [47] which will be kept up-to-date with the latest developments.

A number of ISPs have shown keen interest in our proposal, and we are currently involved in developing a product prototype for a major telecommunications network provider. This is likely to lead to a product offering, as well as applications of the oracle service in various other projects of this network provider.

Another direction of future work is to analyze the use of the oracle in P2P-TV [74] applications. We are in touch with the P2P-Next [73] project consortium and the Tribler [116] software development team, and intend to enhance collaboration with this project in the near future.

Acknowledgments

There are many people and institutions who have contributed generously in making this thesis possible, and I am indebted to all of them! Writing the Acknowledgement section is the most gratifying as well as the most intriguing part of this thesis for me. It is gratifying, because this is my opportunity to thank many of the people from whom I have learnt so much. At the same time, it is also a most intriguing task for me, because I know that I will not be able to name “all” the people who have supported me, and even for those people whom I mention, I will not be able to give the full credit that they deserve. Nevertheless, in my own little way, I will try to list some of the most prominent contributors to my thesis.

I would like to begin with my PhD advisor, mentor and guide, Prof. Anja Feldmann. It will not be untrue to say that she was a major reason why I decided to come to Germany to pursue a PhD. I have spent six valuable years of my life under her guidance, and these years have broadened my horizons in multiple dimensions. She has introduced the world of research and networking to me, and needless to say, I am thankful to her for her technical guidance. However, there are many other important things that she has taught me as well. To name a few, she has taught me that when approached properly, sheer hard work can be a lot of fun. She has shown me how to use an elephantine memory to give attention to minute details while not loosing the overall picture, how to inspire co-workers, how to be a friend and a boss at the same time, how to approach mammoth problems coolly, and how to keep persevering against all odds until success is achieved. I have learnt from her how one can believe in his own ideas while respecting contrasting opinions. Her methodical and logical approach to problem solving and communicating solutions is commendable, and so is her ability to accept success with humility.

Next I would like to thank all my colleagues with whom I have worked at the university in Munich and Berlin. I once again regret that I cannot name all of them, or mention all the things for which I am thankful to them, but I try to mention a very few of them here (in no particular order): Olaf Maennel for helping me learn Internet routing; Robin Sommer and Holger Dreger for helping me with all the bureaucracy in the initial years and to get settled in Munich; Nils Kammenhuber and Arne Wichmann for always lending an ear and giving advice when I needed it; Harald Schiöberg, Gregor Maier and Vlad Manilici for spending weeks reading my thesis drafts and giving detailed feedback; Wolfgang Mühlbauer for showing that when properly planned and efficiently executed, one can do real quality work in normal working hours; Jan Böttger for out-of-the-way system admin support and for always lending a shoulder during my stressful moments; Doris Reim, Thomas Hühn and Bernhard Ager for reading parts of this thesis; Jörg Wallerich, Fabian Schneider, Stefan Kornexl, Britta Liebscher, Petra Lorenz, Amir Mehmood and Andi Wundsam for helping with so many technical, bureaucratic and other matters; and to everybody above for supporting and helping me in so many other ways. Working at the group with all of them has been a pleasant, stress-free, enrichening and fun-filled experience. All of them have become very good friends to me, and I eagerly look forward to maintaining a very healthy relationship with all of them for the rest of my lifetime.

During the course of my PhD work, I had the honour and pleasure to work with many talented students: Obi Akonjang, Rumen Tashev, Ximena Cabezas, Sebastian Mohrs, Shailesh Gupta and Rüdiger Wolf-Sebottendorf. My interactions with each of them have enriched me in many ways.

I thank Prof. Adam Wolisz, Prof. Odej Kao and Prof. Hans Albert Aukes for agreeing to be the examiners of this thesis. I am indebted to Prof. Christian Scheideler for helping me with the theoretical analysis and congestion analysis of biased neighbour selection.

I am also very thankful to Mr. Peter Möckel and Dr. Heinrich Arnold, who have introduced me to a positive, humane and encouraging picture of corporate career, and have mentored me to transfer my research results to the industry.

I also thank the institutions: Deutsche Telekom Laboratories, Max-Planck-Institut für Informatik, University of Saarland, Technical University of Munich and Technical University of Berlin. Working at each of these institutions has been an honour and a great learning experience!

On a more personal note, I thank all the friends I have made in Germany, especially those at the ISKCON temples, who have sweetened my stay in Germany and helped me feel at home here.

