Provisioning IP Backbone Networks Based on Measurements

Konstantina Papagiannaki

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of the

University of London.

Department of Computer Science

University College London

3rd March 2003

Στη μνήμη του πατέρα μου

Γεωργίου Δ. Παπαγιαννάκι

Στη μητέρα μου

Καλλιόπη Γ. Παπαγιαννάκι

To the memory of my father

Georgios D. Papagiannakis

To my mother

Kalliopi G. Papagiannaki

Abstract

The theme of this thesis is the enhancement of current IP backbone provisioning practices in the presence of additional network measurements. Current practices are heavily dependent on the intuition of the human operators. Traffic variability, scalability issues, lack of monitoring information, and complex interactions between inter- and intra-domain routing protocols result in network management techniques that usually rely on trial and error. In contrast with reductionist approaches, we demonstrate the benefits of using different types of monitoring information in the formalisation of different network provisioning tasks, and provide a methodological framework for their analysis.

We use four main sources of network monitoring information: (i) GPS-synchronised packet traces listing every packet traversing a monitored unidirectional link, (ii) BGP routing table dumps, (iii) SNMP information collected since 1999, and (iv) topological information. Combining the above sources of information, and analysing them at the appropriate time scale, we demonstrate the benefits of additional measurements on three specific network provisioning tasks.

First, we measure and analyse delay as experienced by packets while traversing a single router inside the network. We show that packets experience minimal queueing delay and that delay through the network is dominated by the propagation delay. Our results hold when network link utilisation stays moderate. However, links are likely to experience short-lived congestion episodes as a result of link or equipment failures. Our second network provisioning task regards the off-loading of congested links by the re-direction of high-volume flows. We propose a methodology for the identification of those flows traversing a link that contribute significant amounts of traffic consistently over time. Persistent link overload can only be resolved through additional provisioning. Our third task focuses on the prediction of where and when future provisioning will be required in the backbone. We obtain accurate predictions for at least six months in the future.

Contents

List of Figures

List of Tables

Acknowledgements

Pursuing a PhD is an amazing journey. Throughout its duration one meets a lot of interesting people, that help one mature as a scientist and as a person.

The first person I would like to thank is my primary advisor, Prof. Jon Crowcroft, for giving me the opportunity to set off in this amazing journey, and teaching me what it's like to work in a highly stimulating environment. Ever since the beginning, I have always regarded pursuing a PhD as a challenge. Jon supported me through all my decisions and I am really grateful for that.

I would also like to thank my second advisor, Dr. Saleem Bhatti, for his help and support. Saleem always provided me with excellent feedback focusing my attention towards all those small details that make research interesting. I would like to express my gratitude to him for his patience with me. Pursuing a PhD from a different continent imposed a lot of administrative overhead on him, and I am beholden to him for all his help.

I believe that the person I am mostly indebted to in the past three years is my mentor, and friend, Dr. Christophe Diot. Christophe has been my manager at the Sprint Advanced Technology Labs (ATL) since April 2000. He gave me the opportunity to join one of the most lively research groups and familiarise myself with the operational realities of one of the major Tier-1 networks in the Internet, the Sprint IP backbone network. In addition, he turned out to be my most interesting, and usually unpredictable, friend. I got to share his passion for research and value friendship within a working environment. There are not enough words to express the way I feel about his support and encouragement. Without him this work would never have been feasible.

As a member of the IP group at Sprint ATL, I had the pleasure to work with several outstanding researchers and people. I would like to especially thank Dr. Nina Taft for her soothing guidance, her understanding and support. Nina taught me how to properly express my ideas, and how to structure the way I think. My collaboration with her has been an invaluable experience, and I am certain that there is more to come out of it in the future. Thanks also to Dr. Sue Moon for teaching me the importance of thoroughness in research, Dr. Supratik Bhattacharyya for many hours of inspiring meetings, and Dr. Gianluca Iannaccone for always being there to provide me with answers and discuss about everything.

I would also like to express my gratitude to Prof. Patrick Thiran for his immense support and calm encouragement. Patrick was one of the first persons that believed in me and helped me advance my research abilities. Diving into a challenging area such as Computer Science needs a lot of courage. I believe that Patrick greatly helped me to summon the necessary resources to pursue this work.

Last but not least I would like to thank Chuck Fraleigh, Chen-Nee Chuah, Athena Markopoulou, Antonio Nucci, and Richard Gass for our endless discussions, their friendship and support.

Apart from all those people that helped me develop as a scientist I would also like to acknowledge these few people that helped me advance as a person, and made me who I am. My deepest gratitude goes to my mother Kalliopi and my father Georgios. I deeply believe that this thesis is as much their doing as is mine.

Publications

- C. Fraleigh, C. Diot, B. Lyles, S.Moon, P. Owezarski, K. Papagiannaki, and F. Tobagi. Design and Deployment of a Passive Monitoring Infrastructure. In *Passive and Active Measurements Workshop*, Amsterdam, Netherlands, April 2001.
- K.Papagiannaki, P. Thiran, J. Crowcroft, and C. Diot. "Preferential Treatment of Acknowledgment Packets in a Differentiated Services Network". In *Ninth International Workshop on Quality of Service*, Karlsruhe, Germany, June 2001, vol. 2092 of *Lecture Notes in Computer Science*, Springer.
- K. Papagiannaki, S. Moon, C. Fraleigh, P.Thiran, F. Tobagi, C. Diot. Analysis of Measured Single-Hop Delay from an Operational Backbone Network. In *IEEE Infocom*, New York, U.S.A., June 2002.
- K.Papagiannaki, N. Taft, S. Bhattacharyya, P. Thiran, K. Salamatian, C. Diot. A Pragmatic Definition of Elephants in Internet Backbone Traffic. In *ACM Sigcomm Internet Measurement Workshop*, Marseille, France, November 2002.
- K.Papagiannaki, N. Taft, Z. Zhang, C. Diot. Long-Term Forecasting of Internet Backbone Traffic: Observations and Initial Models. In *IEEE Infocom*, San Francisco, U.S.A., April 2003.
- K.Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, C. Diot. Measurement and Analysis of Single-Hop Delay on an IP Backbone Network. To appear in *Journal on Selected Areas in Communications. Special Issue on Internet and WWW Measurement, Mapping, and Modeling.* (3rd quarter of 2003)

Chapter 1

Introduction

1.1 The Internet History

The origins of the Internet go back to the 1960s. The first recorded description of the social interactions that could be enabled through networking is said to have been a series of memos written in August 1962 by J.C.R. Licklider of MIT, discussing his concept of a "Galactic Network" [1]. This network was envisioned as a globally interconnected set of computers through which everyone could quickly access data and programs from any site.

In approximately the same period, July 1961, Leonard Kleinrock of MIT published the first paper on packet switching theory. According to Kleinrock's theory, communication between computers could be achieved with packets instead of circuits. Within the circuit-switching paradigm, deployed in the telephone and telegraph network, users establish a dedicated connection with a fixed amount of bandwidth between the source and the destination for the duration of their communication. This approach is efficient for traffic such as telephone voice calls which transmit data at a constant bit rate and whose connection duration is longer than the amount of time required to establish the connection. However, data traffic is bursty, and therefore allocation of resources for the whole duration of a data transfer may lead to periods of time when network resources are allocated but not used. Moreover, data transfers over the network may be short in duration, and thus lead to a high overhead for the setup and tear-down of the needed circuits. According to the connection-less packet switching paradigm, data transmitted between two network entities are broken down into small units, which are called *packets*. Each packet contains the destination and source address of the two entities exchanging traffic, as well as other control information. In addition, each packet is forwarded independently across the network until it reaches its destination, without requiring reservation of resources across its path. Experiments carried out in 1965 provided confirmation that time-sharing computers could work well together, running programs and retrieving data as necessary on remote machines, but that the circuit-switched telephone system was totally inadequate for this task [2]. Thus, lending the packet switching paradigm as an appropriate technology for inter-computer communication.

In the late 1966 the first concept of what was to become the first computer network was developed [3]. Multiple researchers worked on the optimisation of the network topology, and the development of the first host-to-host protocol, called the Network Control Protocol (NCP) [2]. The first large, successful demonstration of the first computer communications network, called the Advanced Research Projects Agency Network (ARPANET), took place in 1972. Its first application was e-mail, also developed in 1972, soon to be followed by other applications aimed at facilitating the coordination among researchers.

1.2 From ARPANET to the INTERNET

The original ARPANET grew into the Internet based on the idea that there would be multiple independent networks of rather arbitrary design. The key underlying technical concept was "open-architecture networking", introduced by Kahn in late 1972. The "open-architecture networking" concept had four critical ground rules [2].

- Each distinct network had to stand on its own, and no internal changes would be required of any such network before being connected to the Internet.
- Communications would be on a *best-effort* basis. If a packet didn't make it to the final destination, it would quickly be retransmitted from the source.
- Black boxes (later called gateways and routers) would be used to connect the networks. No information would be retained by the gateways about individual flows or packets passing though them, keeping them simple and avoiding complicated adaptation and recovery from various failure modes.
- There would be no global control at the operations level.

These principles led to the design of the first communication protocol on the Internet, named Transmission Control Protocol/Internet Protocol (TCP/IP). TCP/IP embedded all the necessary functionality required for "open-architecture networking". The initial protocol was later reorganised into two distinct protocols: (i) the Internet Protocol (IP) which provides only for addressing and forwarding of individual packets, and (ii) the separate Transmission Control Protocol (TCP) which provides service features such as flow control and recovery from lost packets. For applications that did not need reliability, as provided by TCP, an alternative protocol was devised to provide direct access to the basic IP service. This latter protocol was named User Datagram Protocol (UDP). By the end of 1985, the Internet was established as a technology supporting a broad community of researchers and developers and started to be used by other communities for daily computer communications.

1.3 The Emergence of Internet Service Providers

Until the mid-1970s computer networks were springing up wherever funding could be found for this purpose. In 1984-1985, the British JANET and the U.S. NSFNET programs explicitly announced their intent to serve the entire higher education community regardless of discipline. In that same year, the NSFNET (National Science Foundation Network) recognised the need for a wide-area networking infrastructure to support the general academic and research community. In addition, it became evident that it was essential that a strategy was developed to establish such an infrastructure to be ultimately independent of direct federal funding.

The National Science Foundation (NSF) encouraged regional networks of the NSFNET to seek commercial, non-academic customers. Federal agencies were to share the cost of common infrastructure, such as trans-oceanic circuits. NSF policies dictated that commercial users of the NSFNET were prohibited from using the "Backbone" unless it was related with research or educational efforts. Such a policy led to the emergence and growth of "private" competitive, long-haul networks, like PSI, UUNET, ANS CO+RE, and later others. At this point, the Internet Service Provider (ISP) emerged as a distinct service operation.

The NSFNET Backbone was officially defunded in April 1995. The funds recovered were redistributed to regional networks to purchase national-scale Internet connectivity from the numerous, by then, private, long-haul networks. The Backbone no longer consisted of routers developed within the research community, but of commercial equipment. In less than 10 years the Backbone had grown from six nodes with 56 Kbps links to 21 nodes with multiple 45 Mbps links. The Internet itself had grown to more than 50,000 networks on all seven continents.

On October 24, 1995, the Federal Networking Council (FNC) unanimously passed a resolution defining the term Internet. According to the FNC the term "Internet" is defined as follows[1].

> "Internet" refers to the global information system that – (i) is logically linked together by a globally unique address space based on the Internet Protocol (IP) or its subsequent extensions/follow-ons; (ii) is able to support communications using the Transmission Control Protocol/Internet Protocol (TCP/IP) suite or its subsequent extensions/follow-ons, and/or other IP-compatible protocols; and (iii)

[1]*http : //www.itrd.gov/fnc/Internet_res.html*

> provides, uses or makes accessible, either publicly or privately, high level services layered on the communications and related infrastructure described herein.

1.4 Today's Internet

The Internet has changed much since it came into existence. It was conceived in the era of time-sharing, but has survived into the era of personal computers, client-server and peer-to-peer computing. It was designed before Local Area Networks (LANs) existed, but has accommodated network technologies, such as LAN, ATM, frame switched, and wireless services. It was envisioned as supporting a range of functions from file sharing and remote login to resource sharing and collaboration, and has spawned electronic mail, the World Wide Web, and is now asked to accommodate voice and video traffic. But most importantly, it started as the creation of a small band of dedicated researchers, and has grown to be a commercial success with billions of dollars of annual investment.

Today's Internet consists of thousands of networks. What used to be the "Backbone" in the 1980s is now the interconnection of multiple backbone networks, belonging to large telecommunications providers. Providing Internet services is a new, challenging market. However, basic access has become a market commodity without obvious differentiation, and, as such, is the subject of prolonged price pressure. The Internet market is one of the most aggressively open markets, with few onerous regulatory constraints on the operation of the enterprises that service this market. The environment is one in which access tariffs are driven down to reflect the commodity price of carriage. In fact, in March 1998 the *Financial Times* reported that of the 25 most valuable Internet companies, with a combined value of $37 billion, 20 were still operating at a deficit.

> "The strategic task of the ISP today is to attempt to minimise the investment risk, through the application of business and technical expertise to the operation of the service enterprise, while attempting to maximise the financial outcomes from the investment." [4]

1.5 Challenges

In such a fiercely competitive market, an Internet Service Provider has several challenges to address. These challenges are not only related to business plans and financial investments, but also technological advances. The Internet no longer accommodates a small scientific community, but supports business critical applications, and has come to be relied on by large corporations for their every day activities. Reliability and predictable performance are key differentiators between Internet Service Providers.

However, the performance experienced by a user does not only depend on the network he/she connects to, but also on the networks that need to be traversed until the destination is reached. For that reason, Internet Service Providers expand their networks to increase connectivity to diverse destinations. In addition, they provision their networks in such ways that there is always available capacity to accommodate customer demand.

Current customer-provider contracts contain three main metrics describing the performance of service offered to the user, namely delay through the network, loss, and port availability. These metrics are part of what is often referred to as a Service Level Agreement (SLA). Increased competition has led to SLAs guaranteeing delays through the network that compare to transmission at the speed of light, minimal losses and high availability. Building a network that is capable of conforming to all the requirements above corresponds to an immense investment that needs to be accompanied by justifiable profits.

Currently, plain network connectivity is either charged on a per circuit basis or on a subscription basis. Tight market conditions and increased competition drive down the access tariffs leading ISPs to operate at a marginal profit. As a consequence, ISPs look to offer value-added services on top of their existing infrastructure that could be charged at different rates, and positively contribute to their profit. Integrating the telephone network with the data network presents itself as an attractive proposition that could limit the cost-profit gap currently faced by data providers. However, in order for Internet Service Providers to offer services like voice over IP, or video distribution, they need to evaluate the performance of their network, and ascertain its capabilities with respect to the requirements posed by such applications.

In parallel to such efforts, network operators face an ever-increasing traffic demand, and severe scalability issues with their current infrastructure. IP backbone networks consist of hundreds of nodes and thousands of links. Complicated interactions between inter- and intra-domain routing protocols make IP backbone network management a very challenging task. Abiding to strict SLAs within a complex backbone environment calls for better understanding of the carried traffic and the specifics of network behaviour.

1.6 Provisioning IP networks

The above requirements are currently addressed by backbone network operators through "over-provisioning". The network features excess capacity, allowing for transient bursts of traffic, and short-lived network element or link failures. Whenever parts of the network reach utilisation levels that could be described as congested, the network operators increase the available capacity lowering the experienced delays, and therefore improving the performance offered to the end user.

Such an approach leads to networks that could be described as being continuously "under construction". New links are added or upgraded weekly increasing the amount of available resources in the network. However, such an approach ends up being rather costly, and given the difficult market conditions, is very hard to justify. As a result, network providers need to find ways in which they could make optimal use of their network's resources and delay future investments as long as possible.

The task of optimally designing IP backbone networks imposes challenges for multiple reasons. Firstly, an IP backbone network can easily consist of hundreds of network elements and thousands of operational links. These links may reach 10 Gbps of capacity each[2], while spanning multiple continents. Therefore, there is an evident scalability issue. Secondly, the amount of traffic flowing over an IP backbone network does not only depend on the network itself but also on the traffic injected by its interconnected networks. Thirdly, stringent availability requirements dictate the existence of spare capacity on the network, capable of carrying the traffic affected by failed links. Lastly, at all times the network should be capable of meeting the Service Level Agreements drawn with customers.

1.7 The need for additional network measurements

The only piece of information currently available to the network operator for management and planning is aggregate packet or byte counts for the traffic carried over a network's links, and health status reports from its individual nodes. Most of the time, troubleshooting the network can be a daunting task, given the limited amount of information available and the fact that IP network routing protocols are designed to automatically adjust to current conditions.

Improving the network design and the current best practices in network management requires the presence of additional measurements from the network. Traditional telecommunications network design theory assumes the existence of what is called a "traffic matrix", denoting the amount of traffic flowing between any source and any destination in the network. An IP network was not designed to provide such detailed information, and given the scale of current backbone networks, even estimating such information based on available measurements proves to be challenging.

In this thesis we look into provisioning practices for large scale IP backbone networks, consisting of hundreds of nodes and thousands of links. We demonstrate how additional net-

[2]OC-192 (10 Gbps) links are the highest capacity links available in the Internet at the time of writing.

work measurements can enhance the current network provisioning practices. **We show that network operations can be greatly improved based on network measurements, but those measurements need to be diverse and must be analysed at different granularities of time and traffic aggregation.** Depending on the task at hand, most of the times the network operator needs to collect specific information that will allow him to seek a solution.

In this thesis, we will mainly refer to results obtained from measurements collected in one of the major IP backbone networks in the Internet, the Sprint IP backbone network. We provide results derived from the analysis of four main sources of measurement information:

- **Packet traces** collected from selected links inside the Sprint IP backbone network. These packet traces list the first 44 bytes of *every* packet seen on a unidirectional monitored link, along with GPS timestamps for their time of appearance on the link.
- **Link utilisation** information collected throughout the network using the Simple Network Management Protocol (SNMP). This information contains a 5-minute average value for the utilisation of every operational link in the backbone, and has been collected continuously every 5 minutes since the end of 1999.
- **Border Gateway Protocol routing tables** collected from core routers inside three Points of Presence of the Sprint IP backbone network. These BGP tables contain all the globally addressable network prefixes in the Internet at the time of the collection.
- **Router configuration files** listing all the active links for every router in the network, along with their capacity, their IP address, and the name of the router and interface terminating the link at the other end. Router naming conventions allow us to also identify the Point of Presence where each router is located.

1.8 Thesis Outline

This thesis is organised as follows. In Chapter 2 we provide an overview of how an IP backbone network is structured. We describe the way traffic flow is dictated throughout the network based on the interactions between the inter- and intra-domain routing protocols. We then give a brief introduction on the current best practices in the network management field, and identify issues that could be resolved with additional network measurements.

In Chapter 3 we describe the state of the art in the field of network measurements. We report on active projects that aim at evaluating the performance of operational networks based on active measurements. We describe what active measurements are and show that even though they are capable of identifying possible issues with the current network design, they usually do not offer enough information for their resolution. Resolving such issues, and identifying alternative network designs capable of overcoming such shortcomings require further measurements. In this area, there are far less active projects, and they usually combine active network measurements with passive network measurements. Such projects deploy dedicated monitoring equipment at specific points inside the network. They then attempt to understand the behaviour of the Internet traffic and the network dynamics based on the collected observations. In this thesis we focus on the analysis of passive measurements collected from multiple points in the Sprint IP backbone network in conjunction with router configuration and topological information.

In Chapter 4, we analyse packet traces collected at several points in the network and router configuration information to measure the delay experienced by packets through a single node in an operational IP backbone. We present our single-hop delay measurements and propose a methodology to identify its contributing factors. We show that in addition to packets processing, transmission at the output link, and queueing delay at the output queue, packets may face very large delays that cannot be accounted for within the context of a pure FIFO output queue. These large delay values are thought to be related to router idiosyncrasies, and only affect less than 1% of the packets going through a router. Our main finding is that there is very little queueing taking place in the monitored backbone network. This is consistent with the network design principles that dictate moderate link utilisations across the network. We conclude that in today's moderately utilised backbone networks queueing delay at backbone routers adds insignificantly to the overall edge-to-edge delay.

However, our findings are only applicable to cases when link utilisation stays moderate; in our measurements link utilisation never exceeds 70%, even when computed over 10 ms time intervals. Nonetheless, there will always be cases when unexpected link and equipment failures may lead to transient link overload. In such cases, when particular links go down, other links in the network will be selected by routing protocols to carry the affected traffic, and thus may reach prohibitive utilisation levels. As a consequence, delays through these links will increase and loss may occur. In Chapter 5, we propose a methodology for the classification of flows on a backbone link, so that in cases of transient overload a network operator can offload the highly-utilised links by re-routing high-volume flows toward other parts of the network. For this analysis, we use packet traces collected from multiple points in the network, and BGP routing tables. We analyse traffic flowing on selected backbone links and show that when aggregated on a destination network-prefix level it exhibits what is usually referred to as the "elephants and mice" phenomenon; the majority of the traffic volume is due to a small number of the flows. We present a methodology to identify the network-prefix flows on a link that persistently carry the majority of the traffic over time; flows with small elephant duration are inappropriate for traffic engineering applications. Once the high-volume network-prefix flows have been identified, and in cases of high load, the network operators can relieve overloaded links by re-directing specific "elephant" flows toward less congested parts of the network. Such an approach provides a controlled way of avoiding long-lasting congestion phenomena in the network.

An approach like the one proposed in Chapter 5 can only be applied if the network experiences transient phases of congestion. Consistent overload will be much harder to overcome with simple techniques like the one we present. In such cases, parts of the network will need to be upgraded to accommodate the increased demand. In Chapter 6 we present a methodology to identify *when* and *where* future link upgrades will need to take place in the core of a backbone network. For our analysis we use three years worth of SNMP data collected from *every* operational link in the Sprint IP backbone network. We correlate this information with topological information and identify the amount of traffic flowing between any two adjacent PoPs in the network. Using the Wavelet Multiresolution Analysis, we extract the long and shorter-term trends in the evolution of inter-PoP aggregate traffic. We show that for network provisioning purposes one needs only to account for the overall long term trend and the fluctuations of the traffic at the 12 hour time scale. We model the inter-PoP aggregate demand using the two identified components alone. We then predict the future aggregate demand between any two adjacent PoPs using linear time series models, namely the AutoRegressive Integrated Moving Average (ARIMA) models. We demonstrate that our technique is capable of providing accurate forecasts, within 15% of the actual values, for at least six months in the future with a minimal computational overhead.

Lastly, in Chapter 7, we summarise our contributions, develop on the limitations of our work, and discuss possible directions for future research.

Chapter 2

Background

2.1 The Internet Routing Hierarchy

The Internet is a collection of networks exchanging traffic using the Internet Protocol (IP) [5]. Each one of those networks spans a subset of the IP address space, and is represented by a network and mask pair [6]. This network and mask pair is called a *network prefix*, and is usually denoted as <prefix/length>, where "prefix" denotes the IP address of the network and "length" indicates the number of leftmost contiguous bits corresponding to the network mask to be associated with that network address.

Large corporations or Internet Service Providers (ISPs) are likely to consist of more than one network prefixes, thus forming an Autonomous System. More formally, an Autonomous System (AS) is defined as a set of routers that has a single routing policy, and run under a single technical administration. Hence, large IP networks are likely to be viewed in the Internet as a single entity, denoted by a single identifier called an AS number.

The global Internet is formed by the inter-connection of different autonomous systems, exchanging reachability information using inter-domain routing protocols. The Internet routing hierarchy is built on the business relationships between different ASs, where each level of interconnection is usually described as a *Tier*. According to this hierarchy entities at higher levels $(n - 1)$ provide services to their connected entities at the layer directly below them (n) [7]. What constitutes a *Tier-1* (highest level) network is rather controversial, but the most popular definition is that *Tier-1* ISPs are those networks that have access to the global Internet Routing Table and do not purchase network capacity from other providers [8]. The second tier generally has a smaller national presence than a Tier-1 ISP and may lease part or all of its network from a Tier-1. Lastly, a Tier-3 ISP is typically a regional provider with no national backbone.

In this thesis, we look into provisioning practices for Tier-1 IP networks, also called "backbone" networks. All the measurements presented throughout this thesis are collected inside the

Sprint IP backbone network, one of the few Tier-1 networks in today's Internet.

2.2 Backbone Network Architecture

The topology of an IP backbone network typically consists of a set of nodes known as Points-of-Presence (PoPs) connected through multiple high capacity links. The Sprint IP backbone consists of approximately 40 PoPs worldwide. Out of those 40 PoPs, approximately 25 are located in the U.S.A. Figure 2.1 presents the Sprint IP backbone topology for the continental U.S.[1].

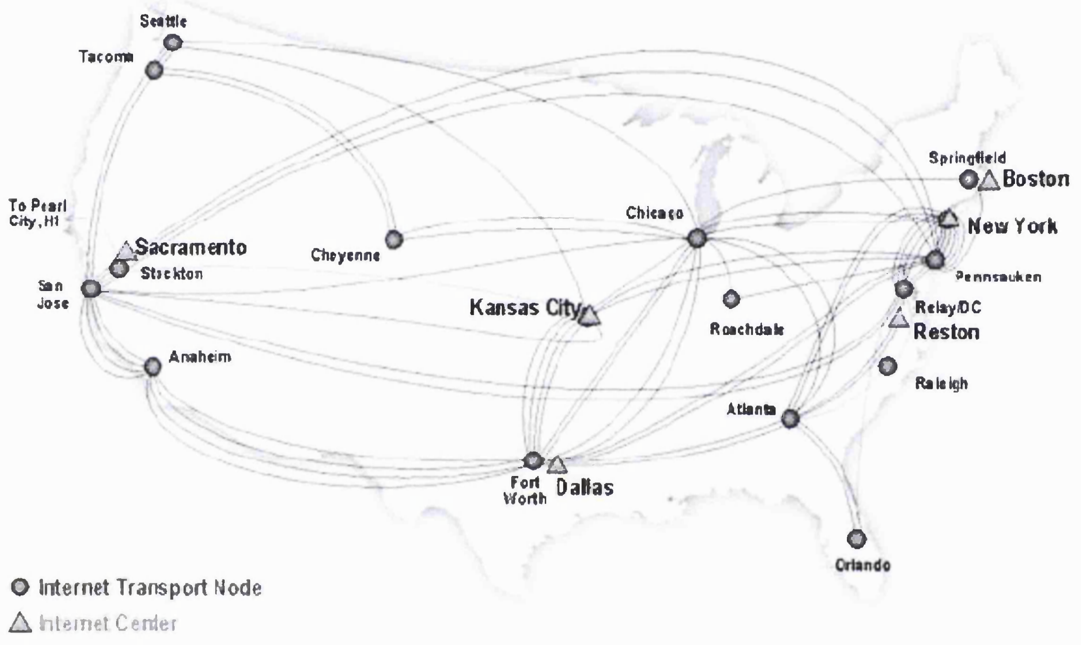


Figure 2.1: The Sprint IP backbone network topology (3rd quarter 2001).

Each Sprint PoP is a collection of routers following a two-level hierarchy, featuring (i) *access routers*, and (ii) *backbone routers* (Figure 2.2). Such an architecture is typical among large Tier-1 ISP providers [9]. Access routers are normally lower-end routers with high port density, where customers get attached to the network. These routers aggregate the customer traffic and forward it toward the PoP's backbone routers. The backbone routers receive the aggregate customer traffic and forward it to other PoPs or the appropriate access routers inside the same PoP (in case the destination networks can be reached through the same PoP). Public and private peering points, where one ISP exchanges traffic with other ISPs (usually of the same Tier), are often accommodated by selected backbone routers inside a PoP.

[1] *http : //www.sprintesolutions.com/resources/maps/bb_map2_popup.html*. Other major Tier-1 ISP providers follow similar architectures, as seen in *http : //navigators.com/isp.html*.

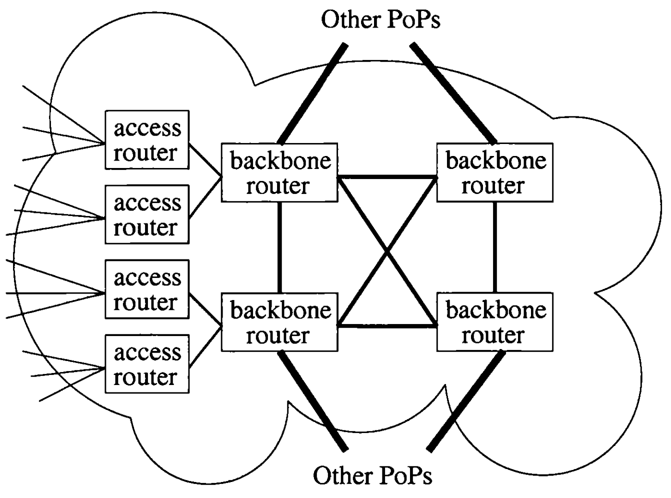


Figure 2.2: Typical configuration for a Point of Presence.

The capacity of links interconnecting routers inside the PoP depend on their level in the hierarchy, i.e. the level of traffic aggregation they correspond to. For instance, customer links are usually 45 Mbps (T3 or DS3) or greater. The links connecting access routers to backbone routers are an order of magnitude larger, reaching 622 Mbps (OC-12). The backbone routers inside a PoP are densely meshed (not necessarily fully meshed), and interconnected through higher speed links, i.e. OC-12 to OC-48 (2.5 Gbps). The inter-PoP links are long-haul optical fibres with bandwidth of 2.5 Gbps (OC-48) or 10 Gbps (OC-192).

IP backbones are engineered for high availability, and resilience to multiple link and router failures, while meeting the contracted Service Level Agreements (SLAs). Traffic is guaranteed to transit the network experiencing bounded edge-to-edge delays. In addition, even in cases when certain IP paths through the network become unavailable, traffic is guaranteed to be delivered at its destination through alternate paths. Consequently, each PoP is designed to connect to multiple other PoPs through multiple high-capacity links (Figure 2.1). Dense connectivity between PoPs guarantees that traffic will go through the network transiting a bounded number of PoPs. In the presence of link failures, there should exist other links available to take on the affected load without introducing additional delay or loss. Given that the inter-PoP links are covering large distances, thus accounting for the largest part in the edge-to-edge delay, rerouting traffic across PoPs should be employed only as a last resort. For that reason, adjacent PoPs are interconnected through multiple links. These links, when feasible, follow at least two physically disjoint paths, thus offering resilience against fibre cuts which may affect more than one physical links sharing the same fibre path.

2.3 Intra-domain Routing

IP paths taken by packets through the network are determined by the intra-domain routing protocol deployed in the network. The most prevalent intra-domain routing protocols in today's Internet are (i) Open Shortest Path First (OSPF) [10], and (ii) Intermediate System-Intermediate System (IS-IS) [11]. Their ability to quickly detect topological changes and adjust traffic flow accordingly makes them the most popular Interior Gateway Protocols (IGPs) currently operational in the Internet [12].

Both OSPF and IS-IS are link-state routing protocols, utilising a replicated distributed database model. They work on the basis that routers exchange information elements, called *link states*, which carry information about links and nodes in the routing domain.

More specifically, each OSPF router internal to an Autonomous System generates its own view of the network consisting of its neighbours and possible *costs* involved in their interconnection. This view of the topology obtained by a single router inside the domain is reliably distributed throughout the network, across all routers, via a flooding mechanism until all nodes in the network have obtained a copy. When the distribution is complete, every router in the network has a copy of the local view of the network topology from every other router in the network. This is usually referred to as a *link-state database*. Each router then autonomously constructs the complete topology for the entire domain. The result of this process is an identical copy of the domain's topology on each router in the network.

Once a router knows the current topology of the network, along with the *costs* involved in using specific links across the network, it applies the Shortest Path First (SPF) algorithm, and calculates a tree of shortest paths to each destination, placing itself as the root. At that point, we say that the routing protocol has *converged*. The *cost* associated to each link in the network is also known as the OSPF/IS-IS weight and is configured by the network operator. Decreasing the weight of a link in the network is likely to attract more traffic towards it. Consequently, link weights express administrative preference toward specific paths throughout the network.

