Issues in Internet's Scalability: Structure and Applications

Dissertation submitted for the degree "Doctor of Philosophy"

Osnat Mokryn

Submitted to the Senate of the Hebrew University in Jerusalem (2003)

This work was carried out under the supervision of
Prof. Danny Dolev and **Dr. Yuval Shavitt**.

To Mark and Tomer

Acknowledgments

A few years ago I joined Professor Dolev's group in the Hebrew University. It took less than a few months for Danny to become a huge source of support and guidance for me. Danny has always encouraged me to explore new directions and I could always count on his deep understanding and experience at the different stages of this work. I thank him deeply for his enriching guidance and friendship.

During the first stages of this thesis I found myself consulting many times with an old friend, Yuval Shavitt. When he returned to Israel, it was only natural that he will become a co-advisor. Yuval is an excellent teacher and a challenging advisor. I benefited a lot from his creativity and dedication, as well as his extensive knowledge in a wide range of subjects. It has always been a great pleasure to work with him. I especially thank Yuval for always being available to discuss an idea, share a thought, and think together on problems I encountered.
I also thank Nira, Yuval's wife and my friend, for her support and friendship.

I would like to thank the members of the Transis and DANSS group I have enjoyed working with, Tal Anker, Ohad Rodeh, Grisha Chokler, David Breitgand and Ilia Shanayderman. It was always a pleasure to come to the DANSS lab. I especially thank them for their patience that allowed me to bring my son to the lab in the first six months of his life.
I thank the people in Tel-Aviv for their great company in the last two years, especially Eli Brosh and Eran Bashan.

Home was always a great source of support. I would like to thank Mark for his constant support and love, for reading and English proofing many of my papers, and for his belief in me. I thank my son, Tomer, for bringing so much happiness to my life, and for the long peaceful hours he spent with me in the lab, enabling me both to work and be with him. Home is where you guys are.

Last, but never least, I would like to thank my parents, and especially my mom, for the support throughout these years. I especially thank her for the exhausting journeys to Modiin and back to be with Tomer, that enabled me to spend peaceful needed hours at work.

Contents

Abstract

The Internet is becoming a vital tool in today's communication, information and data retrieval, as well as commerce. Its fast spreading, non affected even by the recent bubble burst, exceeds the rate of software and hardware development, and makes it harder for monitoring, measuring and understanding.

In this work we investigate different aspects of the structural properties of the Internet, and suggest how to elevate some of our findings to comprise better application and routing layer services.

First, we investigate the exact structure of general shortest path multicast trees in the Internet. We present a thorough investigation of the structure of multicast trees cut from the Internet and power-law topologies. Based on both generated topologies and real Internet data, we characterize the structure of such trees and show that they obey the rank-degree power law; that most high degree tree nodes are concentrated in a low diameter neighborhood; and that the sub-tree size also obeys a power law.

Our most surprising empirical finding suggests that there is a linear ratio between the number of high-degree network nodes, namely nodes whose tree degree is higher than some constant, and the number of leaf nodes in the multicast tree (clients). We also derive this ratio analytically. Based on this finding, we develop the Fast Algorithm, that estimates the number of clients, and show that it converges faster than one round trip delay from the root to a randomly selected client.

We leverage this finding in an application layer scheme for the dissemination of very popular content to a very large audience. The scheme uses an integrated architecture of HTTP unicast and a cyclic multicast delivery of the popular content, and relies on an accurate evaluation of the multicast group size. We also develop an additional end to end counting algorithm for this evaluation.

We further investigate the tomography of multicast trees, and find that not only it conforms to the findings of the scale free properties of the tree, but also has the exact same characteristics as the Internet's tomography. This finding deepens our understanding on the exact structure of multicast trees in the Internet, on a layer by layer basis.

We conclude the work by investigating the nature of the resiliency of the Internet at the Autonomous System (AS) level to failures and attacks, under the real constraint of business agreements between the ASs. The agreements impose policies that govern routing in the AS level, and thus the resulting topology graph is directed, and does not maintain transitivity. We show, using partial views obtained from the Internet, that the Internet's resiliency to a deliberate attack is much smaller than previously found. Its reachability is also somewhat lower under random failures, with the surprising result that it becomes closer to the optimum when the average degree of the Internet increases. We further investigate the effect of added backup connectivity on the resiliency.

Chapter 1

Introduction

1.1 Introduction

A few years ago, a social studies teacher in Taylorsville Elementary School in North Carolina started an email project with her sixth-grade students. She asked them to send a short email message to all of their family and friends asking them to forward it to ”everyone you know so that they can send it to everyone they know (and so on)”. They also requested that each recipient respond to them so they could keep a record of how many people had been reached and where. A few weeks later the project was cancelled, after the class had received over 450,000 responses from all states and eighty-three other countries [Wat03]. The six-graders and their teacher have just received a very good example of a phenomenon called A Small World, used to describe the topological characteristics of complex networks, such as human social connections and the Internet.

The aim of this thesis is to investigate the Internet as a very large complex network, and use its properties to aid in solving and understanding scalability problems. While the Internet is spreading and increasing in size, it was also found, in recent years, that it exhibits characteristics of small world networks. These include mainly a low diameter and a power law degree distribution (See Chapter 2 for a detailed explanation). This discovery can affect almost every possible aspect of service and application over the Internet, as information spreads very fast and reaches a vast amount of receivers within almost a constant time, regardless of the number of possible receivers. For example, pure peer-to-peer applications requiring a full distributed mechanism in which all messages are broadcast, are more likely to jam the Internet than to supply a scalable service to their users. Understanding the exact nature of the Internet structure, will enable an efficient and scalable use of it.

Our work starts with generating Internet like networks, using the scale free Notre-Dame algorithm. Chapter 2 is dedicated to the understanding of the properties of complex networks in general, and the Internet in particular, and describes the properties of the Notre-Dame algorithm we used for generating topologies. We used the generator to further validate different characteristics of the Internet and to deepen our understanding of it. We generated topologies of different characteristics, which were used in this work.

Next, we started exploring and understanding the structure and topology of large scale multicast trees. In the beginning of the 1990’s, IP-multicast has emerged within the research community as the next generation killing underlying service, and was the subject of intensive research. The

advantages seemed very clear: An efficient way to reach a very large group of receivers, while using the minimal possible amount of infrastructure, and duplicating messages as close to the end receivers as possible. In 1998, Chuang and Sirbu [CS98] tried to quantify the gain in multicast, finding a power law relation between the amount of receivers and the average unicast path length to the receivers. They conducted their experiment over exemplary data sets. A later work by Shenker et al. [PST99] explored this relationship analytically on k-ary trees. However, the industry seemed reluctant to find the right applications and up until today IP-multicast has not yet spread as an underlying available service. The structure of such trees could not be determined precisely [CA01], nor their exact properties.

Chapter 2 describes our research on the exact structure of large scale multicast trees in the Internet. We create shortest path trees, cut from power law tail topologies and the Internet, and obtain data of real Internet multicast trees. We show that the trees exhibit small world characteristics. In particular, we find that there is a degree-frequency power law tail distribution of the trees and even at each of the trees' layers. The trees exhibit scale free characteristics, and their inner structure is scale free in nature, as we found that the sub-tree sized exhibit a power law tail distribution as well. We further found that the hierarchical structure of the Internet [SARK02] is also preserved in these trees, and that there is a core in which the high degree nodes are located, which in general is very close to the root of the multicast tree. Studying thoroughly the different topological characteristics of the multicast trees, we discovered a surprising linear ratio between the number of high degree nodes in the trees and the number of receivers the trees span. We further proved this ratio analytically, and found a predictor that enables to estimate the number of receivers from the number of high degree nodes (routers) in the tree. Based on this mechanism, we devised algorithms for estimating the size of a large scale multicast tree in less than the Internet round trip delay, and proved it analytically. Further research we conducted on the topological structure of large scale multicast trees at the different layers proved that similarly to the Internet, the power of the distribution of degrees of nodes at each layer around the root can be calculated as a function of the tree's power law distribution and the number of the layer. Hence, the distribution of the degree of the nodes at each layer can be predicted, and aid in the design of scalable services and applications such as server locations for video on demand, caches, etc.

As a result of our findings on the characteristics of multicast trees, we devised an adaptive scheme for a large scale multicast of semi-dynamic information over the Internet. Our scheme, called the Integrated architecture, enables sites to adjust dynamically to different demands, and stay active at the highest peak times without enlarging their hardware. The Integrated Architecture defines an extension to the HTTP protocol, called HTTPM. The HTTPM extension is used only for pages for which demand is known to be very high regularly, or that have the potential of becoming very hot, such as a page outlining the vote counting in each state on election date. When demand to an HTTPM page crosses a predetermined threshold, the page reverts to a cyclic multicast delivery. A plug-in at the receiver's browser identifies the change, and joins the multicast group. The server then activates a large scale estimations mechanism, such as the one we devised, to identify a decrease in demand, in which case it reverts back to an end-to-end unicast delivery according to the HTTP protocol. In this work we use both the underlying topological structure of the Internet as well as the underlying structure of the World Wide Web (another complex network with small world characteristics). A detailed description of the work is given in chapter 4.

The last chapter of the work investigates a specific property of the Internet, its resiliency to random failures and attacks. Complex networks such as the Internet exhibit similar behavior in

general, due to their unique topological structure. However, the topological structure is only one of the factors that govern the behavior of such networks. We investigate here the mechanisms that govern the operation of the Internet, and show their influence on its tolerance. The Internet today consists of thousands of subnetworks, each with its own administrative management, called autonomous systems (ASs). Each such AS uses an interior routing protocol (such as OSPF, RIP) inside its managed network, and communicates with neighboring ASs using an exterior routing protocol, called BGP. The BGP protocol enables each administrative domain to decide which routes it accepts and which it announces. Through the use of the protocol the autonomous systems select the best route, and impose business relationships between them on top of the underlying connected topology. As a result, paths in the Internet are not necessarily the shortest possible, but rather the shortest that conform to the ASs policies. Such routing is called policy based routing. The business agreements impose restrictions on the usage of network paths [Gao00, SARK02], namely, the existence of a path between two nodes, does not imply that they are reachable from each other. Thus, connectivity and reachability are not identical, the former means that a physical path exists between two nodes, and the latter that communication can flow between them. Reachability in the network maintains reflexivity but not transitivity; A node can be reachable from two nodes that are not mutually reachable. In this work we measure the Internet reachability and compare it to the connectivity used in previous studies [AJB00, CEbAH01, CEbAH00a, BT02, PKP$^+$03]. Our results show that the Internet is much more susceptible to attacks than previously found, though the resiliency to failures is close to the theoretic optimum. The different tests we conducted also led us to conclude that small and medium sized ASs rely heavily on multihoming.

Chapter 2

The Topology Generator

2.1 Introduction

The commercializing of the global IP based network, named Internet, has caused it to grow tremendously, and without any global planning. The Internet today is a vast collection of self administered routing domains, called autonomous systems, communicating using the IP protocol set. The Internet has become a spontaneously growing network of networks, and its structure a subject of extensive research. The Internet's topology is studied at two levels. The first is the routers level,where the routers are the nodes and the edges are the physical links connecting them. The second level is the autonomous systems (AS) level, modeling the ASs as nodes, connecting two ASs with a link if they are BGP neighbors.

In this chapter, we describe the fundamental characteristics of complex networks; Describe related work that charactgerises the Internet as a complex network; and describe in detail the topology generator we chose to work with.

Traditionally, complex large scale networks such as the Internet were described as random graphs, First studied by the Hungarian mathematicians Paul Erdös and Alfred Rényi [ER60]. According to Erdös-Rényi (ER) model, the graph is constructed by starting with the total number of nodes, N, and connecting each pair of nodes with probability p, resulting in a randomly distributed graph connectivity of N(N-1)/2. In such networks, all nodes have the same probability to have the graph average degree.

A first attempt to model the Internet as a complex network was made by Waxman [Wax88]. In his model he made slight modifications to the ER model, by taking into consideration the intuitive observation that links that represent long distances are less likely to appear in the graph than the ones representing short distances. Zagura et al. suggested an intuitive hierarchical model, modeling the Internet with a center.

In the late 1990's there was a significant advancement in our understanding of the structure of complex networks. Researches from different disciplines were engaged in the investigation of complex networks such as physical, biological, social or computers. The research was prompted by the availability of large scale data and advanced computing abilities.

Complex networks, such as the Internet, were shown to have three distinctive characteristics, described bellow.

High Degree of Clustering Real networks were shown to exhibit a high degree of clustering,

quantified by a clustering coefficient. Watts and Storage [WS98] discovered that social networks have an inherent tendency to clustering. A common property of social networks is that they display a large clustering coefficient, i.e., on average a person's friends are far more likely to know each other than two people chosen at random. (On the other hand, it is important to note, that it is possible to connect two people chosen at random via a chain of only a few intermediaries [Mil67]). It was later shown that complex networks such as the Internet have a large clustering coefficient, which is distinctively higher than the one exhibited in random networks.

Small World Classification The small world characteristic is measured by the average shortest path between any two nodes in the network, where nodes represent routers or autonomous systems in the case of the Internet, or human beings in complex social networks. The small world phenomenon was first evident in an interesting social experiment conducted in the 60's by Milgram [Mil67]. People from all over the US were given letters to send to a certain address, under the condition that it will be sent to someone they know and think might be closer to the given address, and so forth. The letters that arrived to the final destination, arrived after traveling through six different people on the average. Thus, Milgram concluded that the average distance between any two people in the US is six. In fact, even traditional random graphs exhibit the small world phenomenon, with an average path length that is logarithmic in the number of nodes. Other popular famous examples of the small world phenomenon are the actors network (two actors are linked if they have played together in a movie), and the network of people who have collaborated with people who have collaborated with the Hungarian mathematician Erdös (specifically, researchers who have a joint worked with Erdös are given Erdös number one, researchers who collaborated with Erdös number one researches are given Erdös number two, and so forth.)

Power Law Tail of the Degree Distribution In a pioneering work, Faloutsos et al. [FFF99a] studied the connectivity patterns of the Internet at both the routers and the ASs level. They have discovered that the Internet's degree distribution follows a power law distribution. Later research has concluded that the Internet maintains a heterogeneous degree distribution with a power law tail. Partial views of the Internet obtained from BGP routing points, such as Oregon [Ore] and RIPE [Rip] were used to further investigate The Internet's inner AS structure [CNS+99, MMB00, CCG+02a, BC99, GT00, SARK02]. The power law tail distribution was found very characteristic of complex networks, with the majority of the nodes having a low degree, and a long tail of very high degree nodes. Typical examples include, apart of the Internet, are the World Wide Web connectivity patterns, biological, chemical and social networks. However, it is important to note that not all real network exhibit a degree distribution of a power law tail, and some do show exponential decay or a combination of power law and exponential decay [AB02]. In this work, we use the terms power law distribution and power law tail distribution interchangeably.

Several interesting works have also tried to characterize the growing mechanisms of the Internet and model it [BA99, AB00, BT02, PKP+03]. In 2001, the pioneering work of Albert and Barabási [AB00, BA99] introduced a novel algorithm for the generation of scale free networks. The algorithm was later on implemented in major network generators [JcJ00, MLMB01]. Back in 2001, when the algorithm was just published, we decided to implement it to build a scale free

network generator. The following chapter describes the characteristics of the Internet as a small world network, and the properties of the Notre-Dame algorithm devised by Albert and Barabási, as implemented in our *ToGend* topology generator [Mok01].

2.2 The Internet Topology

The Internet structure is the subject of an extensive research effort lately. It is measured in two different levels, quite different from each other. At the higher level, called the autonomous systems level, the Internet is viewed as a large collection of routing domains. These routing domains, once assigned a unique number, are referred to as autonomous systems. Autonomous systems interact with other peering autonomous systems. Two autonomous systems are considered neighbors if they have some sort of a peering business relations between them. In a graph notation, the ASs are the nodes, and two such nodes are connected with a link if the corresponding ASs have peering relations. At the lower level, called the routers level, the underlying physical structure of the Internet is considered. The nodes represent the actual routers that make up the autonomous systems. Two nodes are connected if there is a physical link between them. Clearly, the number of nodes in a graph representing the underlying physical routers topology of the Internet is bigger by at least a magnitude than the number of nodes in the AS corresponding graph. It was also expected, that the topological structure would be much different.

In 1999, Faloutsos et al. [FFF99a] publish their research, which investigated the characteristics of the Internet structure at both the ASs and routers layers. Their somewhat surprising results were that the Internet connectivity pattern, gathered from several partial views obtained at three different dates, shows a clear power law distribution characteristics at both levels, the AS and the routers. They found that both the rank-degree and the frequency-degree distribution were power law distributions. Two other power laws they found were the number of node pairs within a neighborhood versus neighborhood size (in hops); and eigenvalues of the adjacency matrix versus rank. A later work by Crovella et al. [CNS+99] showed that rank-degree and frequency-degree distributions are two representations of the same distribution, and thus finding only one of them is sufficient.

Albert and Barabási [BA99, AB00] suggested a dynamic graph generation model that generates such networks and aided in the understanding of the evolvement of the Internet. They suggested that such networks growth pattern is the result of preferential attachment and incremental growth. One of their main findings was the self similarity characteristic of such networks. Their results are further discussed in the following section. Medina et al [MMB00] investigated the different factors that influence power law topologies and compared the main topology generators. They came to the conclusion that frequency-degree (along with the rank-degree) distribution is the most effective factor in distinguishing different kinds of topologies. They also found that preferential connectivity and incremental growth to be the main causes for all power laws in their simulations.

Prompted by these findings, Yook et al. [YJBT01] showed that the Internet, at the domain level, exhibits other small world characteristics. Their research, conducted between 1997 to 1999, showed that the Internet's clustering coefficient ranged between 0.18 to 0.3, compared with 0.001 for random networks of similar parameters. The average path length ranged between 3.70 and 3.77 at the domain level, and at the router level it was around 9.

The research outlined above is part of an ongoing effort to discover and map the exact topology of the Internet [BC99, GT00, CCG+02a]. It is generally agreed today that the Internet, at the AS level, has a highly heterogeneous connectivity patterns, with a highly variable vertex degree distribution. In an effort to gain a better understanding of the Internet's overall structure, [SARK02] have investigated its structure from multiple vantage points. They have discovered that the Internet, as a scale free network, also has a hierarchical structure. The core of the Internet consists of a small collection of very high degree ASs, characterized by their very high connectivity to each other. These are the top tier US providers. Their mutual connectivity almost forms a clique. The second tier, consists mainly of the top European and Asian providers, can be characterized as high degree ASs, with a very high connectivity to the core. The rest of the Internet consists of small and small to medium sized ASs, which for the majority of the Internet's ASs.

Several works have also tried to characterize the growing mechanisms of the Internet and model it [BA99, AB00, BT02, PKP+03], and several networks generators which rely on some of these algorithms exist [JcJ00, MLMB01, DMS03] and evaluated [RTY+00, TGJ+02, MSZ02, BT02]. In this chapter we discuss one of them, the Notre-Dame (or Scale Free) algorithm.

2.3 The Notre Dame Topology Generator

In this section we describe the Albert Barabási algorithm for the creation of scale free topologies, termed the Notre-Dame model, or the Scale Free model (SF).

They noted that former network models assume that the network starts with a fixed number N of vertices that are then randomly connected or rewired, without modifying N. In contrast, most real world networks describe open systems that grow by the continuous addition of new nodes. Starting from a small nucleus of nodes, the number of nodes increases throughout the lifetime of the network by the subsequent addition of new nodes. They termed this characteristic *incremental growth*.

Secondly, they noted that former network models assume that the probability that two nodes are connected (or their connection is rewired) is independent of the nodes' degree, i.e. new edges are placed randomly. Most real networks, however, exhibit *preferential attachment*, such that the likelihood of connecting to a node depends on the node's degree.

Thus, the algorithm of the SF model is the following:

Growth: Starting with a small number (m_0) of nodes, at every timestep a new node is added with $m \leq m_0$ edges that link the new node to m different nodes already present in the system.

Preferential attachment: When choosing the nodes to which the new node connects, it is assumed that the probability Π that a new node will be connected to node i depends on the degree k_i of node i, such that

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j} .$$

After t timesteps this algorithm results in a network with $N = t + m_0$ nodes and mt edges. Numerical simulations indicated that this network evolves into a scale invariant state with the probability that a node has k edges following a power-law with an exponent $\gamma = 3$. The scaling exponent is independent of m, the only parameter in the model.

The network growth model, however, is more complicated than the suggested incremental growth model. A variety of local events effect the connectivity pattern of nodes in the network. These local events can be modeled by the following four elements: addition or deletion of nodes, and the addition or deletion of links in the network. In reality, many times it translates to the action of rewiring a link, namely, changing its end points. Therefore, Albert and Barabási suggested an improved version of their algorithm, which consisted of three stages, including the possibility to rewire a link. According to the new algorithm, at each timestep one of the following operations is performed:

Connectivity Growth With probability p, $m \leq m_0$ new edges are added. One end of a new edge is selected randomly, the other with probability

$$\Pi(k_i) = \frac{k_i + 1}{\sum_j (k_j + 1)} .$$

This probability ensures a preferential attachment, since the possibility to attach to a high degree node is larger.

Rewiring Process: With probability q m edges are rewired. For this a node i is randomly selected and the link $l_{i,j}$ that is connected to it is removed, replacing it with a new edge $l_{i,j`}$ that connects node $i$ with node $j`$. Node $j`$ is chosen with the probability $\Pi(k_{j`})$ given above.

Node Growth: With probability $1 - p - q$ a new node is added. The new node has m new edges that with probability $\Pi(k_i)$ are connected to nodes i already present in the system.

The new SF model has the following parameters: m, m_0, p, q. The number of isolated nodes, m_0, determine the core to which the rest of the nodes tend to connect. The smaller m_0 is, the faster these nodes become highly connected and therefore more preferred in the process. m and p are the dominant parameters on determining the richness of the resulted topology, hence the average degree. q determines the factor of local events on the growth of the network, and determines the rate at which nodes can die, i.e., disconnect from the network.

Our *ToGend* topology generator is a straight forward implementation of the new scale free algorithm described above. It includes a simple command line interface, and is capable of generating networks of magnitude of millions of nodes within seconds. The four parameters m, m_0, p, q are read from a parameter file. The output of the *ToGend* generator is a file, which includes the list of degrees followed by the list of links. The nodes are ordered according to the order they were created. The generator was used in most of the works described in following chapters.

Chapter 3

Global Multicast Trees

3.1 Introduction

There are several inhibitors to the commercial use of multicast protocols. While it is clear that multicast is beneficial for transmitting the same information to large groups, its exact gain over unicast has not yet been determined [CS98, PST99, CA01]. Network suppliers lack a fast and efficient way to estimate the size of large multicast groups, and the research community lacks reliable tree models.

We present here a thorough investigation of the structure and characteristics of multicast trees cut from generated power law topologies and the Internet. While the exact nature of the Internet topology is in debate ([CCG+02b]), our results show that the partial views we have from the Internet obey the power law tail of frequency-degree found by [FFF99a]. These results were also verified by [GT00, MMB00, CNS+99], who conducted further investigations. Moreover, trees cut from the Internet and from the generated topologies had similar characteristics.

We found that trees cut from[1] such topologies and the Internet obey a degree-rank and sub-tree size-rank power law distributions[2]. We also found that the distance distribution of nodes from the root node resembles a Gamma distribution, as shown previously for the Internet [CNS+99]. We observed that nodes with degree higher than five tend to be rare in the resulting trees. These high degree nodes can always be found in several adjacent rings, which reside typically at the core of the network, and in the near vicinity of the tree root.

Our most intriguing result finds a linear ratio between the number of high degree nodes in the tree and the number of clients[3]. The result is shown to be valid for trees cut from scale-free topologies that were generated with various parameters, as well as for experiments conducted on the Internet itself. We further verify this ratio analytically for power law trees. Based on the tree topological characteristics we found, we suggest the Fast Algorithm for estimating the size of large multicast groups. We analyze the algorithm's expected delay in the Internet, which sums up to less than the round trip delay from the root node of the tree to a random client at the edge of the network.