My parents and my sister have always been there for me, and supported me through thick and thin. Their love and belief in me has made this thesis possible. My Gurudeva, Pujya Swami Sampoornanand "Bapuji" has been present with me each and every minute and continues to guide me with His love, wisdom and *ashirwad*. Six years ago, He laid the foundation for my journey to Germany, and continues to sow the seeds of all my current and future endeavours that have led to any degree of success. Words simply fail to express my gratitude to my parents, sister and beloved Gurudeva, hence I express it through silence. The Guru *tattva* has often manifested in the persons of Annant Das ji Maharaj and Srila Sacinandana Swami, and I offer my humble obeisances to them.

Though it may sound unconventional, I offer my respects to my *janmabhoomi*, the sacred land of *Bharatvarsha* (commonly known as India), and to my *karmabhoomi*, the land of Germany.

To all those whom I could not mention by name here, I humbly seek their forgiveness. I also assure them that even though they are not mentioned on this page, they are certainly present in my thoughts. No matter how hard I try, I will never be able to fully acknowledge the support of all the people who have helped me. All I can say is that my gratitude to them will be evident in the love and respect that I will continue to show to them during my lifetime.

I fold my hands in a humble prayer to Lord Shri Radha Krsna!

> *Hare Ram Hare Ram, Ram Ram Hare Hare*
> *Hare Krsna Hare Krsna, Krsna Krsna Hare Hare*
>
> *Om Sri Ram Jai Ram Jai Jai Ram*

- a Hindu prayer.

Bibliography

[1] Abilene Observatory Routing Data. http://abilene.internet2.edu/observatory/.

[2] I. Abraham, D. Malkhi, and O. Dobzinski. LAND: stretch $(1+\varepsilon)$ locality-aware networks for DHTs. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004.

[3] E. Adar and B. Huberman. Free Riding on Gnutella, 2000. http://www.firstmonday.dk/issues/issue5_10/adar/.

[4] M. Adler, R. Kumar, K. Ross, D. Rubenstein, T. Suel, and D. Yao. Optimal Selection of Peers for P2P Downloading and Streaming. In *IEEE INFOCOM*, 2005.

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

[6] D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris. Resilient Overlay Networks. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2001.

[7] D. Applegate and E. Cohen. Making Intra-domain Routing Robust to Changing and Uncertain Traffic Demands. In *ACM SIGCOMM*, 2003.

[8] R. Aringhieri, E. Damiani, S. Vimercati, S. Paraboschi, and P. Samarati. Fuzzy Techniques for Trust and Reputation Management in Anonymous P2P Systems. In *Wiley Journal of the American Society for IST, 57(4)*, 2006.

[9] S. Arora, S. Rao, and U. Vazirani. Expander Flows, Geometric Embeddings and Graph Partitioning. In *ACM Symposium on Theory of Computing (STOC)*, 2004.

[10] Flow network. http://en.wikipedia.org/wiki/Flow_network.

[11] B. Awerbuch and F. Leighton. Improved Approximation Algorithms for the Multi-Commodity Flow Problem and Local Competitive Routing in Dynamic Networks. In *ACM Symposium on Theory of Computing (STOC)*, 1994.

[12] V. Batagelj and M. Zaversnik. Generalized Cores. In *Preprint 40(799), IMFM Ljublana*, 2002.

[13] M. Baur, U. Brandes, M. Gaertler, and D. Wagner. Drawing the AS Graph in 2.5 Dimensions. In *12th International Symposium on Graph Drawing*, 2005.

[14] Bearshare. http://www.bearshare.com/.

[15] BIND - Berkeley Internet Name Domain. http://en.wikipedia.org/wiki/BIND.

[16] 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 *IEEE ICDCS*, 2006.

[17] Box Plot - Wikipedia. http://en.wikipedia.org/wiki/Box_plot.

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

[19] H. Chang, S. Jamin, Z. Mao, and W. Willinger. An Empirical Approach to Modeling Inter-AS Traffic Matrices. In *ACM Internet Measurements Conference*, 2005.

[20] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making Gnutella-like P2P Systems Scalable. In *ACM SIGCOMM*, 2003.

[21] M. Crovella and B. Krishnamurthy. *Internet Measurement: Infrastructure, Traffic and Applications*. Wiley, 2006.