Depending on the link weights configured across the network, the SPF algorithm may compute multiple equal-cost paths toward specific destination networks. In case of equal-cost IP paths, traffic is split among the different available IP paths, also described as *load-balanced*. Such an approach leads to comparable utilisation levels across equally weighted links in equal-cost IP paths, and therefore a better spread of the carried traffic across the network. As presented in the previous section, backbone networks are likely to feature multiple links between two adjacent PoPs. Those links are usually configured with equal link weights. As a consequence, when all links are operational, traffic is spread across all links, and when a link fails, the affected traffic can revert to the remaining links without impacting the performance observed.

2.4 Inter-domain Routing

Intra-domain routing protocols dictate the way traffic flows throughout a single AS. Traffic flow across the Internet and between different autonomous systems is determined by the inter-domain routing protocols.

Border Gateway Protocol (BGP) is the dominant inter-domain routing protocol used to exchange reachability information between ASs in today's Internet [13]. It is described as a *path-vector* protocol. In essence, it is similar to a distance vector protocol [12], but instead of being told a distance to a destination, a router is told an AS path to the destination, i.e. a sequence of AS numbers that correspond to the Autonomous Systems that need to be traversed until the destination is reached.

For inter-domain routing each BGP router sets up TCP connections (also called sessions) with other BGP routers (usually called its neighbours or peers) in neighbouring ASs, or within the same AS. Reachability information for networks accessible through other ASs is conveyed via E-BGP (Exterior BGP) connections between BGP speakers in foreign domains. This information is then re-distributed within the AS via I-BGP (Interior BGP) so that all routers in a given domain know how to access specific networks outside the AS.

Once two E-BGP peers establish a connection, they exchange information about the networks that can be accessed through them, and the AS paths used. This information is then used to construct a graph of AS connectivity from which routing loops can be pruned and some policy decisions at the AS level can be enforced. Each BGP peer calculates the shortest AS path toward each destination network and advertises it to its own neighbours. The resulting routing table featuring entries toward each network prefix, and the AS path associated to it, is called the *BGP routing table.*

Apart from reachability information between ASs BGP is also used to convey policy directives. Local policy can be enforced by configuring E-BGP routers not to announce specific destination prefixes, known to them, or to announce them with a much longer, artificially set, AS path. For instance, if a network known to a particular BGP router does not correspond to a customer or a network receiving transit services from the router's AS, then the BGP router can be configured so as not to announce this network to its peers. Announcing a network using a longer AS path is the current best practice to discourage neighbouring networks from routing traffic toward a particular network through a given AS.

Further route filtering and routing information control in general can be achieved using the

BGP Path attributes. The BGP path attributes are a set of parameters that describe the characteristics of a prefix. The *BGP decision process* couples these attributes with the prefix they describe, compares all the paths available to reach a given destination, and then selects the best routes to be used to reach the destination. Apart from the AS_PATH attribute, which has been described already, other attributes conveying policy information are the MULTI_EXIT_DISC, and LOCAL_PREF [14].

The BGP Multiexit Discriminator (MULTI_EXIT_DISC or MED) attribute is an optional nontransitive attribute (it is not propagated through ASs). It informs external neighbours about the preferred path *into* an AS that features multiple entry points. The MED is also known as the external metric of a route. A lower MED value is preferred over a higher MED value.

The Local Preference (LOCAL_PREF) attribute is a well-known discretionary attribute. The local preference attribute conveys the degree of preference given to a route compared to other routes for the same destination. A higher local preference value indicates that the route is preferred. Local preference, as indicated by the name, is local to the autonomous system and is exchanged between I-BGP peers only. An AS connected via BGP to other ASs will receive routing updates about the same destination from different ASs. Local preference is usually used to set the exit point of an AS to reach a certain destination. Given that this attribute is communicated within all BGP routers inside the AS, all BGP routers will have a common view of how to exit the AS.

Controlling the entry and exit points on an AS for specific prefixes allows operators to influence the traffic flow through their network. The above path attributes are the "knobs" the operator has to tune in order to achieve the desirable result. In more detail, the BGP decision process is as follows. A router first checks for the path with the largest local preference value. If local preferences are the same, then the router selects the path with the shortest AS_PATH. If all advertisements have equal length AS_PATH attributes, then the path selected is the one with the lowest MED. Otherwise, the path is selected so that it corresponds to the path that can be reached via the closest IGP neighbour. If none of the above conditions leads to a best path toward the destination prefix, the tiebreaker is the BGP ROUTER_ID. The BGP ROUTER_ID is usually set to the loopback address of the router, if configured, or the highest IP address on the router. The tiebreaker process selects the best path to the destination as that path that was advertised by the router with the lowest ROUTER_ID [14].

2.5 Network Management

Managing a large IP backbone network is a complex task handled by a group of human operators, usually referred to as a *Network Operation Centre* (NOC). Network management involves observing the state of the network and reacting to events that may lead to degradation in the performance experienced by end users. Such events could involve equipment or link failures, denial of service (DoS) attacks, or even sudden surges of traffic that increase link utilisation beyond the acceptable levels. The ability to detect, and properly diagnose problems in a timely fashion depends on the information available from the underlying network.

The designers of IP networks have traditionally attached less importance to network monitoring and resource accounting than to issues such as distributed management, robustness to failure and support for diverse services and protocols [15]. In other words, IP network elements have not been designed to retain detailed information about the traffic flowing through them. Historically, the only information collected by network elements has been status information (e.g. the operational status of links on a router) and counts of aggregate activity (e.g. total number of packets or bytes) on a per link basis. This information can be retrieved from network elements using the Simple Network Management Protocol.

2.5.1 Simple Network Management Protocol (SNMP)

The "Simple Network Management Protocol" is an evolution of the "Simple Gateway Monitoring Protocol" [16]. The first version of SNMP was defined in 1988, was updated in 1989, and became an Internet standard in 1990 [17]. Most network elements in today's Internet support the second version of SNMP, which was defined by the IETF in 1996 [18]. A third version was proposed in 1999 [19], primarily adding security and remote configuration capabilities.

SNMP is a very simple protocol, based on a client-server model and implemented over UDP. It has been designed to allow implementation in low-cost systems with little memory and computing resources. The simplicity of SNMP made it possible to require its support by all Internet systems, which in turn greatly facilitates the management of the Internet.

SNMP enables network managers to remotely monitor and control the evolution of network entities, e.g. routers or transmission equipment. The state of each entity is represented by a number of variables and tables. The structure of these data is defined by a "management information base" (MIB). Monitoring is performed by reading these variables through SNMP commands. Control is performed by "setting" some essential parameters to a new value. There are MIBs defined for a very large number of "managed objects", notably the generic MIB which describes an IP router [20], also known as MIB-II, and specific MIBs which describe the variables and tables for each routing protocol [21, 22].

SNMP read and write operations are triggered by the network operator. However, SNMP also offers a small number of unsolicited messages, better known as *SNMP traps* that may be generated by the network elements themselves, for instance when communication links fail, or become active [17]. The destination of those messages is configured by the network operators. The collection station, receiving those traps, and recording the health status information is usually called a Network Management Station (NMS).

2.5.2 Remote Monitoring MIB (RMON)

RMON is another standardised SNMP MIB, whose goal is to facilitate remote monitoring of Local Area Networks (LANs) [23]. In the RMON context, a single agent is capable of monitoring a complete shared LAN. The RMON agent is endowed with local intelligence and memory and can compute higher-level statistics and buffer these statistics in case of outage. Alarm conditions are defined as well as actions that should be taken in response, in the spirit of the SNMP traps. Lastly, filtering conditions and the corresponding actions are defined, so that the content of the filtered packets can be captured and buffered.

RMON offers great flexibility in combining the above primitives into sophisticated agent monitoring functions. However, this flexibility makes a full RMON agent implementation costly. Thus, RMON has been only implemented (at least partially) for LAN router interfaces, which are relatively low-speed. Implementations for high-speed backbone interfaces have proved to be infeasible or prohibitively expensive. Instead, router vendors have opted to develop more limited monitoring capabilities for high-speed interfaces [24].

2.6 Challenges

High-availability, adherence to strict SLAs and limited monitoring information from the network itself makes network management a challenging task. In this section, we expand on these challenges of managing a large-scale IP network.

2.6.1 Network Management

Everyday network operations include monitoring of the network's state and taking corrective actions in case of congestion. SNMP provides a simple way of monitoring IP networks by periodic polling of all network entities and keeping track of their health status. However, SNMP information is usually limited to the aggregate number of bytes sent or received by a link in the network, usually reported at 5-minute intervals. Therefore, short bursts or dips in the utilisation of a link are very difficult to troubleshoot. SNMP cannot provide information about the reasons behind such a behaviour. In this case, the network operator needs information about the complete traffic flow throughout the network and possible routing protocol interactions that may have resulted in the current status of the link.

As a consequence, problem diagnosis on the basis of aggregate SNMP information is usually based on the intuition that network operators gain through years of experience managing IP networks. Such a task is made even harder given that IP protocols have been designed to automatically respond to congestion (e.g. TCP) and work around failures (e.g. routing protocols).

Lack of information which could lead to a definite answer as to the cause of certain events in a network's lifetime leads to current network engineering best practices being mostly based on trial and error. It is not uncommon for network operators to notice congestion on the network, and attempt to divert traffic to less congested areas by increasing the intra-domain link weights so as to repel traffic from the affected area, as proposed in [25]. Observation of the resulting network state supports the current configuration change or calls for further action.

2.6.2 Network Design

In the same spirit, even network planning for large-scale IP networks is a process that is heavily based on intuition. Traditional telecommunications network design is a well researched area offering several techniques for network planning [26, 27, 28]. However, in the field of telecommunications networks there is an inherent assumption about the ability to measure traffic flowing between any source and destination accommodated by the network. The representation of the traffic demands between *any source* and *any destination* in the network is usually referred to as a *traffic matrix*. Knowledge of the traffic matrix and the routing policies in effect inside a network can allow for optimal planning and allocation of a network's resources.

Analytical network design techniques are usually specific to circuit-switched networks, where traffic between two nodes follows a well-defined path, and where end nodes keep traffic statistics allowing for the computation of the traffic matrix [28]. In a packet switched network, traffic between any two nodes may follow multiple paths, and each packet in the flow is routed independently. Consequently, network design techniques specific to packet-switched networks usually rely on network simulations and require accurate modelling and forecasting of the incoming traffic, as well as knowledge of the network traffic matrix. As described in Chapter 1, network elements in an IP network are not designed to retain any detailed flow information. As a consequence, the derivation of a traffic matrix for an IP network is a very hard task. Furthermore, depending on the network application the operator needs to address, a traffic matrix may need to take several forms.

A taxonomy of different traffic matrices useful for traffic engineering and planning tasks is presented in [29]. According to [29], traffic matrices can be categorised based on a two level taxonomy. The first level captures the spatial representation of network traffic in the traffic matrix, denoting whether the demand is measured point-to-point, point-to-multipoint or should also include information about the path taken through the network, described as "path point-to-point traffic matrix". The second level describes the aggregation level for the sources and destinations engaging in traffic exchanges, which includes the granularity of aggregation of a PoP, a router, a link or even a network prefix. Depending on the level of aggregation required by the application, the traffic matrix may contain billions of elements, making its derivation computationally prohibitive. Moreover, even estimation techniques aimed at deriving the traffic matrix based on statistical assumptions face challenges when it comes to large-scale networks with hundreds of nodes [30].

Inability to compute the traffic matrix for large-scale IP networks hinders the application of traditional network design techniques on large-scale IP environments, like that of a Tier-1 IP backbone network. This is the main reason why current best practices in large-scale IP network management and planning are largely dependent on human intuition. However, there are important factors that drive the need for better network engineering tools.

2.6.3 Service Level Agreements

Customers increasingly request more stringent assurances with respect to the performance, reliability, and security of their connectivity. These assurances are normally given in the form of a Service Level Agreement (SLA), where an ISP contracts to the performance that a customer should expect from the service provided. Under the fierce competition between ISPs in the past years, those SLAs have been modified, and nowadays, they essentially guarantee end-to-end delays, that compare to transmission of packets across the network at the speed of the light, minimal disruption of customer connectivity, and losses that are very close to zero.

However, the contracted metrics are normally monthly averages of the network-wide performance. In other words, ISPs measure delays, and losses across the entire network, and then they average all the measurements collected for the entire network across an entire month. As a consequence, poor performance on a small number of paths across the network can be concealed by the good performance of other, possibly shorter, paths. Moreover, averaging performance across hours and days leads to metrics that cannot reveal the performance of the network when it is utilised the most, i.e. during the peak working hours. Consequently, inclusion of measurements taken during the weekends and at night skew the performance actually observed by the user contending for resources during the working hours.

2.6.4 New Services over Data Networks

Transmission of voice and video over the Internet demands the definition of new performance metrics, that can characterise the performance a user should expect from such a service. Studies carried out on the Internet showed that voice over IP could well be accommodated by certain backbone networks [31, 32]. However, such a statement does not hold for the global Internet [32]. Moreover, in case of link or router failures, VoIP services may be disrupted beyond acceptable limits [31].

These two findings impose new network design requirements. Routing protocol convergence times and resulting service disruption arise as issues that need to be resolved before such services can be offered on top of a generic data network. In order to be able to address those issues there is a clear need for additional measurements concerning routing protocol behaviour within IP networks.

2.6.5 Traffic Variability

Internet traffic is complex. IP traffic studies demonstrate that Internet traffic volumes fluctuate widely across time [33]. Moreover, equipment or link failures are unexpected events that may lead to sudden rises in traffic and congestion on certain parts of the network. Given the stringent SLAs in place, a network operator has to provision his network in a way that such bursts can be accommodated without any noticeable impact on the performance received by the user.

Current practices call for moderate link utilisations across the network. This technique is usually described as "overprovisioning". In essence, the network provider always makes sure that there is excess capacity available in the network, and that links do not exceed certain provider specific utilisation levels. Only under those conditions, can traffic affected by a link failure be re-routed to other parts of the network that can carry it without affecting the performance received by users. Moreover, only in such a case can a link sustain transient traffic rises, that may increase link utilisation up to 100% for brief periods of time.

Overprovisioning is successful at achieving the above goals, but is a very costly solution. It requires large investments in network infrastructure, that are well justified only in the few occasions when unexpected events occur. However, without any understanding behind the reasons driving bursty behaviour and approaches toward their avoidance, overprovisioning seems the only solution that can offer bounded performance. Without any understanding behind link failures, their frequency, and their effect as seen by routing protocols, network operators need to continue provisioning their networks beyond the current needs. Evidently, network measurements could shed some light on the issues above, and identify whether there are different provisioning approaches, that with a smaller cost could lead to similar results.

2.6.6 Network Growth

Another challenge that needs to be addressed by network operators has to do with the fact that individual backbone networks are growing rapidly in size, speed, and scope. The industry is consolidating many disparate networks into larger integrated networks. As a result, network management functions that could once be handled by a small group of people, based on intuition and experimentation, must now be supported by effective traffic engineering tools that unite configuration and usage information from a variety of sources.

An IP backbone network consists of hundreds of network elements and thousands of active links, some of which carry Gigabits of traffic every second. Interactions between routing protocols and traffic flows across such large networks are very hard to conceive. Rarely can network operators predict accurately and completely what the effect of a routing protocol configuration change may be. Network measurements in conjunction with simulations capable of mimicking the network's functionality would be invaluable for the evaluation of configuration changes in the network.

2.6.7 Accounting and Pricing

Lastly, apart from the multiple applications that network measurements could have within the context of network management and design, they could also be used as a service to the customer. In fact, several large ISPs publish monthly-average values for the metrics included in their SLAs in dedicated web sites reporting on their network performance (Table 2.1).

ISP	**Network Performance Web Site**
AT&T	*http : //ipnetwork.bgtmo.ip.att.net/index.html*
Cable & Wireless	*http : //sla.cw.net/*
Qwest	*http : //stat.qwest.net*
Sprint	*http : //www.sprintesolutions.com/network/slas.jsp*
Worldcom	*http : //www1.worldcom.com/global/about/network/latency/*

Table 2.1: Performance Web Sites for Large IP Backbone Networks

Their methodology usually includes active measurements between different nodes in their network [34]. Additional measurement information specific to the customer could also be provided as a value-added service reporting on the performance experienced within the provider's network; currently, customers have no way of verifying the reported metrics. Similar measurements could also be used towards more intelligent accounting tools providing input to more sophisticated pricing schemes.

2.7 Summary

The distribution of traffic over a backbone network is the result of interactions between BGP, guiding entry and exit points for flows transiting the network, and IS-IS, dictating IP paths followed by flows throughout the network. Requirements for strict guarantees concerning the edge-to-edge delay experienced by packets through the network and availability constraints result in overprovisioned networks with large amounts of excess capacity, capable of absorbing sudden increases in load. Current network management practices are usually based on operational observations and the intuition of the network operators, rather than a complete understanding of the underlying network operations.

Measurements could enhance the current practices by offering some understanding about the underlying network operations. However, the only measurements available in today's networks are limited to the aggregate number of bytes flowing over a specific link, or link utilisation. In most cases, these metrics are provided as average values over fixed-time intervals, usually 5 minutes. There are no widely available systematic, analytical engineering tools that allow network operators to make more rational provisioning decisions and predict future provisioning needs.

In Chapter 3, we describe the State of the Art in the field of network measurements. We differentiate between active and passive measurements, and report on measurement efforts towards understanding of the internals of the dominant routing protocols. Lastly, we present current research efforts towards enhancing network provisioning and management practices for large-scale IP networks based on additional network measurements.

Chapter 3

State of the Art

3.1 Network Measurements

Network traffic has been analysed since the original development of the ARPANET [35]. Measurements collected in operational networks provide useful information about their performance and the characteristics of the traffic they carry. Measurement systems can be classified into two categories: (i) the *passive* and (ii) the *active* measurement systems. Passive measurement systems observe the traffic flowing through selected points inside the network. Analysis of the collected measurements leads to workload characterisation and gives insight into its characteristics. This method is non-intrusive in that it is designed in a way that the collected measurements are not significantly affected by the presence of the measurement system. Active measurements, on the other hand, inject probe traffic into the network and extrapolate the network performance based on the performance experienced by the traffic injected. Such active measurement techniques are usually considered intrusive, in the sense that probe traffic is likely to influence the network operation and hence the collected results. For instance, if an active measurement application aims at measuring the queueing delay observed on a particular link, then the existence of the packet probes themselves may alter the queueing behaviour observed. Therefore, active measurement tools have to be carefully designed in order to provide correct information on the targeted metric.

3.1.1 Active Network Measurements

Examples of simple active measurement tools are *Ping*, *Traceroute*, and *Pathchar*. *Ping* provides the round trip time between any source and any destination on the Internet at the network level, *Traceroute* the hops transited and their round trip times, and *Pathchar* the bandwidth, delay, average queue size and loss rate through every intermediate hop respectively. Other active measurement tools target the estimation of the *per-hop capacity*, such as *clink* [36], *pchar* [37], and the tailgating technique in [38], or the *end-to-end capacity*, such as *bprobe* [39], *nettimer*

[40], *pathrate* [41], and the *PBM methodology* in [42]. Further research toward the estimation of the available bandwidth along an IP path has led to *Delphi* [43], and *pathload* [44].

Within the context of active measurements Paxson has designed more complex systems [45, 42], that have been further developed into the NIMI (National Internet Measurement Infrastructure) project [46]. NIMI is a distributed measurement infrastructure, consisting of servers deployed at different points on the Internet. Monitoring traffic generated at specific NIMI servers is sent toward other servers within the NIMI infrastructure, and traffic characteristics are computed based on the transmitted "active probes". The computed traffic metrics include available bandwidth, delay, and packet loss, but they are specific to the IP paths between the servers. The large population of the deployed NIMI servers can provide some insight into the performance received by traffic throughout the Internet. The nature and number of probes injected into the network are defined based on the metric under observation and the level of accuracy needed in the analysis.

Other active measurement projects include Surveyor, PingER, RIPE Test Traffic, MINC and AMP. Surveyor uses a set of 41 GPS synchronised systems to measure one-way delay and loss [47]. PingER (Ping End-to-end Reporting) measures packet loss and available bandwidth between high energy nuclear and particle physics research facilities [48]. The RIPE Test Traffic measurement project measures bandwidth and delay performance between 24 measurement systems in Europe, North America and Israel [49]. MINC (Multicast-based Inference of Network-internal Characteristics) measurement systems transmit multicast probe messages to multiple destinations and infer link loss rates, delay distributions, and topology on the grounds of the correlations observed for the received packets [50]. Lastly, the AMP (NLANR Active Measurement Project) system consists of a set of monitoring stations that measure the performance of the vBNS network [51]. In addition, companies such as Keynote and Matrix conduct commercial network performance measurements [52, 53].

One of the major drawbacks of active measurement systems is that only a limited number of statistics can be observed at a time. More importantly, active measurement techniques do not offer the possibility of observing additional metrics for periods of time in the past. In other words, one is not capable of reviewing a period in the past and identifying the reason why certain metrics behave the way they did when recorded. Lastly, inability to control the IP path taken by probe packets makes interpretation of the collected measurements a very hard task.

Efforts toward the standardisation of performance metrics and active measurement techniques for their evaluation are carried out within the framework of the Internet Protocol Performance Metrics (IPPM) working group of the IETF (Internet Engineering Task Force). Already proposed standards include metrics for connectivity [54], one-way delay [55], one-way packet loss [56], round-trip delay [57], and bulk transfer capacity [58].

3.1.2 Passive Network Measurements

Passive network measurements are usually reliant on the existence of dedicated monitoring equipment in specific parts of the operational IP network. However, passive measurements can also be collected by network elements inside the network, that support the necessary functionality. For instance, in essence, SNMP provides passive measurements of specific metrics collected on every operational interface of a given router (these metrics are usually number of bytes sent/received, packet drops, and errors). In the same spirit, there are certain proprietary products that expand on this functionality, offering network statistics at a granularity finer than the one of aggregate statistics on a per link basis.

NetFlow is a proprietary passive measurement tool developed by Cisco [24]. NetFlow collects information about every TCP and UDP flow on a particular input or output link of a router. A flow record includes the source and destination addresses and port numbers, along with the number of bytes and packets transmitted, and the duration of the flow. The flow information is recorded by the router and collected by an external system periodically. While NetFlow is a powerful measurement tool, it may lose up to 90% of the measurement data during periods of heavy load at the router [59]. Moreover, it has been reported that it may impact the forwarding performance of the router, and thus is rarely activated by network operators for high-speed core routers. For manageable overhead on high-speed links, recent releases of NetFlow include support for sampling[1] and aggregation[2].

Within the same context, the IETF has made efforts to standardise flow-level measurements. The Real-Time Traffic Flow Meter (RTFM) working group [60, 61] provides a formal definition of flows and describes techniques for collecting the measurement data. More recently, the IP Flow Information Export (IPFIX) working group [62] defines a format and protocol for delivering flow-level data from the measurement device to other systems that archive and analyse the information.

OC3MON is a passive monitoring system for OC-3 links described in [63]. OC3MON collects timestamped packet-level traces or flow-level statistics. Packet-level traces can be collected for limited amounts of time, while flow-level statistics can be provided on a continuous basis. It has been deployed at two locations of the MCI backbone network providing information about daily and weekly variations in traffic volume, packet size distribution, and traffic

[1]http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s11/12s_sanf.htm

[2]http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t3/netflow.htm

composition in terms of protocols and applications [64]. Support for OC-12 link speeds has been incorporated in [65].

Passive monitoring systems require specific hardware to collect data on the network. For OC3MON, data capture relies on tapping the fibre through a dedicated network interface card (NIC). However, as link speeds increase, packet capturing at equivalent rates imposes serious challenges [66].

Passive systems, like OC3MON, are useful in that they allow for an exhaustive analysis and archiving of the collected information. However, they pose challenges as far as their deployment is concerned. Deployment of monitoring hardware throughout the entire network is cost-prohibitive. This is one of the reasons why global observations are usually approached using inference techniques rather than exhaustive monitoring.

3.2 The IP Monitoring (IPMON) infrastructure

The goal of the Sprint IP Monitoring System is to collect data from the Sprint Internet backbone that is needed to support a variety of research projects. The particular research projects include studying the behaviour of TCP, evaluating network delay performance, investigating the nature of denial of service attacks, and developing network engineering tools. While each project could develop a customised measurement system, many of the projects require similar types of data and installing monitoring equipment in a commercial network is a complex task making a general purpose measurement system more preferable.

To meet this goal, the IP Monitoring system is designed to collect and analyse GPS synchronised packet level traces from selected links in the Sprint Internet backbone. These packet traces consist of the first 44 bytes of every packet carried on the links along with a 64 bit timestamp. The clocks which generate the timestamps are synchronised to within 6 μs using a GPS reference clock. This provides the capability to measure one-way network delays and study correlations in traffic patterns.

Trace collection and analysis is accomplished using three separate systems. A set of data collection components (IPMON systems) collect the packet traces. The traces are transferred to a data repository which stores the traces until they are needed for analysis. Analysis is performed on a cluster of 16 Linux servers. The remainder of this section describes the requirements and architecture of the IPMON systems used for the packet trace collection.

3.2.1 IPMON system architecture

The monitoring systems, called IPMON systems, are responsible for collecting packet traces from the network. These systems consist of a Linux PC with a large disk array and a SONET

(Synchronous Optical NETwork) network interface, known as the DAG card [67, 68]. To collect the traces, an optical splitter is installed on selected OC-3, OC-12, and OC-48 links in the network, and one output of the splitter is connected to the DAG card in the IPMON system.

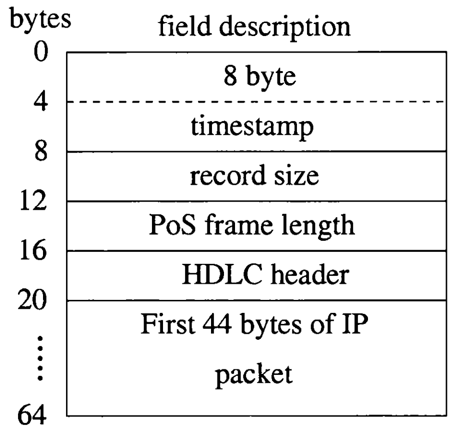


Figure 3.1: Packet Record Format

The DAG card decodes the SONET payloads and extracts the IP packets. When the beginning of a packet is identified, the DAG card generates a timestamp for the packet, extracts the first 48 bytes of the Packet over SONET (PoS) frame which contains 4 bytes of PoS header and 44 bytes of IP data, and transfers the packet record to main memory in the PC using DMA (Direct Memory Access). The format of the packet record is shown in Figure 3.1. If the packet contains fewer than 44 bytes, the data is padded with 0's. Once 1 MB of data has been copied to main memory, the DAG card generates an interrupt which triggers an application to copy the data from main memory to the hard disk. It would be possible to transfer the data from the DAG card directly to the hard disk, and bypass the main memory. Main memory, however, is necessary to buffer bursts of traffic as described later in the section.

The IPMON system has 5 basic design requirements:

- Support data rates ranging from 155 Mbps (OC-3) to 2.5 Gbps (OC-48).
- Provide synchronised timestamps.
- Occupy a minimal amount of physical space.
- Prevent unauthorised access to trace data.
- Be capable of remote administration.

In the next section, we describe how the system design meets each one of these requirements.

Table 3.1: Data rate requirements

	OC-3	OC-12	OC-48
link rate (Mbps)	155	622	2480
peak capture rate (Mbps)	248	992	3968
1 hour trace size (GB)	11	42	176

3.2.2 Data Rate Requirements

The data rate requirements for OC-3, OC-12, and OC-48 links are summarised in Table 3.1. The first row of the table shows the data rate at which the DAG card must be able to process incoming packets. After the DAG card has received a packet and extracted the first 44 bytes, the timestamp and additional header information is added to the packet record and copied to main memory. If there is a sequence of consecutive packets whose size is less than 64 bytes, then the amount of data that is stored to main memory is actually greater than the line rate of the monitored link. The amount of internal bandwidth required to copy a sequence of records corresponding to minimum size TCP packets (40 byte packets) from the DAG card to main memory is shown on the second row of Table 3.1. To support this data rate, the OC-3 and OC-12 IPMON systems use a standard 32 bit, 33 MHz PCI bus which has a capacity of 1056 Mbps (132 MB/sec). The OC-48 system requires a 64 bit, 66 MHz PCI bus with a capacity of 4224 Mbps (528 MB/sec). It is possible to have non-TCP packets which are smaller than 40 bytes resulting in even higher bandwidth requirements, but the system is not designed to handle extended bursts of these packets as they do not occur very frequently. It is assumed that the small buffers located on the DAG card can handle short bursts of packets less than 40 bytes in size.

Once the data has been stored in main memory, the system must be able to copy the data from memory to disk. The bandwidth required for this operation, however, is significantly lower than the amount of bandwidth needed to copy the data from the DAG card to main memory as the main memory buffers bursts of small packets before storing them to disk. Only 64 bytes of information are recorded for each packet that is observed on the link. As reported in prior studies, the average packet size observed on backbone links ranges from about 300 bytes to 400 bytes during the busy periods of the day [69]. For our design, we assume an average packet size of 400 bytes. The disk I/O bandwidth requirements are therefore only 16% of the actual link rate. For OC-3 this is 24.8 Mbps; for OC-12, 99.5 Mbps; and for OC-48, 396.8 Mbps. To support these data rates, we use a three-disk RAID array for the OC-3 and OC-12 systems which has an I/O capacity of 240 Mbps (30 MB/sec). The RAID array uses a software RAID controller available with Linux. To support OC-48 we use a five-disk RAID array with higher performance disks that can support 400 Mbps (50 MB/sec) transfers. To minimise interference with the data being transferred from the DAG card to memory, the disk controllers use a separate 32 bit 33 MHz PCI bus.

3.2.3 Timestamp Requirements

In order to be able to correlate traces collected in diverse geographical locations, the packet timestamps generated by each IPMON system need to be synchronised to a global clock signal. For that reason each DAG card features a dedicated clock on board. This clock runs at a rate of 16MHz which provides a granularity of 59.6 ns between clock ticks. Packets are not timestamped immediately when they arrive at the DAG card. They first pass through a chip which implements the SONET framing, and which operates on 53 bytes ATM cells. Once this buffer is full, an interrupt is generated, and the packet is timestamped. In other words, timestamping happens on the unit of 53 bytes, thus introducing a maximum timestamp error of 2 μs, since this is the time needed for the transmission of 53 bytes on an OC-3 link. OC-3 links are the lowest speed monitored links in our network. Thus, for higher capacity links this error will be smaller.

Due to room temperature and the quality of the oscillator on board the DAG card, the oscillator may run faster or slower than 16 MHz. For that reason, it is necessary to discipline the clocks using an external stratum 1 GPS receiver located at the PoP. The GPS receiver outputs a 1 pulse-per-second (PPS) signal which is distributed to all of the DAG cards located at the PoP.

Clock synchronisation on board the DAG card is achieved in the following fashion [70]. At the beginning of the trace collection the clock is loaded with the absolute time from the PC's system clock (e.g. 7:00 am Aug 9, 2000 PST). The clock then begins to increment at a rate of 16 MHz. When the DAG card receives the first 1 PPS signal after initialisation, it resets the lower 24 bits of the clock counter. Thereafter, each time the DAG card receives the 1 PPS signal, it compares the lower 24 bits of the clock to 0. If the value is greater than 0, the oscillator is running fast and the DAG card decreases the frequency. If the value is less than 0, the oscillator is running slow and the DAG card increases the frequency.

In addition to synchronising the DAG clocks, the monitoring systems must also synchronise their own internal clocks so that the DAG clock is correctly initialised. This is accomplished using the Network Time Protocol (NTP). A broadcast NTP server is installed on the LAN which is connected to the monitoring systems and is capable of synchronising the system clocks to within 200 ms. This is sufficient to synchronise the beginning of the traces, and the

1 PPS signal is used to further synchronise the DAG clock. There is an initial period when the DAG cards adjust the initial clock skew, so we ignore the first 30 seconds of each trace.

There are several sources of error that may occur in the synchronisation of the systems. We'll distinguish between two types of errors: (a) timestamping errors specific to a single DAG card, and (b) synchronisation errors between multiple DAG cards.