[1] We use the term *trees cut from the network* to describe a process where we select a root node from the network; a group of receivers; and the sub graph containing all nodes and links that comprise the tree of shortest paths connecting them

[2] Note that rank-degree and frequency-degree power laws can be derived from each other [MMB00].

[3] We note by clients the group of routers that directly attach clients.

Estimating the population size of large multicast trees can improve the performance of feedback mechanisms of protocols such as RTP [SCFJ96] and SRM [FJM+95]. Current feedback suppression solutions for RTCP use timers at the receivers [RS98, NB99]. Our sender based estimation produces a much faster estimation that can be propagated to the receivers and eliminate the need for such timers. Often, feedback suppression protocols are based on similar techniques as polling based estimation algorithms [BTW94, NB98, FT99] and thus can use our faster estimation instead. Fast estimation may also be beneficial to forward error correction protocols [RKT98].

Our suggested estimation algorithm offers an alternative approach by using the topological characteristics to obtain an estimation on the number of receivers (rather than a specific population count). It does not aggregate information at the router level, but rather polls the high degree routers in the multicast tree. Our results show that paths from the root of the tree to its receivers are very likely to pass through the core of the network; We also observed that high degree routers tend to reside within the core or in its close vicinity. Hence, the polled high degree nodes will be closer to the root than the receivers they connect. The algorithm adapts itself to dynamic topological changes, and can therefore reflect changes in the session size, as does the population sampling algorithm suggested in [AAN02].

To the best of our knowledge, this is the first time that the existence of a power law in the underlying topology is leveraged to construct an algorithm. We believe that more such algorithms can be developed in the future for a variety of purposes.

The second part of this chapter discusses our findings on the tomography of Multicast trees, and is part of a joint work with physicists from Bar-Ilan University, who have studied the tomography of the Internet. We used the analytical model they devised and backed it up with simulation and real network data results. In addition, we further investigated these phenomena on multicast trees.

The work uses the Molloy Reed graph generation method [MR98] in conjunction with similar techniques to study the layer structure (tomography) of networks. Specifically, the work studies the number and degree distribution of nodes at a given (shortest path) distance from a chosen network node. It is shown analytically that the distance distribution of all nodes from a specific network node consists of two regimes. The first can be described as a very rapid growth, while the second is found to decay exponentially. It also shows that the node degree distribution at each layer obeys a power law with an exponential cut-off. The analytical derivations are backed with simulations, and it is shown that they match.

We also study shortest path trees cut from scale free networks, as they may represent the structure of multicast trees. We investigate their layer structure and distribution. We show that the structure of a multicast tree cut from a scale free network exhibits a layer behavior similar to the network it was cut from. We validate our analysis with simulations and real Internet data.

As noted by Lakhina et al [LBCX03], it is a significant challenge to test and validate hypotheses about the Internet topology in a lack of highly accurate maps. The analysis results suggest a simple local test for the validity of the power law model as an exact model of the Internet. Indeed our findings show that there is a good agreement of the empirical and analytical results. The slight difference we had can be attributed to bias in data collection and to second order phenomena such as, degree correlation, hierarchies, and geographical considerations.

Name	Type	Parameters	No. of Nodes	Avg. Node degree
VS	generated	$a = 1; p \in 0 : 0.05 : 0.5$	10000	$1.99 - 3.98$
IS	generated	$a = 2; p \in 0 : 0.05 : 0.5$	10000	$3.99 - 7.9$
LS	generated	$a = 3; p \in 0 : 0.05 : 0.5$	10000	$5.98 - 12.04$
Big IS	generated	$a = 1.5, 2; p = 0.1$	50000;100000	3.3,4.4
BL[1,2]	real data	–	Internet	3.2 [5]
LC	real data	–	Internet	3.2 [6]

Table 3.1: Type of underlying topologies used

3.2 Empirical Characteristics of Multicast Trees

This section details our findings on the structure of multicast trees cut from generated power law topologies, as well as the Internet. These findings are the basis for the estimation method we present in Section 3.3, and are of interest in their own right.

Little work has been done on modeling and characterizing multicast trees. Chalmers and Almeroth [CA01] investigated the branching characteristics of Internet multicast trees on the MBone and their impact on multicast efficiency. They found that multicast trees tend to have low average internal degree that grows logarithmically with the number of receivers in the tree, and a maximum height of approximately 23 nodes. They also found a high frequency of "relay" nodes that have a degree of two throughout the tree. In previous work, Pansiot and Grad, who constructed trees from a graph based on true routing paths in the Internet, also showed a high frequency of relay nodes in the tree graphs [PG98].

3.2.1 Topology and Tree Generation

Our method for producing trees is the following. First, we generate power law topologies based on the Notre-Dame model [AB00]. The model specifies 4 parameters: a_0, a, p and q [4]. Where a_0 is the initial number of detached nodes, and a is the initial connectivity of a node. When a link is added, one of its end points is chosen randomly, and the other with probability that is proportional to the nodes degree. This reflects the fact that new links often attach to popular (high degree) nodes. The growth model is the following: with probability p, a new links are added to the topology. With probability q, a links are rewired, and with probability $1 - p - q$ a new node with a links is added. Note that a, p and q determine the average degree of the nodes. We created a vast range of topologies, but concentrated on several parameter combinations that can be roughly described as very sparse (VS), Internet like sparse (IS) and less sparse (LS). Table 3.1 summarizes the main characteristics of the topologies used in this paper.

From these underlying topologies, we create the trees in the following manner. For each predetermined size of client population we choose a root node and a set of clients. Using Dijkstra's algorithm we build the shortest path tree from the root to the clients. To create a set of trees that realistically resemble Internet trees, we defined four basic tree types. These types are based on the rank of the root node and the clients nodes. The rank of a node is its location in a list of

[4] The notations in [AB00] are m_0, m, p and q, respectively.

descending degree order, in which the lowest rank, one, corresponds to the node with the highest degree in the graph. For the case of a tree rooted at a big ISP site, we choose a root node with a low rank, thus ensuring the root is a high degree node with respect to the underlying topology. Then, we either choose the clients as high ranked nodes, or at random, as a control group. Note, that due to the characteristic of the power law distribution, a random selection of a rank has a high probability of choosing a low degree node. The next two tree types have a high ranked root, which corresponds to a multicast session from an edge router. Again, the two types differ by the clients degree distribution, which is either low, or picked at random.

The tree client population is chosen at the range $[50, 4000]$ for the 10000 node generated topology, $[50, 10000]$ for the 100000 node generated topology, and $[500, 50000]$ for the trees cut from real Internet data. For each client population size, 14 instances were generated for each of the four tree types. All of our results are averaged over these instances. The variance of the results was always negligible.

There are two underlying assumptions made in the tree construction. The first, is that the multicast routing protocol delivers a packet from the source to each of the destinations along a shortest path tree. This scenario conforms with current Internet routing. For example, IP packets are forwarded based on the reverse shortest path, and multicast routing protocols such as Source Specific Multicast [HC02] deliver packets along the shortest path route. In addition, we assume that client distribution in the tree is uniform, as has been shown by [PST99, CA01].

3.2.2 Tree Characteristics

Degree-Rank and Size-Rank Power Laws

Our results show that trees cut from a power law topology obey a similar power law. Specifically, we compared the degree-frequency power law found by [FFF99a]. Figure 3.1 shows in log-log scale the degree frequency plot for 10000 nodes topology generated with the parameter set $a_0 = 6, a = 1, p = 0.3, q = 0$. The dotted lines here, and in the rest of the linear fit figures, mark the 95% confidence interval.

Figure 3.2 shows the same plot for a multicast tree with 500 low degree clients and a root with a high degree. In Table 3.2 we summarize the best linear fit parameters in a log-log scale for all trees generated for the topology set $a_0 = 6, a = 2, p = 0.1, q = 0$. It can be seen that the power law holds even for very small trees, e.g., for a tree with 50 multicast clients that has on the average around 200 nodes. The same phenomenon appears in all the trees cut from all topologies, regardless of the way the root and the client nodes were chosen.

These findings conform with the findings of [CA01, PG98] who found a very large frequency of relay nodes in the trees, i.e., nodes with a degree of two. In a power law relationship of frequency and degree, the frequency of two degree nodes is the highest in the tree. Leaf nodes are determined by clients, and are a subset of the clients.

We also found that the distribution of degrees at a specific distance from the root, i.e., in a certain depth ring, also showed a power law distribution of degree-rank, but with different slopes.

Given the above findings, it is important to note the following. Cohen et al. [CEbAH00b] showed that the maximal node degree in a graph of N nodes is proportional, for Internet-like

[5] based on [KRS00]

[6] based on [BC99]

topologies, to approximately the square root of the number of nodes. More precisely, $D_{max} \sim N^{\frac{1}{\alpha-1}}$, where α is the exponent of the degree-frequency power law of the topology. Hence, all resulted degree-frequency graphs of finite sizes exhibit a cut-off at the tail. This holds true for partial views taken from the Internet, with the cut-off being a result of the partiality as well as from the finite size of the Internet itself.

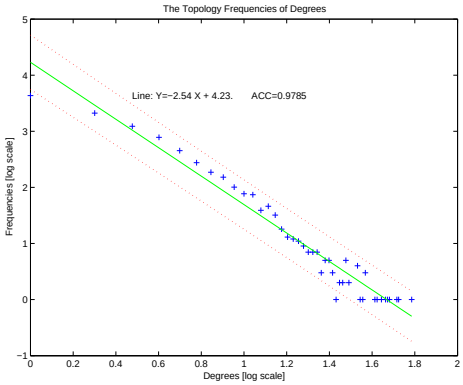

Figure 3.1: Frequency of degrees for a 10000 node topology with $a_0 = 6, a = 1, p = 0.3, q = 0$.

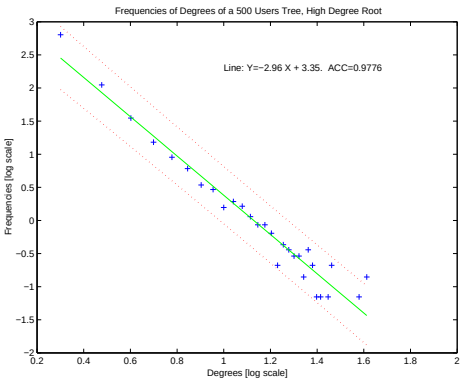

Figure 3.2: Tree with 500 low degree clients, high degree root. Cut from topology $a_0 = 6, a = 1, p = 0.3, q = 0$.

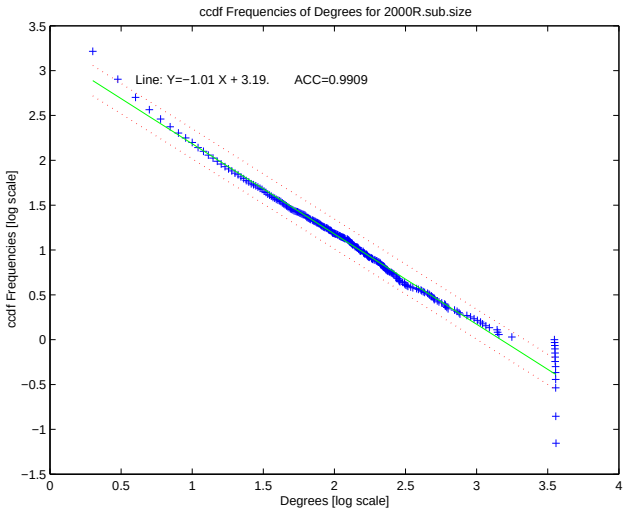

Figure 3.3: Sub-tree size CCDF distribution, for a 2000 node tree cut from topology $a_0 = 6, a = 2, p = 0.1, q = 0$.

The second power law we found for the trees is of frequency and size of the sub-trees in each tree. Namely, the self similarity holds not only for the degree distribution in the tree, but also for its inner structure. Figure 3.3 shows the excellent fit of the complementary cumulative distribution

function of the sub-tree sizes of a 2000 Node tree. The tree, with a high degree root, is cut from a 10000 node topology with the parameter set $a_0 = 6, a = 2, p = 0.1, q = 0$. The size distribution differs from the degree distribution in that the big sub-trees, although almost similar in size, may differ by one or two nodes, which is negligible compared to their overall size. Thus we give the ccdf graph, which plots the probability that the observed values are greater than the ordinate. It can be seen that the fit to a power law is over 99%. The slope computed for the PDF graph without the tail, resembles the one of the degree distribution.

	a	p	Y	ACC
topology	2	0.1	-2.50X + 4.49	0.9721
	High degree root, low degree clients		Root and clients chosen randomly	
Receivers	Y	ACC	Y	ACC
50	-2.76X + 2.25	0.9337	-3.27X+2.68	0.9752
100	-2.64X + 2.42	0.9613	-2.96X+2.71	0.9611
300	-2.50X + 2.73	0.9730	-2.64X+2.85	0.9717
500	-2.58X + 2.97	0.9732	-2.58X+2.96	0.9654
750	-2.57X + 3.12	0.9825	-2.59X+3.09	0.9609
1000	-2.56X +3.23	0.9785	-2.59X+3.21	0.9728
1500	-2.64X + 3.45	0.9812	-2.56X+3.32	0.9741
2000	-2.58X + 3.52	0.9858	-2.60X+3.44	0.9620
2500	-2.65X + 3.66	0.9817	-2.63X+3.57	0.9731
3000	-2.66X+ 3.75	0.9851	-2.58X+3.57	0.9670
4000	-2.70X + 3.90	0.9825	-2.64X+3.73	0.9611

Table 3.2: Linear fit of degrees and frequencies

Per Degree Distance Distribution

Cheswick at al. [CNS+99] found that the distribution of the number of nodes at a certain distance from a point in the Internet is similar to the Gamma distribution. Our results show that the distribution of distance from the root of nodes of a certain degree seems close to a gamma distribution, although we did not determine its exact nature. Figure 3.4 shows the distribution of the distance of two to five degree, leaf and high degree nodes, where high degree nodes are nodes with a degree six and higher. In this case the root is a low degree node, and the tree has 1000 low degree clients. As can be seen, the high degree nodes tend to reside much closer to the root than the low degree nodes, and in adjacent rings. In this example, most of them are in the second to forth depth rings around the root.

This phenomenon was even more obvious when the root was a high degree node. We found the following observation with regard to power law generated topologies. The high degree nodes seem to form a 'core' with a low diameter (around five hops for trees cut from the generated topologies, and seven for trees cut from Internet data) and most of the other nodes in the network are not distanced more than three to five hops away from this core. Subramanian et al. [SARK02] observed a similar phenomenon at the Internet AS topology, although obtained from directed BGP routing tables.

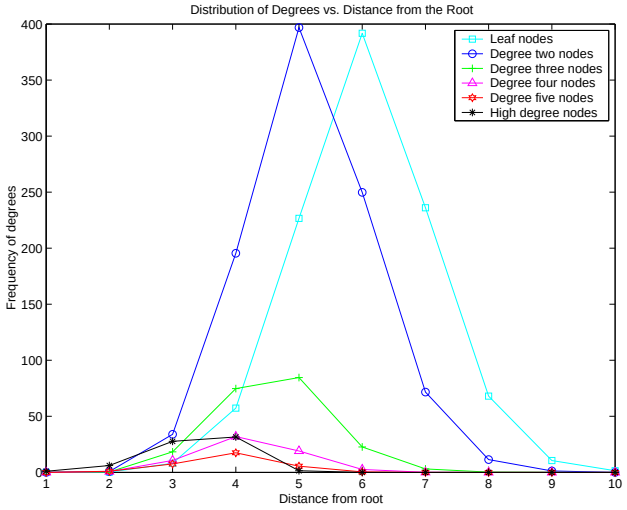

Figure 3.4: Distribution of the distance of high degree, two to five degree and leaf nodes in a tree cut from topology $a_0 = 6, a = 1, p = 0.3, q = 0$

The distribution of client distances from the tree root is given by the leaves distances in Figure 3.4. Note that the longest path to a client is the tree height. Our results show that the less connected the underlying topology, the taller is the average tree cut from the topology.

Empirical Results from Internet Data

We verify the above findings with results obtained from real Internet data. Our results are verified on two different data sets. The first is an Internet partial view at the routers level, obtained from the Lucent Internet Mapping Project [BC99]. We used this data set as the underlying topology, from which we cut trees in the same manner described in Section 3.2.1. We denote this topology by *LC*.

For the second data set we use the client population of `www.bell-labs.com` which is a medium size web site. This may represent the potential audience of a multicast of a program with scientific content (such as the livecast of the INFOCOM conference). From this set two lists of clients were obtained, and traceroute was used to determine the paths from the root to the clients. It is important to note, that the first three levels of the tree consist of routers that belong to the site itself, and therefore might be treated as the root point of the tree, although in these graphs they appear separately. We denote this tree as the *BL* tree.

Figure 3.6 shows the frequency of degrees for a 10000 node tree cut from the LC topology. The tree, which is an average of 14 instances, exhibits a clear degree-frequency power law with a good fit[7]. The tree was chosen with a high degree root, and low degree leaf nodes. The variance of the instances of each tree was negligible, and the same result was obtained for each of the generated trees, with as low as 1000 clients and as high as 50000. Figure 3.23 shows the frequency of degrees

[7]We fit the data for the points above the line Y=0 which capture all the degrees that appear on average, at least, once in every tree. To extend the fit below this line we need more trees. If we want to get rid of the noisy tail all together we need to generate, at least, an order of 10^4 trees as our fit predicts that the highest degree points will appear on the average in less than one of every 10^3 trees.

for the BL tree. The linear fit of the log-log ratio is excellent, with a correlation coefficient of 0.9829.

Figure 3.5 shows the ccdf of the sub-tree sizes of a tree with 7000 clients cut from the LC data. The root is a high degree node, and the clients are low degree nodes. Note, that every point in the graph is the result of an average of 14 instances therefore the tail was omitted from the fit. The size-rank power law appears in all the trees cut from this data.

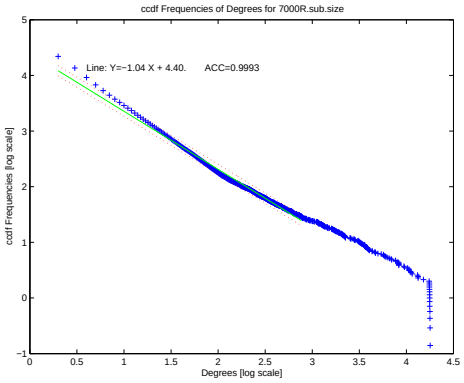

Figure 3.5: Size distribution of a 7000 clients tree cut from the LC data

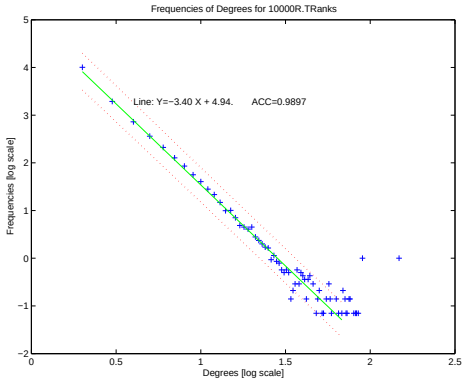

Figure 3.6: Frequency of degrees of a 10000 node tree cut from the LC Internet data.

Figure 3.8 shows the distribution of the distance of two degree, leaf and high degree nodes, for a 15000 client tree, cut from the LC data. The majority (90%) of the high degree nodes reside within a distance of eight hops from the root, while the clients are distanced up to 18 hops from the root.

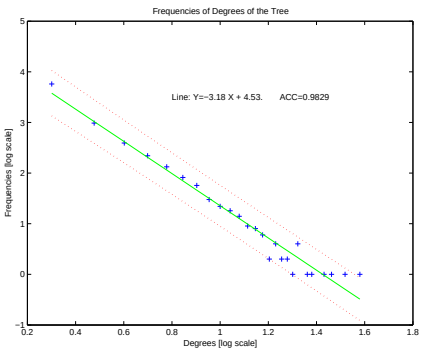

Figure 3.7: Frequency of degrees of the BL Internet tree.

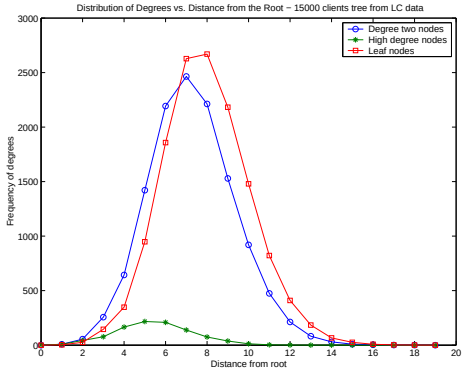

Figure 3.8: Distance of two, high degree and leaf nodes from the root of a 15000 client tree cut from the LC Internet data

3.3 Receiver Group Size Estimation Method

While all of the above observations are interesting and help in our understanding of multicast trees, we were intrigued whether we can use any of this knowledge to evaluate the size of a multicast tree. We compared the degree of the nodes in the tree to their degree in the topology, and focused on the high degree nodes. Interestingly, we found that while some nodes had a tree degree that is significantly smaller than their degree in the underlying network topology, other nodes seemed to have a tree degree close to their network degree. We then compared the frequency of nodes with degree i and above (high degree nodes) to the number of clients in the tree, and found a linear ratio with a correlation coefficient of not less than 0.99. We term this ratio the HCN_i ratio (hubs-to-client number ratio), and define it as follows: Let H_i be the group of routers, or hubs, with degree $d \geq i$ in the tree; Let C be the group of receivers, or clients, that the tree span; Than,

$$HCN_i = \frac{H_i}{C}.$$

Next, we outline our findings on HCN_i ratio for both simulated trees and trees cut from the real Internet. We proceed by giving a mathematical analysis of our results for power law trees.

3.3.1 Empirical Findings

We have found that an HCN_6 ratio of 1:16 is a very good predictor for trees cut from the Internet, and most generated topologies. Figure 3.9 shows the HCN_6 ratio in trees cut from a 100000 node topology. The topology parameters are $a_0 = 6, a = 1.5, p = 0.1, q = 0$, and the root node of all trees is a high degree node. The linear ratio is obtained after gathering the information from not more than five depth rings around the root, where the $j - th$ depth ring around the root is comprised of all nodes at distance j hops from the root. We plotted the frequency of high degree nodes obtained after scanning three, four, five, six and nine depth rings around the root. As can be seen from the graph in Figure 3.9, the entire information was obtained until the sixth depth ring - the following rings did not add any more information. The HCN_6 ratio was found to be 16. Figure 3.10 shows the excellent fit of the HCN_6 ratio with a correlation coefficient of 0.9998. When we plotted the data for trees cut from this topology with a low degree root, we obtained very similar results. The ratio was again 16, with a correlation coefficient of 0.9996. However, another depth ring was needed to obtain accurate results, since the root was not as close to the core of high degree nodes as in the previous case.

We verified our results using actual Internet data on the client population of the Bell-Labs web site described in Section 3.2, and on trees cut from the data from Cheswick's Lucent Internet mapping project, noted LC, also described there. The Bell-Labs client population data contains two log files. The first, denoted BL1, has 10897 clients and the second, BL2, has 7356. We created subsets of clients by randomly selecting entries from the log files, and cut the corresponding trees for these subsets from the original trees. Figure 3.11 shows the ratio between the 16 predictor and the actual number of clients in the generated trees. For BL1 the ratio was $1{:}16^{0.99}$ with a fit of 99.75%, for BL2 the ratio was $1{:}16^{1.04}$ with a fit of 99.72%. For client populations larger than roughly 1500 clients the predictor of 16 gives an excellent estimate - within 9% of the actual number of clients.

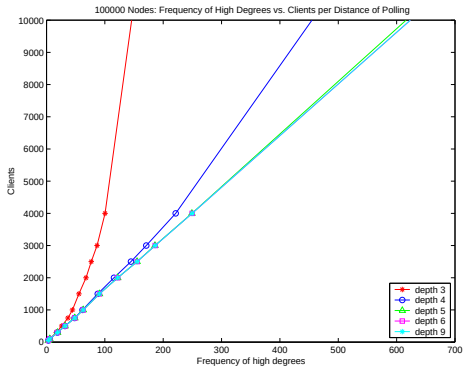

Figure 3.9: Clients vs. frequency of high degree nodes. Cut from a 100000 nodes topology with $a_0 = 6, a = 1.5, p = 0.1, q = 0$.