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

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

[24] A. Dhamdhere, E. Zegura, and R. Liston. Determining Characteristics of the Gnutella Network. In *Technical Report, College of Computing, Georgia Tech*, 2002.

[25] F. Fessant, S. Handurukande, A. Kermarrec, and L. Massoulie. Clustering in P2P File Sharing Workloads. In *International Workshop on P2P Systems (IPTPS)*, 2004.

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

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

[28] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. IDMaps: A Global Internet Host Distance Estimation Service. In *IEEE/ACM Transactions on Networking*, 2001.

[29] M. Gaertler and M. Patrignani. Dynamic Analysis of the Autonomous System Graph. In *Inter-Domain Performance and Simulation*, 2004.

[30] L. Gao. On Inferring Autonomous System Relationships in the Internet. In *IEEE Global Internet Symposium*, 2000.

[31] A. Gish, Y. Shavitt, and T. Tankel. Geographical Statistics and Characteristics of P2P Query Strings. In *International Workshop on P2P Systems (IPTPS)*, 2007.

[32] C. Gkantsidis, M. Mihail, and E. Zegura. Spectral Analysis of Internet Topologies. In *IEEE INFOCOM*, 2003.

[33] Gnucleus. http://www.gnucleus.com/.

[34] Gnutella Hostcache. http://wiki.limewire.org/index.php?title=The_Local_Hostcache.

[35] Gnutella v0.6 RFC. http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.

[36] Gnutella Wikipedia. `http://en.wikipedia.org/wiki/Gnutella`.

[37] R. Goerke, M. Gaertler, and D. Wagner. LunarVis - Analytic Visualizations of Large Graphs. In *15th International Symposium on Graph Drawing*, 2008.

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

[39] S. Gupta. Preparation of Testlab and Experiments with Selective Neighbour Selection for P2P systems. In *Master thesis, Indian Institute of Technology (IIT) Kharagpur and Technical University of Munich*, 2006.

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

[41] Java Docs - HashSet. `http://java.sun.com/j2se/1.4.2/docs/api/java/util/HashSet.html/`.

[42] K. Ho, J. Wu, and J. Sum. On the Session Lifetime Distribution of Gnutella. In *International Journal of Parallel, Emergent and Distributed Systems, 23(1)*, 2007.

[43] IEEE Computer Society: 802.1Q IEEE Standards for Local and Metropolitan Area Networks - Virtual Bridged LANs. `http://standards.ieee.org/getieee802/download/802.1Q-2003.pdf`.

[44] Intel-DANTE Monitoring Project. `http://www.cambridge.intel-research.net/monitoring/dante/`.

[45] Internet Deutschland. `http://www.internet-sicherheit.de/internet-deutschland.html`.

[46] Ipoque: Internet Study 2007. `http://www.ipoque.de/news_&_events/internet_studies/internet_study_2007`.

[47] ISP-P2P Collaboration project homepage. `http://www.net.t-labs.tu-berlin.de/research/isp-p2p`.

[48] M. Kaafar, L. Mathy, C. Barakat, K. Salamatian, T. Turletti, and W. Dabbous. Securing Internet Coordinate System: Embedding Phase. In *ACM SIGCOMM*, 2007.

[49] M. Kaafar, L. Mathy, T. Turletti, and W. Dabbous. Virtual Networks under Attack: Disrupting Internet Coordinate Systems. In *ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2006.

[50] T. Karagiannis, A. Broido, N. Brownlee, kc Claffy, and M. Faloutsos. Is P2P Dying or Just Hiding? In *IEEE GLOBECOM*, 2004.

[51] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should ISPs fear Peer-Assisted Content Distribution? In *ACM Internet Measurements Conference*, 2005.

[52] R. Keralapura, N. Taft, C. Chuah, and G. Iannaccone. Can ISPs Take the Heat from Overlay Networks? In *Hot Topics in Networking (HotNets)*, 2004.

[53] A. Klemm, C. Lindemann, M. Vernon, and O. Waldhorst. Characterizing the Query Behavior in P2P File Sharing Systems. In *ACM Internet Measurements Conference*, 2004.

[54] D. Knuth. Two Notes on Notation. In *American Mathematical Monthly, 99*, 1990.

[55] P. Kolman and C. Scheideler. Improved Bounds for the Unsplittable Flow Problem. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2002.