As already mentioned, the DAG card uses an ATM cell buffer to store the captured packet until it is timestamped. That may introduce a maximum error of 2 μs. Given that the resolution of the time tick on board the DAG card is 59.6 ns, the use of this ATM cell buffer introduces a maximum *timestamping* error of 2 μs.

All DAG cards in a specific PoP use the same GPS receiver for their clock synchronisation. Therefore, synchronisation errors between DAG cards in the same PoP could be due to two possible reasons. The first one is the difference in propagation time for the 1 PPS signal. The 1 PPS signal is distributed to the DAG cards using a daisy chain topology. The difference in cable length between the first and the last system is 8 meters, which corresponds to a propagation delay of 28 ns. The second source of error is due to the fact that the clock synchronisation mechanism cannot immediately adjust to changes in the oscillator frequency. Once the DAG card receives the 1 PPS interrupt, it has to increase or decrease the oscillator frequency depending on the clock offset. We measured in the lab the maximum clock offset observed when the card receives the 1 PPS interrupt. Its maximum value was 30 clock ticks, representing an error of 1.79 μs, while the median error was a single clock tick (59.6 ns). Accounting for those errors, the worst case skew between any two DAG clocks, participating in single-hop measurements, is less than 2 μs. The total effect of both types of errors is a maximum clock skew of 6 μs.

3.2.4 Physical Requirements

In addition to supporting the bandwidth requirements of OC-3, OC-12, and OC-48 links, the IPMON systems must also have a large amount of hard disk storage capacity to record the traces. As the systems are installed in a commercial network facility where physical space is a scarce resource, this disk space must be contained in small form factor. Using a rack-optimised system, the OC-3 and OC-12 IPMONs are able to handle 108 GB of storage in only 4U of rack space[3]. This allows the system to record data for 9.8 hours on a fully utilised OC-3 link or 2.6 hours on a fully utilised OC-12 link. The OC-48 systems have a storage capacity of 360 GB, but in a slightly larger 7U form factor. This is sufficient to collect a 2 hour trace on a fully utilised link. Fortunately, the average link utilisation on most links is less than 50%, allowing for longer trace collection.

[3]1U is a standard measure of rack space and is equal to 1.75 inches or 4.45 cm.

The physical size constraint is one of the major limitations of the IPMON system. Collecting packet level traces requires significant amounts of hardware. These traces are critical for conducting research activities, but trace collection is not a scalable solution for operational monitoring of an entire network. The ideal solution is to use traces collected by the IPMON system to study the traffic and develop more efficient monitoring systems targeted toward monitoring of all the links inside a network for operational purposes.

3.2.5 Security Requirements

The IPMON systems collect proprietary data about the traffic on the Sprint Internet backbone. Preventing unauthorised access to this trace data is an important design requirement. This includes preventing access to trace data stored on the systems as well as preventing access to the systems in order to collect new data. To accomplish this, the systems are configured to accept network traffic only from two applications: *ssh* and NTP. *ssh* is an authenticated and encrypted communication program similar to *telnet* that provides access to a command line interface to the system. This command line interface is the only way to access trace data that has been collected by the system and to schedule new trace collections. *ssh* only accepts connections from a server in our lab and it uses an RSA key based system to authenticate users. All data that is transmitted over the *ssh* connection is encrypted.

The second type of network traffic accepted by the IPMON systems is NTP traffic. The systems only accept NTP messages which are transmitted as broadcast messages on a local network used exclusively by the IPMON systems. All broadcast messages which do not originate on this network are filtered.

3.2.6 Remote Administration Requirements

In addition to being secure, the IPMON systems must also be robust against failures since they are installed, in some cases, where there is no human presence. To detect failures, a server in the lab periodically sends query messages to the IPMON systems. The response indicates the status of the DAG cards and of the NTP synchronisation. If the response indicates either of these components fails, the server attempts to restart the component through an *ssh* tunnel. If the server is not able to correct the problem it notifies the system administrator that manual intervention is required. In some cases, even the *ssh* connection will fail, and the systems cannot be accessed over the network. To handle this type of failure, the systems are configured with a remote administration card that provides the capability to reboot the machine remotely. The remote administration card also provides remote access to the system console during boot time. In cases of extreme failure, the system administrator can boot from a write-protected floppy installed in the systems and completely reinstall the operating system remotely.

The one event that cannot be handled remotely is hardware failure. The monitoring systems play no role in the operation of the backbone, and thus we decided not to provide hardware redundancy to handle failures.

> The design of the IPMON monitoring systems was primarily performed by Chuck Fraleigh. My involvement was mainly in the debugging process of the DAG cards and the resolution of GPS synchronisation issues between systems. Later on, I was also responsible for the trace collection and the verification of correctness of the collected information.

3.3 Routing Protocol Listeners

Traffic flow over the Internet is determined by routing protocols. As described in Chapter 2, these protocols are divided into two categories: (i) the intra-domain protocols, guiding the flow of traffic throughout a single autonomous system, and (ii) the inter-domain routing protocols overseeing the exchange of traffic between autonomous systems. All network measurement systems mentioned so far observe the behaviour of the network as the result of the effect that those routing protocols have on the network traffic demands. Recently, research in the area of network measurements turned its attention to the internals of those protocols within a measurement context. The systems described in this section can also be considered passive measurement systems, but their functionality is limited to recording routing directives as issued by the routers in a network.

A *routing protocol listener* is a system establishing routing sessions with the operational routers inside a network with the intent to record all the messages exchanged in the routing domain. To the best of our knowledge, there are only two publicly available implementations of such systems: (i) zebra[4], and (ii) PyRT[5]. In fact, zebra is more than just a routing protocol listener. A system with the zebra software installed acts as a dedicated router. It can exchange routing information with other routers, and update its kernel routing table accordingly. Currently supported routing protocols include RIP, OSPF, and BGP.

The Python Routeing Toolkit (PyRT) is a tool written in Python [71] that supports the passive collection and the off-line analysis of routing protocol data forming minimal router peering sessions and dumping the routing PDUs (Packet Data Units) received over these sessions. PyRT currently supports BGPv4 and IS-IS. Compared with full implementations of routing protocols, such as zebra, an advantage of PyRT is that no routing information is ever advertised on these

[4]The official distribution site for zebra is at $http://www.zebra.org$.

[5]The official distribution site for PyRT is at $http://www.sprintlabs.com/Department/IP-Interworking/Routing/PyRT/$.

sessions and only a minimal amount of information is injected into the network.

Only until recently were such systems deployed within operational networks. However, BGP routing information has been collected since 1989 in the form of periodical BGP routing table dumps. The collected information is analyzed in [72].

3.4 Network Provisioning Based on Measurements

The majority of the projects in the field of active and passive network measurements usually focuses on the performance evaluation of a network, or the characterization of its workload. In other words, it focuses on the estimation and presentation of specific network performance metrics, such as delay and loss, or the breakdown of the carried traffic into flows, applications, and protocols [65, 73].

However, the analysis of network measurements collected on the Internet is capable of providing us with more than just simple performance statistics. Thorough analysis of collected active and passive measurements can lead to a better understanding of the way IP networks function, and ways in which network management and engineering tasks can be enhanced. In order to achieve such a goal, though, several types of measurement data, collected at multiple locations inside the network, must be carefully combined.

3.4.1 Network Traffic Demand Matrix

The Netscope project [25] collects measurements from the AT&T network in order to study the effect of alternate routes and route configuration onto the network. The Netscope project collects packet-level data from T3 and FDDI links using the proprietary PacketScope measurement system, as well as NetFlow measurements from routers [74]. Additional information on router configuration and forwarding tables is downloaded once per day from the operational routers. Combining all sources of information, the traffic demands across the network are computed [59]. Those traffic demands correspond to the volume of traffic exchanged between an ingress link, where flow measurements are available, and the set of possible egress points as computed based on the information exchanged through BGP. The traffic demand is then input into a simulator which is used to determine the effect of changing the network configuration.

The traffic demands derived for each ingress to a set of possible egress points correspond to one row of the point-to-point *traffic matrix*, at the aggregation level of a link (as described in Section 2.6.2). As a result, the proposed methodology can only be used to evaluate the effect of routing protocol configuration changes on the customer traffic, for which measurements are available. Evaluation of different scenarios concerning different traffic flows require the existence of the complete network traffic matrix.

A point-to-point traffic matrix on the aggregation level of a link contains one million entries, for a network featuring one thousand active links. Therefore, for large-scale IP networks, like the Sprint IP backbone, computation of the network-wide traffic matrix involves large overhead. As a result, in the past few years, there has been a research effort toward estimating the traffic matrix based on available network-wide measurements, like SNMP information [75, 76, 77]. The proposed techniques are compared in [30]. Scalability and accuracy of the proposed methods are limiting factors in the estimation process of a network-wide traffic matrix. An alternative technique based on "gravity models" is used for the estimation of a PoP-to-PoP traffic matrix in [30]. Knowledge of this matrix could provide network operations with useful knowledge about the traffic flowing on the most expensive part of the network, i.e. the backbone.

In the following section we provide a description of other network management tasks that could benefit from the existence of a network-wide traffic matrix.

3.4.2 Traffic Engineering

Traffic engineering involves controlling the flow of traffic on the backbone, and is dictated by the routing protocols deployed. As described in Chapter 2, OSPF/IS-IS weights express administrative preference of certain IP paths over others. These paths may be preferred because they correspond to increased capacity, smaller propagation delays, or even redundancy at the physical layer. Knowledge about the traffic demands flowing on the network could be useful for the optimal setting of the weights used by intra-domain routing protocols.

A significant amount of research has been carried out in the area of intra-domain routing with respect to techniques that are able to produce link weights across an entire network so as to achieve specific performance objectives. The simplest objective function investigated is the minimisation of the maximum utilisation across the network. Under such an objective link weights are selected such that network traffic is spread across the network in a way that no parts are overutilized while others carry minimal amounts of traffic. Heuristics for the computation of OSPF/IS-IS weights achieving the described objectives are proposed in [78, 79], and have been shown to provide solutions within a few percent of the optimal general routing, where the flow for each demand is optimally distributed over all paths between source and destination. Other objective functions could target the minimisation of the maximum delay across the network, maximisation of the average network-wide link utilisation, etc.

The techniques proposed for the selection of OSPF and IS-IS weights lead to near-optimal routing of the traffic demands across the network under normal operation. However, in the case of a link failure, new IP paths across the network are going to be selected and the resulting flow of traffic may lead to violations of the performance objectives defined by the network operator. Under such circumstances, the operator has to manually interfere and divert traffic from the congested parts in the network, increasing the link weights of the congested links or re-calculating the optimal link weights under the new network topology, and traffic demands. Within the same context as the one for the selection of OSPF/IS-IS weights new heuristics have been proposed to compute intra-domain link weights that will achieve the desired objectives even in case of failures [80, 81].

The above techniques lead to promising results regarding the ability of intra-domain routing protocols of making optimal use of the network resources by distributing the traffic across diverse paths to achieve a provider specific performance objective. However, all solutions rely on the explicit knowledge of the network traffic matrix.

3.4.3 Tuning Routing Protocol Parameters

Emerging voice and video applications assume a stable underlying inter-domain forwarding infrastructure and fast IP path restoration. End-to-end availability and reliability of data networks is essential for the rollout of such services. However, recent studies in the area of inter-domain routing have shown that current BGP implementations may lead to temporary routing table oscillations during the BGP path selection process on Internet backbone routers [82]. During these periods of delayed convergence end-to-end Internet paths will experience intermittent loss of connectivity, as well as increased packet loss and delay. Moreover, it may take up to fifteen minutes until a consistent view of the network topology is reached after a failure [82, 83]. Addition of new autonomous systems to the Internet is likely to aggravate this condition. Studies have shown that the BGP convergence time increases linearly with the number of active autonomous systems in the best case, and exponentially in the worst [83].

The results reported above are derived from the experimental instrumentation of key portions of the Internet infrastructure. In [83], the authors injected 250,000 routing faults, over the course of two years, into geographically and topologically diverse peering sessions with five major commercial Internet service providers. They then measured the impact of these faults through both end-to-end measurements and recording of ISP backbone routing table changes. The behaviour reported can be improved upon through changes in current BGP implementations. However, these changes to BGP will come at the expense of a more complex protocol and increased router overhead.

Previous work by Labovitz et al. had shown that most Internet routing instability in 1997 was pathological and stemmed from software bugs and artifacts of router vendor implementation decisions [84]. Some of these implementation decisions were related to the default values of specific protocol timers responsible for the re-computation of the BGP routing table. Later work by Labovitz et al. showed that after the changes suggested in [84] were incorporated in commercial implementations the level of routing instability dropped by several orders of magnitude [85].

All previous efforts have targeted the evaluation of the convergence properties of the dominant inter-domain routing protocol, BGP. Recently, there have been further efforts towards the understanding of the converge properties of intra-domain protocols, such as IS-IS. In [86] is presented a controlled experiment performed in the operational Sprint IP backbone network, where specific links throughout the network are shut down and IS-IS convergence time is measured through a combination of active and passive measurements. The total convergence time is broken down into the time taken for specific operations in the convergence process. Results show that IS-IS takes seconds to converge due to the presence of two protocol timers, set to default values of seconds.

In effect, both inter-domain and intra-domain routing protocols offer the network operators knobs that they could tune to achieve the performance desired. However, most of the time, those knobs, usually protocol timers, are left set at the default values, since the network operator has no clear intuition about the effect that a timer change may have. Only experimental analysis and actual measurements taken in operational networks can provide some understanding of the magnitude of the values that these timers should be set to.

3.5 Summary

In this chapter, we demonstrated the diverse applications of network measurements in the context of network provisioning. It is evident that network management and planning for large ISP networks is a very challenging task. Current market conditions dictate optimal use of the network resources, and performance within the bounds defined in rather competitive SLAs. Network measurements can provide the necessary knowledge that could allow the formalisation of different network provisioning and planning tasks. Moreover, the development of sound methodological approaches toward the estimation of different performance metrics can serve as a framework, according to which ISPs and customers can evaluate the performance offered by packet-switched networks like the Internet.

In the next three chapters we present our work on ways in which current network management and planning tasks could be improved upon in the presence of additional network measurements.

Chapter 4

Analysis of Measured Single-Hop Delay from an Operational Backbone Network

In this chapter, we measure and analyse the single-hop packet delay through operational routers in the Sprint IP backbone network. After presenting our delay measurements through a single router for OC-3, and OC-12 link speeds, we propose a methodology to identify the factors contributing to single-hop delay. In addition to packet processing, transmission, and queueing delay at the output link, we observe the presence of very large delays that cannot be explained within the context of a FIFO output queue model. We isolate and analyse these outliers.

Results indicate that there is very little queueing taking place in today's backbone. As link speeds increase, the most dominant part of single-hop delay is packet processing. We show that if a packet is received and transmitted on the same linecard, it experiences less than 20 μs of delay. If the packet needs to be transmitted across the switch fabric, its delay doubles in magnitude. We observe that processing due to IP options results in single-hop delays in the order of milliseconds. Milliseconds of delay may also be experienced by packets that do not carry IP options. We attribute those delays to router idiosyncratic behaviour, that affects less than 1% of the packets. Lastly, we show that the queueing delay distribution is long-tailed and can be approximated with a Weibull distribution with the scale parameter, $a = 0.5$, and the shape parameter, $b = 0.6$ to 0.82. The measured average queueing delay is larger than predicted by M/M/1, and FBM models when the link utilisation is below 70%, but its absolute value is quite small.

Required Network Measurements and Time Granularity

Computation of single-hop delay requires measurements taken at the finest granularity, i.e. detailed packet traces with accurate GPS timestamps. The results presented in this chapter are based on IPMON packet traces collected on specific links in the Sprint IP backbone network, and topological information processed to identify links connected to the same routers inside the network.

4.1 Introduction

Delay is a key metric in data network performance and a parameter in ISP's Service Level Agreements. In the Internet, packets experience delay due to transmission and propagation through the medium, as well as queueing due to cross traffic at routers. The characteristics of the traffic have significant impact on the queueing delay. Willinger et. al first reported that network traffic is self-similar rather than Poisson [33], and much research has been done since to explore the consequences of non-Poisson traffic on queueing delay. The Fractional Brownian Motion (FBM) model has been proposed to capture the coarse time scale behaviour of network traffic, and results in queueing behaviour that diverges significantly from that of the Poisson traffic model [87, 88]. Follow-up work shows that the wide-area network traffic is multi-fractal and exhibits varying scaling behaviour depending on the time scale [89]. Recent work reveals that the queueing behaviour can be approximated differently depending on the link utilisation [90].

The above analyses, however, have been based on packet traces collected from a single link and fed into an output buffer, whose size and service rate vary. We are not aware of any measurement of the queueing delay on operational routers. The difficulty in measuring single-hop delay in a real network is threefold:

- Packet timestamps must be accurate enough to allow the calculation of the transit time through a router. This requires in particular that the measurement systems (i) offer sufficient resolution to distinguish the arrival times of two consecutive packets, and (ii) are synchronised to an accurate global clock signal, such as Global Positioning System (GPS). These two conditions need to be met so that the maximum clock skew between any two measurement cards is limited enough to allow accurate calculation of the transit time of a packet from one interface to another interface of the same router.

- The amount of data easily reaches hundreds of gigabytes. Data from input and output links need to be matched to compute the time spent in the router.

- Routers have many interfaces; tapping all the input and output links to have a complete picture of the queueing behaviour of any single output link is unrealistic in an operational network.

As described in Section 3.2, we have designed a measurement system that addresses the first two of the above difficulties, and deployed it in the Sprint Tier-1 IP backbone network to collect packet traces with accurate timestamps. We use optical splitters to capture and timestamp every packet traversing a link (see details in Section 3.2). We obtain the single-hop delay of packets by computing the difference between the timestamps at the input and output monitored links. The third difficulty is not easy to overcome due to deployment cost and space issues. Although this prevents us from characterising the queueing experienced by *all* packets, it does not affect the evaluation of the single-hop delay reported in this chapter.

In Section 4.2 we describe the measurement environment where our data were collected. In Section 4.3 we present delay measurements of one hundred million packets matched among more than four billion packets and 400 gigabytes of data collected from the Sprint IP backbone network. In Section 4.4 we provide a methodology for quantifying the various elements in single-hop delay. We identify the impact that (i) transmission across the switch fabric, (ii) the presence of IP options, and (iii) increased output link speed have on the delay experienced by packets through a single node. Surprisingly, in addition to the expected elements, such as transmission, queueing, and processing delays, we observe very long delays that cannot be attributed to queueing at the output link. We use a single output queue model to isolate these delays, and discuss their potential origin. Once the queueing delay component has been quantified, we analyse its tail behaviour in Section 4.5. We summarise our findings in Section 4.6.

4.2 Measurement Environment

We have designed passive monitoring systems that are capable of collecting and timestamping the first 44 bytes of all IP packets at link speeds up to OC-48 (2.5Gbps), using the DAG card [67]. These monitoring systems have been deployed on various links in four Points of Presence (PoPs) of the Sprint IP backbone. We have collected day-long packet traces, and analysed them off-line. Details about the measurement infrastructure can be found in Section 3.2.

The monitoring systems are GPS synchronised and offer a maximum clock skew of 6 μs. Details on the clock synchronisation and possible errors in the accuracy of our delay measurements can be found in Section 3.2.3.

4.2.1 Collected Data

We tap into the optical fibre and capture packets just before they enter and just after they leave a router. We denote the packet arrival time at an input link as T_{in} and the packet departure at an output link, as T_{out}. For any given packet n, the single-hop delay through the router is the difference between its arrival and departure timestamps: $d(n) = T_{out}(n) - T_{in}(n)$. This single-hop delay value corresponds to the total time a packet spends inside a router.

Packet traces from more than 30 links, both OC-3 and OC-12, have been analysed. In this chapter, we use packet traces from four OC-3 links, collected on August 9th, 2000, and four OC-12 links, collected on September 5th, 2001. Those link pairs have been selected because they exhibit the highest delays observed among all our measurements. Table 4.1 provides further details about these eight traces. We label a router's inbound link as `in`, and a router's outbound link as `out`, and refer to them as a *data set* for the remainder of the chapter.

Figure 4.1 depicts a typical setup in a PoP. Monitoring systems are attached to selected links inside a PoP. For the eight traces selected, those links were attached to quad-OC-3 and quad-OC-12 linecards. Those linecards accommodate four equal speed interfaces, as shown in Figure 4.1. Packets may transit the router from one linecard to another (`set1`, `set2`), or from one interface of a linecard to another interface of the same linecard (`set3`).

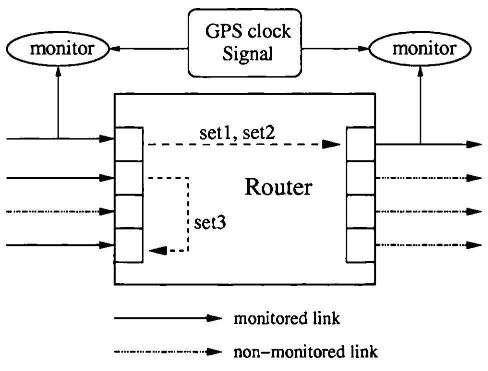


Figure 4.1: Typical configuration of monitoring systems in a PoP.

Set	Link	Speed	Date dd/mm/yy	Start (UTC)	End (UTC)	# packets	Avg. Util.	# matches
1	`in1`	OC-3	09/08/00	16:56:33	02:56:07	793.5×10^6	70 Mbps	2.8×10^6
	`out1`	OC-3	09/08/00	16:56:00	02:56:07	567.7×10^6	60 Mbps	
2	`in2`	OC-3	09/08/00	16:56:03	17:41:04	28.2×10^6	30 Mbps	1.2×10^6
	`out2`	OC-3	09/08/00	16:56:04	17:41:04	48.9×10^6	50 Mbps	
3	`in3`	OC-12	05/09/01	05:00:34	11:17:11	1.4×10^9	150 Mbps	17.6×10^6
	`out3`	OC-12	05/09/01	05:00:34	11:17:11	1.1×10^9	250 Mbps	
4	`in4`	OC-12	05/09/01	05:03:15	17:32:50	157.5×10^6	6 Mbps	70.4×10^6
	`out4`	OC-12	05/09/01	05:03:15	17:32:50	169×10^6	6 Mbps	

Table 4.1: Details of traces

In Table 4.2, we present the architectural details of each data set collected. We denote each router participating in the measurements with its own index j. Data `set1` and `set2` were collected through the same router (Router1), and they correspond to the forward and reverse direction of the same router path (the incoming link of `set1` is the outgoing link of `set2`, and vice-versa). All data sets capture the behaviour of the path between two quad linecards, with the exception of `set4`, that corresponds to the path between a quad-OC12 card and a single OC-12 card. Data `set1`, `set2`, and `set4` correspond to measurements involving two different linecards, whereas `set3` corresponds to measurements collected on the same linecard. In Section 4.4.2 we show how such architectural differences affect the delay values experienced by packets through a router.

Set	From	To	Router Name	Same Linecard	Different Linecard
1	quad-OC3	quad-OC3	Router1		√
2	quad-OC3	quad-OC3	Router1		√
3	quad-OC12	quad-OC12	Router2	√	
4	quad-OC12	OC12	Router3		√

Table 4.2: Architectural details for the routers where the traces were collected.

4.2.2 Router Architecture

All measurements are collected from routers of the same manufacturer, and of the same architecture, running the same operating system version. Nevertheless, each router features a different number of active links, of different capacities, and is located in a different part of the network. In this section, we briefly describe their architectural design.

In the general architecture of an IP router, router functions can be separated into two types:

- Datapath functions: operations that are performed on every datagram traversing the router. Such operations are most often implemented in special purpose hardware, and include the forwarding decision, the backplane and output link scheduling.
- Control functions: operations that are performed relatively infrequently. Such operations are invariably implemented in software and include the exchange of routing table information with neighbouring routers, as well as system configuration and management.

As a consequence, when investigating the performance experienced by different packets through a single router, we naturally focus on the datapath functions. Tracing the typical path a packet has to take through an IP router we have:

- Forwarding Decision: When a packet arrives at the input link of an IP router its destination address is looked up in the forwarding table, and the appropriate destination interface is identified. Its TTL (Time to Live) field is decremented, and a new header checksum is computed.

- Backplane Transit: The packet is then forwarded across the backplane to the appropriate output port. If the backplane is not available to transmit the packet from the input to the output port, then the packet has to be queued. Moreover, if there is not sufficient queue space for the packet to be stored, then the packet may need to be dropped or preempt some other packet from the queue.

- Output-Link Scheduler: Once the packet arrives at the outgoing port, it may need to wait for its turn at the output queue. Usually this queue is a First Come First Serve (FCFS) queue, which implies that packets are served in order of arrival at the queue.

The routers participating in our measurements are usually classified as high-performance routers. Specific architectural design decisions include [91]:

- The forwarding table is updated at the main router processor and distributed across all the line cards. The forwarding decision is made by dedicated CPUs residing at the line cards themselves.

- The backplane is a *crossbar switch*, where multiple cards can communicate with each other simultaneously.

The crossbar switch can be conceived as a grid among all the line cards of a router (Figure 4.2). By closing several crosspoints at the same time, the switch can transfer packets between multiple ports simultaneously. The crosspoints of this grid are controlled by a centralised scheduler, which selects the configuration of the crossbar such that at any one instant each input port is connected to at most one output, and that each output port is connected at most to one input.

Crossbar switches may be designed to transfer fixed or variable length units. The routers participating in our measurements feature crossbar switches designed to transfer fixed size units, also known as cells. In other words, when a packet is ready to be transmitted across the crossbar, it is broken down into fixed-length cells, which are transmitted independently. When all the cells arrive at the output port, then the packet is reassembled and transmitted at the output link.

Although a crossbar switch is internally non-blocking, there are three other types of blocking that can take place inside the router: (i) Head Of Line blocking (HOL), (ii) input blocking, or (iii) output blocking. If all the packets at the input link are stored in a First In First Out (FIFO) queue, then head of line blocking occurs because a packet is considered by the centralised scheduler only when it reaches the head of the queue. Therefore, if the front packet

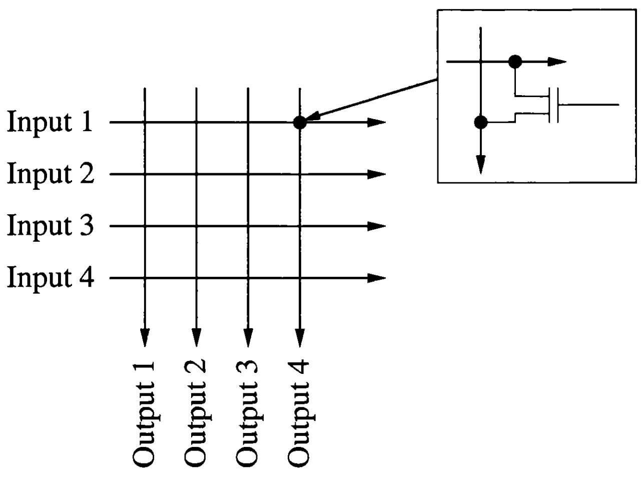


Figure 4.2: A Four-Input Crossbar Interconnection Fabric.

in the input queue is destined towards a particular output port, which happens to be busy, it prevents from transmission all other packets behind it, which may be destined to other, possibly idle output ports.

In order to avoid such a phenomenon, the routers at our disposal implement what is often referred to as "Virtual Output Queueing (VOQ)". Under such a scheme, each input interface features a separate FIFO queue for each output interface (Figure 4.3). Once the forwarding decision is made, packets are placed in the appropriate virtual output queue. A centralised scheduler examines the contents of all the input queues, and produces the appropriate crossbar configuration.

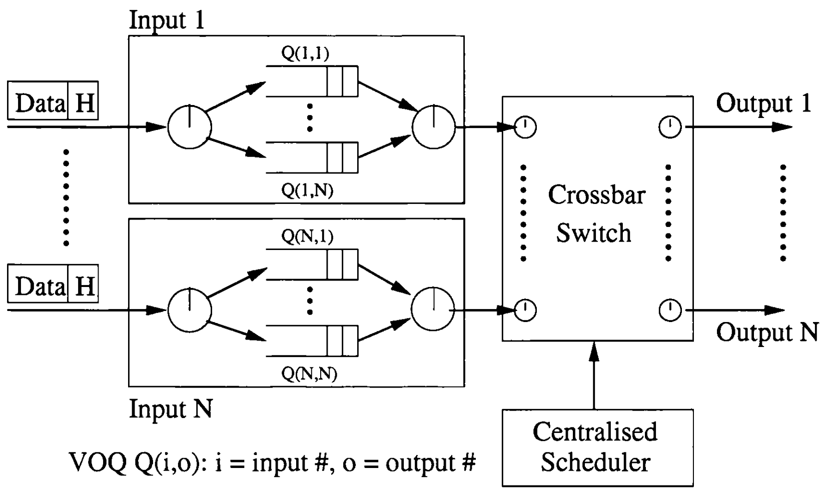


Figure 4.3: Architecture of the routers participating in the measurements.

Virtual Output Queueing resolves the issue of HOL, but not of the input or output blocking.

Input and output blocking are present in all crossbar switches. They arise due to contention for access to the crossbar: each input and each output can transport or receive only one cell at a time. Therefore, if multiple cells require access to the same input or output simultaneously, only one will be granted access, and the others will have to wait. Such contention for access to the crossbar may lead to increased delays for packets that have to wait multiple time slots until they are transmitted to the appropriate output.

Arbitration at the crossbar is based on an iterative algorithm called iSLIP [92]. According to [92], at each time slot, multiple iterations are performed to select a crossbar configuration, matching inputs to outputs. Using rotating priority ("round-robin") arbitration each input and output are scheduled in turn.

The current best practice in order to limit the effect of input and output blocking inside a router is to run the crossbar switch faster than the external line rate. Such a technique is usually referred to as "speedup" [91]. If one chooses to operate the crossbar switch twice as fast as the external line, then during each cell time one can transfer two cells from each input port to each output port. In such a case, we say that the router offers a speedup of two. Alternatively, one may choose to use multiple crossbar switches, or "slices", in parallel in order to form a switch core. During each time slot, all n crossbar slices are configured the same way by the centralised scheduler, forming a n-bit wide bus for each line card to the switch core. Our routers use the latter approach offering a switch core comprising four crossbar "slices", with access of 1.25 Gbps towards each one of them. This translates into per link transmission rates of 5 Gbps across the crossbar.

Knowledge of such router architectural details will be proven essential in the delay analysis presented in Section 4.4.

4.3 Delay Measurement

4.3.1 Matching Packets

The first step in our methodology is to identify those packets that arrive on the input links and depart on the output links we monitor. We use hashing to match packets efficiently. The hash function is based on the CRC-32 algorithm [93]. Only 30 bytes out of the 44 bytes are hashed, including the source and destination IP addresses, the IP header identification number and the whole IP header data part, as shown in Figure 4.4. The other fields are not used since they may be modified by the router (e.g. TTL) or carry almost identical information in all IP packets (e.g. IP version field, TOS byte[1]). Using the 24 least significant bits of the CRC-32 value,

[1]Our routers are never configured to modify the TOS byte.

the hash function offers an average load factor of 5.7% when one million packets are hashed into a single table. We decided to use hash tables of one million packets, because one million average-sized packets transmitted at OC-3 speeds correspond to time periods larger than one second. We assume that one second is the maximum delay a packet can experience through a single node. The hash table size is increased to four million packets for the processing of the OC-12 traces for similar reasons.

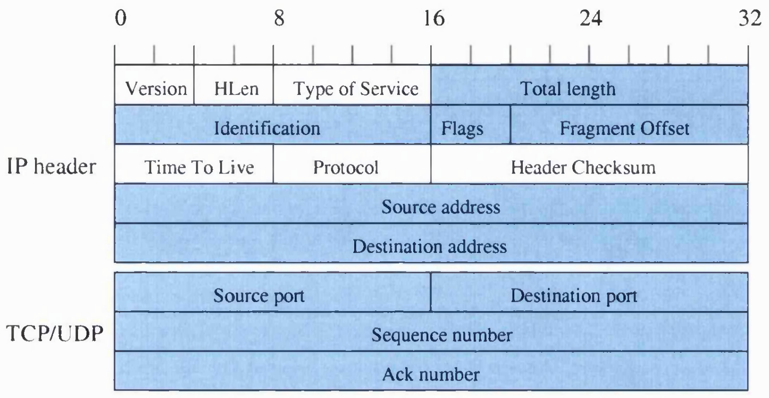


Figure 4.4: Fields in the packet header used in the hash function.