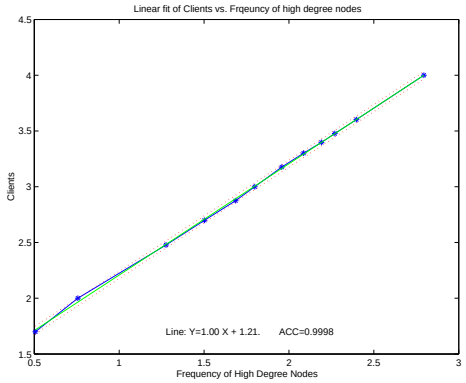

Figure 3.10: Linear fit of Clients vs. frequency of high degree nodes. Cut from a 100000 nodes topology with $a_0 = 6, a = 1.5, p = 0.1, q = 0$.

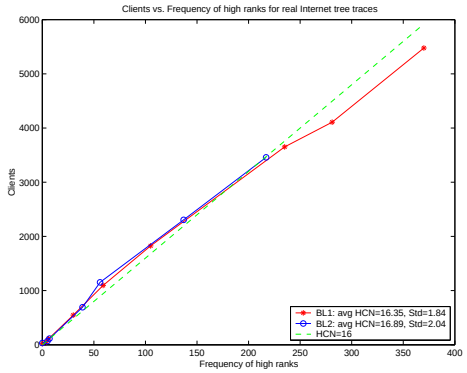

Figure 3.11: Clients vs. high degree nodes and the HCN predictor for the BL[1,2] trees

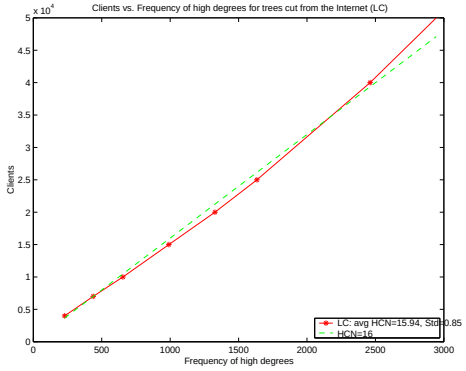

Figure 3.12: Clients vs. high degree nodes and the HCN predictor for the trees cut from the LC topology

The LC data gives a partial view of the Internet at the router level with more than 110000 routers. From this topology, we cut trees in the same manner described in Section 3.2. Again, each result is averaged over 14 instances. Figure 3.12 shows the ratio between the number of clients and high degree nodes, compared with the predicted value from the simulations, 16. The average value of the ratio is 15.89, with a standard of deviation of 0.9. Hence, a 16 predictor for the ratio gives a very good estimation for this data also.

For the generated topologies and the Internet experiments, our results are less definite for very small trees. We found that HCN_6 ratio=16 is accurate when client population is at least 0.1% the size of the underlying topology. Nevertheless, for the Internet, our experiments yielded very good results for group sizes of 1500 clients and more. Note that when the group size is small enough, exact counting of the clients can be done with reasonable cost.

While a predictor of 16 was shown to be a very good predictor for large groups, it becomes less scalable when the group size is extremely large. For example, in the case of a multicast tree with a million clients, the expected number of high degree nodes is 62500. A good solution for this problem is to increase the degree of the sample nodes. For example, in the case of very large groups, counting the number of nodes with degree higher than nine will produce an accurate prediction, with a ratio of 1:48. Note that sampling nodes with a larger degree gives us a coarser estimation. Our experiments show that when we sample nodes of degree ten and above the estimation is accurate only for group sizes of at least 1.5% the size of the underlying topology. Remember that sampling nodes of degree 6 and above yields a good estimation for trees as small as 0.1% of the network.

3.3.2 Analytical derivation of HCN_6 ratio

In this section, we derive the HCN_6 ratio for trees in power law topologies. Our experiments have shown that the group of leaf nodes of a tree closely approximates the tree's client population. For simplicity we take the exponent of the underlying topology degree probability instead of the tree's, but these are fairly close.

Given a tree with N nodes, we denote by L the number of leaf nodes and by $\tilde{N}$ the number of non leaf nodes. Let $\tilde{\mathcal{N}}$ be the group of non leaf nodes. The average internal degree is defined by: $r = \frac{\sum_{j\in\tilde{\mathcal{N}}} d_j}{\tilde{N}}$ where d_j is the degree of node j. But by its definition it also holds that $\sum_{j\in\tilde{\mathcal{N}}} d_j = 2\tilde{N} + L - 1 \approx 2\tilde{N} + L$, and $\sum_{j\in\tilde{\mathcal{N}}} d_j = N + \tilde{N} - 1 \approx N + \tilde{N}$. Given all the above we can write

$$L = N \cdot \frac{r-2}{r-1}. \tag{3.1}$$

which holds for any tree.

Given that p_i is the probability to find a node with degree i in the tree we can rewrite the above expression for r

$$r = \frac{1}{1-p_1} \cdot \sum_{i=2}^{N} i \cdot p_i. \tag{3.2}$$

and the probability conservation equation

$$\frac{L}{N} + \sum_{i=2}^{N} p_i = 1. \tag{3.3}$$

Substituting (3.1) in equations (3.2) and (3.3), and given that the degree distribution obeys the power law $p_i = c \cdot i^{-\alpha}$, we get that:

$$r = \frac{S_1}{S_2} \quad ; \quad c = \frac{r}{S_1 \cdot (r-1)}. \tag{3.4}$$

Where $S_1 = \sum_{i=2}^{N} i^{-(\alpha-1)}$ and $S_2 = \sum_{i=2}^{N} i^{-\alpha}$.
The HCN_6 ratio is defined by:

$$HCN_6^{-1} = \frac{\sum_{i=6}^{i=N} p_i \cdot N}{L}. \tag{3.5}$$

Plugging (3.1) and (3.4) in equation (3.5) yields

$$HCN_6^{-1} = \frac{(1 - S_3) \cdot (r - 1)}{r - 2} - 1. \tag{3.6}$$

Where $S_3 = \sum_{i=2}^{i=5} i^{-\alpha}$.

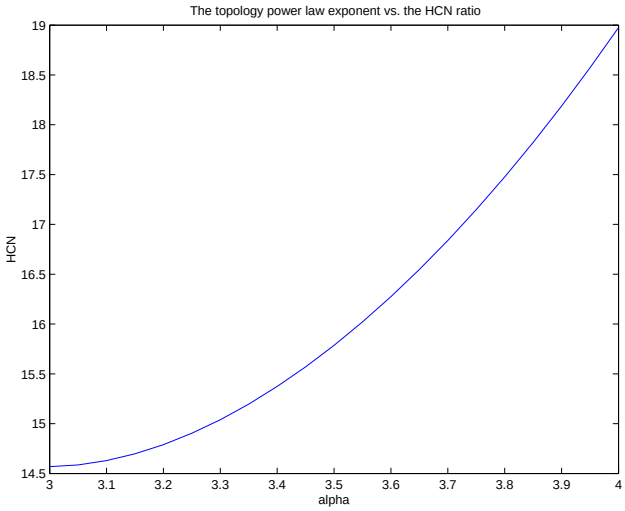

Figure 3.13: The change in HCN_6 ratio with α

Figure 3.13 shows how the HCN_6 ratio in equation (3.6) changes with α. For $3 \leq \alpha \leq 4$ the HCN_6 ratio changes between 14.5 and 19. Hence, a precise value for the tree's α will yield an excellent evaluation of the number of leaf nodes in the tree, and hence a good estimation to the client population. Nevertheless, our results show that for the shortest path trees cut from the Internet, as well as from most of our generated topologies, *HCN_6 ratio = 16* gives a very good estimation. Understanding the precise correlation between our empiric and analytical results may lead to a deeper understanding of the Internet topology, and is the subject of our next work.

3.4 Estimation Algorithms

3.4.1 A Basic Algorithm

The findings in the previous section give rise to an algorithm for estimating the number of clients in a multicast tree, in which the number of nodes with six or more child nodes can be counted. The main idea, given formally in Figure 3.14, is that the root multicasts a feedback request, Req, along the multicast tree. The request carries the parameter d, which indicates the minimal node degree that needs to report back. Such a node, upon receiving the request, replies with a UDP Rep packet sent directly to the root. The root waits for a time long enough to ensure that most replies are accepted. The root then counts the number of different replies it receives, and by multiplying with the appropriate coefficient produces the estimate.

Algorithm 1 (Basic)

1. Send $Req(d)$
2. $n \leftarrow 0$
3. Activate Timer$_1(T_{d_1})$

When Rep arrives

4. $n{+}{+}$

When TimeOut$_1$

5. return($c_d \cdot n$)

Figure 3.14: A formal description of the basic algorithm for the root node.

Note that for the Internet, T_{d_1}, the time the root waits for the replies to arrive, should be quite large. Specifically, T_{d_1} needs to be long enough such that the vast majority of slow responses due to round trip and processing delays are not lost. (We assume that T_{d_1} of several seconds satisfies these requirements.)

3.4.2 Fast Algorithm

The Fast Algorithm, formally presented in Figure 3.15, is motivated by the need to obtain a fast estimation on the client population. We would like to determine the termination rule in a way that guarantees that a significant portion of the Rep messages has already arrived. In the basic algorithm we achieve this by setting a very large timeout. Here, we monitor the Rep message arrival process to achieve this goal.

We start the algorithm with an *initial sampling period*, T_{d_1}, whose purpose is to enable responses from the high degree nodes in the k-neighborhood of the root to arrive back at the root. If by the end of the initial sampling period the root receives no replies, it assumes the group is either very small or inactive. If the root receives Rep messages, a shorter sampling period termed the *iterative sampling period* is activated repeatedly until the termination condition is satisfied. The purpose of the iterative sampling period, noted T_{d_2}, is to enable the algorithm to converge to a good estimate within a short time.

There are several options to determine a termination condition based on the Rep message arrival process. We can choose a threshold and stop when the message arrival rate drops below it. This solution, however, is not immune to network jams, and is very sensitive to the threshold's value. Another option is to stop when the rate keeps dropping for several successive iterative sampling periods. In this case, the algorithm is very sensitive to the length of the iterative sampling period. If it is too short the algorithm might terminate too early with a large estimation error. On the other hand, a long iterative sampling period might cause the algorithm to run longer than necessary.

Thus, we devised a termination rule (see line 12 in Figure 3.15) that can self-tune according to the arrival process. Under reasonable conditions it will guarantee termination within a preset estimation error. The algorithm terminates when the number of replies received at the root during one of the iterative sampling periods *does not* improve the estimation by more than K_{th}, where K_{th} is the estimation error. For example, setting the iterative sampling period to the average two-hop

Algorithm 2 (Basic)

1. Send $Req(d)$
2. $n \leftarrow 0$
3. $ndt \leftarrow 0$
4. Activate Timer$_1(T_{d_1})$

When TimeOut$_1$

5. if $ndt = 0$ then
6. return(0)
7. else
8. Activate Timer$_2(T_{d_2})$
9. $n \leftarrow ndt$
10. $ndt \leftarrow 0$

When TimeOut$_2$

11. $n+ = ndt$
12. if $ndt \leq K_{th} \cdot n$
13. return($c_d \cdot n$)
14. else
15. Activate Timer$_2(T_{d_2})$
16. $ndt \leftarrow 0$

When Rep arrives

17. $ndt{+}{+}$

Figure 3.15: A formal description of the Fast Algorithm for the root node.

delay and the initial sampling period to $2T$, causes the algorithm to terminate when the replies gathered from the $T + i$-th depth ring, at the ith iterative sampling period, do not improve the estimation by more than K_{th}. Under reasonable network conditions, about half of the replies from this depth ring reach the root node by the end of the ith iterative sampling period. Thus, the termination condition enables the algorithm to stop when it identifies the end of the adjacent depth rings around the root.

Performance Evaluation of the Fast Algorithm

In this section we estimate the delay of the Fast Algorithm and define the average values for T_{d_1} and T_{d_2}. The delay of a packet traversing a single link, d, is comprised of two components: $d = \Delta + q$, where Δ is the fixed minimum link delay and q is a random variable representing the queuing delay, which is exponentially distributed. We would like to derive the distribution of the queuing delay of a packet traveling h links. The density function of the delay, $d_h(t)$, is a convolution of the

density functions of $q(t - h\Delta)$, h times:

$$d_h(t) = q(t - h\Delta) * q(t - h\Delta) * \cdots q(t - h\Delta). \tag{3.7}$$

Let us define, for simplicity:

$$\tau = t - h\Delta. \tag{3.8}$$

Thus, $d_h(\tau)$ is a gamma random variable with parameters h and λ. Namely:

$$d_h(\tau) = \frac{\lambda^h \tau^{h-1} e^{-\lambda\tau}}{(h-1)!}. \tag{3.9}$$

Where λ^{-1} is the average queuing delay. Assuming that all high degree nodes reside within h hops from the root node of the tree, and let the probability of a high degree node to reside at distance h from the root be $p_{hd}(h)$, from Equations (3.7) and (3.9) we get that the probability distribution function of the *total* delay is:

$$D(\tau) = \sum_{i=0}^{h} D_h(\tau) p_{hd}(i) = \sum_{i=0}^{h} \frac{\lambda^i \tau^i \Gamma(i, i\tau)}{(i\tau)^i} p_{hd}(i). \tag{3.10}$$

Where $\Gamma(\cdot, \cdot)$ is the incomplete gamma function [Knu97, sec. 1.2.11]. Plugging back (3.8) in (3.10) we get that the final form of the total delay probability distribution function is:

$$D(t) = \sum_{i=0}^{h} \frac{\lambda^i \Gamma(i, i(t - h\Delta))}{i^i} p_{hd}(i). \tag{3.11}$$

The values of T_{d_1} and T_{d_2} need to be established in a way that will ensure that the majority of the replies are gathered. For example we can select T'_{d_1} to the value of t that minimizes $D(t) = 0.5$, meaning that ensures that on the average we wait for half of the replies to be done waiting at queues.

Alternatively we should chose T_{d_1} to be long enough for each node to at least reach the core, preferably its center. Let us define by r_c the estimated radius of the core, in which we have established that most high degree nodes reside. Let us define by r_e the average distance from an edge node to the core. Then,

$$T_{d_1} = 2(r_c + r_e)(\Delta + \bar{q}) \tag{3.12}$$

Thus ensuring that T_{d_1} is sufficient for the request to reach the core vicinity and for some of the replies of high degree nodes to arrive back to the root. In the same manner, setting:

$$T_{d_2} = 2(\Delta + \bar{q}) \tag{3.13}$$

yields an iterative sampling period of one hop round trip delay, thus enabling the algorithm to obtain most of the information from the next hop. From our experiments, as described in Section 3.2, we discovered that the values of $r_c = 7$ and $r_e = 6$ are sufficient for today's Internet.

In Table 3.3 we summarize the simulation results of the Fast Algorithm. We denote by τ the average one hop delay. The hop delay is either normally distributed (ND) or exponentially distributed (ED). The length of the initial sampling period is 8τ, and the length of the iterative sampling period is 2τ. The results in this table are obtained for trees cut from topology $a_0 = 6, a =$

$1, p = 0.3, q = 0$, and the Fast Algorithm was executed with an estimation error of $K_{th} = 2\%$. All the high degree nodes in the generated trees reside within five depth rings from the root. Time units are in $[\tau]$. Note that due to the long tail of the exponential distribution, an iterative sampling period of 2τ is shown to be too short, since the exponential case represents a bursty network. However, when the delay is normally distributed with variance τ, the algorithm counts all of the high degree nodes in the tree within less than 12τ time units, which is less than the measured average clients' round trip delay of 16τ for these trees.

Clients	300	500	750	1000	1500	2000	3000	4000
ND prediction	304	512	736	992	1472	2000	2992	4000
ND time	10.0	12.0	10.0	10.0	10.0	10.0	12.0	12.0
ED prediction	256	400	672	960	1456	1920	2736	3856
ED time	12.0	12.0	18.0	14.0	20.0	18.0	16.0	20.0

Table 3.3: Fast Algorithm time and prediction

3.5 Discussion

Our results, which show a strong correlation between the number of high-degree nodes and the number of clients, hold for all tree types over all tested power laws topologies. As stated before, all of the results obtained from the simulations as well as the LC data were averaged over 14 instances. When degrees six and higher are chosen (i.e., $d = 6$), we found that 16 is a very good predictor in the average case. In this section we discuss the accuracy of this result for specific trees.

We examined the specific predictors of the 14 instances of a 7000 clients tree cut from the LC data. The smallest ratio was 15.52 and the largest 16.78, yielding a maximal error of 5%. Figure 3.16 shows our results for 14 trees that were cut from a 100000 node topology. The root is a randomly chosen high-degree node and the clients are chosen uniformly. The figure legend details for each of the trees its specific slope, i.e., its average ratio between the number of clients and the high degree nodes over all points. It also specifies for each tree the maximal and minimal deviation points, i.e., the ratio at the points which are furthest from the average for that tree. We can see that the slopes of most of the trees are within 10% of the average predictor. This phenomenon can be seen throughout the different tree types. The worst deviation from the average predictor of a slope was 12.5%. A few points diverge up to 30% from the estimation, yet this should be expected, given the statistical nature of the estimation method.

We found that the reliability of the prediction increases with the group size. According to our findings, described in Section 3.3, the found predictor is accurate only for medium to large groups. When group size exceeds 1000 clients, the average predictor yields very good estimations, with not more than a 10% error. For the general case, for all group sizes, the vast majority of the individual test points are within a marginal limit of 15%. For our analysis on Internet logs the estimation error was no more than 15% in almost all cases. The single exception was for a group of size 1153, which exhibited a 22% estimation error.

We have found that instances of a tree with the same root node tend to have a more stable behavior. Thus, a root can calibrate the estimator for its trees by counting the number of clients and

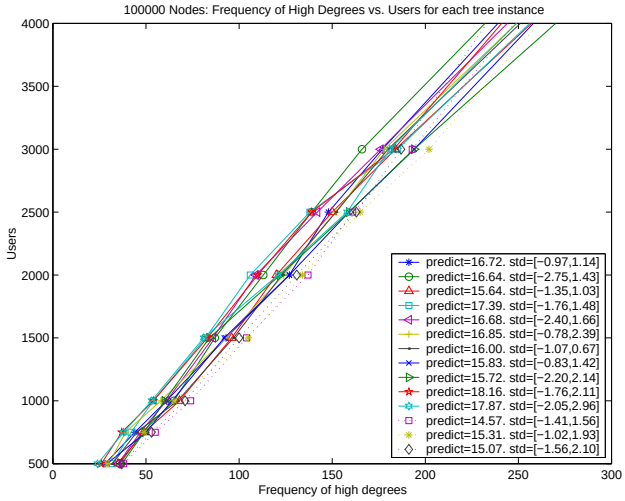

Figure 3.16: Clients vs. high degree nodes for each of the 14 instances of the tree

the number of high degree nodes when the trees are reasonably small, and use the more accurate estimator when the trees grow. Figure 3.17 demonstrates this for 14 trees that were generated with the same root. It is clear that the best estimator for these trees is around 15 and the deviation is less than 4% (compared with 12.5% for the general case). The individual point estimates here are also much better - within 16% of the calibrated estimate, 15.

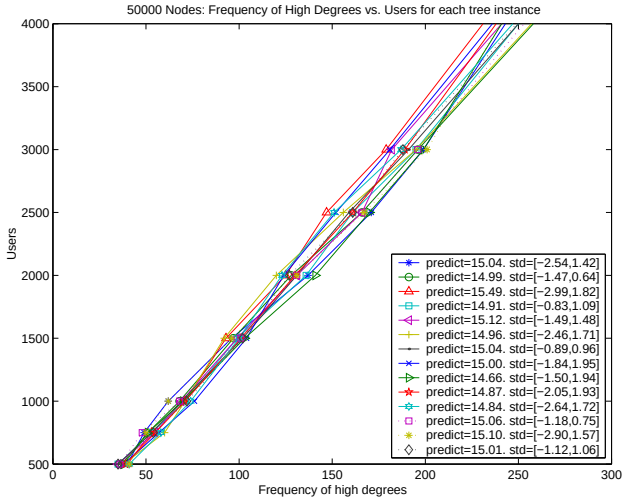

Figure 3.17: Clients vs. high degree nodes for each of the 14 instances of the tree

3.6 Results on the Tomography of Multicast Trees

3.6.1 Background

Graph Generation

Recent studies have shown that many real world networks, and, in particular, the Internet, are scale free networks. That is, their degree distribution follows a power law, $P(k) = ck^{-\lambda}$, where c is an appropriate normalization factor, and λ is the exponent of the power law.

Several techniques for generating such scale free graphs were introduced [BA99, MR98]. Molloy and Reed suggested an interesting construction method for scale free networks in [MR98]. The construction was part of a model describing an "exposure" process used to evaluate the size of the largest component in a random scale free network. We term this model the *MR model*. The construction method is as follows. A graph with a given degree distribution is generated out of the probability space (ensemble) of possible graph instances. For a given graph size N, the degree sequence is determined by randomly choosing a degree for each of the N nodes from the degree distribution. Let us define V as the set of N chosen nodes, C as the set of unconnected outgoing links from the nodes in V, and E as the set of edges in the graph. Initially, E is empty. Then, the links in C are randomly matched, such that at the end of the process, C is empty, and E contains all the matched links $< u, v >$, $u, v \in V$. Throughout this paper, we refer to the set of links in C as *open connections*.

Note, that while in the BA model the graph degree distribution function emerges only at the end of the process, in the MR model the distribution is known a-priori, thus enabling us to use it in our analysis during the construction of the graph.

Distribution Cut-Off

Recent work [CH03, CbAH02], has shown that the radius[8], r, of scale free graphs with $2 < \lambda < 3$ is extremely small and scales as $r \sim \log\log N$. The meaning of this is that even for very large networks, finite size effects must be taken into account, because algorithms for traversing the graph will get to the network edge after a small number of steps.

Since the scale free distribution has no typical degree, its behavior is influenced by externally imposed cutoffs, i.e. minimum and maximum values for the allowed degrees, k. The fraction of sites having degrees above and below the threshold is assumed to be 0. The lower cutoff, m, is usually chosen to be of order $O(1)$, since it is natural to assume that in real world networks many nodes of interest have only one or two links. The upper cutoff, K, can also be enforced externally (say, by the maximum number of links that can be physically connected to a router). However, in situations where no such cutoff is imposed, we assume that the system has a natural cutoff.

To estimate the natural cutoff of a network, we assume that the network consists of N nodes, each of which has a degree randomly selected from the distribution $P(k) = ck^{-\lambda}$. An estimate of

[8]We define the radius of a graph, r, as the average distance of all nodes in the graph from the node with the highest degree (if there is more than one we will arbitrarily choose one of them). The average hop distance or diameter of the graph, d, is restricted to:

$$r \leq d \leq 2r, \tag{3.14}$$

Thus the average hop sequence is bound from above and from below by the radius.

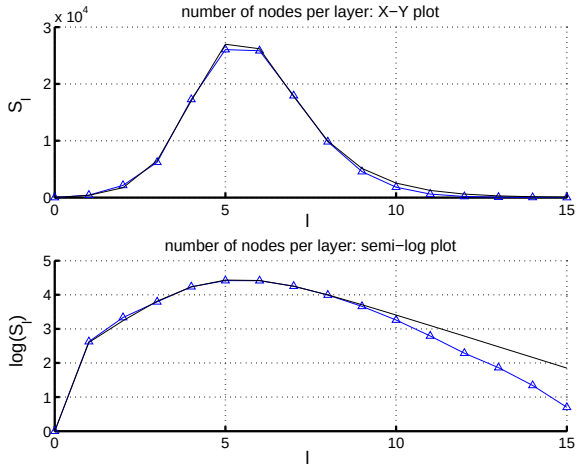

Figure 3.18: Number of nodes at each layer for a router level cut of the Internet with $N = 112,969$ nodes (LC topology). Analytical reconstruction for S_l is done with $\lambda = 3$, and $m = 1$.

the average value of the largest of the N nodes can be obtained by looking for the smallest possible tail that contains a single node on the average [CEbAH00b]:

$$\sum_{k=K}^{\infty} P(k) \approx \int_{K}^{\infty} P(k)dk = 1/N. \tag{3.15}$$

Solving the integral yields $K \approx mN^{1/(\lambda-1)}$, which is the approximate natural upper cutoff of a scale free network [CEbAH00b, DMS01, MJSAA02].