[56] J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In *USENIX Networked Systems Design and Implementation (NSDI)*, 2007.

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

[58] J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in P2P File Sharing Systems. In *IEEE INFOCOM*, 2005.

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

[60] H. Lim, J. Hou, and C. Choi. Constructing Internet Coordinate System Based on Delay Measurement. In *IEEE/ACM Transactions on Networking, 13(3)*, 2005.

[61] LimeWire. `http://www.limewire.com/`.

[62] LimeWire - Query Routing for the Gnutella Network. `http://www.limewire.com/developer/query_routing/keywordrouting.htm`.

[63] P. Linga, I. Gupta, and K. Birman. A Churn-Resistant P2P Web Caching System. In *ACM Workshop on Self-Survivable and Regenerative Systems (SSRS)*, 2003.

[64] E. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft. On the Accuracy of Embeddings for Internet Coordinate Systems. In *ACM Internet Measurements Conference*, 2005.

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

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

[67] T. Mennecke. New Breed of Corrupt Torrent Infiltrates BitTorrent. `http://www.slyck.com/news.php?story=926`.

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

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

[70] M. Naor and U. Wieder. Novel architectures for P2P applications: the continuous-discrete approach. In *ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, 2003.

[71] T. Ng and H. Zhang. Predicting Internet Network Distance with Coordinates-Based Approaches. In *IEEE INFOCOM*, 2002.

[72] Bill Norton. The Art of Peering: The Peering Playbook, 2002.

[73] P2P-Next. `http://www.p2p-next.org`.

[74] P2P-TV. `http://en.wikipedia.org/wiki/P2PTV`.

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

[76] PeerSim. `http://peersim.sourceforge.net`.

[77] Planetlab. `http://www.planet-lab.org/`.

[78] G. Plaxton, R. Rajaraman, and A. Richa. Accessing Nearby Copies of Replicated Objects in a Distributed Environment. In *ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, 1997.

[79] Pong Caching. `http://wiki.limewire.org/index.php?title=Pong_Caching_GDF`.

[80] R. Prasad, M. Murray, C. Dovrolis, and K. Claffy. Bandwidth Estimation: Metrics, Measurement Techniques and Tools. In *IEEE Network*, 2003.

[81] Associated Press. Comcast blocks some Internet traffic. `http://www.msnbc.msn.com/id/21376597`.

[82] Associated Press. Comcast to stop blocking Internet traffic. `http://www.msnbc.msn.com/id/23827953`.

[83] pWhoIs. `http://pwhois.org`.

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

[85] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Topologically Aware Overlay Construction and Server Selection. In *IEEE INFOCOM*, 2002.

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

[87] Y. Rekhter and T. Li. Border Gateway Protocol. `http://www.ietf.org/rfc/rfc1771.txt`.

[88] S. Rewaskar and J. Kaur. Testing the Scalability of Overlay Routing Infrastructures. In *Passive and Active Measurements (PAM)*, 2004.

[89] RIPE Routing Information Service. `http://data.ris.ripe.net/`.

[90] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the Gnutella Network: Properties of Large-Scale P2P Systems and Implications for System Design. In *IEEE Internet Computing Journal*, 2002.

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

[92] S. Saroiu, K. Gummadi, and S. Gribble. A Measurement Study of P2P File Sharing Systems. In *Multimedia Computing and Networking*, 2002.

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

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

[95] C. Scheideler. Towards a Paradigm for Robust Distributed Algorithms and Data Structures. In *HNI Symposium on New Trends in Parallel and Distributed Computing*, 2006.

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

[97] S. Seidman. Network Structure and Minimum Degree. In *Social Networks, 5*, 1983.

[98] K. Shanahan and M. Freedman. Locality Prediction for Oblivious Clients. In *International Workshop on P2P Systems (IPTPS)*, 2005.

[99] Y. Shavitt and T. Tankel. On the Curvature of the Internet and its Usage for Overlay Construction and Distance Estimation. In *IEEE INFOCOM*, 2004.

[100] G. Shen, Y. Wang, Y. Xiong, B. Zhao, and Z. Zhang. HPTP: Relieving the Tension between ISPs and P2P. In *International Workshop on P2P Systems (IPTPS)*, 2007.

[101] Slyck. `http://www.slyck.com/`.

[102] Slyck: P2P Remains Dominant Protocol. `http://www.slyck.com/story1502_P2P_Remains_Dominant_Protocol`.