To match packets, the traces are processed as follows: The first million packets from `out` are hashed into a table called H_1, and the timestamp of the last packet is recorded as $e(H_1)$. Then, in order of arrival, each packet from `in` is hashed and its key value is used as an index in H_1. If table H_1 contains a packet for that specific index, we compare *all* 44 bytes of the two packets. If they are the same, we have a match and we output a record of all its 44 bytes, along with the timestamps for its arrival on link `in` and departure on link `out`. This process continues until we reach a packet from `in` that has a timestamp one second or less than $e(H_1)$. Then we hash the next one million packets from `out` and create a second hash table H_2. Both H_1 and H_2 are used until the timestamp for a packet from `in` is greater than $e(H_1)$. When this happens, H_2 replaces H_1, and the processing continues.

Duplicate packets have been reported previously [94]. We occasionally observe them in our traces (they account for less than 0.001% of our packets), and have paid special attention to matching them. Duplicate packets have all 44 bytes identical, and therefore hash to the same value. In most cases we find that only after a packet left `out`, its duplicate arrived on `in`, making the classification unambiguous. If that is the case, then we can match the duplicate packets with the correct arrival and departure timestamps. In other cases, we ignore the matches.

As a result of the above process, four traces of *matched* packets are produced. The numbers of matched packets are given in Table 4.1. We use these traces in Section 4.4 to analyse the single-hop delay components.

4.3.2 Representativeness of the Data

Our traces of matched packets provide us with complete information about the path between a specific incoming and a specific outgoing link. We have records of the arrival and departure times of each matched packet, as well as timestamps for all the other packets sharing the same incoming and outgoing link. We calculate the delays experienced by the matched packets. In this section, we investigate how representative those delay results are for the rest of the traffic flowing on the same monitored links.

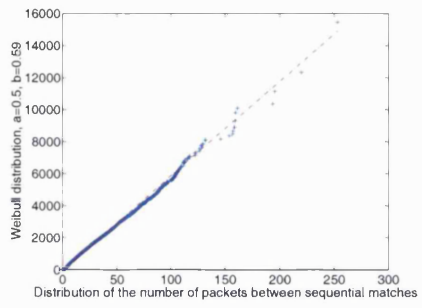
The matched packets form a subset of the traffic on the output link; matched packets in `set1` and `set2` constitute 0.5% and 2.4% of the total packets on links `out1` and `out2` respectively. Similarly, matched packets in `set3` and `set4` constitute 1.5% and 4.2% of the total packets on links `out3` and `out4`. Although this subset results from a single input port, it will be equivalent to a pure random sampling if the matched packets on the output link are geometrically distributed and independent [95].

We first analyse the distribution of the distance between matched packets in terms of packet counts. We find that it fits a Weibull distribution[2]. Figure 4.5(a)-(d) shows the Quantile-Quantile plot of the distribution of the number of packets between sequential matches and a Weibull distribution with a given b parameter for all four data sets. A shape parameter of $b = 1$ makes the Weibull distribution coincide with the exponential distribution, and indicates a pure random sampling (as the discrete equivalent of the continuous exponential distribution is a geometric distribution). If b is close to 1, a sample set is not purely random, but is close to random with occasional large gaps between matched packets.

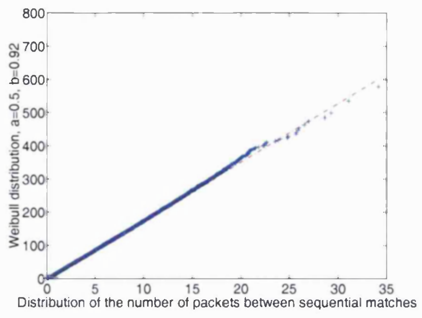
Our two first data sets exhibit $b = 0.59$ and $b = 0.92$, respectively. The inter-packet distribution of `set2` is therefore very close to an exponential distribution, while for `set1` is not. Similarly, for the third and fourth data sets the b parameter is found to be equal to 0.94, and 0.7 respectively. Analysis of the distribution of the distance between two matched packets in terms of time yields similar results (Figure 4.5(e)-(h)).

We further look into the sample autocorrelation function (ACF) of the inter-packet distance to investigate any correlation. In Figures 4.6(a)-(d) we present the ACF when the distance between matched packets is measured in packets. In Figures 4.6(e)-(h) we present the ACF when the distance between matched packets is expressed as the difference in their timestamps at the output link, measured in μs. The y-axis on all the figures has been scaled such that

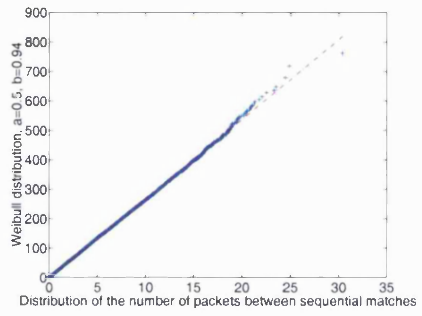
[2]The probability density function of a Weibull distribution is given by $f(x) = \frac{bx^{b-1}}{a^b}e^{-(\frac{x}{a})^b}$, with $a > 0,\ b > 0$; α is called the scale parameter, while b is called the shape parameter.



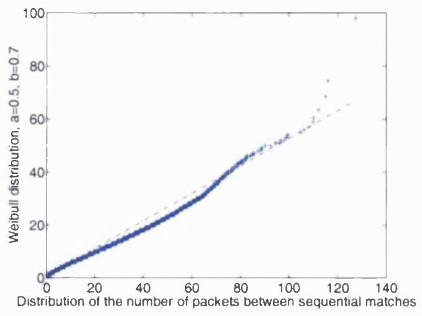
(a) `set1` ($b = 0.59$)



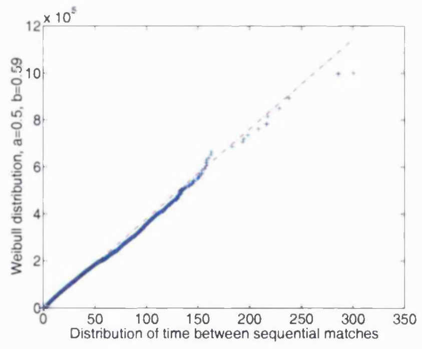
(b) `set2` ($b = 0.92$)



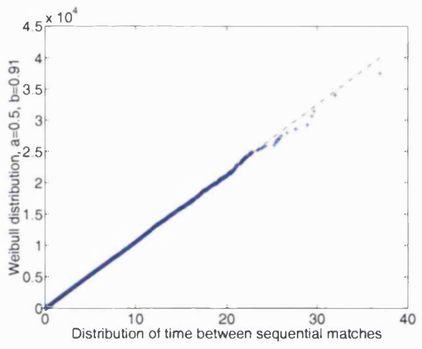
(c) `set3` ($b = 0.94$)



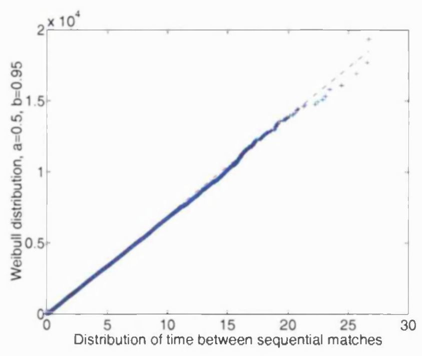
(d) `set4` ($b = 0.7$)



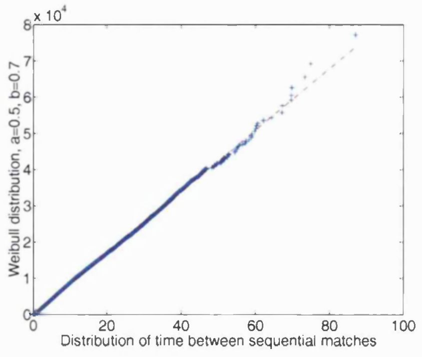
(e) `set1` ($b = 0.59$)



(f) `set2` ($b = 0.91$)

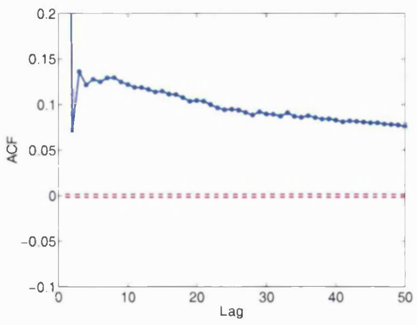


(g) `set3` ($b = 0.95$)

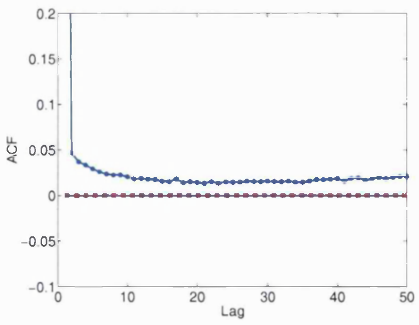


(h) `set4` ($b = 0.7$)

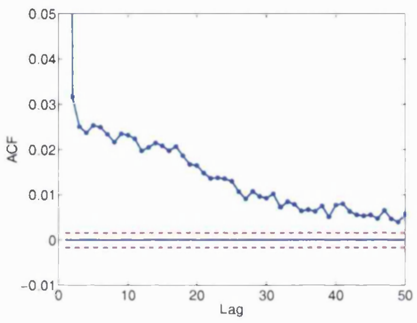
Figure 4.5: QQ-plot of the distribution of the number of packets and amount of time between two sequential matched packets in the output trace *out* (x-axis) versus a Weibull distribution.



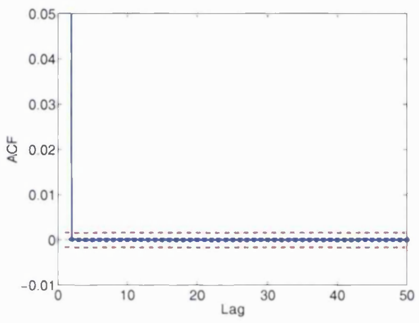
(a) set1



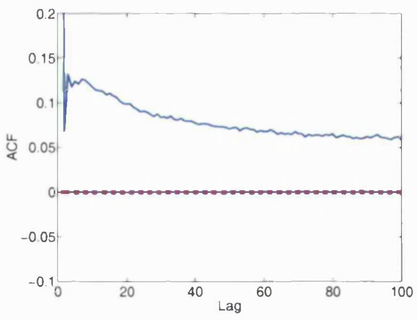
(b) set2



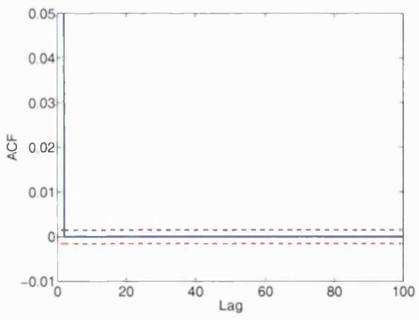
(c) set3



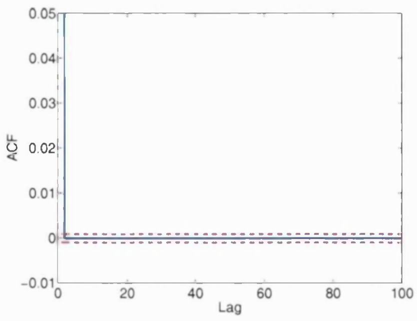
(d) set4



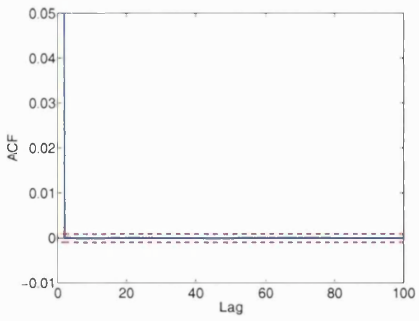
(e) set1



(f) set2



(g) set3



(h) set4

Figure 4.6: ACF of the number of packets and amount of time between two sequential matched packets in the output trace *out*. In Figures (a)-(d) the lag is expressed in number of packets, and in Figures (e)-(h) the lag is expressed in the number of μs of difference between the timestamps of the matched packets. The dashed lines denote the 99% confidence intervals.

comparison with the 99% confidence interval is feasible. From Figure 4.6, we see that both in time and number of packets, the ACF of `set1` exhibits significant correlation. The other three data sets exhibit much less correlation, but some correlation still exists at the lag of 50, when distance is measured in packets. Once distance between matched packets is measured as the difference in their timestamps, then `set2`, `set3` and `set4` show no correlation. Thus, although the distribution of the inter-packet distance in the second and third data sets is close to an exponential distribution (Figure 4.5), it is difficult to assess how close our sample is to a pure random sample, due to some correlation structure in the data set.

In summary, `set1` and `set4` cannot be considered as pure random samples of the queueing behaviour at the output link, while the two other data sets are very close. However, because we have a large number of samples, we can consider our data sufficient for studying the queueing behaviour. Moreover, in the case of the second and third data sets, our conclusions will be very close to the complete queueing behaviour at the output link.

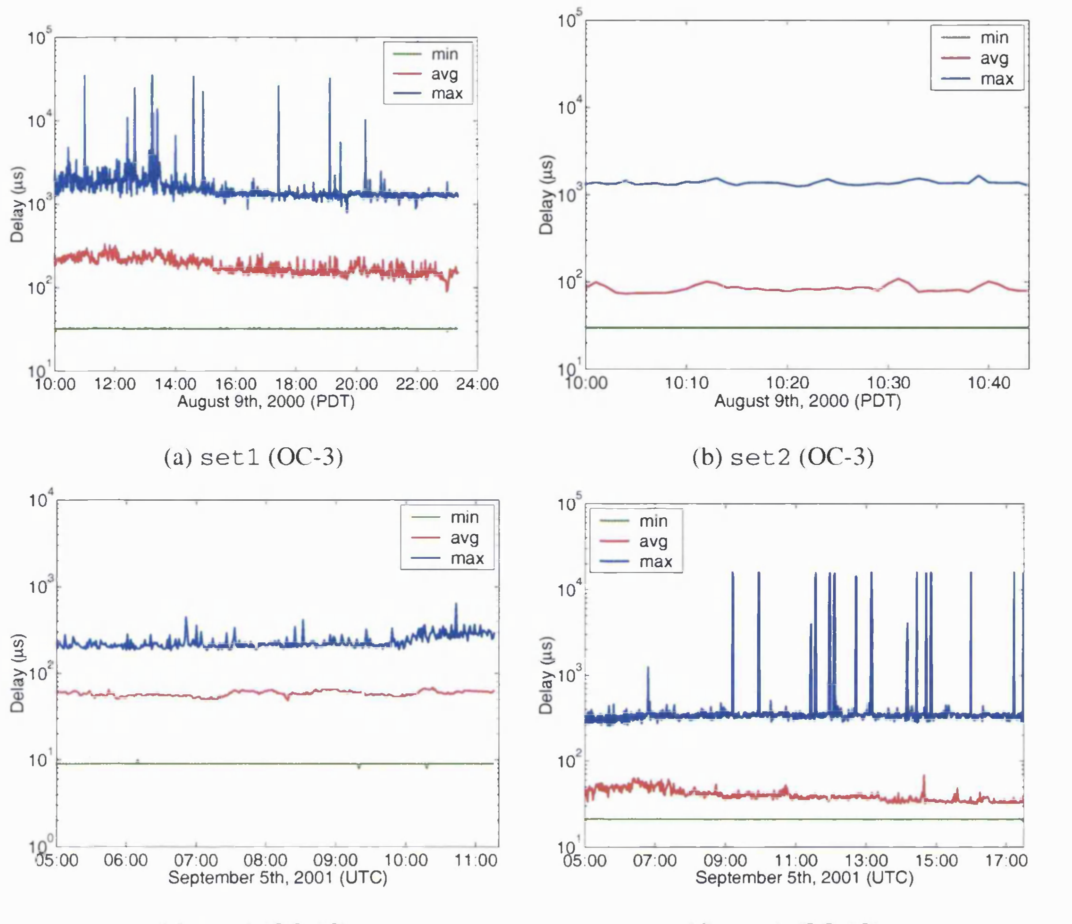
4.4 Delay Analysis

We start with general observations on the delay measurements. We plot the empirical probability density function of the measured single-hop delay, and quantify step-by-step its contributing factors. The outcome of this step-by-step analysis is the derivation of the output queueing delay, which is analysed in Section 4.5.

4.4.1 General Observations

We denote the m-th matched packet as m, and the total number of matched packets for a given set by M. Figure 4.7 plots the minimum, average, and maximum values of the single-hop delay $\{d(m)\}$ across each one minute interval for all four data sets. We observe first that the minimum delay is stable throughout all traces, while the average delay exhibits more oscillations and may drop as the link utilisation decreases towards the evening. The minimum delay corresponds to the minimum amount of time a packet needs to go through a router. Therefore, given that the minimum delay is constant throughout the day, there is at least one packet that experiences no queueing in each one minute interval.

The maximum delay is more variable than the average delay. It shows occasional spikes of a few milliseconds reaching up to 35 ms for `set1` and 172 ms for `set4`. We also note that the maximum delay remains consistently above 1 ms for the OC-3 data sets, and 0.2 ms for the OC-12 data sets, even though the average delay decreases. We provide possible explanations in Section 4.4.4.



(a) `set1` (OC-3) (b) `set2` (OC-3)

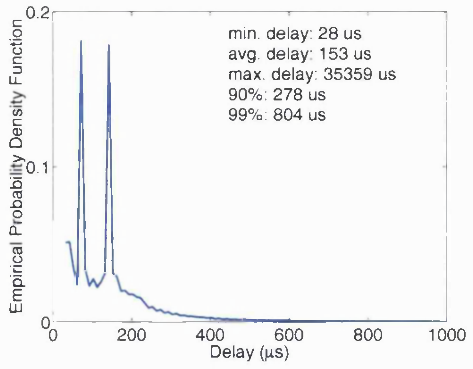
(c) `set3` (OC-12) (d) `set4` (OC-12)

Figure 4.7: Minimum, average, and maximum delay per minute for the matched packets of all four data sets. The x-axis is adjusted to the duration of the data set. The y-axis spans between 10 μs and 100 ms for `set1`, `set2`, and `set4`. For `set3` the y-axis spans between 1 μs and 10 ms, given that delays are much lower than in the other three data sets.

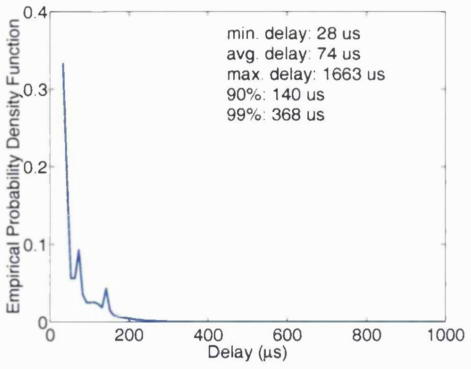
4.4.2 Step-by-Step Analysis of the Single-Hop Delay

Figure 4.8 presents the empirical probability density function of $\{d(m)\}, 1 \leq m \leq M$, along with brief statistical data on the upper right corner of each plot. Average delay values are around 100 μs for the OC-3 data sets and show a four times decrease when the link speed increases to OC-12. We see that 99% of packets experience less than 1 ms of delay on OC-3 links. For the OC-12 traces the 99th percentile of the single-hop delay distribution is below 100 μs. However, the maximum delay observed is data set specific reaching up to 35 ms in `set1` and 172 ms in `set4`.

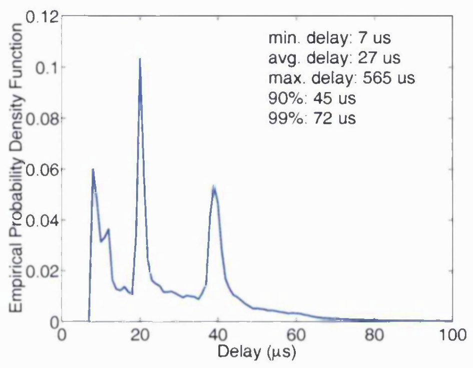
There are three distinct peaks at the beginning of each density function. Previous work by Thompson et al. reports that packets in the backbone do not have a uniform size distribution,



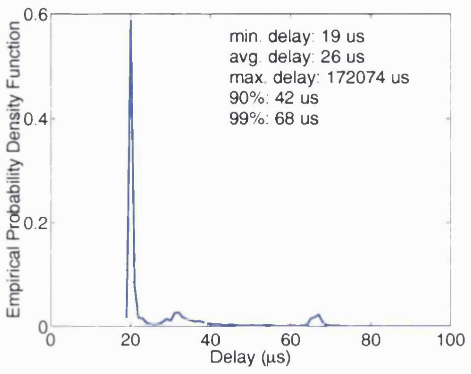
(a) set1 (OC-3)



(b) set2 (OC-3)



(c) set3 (OC-12)



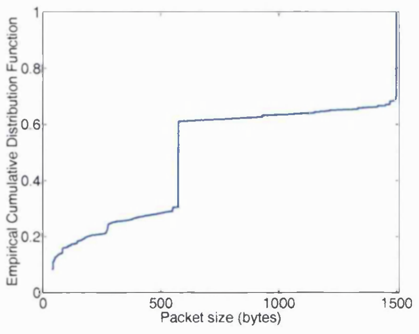
(d) set4 (OC-12)

Figure 4.8: Empirical probability density function of delay of matched packets $\{d(m)\}$.

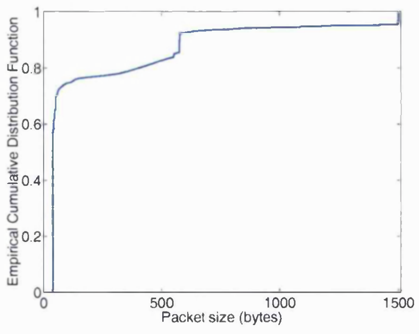
but instead have three unique peaks at 40 to 44, at 552 to 576, and at 1500 bytes [64]. The sizes of 40 and 44 bytes correspond to minimum-sized TCP acknowledgement packets and *telnet* packets of a single key stroke; 552 and 576 byte packets correspond to default MTU sizes when path MTU discovery is not used by a sending host; and 1500 byte packets correspond to the Ethernet MTU size. In all data sets, more than 70% of the packets are 40, 576, and 1500 bytes long (Figure 4.9). We thus conjecture the three peaks at the beginning of the delay distribution to be related to the packet size. To verify this conjecture, we group the packets of those three sizes for set1, and separately plot the empirical probability density functions of the delay experienced by packets of the given size in Figures 4.10(b)-(d). Each distribution has a unique peak that matches one of the three peaks in Figure 4.10(a). We now identify and quantify the factors that contribute the same amount of delay for packets of the same size.

Transmission Delay on the Output Link

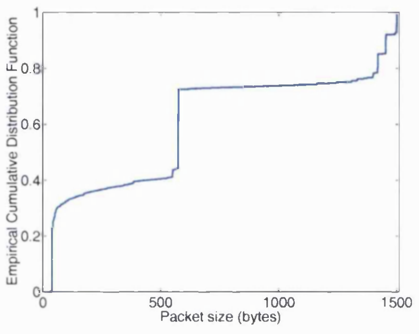
A first cause is the *transmission delay* on the output link. Transmission delay is proportional to the packet size and to the speed of the output link: l_m/C_{out}, where l_m is the length of the m-th



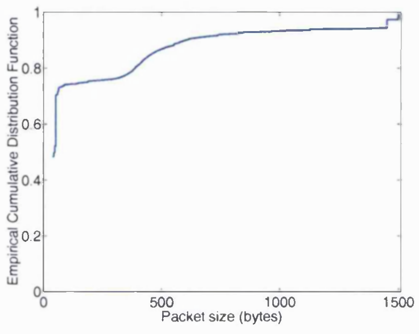
(a) set1 (OC-3)



(b) set2 (OC-3)



(c) set3 (OC-12)



(d) set4 (OC-12)

Figure 4.9: Packet size distribution for all four data sets.

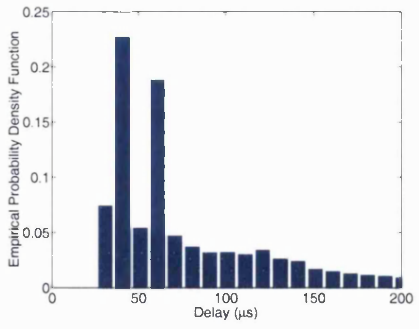
matched packet, and C_{out} is the output link capacity[3]. We refer to the difference between the total delay of packet m and its transmission time on the output link as the *router transit time*, denoted by $d_{tx}^{-}(m)$: $d_{tx}^{-}(m) = d(m) - l_m/C_{out}$. The empirical probability density function of $d_{tx}^{-}(m)$ is plotted in Figure 4.11.

There still are three distinct peaks in the distribution, even though they are less pronounced than in Figure 4.8. This may indicate that there is still a part of the router transit time that depends on the packet size.

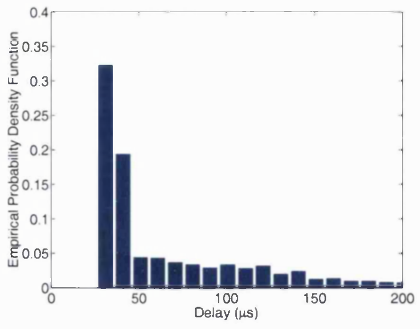
Minimum Router Transit Time

As described in 4.2.2, when a packet arrives at a router, its destination address is looked up in the forwarding table, and the appropriate output port is determined. The packet is then broken down into the appropriate number of cells, those cells are transmitted across the crossbar, and the packet is reassembled at the output port. These latter operations impose a minimum amount of delay on *every* packet, proportional to its size, which is likely to explain those remaining peaks in Figure 4.11. Below we quantify the minimum router transit time experienced by packets in our data sets.

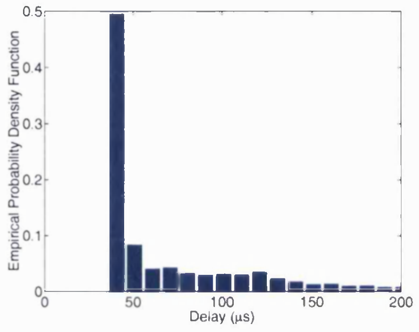
[3]Throughout this chapter, for the OC-3 traces we set $C_{out} = 150.336\ Mbps$, which is the effective payload of PoS OC-3. For the OC-12 traces $C_{out} = 601.344\ Mbps$.



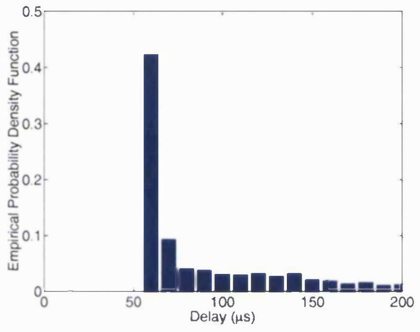
(a) set1.



(b) Only 40 byte packets of set1.



(c) Only 576 byte packets of set1.



(d) Only 1500 byte packets of set1.

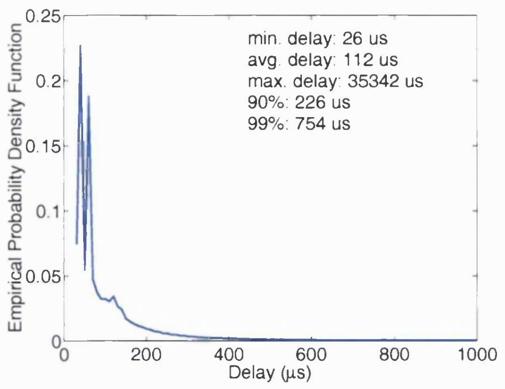
Figure 4.10: Single-hop delay distribution for the dominant packet sizes of set1.

We plot the minimum router transit time per packet size L, $d_{min}(L)$, in Figure 4.12.

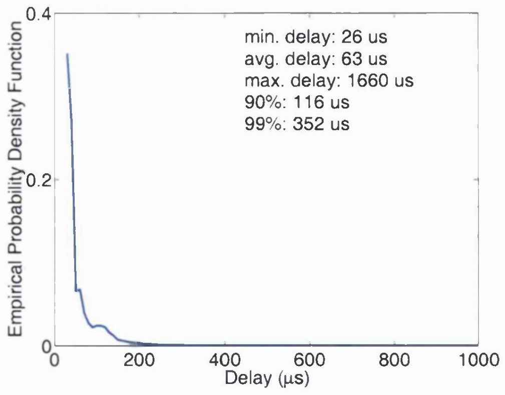
$$d_{min}(L) = \min_{1 \leq m \leq M} \{d_{tx}^{-}(m) | l_m = L\}$$

Figure 4.12 indicates that there exists a linear relationship between the two metrics. Moreover, we clearly observe the effect of the crossbar cell size on the obtained minimum router transit time. Notice that packets with different packet size in set1, set2, and set4 may face the same minimum router transit time. Their maximum difference in size is 53 bytes, which is the size of a cell. All the packets that get broken into the same number of cells face the same minimum delay while being transmitted from the input to the output port of the router. A similar effect can be witnessed for set3, that does not correspond to delay measurements through the crossbar switch. This effect is due to the granularity of our delay measurements, which is 1 μs, as well as possible differences in memory access time for small and large packets.

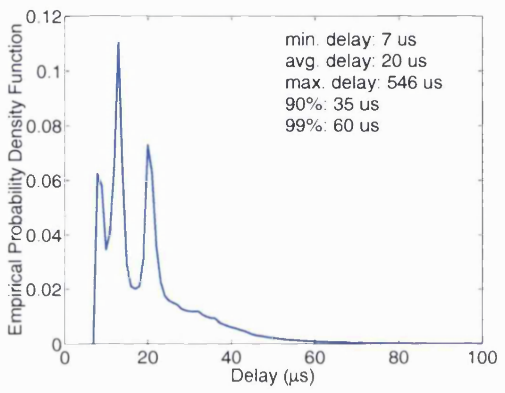
In order to derive the linear relationship between the minimum router transit time and the packet size we proceed as follows. Given that all data sets feature an order of magnitude more packets for the size of 40, 576, and 1500 bytes, those three packet sizes are more likely to provide us with accurate minimum values for the router transit time. For this reason, we rely on the measurements for these three packet sizes in linear regression, and obtain Equation (4.1).



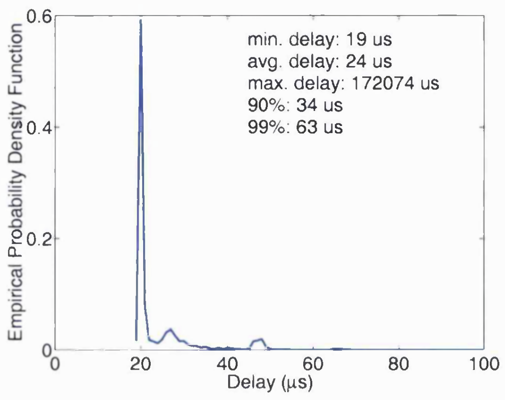
(a) `set1` (OC-3)



(b) `set2` (OC-3)



(c) `set3` (OC-12)



(d) `set4` (OC-12)

Figure 4.11: Empirical probability density function of router transit time, $d_{tx}^{-}(m) = d(m) - l_m/C_{out}$.

$$d_{min}(L) = \begin{cases} 0.0213 \cdot L + 25, & \text{OC-3 on different linecards} \\ 0.0089 \cdot L + 7, & \text{OC-12 on same linecard} \\ 0.0192 \cdot L + 18, & \text{OC-12 on different linecards} \end{cases} \tag{4.1}$$

As seen in Figure 4.12, the linear relationship between minimum router transit time and packet size is consistent among the OC-3 data sets, and differs among the OC-12 data sets. Moreover, for `set3` and `set4` we need two different equations to express the relationship between the minimum router transit time and the packet size. The reason for that is that packets that are received and transmitted on the same linecard exhibit different behaviour compared to the packets that need to transit the switch fabric. From Equation (4.1), we can identify the effect that such features of the router architecture may have on the packet delay.