In the rest of this section, in order to simplify the analysis presented, we will assume that this natural cutoff is imposed on the distribution by the exponential factor $P(k) = ck^{-\lambda}e^{-k/K}$.

Results on Tomography of the Internet

We discuss here briefly the results obtained for the Internet in our joint work with the Bar-Ilan group. The full paper can be found at [CDH$^+$].

The results, presented in Figure 3.18, show that starting from a given layer $l = L$ the number of nodes decays exponentially. The actual probability distribution is not a pure power law, rather it can be approximated by $\lambda = 2.3$ for small degrees and $\lambda = 3$ at the tail. Our analytical reconstruction of the layer statistics assumes $\lambda = 3$, because the tail of a power law distribution is the important factor in determining properties of the system. This method results in a good reconstruction for the number of nodes in each layer, and a qualitative reconstruction of the probability distribution in each layer. In general, large degree nodes of the network mostly reside in the lower layers, while the layers further away from the source node are populated mostly by low degree nodes [DMS03]. This implies that the tail of the distribution affects the lower layers, while the distribution function for lower degrees affects the outer layers. Thus the deviations in the analytical reconstruction of the number of nodes per layer for the higher layers may be attributed to the deviation in the assumed distribution function for low degrees (that is: $\lambda = 3$ instead of $\lambda = 2.3$).

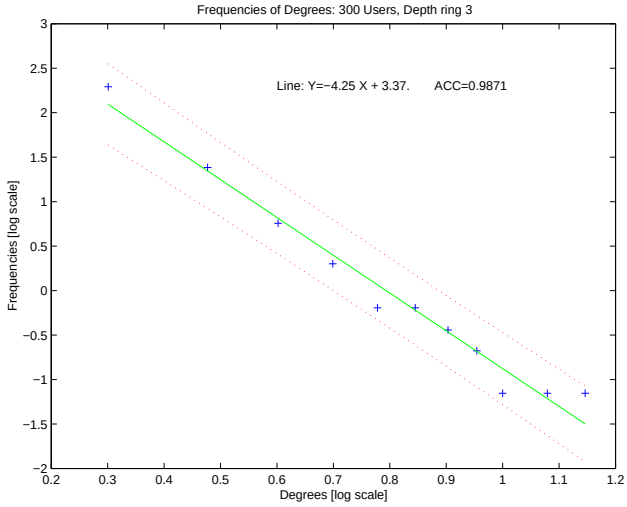

Figure 3.19: Third layer of a 300 client tree cut from topology $a_0 = 6, a = 1.5, p = 0.1, q = 0$

3.6.2 Empirical Findings on the Tomography of Multicast Trees

In this sub-section, we detail some of our findings on the structure and characteristics of the depth rings around the root node of shortest path trees. All of our findings were also validated on real Internet data.

It was rather interesting to observe that any layer with sufficient number of nodes to create a valid statistical sample obeyed a degree-frequency relationship which was similar to a power law, although with different slopes. We suspect that this is due to the exponential cut-off phenomenon discussed in the previous sections. Figure 3.19 shows this for the third layer around the root (i.e., nodes at distance three from the root) of a 300 client tree cut from a big IS topology (100000 nodes). The root was chosen with a high degree, and the clients with a low degree. Although the number of nodes is quite small, we see a very good fit with the power law. Figure 3.20 shows an excellent fit to the power law for the fifth layer around the root of a 10000 client tree, cut from the same topology. This phenomenon is stable regardless of the tree type, and the client population size. Note that the range of the power laws seen in figures 3.19 and 3.20 is less than one order of magnitude. This could indicate a crossover to exponential behavior.

To understand the exact relationship of the degree-frequency at different layers, we plotted the distribution of each degree at different layers. heswick at al. [CNS+99] found a gamma law for the number of nodes at a certain distance from a point in the Internet. Our results show that the distribution of nodes of a certain degree at a certain distance (layer) from the root seems close to a gamma distribution, although we did not determine its exact nature. Figure 3.21 shows the distribution of the distance of two degree nodes, and Figure 3.22 the distribution of the distance of high degree nodes, i.e., nodes with a degree six and higher. In both figures the root is a low degree node, and the tree has 1000 low degree clients. As can be seen, the high degree nodes tend to reside much closer to the root than the low deg ree nodes, and in adjacent layers. In this example, most of them are in the second to forth layers around the root, with only two more at layer five. This phenomenon was even more obvious when the root was a high degree node.

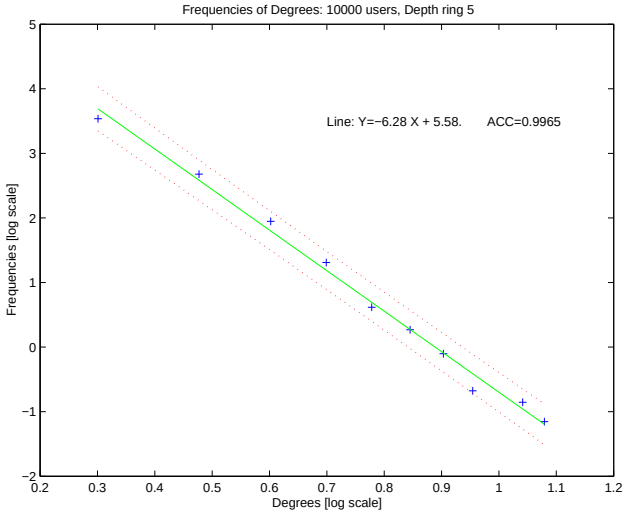

Figure 3.20: Fifth layer of a 10000 client tree cut from topology $a_0 = 6, a = 1.5, p = 0.1, q = 0$

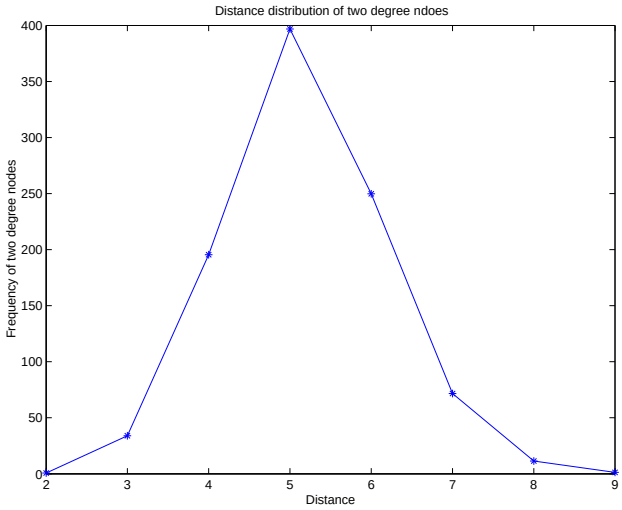

Figure 3.21: Distribution of degree two nodes in a tree cut from topology $a_0 = 6, a = 1, p = 0.3, q = 0$.

We also checked the distribution of the lengths of the paths to the clients. Our results show that the less connected the underlying topology, the higher is the average tree cut from the topology. For a 10000 node underlying topology with an average degree of three and higher, the height of the trees was not more than ten. On an underlying topology of 100000 nodes, the height of the trees was not more than 12. In accordance with our findings of a 'core' of high degree nodes, the trees were higher on the average when the root was a low degree node, compared to trees with a high degree root.

We verify the above findings with results obtained from a real Internet data set. Since we

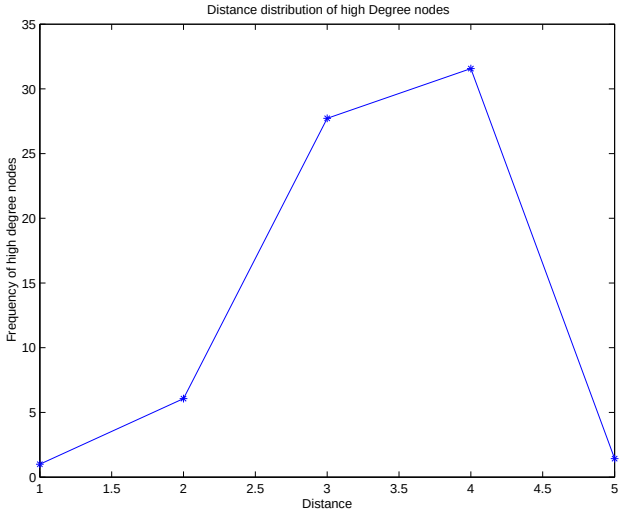

Figure 3.22: Distribution of the high degree nodes ($k \geq 6$) in a tree cut from topology $a_0 = 6, a = 1, p = 0.3, q = 0$.

have no access to multicast tree data we use the client population of a medium sized web site with scientific/engineering content. This may represent the potential audience of a multicast of a program with scientific content.

Two lists of clients were obtained, and traceroute was used to determine the paths from the root to the clients. It is important to note, that the first three levels of the tree consist of routers that belong to the site itself, and therefore might be treated as the root point of the tree, although in these graphs they appear separately. Figure 3.23 shows the frequency of degrees in the tree. The linear fit of the log-log ratio is excellent, with a correlation coefficient of 0.9829. The exponent is very close to the exponent we derived for trees cut from topologies that resemble the Internet.

Figures 3.24 and 3.25 show the frequency of degrees at layers 5 and 10 of the tree, respectively. It can be seen that the slope λ of the distribution increases with the layer number, e.g., layer 5 has a slope $\lambda \approx 2.34$, and layer 10 has a slope $\lambda \approx 2.99$. As we claimed, the shortest path tree cut from a scale free topology inherit many of the characteristics of the network topology. Moreover, we found for networks that the frequency-degree for each separate distance around the root can be approximated by a power law with an exponential cut-off, which is becoming stronger with the layer number. In Fig. 3.26 we plotted the slope of the distribution in the layer against the layer number and found a very good linear fit (note the outlier at $l = 7$ which was not included in the fit). The linear fit indicates that for the first layer the slope will be -1.83 ± 0.125.

For scale free networks, it has been shown [NSW01] that the first layer surrounding a chosen network node has a distribution $kP(k) \sim k^{-\lambda+1}$. Therefore, we can expect that in the first tree layer surrounding the tree root will have a frequency-degree slope of approximately $-3.18 + 1 = -2.18$ ($\lambda = -3.18$, the slope of the tree, is taken from Fig. 3.23) which is close to the linear prediction. While the results for the degree distribution in the first layer did not have statistical sig nificance the slope for the second layer was -2.09 which conforms to the above numbers.

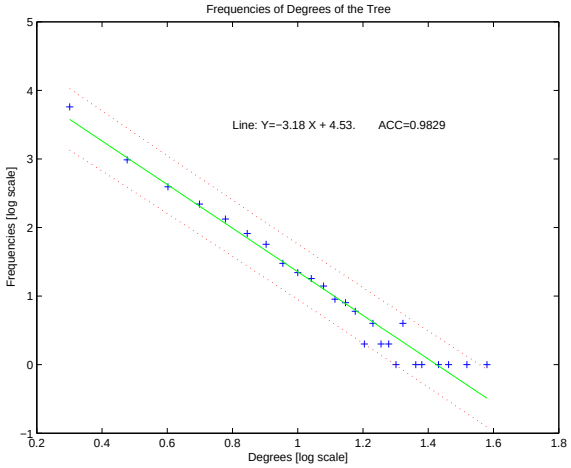

Figure 3.23: Frequency of degrees of the Internet tree.

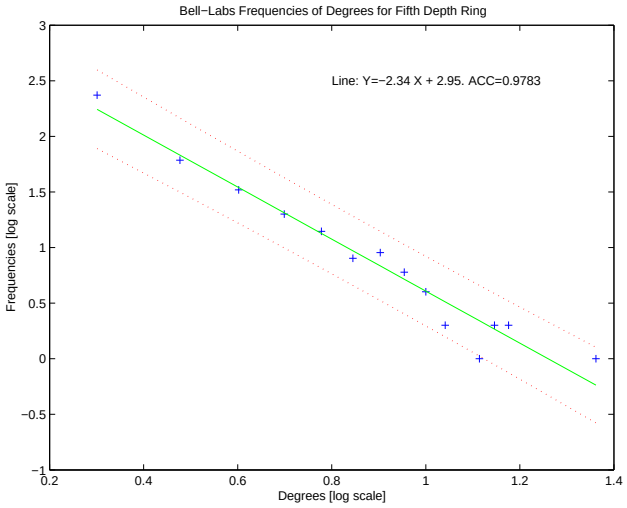

Figure 3.24: Frequency of degrees at layer 5 of the Internet tree.

3.7 Conclusions

We presented our findings on the characteristics of shortest path trees cut from power law topologies and the Internet. These findings may improve our understanding of multicast trees and therefore may help theoretical and practical research done in this area. We have shown that the structure of such trees follows power laws of rank-degree and rank-size, and that high degree nodes tend to reside in a low diameter neighborhood.

We found a linear ratio between the number of high degree nodes and the number of multicast tree leaves. We also proved this ratio analytically, and devised the Fast Algorithm that uses this ratio to estimate the tree client population in less than the Internet round trip delay. This algorithm,

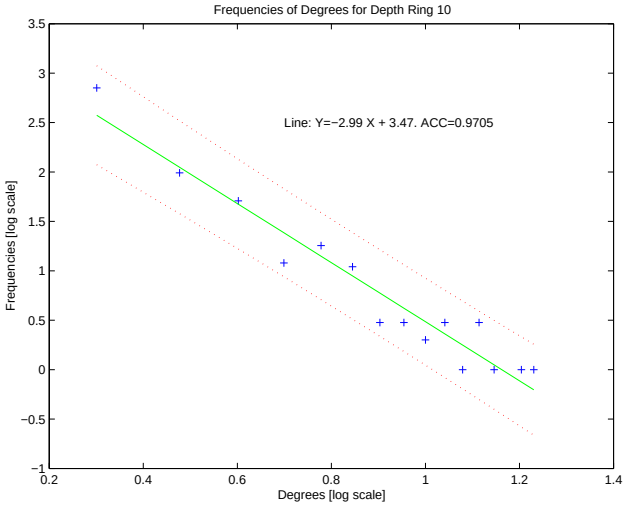

Figure 3.25: Frequency of degrees at layer 10 of the Internet tree.

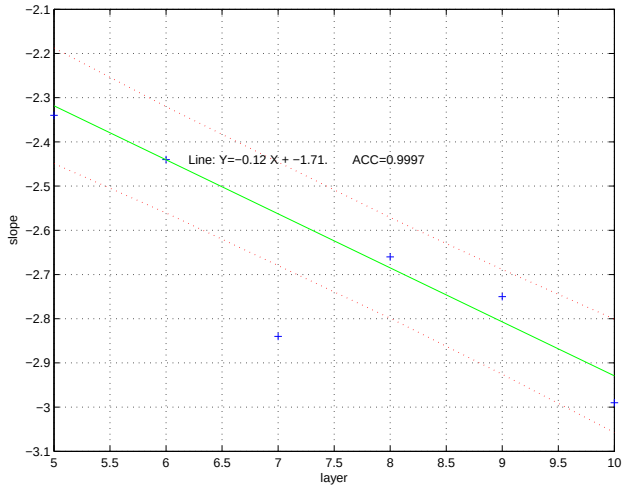

Figure 3.26: Slope of the degree distribution at specific layer as a function of the layer number .

when used as an initial estimator to polling based counting algorithms such as [BTW94, FT99], enables these algorithms to converge much faster, especially for medium and large groups. Note, that these algorithms performance is improved significantly with a tight initial group size estimation. It is also beneficial for transport layer feedback suppression algorithms and control algorithms which need to know the session size such as RTCP [RS98]. Finally, the Fast Algorithm can be used by network providers in calculating the gain from multicast with metrics such as the one suggested by Chuang and Sirbu [CS98]. As part of our future work, we intend to include an addition to the Fast Algorithm that enables the root to receive online updates on the changes of the branching characteristics of the trees. These online updates sent by nodes going in or out of the *high degree*

nodes group, enable efficient tracking over time of the multicast group size.

In general, we have found only a few examples where the estimator was off by more than 15%. When the estimator was calibrated to a specific root node the accuracy was a factor of four better.

This work presents a novel way for leveraging topological characteristics of a tree to obtain important knowledge such as its size. A further understanding of the exact ratio between the trees and the underlying topology characteristics is the subject of our future work.

In the second part of this chapter, we discussed the tomography of multicast trees and the Internet. We define a "layer" in a network as the set of nodes at a given distance from a chosen node. We find that the degree distribution of the nodes of a scale free network at each layer obeys a power law with an exponential cutoff. We derive equations for this exponential cutoff and compare them with empirical results. We also model the behavior of the number of nodes at each layer, and explain the observed exponential decay in the outer layers of the network. We obtain similar results for layers surrounding the root of multicast trees cut from such networks, as well as the Internet.

We believe our findings can have dual importance. First, they can help in devising better network algorithms that take advantage of the network structure. For example, we presented in the past [DMS03] an algorithm for fast estimation of the multicast group size that is based on our previous finding regarding the distribution of high degree nodes in Internet multicast trees. Second, our analytical findings suggest a simple local test for the validity of the power law model as an exact model of the Internet. Indeed our findings suggest that there is a good agreement of the empirical and analytical results.

Chapter 4

Scalable Content Delivery

4.1 Introduction

The efficient and timely delivery of web content is one of the most important challenges in today's Internet industry and research community. Sites need to cut delays while delivering content to a large number of users. This need, along with the high costs of bandwidth at the Internet core, drives the large sites to use caching and content delivery networks (CDNs) [Dor00, Ang00]. However, in their competition to attract users to return, sites alter content more frequently than before. Information and objects stored on web servers change quite frequently, often every few minutes (e.g., news flashes, bids, stock quotes) [DFKM97]. Recent studies suggested that caches or CDNs are responsible for not more than 50% of the content delivered to users [Mog00, WVS+99].

The question that arises is which type of objects change often, and what is the access pattern to highly changed objects. Breslau at al. [BCF+99] determined that page request distribution follows a Zipf-like distribution, but found a weak correlation between access frequency and the rate of change. Douglis at al. [DFKM97] found that 16.5% of resources that were accessed at least twice were modified every time they were accessed, and almost half of the text/html resources changed on each access after the first. [PQ00] found that a large part of users requests, and over half of the repeated requests, are to modified files. Access pattern to files follows, as reported in many papers, a Zipf-like distribution [BCF+99, DFKM97, KRS00, PQ00].

In this paper we target this phenomenon by presenting an architecture that enables sites to deliver frequently changing information to (an almost) unlimited number of users, in an efficient and scalable manner. Our architecture targets the top 2% hottest pages of busy sites, which account for the majority of accesses [PQ00]. At the heart of the architecture is a dynamic distribution selection mechanism that enables the server to identify an increase in demand and to activate cyclic multicast delivery for high demand pages before the site performance decreases. When demand abates our mechanism reverts to unicast delivery. We show how this scheme can be transparently integrated into the current web operation mode in a transparent way, requiring a simple plug-in at the client side. We supply mechanisms based on current DNS to dynamically direct browsers seeking URLs to multicast channels. To do so, we define a new protocol specifier, called *httpm*, which replaces the http protocol specifier for the potential hot pages.

The integrated architecture model is designed in a way that requires minimal changes to current architectures, and relies mainly on existing mechanisms. For the deployment of our architec-

ture we suggest two different schemes, which are based on existing building blocks. We use the digital fountain protocol [BLMR98] that offers an excellent tradeoff between transmission time and efficiency, for reliable data multicast. We also suggest different schemes for a feedback mechanism, that enables the site to estimate the size of a hot page multicast group. The first, based on our work in [DMS03], uses the Fast Algorithm for a rapid size estimation, with a maximal error of 15%. The second is based on the algorithm suggested in [BTW94], that enables the site to estimate the size of a hot page multicast group with an error rate that decreases with time. All the above described building blocks are altered to handle the multicast of dynamically changing content.

Our simulation results show that when switching from the unicast of a hot page to the multicast mechanism, the load on the server decreases significantly, while both the throughput and the goodput clients experience increase, and may arrive to 4 times the data received in the HTTP model. In the Integrated architecture clients experience a much lower delay than in the HTTP architecture. In addition, the load on core links decreases, thus enabling TCP users to receive better performance.

4.2 Motivation

4.2.1 Why caching is not the ultimate answer

As the Internet is growing, the cost of delivering content to the user at the edge of the Internet has become an important problem. The most intuitive way to decrease costs, is to move content to the edges of the Internet and closer to the user, often to a distributed cache system. The benefits of caching are many. It saves bandwidth, while allowing head-end links to be shared by more clients, to the benefit of the ISPs. It reduces costs to both sites managers and ISPs, by reducing their use of the Internet core links. Caching biggest advantage, which accounts for the rise of companies like Adero, Akamai, Exodus and Sandpiper, is of course the much faster response time a customer experiences, when receiving content from a nearby site [Dor00], [Ang00].

Although caches form a valuable and important solution for improving web performance and decreasing the load from sites servers, a substantial fraction of the content, as seen above, is delivered from the sites themselves, to either the users or the different caches. This content falls into the following categories:

- Uncachable information, which is either personalized information, query results, real time streams, and any time dependent information.

- Dynamically generated data: Several researchers have tried to estimate the update rate of sites. Douglis at al. [DFKM97] observed a rather high update rate for about 13% of re-referenced pages, and among the highly referenced html pages, about fifth changed every 15 minutes or less. Overall, they found that a large number of pages are updated periodically. Their findings were also confirmed by Rodrigez and Sibal [RS00], who found that a large number of web sites update their content, following an exponential distribution. As a result, many potentially cachable resources change fairly rapidly. A recent paper by Padmanabha and Qiu [PQ00], that studied the behavior of a busy web site, MSNBC, found that server content tends to be highly dynamic, and the duration between successive modifications usually lies between an hour and 24 hours.

- Cache misses and delta encoding deliveries, wherein the site transmits to the different caches either the entire page or only the difference between older and current versions of a page. In order to maintain cache consistency, caches use different Time-To-Live and leasing heuristics [GC89], [Wes96], [LC97], [DST00]. Delta encoding, used in many architectures, takes advantage of the observation that the changes are often minor. It is important to observe, though, that these transmissions also impose some sort of load on the server and core lines, especially for very popular pages, which are cached in many mirror sites.

4.2.2 Characteristics of semi-dynamic content

The question that arises is which type of objects change often, and what is the access pattern to highly changed objects. Breslau at al. [BCF+99] determined that page request distribution follows a Zipf-like distribution, but found a weak correlation between access frequency and the rate of change. Douglis at al. [DFKM97] found that 16.5% of resources that were accessed at least twice were modified every time they were accessed, and almost half of the text/html resources changed on each access after the first. [PQ00] found that a large part of users requests, and over half of the repeated requests, are to modified files. Access pattern to files follows, as reported in many papers, a Zipf-like distribution [BCF+99], [DFKM97], [KRS00], [PQ00]. The latter reported that the alpha parameter, as seen by the server, is larger than previously reported, and is in the range of 1.4 - 1.6. This implies, for instance, that just the top 2% of documents account for 90% of the accesses. They also found that the popularity of hot pages tends to be stable over time scale of days.

The above discussion led us to the conclusion, that by dealing with the top most popular objects, we will be able to take a lot of the load off sites and core links. We were then faced with the question of identifying these popular objects. When viewed on a per-site basis, it became rather easy to identify them: The changing content of the homepage of popular sites was always at the top of the list. There were also pages that were a result of special occasions. For example, breaking news; the homepage of Napster the day the court ruled against it; The Starr report; Or the votes count page on the Florida site at election day. There were very few such pages at each site, actually one or two, on a per day basis. The most important thing we noted was, that the most popular pages were a result of some event, and as such, could be classified as new objects (see the examples above). We then concluded that the knowledge of which pages or objects have the potential of becoming most popular exists only from the time of their creation. For example, all voting pages of all states had the potential of becoming extremely popular on US election day (that is, one page per site), and could be noted as such at their creation. One of these pages, the Florida voting page, actually became extremely popular, but the site could not respond to the demand, even after additional hardware was installed. Moreover, in the referred elections, it was impossible to get to any major news site online, although most surfers were looking for the same specific fragments of information. The multicast of these pages could have solved these problems.