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

[104] IP Addresses in SSFNet. `http://www.ssfnet.org/InternetDocs/ssfnetTutorial-1-vlsm.html`.

[105] SSFNet. `http://www.ssfnet.org`.

[106] R. Steinmetz and K. Wehrle. *P2P Systems and Applications*. Springer Lecture Notes in Computer Science, 2005.

[107] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A Scalable P2P Lookup Service for Internet Applications. In *ACM SIGCOMM*, 2001.

[108] D. Stutzbach and R. Rejaie. Understanding Churn in P2P Networks. In *ACM Internet Measurements Conference*, 2006.

[109] D. Stutzbach, R. Rejaie, and S. Sen. Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems. In *ACM Internet Measurements Conference*, 2005.

[110] A. Su, D. Choffnes, F. Bustamante, and A. Kuzmanovic. Relative Network Positioning via CDN Redirections. In *IEEE ICDCS*, 2008.

[111] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz. Characterizing the Internet Hierarchy from Multiple Vantage Points. In *IEEE INFOCOM*, 2002.

[112] Symella. `http://symella.aut.bme.hu`.

[113] R. Tashev. Experimenting with Neighbour Discovery Schemes for P2P Networks in a Simulation Framework. In *Master thesis, Department of Computer Science, Technical University of Munich*, 2006.

[114] J. Tian and Y. Dai. Understanding the Dynamic of P2P Systems. In *International Workshop on P2P Systems (IPTPS)*, 2007.

[115] TK Fachbegriffe. `http://www.tk-fachbegriffe.de/index.php?id2=2400&a=320`.

[116] Tribler. `http://www.tribler.org`.

[117] International Herald Tribune. Internet providers wary of being cybercops. `http://www.iht.com/articles/2008/04/13/business/ISP14.php`.

[118] Virtual LAN. `http://en.wikipedia.org/wiki/Vlan`.

[119] K. Walsh and E. Sirer. Experience with an Object Reputation System for P2P Filesharing. In *USENIX Networked Systems Design and Implementation (NSDI)*, 2006.

[120] WHOIS. `http://en.wikipedia.org/wiki/Whois`.

[121] Wired. Hitting P2P Where It Hurts. `http://www.wired.com/entertainment/music/news/2003/01/57112`.

[122] R. Wolf-Sebottendorf. Building Complex P2P Topologies in a Testlab Environment. In *Master project, Department of Computer Science, Technical University of Munich*, 2006.

[123] R. Wolf-Sebottendorf. Experiments with P2P Neighborhood Discovery Algorithms in Globally Distributed Environments. In *Master thesis, Department of Computer Science, Technical University of Munich*, 2007.

[124] B. Wong, I. Stoyanov, and E. Sirer. Octant: A Comprehensive Framework for Geolocalization of Internet Hosts. In *USENIX Networked Systems Design and Implementation (NSDI)*, 2007.

[125] H. Xie and Y. Yang. A Measurement-based Study of the Skype P2P VoIP Performance. In *International Workshop on P2P Systems (IPTPS)*, 2007.

[126] H. Xie, Y. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Portal for (P2P) Applications. In *ACM SIGCOMM*, 2008.

[127] yWorks. `http://www.yworks.com`.

[128] ZDNet. ISPs see costs of file sharing rise. `http://www.zdnet.com/2100-9584_22-1009456.html`.

[129] Y. Zhang, N. Duffied, V. Paxson, and S. Shenker. On the Constancy of Internet Path Properties. In *ACM Internet Measurements Workshop*, 2001.

[130] B. Zhao, Y. Duan, L. Huang, A. Joseph, and J. Kubiatowicz. Brocade: Landmark Routing on Overlay Networks. In *International Workshop on P2P Systems (IPTPS)*, 2002.

[131] S. Zhao, D. Stutzbach, and R. Rejaie. Characterizing Files in the Modern Gnutella Network. In *Multimedia Computing and Networking*, 2006.

[132] H. Zheng, E. Lua, M. Pias, and T. Griffin. Internet Routing Policies and Round-Trip-Times. In *Passive and Active Measurements (PAM)*, 2005.

[133] B. Zhu, S. Jajodia, and M. Kankanhalli. Building Trust in P2P Systems: a Review. In *International Journal Security and Networks*, *1(1/2)*, 2006.