According to Equation (4.1), transmission of packets across different linecards for OC-3 and OC-12 rates (`set1`, `set2`, and `set4`) leads to similar values for the slope capturing the

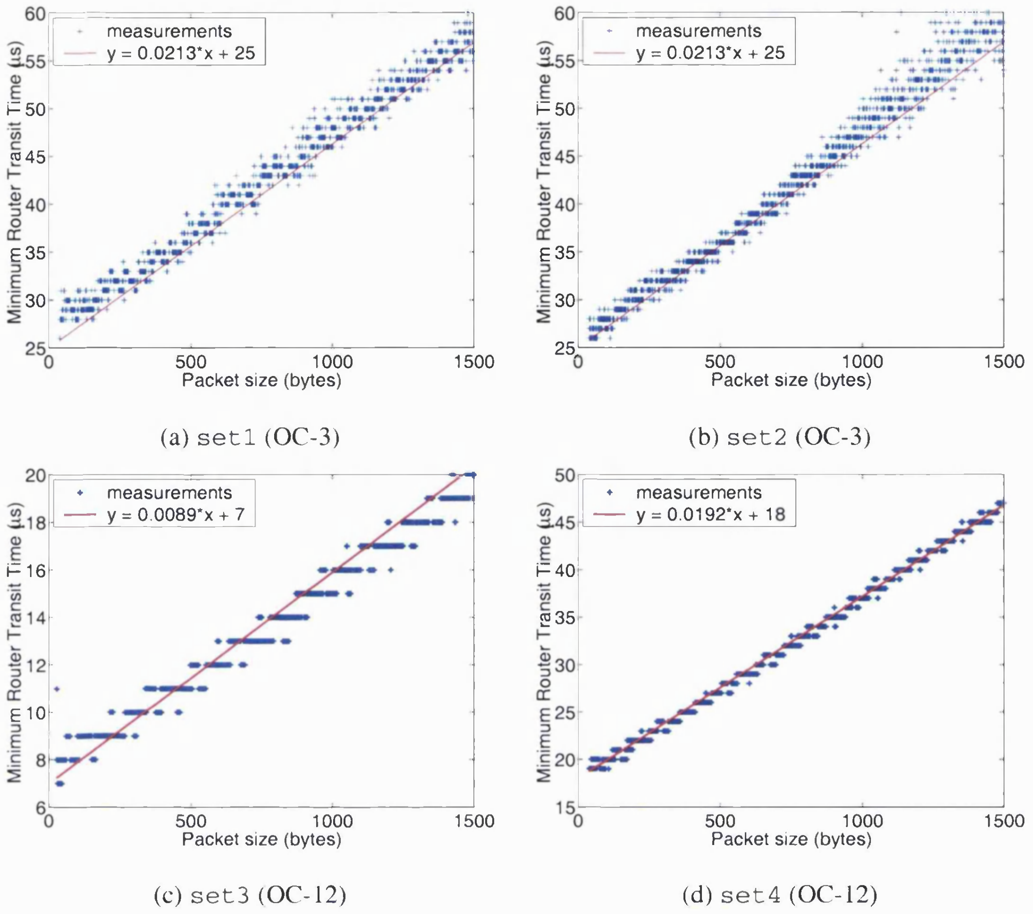
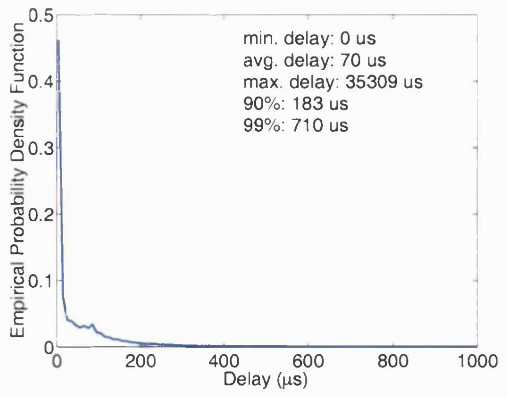


Figure 4.12: Minimum router transit time versus packet size L: $\min_m d_{tx}^{-}(m)$ for $m \in \{i|l_i = L\}$. Each figure contains one value, for the minimum router transit time, for each packet size observed in our data. Delay granularity is 1 μs.

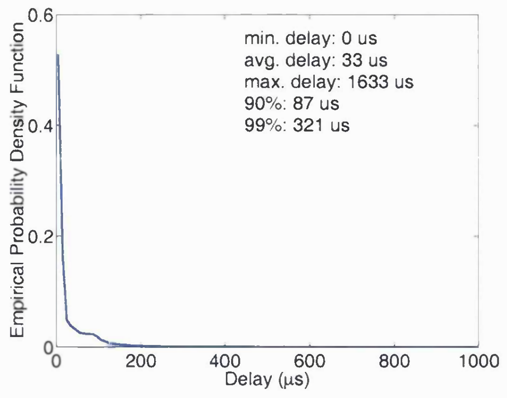
linear relationship between packet size and minimum router transit time. However, the constant term is different and larger for the OC-3 case. This could be attributed to the fact that set4 does not involve two quad linecards (Table 4.2), or that OC-12 linecards utilise more recent technology, and thus may be offering faster packet processing than OC-3 linecards.

One important result derived from Equation (4.1) and Figure 4.12 is that packets which remain in the same linecard are served much faster, i.e. in 7 to 20 μs. Packets that have to be transmitted across the switch fabric are served in 19 to 39 μs. Similar analysis performed on other data sets containing packets received and transmitted in the same quad linecard or across different linecards led to results consistent with Figure 4.12, and Equation (4.1).

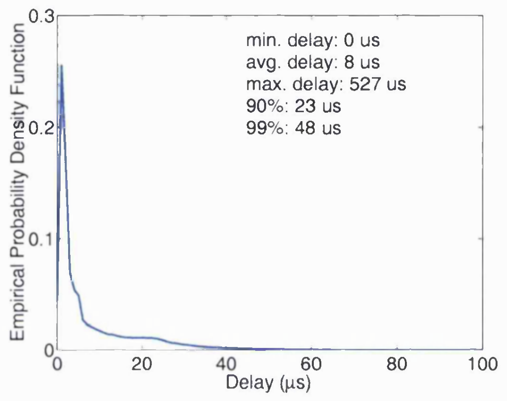
Subtracting $d_{min}(l_m)$ from the router transit time, $d_{tx}^{-}(m)$, we obtain the actual amount of time packets have to wait inside the router. The new empirical probability density function is presented in Figure 4.13.



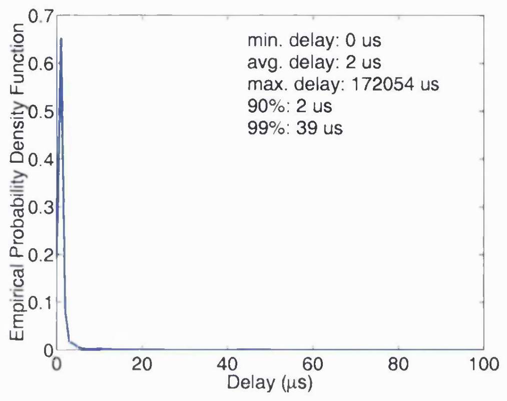
(a) `set1` (OC-3)



(b) `set2` (OC-3)



(c) `set3` (OC-12)



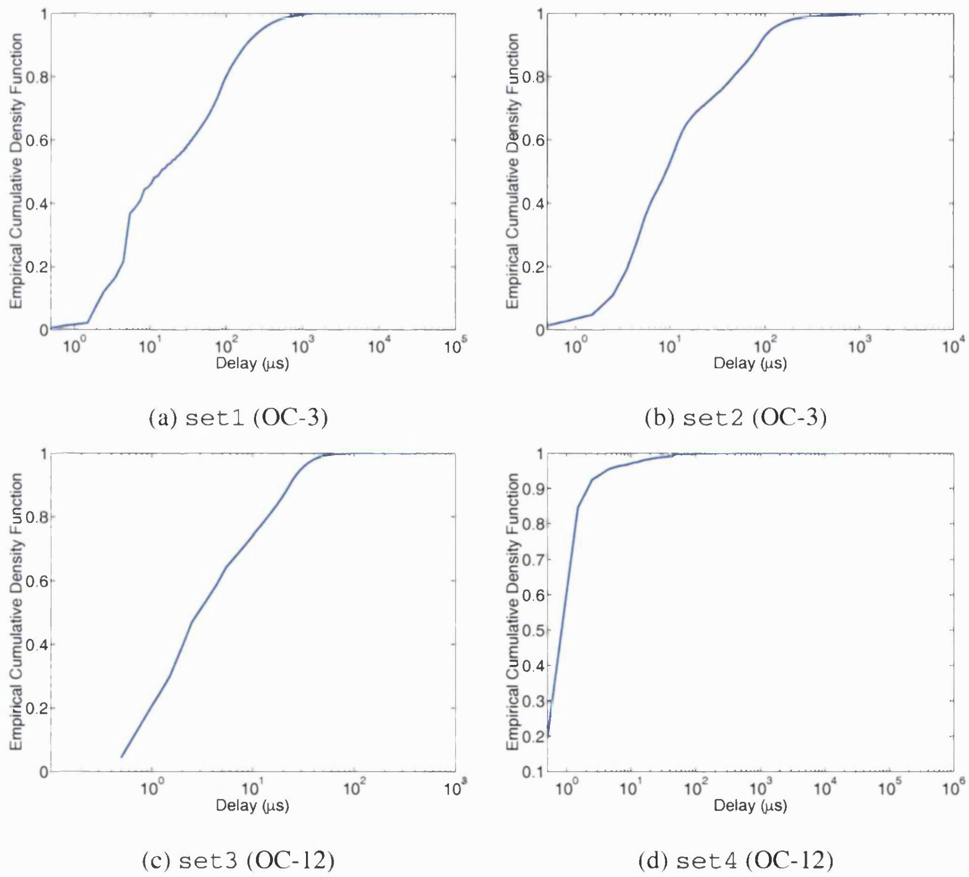
(d) `set4` (OC-12)

Figure 4.13: Empirical probability density function of $(d_{tx}^{-}(m) - d_{min}(l_m))$.

Packet size related peaks have now disappeared and the delay distributions look similar for all data sets. The distribution is characterised by very low delays: 45% of the packets in `set1`, and almost 50% of the packets in `set2` experience zero queueing delay. For the OC-12 data sets 30% of the packets in `set3`, and 80% of the packets in `set4` go through the router without any queueing at the output link. Differences in the average delay can be explained by the packet size distribution of the two sets: `set1` and `set3` are dominated by packets larger than 500 bytes, while `set2` and `set4` contain mostly 40 bytes packets (Figure 4.9). In addition, `set1` and `set3` consist of highly utilised links (Table 4.1), thus featuring higher queueing delay values than `set2` and `set4`. Small peaks around 100 μs for the OC-3 data sets, and 20 μs for the OC-12 data sets correspond to the transmission of a maximum sized packet at the respective line rate; thus accounting for the fact that a packet may arrive at the output queue, and find it occupied by another, possibly maximum-sized, packet. However, the 99th percentiles are very small; below 750 μs for the OC-3 data sets, and below 50 μs for the OC-12 data sets. As observed in Figure 4.13, the maximum delay is not consistent across data sets and reaches up to 172 ms for `set4` and 35 ms for `set1`.

4.4.3 Possible Causes for Very Large Delay

In Figure 4.14, we present the cumulative distribution function (cdf) for the output queueing delay $((d_{tx}^{-}(m) - d_{min}(l_m)))$ observed for all four data sets. A key observation is that across all data sets the tail of the delay distribution is very long, accounting for the presence of very large delays. However, an examination of the output link data when the very large delays were observed shows that the link was not fully utilised while those packets were waiting. Therefore part of the long delays is not due to queueing at the output link. Those delays could be due to input or output blocking at the switch fabric, as described in 4.2.2, or router design idiosyncrasies. In the remainder of this section, we look into possible explanations for these large delay values.



(a) `set1` (OC-3) (b) `set2` (OC-3)

(c) `set3` (OC-12) (d) `set4` (OC-12)

Figure 4.14: Empirical cumulative density function of $(d_{tx}^{-}(m) - d_{min}(l_m))$.

One possible reason for these very long delays through a single router could be that the monitoring systems lose synchronisation. We exclude measurement equipment fault as a cause for large delays for the following reasons. If the two measurement systems had gone out of time synchronisation, the minimum and average delay in Figure 4.7 would exhibit a level shift over time, which is not visible. There is no way to tell if the system's software had a bug, and produced the very large delays. However, it is extremely unlikely that a software bug affected only a handful of packets, still maintaining the strictly increasing nature of timestamps and keeping the minimum packet delay constant, both of which we checked in our traces.

A second reason, that can be easily verified, is that the packets experiencing long delays contain IP options. Most routers are designed to optimise the performance for the majority of packets. IP packets with options require additional processing in the protocol stack, and travel through a slower, software path than packets without options. IP option packets are present in `set2`, `set3`, and `set4`. In Table 4.3, we include the main statistics of the delay distribution derived from packets carrying IP options. Results indicate that packets with IP options spend at least 36 μs inside the router, and they usually account for the maximum delay in our single-hop delay distributions. The derived statistics should only serve as an indication for the magnitude of delay that packets with IP options may face while traversing a router, since the observed sample is too small to allow for generalisation. Given that delay measurements for packets carrying IP options are not solely due to queueing, we do not include them in the remainder of our analysis.

	`set2 (9 matches)`				
(μs)	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	242	453	1,145	1,659	1,659

	`set3 (21 matches)`				
(μs)	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	36	225	367	438	438

	`set4 (39 matches)`				
(μs)	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	270	11,219	10,629	172,074	172,074

Table 4.3: Delay statistics for the packets in the data sets that carry IP options.

Once IP option packets have been removed from our data sets, we find that the maximum delay for `set2` and `set4` drops significantly (Table 4.4). Due to the fact that there is a very small number of IP options packets present in our measurements, none of the other statistics of the distribution have significantly changed. Packets carrying IP options are capable of justifying the maximum delay in our data sets, but even after their removal the maximum delay experienced by packets in `set1` and `set4` remains in the order of tens of milliseconds. Other potential sources for non-queueing delays are: (i) routers stopping forwarding packets for a short period of time when busy with some other CPU intensive task, (*e.g.* routing table updates, SNMP requests, and garbage collection in memory), an effect usually referred to as a "coffee break", (ii) router interface cards with multiple ports or backplane switch fabrics that could allow blocking, (iii) memory locks or poor scheduling, etc.

	original set 2,781,201 matches					non IP-options set 2,781,201 matches				
in1 to *out1* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d^-_{tx}(m)$	26	112	226	754	35,342	26	112	226	754	35,342
$d^-_{tx}(m) - d_{min}(l_m)$	0	70	183	710	35,309	0	70	183	710	35,309
	original set 1,175,665 matches					non IP-options set 1,175,656 matches				
in2 to *out2* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d^-_{tx}(m)$	26	63	116	352	1,660	26	63	116	352	1,547
$d^-_{tx}(m) - d_{min}(l_m)$	0	33	87	321	1,633	0	33	87	321	1,520
	original set 17,613,183 matches					non IP-options set 17,613,162 matches				
in3 to *out3* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d^-_{tx}(m)$	7	20	35	60	546	7	20	35	60	546
$d^-_{tx}(m) - d_{min}(l_m)$	0	8	23	48	527	0	8	23	48	527
	original set 70,423,140 matches					non IP-options set 70,423,101 matches				
in4 to *out4* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d^-_{tx}(m)$	19	24	34	63	172,074	19	24	34	63	16,091
$d^-_{tx}(m) - d_{min}(l_m)$	0	2	2	39	172,054	0	2	2	39	16,045

Table 4.4: Statistics for the OC-3 and OC-12 data sets after the removal of packets with IP options.

4.4.4 Filtering Based on a Single Output Queue Model

When packets arrive at a router, they contend for resources to be forwarded to the destination output interface. The router can use various policies to resolve this contention. The FIFO (First-In First-Out) output queue model captures the essence of how a router should serve packets contending for the same resource in a best-effort fashion [96]. Thus, we model an output port of a router as a single output queue. While a single output queue is not an accurate model of all the operations performed in the router, it is sufficient to allow us to determine if the delay of a packet is due to output queueing or not, using *only* the measurements we have at our disposal.

In the routers deployed in our network packet processing is heavily pipelined so that a packet experiencing the minimum router transit time should not introduce extra queueing for the next packet arriving at the input port. That is, the minimum router transit time simply delays a packet's arrival time at the output queue, without affecting other packets. We can thus assume that the mth packet arrives at the output queue at $T'_{in}(m) = T_{in}(m) + d_{min}(l_m)$, and set the service rate of the single output queue to the transmission rate of the output link, as illustrated in Figure 4.15.

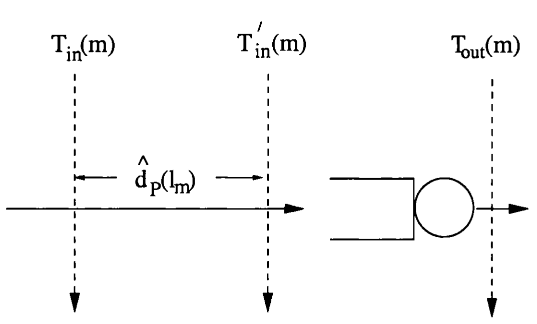
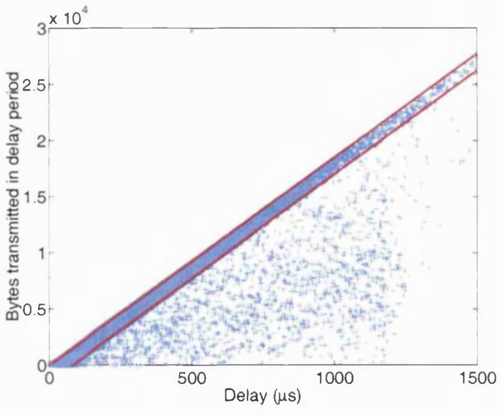
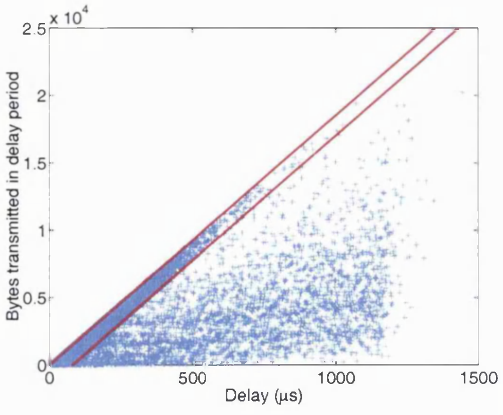


Figure 4.15: Single output queue model of a router.

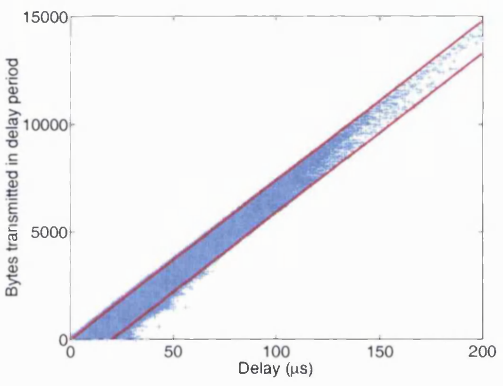
We expect a packet to wait at the output queue *if and only if* the output queue is busy serving other packets. The waiting time of a packet is $T_{out}(m) - l_m/C_{out} - T'_{in}(m)$. In Figure 4.16 we plot the number of bytes transmitted at the output link *out* during the time interval of $[T'_{in}(m), T_{out}(m) - l_m/C_{out}]$ versus the size of the interval for all four data sets. The top line corresponds to the case when the number of bytes transmitted between the packet's arrival and departure time ($T'_{in}(m)$ and $T_{out}(m) - l_m/C_{out}$) is *equal* to the number of bytes that would be transmitted if the link continuously sent packets at line rate. We observe that all data points lie below this top line. Moreover, most of the points that fall off this line are bounded by another line below, of the same slope, which allows for the transmission of one less maximum-sized packet. This latter line is described by $y = (C_{out} \cdot x)/8 - 1500$, where x is the size of the time interval, and y is the number of bytes. We allow one maximum-sized packet as the error margin in our waiting time calculation, since the accuracy of the timestamps, the non-uniform distribution of SONET overhead in the signals, and the uncertainty about operations inside the router are likely to affect our computation. Those packets whose waiting times lie between the two lines are interpreted as follows: while a matched packet is waiting to be transmitted between $T'_{in}(m)$ and $T_{out}(m) - l_m/C_{out}$, the output link is fully utilised. We consider as the *filtered* data set those packets that lie between the two bounding lines in Figure 4.16. For set1,



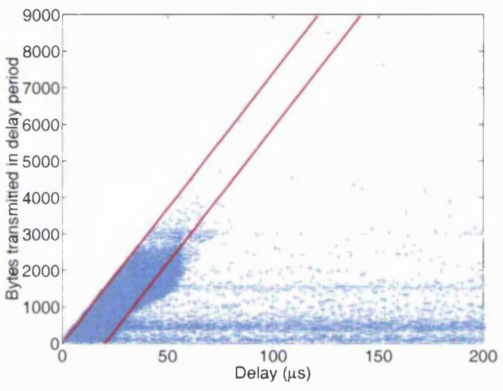
(a) `set1` (6.6% filtered out)



(b) `set2` (2.5% filtered out)



(c) `set3` (0.0009% filtered out)



(d) `set4` (1% filtered out)

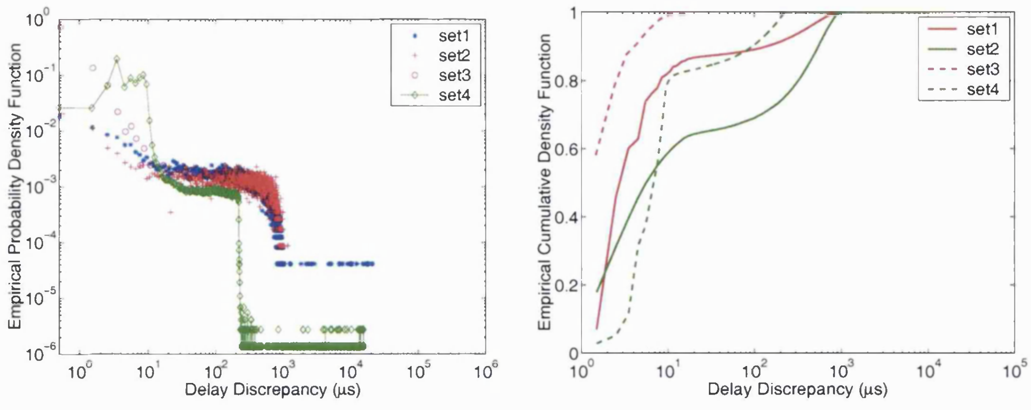
Figure 4.16: Number of bytes transmitted between $T_{in}^{'}(m)$ and $T_{out}(m) - l_m/C_{out}$ on `out1`, `out2`, `out3`, and `out4`.

for instance, the filtered set contains 93.4% of the total number of packets in the set. Other packets are considered to have experienced delay not due to output queueing[4] beyond the error margin and are filtered out. To evaluate the magnitude of the delay values that get filtered out by our simple output queue model, we proceed as follows.

We compute the amount of delay that each packet should have experienced in all four data sets according to the observed output link utilisation. We then subtract the computed delay from the actual delay measured. The difference between those two values corresponds to the amount of additional delay that a packet experienced inside the router. In Figure 4.17(a) and 4.17(b), we present the empirical probability and cumulative density function for the difference in delay experienced by the set of packets that got filtered out.

Figure 4.17(a) shows that the part of our delay measurements, that cannot be explained

[4]Strictly speaking, transmission and propagation delays are not due to queueing. However, we limit the use of non-queueing delay only to the delay experienced at the output queue due to congestion.



(a) Empirical Probability Density Function (log-log)

(b) Empirical Cumulative Density Function (log-normal)

Figure 4.17: Unexplained part of the delay in the measurements that get filtered out.

according to our single output queue model, may reach up to tens of milliseconds. An important observation is that for `set1` and `set2` the empirical probability density function shows a plateau between 10 μs and 1 ms. For `set4`, the plateau area spans between 10 and 200 μs. Such a behaviour is consistent with a "coffee break" effect. When a router goes into a "coffee break", it stops forwarding packets for the duration of the break. Therefore, all packets that arrive at the router during this period have to wait until the router resumes the forwarding process, and therefore experience delays that may reach the duration of the break. In our case, the observed behaviour resembles a 1 ms "coffee break" in our OC-3 measurements, and 200 μs in our OC-12 measurements. What is interesting is that no such effect is evident for `set3`, where the maximum delay difference is limited to 17 μs. Recall that `set3` corresponds to delay measurements taken inside the same quad-OC-12 linecard. Therefore, such a finding could be an indication that the "coffee break" effect is not taking place at the linecards themselves.

Unfortunately, seeking explanation for such a phenomenon requires detailed knowledge of the router architecture, and constitutes proprietary information. Therefore we can only conjecture about possible reasons behind this behaviour. Justification for the existence of delay discrepancies larger than 1 ms is even harder to provide. Queueing taking place at the input link and contention for access to the switch fabric could be possible explanations, but cannot be verified solely based on our measurements. In any case, as can be seen from Figure 4.17(b) such a phenomenon affects a very small number of packets, namely between 20% and 40% of the filtered out packets. This percentage corresponds to less than 1% of the total number of packets in each data set [5].

[5]The final percentage of packets that get filtered out is higher than 1% because of small delay discrepancies,

Given that delays experienced by packets beyond our error margins are not related to the cross-traffic at the output link, we continue our analysis only with the filtered data sets. We summarise the statistics for the router transit time and queueing delay of filtered and not filtered packets in Table 4.5. The average delay, the 90th, and the 99th percentiles of the filtered data sets are now lower than in the original data sets. Moreover, all of the delays larger than 5 ms in `set1` and `set4` have disappeared, and the maximum delay has dropped to 3.9 ms and 160 μs respectively. On the other hand, the maximum delay for `set2` and `set3` remains unchanged, indicating that the output link was fully utilised when the maximally delayed packet was being held back from transmission. We plot the minimum, average, and maximum values of the filtered delays for all four data sets in Figure 4.18. We compare to Figure 4.7 and notice that the maximum delay does not stay over 1 ms throughout the whole day. Consequently, the single queue model is effective in filtering the delays which are not due to queueing at the output link.

4.5 Analysis of Output Queueing Delay

4.5.1 Tail Behaviour

In this section, we analyse the tail of the queueing delay distribution. This analysis will help us identify possible models for the queueing delay in the backbone that could be exploited in simulation environments. We show that our results agree with previous analytical studies [87].

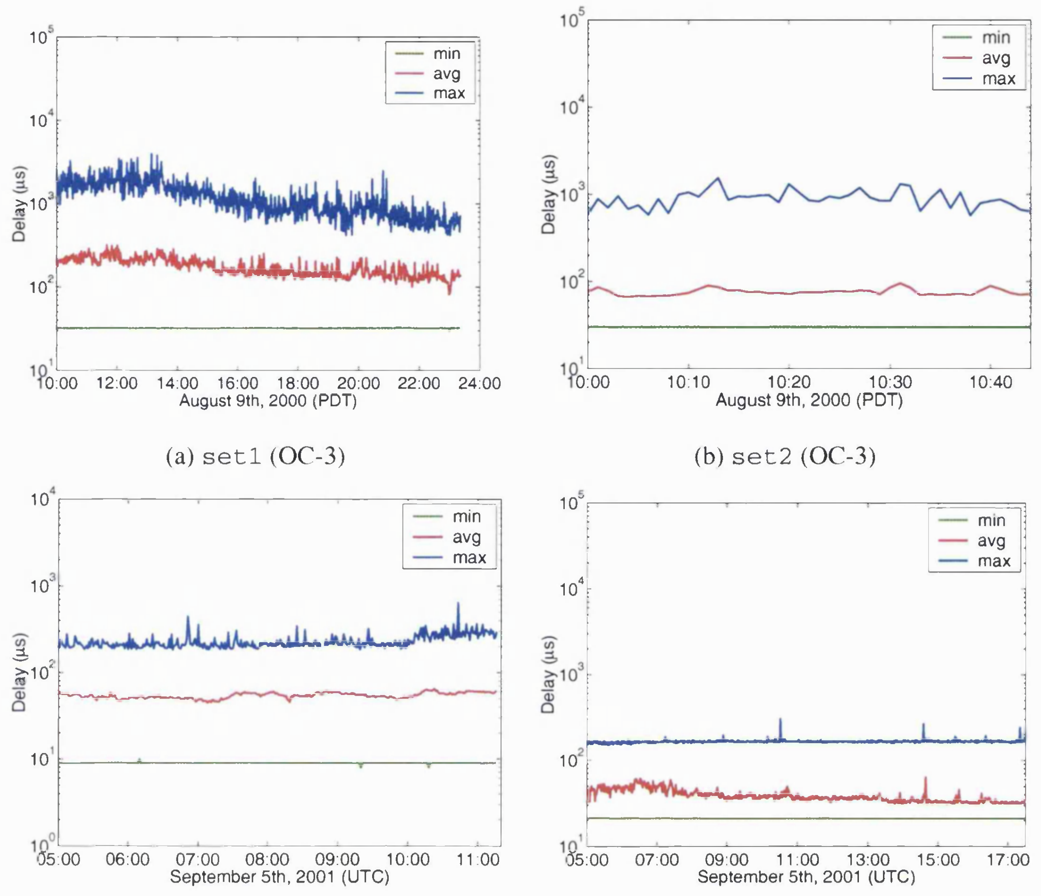
Tail behaviour can be categorised into three types: light tailed, long tailed, and heavy tailed. A *light tailed* distribution has a probability density function whose tail approaches zero more rapidly than the exponential distribution. A distribution is said to have a *heavy tail* if $P[X > x] \sim kx^{-a}$ as $x \to \infty$, $0 < a < 2$ [97]. This means that regardless of the distribution for small values of the random variable, if the asymptotic shape of the distribution is hyperbolic, the distribution is heavy tailed. The simplest heavy tailed distribution is the Pareto distribution which is hyperbolic over its entire range and has a probability mass function $p(x) = ak^a x^{-a-1}$, $a, k > 0$, $x \geq k$, where k represents the smallest value the random variable can take. *Long tailed* distributions are not strictly heavy tailed, but decay slower than exponential. Lognormal and Weibull distributions with the shape parameter $b < 1$ belong to long tailed distributions.

The network traffic is known to be long-range dependent, and such traffic can be modelled as Fractional Brownian Motion (FBM) [89]. Norros shows that the queueing delay distribution of the FBM traffic is approximated by a Weibull distribution [87].

below 10 μs.

	original set 2,781,201 matches					filtered set 2,596,486 matches (6.6% filtered out)				
in1 to *out1* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	26	112	226	754	35,342	26	106	217	603	3,937
$d_{tx}^{-}(m) - d_{min}(l_m)$	0	70	183	710	35,309	0	65	174	558	3,903
	non IP-options set 1,175,665 matches					filtered set 1,145,170 matches (2.5% filtered out)				
in2 to *out2* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	26	63	116	352	1,547	26	56	112	230	1,547
$d_{tx}^{-}(m) - d_{min}(l_m)$	0	33	87	321	1,520	0	27	82	200	1,520
	non IP-options set 17,613,183 matches					filtered set 17,613,018 matches (0.0009% filtered out)				
in3 to *out3* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	7	20	35	60	546	7	20	35	60	546
$d_{tx}^{-}(m) - d_{min}(l_m)$	0	8	23	48	527	0	8	23	48	527
	non IP-options set 70,423,140 matches					filtered set 69,710,887 matches (1% filtered out)				
in4 to *out4* (μs)	min.	avg.	90%	99%	max.	min.	avg.	90%	99%	max.
$d_{tx}^{-}(m)$	19	24	34	63	16,091	19	24	33	49	362
$d_{tx}^{-}(m) - d_{min}(l_m)$	0	2	2	39	16,045	0	1	2	16	160

Table 4.5: Statistics for the OC-3 and OC-12 data sets before and after filtering.