4.2.3 Multicast

The use of multicast for highly popular items has many advantages. It significantly decreases the site load, especially at maximum peak load times, while lowering the amount of traffic in core links. As a result the site is able to serve a larger number of users, even during denial of service

attacks. However, multicast itself has a management cost, which consists of join operation and the creation and maintenance of multicast delivery trees. Moreover, implementing multicast is still an open question for a lot of network providers. Several studies have been performed lately in order to estimate the efficiency of multicast versus unicast delivery [CS98, PST99, CA01]. [CA01] developed a metric for measuring multicast efficiency, based on the number of traversed links. They found that multicast is extremely efficient for large group sizes, and even offers a 60-70% reductions in the number of links for group sizes as small as 20-40 receivers.

The dynamic mechanism we propose enables the server to switch between unicast and multicast mechanisms according to the identified load. Our approach is scalable and enables sites to cut down costs, and still be able to handle increased demand. Users benefit from better response time due to lower load on the server, and less congested core links. The incorporation of multicast as a means for web content delivery is not new. Several efforts were made to deal with it, which we discuss later in Section 4.5. These works either reviewed the problems which are to be confronted, or suggested a rather limited use of multicast. We assume the existence of multicast infrastructure, and do not tackle the issues involved in deploying this infrastructure.

Our integrated architecture, discussed next, enables the efficient and scalable delivery of very popular dynamically changing content to a large number of users. It's a complementary solution to existing caching mechanisms, which enables also scalable delivery of content to caches.

4.3 Integrated Architecture Description

The integrated WWW architecture was designed to allow a smooth transition from TCP based connections to a multicast based mechanism for uncachable hot pages. The architecture is based on a dynamic selection mechanism, which determines which of the hot pages should be multicast, based on inputs received from both the relevant TCP connection requests and UDP feedback information mechanism.

4.3.1 Server Side

The site is responsible for maintaining the hot page selection and multicast mechanisms. We classify and motivate here each of its parts. The rest of this section gives a detailed description of each of these parts.

The site predetermines a list of pages which are candidates for becoming hot pages. These pages are given the *httpm* protocol prefix. This list is quasi-static (e.g., Krishnan et al. [KRS00, Fig. 12] showed that the list of popular entities is quite stable, in a medium size site they examined) and thus can be predetermined by the site's administrators off-line. The list can be edited to reflect the relatively slow changes in the popularity of pages, which may demand configuring new entries or deleting existing ones at the DNS.

The dynamic selection mechanism is activated for each of the potential hot pages. It processes the number of incoming connection requests for them. Once a threshold has been crossed for one of the pages, the server begins the process of moving it to the multicast mechanism. It then notifies the DNS of the new address for the appropriate entry.

Once the page is multicast, new connection requests for it are answered with a 3xx HTTP response. This response indicates which multicast address to join, and specifies the needed plug-

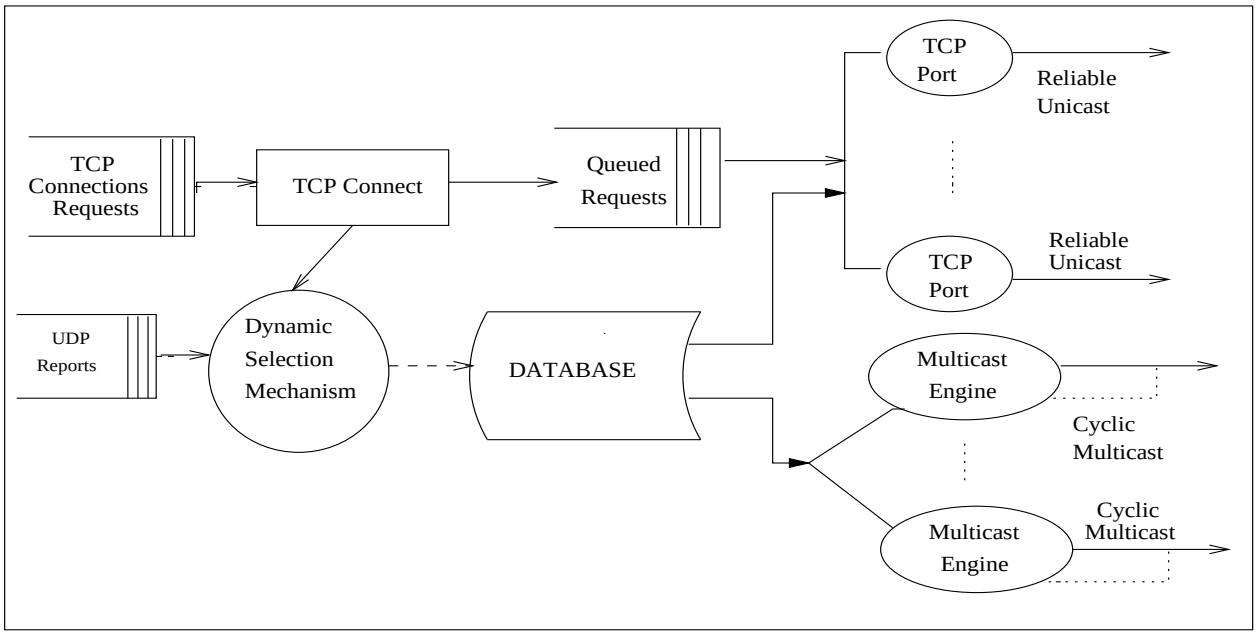

Figure 4.1: The Site Distribution Selection Mechanism

in. The response has to be determined as part of the HTTP protocol specification.

When the page is multicast, the server periodically estimates the size of its multicast group. When the estimation crosses down a predefined group size threshold, the site distribution selection mechanism reverts the page from cyclic multicast back to unicast, and notifies the DNS of the new address, the server IP address. While multicast is beneficial for periods of high load, an estimation of the amount of needed resources at other times show it is less beneficial ([AA97]). Unnecessary edge links are loaded due to the propagation time of prune messages. This unnecessary load at edge networks impacts the performance other clients, and therefore should be avoided when possible. It is clear that at high load times, multicast is better than concurrent unicast.

Switching a page from multicast to unicast requires a grace period, in which existing members of the group finish receiving the page and new joiners are diverted back to unicast. Therefore, the page is still multicast for a few more rounds, and then a diversion code, which can be recognized by the client browser, is multicast for another period. In this last step the multicast group is exhausted.

Existing Building Blocks

For the multicast mechanism we incorporate two existing building blocks. One is used for efficient delivery, and the other for the group size estimation. Our design describes two different schemes in which to incorporate these building blocks, and enable their use for dynamic content. We give here a short description of these building blocks.

Coding Scheme for Multicast: The digital fountain [BLMR98] provides an efficient multicast mechanism. Reliability is achieved without the use of feedback messages from clients, and at minimal costs. The digital fountain requires an encoding system at the sending side, i.e., the site, and a decoding mechanism at the receiving side, i.e., the client. In the scheme, a page P_i, which consists of a set of k packets, is encoded into a set of $k + \ell$ packets, such that ℓ of which are redundant, to

a total of n. All n packets are sent cyclically over time. The encoding/decoding process is highly efficient, due to the use of Tornado codes, and therefore imposes very low overhead. The efficiency of the encoding system requires a small stretch factor, which determines the ratio between k and ℓ, even at rather high loss rate. The digital fountain server design uses layering across multiple multicast groups, and is based mainly on [VRC98]. In this approach, the server organizes the data into g layers, each corresponding to a multicast group. The layers are organized in increasing transmission rate. The layers are cumulative, in that a client who joins layer i, actually joins layers $0..i$. Each layer transmits in an increasing rate. If the ratio between the layers transmission rate is B_i, then a client who joins layer i, receives bandwidth proportional to $2 \cdot B_i$, for $i \geq 1$.

Multicast Group Size Estimation We describe here two different methods, based on exiting results.

End to end : The first mechanism we suggest is an end to end scheme, based on the feedback control mechanism presented by [BTW94]. The scheme was originally intended for multicast video distribution feedback. It requires all participants of a multicast group to generate a random key of 16 bits. The sender sends its key, and awaits an answer from receivers with a matching key. Each period the key is sent with mask of increasing length, until an answer is obtained. Answers also contain the state of the net as perceived by receivers, for rate control mechanisms and timing.

Router layer level: This mechanism is based on our work from [DMS03], which uses the found topological characteristics of a multicast tree for a fast evaluation of the size of the mutlicast group, by counting the number of high degree routers in the resulted IP-multicast tree. The source of the session initiates the Fast Algorithm, by activating a timer, and sending a multicast message to all the routers in the underlying IP-multicast tree. Each highly connected router, i.e., a router with degree six and above, unicasts a reply to the source. The first time out is determined by the distance of the source to the core of the network. A second timer is then activated repeatedly, its length determined by the estimated delay of one hop round trip. This repeated timer is activated until the data gathered from the next hop does not improve the estimation by more than a predetermined threshold. For a detailed description of the Fast Algorithm see [DMS03].

Site Distribution Selection Mechanism

As can be seen in Figure 4.1, the dynamic selection mechanism (DSM) is part of the site's software, which determines whether hot pages are unicast or multicast. The DSM is given the following parameters:

- List of potential hot pages: This list is predetermined by the site's administrators. (A discussion of the quasi-static nature of the list is appeared in Section 4.2).
- Unicast to multicast threshold: This threshold is actually the number of simultaneous connections that the site cannot tolerate for any one page, and would rather carry the expenses of the multicast delivery. For example, let us assume that it takes an average of 2 seconds to deliver a hot page, when the server is not overloaded and the net is not congested. (For

simplicity, we assume the page requires only one connection, as is specified in HTTP1.1). If, during a minute time, the server receives 300 requests for that page, then it has to reserve 10 *simultaneous* connections throughout the minute for the delivery of this page only. We formalize this discussion in the following way:

Let Δ_i be the average time estimated for any one connection to a page P_i. Let us define the number of requests to P_i in the j the period of time (typically a minute) by R_i^j. Let T_i be the connection threshold, set by the site administrators, for page P_i. Let $S_i^j = R_i^j \cdot \Delta_i$.
Then, the following conditions should be met, for the DSM to decide to multicast page P_i:

Threshold

$$k \in [0...m] : S_i^{j-k} \geq T_i$$

Monotonically non-decreasing

$$k \in [0...m-1] : S_i^{j-k} \geq S_i^{j-k-1}$$

The range $k \in 0..m$ determines the number of periods, in which the conditions above should hold. This interval is needed to determine a pattern, rather than a momentary deviation in requests rate. For example, let $m = 3$. In this case we get that during 3 consecutive periods (minutes) there was a raise in demand for page P_i, and the demand was above the threshold, T_i.

- Multicast to unicast threshold: This threshold is the number of connections that the site can tolerate for a hot page P_i. When this threshold is met during several consecutive estimations done at the DSM, and each time the size of the multicast group decreases, then the DSM switches the page from the cyclic multicast scheme to the HTTP unicast one. As discussed above, this switch is done to decrease potential unnecessary overhead encountered at the edges at low load times.

The DSM obtains the data it needs from two sources: One is the HTTP connections requests, and the other is a counting feedback mechanism. Obtaining the number of requests per hot page is rather straight forward, and involves simple lookup mechanisms. To obtain an estimation of the size of the group, a feedback mechanism based on [BTW94] is activated. This mechanism is part of the multicast mechanism, and will be detailed there.

We assume here that as long as IPv6 is not embedded, multicast addresses constitute a precious resource, that might be rather expensive. Therefore, a site will be able to acquire, either due to the price or due to regulations, only a limited amount of such addresses, and share them between its current hot pages. Thus, even in the absence of high maintenance costs for multicast, a site will be able to use multicast only for a small subset of its pages.

Multicast Mechanisms

We suggest here two different schemes for the multicast of hot pages. Our schemes, which are based on the digital fountain mechanism, also enable an efficient delivery of dynamically changing content. The schemes differ in the levels and rate of transmission, number of concurrent transmission and the way the feedback mechanism is incorporated and used. Each scheme requires different resources from the site, and is intended for different site characteristics. Both schemes

cyclically multicast the data over UDP. Users may join at arbitrary times, and leave once the entire data is obtained.

Since pages change dynamically, a page which is currently transmitted may change on disk. A major requirement in our scheme is to incorporate the change as fast as possible, as users rather have an up-to-date version, if one exists. For this purpose, the header of each multicast packet contains a *continuous* bit. When a page changes on disk, the new version of the page is multicast in the following round, and the bit is toggled. The decoder at the clients side assemblies packets with the same value in the *continuous* bit. If a change is detected, then the decoder discards all packets obtained so far, and starts assembling the packets again, until it can reconstruct the page. Since a change in content occurs only when a new round of the cyclic multicast begins, one bit is sufficient.

Scheme 1: In this scheme the site uses only one channel per page. The site encodes the page for some initial loss rate (for example 10%) and sets the digital fountain's stretch factor accordingly. The encoded information is sent in a rate calculated to match the slowest client possible. Here, the feedback mechanism is used to estimate both the size of the group (for the DSM) and the current loss rate in the net. Since data is transmitted at a low rate, no rate control mechanism is needed.

Once in a few rounds, the feedback mechanism is activated. Every packet in this round contains the following fields:

- The *key* field, which consists of 16 bits, and contains a randomly selected set of 16 bits.
- The *mask* field, which is a byte long, and specifies the number of masks bits in the key. The value of the mask field is determined by the increase in requests rate exhibited at the DSM for this page. Let R_i^M = Max R_i^j, $j \in \tau$. Then, the value of the mask field is $logR_i^M$.

If no answer was received during this round, the mask field is incremented by one, and the new mask field is sent in the next round. The process repeats until answers are obtained.
The client's decoder, upon receiving a packet containing these two fields, randomly picks a key of 16 bits. If the first *mask* digits are the same as in the site's key, it counts the exact number of packets it needs to reconstruct the page, and delivers this information back to the site, along with the read value of the *mask* field.
The site then estimates the size of the group from the number of responses obtained for a round, according to the *mask* field reported. It can also estimate the congestion along the way to the users by the number of packets received until the page could be reconstructed . If this number is bigger than the number of packets sent in a cycle, then the stretch factor is incremented, thus increment the amount of encoding information sent.

Scheme 2: This scheme incorporates fully the digital fountain multi-level approach (see section 4.3.1). Therefore, the feedback mechanism does not need to incorporate in it any congestion control information. The mechanism is incorporated as described in scheme 1, and is used only to estimate the size of the group. The two fields query is sent only at the lowest layer, so that it is received by all clients.

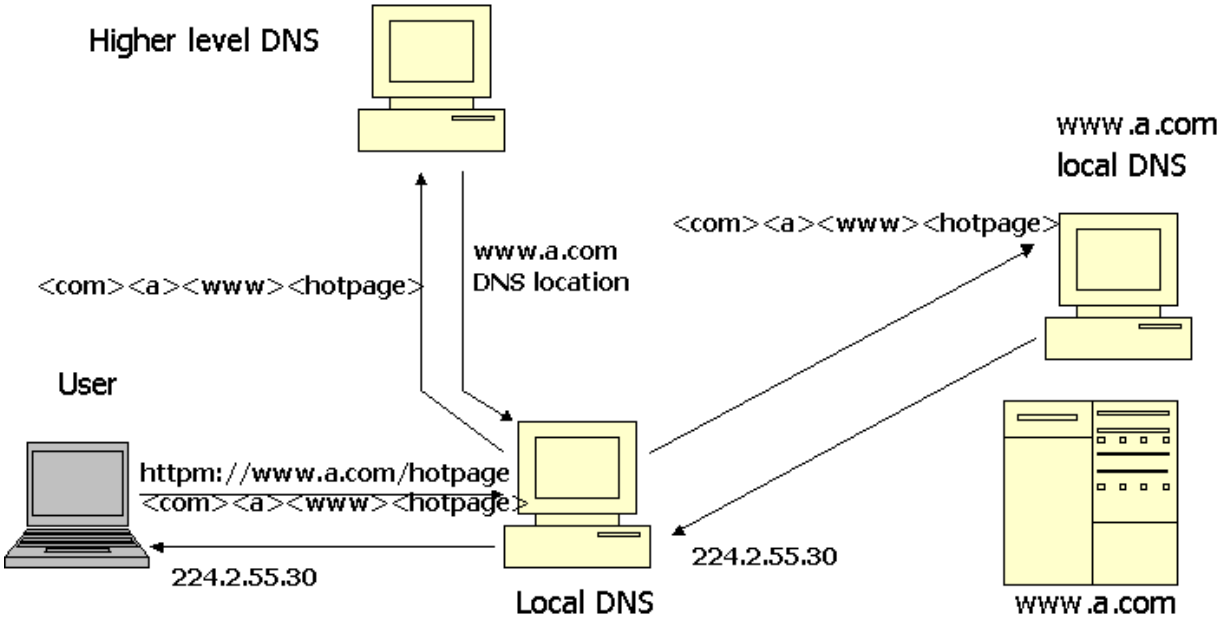

Figure 4.2: dns

4.3.2 Client Side

Browsers at the client side require an additional plug-in. This plug-in identifies an *httpm* address, and acquires its IP address from the DNS. The plug-in then determines whether the address is a server's IP or a multicast address. In the case it is a multicast address, the client joins the multicast group, and activates a decoder, that decodes packets received using the digital fountain scheme. During the decoding process, control information sent along with the packets is processed, and the plug-in acts upon it.

4.3.3 Hot page address resolution

In section 4.2, we discussed the quasi-static nature of the list of potential hot pages at a site, and concluded that only one or two of these pages can be a candidate for multicast at each site. Furthermore, our discussion showed that many times a site can determine the potential of such pages only at creation time. Therefore, it is best that each site predetermines this list manually. It can be automated quite easily using the DSM mechanism, but according to our understanding such lists are usually manually configured.

Each page on this list is given the *httpm* protocol specifier. As discussed before, *httpm* pages are treated differently both at the site and at the client side. In this section, we discuss how our system enables the plug-in at the client browser to obtain the right address for a *httpm* page.

At the site, the authoritative DNS is updated with the list of *httpm* pages, and the appropriate IP address for each of them. At initialization, probably most of them are not multicast yet, and therefore their IP address is still the domain's IP address. Once the DSM decides to multicast a page, it changes the page's IP address at the authoritative DNS to the multicast channel IP address, and sets a rather low TTL for it. When the DSM decides to revert this page back to unicast, it

updates its DNS entry again, to the site's IP address.
At the client side, when a *httpm* URL is entered, the browser activates the plug-in. The plug-in sends the entire URL to the local DNS for address resolution, as shown in figure 4.2. When a resolver sends a domain name for resolution, it sends it with no delimiters and in a reverse order, e.g., http://www.a.com is sent as <com><a><www>. Each DNS then looks for the longest prefix it can match. In the *httpm* case the resolver sends the entire URL for resolution, with the domain name as the prefix. For example, the URL httpm://www.a.com/hotpage is sent to resolution as <com><a><www><hotpage>. Accessing the DNS with the entire URL requires no change in current DNS mechanisms, since the DNS is a hash table which receives strings and performs a look-up operation. Our scheme only requires a change in the resolution library function, so that the entire URL string is sent. The use of an iterative (non-recursive) resolution process assures that accessed DNS's do not become clients.
Upon receiving the DNS response, the client plug-in determines whether the address is a multicast address, or the web server's IP address. If the resolved address is a standard IP address, the client uses the normal HTTP GET mechanism. If the resolved address is a multicast address, the client seeks to join the multicast group.

4.3.4 Scalability

Translation of names to IP-addresses is done via the Domain Name Server (DNS) mechanism. Our design uses the DNS for consistency reasons. Next we show that our scheme does not cause DNS scalability problems.

Remember that the number of *httpm* pages is limited and the URLs are cached in local DNS servers only on demand and for a short period. For example, assume that 500 sites publish two *httpm* pages at the exact same time. In this case, there will be only 1000 new DNS entries added to the entire DNS system. There are more than 30 million registered domain names today in the Internet, hence the extra burden is of less than 0.01%. Moreover, since the Zipf distribution is also valid for clients accesses, the number of such cached entries is limited at any specific time at local DNS servers.

4.4 Simulation Results

In this section we show how our architecture affects performance for both clients and sites. Other works [CS98, CA01] examined the efficiency of multicast at the network level. Our objective is to provide a first order understanding of both the throughput seen by clients and the load on the server at peak times. We compare the two suggested multicast schemes to unicast. All results are given for the general case of scheme2. In cases where the two schemes differ significantly, the special case of scheme1 is also given.

For our simulations we used the *ns-2* [ns-] network simulator. We used a generator that creates Internet-like random topologies with a power law relationship of the node degree (based on [FFF99b]). We simulated a WAN network as a network of Autonomous Systems (AS's). Each AS represents either a LAN or a dial-up line switch with active web clients. LAN speed is 10 Mbit per sec, while dial-up line is 56 Kbit per sec. Dial-up line switches have very low connectivity. A web server is located in one of the backbone ASs, connected through a dedicated router. It

supports a large number of concurrent connections, each is simulated through an *ns* port entity. These ports play the role of pre-forked slave processes in the server, and are either TCP or UDP ports. There are 100 TCP ports and 4 UDP ports, the latter transmit in different rates. Arriving TCP connection requests are queued in a *connection request queue*, which simulates the two TCP queues of completed/incomplete connections. It is limited in size, and once filled, in-coming requests are dropped. The unicast mechanism is implemented on top of a two-way full TCP stack, which uses the Reno congestion control protocol. The multicast mechanism we simulated for both schemes. Clients in the AS's make requests to the server, according to a predefined behavior mechanism. The server has 7 available Web pages of different sizes and different popularity. Page size varies according to a normal distribution, according to minimal, maximal and average parameters. The popularity was chosen so that 2 of the pages are responsible for 80% of the accesses. The pages parameters are detailed in the following table:

Type	Size (KBytes)			Popularity
	Minimal	Average	Maximal	
1	25	100	300	4%
2	20	80	240	4%
3	15	50	150	4%
4	10	40	80	4%
5	5	20	60	4%
6	40	40	40	40%
7	60	60	60	40%

We alter the client's waiting time to achieve different loads on the server. A client asks for a page according to the distribution, receives the page (or times out), waits and the ask for another page. Clients may terminate the connection due to slow service either at connection establishment or while receiving the page.

4.4.1 Goodput

We measured here the amount of data bytes sent by the server vs. the amount of data bytes received by the users for both models. In the Integrated Architecture, the two hot pages (pages 6 and 7) are multicast at different rates. Figure 4.3 and Figure 4.4 show the amount of data sent and received in both architectures. While Figure 4.3 shows the behavior at peak times, in Figure 4.4 we gradually increase the load on the server for both architectures. In order to increase the load in the Integrated model, the users request pages at 4 times the rate of users in the HTTP model. The results show, that in the HTTP model, the higher the load, relatively less data is received, up to 30% less received than sent at peak times. In the Integrated model, on the other hand, even at peak periods, the users receive up to 4 times the data sent by the server. As will be seen in this section, not only server load decreases, which enable it to serve on the less hot TCP clients more efficiently, but also network load decreases. This leads to a much better service to both multicast and unicast users at the Integrated model.

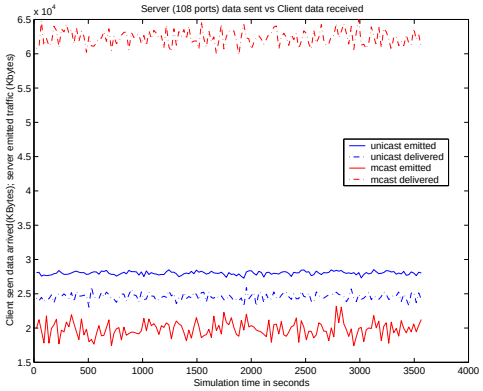

Figure 4.3: High Load: Transmitted data vs. Received data

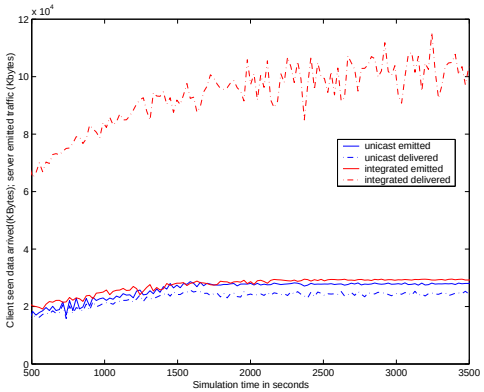

Figure 4.4: Increased Load: Transmitted data vs. Received data

4.4.2 Server Load

The server load is obtained from several different statistics. First, we examine the ratio between established connections and available out channels (i.e., ports). Obviously, the server is overloaded when there are awaiting established connections in the queue, and all out channels are busy. Other criteria are the amount of dropped connections from the connection request queue, as well as the rate of establishing connections. Our results show that when the hot pages are multicast, the server load decreases dramatically. Figure 4.5 shows the amount of load the server maintains. A 100% load is when all the ports are busy, yet the connection request queue is empty. As described earlier, we create more load by shortening the interval between successive client requests. To maintain perspective, we had to quadruple the load in the Integrated model in compare to the HTTP model. Yet, as can be seen in Figure 4.5, the server reached saturation much earlier (around 1700 seconds) than the Integrated model (around 3000 seconds). Figure 4.6 demonstrates the server load when we do not quadruple the load in the Integrated model.