(a) `set1` (OC-3)

(b) `set2` (OC-3)

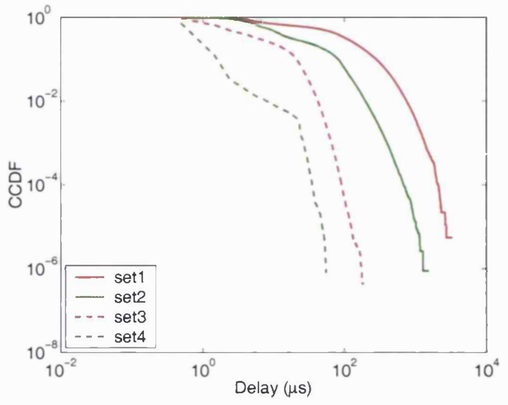
(c) `set3` (OC-12)

(d) `set4` (OC-12)

Figure 4.18: Minimum, average, and maximum single-hop delays per minute for the filtered packets of all four data sets. The x-axis is adjusted to the duration of the data set. The y-axis spans between 10 μs and 100 ms for `set1`, `set2`, and `set4`. For `set3` the y-axis spans between 1 μs and 10 ms, given that delays are much lower than in the other three data sets.

To examine what tail category our delay distributions fall into, we first plot the complementary cumulative distribution function (CCDF) of $(d_{tx}^{-}(m) - d_{min}(l_m))$ in log-log scale for the first hour of the filtered sets, where link utilisation remains approximately constant (Figure 4.19). If the tail of the CCDF forms a straight line, then the distribution may be heavy tailed. From Figure 4.19, it is not clear whether this is the case for our data sets. We use the `aest` tool to formally check if the queueing delay distribution is heavy tailed [98]. The obtained results indicate that our delay distributions do not have the power-law tail like the Pareto distribution, and are *not heavy tailed.*

We then look into whether our queueing delay distributions are long-tailed. As already mentioned, a Weibull distribution with a shape parameter b less than 1 belongs to the long-tailed



Filtered data set

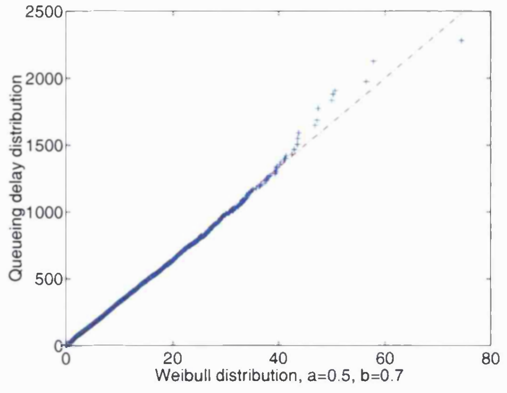
Figure 4.19: Log-log plot of CCDF for the queueing delay measured for all four data sets (data set 1, 2: OC-3, data set 3, 4: OC-12)

distributions. We fit a Weibull distribution to our queueing delay distributions, and present our results in Figure 4.20 for the first three data sets. `Set4` is omitted because it is characterised by very low queueing delays, leading to a very small number of samples in the tail of its distribution[6]. Figure 4.20 shows that the queueing delay distribution for `set1` and `set2` fits to a Weibull distribution with a shape parameter b equal to 0.6, and 0.7 respectively. The OC-12 distribution for queueing delay inside the same linecard (`set3`) can be approximated with a Weibull distribution with a shape parameter b equal to 0.82. Therefore, the distribution of queueing delay is *long tailed*, confirming the finding in [87].

4.5.2 Impact of Link Utilisation on Output Queueing Delay

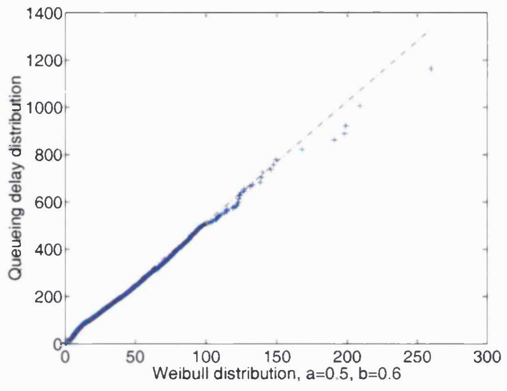
In this section, we investigate the evolution of queueing delay with respect to link utilisation in our backbone network, where link utilisation rarely exceeds 70%. Simple models, such as M/M/1 and M/G/1, fail to account for long-range dependence in the traffic, and are thus likely to predict smaller delays than the ones actually experienced. On the other hand, the Fractional Brownian Motion (FBM) model captures the characteristics of the observed traffic, but is mostly used in estimating the queueing delay for links which are utilised above 80%; a highly untypical operating region for backbone links. We use our measurements to study the effect of link utilisation on queueing delay, and to understand the delay guarantees that can hold

[6]The link utilisation of both links in `set4` is very low, equal to less than 10 Mbps out of the 622 Mbps of the link's capacity. As a consequence, the respective queueing delay distribution is characterised by a 99th percentile equal to 15 μs. This means that the number of samples we have in the tail of this particular distribution is very limited, and since they are bounded by a maximum queueing delay of 52 μs, they are very sensitive to our clock accuracy and our 1 μs granularity.



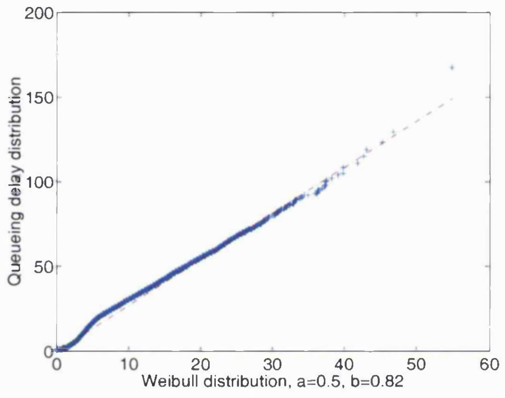
(a) set1 (OC-3)

with $b = 0.7$



(b) set2 (OC-3)

with $b = 0.6$



(c) set3 (OC-12)

with $b = 0.82$

Figure 4.20: Quantile-Quantile plot of the queueing delay distribution against a Weibull distribution (OC-3, OC-12).

inside a backbone network.

We segment each trace into 100 ms intervals. We calculate average link utilisation and average delay measured for each 100 ms interval. We proceed to correlate link utilisation with delay and we calculate the average delay seen for each level of link utilisation. In Figure 4.21, we present the average queueing delay versus link utilisation for both OC-3 data sets, and compare them with the M/M/1 and FBM models. The parameters of the models are estimated from the first data set. The M/M/1 takes as input the average link utilisation, the average packet size, and the capacity of the output link. The parameters for the FBM are estimated from the trace, and are equal to $m = 46.745\ Mb/sec, a = 350\ Kbit \times sec, H = 0.885$.

As can be seen from Figure 4.21, the M/M/1 model captures the trend of the relationship between queueing delay and link utilisation, but underestimates it. It is well known that the

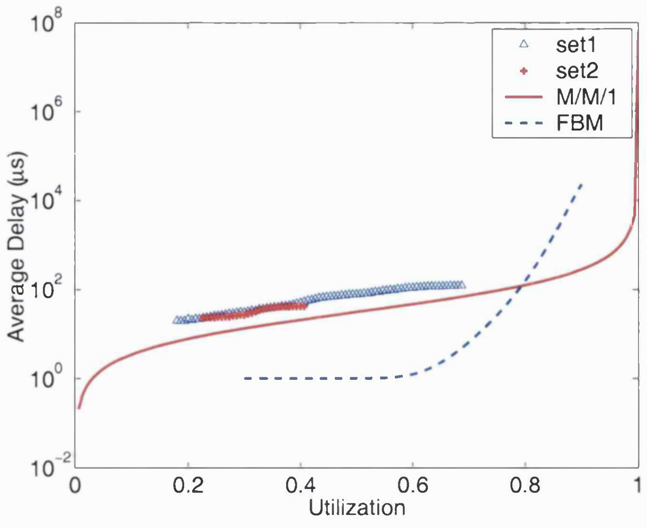


Figure 4.21: Comparison of the average queueing delay vs. link utilisation, derived from the OC-3 measurements, the M/M/1 model, and the FBM model.

FBM model is designed to capture the queueing behaviour at high link utilisations, while underestimating it at low and intermediate link utilisations. Indeed, FBM is performing poorly for link utilisations below 70%. Moreover, given that our links are never more than 70% loaded, we cannot compare the queueing delay predicted by the FBM in cases of high load, with actual measurements. The OC-12 measurements correspond to even lower link utilisation, never exceeding 55%, thus not shedding more light into what the effect of higher link utilisation may be. In summary, for the operating regions that a large network would choose to utilise its links at, both M/M/1 and FBM models underestimate the average queueing delay.

4.6 Conclusions

To the best of our knowledge, this work is the first to provide data about actual delays incurred through a single router in the backbone. We measure single-hop delay as experienced by packets in the Sprint IP backbone network. We develop a methodology to identify the contributing factors to the single-hop delay that is simple and applicable to *any* single-hop delay measurements. We demonstrate its applicability on OC-3 and OC-12 packet traces. In addition to packet processing, transmission, and queueing delays, we identify the presence of very large delays that cannot be explained within the context of a work-conserving FIFO output queue. We provide a simple technique to remove these outliers from our measurements, and offer possible explanations regarding the events that may have led to such extreme delays through a single node.

According to our results, 99% of the packets in the backbone experience less than 1 ms of delay going through a single router when transmitted at OC-3 speeds. At OC-12 line rates, the single-hop delay drops to less than 100 μs. After the extraction of the queueing delay component in our measurements, we show that the largest part of single-hop delay experienced by a packet is *not* due to queueing, but rather to the processing and transmission of a packet across the switch fabric. In addition, we observe a small number of packets (less than 1% in our measurements) that may experience significantly larger delays, either because they carry IP options or because they are affected by idiosyncratic router behaviour.

The analysis of the queueing delay distribution reveals that it is long tailed, and can be approximated by a Weibull distribution with a scale parameter $a = 0.5$, and a shape parameter $b = 0.6 \sim 0.7$ for transmission of packets across two different OC-3 linecards. If the forwarding of the packet does not include transmission across the switch fabric, then the queueing delay distribution can be approximated with a Weibull distribution with a higher shape parameter $b = 0.82$. These results confirm previous findings by Norros [87]. The measured average queueing delay is larger than predicted by M/M/1, and FBM models when the link utilisation is below 70%, but its absolute value is quite small. Identification and modelling of the several components comprising single-hop delay allows for more realistic backbone router models, that could easily be used in simulation environments.

In summary, packets in the Sprint IP backbone network experience edge-to-edge delays that are dominated by the propagation delay, and face minimal jitter. This result, though, should be evaluated within the context of Sprint's backbone design principles, that dictate moderate link utilisation across the network; in our measurement all links were utilised less than 70% even at a 10 ms time scale.

The work presented in this chapter is the result of collaboration with Sue Moon, Chuck Fraleigh, Patrick Thiran, and Christophe Diot. Chuck Fraleigh was mainly responsible for the IP monitoring systems. Sue Moon contributed in the methodology. Patrick Thiran provided help with the statistics, and Christophe Diot with the writing. However, the largest part of the methodological approach, all the code and the detailed analysis of the collected traces have been done by me. During the analysis of the collected information, certain issues with the monitoring cards and their synchronisation came up. These issues were mainly resolved by Chuck Fraleigh and me.

Chapter 5

A Pragmatic Definition of Elephants in Internet Backbone Traffic

In the previous chapter, we showed that very little queueing takes place in the monitored IP backbone network. This is consistent with the network design principles that dictate moderate link utilisations across the network. We concluded that in today's moderately utilised backbone networks, queueing delay at backbone routers adds insignificantly to the overall edge-to-edge delay.

However, our findings are only applicable to cases when link utilisation stays moderate. There will always be cases when unexpected link and equipment failures may lead to transient link overload. In those cases, when particular links fail, other links in the network will be selected by routing protocols to carry the affected traffic, and thus may reach prohibitive utilisation levels. Under such conditions, the current best practice is to increase the IS-IS/OSPF weight of the overloaded link and re-direct the traffic to other areas inside the network, rarely being able to predict the future network state without direct knowledge of the traffic matrix. In this chapter, we look into a controlled way of off-loading a link by the re-direction of selected destination network prefixes that carry the majority of the traffic persistently over time.

We propose a methodology for the classification of destination network prefixes on a link into two classes, namely the "elephants", that are limited in number and carry the majority of the load, and the "mice", that are large in number but carry insignificant amounts of traffic. We show that classifying flows based only on their bandwidth during a single interval leads to high volatility in the elephant class, i.e. the set of flows classified as elephants (or mice) differs significantly from one interval to the next. We conclude that for traffic engineering applications, that require persistence of the derived classification, one needs to also account for the temporal behaviour of the flows. We propose a new classification scheme that incorporates a metric, called *latent heat*, that tracks the evolution of the difference in elephant bandwidth and the class separation threshold over time. We demonstrate that the latter approach is capable of correctly identifying elephant flows that do not only contribute high volumes of traffic but are also more likely to maintain this property in time.

Required Network Measurements and Time Granularity

Identification of the elephant destination network-prefix flows on a link requires flow measurements at the granularity of the BGP network prefix. The results presented in this chapter are based on IPMON packet traces collected on specific links in the Sprint IP backbone network. The selected links correspond to sufficiently aggregated traffic, two hops away from the periphery of the network capturing traffic towards the core. Destination network-prefix flows are defined according to the collected BGP routing table. Bandwidth is computed at the time granularities of 1 minute, 5 minutes and 60 minutes.

5.1 Introduction

Traffic engineering of IP networks has attracted a lot of attention in the recent past. Network operators are seeking ways in which they could make optimal use of their networks' capacity, without introducing complexity and maintaining the simplicity of the packet switching paradigm. With that in mind, researchers have proposed different algorithms for traffic engineering tasks that could lead to better utilised networks by exploiting certain properties of the underlying traffic [99, 100, 101, 102].

Studies of the Internet traffic at the level of network prefixes, fixed length prefixes, TCP flows, AS's, and WWW traffic, have all shown that a small percentage of the flows carries the largest part of the information [99, 100, 101, 102]. This behaviour is known as "the elephants and mice phenomenon", and is considered to be one of the few invariants of Internet traffic [103]. In statistics, this phenomenon is also called the "mass-count disparity". The "mass-count disparity" states that for heavy-tailed distributions the majority of the mass in a set of observations is concentrated in a very small subset of the observations [104].

In order to be able to exploit such a property for traffic engineering purposes, one has to be able to identify which flows carry the majority of the bytes. The attraction of classifying flows into elephants and mice, is that by treating a relatively small number of flows differently one can affect a large portion of the overall traffic. For this to be practical though, elephant flows would need to remain elephant flows for reasonably long periods of time, so that a traffic engineering application need not change its policy or state frequently.

To the best of our knowledge no systematic way has been proposed so far for choosing the criterion that isolates the high-volume flows. This criterion should allow one to consider any particular flow and to decide if it should be categorised as an elephant or a mouse. Our aim is to separate out a small number of high-volume flows that persistently account for most of the traffic, from a very large number of flows that contribute negligibly to the overall load. In the rest of the chapter we refer to the former as the *elephant* class, and the latter as the *mouse* class. Note that previous definitions of elephant and mice flows on the Internet were based on arbitrary criteria [101, 102, 105] and did not incorporate this notion of persistence in time. They usually characterised flows as elephants or mice based on their instantaneous behaviour, and focused on the usefuleness of such a classification with respect to specific traffic engineering applications. The classification criteria were usually defined in terms of duration or bandwidth, and the separating threshold between the two classes was set in an ad hoc fashion. Therefore, our goal is twofold. Firstly, we want to define the criterion for the separation of the flows into two classes. Secondly, we want to verify that the proposed criterion leads to time persistence in the resulting classification.

Separation of the flows that persistently contribute the majority of the traffic could have several applications within a traffic engineering context. Some of these applications include multi-path routing, as suggested in [106, 107], the setup of MPLS tunnels throughout the network to accommodate elephant flows [102], accounting as proposed in [105], or selective re-routing of elephant flows. The latter application is envisioned to target the off-loading of congested links by the re-direction of elephant destination network prefixes to less congested parts of the network. The current best practice to overcome transient overload on a link in the network is the increase of the corresponding IS-IS/OSPF weight [25]. However, without specific knowledge of the network traffic matrix, this type of operation usually results in a network whose state can rarely be predicted. Therefore, in such cases, it may be preferable for the network operator to selectively re-direct specific destination prefixes away from the congested link, by changing the next-hop address for this particular prefix. In order for such a configuration change to have the desirable effect, the elephant flow which is going to be re-routed has to be selected among those network prefixes that not only contribute sufficient amounts of traffic, but that exhibit this property persistently over time.

In accordance to the above requirements, we define flows at the traffic aggregation of a destination network prefix, and we observe their bandwidth over fixed time intervals. Given that there is no systematic way proposed for the identification of elephant flows on a link based on their bandwidth, we first investigate a number of schemes to identify elephants based on measurements collected in a single interval. We call these schemes "single-instant" classification schemes since they classify flows according to their behaviour during a single time interval.

Our first single-instant classification scheme exploits the property that the network-level prefix flow bandwidth distribution is heavy tailed. It identifies the threshold separating elephants from mice, as a cut-off point in the Complementary Cumulative Distribution Function (CCDF). This technique makes no assumptions about how high or low the threshold should be. Instead, it defines the threshold as the first point in the flow bandwidth distribution after which power-law behaviour can be witnessed. The second method classifies as elephants the high-volume flows whose cumulative bandwidth is equal to a certain fraction of the overall traffic. This second approach is more intuitive and may be preferred in an operational setting.

The performance evaluation of the proposed schemes is done on the grounds of four performance metrics. The first two metrics verify that the elephant flows identified indeed conform to the definition of the elephant class. The remaining two metrics evaluate the persistence of the derived classification. Our findings illustrate that classifying flows according to their "instantaneous" behaviour (i.e. their behaviour during a single interval) leads to high volatility in the resulting elephant class. We demonstrate this volatility in a number of ways: (i) the number of elephants at any time is highly variable, (ii) elephant flows need frequent reclassification, and (iii) elephants do not remain elephants sufficiently long for traffic engineering applications.

Because persistence in time is an important property and simple single-instant classification schemes cannot achieve this, we incorporate in the classification a metric that captures the temporal behaviour of a flow, called *latent heat*. The latent heat metric is appealing because it reduces the classification schemes' sensitivity to the natural variability of flows. We show that the "latent heat" scheme avoids unnecessary reclassification of flows due to short-term fluctuations that do not change a flow's essential nature. Performance evaluation of the proposed scheme demonstrates that it is capable of identifying a small number of elephant flows, that capture the majority of the traffic, while maintaining their state for much longer periods of time.

The remainder of this chapter is organised as follows. In Section 5.2, we describe the data analysed throughout the chapter. In Section 5.3 we present our methodology and in Section 5.4 we provide results on the performance of the classification when it is based on the single-instant flow behaviour. Results on the "latent-heat" classification scheme are presented in Section 5.5. We characterise the identified elephant network prefixes in Section 5.6 and conclude in Section 5.7.

5.2 Measurement environment

5.2.1 Collected measurements

The data used in this chapter come from packet traces collected in the core of Sprint's Tier-1 IP backbone network [73]. The analysis described throughout the chapter has been performed on 6 traces collected on OC-12 and OC-48 links, yielding similar results. In this chapter we present results only from two different OC-12 links. The specifics of those two traces are presented in Table 5.1. The links used are two hops away from the periphery of the network, and traffic is captured on its way towards the core of the network. Therefore, traffic towards specific destinations should exhibit a sufficient level of aggregation. The utilisation levels of these two links are given in Figure 5.1 (The times displayed in the figures are always in EDT.). We selected those two particular links because they exhibit different behaviour, which has implications on the classification process as will be seen in later sections. The first of these links exhibits a more pronounced bursty or busy behaviour. The second link provides a data set that varies in a smooth fashion throughout the day.

Trace label	Start time (EDT)	End time (EDT)	#packets	Link Speed
west coast	Jul 24 08:00:35 2001	Jul 29 02:42:55 2001	1,677,983,111	OC-12
east coast	Jul 24 08:00:34 2001	Jul 25 13:21:55 2001	1,678,055,791	OC-12

Table 5.1: Description of collected traces.

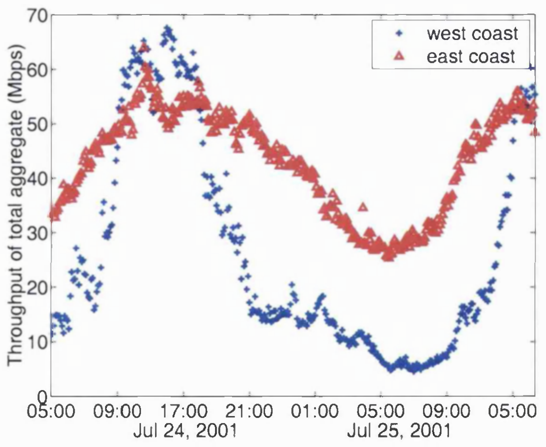


Figure 5.1: Link utilization for the west and east coast trace.

In parallel to the packet trace collection, we collect the BGP routing tables at the corresponding PoPs. Those BGP tables are default-free and contain approximately 120K entries, denoting all the IP domains that are likely to receive traffic through the monitored network.

5.2.2 Traffic and Time Granularity

The first issue addressed is the granularity level of the flows to be classified. For the purpose of intra-domain traffic engineering the natural granularity level is that of network prefixes appearing in routing tables. The granularity of a network prefix is a natural candidate for several reasons. First, such flows can easily be manipulated by simply changing the next hop address in the routing table for that particular flow. Second, routing policies tend to be applied to a network prefix as a whole, since network prefixes are the smallest routable entities in the Internet. Therefore any change in policy would affect the entire flow as we define it. This would not be the case, for example, for flows defined by destination address or prefixes of a fixed length. Third, it has been observed that elephants and mice do exist at this level of granularity [100].

Traffic engineering applications target optimal allocation of resources across the network by intelligent handling of the carried traffic. Such applications need to operate at timescales that are not too short, so that they are not affected by traffic variations due to protocol-specific behaviour. On the other hand, they should not operate at timescales that are too long, in order to exhibit sufficient reactiveness to events taking place inside the network. For instance, if the targeted application is the off-loading of congested links through the re-direction of high-volume destination network prefixes to other places in the network, then the time granularity over which volume measurements are computed should be limited, i.e. in the order of minutes. This is motivated by the fact that network operators may need to wait up to the duration of an entire interval (when new measurements become available) in order to react to the observed congestion (which is likely to be monitored through SNMP at 5-minute intervals).

For those two reasons, we select 5 minutes as the default duration of the time intervals, over which flow volume is computed. The time granularity of 5 minutes is also the default time granularity at which network monitoring information is usually collected through SNMP. Therefore, such a choice seems to agree with current network monitoring time scales. We investigate the effect of such a design choice in later sections, by looking into other time scales for flow volume computation.

Previous attempts at classifying flows into elephants and mice use a single flow feature, which is either 1) bandwidth [100, 102], or 2) duration [101]. Given that we are interested in persistently high-volume flows, for each flow we compute its bandwidth as the volume of the flow divided by the duration of the measurement interval, and focus on techniques that identify elephants based on their bandwidth alone. The duration of a network prefix flow bears no relevance in our problem, since our requirement for time persistence inherently imposes restrictions on the flow duration.

Computing flow volume every 5 minute interval, we find that in any given measurement interval, approximately 90% of the network prefixes have no traffic travelling towards them. We define a flow to be *active* if it receives at least one packet during the measurement interval. Figure 5.2 presents the number of active prefixes at each measurement interval over a two day period. The observation that there are only a few flows active at any given time slot is further confirmation that this aggregation level is appropriate for the identification of elephants in that it produces a limited number of flows on which computations need to be carried out. Thus, allowing for a network-level traffic abstraction that could also be applied to QoS and accounting policies.

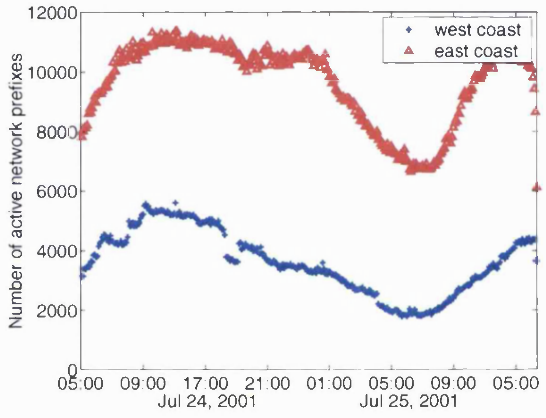


Figure 5.2: Number of active network prefixes for the west and east coast trace.

The cumulative distribution function (cdf) for the bandwidth of the derived flows is presented in Figure 5.3. Each curve corresponds to a different 5 minute interval within the first hour of the trace. We see that at any time during this hour, the top 100 flows account for 50% to 70% of the total traffic, while the top 400 flows account for 75%-85% of the total traffic. This confirms that elephants and mice do exist in our data set at the chosen granularity level.

We further examine our dataset for heavy-tail properties. In Figure 5.4 we present the complementary cumulative distribution function plot of the bandwidth values of our flows. The straight line at the end of the distribution indicates that it may be heavy-tailed. We use the *aest* test designed by Crovella and Taqqu (described in [108]) to estimate the scaling parameter of the distribution of the flows' bandwidths. We find a scaling parameter whose values vary between 1.03 and 1.2, verifying that the bandwidth distribution is indeed heavy-tailed. In Section 5.3, we describe the properties of the heavy-tail distributions, and we devise a method in which we make use of these properties in the classification process.

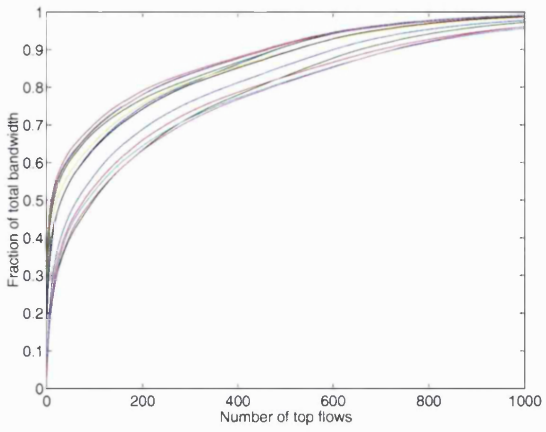


Figure 5.3: Cumulative Distribution Function of flow bandwidths for the first 12 5-minute intervals of the east coast trace.

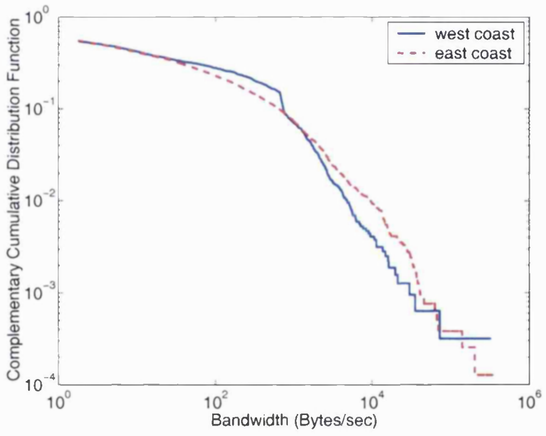


Figure 5.4: Complementary Cumulative Distribution Function of flow bandwidths for the east and west coast trace.

5.3 Elephant Classification

Before proceeding we introduce some notation. Let i denote the index of a network prefix flow, i.e., a BGP routing table entry. Let τ denote the length of the time interval over which measurements are taken. Time is discretized into these intervals, and n is the index of time intervals. We define $X_i(n)$ to be the average bandwidth of the traffic destined to a particular network prefix i during the n^{th} time slot of length τ. We use C_1 to represent the set of all flows considered elephants, and C_2 to represent the set of flows considered mice.

Using our packet-level measurements, we aggregate packets into "flows", based on their destination network prefix as announced through BGP [109], and measure their bandwidth across fixed time intervals. This gives us a set of flows, and an associated bandwidth for each flow. Given that there is no well defined technique for the identification of elephant flows on a given time interval, we begin our investigation focusing on a single time interval, and look into two different techniques for the classification of flows into "elephants" and "mice". We call these techniques "single-instant classification techniques", since they classify flows based on their behaviour during a single time interval.

5.3.1 Single-Instant Classification

Our methodology consists of two phases: 1) threshold detection phase, and 2) threshold update phase. In the first phase, we seek a bandwidth value $v(n)$, such that if a flow's average bandwidth in the chosen time interval exceeds the threshold, it is classified as an elephant. Otherwise, it is considered a mouse. In the second phase, we update the threshold, i.e., at time $n + 1$, based on the current threshold detected plus past threshold values.

Threshold Detection Phase

In the detection phase of our classification process, we analyse the properties of the dataset collected and calculate a threshold value $v(n)$ accordingly. This value $v(n)$ is likely to change with time always isolating those flows that contribute the highest amount of information in each time interval.

We propose two different techniques to identify the initial threshold value.

1. The first method takes into account the heavy tail property of the collected data set. The threshold value is set in such a way so that a flow is characterised as an elephant, only if it is located in the tail of the flow bandwidth distribution.

2. The second technique is more intuitive, and could be preferred in operational environments. It requires the setting of an input parameter corresponding to the fraction of total traffic we would like to place in the elephant class. The threshold is set in such a way that all the flows exceeding it account for the requested fraction of the total traffic.

(1) aest approach

A random variable X follows a heavy-tailed distribution (with tail index α) if

$$P[X > x] \sim cx^{-\alpha},\ as\ x \rightarrow \infty,\ 0 < \alpha < 2 \tag{5.1}$$

where c is a positive constant, and where $\sim$ means that the ratio of the two sides tends to 1 as $x \rightarrow \infty$. Aest is a method for the estimation of the scaling index α proposed by Crovella and Taqqu [108].

The particular value of α is important in many practical situations, and a number of methods are commonly used to estimate its value. For instance, values of α lower than 1 indicate that the random variable X has an infinite mean. Values of α between 1 and 2 indicate infinite variance.

The aest method identifies the portion of a dataset's tail that exhibits power-law behaviour. It is based on the fact that the shape of the tail determines the scaling properties of the dataset when it is aggregated. By observing the distributional properties of the aggregates one can make inferences about where in the tail power-law behaviour begins. We use this method to identify the first point in the distribution after which power law properties can be witnessed. We set this particular bandwidth value as the threshold value separating elephants and mice in that given measurement interval.

In other words, if we define $\bar{F}(x)$ as the complementary cumulative distribution function of the flow bandwidths, i.e. $\bar{F}(x)$ = 1 - F(x) = P[X>x], then $v(n)$ = $\hat{x}$, so that

$$\hat{x} = \min(x; \frac{dlog\bar{F}(x)}{dlogx} \sim -\alpha)$$

The aest methodology has the advantages of being nonparametric and easy to apply. Moreover, it has been shown to be relatively accurate on synthetic datasets. More importantly, there has been evidence that the scaling estimator, as calculated by aest, appears to increase in accuracy as the size of the dataset grows. Given that our datasets feature at least two thousand measurements, this technique will provide us with reasonably accurate values for the cutoff points in the heavy tail flow bandwidth distribution measured in each time interval.

(2) Constant load approach

This second approach is a more intuitive approach towards identifying the threshold bandwidth values separating elephants from mice. For each time interval n, we define local threshold $v(n)$, such that the flows exceeding this threshold account for $\alpha\%$ of the total traffic on the link. We sort the flows and starting from the largest, we proceed down the list adding flows into the elephant class. We stop when the volume of the elephant class reaches $\alpha\%$ of the link load. If we rank bandwidths in ascending order, let j denote the index of a flow, and M denote the highest bandwidth flow in timeslot n, then $v(n) = X_{m-1}(n)$ where m is chosen such that:

$$\sum_{j=M}^{m} X_j(n) < \alpha \sum_{j=M}^{1} X_j(n),\ and\ \sum_{j=M}^{m-1} X_j(n) >= \alpha \sum_{j=M}^{1} X_j(n).$$

Threshold Update Phase

To make use of the threshold derived for engineering purposes, we need to classify the data in time slot $(n + 1)$ according to $v(n)$. This may not be adequate because we may not have a sufficiently large number of measurements in each time interval to yield representative estimates.