The server reaches saturation when all of its ports are busy, and the connection request queue is full. Figure 4.7 presents the amount of dropped connections from this queue. The HTTP model reaches saturation much quicker than the Integrated model, and when it does, the amount of dropped connections is much higher.

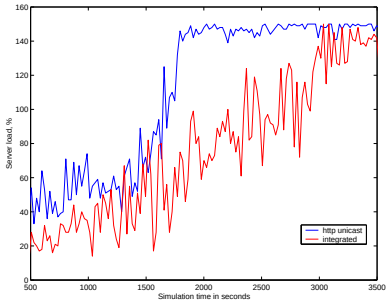

Figure 4.5: Server Load

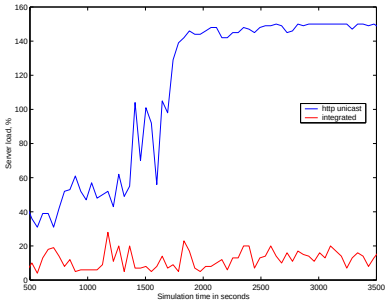

Figure 4.6: Server Load - Equal Delays between Clients Requests

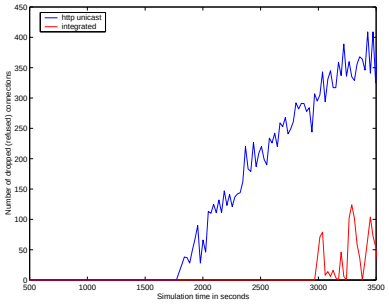

Figure 4.7: Dropped Connections

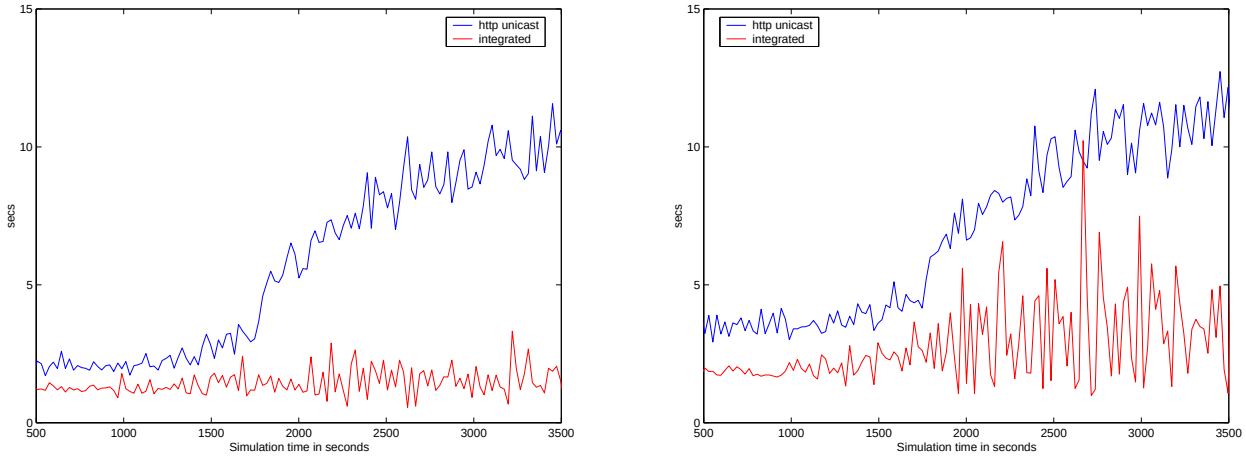

Figure 4.8: Client Seen Delay: pages 6 and 7

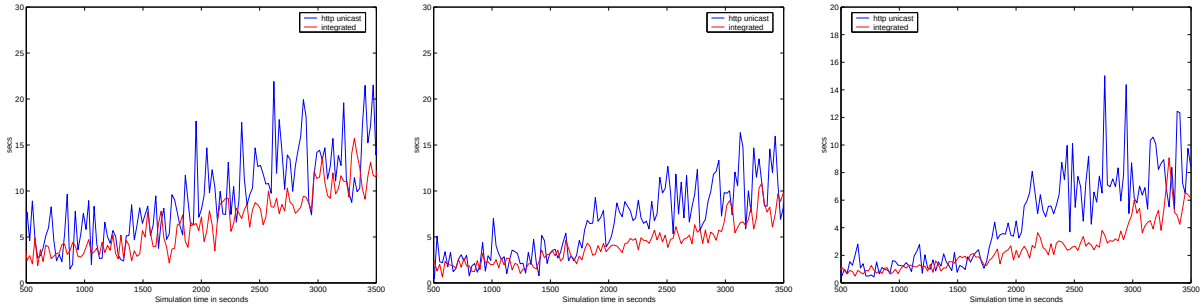

Figure 4.9: Client Seen Delay: pages 1, 3 and 5

4.4.3 Clients Seen Delay: Server vs. Network load

We measured the delay perceived by clients in both models as a function of time, while increasing the load with time until we saturate the server (see Figure 4.5).
Figure 4.4.3 shows the amount of time it takes users in both architectures to receive pages 6 and 7, the hot pages that are multicast in the integrated architecture. The results show quite clearly that clients in the Integrated model receive these pages much faster than in the HTTP model. Furthermore, it is beneficial to multicast these pages even if the server is not loaded (we refer here to scheme 2). As the load increases, our results show that the delay a user suffers in the HTTP model can be 4 times the delay in the Integrated model.
Figure 4.9 shows the amount of time it takes clients to receive pages which are unicast in both models. Page 1 represents a large file, Page 3 a medium size file and page 5 a rather small file. All three files suffer smaller delays on their way to the user in the Integrated model, and the effect is more obvious with the increase in server load. A similar effect can be seen when viewing the first packet delay, in Figure 4.10. As the load on the server increases, it takes longer to create new connections in both architectures, although significantly longer for the HTTP model.

An interesting point is that the smaller the page, the bigger is the difference in delay between the two architectures. Although this can also be explained by first packet delay, which is more significant for shorter connections, we suspected there is another factor that influences this delay - the load on the network itself. Figure 4.11 shows the percentage of retransmitted packets from the server, due to time outs in the TCP protocol. These time outs indicate that the network is

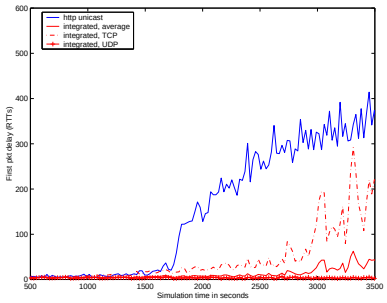

Figure 4.10: First Packet Delay

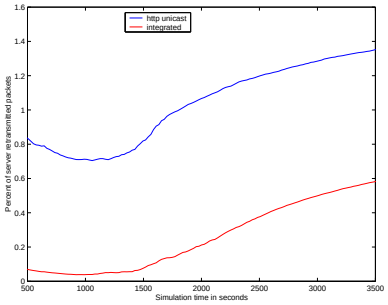

Figure 4.11: Retransmissions from the Server

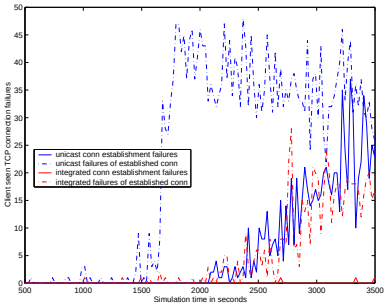

Figure 4.12: Established Connections Failures

congested, which causes the TCP flow control mechanism to adjust its window size and slow down the connection. This retransmission percentage, although quite low in both models, is more than double the size in the HTTP model than in the Integrated model, indicating that the network is more congested in the HTTP model. Figure 4.12 shows the amount of connections terminated by users, due to too long delays. As soon as the server reaches peak load, clients start to give up on connections. Since the server in our simulation does not impose additional delays on established connections, then this extra delay is due to the TCP flow control mechanism, that adapts to a congested network.

From the above we conclude, that one of the reasons for the degradation in performance seen by clients at peak times is the congestion in core links. The closer the link is to the server, the more congested it can become. Although the TCP congestion control mechanism tries to limit the effect of such peak times, by exhibiting a social behavior, the effect on both the site and the client is big. The site retransmits an extensive amount of packets until the TCP slow start mechanism takes effect, while the client experiences degradation in performance. The use of multicast mechanism for the delivery of hot pages, reduces both the traffic on core links, and the load on the server. Clients which connect by http at such times receive better service, and the amount of retransmission decreases significantly.

4.4.4 Special Case: Scheme1

We look here at the special case, where the server uses only one outgoing channel per multicast page, therefore transmitting it in the rate of the slowest receiver. In our simulation, it means at a rate of 56Kbits per second.

Figures 4.13, 4.14 and 4.15 show that the load on the server decreases significantly in the Integrated model, while the goodput is much higher than in the HTTP model.

The special case of scheme 1 enables us to investigate the effect of using multicast instead of unicast on the network itself. Since the server transmits UDP data in a low rate, clients receive hot pages rather slow, and the server cannot be over loaded in the Integrated architecture. Therefore, when we quadruple the amount of requests made by clients in the Integrated architecture, we create extra load on the network, rather then on the server. Figures 4.16 and 4.17 show the effect of slow UDP connection on congested network - fast receivers are bound to lower rate when receiving multicast packets, while TCP packets are delivered in close to dull speed. As traffic increases, UDP packets that are lost cause receivers to wait till the next multicast cycle. Multicasting hot

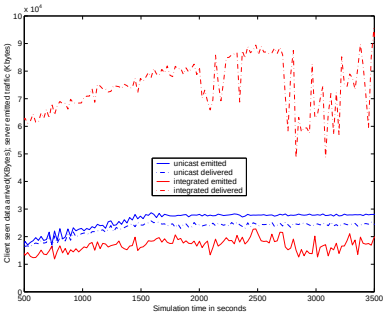

Figure 4.13: Scheme1 Increased Load: Transmitted data vs. Received data

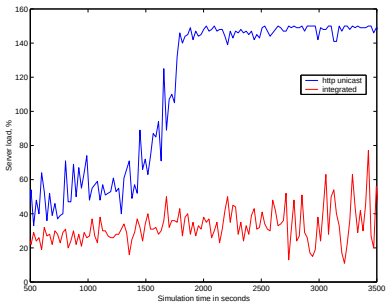

Figure 4.14: Scheme1: Server Load

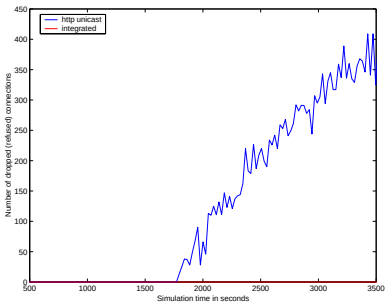

Figure 4.15: Scheme1: Dropped Connections

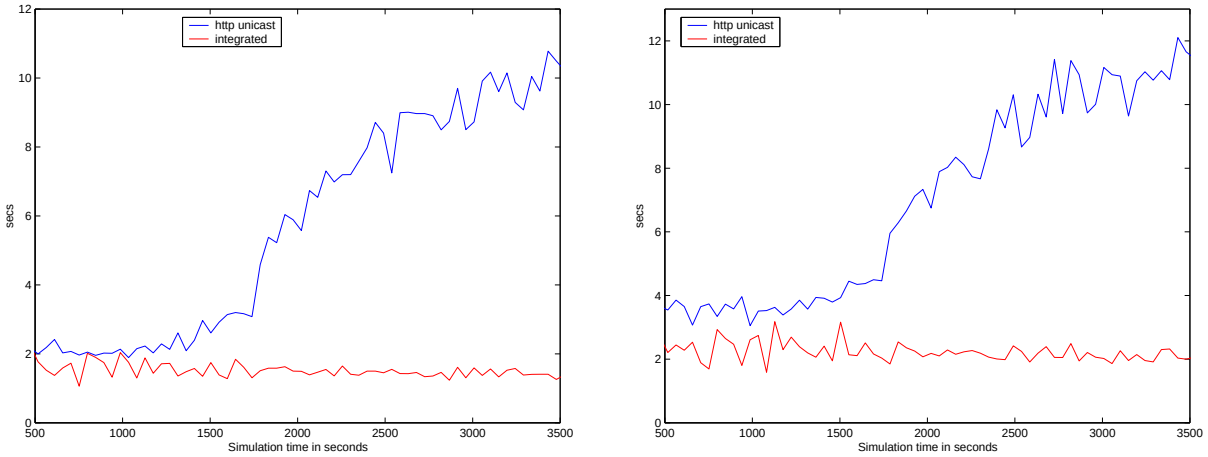

Figure 4.16: Scheme1 Client Seen Delay: pages 6 and 7

pages in low rates is therefore beneficial only when the server is overloaded.

Figures 4.18 and 4.19 show that HTTP clients receive better performance when the server is not loaded, and therefore the Integrated model is indeed beneficial also in scheme 1, when the server becomes loaded.

4.5 Related Work

An approach to use multicast in the delivery of web resources was first introduced in [CA95]. In their architecture, clients use http requests. Several simultaneous requests are assembled, and grouped into a multicast group. Then, the clients join the dynamically generated multicast group and receive the page using a reliable multicast protocol. A later work [AAF98] suggested the use of cyclic multicast over UDP for the same architecture, while determining the amount of time the server has to cyclicly transmit the same page until all users in the group receive it. The disadvantage of using this mechanism lies in the unnecessary overhead compared with our scheme: TCP connections are first established, and only then closed. Overhead associated with building and maintaining multicast groups exists for each group, while many groups are created for the same hot page. As a matter of fact, the hotter the page, the more groups are created, and the bigger the overhead.

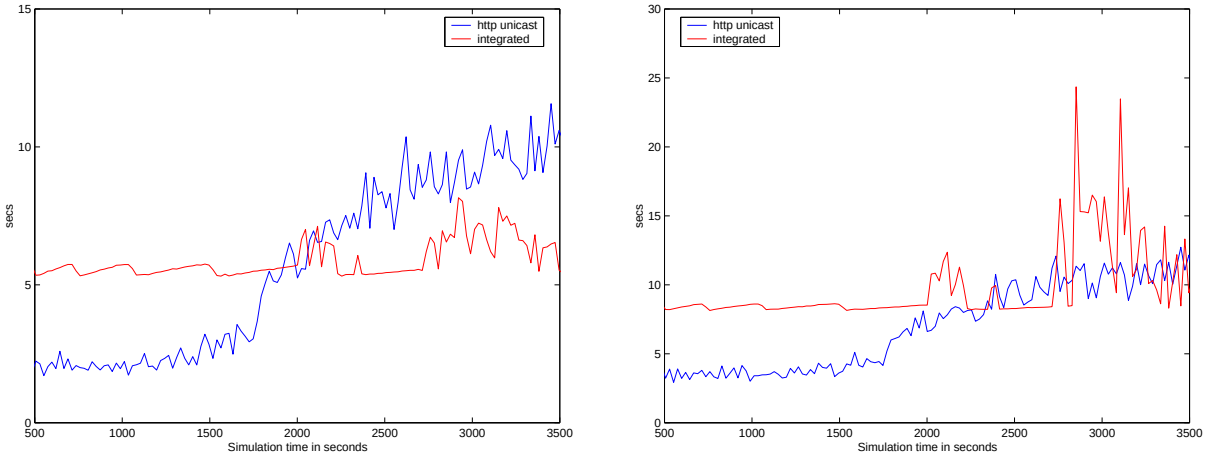

Figure 4.17: Scheme1, quadruple integrated request rate, Client Seen Delay: pages 6 and 7

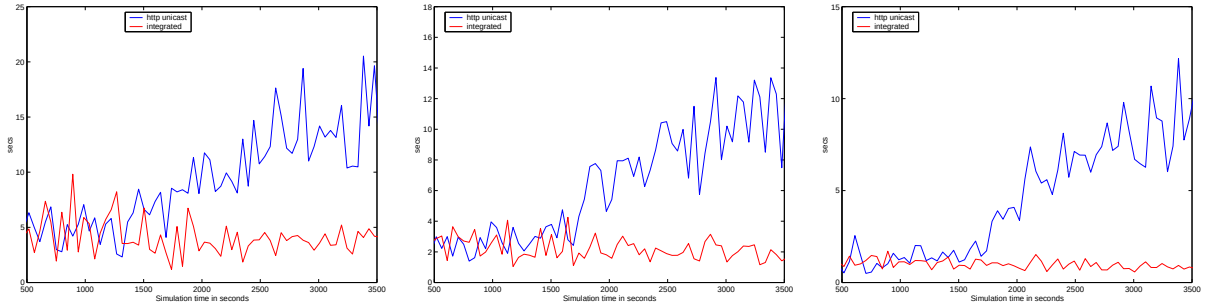

Figure 4.18: Scheme1, quadruple integrated request rate, Client Seen Delay: pages 1, 3 and 5

In [DD99] a scheme for the use of reliable multicast channels to deliver frequently updated objects to caches is suggested. Their scheme consists of control and data channels. Each channel is associated with a group of objects. A cache that is interested in any of the objects in a group subscribes to the corresponding channels. When an object in a group changes, a control message is sent over the control channel, so that all subscribed caches know to listen over the data channel. Then, the changed object is multicast over the data channel. The main disadvantage of this scheme lies in the grouping. While grouping enables the deployment of the scheme, due to scalability issues, it also causes for a lot of bandwidth waist. For example, if most of the caches subscribe to a group in order to receive object A, and another object, B, changes more frequently, it will be sent over the data channel each time to all subscribed caches. This scheme, when used on a channel per object basis, is not scalable. It requires two channels per object, and each cache needs to keep state for each object. The scheme alters the receivers as well as the senders, and requires state change for each channel. Unless grouped, it is not scalable, and compared to our scheme, the grouping causes bandwidth waist and some caches receive data they do not require.

4.6 Multi Level Architecture

In this section we present a way to integrate our mechanisms into the current hierarchical structure of the Internet. While our integrated architecture was first designed for sites, it seems just as beneficial for proxies, in general, and for proxies used by Internet service providers, in particular.

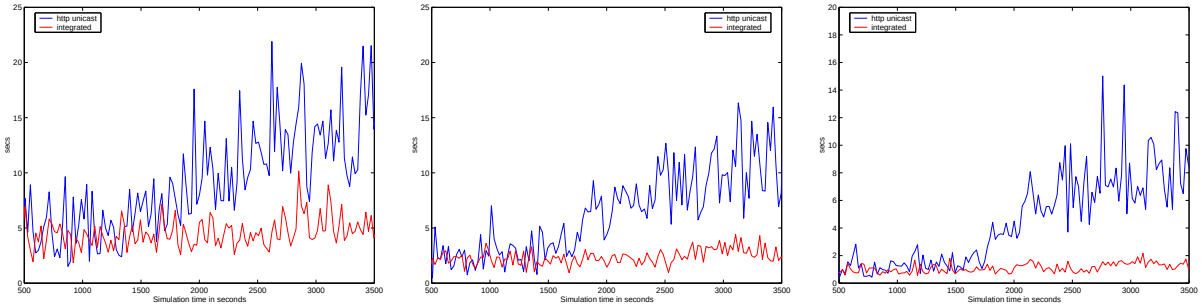

Figure 4.19: Scheme1 Client Seen Delay: pages 1, 3 and 5

Current ISPs maintain big autonomous networks and support very large number of users. In the integrated architecture suggested so far proxies play the role of clients. However, for their clients, proxies may take the role of the server. Thus, we suggest here the implementation of our integrated architecture one (or more) levels deeper. In this implementation, once a proxy detects an increase in the number of requests to a specific page, it multicasts this page to its clients, using our scheme. Using multicast for the delivery of hot pages within the ISPs' subnetwork can be very beneficial not only to the users but to the ISP as it both reduces its internal traffic and give its paying customers better service. It has special performance benefits when the structure of the subnet is tree-based, such as with cable network based service providers.

While the implementation of the scheme in the ISPs level is straight forward for hot pages with the *httpm* protocol specifier, another scheme is needed for locally cached hot pages. Such a scheme is beyond the scope of this paper, and requires further research.

4.7 Conclusions

Frequently changing web content requires the use of new mechanisms for its scalable delivery. Our integrated architecture enables the efficient and scalable delivery of such content. Its main advantage is its simplicity and transparency to users. Our results show that it allows sites to serve a growing amount of users at times of peak in load without experiencing performance degradation. Users delay decreases significantly, and both goodput and throughput quadruple. An important conclusion from our simulations is that, many times, performance degradation is due to increased load on core links. This increase is caused from the large amount of concurrent connections to sites, all aimed to get the exact same information. The result is congested links, which have the immediate effect of increasing the load on both the sites and the links in the short run because of retransmissions. Our scheme solves this problem by using multicast at such times, thus enabling economical use of core links.

Chapter 5

Internet Resiliency under BGP Routing

5.1 Introduction

In recent years there is a growing interest in the resiliency of the Internet, as it represents the network's availability in times of instabilities or under extreme conditions. Research in this field took two distinct paths. One is the stability of routing protocols in case of errors and failures [LGJ97, GW99], and the other, which also draws attention outside the computer networking community, focuses on the resiliency of the Internet to random failures and attacks on strategic locations [AJB00, CEbAH01, CEbAH00a]. Such events can happen as a result of a disaster, or manipulated online attack on key Internet elements.[1] In this research we focus on the latter.

Research in the field was motivated by the finding that Internet AS topology can be classified as scale free, belonging to a class of networks for which the connectivity resembles a power law distribution [FFF99a, GT00, MMB00, CNS+99]. In physics terminology, the susceptibility of the Internet to node deletion is considered in terms of network *phase transition*, representing the transition from a connected phase to a disconnected phase. The research in this field [AJB00, CEbAH01, CEbAH00a, BT02, PKP+03] showed that the Internet has a high tolerance to random failures, and does not break until more than 95% of the nodes have failed. On the other hand, it was found that the Internet is highly sensitive to deliberate attacks that target its most connected nodes. Under such an attack, the network transitions to a set of small disconnected components, after the removal of a small fraction of the highly connected nodes. Cohen et al. [CEbAH01] have shown that the rate of transition under a deliberate attack depends on the minimal connectivity, hence on the average degree. They have also shown that the average path length grows dramatically under such attack near the critical point of transition in which the network disintegrates.

A significant drawback of the works in [AJB00, CEbAH01, CEbAH00a, BT02, PKP+03] is that they treat the Internet as an undirected graph. However, routing in the Internet between the ASs is governed by policies that are set locally with the aid of BGP, the inter-network routing protocol, according to business agreements [III99]. The implication of policy based routing is that not every two nodes (ASs) that have a physical path connecting them can indeed exchange information; a valid path that conforms to the policies of the ASs along it must exist. These considerations and

[1] While the collapse of an entire large ISP seems unlikely, it actually happened a few times in the recent past for the largest AS in the Internet, UUNet. On April 22nd and October 3rd 2002 the UUNet network collapsed due to software problems in its routers, and in January 25th 2003 due to a DoS attack [lig].

agreements create a network far different from the one used in all the above listed works, and calls for revisiting the question on the resiliency of the Internet. In addition, the data used for obtaining the above results was of partial views of the Internet. These partial views, obtained mainly through dumps of BGP announcements, lack in connectivity due to two main reasons. The first is that these views are taken from a few sites in the Internet. While they contain most of the nodes, they lack in connectivity information, since they contain mostly links that are on the shortest BGP path from the source site to the other nodes [CCG+02a]. The second lies with the rules of the BGP protocol, which tend not to advertise a backup path which is not in current use.

In this work, we first suggest a paradigm for finding Internet connectivity under BGP policy routing based on existing business agreements. We discuss the different metrics suggested for measuring the resiliency of the network, and suggest our own. We find the resiliency of the Internet to attacks and random failures, and show that it is even more susceptible to attacks than previously found. We show that previous Internet models, which did not take into account the connectivity constraints imposed by policy based routing, yielded too optimistic results for the case of a deliberate attack. In the case of random failures of nodes, the results show that the difference in resiliency is small.

Our testbed consists of partial Internet views obtained from the Oregon site [Ore] and from European exchange points [Rip]. We also obtained the very rich database collected by Chen *et al.* [CCG+02a], who assembled 41 partial views along with added Looking Glass information and showed that the actual connectivity between ASs is higher than was previously known. Our results show that the added connectivity improves the resilience of the networks, and therefore results obtained on partial views are somewhat misleading. Moreover, hidden backup links which are used only in case of a disaster, would probably improve the resilience of the network even better. We made some first attempts to model how backup links may improve Internet reachability.

5.2 A Heuristic for Added Backup Connectivity

In this section, we make a first attempt to quantify the effect of existing backup links, which are usually not advertised through BGP until used, on the reachability and resiliency of the AS graph under attacks. We constructed a backup scenario, which relies on the existing connectivity, and provides alternate paths to small- and medium-sized ASs which connect only to one provider. These ASs, once their provider fails, use their peering links as backup links, effectively using them as customer-to-provider links. Since we do not add links to the existing graph, the effect of such a backup scenario is only meaningful in the case of attacks. As we have shown in Section 5.5.2, Internet reachability under random failures is very close to its connectivity. Therefore the added paths gained from using the backup links can hardly improve the resiliency in this case. However, in the case of attacks, it might allow single-homed ASs to use alternate paths. If there are many such ASs, which do not rely on multihoming, we expect an increase in both the size of the largest component and the reachability.

Our backup scenario is as follows:

- AS x has one provider.
- link $< x, y >$ is a peer link.

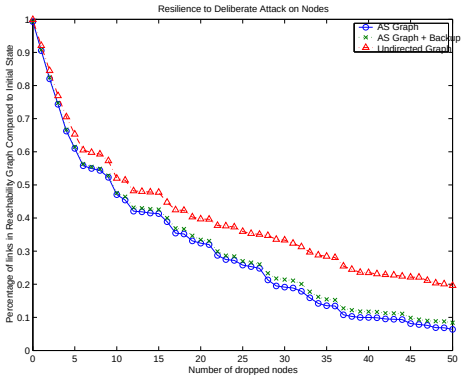

Figure 5.1: Added backup: Reachability under attacks in UM

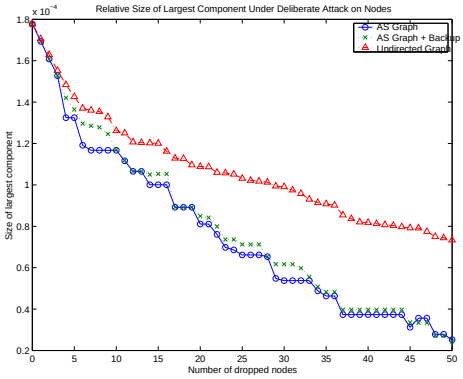

Figure 5.2: Added backup: Largest component size under attacks in UM

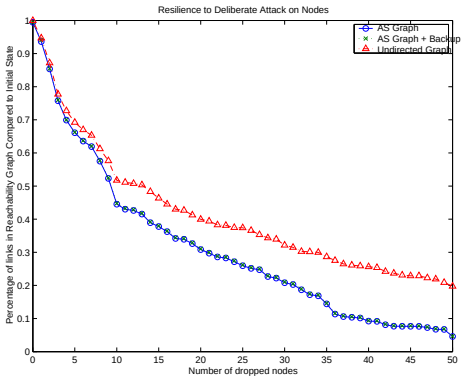

Figure 5.3: Added backup: Reachability under attacks in OR2

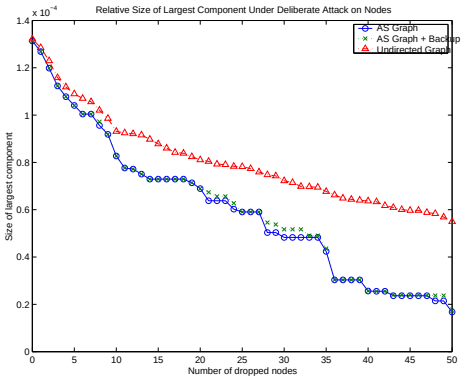

Figure 5.4: Added backup: Largest component size under attacks in OR2

- if AS x disconnects from its provider, then link $< x, y >$ becomes a customer to provider link.

We found that the added backup connectivity is more meaningful for the sparse topologies (LZ, OR1) than for the richer topologies (OR2, UM). We give here the results for the UM topology in Figs. 5.1 and 5.2. We see that the reachability is somewhat better with the added backup links, as is the size of the largest component, but the behavior of the topologies with and without the backup links is very similar. Figs. 5.3 and 5.4 show that for the newer partial-view OR2, which is also rich in connectivity, the reachability and size of the largest component are hardly effected by the added backup connectivities. These results suggest that there has been a vast increase in the number of ASs that use multihoming, and are therefore hardly affected if one of their providers fails.

A more strict backup scenario, which enabled two ASs to use a peer link between them as backup only if both ASs have only one provider, yielded even smaller improvement in both reachability and the size of the largest component. Additionally, we examined the resiliency to a partial

attack on the core, i.e., a few of the 30 most connected nodes were removed randomly. We examined the effect of the backup links in this scenario. The results showed no improvement in the size of the largest component, and a negligible improvement in reachability.

5.3 Modeling Reachability in a Directed AS Graph

In this section we characterize our graph model, and describe our reachability algorithm.

5.3.1 AS Graph Model

We model the AS graph as a *directed* graph, in which the set of nodes is the set of distinct autonomous systems and a link exists between two such nodes if the respective ASs have peering (business) relationship[2] and are BGP neighbors. For each link we maintain its direction and characteristics. For example, between two nodes that represent a provider and its customer, there will be an *uphill* link from the customer to the provider, and a *downhill* link from the provider to the customer. Between peers there is a directed peer edge in each direction, and between siblings there is an undirected link.

Connectivity in the AS graph with the valley free policy rules described in Section 5.4 maintains reflexivity, but does not maintain transitivity. For example, a small ISP with two providers reaches each of them on the directed link that connects the customer to its provider, but the providers can not use the two link path through the customer to communicate. An algorithm for finding the shortest path under these restrictions was suggested in [KBHL01]. The algorithm uses an adaptation of the Dijkstra shortest path algorithm to the AS graph, for the problem of proxy and cache location.

5.3.2 AS Reachability Algorithm

The reachability algorithm we developed maintains a reachability map. It finds, for each node, the set of nodes that can be reached from it in the policy constrained AS graph, regardless of the path taken. The algorithm does not look for the best path to a node, but rather, for each node, looks for all nodes reachable from that node through *some valid* AS path.

The algorithm is a free adaptation of a breadth first search (BFS) to the AS graph. Starting from a source node, the algorithm looks for uphill paths. A link can be taken only if by taking it the path is still a valid AS path, i.e., valley free. Each node, when reached for the first time, marks its state by the direction it was reached. Thus, a node reached through a downhill path is marked as *down*, etc. Then, the node examines all of its neighbors. A link to a neighbor is taken only if it provides a valid AS path.
A node can be reached again only if the path taken through it can open more opportunities. For example, a node in state *down* that is reached by an uphill path will explore it and change its state to the *up* state. Thus, each node can be in one of the following states:

[2]Note that the derivatives of “peer” appears in two distinct meaning. We say that two ASs have “peering relationship” if they exchange BGP messages, and that two ASs are “peers” if they have peer-to-peer exchange agreement in BGP, namely if they are neither provider-customer nor siblings.

none - The node has not been traversed yet.

up - The node was in either *none*, *side* or *down* states, and there is an uphill path that can be traversed through it.

side - The node was in either *none* or *down* state and there exists a peer link that can be traversed through it.

down - The node was in *none* state, and there is a downhill path that can be traversed through it.

The algorithm gives the highest priority to an uphill path through a node, the next priority to traversing a peer-to-peer link from that node, and the lowest priority to a downhill path through the node. Each node, once reached, examines all of its links. A link is taken only if by taking it the state of the node reachable through it can be improved, according to the description above.

Figure 5.5 presents a formal description of the algorithm. The following variables are used in the algorithm: V is the set of all nodes representing ASs in the graph; $\mathcal{R}_i$ is the reachability bitmap of node i, in which bit j is set if there is a legal BGP path between node i and node j, and st_i is the state of node i. $\mathcal{N}_i$ is the set of immediate BGP neighbors of node i.

The algorithm time complexity is as follows. Each node starts in state *none*, and can change its state at most three times. Hence, each node is reached *at most* three times, giving a worst case time complexity of $O(|E|)$, where E is the number of links in the graph (in a worst case scenario, each link is examined three times).

5.3.3 Anomalies in the AS Graph

The algorithm described in Figure 5.5 is resilient to anomalies in the AS graph. However, there are two anomalies that need to be considered. The first, and more rare, is called a black hole, and the second is inference mistakes.

Today, each AS administration is responsible for advertising its own CIDR prefix through the BGP protocol. A black hole happens when a wrong prefix is announced by a BGP speaker, thus hiding from other ASs the real owner of the prefix. Such a phenomena can be traced today only through the awareness of network managers, and solved only through exhaustive search and mailing list queries (an interesting example is AS3908, which used to be SuperNet Inc., and was acquired by Qwest Inc. It had wrongly advertised prefix 157.237.0.0/16 instead of the more specific 157.237.144.0/24 which it owned, hiding away a small ISP in Sweden. For complete trace of events follow [Nan]).

Gao's inferring algorithm was shown to be 97% accurate on a test case database of AT&T, having inference problems only for links suspected as siblings. Out of the 3% inferred as sibling links, the actual relationships obtained from the AT&T data were almost half peering links, a quarter customer-provider links, and only the rest were actual sibling links. Battista *et al.* [BPP03] have investigated the anomalies in AS graphs, showing that the problem of solving the AS relationships while minimizing the anomalies is NP-hard in the general case, and suggested heuristics for minimizing the number of anomalies. A recent work [HBC03], that compares traceroutes to BGP AS paths, finds that much of the disparity results from ASs connected through exchange points, and by groups of ASs under the same ownership.

Algorithm 3 (Reachability Algorithm)

1. $\forall v \in V$ do:
2. set $s \leftarrow v$; direction $\leftarrow$ *up*;
3. set $\mathcal{R}_s \leftarrow \emptyset$; st_s = *up*;
4. inspect(s, direction)
5. output $\mathcal{R}_s$

6. function inspect(k,d)
7. set k in $\mathcal{R}_s$
8. $\forall i \in \mathcal{N}_k$ do:
9. if $< k, i >$ = *sib_link* then
10. inspect(i, d)
11. return
12. switch d: * Note the fall through *
13. case *up*:
14. if $< k, i >$ = *customer_to_provider* then
15. if $st_i =$ *none* or *side* or *down* then
16. $st_i \leftarrow$ *up*
17. inspect(i,*up*);
18. case *side*:
19. if $< k, i >$ = *peer* then
20. if $st_i =$ *none* or *down* then
21. $st_i \leftarrow$ *side*
22. inspect(i,*down*);
23. case *down*:
24. if $< k, i >$ = *provider_to_customer* then
25. if $st_i =$ *none* then
26. $st_i \leftarrow$ *down*
27. inspect(i,*down*);
28. return

Figure 5.5: A formal description of the basic algorithm for the root node.

To obtain accurate results, we inferred manually through the use of WHOIS servers and Internet searches all of the automatically inferred sibling relationships in the databases obtained from [Rip]. For a combined view of the London and Zurich exchange points, gathered at the same time for the same set of ASs, we obtained the following results: Out of the 81 inferred sibling relations, only 32% (26) were actual siblings. 27% (22) were peers, 8% (7) were customer-to-provider links and 32% (26) were provider-to-customer links.

5.3.4 Metrics for defining Resiliency

The problem of finding the right metric for evaluating the network resiliency was reduced in previous works to the problem of finding the connectivity of the graph [AJB00, CEbAH00a, CEbAH01, PKP+03]. Although the problem itself remains an open problem [FFF99a], the above mentioned works used some of the following metrics: Average diameter or average shortest path length $\overline{d}$; the giant component size S; the number of connected node pairs in the network, K; diameter-inverse-K, *DIK*.

The definition of $\overline{d}$ is as follows: let $d_{min}(v,u)$ denote the minimal path between any connected pair of distinct nodes u and v, and Π the number of such distinct node pairs. Then: $\overline{d} = \frac{\sum d_{min}(v,u)}{|\Pi|}$. According to [CEbAH01] $\overline{d}$ can be used to asses when a network under attack reaches criticality. A measure of the size of the largest component, S, is the ratio between the number of nodes in the largest connected component and the number of nodes in the graph. The two metrics K and *DIK*, defined in [PKP+03], are as follows. K describes the whole network connectivity, by measuring all connected node pairs in a network: let Ψ be the number of distinct node pairs, and Π defined as above, then: $K = \frac{|\Pi|}{|\Psi|}$. Park *et al.* [PKP+03] have suggested a different version of K, *DIK*, which measures both the expected distance between two nodes and the probability of a path existing between two arbitrary nodes: $DIK = \frac{\overline{d}}{K}$.

We noted that the measures described above cannot be directly applied in our case, when reachability is *not* equivalent to connectivity, since the directed AS graph lacks transitivity. In this case, for example, the minimal distance between two nodes, $\overline{d}$, becomes the minimal BGP distance between two nodes, depending on policy constraints. Thus, we chose two different ratios, that capture best, in our understanding, the actual resilience of the Internet.

The first, denoted by R, captures the reachability of the Internet, and is defined as follows: let $r(v,u) \in 0,1$ denote reachability between an arbitrary distinct pair of nodes v and u, $v,u \in V$, where V is the set of nodes describing ASs. Let Π_r denote the number of distinct node pairs in the graph, for which $r = 1$, and let O_r denote the theoretical limit of Π_r for the Internet (when there are no failures in the Internet we expect to have full reachability between all ASs). Then, we define R as the ratio:

$$R = \frac{\Pi_r}{O_r}.$$

The second metric quantifies the size of the strongly connected component in the directed AS graph, termed RS. We create a *reachability graph*, in which there exists an edge between two nodes v and u if and only if $r(v,u) = 1$.[3] Then, in order to find the largest strongly connected component in the original graph, we need to find the maximal clique in the reachability graph. The problem of finding the maximal clique in a graph is NP-complete [Kar72]. The best known ap-

[3]Note that while reachability is not transient, it is symmetric under the valley free rule.

proximation for finding the maximal clique [BH92] gives an $O(n/(\log_n)^2)$ performance guarantee. Hence, for our topologies, we can expect a maximal mistake of around 0.6% (see Table 5.1). We use a greedy heuristic for finding the maximal clique in the graph. Since we know which nodes still exist in the graph after the simulated failure or attack, and their respective degrees, we start with the one with the largest degree. Due to the hierarchical nature of the Internet [TGJ+02], it is likely that such a node resides in the core, and therefore is used by many other nodes for reachability. We denote all of the nodes reachable from that node by C. Then, iteratively, we look for the maximal degree node in C, i, and extract from C all the nodes not reachable from i. We continue this process until all the nodes in the component are reachable from each other. The process is repeated several times with different starting nodes selected from the top connected ones. The size of the strongly connected largest component, S_d, is then divided by the number of nodes in the original graph, to obtain the ratio RS.

5.3.5 Critical Point of Failure (Phase Transition)

From a physical point of view, a phase transition occurs only when the network disintegrates [CEbAH01]. The network is considered connected as long as RS, the ratio between the size of the largest component and the number of initial nodes in the graph, is a fraction of the number of nodes in the graph. For example, the removal of the top 20% of the nodes of a 100 nodes network, yields $RS = 0.2$. For a network with the same connectivity distribution, regardless of its size, any such removal of the top 20% of the nodes will yield a similar RS. Thus, as long as the size of the largest component is a fraction of the initial size of the network, the network is considered connected. The phase transition occurs when $RS \approx 1/N$, where N is the initial number of nodes in the network. Hence, physically speaking, the network is considered disintegrated only when the size of the largest component is one. The same discussion holds for the reachability function, R.

¿From a routing perspective, reachability is considered lost long before the Internet disintegrates. We assume here, that when $R < 0.5$, i.e., the overall reachability is less than 50% of the original reachability, or when $RS < 0.5$, i.e., the comparable size of the largest component is half the original network, the network is no longer considered connected.

5.4 Background on AS Connectivity and Internet topology

The Internet today consists of thousands of subnetworks, each with it own administrative management, called autonomous systems (ASs). Each such AS uses an interior routing protocol (such as OSPF, RIP) inside its managed network, and communicates with neighboring ASs using an exterior routing protocol, called BGP. The BGP protocol enables each administrative domain to decide which routes to accept and which to announce. Through the use of the protocol the autonomous systems select the best route, and impose business relationships between them on top of the underlying connected topology. As a result, paths in the Internet are not necessarily the shortest possible, but rather the shortest that conform to the ASs' policies. Such routing is called policy-based routing.

The commercial agreements between the ASs create the following peering relationships: customer-provider and provider-customer, peer-to-peer, and siblings. A customer pays its provider for transit services, thus the provider transits all information to and from its customers. The cus-

tomer, however, will not transit information for its provider. For example, a customer will not transit information between two of its providers, or between its provider and its peers. Peers are two ASs that agree to provide transit information between their respective customers. Such agreements are very common between ASs that connect at an exchange point (IX) and between smaller ISPs residing at the same geographical vicinity. In sibling relationships, the two ASs provide full transit services for each other. Such relationships are mainly due to financial acquisitions, mergers, or to a smaller degree, business transactions between smaller ISPs that maintain their own administration but unify their networking services.

In a pioneering work, Lixin Gao [Gao00] suggested an algorithm for inferring the type of relationships between ASs through their advertised BGP paths. The algorithm assumes that the degree of connectivity of an autonomous system is an indication of its size, and infers the relationships between the ASs according to a set of rules obtained from the above description of commercial relationships. Gao has deduced, that a legal AS path may take one of the following forms:

1. *Up hill* path, followed by a *down hill* path.

2. *Up hill* path, followed by a peering link, followed by a *down hill* path.

Where an *up hill* path is a sequential set, possibly empty, of customer-provider links, and a *down hill* path is a sequential set, possible empty, of provider-customer links. Thus, a legal route between autonomous systems can be described as a *valley free* path. A peering link can be traversed only once in each such path, and if it exists in the path it marks the turning point down hill.

Further work on AS relationships [SARK02] have characterized the Internet as hierarchical. They found that the top big American providers form a core with almost complete clique connectivity, and the second layer around this core consists of big providers from the USA and Europe, characterized mainly by their very rich connectivities to the core. The third layer consists of smaller providers, and forms the majority of the network. Recent works have investigated the relations between ASs, looking for anomalies and their possible solutions [BPP03, PKP+03].

Inferring the AS relationships can be viewed as part of an ongoing effort to discover and map the exact topology of the Internet [FFF99a, CNS+99, MMB00, CCG+02a, BC99, GT00]. It is generally agreed today that the Internet, at the AS level, has a highly heterogeneous connectivity patterns, with a highly variable vertex degree distribution. Several works have also tried to characterize the growing mechanisms of the Internet and model it [BA99, AB00, BT02, PKP+03], and several networks generators which rely on some of these algorithms exist [JcJ00, MLMB01, DMS03] and evaluated [RTY+00, TGJ+02, MSZ02, BT02].

In all previous works on the resilience of the Internet, it was assumed that the connectivity of the network is equivalent to its reachability. We show in this work that the two are not equivalent, and find the actual reachability of the network under different constraints.

5.5 Resiliency of The Internet

In this section we present our results for the resiliency of the Internet to random failures and attacks, given the policy routing constraints.

Table 5.1 describes the different data sets used in these tests and their characteristics. The topologies differ mainly in their connectivity. The LZ dataset, from the RIPE Routing Information

Service [Rip], is the result of combining routing information from two exchange points, one in London and the other in Zurich. The data lacks most of the largest top US providers. The largest AS in this data set has a rather low degree of 1958, and the average degree in the set is also rather low. This implies that there are fewer alternative paths between the nodes in this topology, i.e., less redundancy, and therefore we expect it to be the most vulnerable to deliberate attacks. As discussed in Section 5.3.3 the topology is also inference-anomaly free, as all automatically inferred sibling relations were manually checked using WHOIS databases and Internet searches. Datasets OR1 and OR2 are both partial views from the Oregon routeview project [Ore], collected March and April, 2003 respectively. The topologies differ greatly in the richness of the connectivity, as OR2 has 27% added connectivity compared to OR1. The last view, and the richest in connectivity, UM, is the enriched topology obtained by Chen *et al.* [CCG+02a]. Although collected three years ago, the topology is the richest in connectivity, since it was collected from 41 BGP databases and augmented with summary data from different looking glass sites. The ongoing growth of the Internet, which increases its average degree, implies that such an enriched view of today's Internet will yield a much higher average degree than seen from the partial views OR1 and OR2.

In the graphs presented in this section, we compare the resiliency of the policy-constrained AS graph, referred to as the *directed* graph or the *reachability* graph to the resiliency of the graph used in previous works, referred to as *undirected* graph. For each topology, we present both the reachability R and the evaluation of the largest component, RS, as discussed in Section 5.3.4. Some of the partial views do not have 100% *reachability* to begin with, as can be seen in Fig. 5.6, for example.

Name	source	Date	No. of ASs	No. of Links	Avg. Degree	Max Degree
LZ	RIS	2002/07/03	13393	22001	3.28545	1958
OR1	Oregon	2003/03/01	14704	24020	3.26714	2330
OR2	Oregon	2003/04/01	15128	31426	4.15468	2503
UM	Umich	2001/05/26	11204	25980	4.63763	2417

Table 5.1: Characteristics of data sets used

5.5.1 Resiliency of the Internet to Deliberate Attacks

We evaluate the resiliency of the Internet to deliberate attacks by targeting the topology's most connected ASs, dropping each time the next most connected node in the graph, and measuring both metrics R and RS for the directed and undirected graphs.

Figs. 5.6 and 5.7 show the resiliency of the LZ topology to deliberate attacks. Even before any node was dropped, the reachability is less than the connectivity, due to the partiality of the topology. The same can be seen in Fig. 5.8 and 5.9, representing the resiliency of topology OR1, also a rather sparse partial view. However, in the more connected views, OR2 and UM, the partial view gives a fully connected network, in which all nodes are reachable to begin with.

In the sparse topologies the overall reachability decreases very fast. Fig. 5.6, which starts from a 95% reachability, shows that after dropping only the sixth most connected nodes, the gap in reachability is 12%. The gap is even larger when we check how much of this reachability is within the same component of nodes that communicate with each other (Fig. 5.7). After the

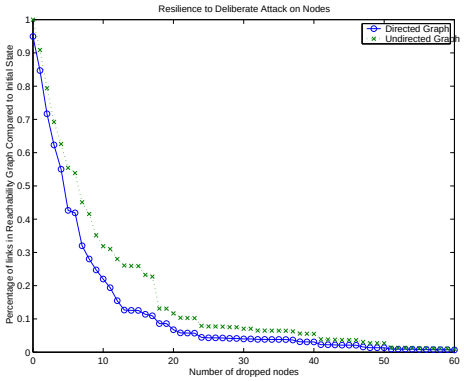

Figure 5.6: Reachability under attacks in LZ

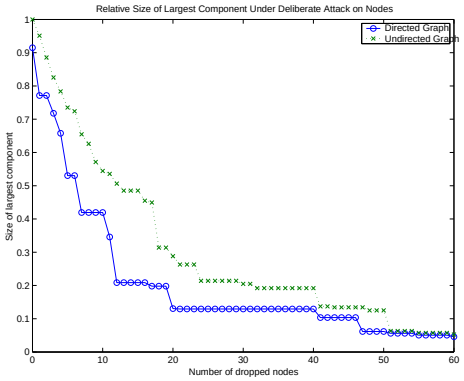

Figure 5.7: Largest component size under attacks in LZ

removal of these six nodes, there is a gap of 19%. The gap between reachability and connectivity increases as the network starts to break up—after dropping the 12th most connected nodes the largest component consists of only 20% of the nodes in the topology, while previously it was thought that it still consists of 50% of the nodes, as we can see from the results for the undirected graph.