In addition, we expect that correlations exist across time slots (shown later in Section 5.4.2) and are thus motivated to allow past data to influence our choice of threshold. We thus calculate a more general classification threshold $\hat{v}(n)$ using a simple exponential smoothing on $v(n)$ as follows.

$$\hat{v}(n) = (1 - \beta)\, \hat{v}(n-1) + \beta\, v(n-1) \tag{5.2}$$

We call this the *update rule.* We found that a value of $\beta = 0.2$ leads to a sufficiently smooth $\hat{v}(n)$. We now use the threshold $\hat{v}(n)$ to classify the traffic during the n^{th} time interval. In other words, the final *decision rule* is the following.

$$\text{if } X_i(n) > \hat{v}(n) \text{ then } i \in C_1 \text{ else } i \in C_2 \tag{5.3}$$

5.3.2 Incorporating temporal behaviour in the classification

As will be seen in Section 5.4, the two classes obtained using the single-instant classification schemes exhibit insufficient persistence in time. What we want to avoid is reclassification of a flow if it transitions to the other side of the threshold for a short time. In other words, large flows should be allowed to experience short transient periods in which their volume drops. Similarly, small flows should be allowed short-lived bursts. This motivated us to compute a second per-flow metric that takes into account the temporal component of an elephant's behaviour.

In thermodynamics, "latent heat" is the heat energy required to change a substance from one state to another. In order to be able to identify the points in time, when elephant flows below the threshold change state into mice, we define a new metric, which we call "latent heat". An elephant is assumed to accumulate "latent heat" as long as it exceeds the identified threshold. Once it falls below the threshold, it starts losing "heat", and changes state (i.e. becomes a mouse) when its "latent heat" becomes lower than a preset value.

In Figure 5.5, we illustrate the idea behind the use of the "latent heat" metric. Figure 5.5 presents the bandwidth measured for one particular destination network prefix on the west coast trace, along with the computed "0.8-constant load" threshold separating the flow measurements into two classes. One can see that this particular flow most of the time receives traffic at greater rates than the computed threshold rate. Moreover, in most of the cases its positive bursts are greater than its negative dips. One could argue that this flow should not be classified as a mouse for its short stays below the computed threshold. With the "latent heat" metric we aim at postponing reclassification of flows due to short-lived bursts or drops.

In order to achieve this, at each time interval n, apart from the flow's bandwidth, $X_i(n)$, we calculate the distance between the bandwidth achieved and the corresponding threshold value, $v(n)$, derived using the proposed approaches such as the "aest", and the "constant load". One

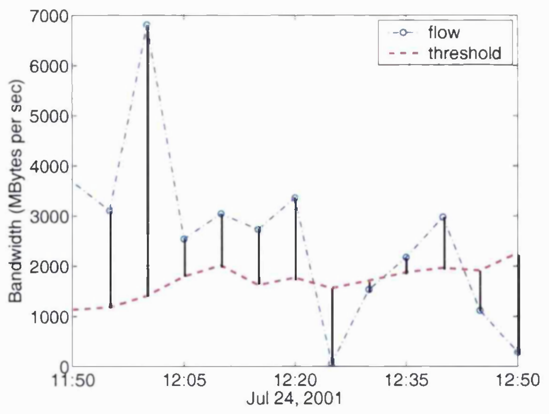


Figure 5.5: Illustration of the idea behind the "latent heat" metric.

could define the "latent heat" as a function of the differences between the measured bandwidth and the computed threshold over a number of intervals. Nevertheless, such an approach would imply maintaining statistics for the past flow behaviour equal to the number of intervals we decide to use. In order to avoid maintaining large per-flow statistics, we decide to use an exponential weighted moving average (ewma) for the computation of the "latent heat" (for details on the ewma model please refer to Appendix A). We thus define "latent heat" as follows:

$$LH_i(n) = \begin{cases} (1-\gamma)\, LH_i(n-1) + \gamma\, (X_i(n-1) - v(n-1)) & \text{if } n > 1 \\ 0 & \text{if } n = 1 \end{cases} \tag{5.4}$$

We would like our latent heat values to capture the temporal behaviour of a flow in the past hour (i.e. 12 5-minute intervals). Notice that allowing the latent-heat metric to incorporate the behaviour of the flow during the past 12 intervals will allow the flow in Figure 5.5 to be classified as an elephant flow despite its few drops. As described in Appendix A, there is a clear relationship between γ and the number K of past measurements that are allowed to influence the current value of an EWMA[1]. According to Equation (A.2), we set γ equal to $2/(12+1) = 0.15$. If one prefers longer or shorter influence of the past values on the current value, then one can set K, and consequently γ, accordingly.

The value for the "latent heat" is a smoothed out value of the flow's past behaviour. Thus, the "threshold update" phase is already incorporated in the "latent heat" computation. Hence, we proceed to classify flows at interval n as follows:

$$if\ LH_i(n) > thres,\ i \in C_1,\ else\, i \in C_2.$$

[1]In Appendix A, the number of past measurements that are allowed to influence the current value of the EWMA is denoted by n and the scaling parameter in the definition of the EWMA is denoted by α.

For the purposes of this work, we set $thres$ equal to 0. However, in case higher persistence in time is desired, one could increase the value $thres$ to identify as elephants those flows that are characterised by "latent heat" metrics greater than specific rate values. Moreover, if one wishes to identify the elephant flows that have exhibited the highest persistence in time, then one can select these flows that are characterised by the highest values of "latent heat".

In the remainder of this section, we present the performance metrics used for the evaluation of the proposed classification schemes.

5.3.3 Performance Metrics

In most applications of classification each data object inherently belongs to some particular class. This class is unknown to the observer whose job is to guess correctly the true state of nature. Our situation is different in that the flows do not naturally belong to a particular class. Instead we are imposing a categorisation onto our flows because it is useful for traffic engineering purposes. Because of this, the traditional performance measures based on misclassification errors cannot be used. The evaluation of the type of classification techniques proposed here should be based on the needs of the particular traffic engineering application that uses the classification. Given that we want to identify those flows that contribute significant amounts of traffic persistently over time, we need to verify that the elephant class obtained through our classification schemes indeed captures the majority of the traffic. Moreover, we need to confirm that the number of elephants identified is limited. Consequently, the first two metrics we use for the validation of our approach are the following.

1. The fraction of total traffic apportioned to the elephant class should be sufficiently large.

2. The number of elephant flows should be reasonably small, say less than a number F, where F is constrained by the router capabilities, and denotes a reasonable number of flows for which the router can keep state.

We explained earlier that our goal is to identify those flows that contribute sufficient volumes of traffic **persistently over time**. Ideally the majority of the elephants should remain elephants for long periods of time, i.e. in the order of few hours. The length of time that an elephant can remain an elephant is both a function of the flow itself and of the classification. It is a function of the classification in the sense that a particular high-volume flow will remain in the elephant class as long as the continually adjusting threshold stays lower that its average bandwidth. Because traffic on the Internet is highly variable, we don't expect flows to retain their classification indefinitely. We know that there will be many flows that will burst for a small amount of time, and become quiet afterwards.

To define a performance metric to capture time persistence we proceed as follows. Note that the classification scheme we have proposed, induces the following underlying two-state process on each flow. Let $I_i(n) = 1$ if $i \in C_1$, and $I_i(n) = 0$ if $i \in C_2$. At each classification time interval, the process either transitions to the other state or stays in the same state. We define the following two performance metrics in terms of this process.

3. For each flow i, we calculate the autocorrelation for a lag of 1, namely $Corr_i(1) = E[I_i(n)I_i(n+1)]$. This metric reflects how predictable the class of flow i is at time $n+1$ based on its class at time n. The classification scheme with higher per-flow autocorrelation values is the one in which the flows are more likely to retain their classification.

4. For each flow i, we calculate the average holding time the flow spends in the elephant state, and its average holding time in the mouse state. The classification scheme that yields larger holding times is more attractive for our purposes.

5.4 Results on the Single-Instant Classification Schemes

In this section we present the results derived applying the two single-instant classification methods described in Section 5.3.1. We compare the two methods with respect to the four performance metrics identified. We demonstrate that the proposed single-instant classification schemes lead to a highly volatile elephant (and mouse) class. This provides us with the motivation to introduce the "latent heat" metric, as presented in Section 5.3.

5.4.1 Classification Results

Figures 5.6, 5.7, and 5.8 present the threshold values computed using the two proposed techniques, the number of elephants, and the fraction of total traffic apportioned to elephants for both traces.

For the west coast trace, the threshold obtained with the "constant load" approach fluctuates throughout the day between 1 KBytes/sec and 2.5 KBytes/sec. As already shown in Section 5.2, the west coast trace exhibits high utilization levels during the peak period (Figure 5.1); a burst which is not accompanied by similar bursty behaviour in the number of active network prefixes (Figure 5.2). This implies that the burst in link utilization is not due to new network prefixes becoming active, but rather due to existing network prefixes receiving traffic at higher rates. Given this rise in the bandwidths achieved by the already active network prefixes, it is expected that now the threshold value isolating the flows contributing 80% of the total traffic is much higher.

The "aest" approach provides a lower estimate for the threshold. The reason why the

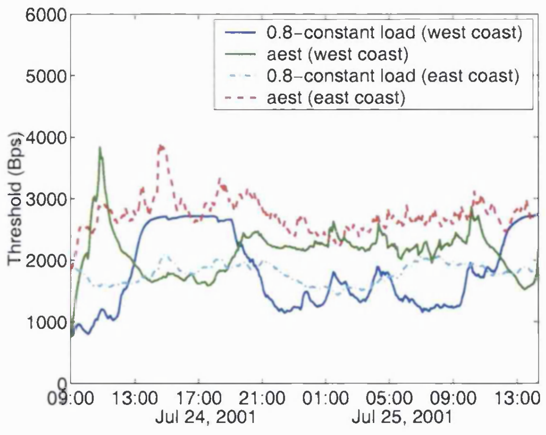


Figure 5.6: Derived thresholds $\hat{v}(n)$ for *0.8-constant load*, and *aest*.

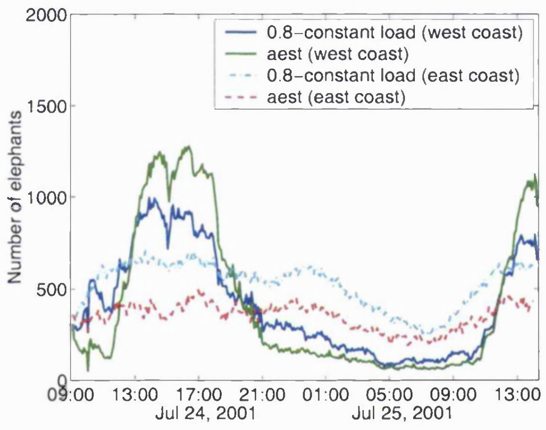


Figure 5.7: Number of elephant flows for *0.8-constant load*, and *aest*.

threshold is lower in the "aest" approach is because many of the flows in the peak period have now been positioned in the tail of the bandwidth distribution.

The effect of the burstiness in the utilization of the west coast trace is that the number of elephants identified with both techniques is much larger. Given that the "aest" threshold is much lower than the "constant load" one, the number of elephants identified with "aest" is much higher, thus accounting for as much as 90% of the total traffic on the link.

For the smoother east coast trace, where the trend in the link utilization is accompanied by similar trend in the number of active network prefixes, the threshold values calculated with both techniques behave similarly in time. The threshold values are slightly higher for the "aest" approach, accounting for less elephants, and smaller fraction of the total traffic. It is important to notice, though, that for the east coast trace, the total amount of traffic apportioned to elephants is approximately constant with time, and equal to 80% for the "constant load" (imposed by the technique itself), and 70% for the "aest" approach respectively.

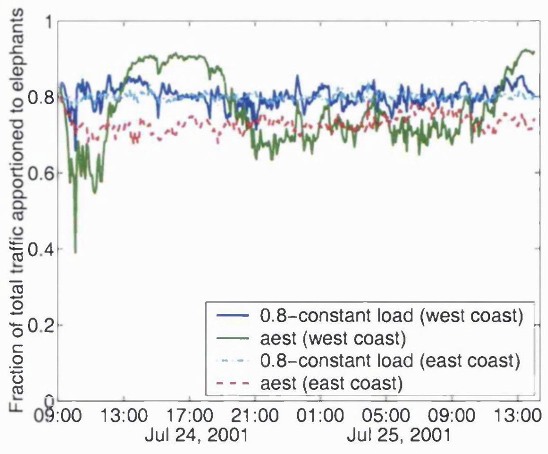


Figure 5.8: Fraction of total traffic accounted to elephants for *0.8-constant load*, and *aest*.

From the previous discussion it can be easily seen that in cases of bursty link utilization, maintaining a constant amount of traffic in the elephant class leads to a volatile number of elephant flows, and equally bursty threshold values. Moreover, in such cases, the performance achieved using the two suggested techniques is much different. In the case of smooth traffic the two approaches achieve comparable results.

5.4.2 Temporal Behaviour of Classification

One of our primary requirements is that the flows retain their class as long as possible. We say "as long as possible" because on the one hand we want to avoid updating state too frequently, but on the other hand we do not want to keep flows in a particular state if their bandwidth changes significantly. To assess this issue of the persistence of the classification with time, we examine our last two metrics, namely, the autocorrelation value at lag 1, and the average holding times in the elephant state. We compute these values for the peak hours, which is the period during which persistence is perhaps mostly needed. The peak hours are between 12pm and 5pm EDT.

Table 5.2 shows that the $I_i(n)$ processes achieve high values of autocorrelation for both approaches and for both traces. What has to be noted is that the values presented in Table 5.2 are highly biased towards the performance of the mice flows. The total number of destination prefixes that become active at some point throughout the duration of the west coast trace is 30,241 out of which the elephants vary between 454 and 994 (depending on the time interval n). In other words, only a small number of the autocorrelation values correspond to elephant flows. This is one of the reasons we were motivated to look at the holding times of the classification

	East Coast		West Coast	
	AEST	constant load	AEST	constant load
min.	0.8611	0.8687	0.8497	0.8583
5%	0.9789	0.9786	0.9737	0.9769
avg.	0.9827	0.9823	0.9814	0.9819
95%	0.9833	0.9833	0.9833	0.9833
max.	0.9913	0.9913	0.9906	0.9912

Table 5.2: Autocorrelation values at lag 1 for peak hours.

	East Coast		West Coast	
	AEST	constant load	AEST	constant load
min.	1	1	1	1
20%	1	1	1	1
50%	1.5	1.5	2	2
avg.	3.72	4.1	8.63	8.09
80%	3.5	3.5	12	10.5
max.	60	60	60	60

Table 5.3: Average holding times in the elephant state for peak hours (in 5-minutes slots).

process.

We compute the average holding time for each flow in the elephant state for the five hour busy period. The distribution of the average holding times is shown in Figure 5.9, where the x-axis values represent time slots of 5 minutes (so a value of 12 corresponds to 1 hour). Table 5.3 presents the basic statistics for the average holding times in the elephant state. We see from this table that the average holding time is approximately 20 minutes for the east coast trace, and 40 minutes for the west coast trace.

We further show the cumulative density function of the average holding times in Figure 5.10. More than 80% of the flows have average holding times less than half the maximum. It is clear from Table 5.3 and Figure 5.10 that few flows remain elephants "forever" (i.e., the duration of the entire peak period). It is natural to ask how well past behaviour can predict future behaviour. In particular we would like to know what is the likelihood that an elephant

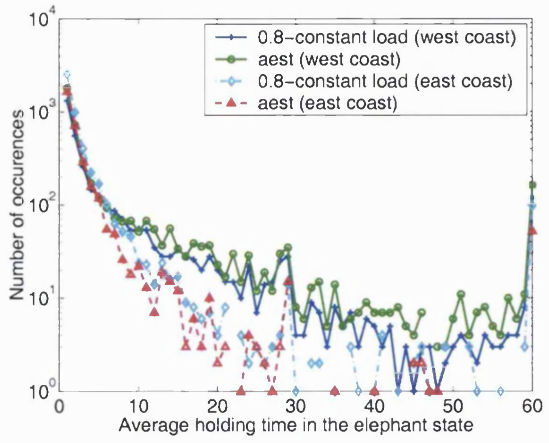


Figure 5.9: Distribution of the average holding time in the elephant state for peak hours (in 5-minutes slots).

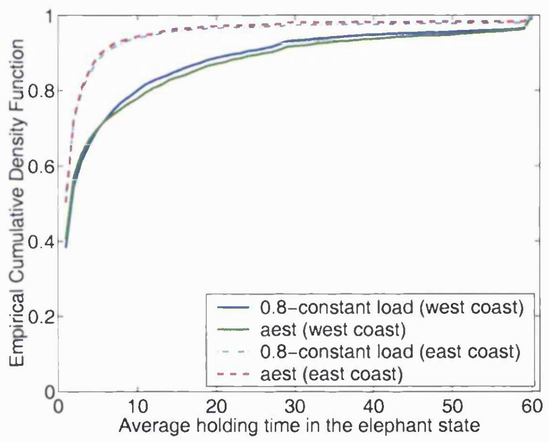


Figure 5.10: CDF of the average holding time in the elephant state for peak hours (in 5-minutes slots).

that has been an elephant for a certain amount of time will remain an elephant "forever". To measure this we calculate the probability that an elephant will remain an elephant for the entire duration of the peak period, given that it has stayed in the elephant state for a given number of intervals. In Figure 5.11, we present those probabilities for both traces and for both approaches.

Figure 5.11 shows that for the bursty west coast trace the predictability of the elephants is very limited. Flows that remain elephants for 36 intervals (i.e. 3 hours) during the peak period have a probability of less than 0.2 of remaining in that state for the whole 5 hours. For the east coast trace elephants show a higher level of predictability, and if they remain elephants for 3 hours, they have a probability greater than 0.5 of remaining elephants for the whole duration. Therefore, Figure 5.11 provides another indication that network prefix flows are very volatile,

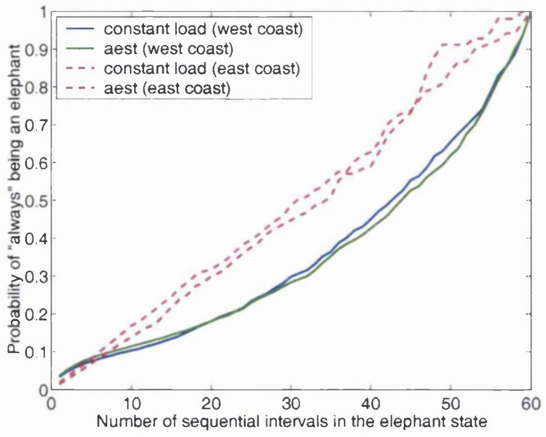


Figure 5.11: Probability of "always" remaining an elephant for *0.8-constant load*, and *aest*.

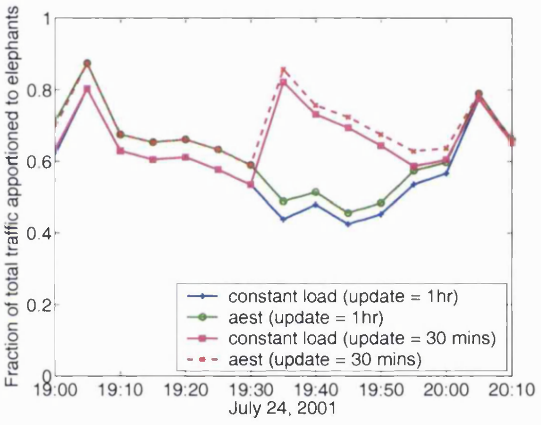


Figure 5.12: Effect of infrequent classification on the load of the elephant class.

and if persistence is the desired property, simply classifying flows based on their bandwidth at each interval is not sufficient. In practical settings such small probabilities of successfully identifying elephants in the future has limited applicability.

These results have implications for other methods proposed in the literature. For example, in [105] the authors create state for each flow that exceeds their threshold, and they keep that state for the lifetime of the application. That wouldn't make sense for our data since less than 5% of the flows classified as elephants in any instant remained elephants for the entire 5-hour period. We conclude that flows need to be reclassified fairly frequently.

Until now we have considered reclassifying flows every 5 minutes. In order to evaluate other time scales for reclassification, we consider two other cases, namely reclassifying every 30 minutes and 1 hour. Even though we reclassify at these intervals, we continue to measure flow bandwidth every 5 minutes. Figure 5.12 shows the amount of traffic apportioned to elephants using both techniques with these longer reclassification times for the west coast trace. This helps us to understand how fast the list of flows belonging to the elephant class becomes stale. Regardless of the method used for classification, we see that when flows are updated at longer intervals, the fraction of total traffic apportioned to the elephant class may drop dramatically, in particular it can fall as low as 40% within 25 minutes from the last classification interval. Flows classified as elephants at 19:05 (Figure 5.12) experience lower bandwidths in the next intervals, while flows that may experience high loads cannot be placed in the elephant class, until the new update period. If we update too slowly, we can experience a significant jump in the elephant load at the time of reclassification (e.g., at 19:35). Note that not updating the list of elephants sufficiently often may endanger the efficacy of the traffic engineering application because it may no longer be giving differential treatment to the largest flows.

5.4.3 Other Single-Instant Classification Techniques

The two metrics for the number of elephants identified, and the fraction of total traffic apportioned to the elephant class are clearly related. The target of our second classification approach is to maintain a constant elephant load. In this section, we report results on two additional single-instant classification approaches that could be selected and would target at maintaining a constant number of elephants or a threshold consistently equal to a specific fraction of the link utilization or capacity. This section should not be conceived as a complete evaluation of the techniques described, but a simple illustration of how alternative techniques would perform.

A simple way to identify the high-volume flows in a network is to collect the bandwidth distribution for the destination prefixes in each interval, and to classify the top N flows at each interval as elephants. For our west coast trace, we set N equal to 100 (approximately the number of flows consistently in the elephant class for the 5-peak hours), and measure the fraction of total traffic apportioned to the elephant class[2]. For comparison reasons, we also calculate the threshold value as the bandwidth of the smallest elephant. This is the bandwidth values flows would have to exceed to be classified as elephants in the threshold-based approaches.

Our results indicate that with such a technique the elephant load may fluctuate between 35% and 94% of the total traffic. This points to an important tradeoff. We saw in Figures 5.7 and 5.8 that methods that succeed in keeping a reasonably consistent fraction of the total traffic in the elephant class yield large fluctuations in the number of elephants. In this approach, that focuses on keeping the number of elephants constant, we observe large fluctuations in the load due to elephant flows. It thus appears difficult to keep both the elephant load and the number of elephant from experiencing fluctuations simultaneously.

[2]The nature of our results did not change for different values of N, ranging from 10 to 500.

The corresponding threshold range for this method was from 1 to 12 KBytes/sec depending on the time of day, exhibiting similar busy periods as the utilization of the respective link. An important result of this classification approach is that only 11 flows remained elephants for the whole 5-hour peak period, while 97% of the other elephant flows featured holding times of less than 24 intervals (i.e. 2 hours). During this 5-hour period, approximately 1800 flows became elephants at some point (in both traces). Since only 11 of these remained elephants for the entire period, the probability that a given flow in the set of top 100 at any given interval remains among the top 100 flows for a very long duration is minimal.

In an alternative approach we set the threshold value equal to 1% of the utilization of the monitored link, as proposed in [105] (1% of the capacity of our OC-12 links was never exceeded by a single flow in our traces). Under such a scheme, the number of elephants identified is limited between 3 and 23 flows, and the amount of captured traffic fluctuates between 10% and 60% of the total traffic. The maximum fraction of traffic is apportioned to the elephant class in the off-peak hours, when the utilization is much lower, and therefore the computed threshold is easier to exceed. This classification process offers much lower autocorrelation values than any other classification technique, while the maximum holding time in the elephant class is 60 5-minute intervals, achieved by only 2 flows. All the other elephant flows achieve holding times of less than 6 intervals (i.e. 36 minutes). The statistics for the average holding times achieved with the four single-instant classification methods described are summarised in Table 5.4.

	Average holding time statistics					
	min.	20%	50%	avg.	80%	max.
AEST	1	1	2	8.63	12	60
constant load	1	1	2	8.09	10.5	60
constant number	1	1	1.2	3.47	2.75	60
1% utilization	1	1	1	4.12	3	60

Table 5.4: Statistics for the holding times at the elephant state for all four single-instant classification methods (west coast in 5-minute slots).

This brief evaluation of alternate single-instant classification schemes gives us confidence in that the volatility in the derived classes is not an artifact of the way the separation threshold is selected in our proposed schemes. Rather, it stems from the fact that elephant destination network prefixes in the core of our backbone network are not characterised by consistently high volumes of traffic. This result was expected, since the bandwidth achieved by a destination network prefix will depend on current network conditions, e.g. bottleneck bandwidth, etc.

5.4.4 Other Timescales for Measurement and Classification

Throughout this section we have considered measurement and classification intervals of $\tau = 5$ minutes each. In order to assess the utility of this time scale we also consider $\tau = 1$ and $\tau = 60$ minutes.

Table 5.5 presents the statistics for autocorrelation values achieved at lag 1, calculated over the five peak hours, when $\tau = 60$ minutes. These values are much smaller than the ones in Table 5.2. At $\tau = 60$ we see more of a distinction between the two schemes than we did when $\tau = 5$. We also calculate the main statistics for the average holding time at the elephant state, and we observe that the 80th percentile of the derived distribution is equal to 3 intervals (i.e. 3 hours).

	East-Coast τ = 60 mins	
	AEST	0.8-constant load
min.	0.5833	0.8333
5%	0.8333	0.8333
avg.	0.8318	0.8373
95%	0.8333	0.8571
max.	0.8889	0.8571

Table 5.5: Autocorrelation values at lag 1 for the peak 5 hours (τ = 60 mins).

Therefore, it is evident that even at higher time scales, when flows' bandwidth would be expected to behave in a smoother fashion, volatility is the main property of the elephant flows. We notice that the autocorrelation at $\tau = 60$ is approximately 10% less than what it is at $\tau = 5$, and the number of intervals an elephant may remain an elephant is much smaller than the one achieved when $\tau = 5$. In addition, as noted in Section 5.2, such a large time scale for classification offers limited reactiveness to the network operators.

We also consider the interval $\tau = 1$ minute. We find that the autocorrelation for $\tau = 1$ and $\tau = 5$ are almost identical, indicating that there is not any need to go lower than 5 minutes. This is only an initial examination into the appropriate time scale for such a classification. However, these remarks give us confidence in that the results presented so far are not an artifact of the time scale used but rather a property of the traffic itself.

5.5 Latent Heat Classification Results

In this section, we report on the results obtained using the latent heat classification scheme. As described in Section 5.3, the latent heat metric is defined based on both the flow bandwidth and the classification threshold separating flows into two classes. Therefore, the results presented in this section will report on the performance of the latent heat scheme as applied on threshold values obtained using the "0.8-constant load" and "aest" approach.

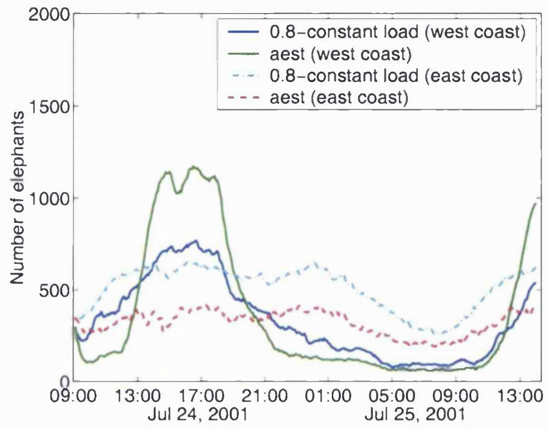


Figure 5.13: Number of elephants derived using the latent-heat classification method.

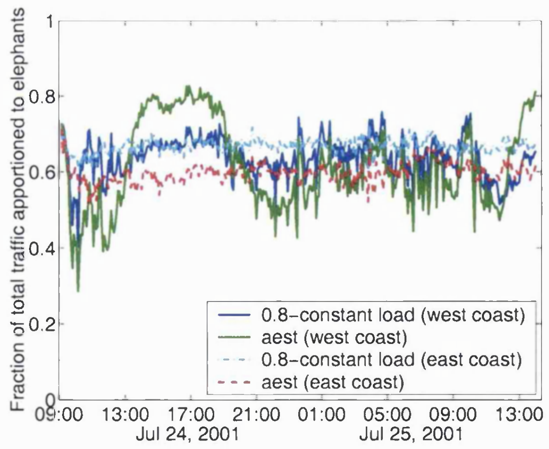


Figure 5.14: Fraction of total traffic apportioned to elephants using the latent-heat classification method.

In Figures 5.13 and 5.14, we present the number of elephants and fraction of total traffic apportioned to the elephant class for both traces under the latent heat approach. If we compare these figures with those of Figures 5.7 and 5.8, we see that with this classification approach we can keep the load reasonably consistent (Figure 5.14) while simultaneously experiencing less fluctuations in the number of elephants (Figure 5.13). Hence our latter classification scheme using latent heat achieves a better tradeoff between reducing fluctuations in either elephant load or the number of elephants, than any of the single-instant classification schemes.

The empirical probability density function for the average holding time in the elephant state is presented in Figure 5.15. Specific statistics are given in Table 5.6. There is no longer a peak at small holding time values. Recall that for single-instant classification we have average holding times of 20 and 40 minutes (depending upon the trace). Under this scheme we observe that average holding times for both traces are 1 1/2 hours or longer. In Figure 5.16 we further show the cumulative density function for the average holding times. Comparison of Figure 5.16 with Figure 5.10 clearly demonstrates the improvement in the persistence of the elephant class once one allows flows to experience short-lived bursts or drops. We believe that classification schemes such as this one, that avoid reclassification for short-term fluctuations, identifies the type of long-lived elephant flows that are good candidates for traffic engineering applications.

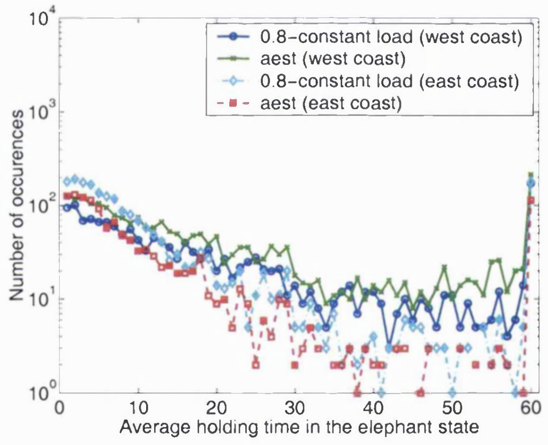


Figure 5.15: Distribution of the average holding time in the elephant state, when classifying based on the latent heat of a flow.

5.6 Profile of Elephant Flows

In this section we examine the profile of the elephants. We consider those flows that are classified as elephants by our latent heat classification method (the threshold is set according to the "0.8-constant load" scheme) throughout the entire 5 hour peak period. As a preliminary assessment on the profile of the elephants, we examine their network prefix lengths and the type of organisation they belong to.

One would expect that the network prefixes of the elephants would correspond to large domains advertised through BGP. As already mentioned, the monitored links are OC-12 links

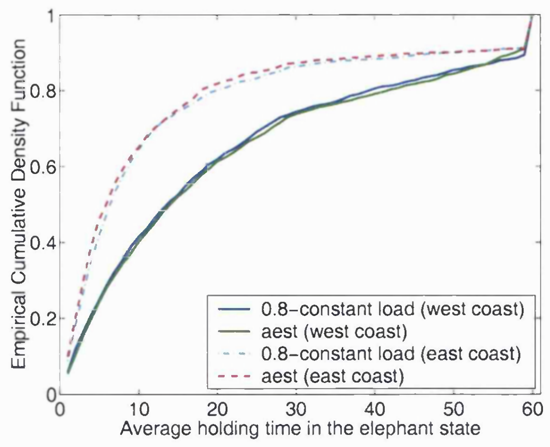


Figure 5.16: CDF of the average holding time in the elephant state, when classifying based on the latent heat of a flow.