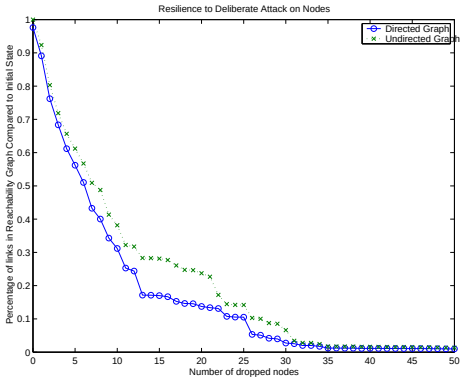

Figure 5.8: Reachability under attacks in OR1

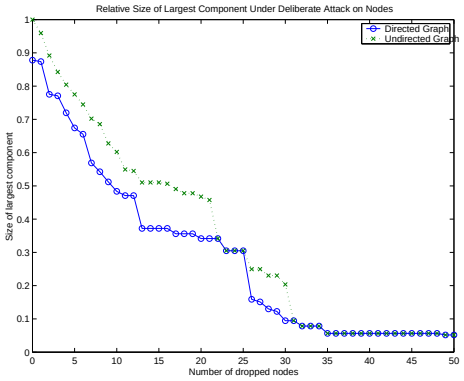

Figure 5.9: Largest component size under attacks in OR1

Figs. 5.8 and 5.9 give similar results, namely, that the Internet is much more susceptible to deliberate attacks than previously thought. While the overall reachability drops at the same rate as the connectivity, it can be seen from Fig. 5.9 that the first node that was dropped was a large AS with a lot of customers, that lost reachability to the rest of the network. After the eighth most connected nodes were removed, the size of the largest component is less than 50% the size of the network, while in the case of the undirected graph it contains 69% of the nodes. We see here that after attacking only the 8th most connected nodes, the Internet's largest component contains less than 50% of the nodes.

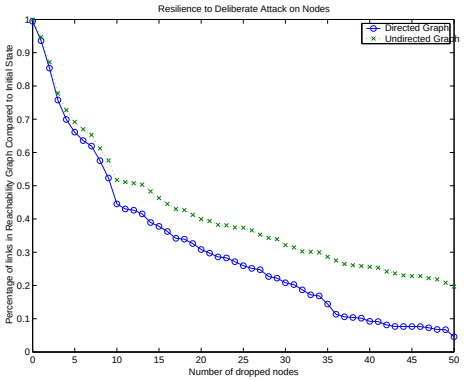

Figure 5.10: Reachability under attacks in OR2

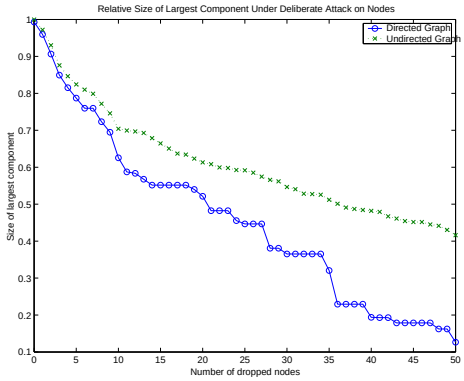

Figure 5.11: Largest component size under attacks in OR2

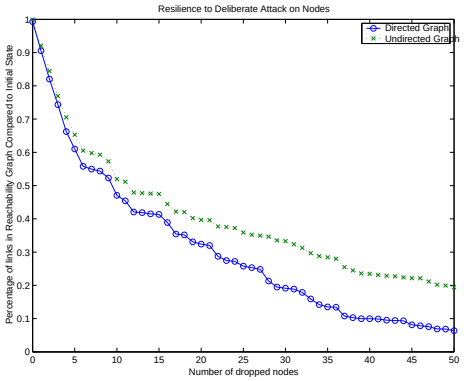

Figure 5.12: Reachability under attacks in UM

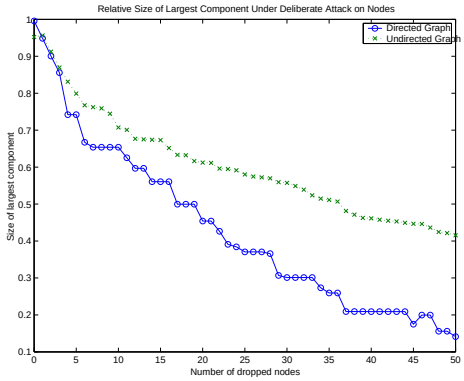

Figure 5.13: Largest component size under attacks in UM

Figs. 5.10,5.11, 5.12 5.13 represent the resiliency of highly connected topologies (OR2 and UM), in which most nodes can be reached through several AS paths. Therefore, we expected that the resiliency of the directed AS graph would resemble the one of the undirected graph. Indeed, the reachability R is almost the same for both topologies when the five most connected nodes are dropped, since alternate paths are taken. The size of the largest component is also quite similar for both the directed and undirected graphs, although for the UM topology (Fig. 5.13) the gap between the component sizes reaches almost 7% after the removal of only 5 nodes. In all cases, the gap between the directed and undirected graphs increases after the ten most connected nodes are removed. After the removal of 28 nodes the gap in largest component size is over 15% (Figs. 5.10, 5.13). After removing the 50 most connected nodes, for the highly connected OR2 topology, the network disintegrated to the point where the largest component holds only 4% of the nodes. In the unconnected case, the component holds more than 42% of the nodes, an order of magnitude difference.

In summary, the resilience of the Internet, under current policy constraints, to deliberate at-

tacks against the top most connected ASs is much lower than previously found. On all partial views we obtained, the largest component dropped to less than 50% the size of the network after the removal of only the ten most connected nodes. The network disintegrated completely after the removal of the top 30 ASs (0.2% of the number of ASs).

5.5.2 Resiliency to Random Failures of Nodes

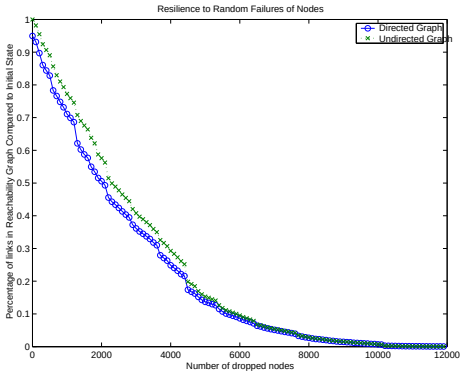

Figure 5.14: Reachability under random failures in LZ

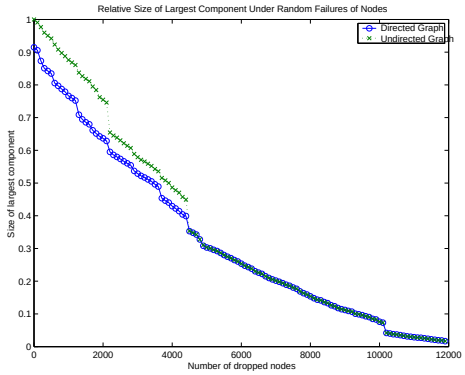

Figure 5.15: Largest component size under random failures in LZ

We checked the resiliency of the Internet to random failures by a random removal of 100 nodes at a time, until more than 95% of the nodes were removed.

Figs. 5.14 and 5.15 show the comparable resiliency of the Internet to random failures for the LZ topology. As previously found, the Internet is not susceptible to such random failures, and both R and RS do not fall below 0.8 even after the removal of 1000 nodes. The network starts to break down only after the removal of more than 2000 random nodes. the Internet disintegrates only after the removal of almost 95% of the nodes. The difference between the two graph models, the directed (policy-constrained) AS graph, and the undirected graph, is small. However, we found that the gap is larger for the sparse views than for the views richer in connectivity, where it is negligible. This result is somewhat surprising, indicating that the Internet maintains reachability of almost the same degree as its connectivity, under random failure of nodes.

Due to the high degree of the nodes in the core, and the fact that these nodes are rare in a scale free distribution, the statistical probability that they will be removed in a random failure scenario is low. However, it could be expected that the removal of small and medium sized nodes will effect the reachability of the smaller ASs and therefore the size of the largest connected component. The surprising results, indicating that the reachability is very close to the possible limit, the undirected connectivity, prove differently. These results may indicate that most ASs use multihoming to several providers, and thus are less susceptible to these random failures.

5.6 Conclusions

We examine the resiliency of the Internet to deliberate attack and random failures at the AS level, given that routing paths conform with the policy imposed by BGP. We compare our findings with previous findings that did not consider these constraints, and evaluated reachability as connectivity. We suggest an efficient algorithm that determines reachability in such AS graphs, and discuss and suggest metrics for measuring the resiliency.

Our results show that the Internet is much more susceptible to deliberate attacks than previously found, and that reachability, as well as the size of the largest component, drop to less than half after the removal of the 25 most connected nodes—less than 0.2% of the nodes. The Internet also disintegrates much faster than previously found, under an attack that targets the top 0.5% ASs. We also found that the Internet is rather resilient to random failures, and its reachability is surprisingly close to the graph connectivity without policy constraints. These results can be attributed to that routing in the Internet is mainly through its core of highly connected ASs.

Our initial results on the effect of backup links suggests that they do not improve resiliency of the Internet by much. The decrease in the added resiliency of the partial views over the years suggest that ASs tend today to rely more on multihoming, and thus are less susceptible to a failure of one of their providers. We believe that further research to model backup connectivities at the AS level is important.

Bibliography

[AA97] K. Almeroth and M. Ammar. Multicast group behavior in the internet's multicast backbone (mbone). *IEEE Communications*, June 1997.

[AAF98] K.C. Almeroth, M. Ammar, and Z. Fei. Scalable delivery of web pages using cyclic-best-effor (udp) multicast. *Infocom'98*, March 1998.

[AAN02] A. Alouf, E. Altman, and P. Nain. Optimal online estimation of the size of a dynamic multicast group. In *IEEE INFOCOM '02*, New York, NY, USA, 2002.

[AB00] Réka Albert and Albert-László Barabási. Topology of evolving networks: local events and universality. *Physical Review Letters*, 85(24):5234–5237, 11 December 2000.

[AB02] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Rev. Mod. Physics*, 74(47), 2002.

[AJB00] Réka Albert, H. Jeong, and Albert-László Barabási. Attack and error tolerance of complex networks. *Nature 406*, page 378, 2000.

[Ang00] J. Angel. Caching in with content delivery. *NPN: New Public Network Magazine; http://www.networkmagazine.com*, March 2000.

[BA99] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *SCIENCE*, 286:509 – 512, 15 October 1999.

[BC99] H. Burch and B. Cheswick. Mapping the internet. *IEEE Computer*, 32(4):97–98, 1999.

[BCF+99] L. Breslau, P. Cao, L. Fan, G. Philips, and S.Shenker. Web caching and zipf-like distributions: evidence and implications. *Infocom'99*, March 1999.

[BH92] Ravi Boppana and Magnús M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT*, 32:180–196, 1992.

[BLMR98] J.W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. *Sigcomm'98, Vancouver, Canada*, September 1998.

[BPP03] G.D. Battista, M. Patrignani, and M. Pizzonia. Computing the types of relationships between autonomous systems. In *IEEE Infocom 2003*, San-Francisco, CA, USA, April 2003.

[BT02] T. Bu and D. Towsley. On distinguishing between internet power law topology generators. In *IEEE Infocom 2002*, New-York, NY, USA, April 2002.

[BTW94] J-C. Bolot, T. Turletti, and I. Wakeman. Scalable feedback control for multicast video distribution in the internet. In *ACM SIGCOMM 1994*, pages 58–67, London, UK, September 1994.

[CA95] R. Clark and M. Ammar. Providing scalable web services using multicast communication. *Proceedings of the IEEE Workshop on Services in Distributed and Networked Environments, Whistler, Canada*, February 1995.

[CA01] R. C. Chalmers and K.C. Almeroth. Modeling the branching characteristics and efficiency gains in global multicast trees. In *IEEE INFOCOM'01*, Anchorage, Alaska, 2001.

[CbAH02] R. Cohen, D. ben Avraham, and S. Havlin. Structural properties of scale free networks. In S. Bornholdt and H. G. Schuster, editors, *Handbook of graphs and networks*, chapter 4. Wiley-VCH, 2002.

[CCG+02a] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. The origin of power-laws in internet topologies revisited. In *IEEE Infocom 2002*, New-York, NY, USA, April 2002.

[CCG+02b] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. Shenker, and W. Willinger. The origin of power-laws in internet topologies revisited. In *IEEE Infocom 2002*, New-York, NY, USA, April 2002.

[CDH+] R. Cohen, D. Dolev, S. Havlin, T. Kaliski, O. Mokryn, and Y. Shavitt. In *Hebrew University technical report TR49 (2002) and cond-mat/0305582*.

[CEbAH00a] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin. Resilience of the internet to random breakdowns. *Physical Review Letters*, 4626:85–89, 2000.

[CEbAH00b] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin. Resilience of the internet to random breakdowns. *Physical Review Letters*, 85:4626–4628, 2000.

[CEbAH01] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin. Breakdown of the internet under intentional attack. *Physical Review Letters*, 86:3682, 2001.

[CH03] R. Cohen and S. Havlin. Ultra small world in scale free graphs. *Physical Review Letters*, 90:058701, 2003.

[CNS+99] W. Cheswick, J. Nonnenmacher, Cenk Sahinalp, R. Sinha, and K. Varadhan. Modeling internet topology. Technical Report Technical Memorandum 113410-991116-18TM, Lucent Technologies, 1999.

[CS98] J. Chuang and M. Sirbu. Pricing multicast communication: A cost based approach. In *INET'98*, Geneva, Switzerland, 1998.

[DD99] D.Li and D.R.Cheriton. Scalable web caching of frequently updated objects using reliable multicast. *2nd USENIX Symposium on Internet Technologies and Systems (USITS)*, October 1999.

[DFKM97] F. Douglis, A. Feldman, B. Krishnamurthy, and J. Mogul. Rate of change and other metrics: a live study of the world wide web. *In Proceedings of USENIX symp. on Internet Tech. and Systems Monterey, CA*, December 1997.

[DMS01] S.N. Dorogovtsev, J.F.F. Mendes, and A.N. Samukhin. Size-dependent degree distribution of a scale-free growing network. *Phys. Rev. E*, 63:062101, 2001.

[DMS03] Danny Dolev, Osnat Mokryn, and Yuval Shavitt. On multicast trees: Structure and size estimation. In *IEEE Infocom 2003*, San-Francisco, CA, USA, April 2003.

[Dor00] A. Dornan. Farming out the web servers. *NPN: New Public Network Magazine; http://www.networkmagazine.com*, March 2000.

[DST00] V. Duvvuri, P. Shenoy, and R. Tewary. Adaptive leases: A strong consistency mechanism for the word wide web. *In Proceedings of Infocom'00*, 2:834 – 843, March 2000.

[ER60] Paul Erdös and A. Rényi. On the evolution of random graphs. *Publ Math Inst Hungar Acad Sci*, 5:17–61, 1960.

[FFF99a] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM 1999*, Boston, MA, USA, August/September 1999.

[FFF99b] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM*, August 1999.

[FJM+95] S. Floyd, V. Jacobson, S. McCanne, C.G. Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. In *ACM SIGCOMM 1995*, New-York, NY, USA, 1995.

[FT99] Timur Friedman and Don Towsley. Multicast session membership size estimation. In *IEEE INFOCOM '99*, New York, NY, USA, March 1999.

[Gao00] Lixin Gao. On inferring automonous system relationships in the internet. In *IEEE Global Internet*, November 2000.

[GC89] C. Gray and D. Cheriton. Leases: An efficient fault tolerant mechanism for distributed file cache consistency. *In Proceedings of the 12th ACM Symposium on Operating Systems Principles*, pages 202–210, 1989.

[GT00] Ramesh Govindan and Hongsuda Tangmunarunki. Heuristics for internet map discovery. In *IEEE Infocom 2000*, pages 1371–1380, Tel-Aviv, Israel, March 2000.

[GW99] Timothy Griffin and Gordon T. Wilfong. An analysis of BGP convergence properties. In *ACM SIGCOMM 1999*, pages 277–288, 1999.

[HBC03] Young Hyun, Andre Broido, and K Claffy. Traceroute and bgp as path incongruities. In *Cooperative Association for Internet Data Analysis - CAIDA*, San Diego Supercomputer Center, University of California, San Diego, USA, 2003.

[HC02] H. Holbrook and B. Cain. Source-specific multicast for IP, November 2002. IETF Internet draft draft-ietf-ssm-arch-01.txt.

[III99] John W. Stewart III. *BGP4 Inter-Domain Routing in the Internet*, volume 1. Addison-Wesley, first edition, 1999.

[JcJ00] C. Jin, Q. chen, and S. Jamin. Inet: Internet topology generator. In *Technical Report CSE-TR-433-00, University of Michigan, EECS dept.*, http://topology.eecs.umich.edu, 2000.

[Kar72] R. M. Karp. Reducibility among combinatorial problems. *R. Miller and J. Thatcher, editors, Complexity of Computer Computations*, Plenum Press:85–103, 1972.

[KBHL01] K.M. Kamath, H.S. Bassali, R.B. Hosamani, and L.Gao. Policy-aware algorithms for proxy placement in the internet. In *ITCOM 2001*, Denver, CO, USA, August 2001.

[Knu97] Donald E. Knuth. *The Art of Computer Programming*, volume 1. Addison-Wesley, third edition, 1997.

[KRS00] P. Krishnan, Dan Raz, and Yuval Shavitt. The cache location problem. *IEEE/ACM Transactions on Networking*, 8(5):568–582, October 2000.

[LBCX03] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in ip topology measurements. In *IEEE INFOCOM'03*, San Francisco, CA, USA, March 2003.

[LC97] C. Liu and P. Cao. Maintaining strong cache consistency in the world wide web. *In Proceedings of the ICDCS*, May 1997.

[LGJ97] C. Labovitz, G.R.Melan, and F. Jahanian. Internet routing instability. In *ACM SIGCOMM 1997*, August 1997.

[lig] www.lightreading.com: "WorldCom's IP Outages: Whodunnit?" April 25th, 2002; "WorldCom Outage Only the Start" October 14, 2002 ; "The Internet Has Broken" January 25, 2003.

[Mil67] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.

[MJSAA02] Andr Auto Moreira, Jr. Jos S. Andrade, and Lus A. Nunes Amaral. Extremum statistics in scale-free network models. *Physical Review Letters*, 89:268703, 2002.

[MLMB01] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. Brite: An approach to universal topology generation. In *In Proceedings of MASCOTS 2001*, IEEE Computer Society, August 2001.

[MMB00] A. Medina, I. Matta, and J. Byers. On the origin of power laws in internet topologies. *ACM Computer Communications Review*, 30(2):18–28, 2000.

[Mog00] J.C. Mogul. Squeezing more bits out of http caches. *IEEE Network magazine*, pages 6 – 14, May 2000.

[Mok01] Osnat Mokryn. Notre dame based topology generator, 2001. http://www.cs.huji.ac.il/∼osnaty/togend.html.

[MR98] M. Molloy and B. Reed. The size of the giant component of a random graph with a given degree sequence. *Combinatorics, Probability and Computing*, 7:295–305, 1998.

[MSZ02] Milena Mihail, Amin Saberi, and Ellen Zegura. Graph theoretic enhancements of internet topology generators. In *DIMACS Workshop on Internet and WWW Measurement, Mapping and Modeling*, Piscataway, NJ, USA, February 2002.

[Nan] Noc contact for as3908 (supernet inc.)? http://www.cctec.com/maillists/nanog/historical/0301/msg

[NB98] Jörg Nonnenmacher and Ernst W. Biersack. Optimal multicast feedback. In *IEEE INFOCOM '98*, San Francisco, CA, USA, March 1998.

[NB99] Jörg Nonnenmacher and Ernst W. Biersack. Scalable feedback for large groups. *IEEE/ACM Transactions on Networking*, 1999.

[ns-] UCB/LBNL/VINT Network Simulator - ns (version 2), 1997. URL: http://www.isi.edu/nsnam/ns.

[NSW01] M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Random graphs with arbitrary degree distributions and their applications. *Phys. Rev. E*, 64:026118, 2001.

[Ore] University of oregon route views project. http://www.antc.uoregon.edu/route-views/.

[PG98] J. Pansiot and D. Grad. On routes and multicast trees in the internet. *ACM Computer Communications Review*, 28(1):41–50, 1 January 1998.

[PKP+03] S.T. Park, A. Khrabrov, D.M. Pennock, S. Lawrence, C. L. Giles, and L. H. Ungar. Static and dynamic analysis of the internet's susceptibility to faults and attacks. In *IEEE Infocom 2003*, San-Francisco, CA, USA, April 2003.

[PQ00] V.N. Padmanabhan and L. Qiu. The content and access dynamics of a busy web site: findings and implications. *ACM SIGCOM*, Aug 2000.

[PST99] G. Philips, S. Shenker, and H. Tangmunarunkit. Scaling of multicast trees: Comments on the chuang-sirbu scaling law. In *ACM SIGCOMM'99*, Cambridge, Massachusetts, USA, 1999.

[Rip] Routing information service. http://data.ris.ripe.net/.

[RKT98] Dan Rubenstein, Jim Kurose, and Don Towsley. Real-time reliable multicast using proactive forward error correction. In *NOSSDAV '98*, Berlin, Germany, 1998.

[RS98] J. Rosenberg and H. Schulzrinne. Timer reconsideration for enhanced scalability. In *IEEE INFOCOM '98*, San Francisco, CA, USA, March 1998.

[RS00] P. Rodriguez and S. Sibal. Spread: Scalable platform for reliable and efficient automated distribution. *WWW9*, May 2000.

[RTY+00] P. Radoslavov, H. Tangmunarunkit, H. Yu, R. Govindan, S. Shenker, and D. Estrin. On characterizing network topologies and analyzing their impact on protocol design. In *Technical Report 00-731, Dept. of CS, University of Southern California*, February 2000.

[SARK02] L. Subramanian, S. Agarwal, J. Rexford, and R.H. Katz. Characterizing the internet hierarchy from multiple vantage points. In *IEEE Infocom 2002*, New-York, NY, USA, April 2002.

[SCFJ96] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applications, 1 January 1996. RFC 1899, Internet Engineering Task Force.

[TGJ+02] Hongsuda Tangmunarunkit, Ramesh Govindan, Sugih Jamin, Scott Shenker, and Walter Willinger. Network topology generators: Degree based vs. structural. In *Proc. of ACM SIGCOMM 2002*, Pittsburg, PA, USA, August 2002.

[VRC98] L. Vicisano, L. Rizzo, and J. Crowcroft. Tcp-like congestion control for layered multicast data transfer. In *Infocom'98*, March 1998.

[Wat03] Duncan J. Watts. *Six degrees: The science of a connected age*, volume 1. W.W.Norton & Company, first edition, 2003.

[Wax88] Bernard M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6:1617 – 1622, 1988.

[Wes96] D. Wessels. Squid internet object cache. http://www.nlanr.net/squid/. 1996.

[WS98] D.J. Watts and S.H. Strogats. Collective dynamics of small world networks. *Nature 393*, pages 440–442, 1998.

[WVS+99] A. Wolman, G.M. Voelker, N. Sharma, N. Cardwell, A. Karin, and H.M. Levy. On the scale and performance of cooperative web proxy caching. *In Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP99)*, pages 16 – 31, December 1999.

[YJBT01] S.H. Yook, H. Jeong, A.-L. Barabási, and Y. Tu. Weighted evolving networks. *Physical Review Letters*, 86:5835, 2001.