	East Coast		West Coast	
	AEST	constant load	AEST	constant load
min.	1	1	1	1
20%	2.5	3	4.8	4.4
50%	6.33	7	14.66	14
avg.	13.59	14.21	21.71	21.25
80%	18.15	20.5	42	40
max.	60	60	60	60

Table 5.6: Statistics for the holding times at the elephant state under the latent heat classification scheme (in 5-minute slots).

interconnecting backbone routers in a PoP, and traffic is captured as it travels towards the core of the network. In Figure 5.17 we present the density function for the length of network prefixes that become active at some point in our trace. We observe that most of our active network prefixes come from /24 networks, followed by /16 and /18. Active networks with prefixes of /8, and /10 are limited in number to less than 100. In the same figure we present similar statistics for the network prefixes that get classified as elephants. The lengths for the elephant prefixes are contained in a smaller region, namely between /12 and /26. For the west coast we even witness a /32 network always in the elephant class. This is a customer network using NAT, advertising a single IP address within the Sprint IP backbone.

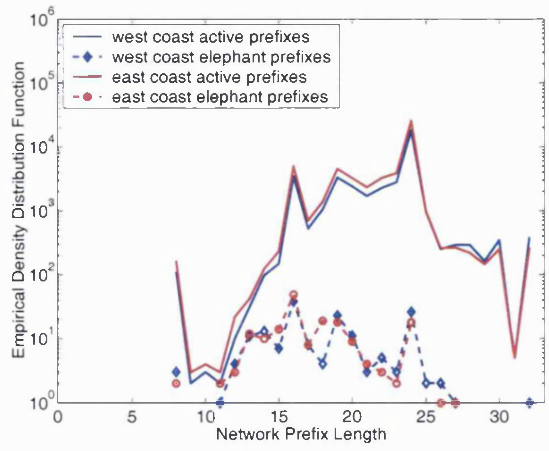


Figure 5.17: Network prefix length distribution for the elephants.

Finding only two or three (depending upon the trace) /8 flows among our elephants is an unexpected result. Such a result cannot be generalised for the whole Internet because it is influenced by the way our network is engineered, and the way BGP policies impact traffic flow throughout our network. Indeed, comparing the elephants identified on the west and the east coast we found that some network prefixes showed up on only one of the two coasts. This result is well expected since BGP policies guide the entry and exit points in our network.

In order to gain a better understanding about the type of businesses contributing the most to the overall load, we classify elephants into the following categories: Tier-1, Tier-2, Tier-3 ISPs, corporations, universities, and government institutions. The definition of what constitutes a Tier-1 network is rather controversial. The most popular definition is that Tier-1 ISPs are those networks that have access to the global Internet Routing Table and do not purchase network capacity from other providers [8]. The second tier generally has a smaller national presence than a Tier 1 ISP and may lease part or all of its network from a Tier 1. Lastly, a Tier 3 ISP is typically a regional provider with no national backbone.

According to the above categorisation, we profile[3] the elephants in our network as shown in Table 5.7. The vast majority of the elephant network prefixes belongs to Tier-1, Tier-2, and Tier-3 ISP providers. We believe that profiling elephant flows is essential once one intends to apply specific traffic engineering policies on the traffic destined towards them.

Notice that certain businesses may advertise more than one network prefixes for their net-

[3] In order to derive the profile of the different elephant destination network prefixes, we had to name resolve the network prefix, and determine the company it belongs to. Then we searched through its primary web site to identify whether the company description complied with any of the above Tier-n definitions. For instance, most of the Tier-3 prefixes present in our data belong to regional DSL and dialup providers.

#elephants	Tier-1	Tier-2	Tier-3	corp.	.edu	gov.
west coast	40	42	28	32	19	6
east coast	22	72	27	40	7	7

Table 5.7: Profile of the elephant network prefix flows.

work. Therefore, the results presented in Table 5.7 do not correspond to unique businesses, but rather to the number of elephant network prefixes in each category.

5.7 Conclusions

One of the invariants of Internet traffic is what is often called the "elephants and mice" phenomenon; a small number of flows contributes the majority of the traffic. Such a property could be exploited in a traffic engineering context, since with the differential treatment of a small number of flows, the operator is likely to affect a large amount of traffic. In order to capitalise on this phenomenon network operators must be capable of identifying elephant flows in their network. Moreover, in order for such a classification to be useful within a traffic engineering context, the resulting classes have to be characterised by sufficient persistence in time.

In this chapter, we defined flows at the granularity of a destination network prefix and measured their bandwidth at fixed intervals. We defined as elephants a small number of flows that contributes high volumes of traffic persistently over time. Our contributions can be summarised as follows.

Firstly, we proposed two "single-instant" classification schemes for the separation of the flows into two classes according to their properties during a single time interval. These schemes capitalise on the properties of the collected data sets and compute the separation threshold between the two classes such that (i) all the flows characterised by bandwidth values greater than the threshold are located in the tail of the flow bandwidth distribution, or (ii) the amount of traffic placed on the elephant class equals a preset fraction of the overall traffic. These schemes do not make any assumptions about how high or low the separation threshold should be. Even though they were shown to result in highly volatile classes, they could be employed in case persistence is not a requirement.

Secondly, we proposed four performance metrics for the evaluation of the proposed schemes. Two of these metrics are specifically designed so that they evaluate the persistence of the derived classification in time. Given that the issue of persistence in time for the elephant flows has never been addressed before, to the best of our knowledge, this is the first work reporting on such metrics.

Thirdly, we showed that classifying flows based on their bandwidth during a single interval leads to high volatility in the resulting classification. This volatility has been illustrated in three ways: (i) we found that flows remain in the elephant state for surprisingly short periods of time; (ii) the number of elephants varies considerably over different time intervals; and (iii) if elephants are not reclassified frequently, the total load in the elephant class can change substantially. Consequently, classifying flows as elephants according to their volume during a single time interval does not guarantee persistence of their volume in future intervals.

Lastly, we presented a new classification approach that incorporates the temporal behaviour of the destination network prefix flows. Under this scheme, classification of the flows is not done on the grounds of their bandwidth during a single interval, rather based on their "latent heat". We defined "latent heat" as the evolution of the difference between the flow's bandwidth and the threshold computed by the single-instant classification schemes. This new metric was shown to be capable to allow flows to experience short-lived bursts or drops that would otherwise lead to a change of state. We showed that classifying flows according to their "latent heat" leads to smaller fluctuations in the number of elephants identified and reasonably consistent load in the elephant class. More importantly, under this classification scheme, elephant flows maintain their class for much longer periods of time. Under our initial schemes, the average lifetime of an elephant was between 20 and 40 minutes. Under the latent heat scheme, the average lifetime of an elephant is greater than 1.5 hours.

In profiling the elephants, we found that elephant prefixes generally lie within a limited range of network prefix lengths, between /16 and /26. Surprisingly, we found few large prefixes such as /8 and /10. Although there were a few corporations, government agencies and universities among our elephants, we learned that the vast majority of them are Tier-1, Tier-2 and Tier-3 ISPs.

Overall, we conclude that bandwidth variability is one of the main properties of a destination network prefix flow, despite its level of traffic aggregation. Consequently, single-instant classification schemes, even though attractive in their simplicity, they are probably insufficient for most traffic engineering applications. While we believe that our latent heat classification scheme is more attractive to traffic engineering applications, we also conclude that identifying persistent elephants imposes challenges. This is important because the idea of isolating elephants for traffic engineering purposes has been widely proposed, but there has been no prior effort on assessing the feasibility and issues involved in doing so.

The work presented in this chapter is the result of collaboration with Nina Taft, Supratik Bhattacharyya, Patrick Thiran, Kavé Salamatian, and Christophe Diot. Patrick Thiran and Kavé Salamatian helped in earlier stages of this work. Nina Taft, and Supratik Bhattacharyya have contributed to the methodology and the interpretation of the results. In addition, Nina Taft and Christophe Diot have helped with the writing. However, the largest part of the methodological approach, all the code, and the detailed analysis and presentation of the collected results have been done by me.

Chapter 6

Long-Term Forecasting of Internet Backbone Traffic

In the previous chapter, we proposed a scheme for the identification of these destination network prefixes that carry the majority of the traffic flowing on a link persistently over time. We suggested that in cases of link overload, the network operator may offload the congested link in a controlled fashion by re-directing selected elephant prefixes to less congested parts inside the network. Such an approach can lead to the desirable result only when the congestion effect witnessed is transient. Consistent overload will be much harder to overcome with simple techniques like the one presented. In such cases, there is an indication of traffic growth, which may require additional network provisioning.

In this chapter, we introduce a methodology to predict *when and where* link additions/upgrades have to take place in an IP backbone network. Using SNMP statistics, collected continuously since 1999, we compute aggregate demand between any two adjacent PoPs and look at its evolution at time scales larger than one hour. We show that IP backbone traffic exhibits visible long term trends, strong periodicities, and variability at multiple time scales.

Our methodology relies on the wavelet multiresolution analysis and linear time series models. Using wavelet multiresolution analysis, we smooth the collected measurements until we identify the *overall long-term trend.* The fluctuations around the obtained trend are further analysed at multiple time scales. We show that the largest amount of variability in the original signal is due to its *fluctuations at the 12 hour time scale.*

We model inter-PoP aggregate demand as a multiple linear regression model, consisting of the two identified components. We show that this model accounts for 98% of the total energy in the original signal, while explaining 90% of its variance. Weekly approximations of those components can be accurately modelled with low-order AutoRegressive Integrated Moving Average (ARIMA) models. We show that forecasting the long term trend and the fluctuations of the traffic at the 12 hour time scale yields accurate estimates for at least six months in the future.

Required Network Measurements and Time Granularity

Forecasting *when* and *where* link upgrades/additions have to take place in the core of a backbone network requires SNMP information collected throughout the core of the network, and topological information, listing all the active backbone routers, their operational links, and their destinations. The results presented in this chapter are derived from the correlation of the SNMP information with the topological information so as to identify aggregate demand between any two adjacent PoPs. Inter-PoP aggregate demand is computed as the average utilisation of a link in the aggregate at the time scale of 1.5 hours.

6.1 Introduction

IP network capacity planning is a very important task that has received little attention in the research community. The capacity planning theory for traditional telecommunication networks is a well explored area [28], which has limited applicability in a packet-based network such as the Internet. It normally depends on the existence of a traffic matrix, identifying the amount of traffic flowing from any source to any destination of the network under investigation. Moreover, it requires accurate modelling of the incoming traffic, as well as accurate predictions of its future behaviour. The above information is then combined in a network simulation to identify the points where future upgrades will be needed.

This approach cannot be used in the environment of an IP backbone network because (i) we do not have a way of measuring or accurately estimating the traffic matrix for such a large scale network, (ii) we do not really know how to model the incoming traffic of a backbone network, and (iii) simulating such a large scale network is typically not feasible.

The current best practice in the area is based on the experience and the intuition of the network operators. Moreover, it usually relies on marketing information regarding projected number of customers at different locations within the network. Given provider-specific over-subscription ratios, and traffic assumptions, the operators estimate the effect that the additional customers may have on the network-wide load. The points where link upgrades will take place are selected based on experience, and/or current network state. For instance links that currently carry larger volumes of traffic are likely to be upgraded first.

Our goal is to enhance the above practices using historical network measurements collected with the Simple Network Management Protocol (SNMP). The intuition behind our approach is to use mathematical tools to process historical information and extract trends in the traffic evolution at different time scales. This approach requires the collection of network measurements over long periods of time.

In this chapter, we analyse three years of SNMP information collected throughout a major Tier-1 IP backbone. Correlating those measurements with topological information, we calculate the traffic aggregate between any two adjacent PoPs and track its evolution over time. We explore the properties of these time series, and propose a methodology that can be applied to forecast network traffic volume months in the future.

Our methodology relies on wavelet multiresolution analysis and linear time series models. Initial observations on the traffic reveal strong periodicities, evident long term trends, and variability at multiple time scales. We use wavelets to smooth out the original signal until we identify the overall long term trend. The fluctuations of the traffic around the obtained trend are further analysed at multiple time scales. This analysis reveals that 98% of the energy in the signal is captured by two main components, namely the long term trend, and the fluctuations at the 12 hour time scale. Using the analysis of variance (ANOVA) technique, we further show that a multiple linear regression model containing the two identified components also explains 90% of the signal's variance.

We model the weekly approximations of the two components using ARIMA models, and develop a prediction scheme that is based on their forecasted behaviour. We show that forecasting network backbone traffic based on our model can yield accurate estimates for at least six months in the future. Moreover, with a minimal computational overhead, and by modelling only the long term trend and the fluctuations of the traffic at the 12 hour time scale, we produce estimates which are within 10-15% of the actual measured behaviour.

Our methodology combined with actual backbone traffic measurements leads to different forecasting models for different parts of the network. Our results indicate that different PoP-pairs exhibit different rates of growth and experience different types of fluctuations. This illustrates the importance of defining a methodology for deriving models as opposed to developing a single model for inter-PoP aggregate traffic flows.

In section 6.2 we present previous efforts at forecasting Internet traffic. Our objectives are presented in section 6.3. In section 6.4, we present the data analysed throughout the chapter and make some initial observations. Section 6.5 provides an overview of the wavelet multiresolution analysis, along with results of its application on our measurements. Forecasts are derived using linear time series models, presented in section 6.6. We discuss our findings and future work in section 6.7. We conclude in Section 6.8.

6.2 Related Work

An initial attempt toward long-term forecasting of IP network traffic is described in [110]. The authors compute a single value for the aggregate number of bytes flowing over the NSFNET, and model it using linear time series models. They show that the time series obtained can be accurately modelled with a low-order ARIMA model, offering highly accurate forecasts (within 10% of the actual behaviour) for up to two years in the future.

However, predicting a single value for the future network-wide load is insufficient for capacity planning purposes. One needs to pinpoint the areas in the network where overload may occur in order to identify the locations where future provisioning will be required. Thus per-node or per-link forecasts are required. The authors of [110] briefly address this issue, mentioning that initial attempts toward this direction did not prove fruitful.

Other work in the domain of Internet traffic forecasting typically addresses small time scales, such as seconds or minutes, that are relevant for dynamic resource allocation [111, 112, 113, 114, 115, 116]. To the best of our knowledge, our work is the first to model the evolution of IP backbone traffic at large time scales, and to develop models for long-term forecasting that can be used for capacity planning purposes.

6.3 Objectives

The "capacity planning" process consists of many tasks, such as addition or upgrade of specific nodes, addition of PoPs, and expansion of already existing PoPs. For the purposes of this work, we use the term "capacity planning" only to refer to the process of upgrading or adding links between two PoPs in the core of an IP network.

The core of an IP network is usually overprovisioned and consists of very high speed links, i.e. OC-48, OC-192. Those links are a rather large part of a network operator's investment and have a provisioning cycle between six and eighteen months. Therefore, the capability to forecast *when and where* future link additions or upgrades will have to take place would greatly facilitate network provisioning.

In order to address the issue of *where* upgrades or additions should take place, we measure and forecast aggregate traffic between adjacent PoPs. In that way carriers can determine which pair of PoPs may need additional interconnecting capacity. There are a number of factors that influence *when* an upgrade is needed. These factors include service level agreements with customers, network policies toward robustness to failures, the rate of failures, etc. We assume that carriers have a method for deciding how many links should interconnect a given pair of PoPs and the acceptable levels of utilisation on these links. Once carriers articulate a threshold beyond which traffic levels between PoPs are considered prohibitive, one can schedule an upgrade before these levels are actually exceeded. Our task is to predict when in the future the traffic levels will exceed these acceptable thresholds.

In this work, we use historical information collected continuously since 1999 on the Sprint IP backbone network. There are many factors that contribute to trends and variations in the overall traffic. Our measurements come from a highly dynamic environment reflecting events that may have short or long-lived effects on the observed behaviour. Some of the events that may have a long-lived effect on the observed behaviour include changes in the network topology and in the number of connected customers. These events influence the overall long-term trend, and the bulk of the variability observed. Events that may have a short-lived effect include link failures, breaking news or flash crowd events, as well as denial of service attacks. These events normally have a direct impact on the measured traffic but their effect wears out after some time. As a consequence, they are likely to contribute to the measured time series with values which lie beyond the overall trend. Given that such events are very hard to predict, and are already taken into account in the calculation of the threshold values that will trigger upgrades, as described earlier in this section, we will not attempt to model them in this work.

6.4 Measurements of inter-PoP aggregate demand

We now describe the measurements collected and analysed throughout the chapter. We present some initial observations about Internet traffic at time scales larger than one hour. These observations motivate the approach used in later sections.

6.4.1 Data collected and analysis

We collect values for two particular SNMP MIB (Management Information Base) objects, incoming and outgoing link utilisation in bps, for all the links of all the routers in the Sprint IP backbone throughout a period that spans from 1999 until July 1st 2002. This operation yields traces from more than 2000 links, some of which may not be active anymore. The values collected correspond to an exponentially weighted moving average computed on 10 second link utilisation measurements. The exponential weighted average has an average age of 5 minutes and allows for more recent samples to be weighted more heavily than samples earlier in the measurement interval[1].

Along with the SNMP data, we collect topological information. This information is collected several times per day by an agent downloading configuration information from every

[1]Because these objects belong to a proprietary MIB, we have no further information about how this average value is calculated.

router in the network. It contains the names of the routers in each PoP, along with all their active links, and their destinations. Therefore, it allows us to identify those links in the SNMP data set that interconnect specific PoPs in our network.

We correlate the SNMP data with the topological information, and derive aggregate demands, in bps, between any two adjacent PoPs. In this procedure we need to address two issues. Firstly, the collection is not synchronised, i.e. not all links are polled at the same time to avoid overload at the collection station. Secondly, the collection is not reliable (SNMP messages use UDP as their transport protocol), i.e. we may not have one record for each 5 minute interval for every link in the network. As a consequence, the derivation of the aggregate demands is performed as follows:

- For each link in the SNMP data, we identify its source and destination PoP. We use the notation $l_{sd}(k)$ to denote the kth link connecting PoP s to PoP d.
- Time is discretised in 90 minute intervals. We denote time intervals with index t. The reasons why we selected intervals of 90 minutes are provided in Section 6.5.1.
- The aggregate demand for any PoP-pair (s, d) at time interval t is calculated as the sum of all the records obtained at time interval t from all links k in $\{l_{sd}(k)\}$, divided by the number of records. This metric gives the *average aggregate demand of a link* from PoP s to PoP d at time interval t.

This approach allows us to handle the case of missing values for particular links in the aggregate flow. Moreover, it does not suffer from possible inaccuracies in the SNMP measurements, since such events are smoothed out by the averaging operation. With the aforementioned procedure we obtain 169 time series (one for each pair of adjacent PoPs in our network). For the remainder of the chapter we focus our discussion on eight of those. These are the longest traces at our disposal which also correspond to highly utilized paths throughout the network. In the following sections we look into their properties, and devise techniques for forecasting their values in the medium (i.e. months ahead) and long-term future (i.e. 6 months).

6.4.2 Initial observations

In Figure 6.1 we present the aggregate demand for three PoP pairs in our network. The time series span from October 2000 to July 2002. Those time series correspond to an increasing number of links in time. With vertical bars we denote the time when additional links became active in the aggregate. As can be seen, link additions are rarely preceded by a visible rise in the carried traffic. This behaviour is due to the long provisioning cycles.

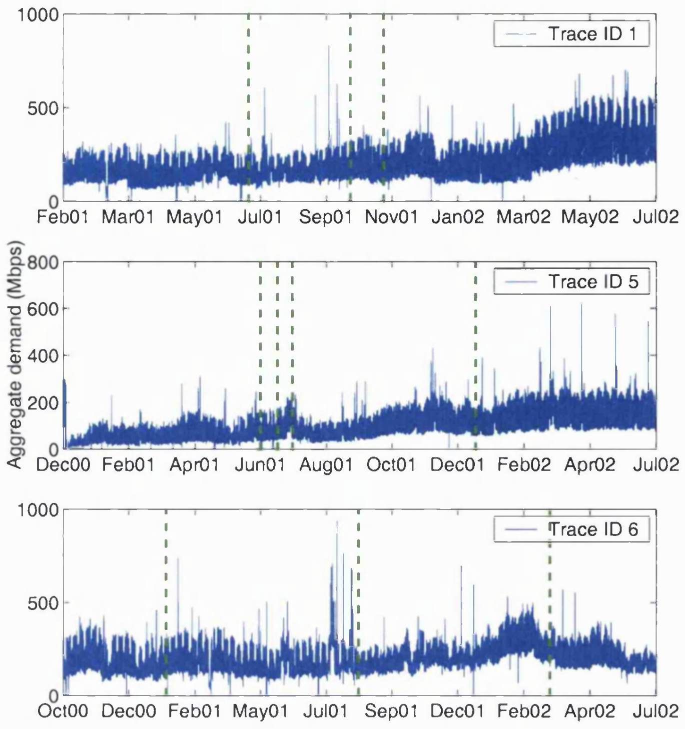


Figure 6.1: Aggregate demand for Traces 1, 5, and 6.

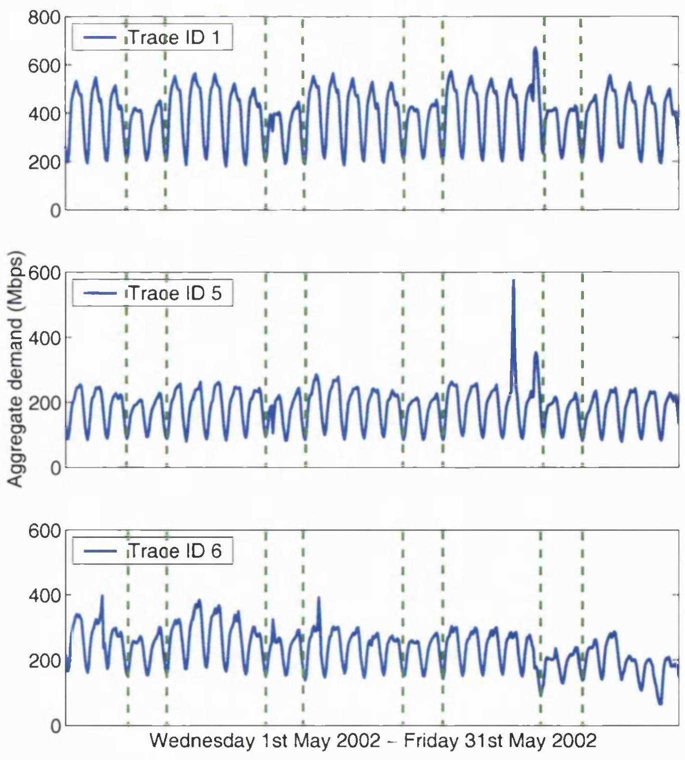


Figure 6.2: Aggregate demand in May 2002 for Traces 1, 5, and 6.

From the same figure we can see that different PoP pairs exhibit different behaviours as far as their aggregate demand is concerned. A *long term trend* is clearly visible in the traces over the observation period. For trace 1 and 5, this trend is increasing with time, while for trace 6 it looks more constant with a sudden shift in January 2002, that lasts for two months.

Shorter-term fluctuations around the overall long term trend are also present across all traces, and manifest themselves in different ways. For instance, trace 1 shows an increasing *deviation* around its long term trend. On the other hand, trace 6 exhibits smaller fluctuations, that look consistent over time.

Regardless of the differences observed in the three traces, one common property is the presence of large spikes throughout them. Notice that those spikes correspond to average values across 90 minutes, which indicate a surge of traffic in that particular interval that is high or constant enough to have a significant effect on a 90 minute average. Those spikes may correspond to link failures, which re-route part of the affected traffic onto this particular path, routing changes, or even denial of service attacks. As mentioned in Section 6.2, we decide to treat those spikes as outliers. This does not mean we ignore the data but simply that we do not attempt to model or predict these spikes.

In Figure 6.2 we present a detail of Figure 6.1, which corresponds to the month of May 2002. This figure indicates the presence of strong daily and weekly cycles. The drop in traffic during the weekend (denoted by the dashed lines) may be substantial as in trace 1, smaller as in trace 5, or even non-existent as in parts of trace 6.

From the previous observations it is clear that there are strong periodicities in the data. In order to verify their existence, we calculate the Fourier transform for the eight traces at our disposal. Our results indicate that the most dominant period across all traces is 24 hours. Other noticeable periods correspond to 12, and 168 hours (i.e. weekly period). Figure 6.3 presents the Fast Fourier Transform for three of our traces. One can notice that when there is a strong 12 hour period, then it is usually stronger than the weekly one. However, depending on the trace, such periods may not even be present[2].

In summary, initial observations from the collected time series lead to three main findings: 1) there is a multi-timescale variability across all traces (traces vary in different ways at different time scales), 2) there are strong periodicities in the data, and 3) the time series exhibit evident long-term trends, i.e. non-stationary behaviour. Such findings can be exploited in the forecasting process. For instance, periodicities at the weekly cycle imply that the time series

[2]Trace 5, and 6, presented in the previous figures, exhibit similar behaviour to Trace 1. However, for Trace 6, the weekly period is stronger than the 12 hour one.

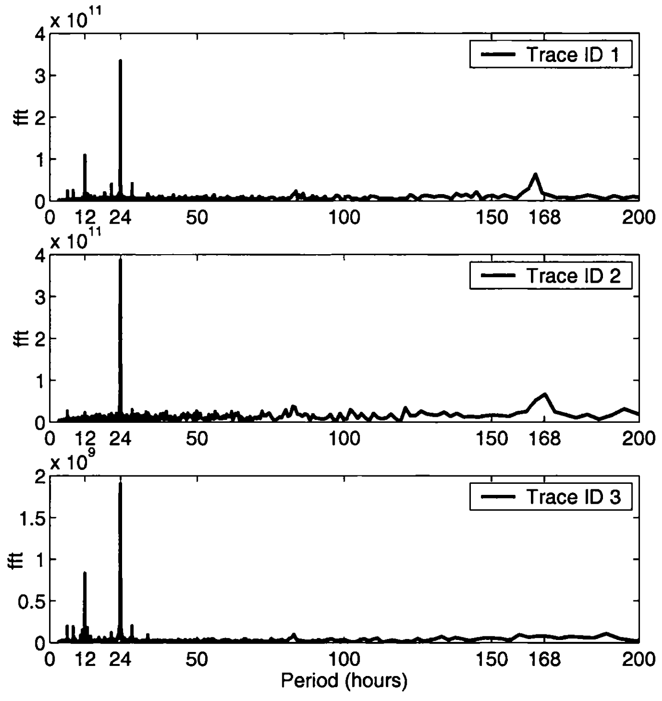


Figure 6.3: Fast Fourier Transform for Trace 1, 2, and 3.

behaviour from one week to the next can be predicted. In the next section, we address the three points in our findings.

6.5 Multi-timescale Analysis

In this section, we analyse the collected measurements at different time scales. We show that using the wavelet multiresolution analysis we can isolate the underlying overall trend, and those time scales that significantly contribute to its variability.

6.5.1 Wavelet MRA overview

The wavelet multiresolution analysis (MRA) describes the process of synthesising a discrete signal by beginning with a very low resolution signal (at the coarsest time scale) and successively adding on details to create higher resolution versions of the same signal [117, 118, 119]. Such a process ends with a complete synthesis of the signal at the finest resolution (at the finest time scale). More formally, at each time scale 2^j, the signal is decomposed into an *approximate* signal (or simply, *approximation*) and a *detailed* signal through a series of scaling functions $\phi_{j,k}(t)$ and wavelet functions $\psi_{j,k}(t)$, where $k \in Z$ is a time index at scale j. The scaling and wavelet functions are obtained by dilating and translating the mother scaling function $\phi(t)$, $\phi_{j,k}(t) = 2^{-j/2}\phi(2^{-j}t-k)$, and the mother wavelet function $\psi(t)$, $\psi_{j,k}(t) = 2^{-j/2}\psi(2^{-j}t-k)$. The approximation is represented by a series of (scaling) coefficients $a_{j,k}$, and the detail by a series of (wavelet) coefficients $d_{j,k}$.

Consider a signal/time series $x(t)$ with N data points at the finest time scale. Using MRA, $x(t)$ can be written as

$$x(t) = \sum_{k \in Z} a_{p,k} \phi_{p,k}(t) + \sum_{0 \le j \le p} \sum_{k \in Z} d_{j,k} \psi_{j,k}(t) \tag{6.1}$$

where $p \le \log N$. The sum with coefficients $a_{p,k}$ represents the approximation at the coarsest time scale 2^p, while the sums with coefficients $d_{j,k}$ represent the details on all the scales between 0 and p.

Using the signal processing parlance, the roles of mother scaling and wavelet function $\phi(t)$ and $\psi(t)$ can be described and represented via a *low-pass* filter h and a *high-pass* filter g [119]. Consequently, the multiresolution analysis and synthesis of a signal $x(t)$ can be implemented efficiently as a filter bank. The approximation at scale j, $\{a_{j,k}\}$, is passed through the low-pass filter h and the high-pass filter g to produce the approximation, $\{a_{j+1,k}\}$, and the detail, $\{d_{j+1,k}\}$, at scale $j+1$. Note that at each stage, the number of coefficients at scale j is decimated into half at scale $j+1$, due to downsampling. This decimation reduces the number of data points to be processed at coarser time scales, but also leaves some "artifacts" in coarser time scale approximations.

More recently, the so-called *à-trous wavelet transform* has been proposed, which produces "smoother" approximations by filling the "gap" caused by decimation, using redundant information from the original signal [120, 121]. Under the à-trous wavelet transform, we define the approximations of $x(t)$ at different scales as:

$$c_0(t) = x(t) \tag{6.2}$$

$$c_j(t) = \sum_{l=-\infty}^{\infty} h(l) c_{j-1}(t + 2^{j-1} l). \tag{6.3}$$

where $1 \le j \le p$, and h is a low-pass filter with compact support[3]. The detail of $x(t)$ at scale j is given by

$$d_j(t) = c_{j-1}(t) - c_j(t). \tag{6.4}$$

Let $d_j = \{d_j(t), 1 \le t < N\}$ denote the wavelet coefficient at scale j, and $c_p = \{c_p(t), 1 \le t < N\}$ denote the signal at the lowest resolution, often referred to as the residual. Then the set $\{d_1, d_2, \cdots, d_p, c_p\}$ represents the wavelet transform of the signal up to the resolution level p, and the signal $x(t)$ can be expressed as an expansion of its wavelet coefficients:

$$x(t) = c_p(t) + \sum_{j=1}^{p} d_j(t) \tag{6.5}$$

[3]Compact support means that the basis functions are non-zero only on a finite interval.

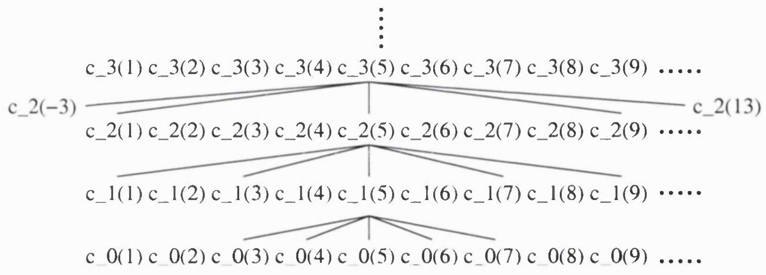
At this point we can justify our decision about averaging our measurements across 90 minutes intervals. We know that using the wavelet MRA we can look into the properties of the signal at time scales 2^j times coarser than the finest time scale. Furthermore, the collected measurements exhibit strong periodicities at the cycle of 12 and 24 hours. Using 1.5 hours as the finest time scale allows us to look into the behaviour of the time series at the periods of interest by observing its behaviour at the 3rd ($2^3 \times 1.5 = 12$) and 4th ($2^4 \times 1.5 = 24$) time scale.

6.5.2 MRA application on inter-PoP aggregate demands

For the smoothing of our data we chose as the low-pass filter h in Equation (6.3) the B_3 spline filter, defined by (1/16, 1/4, 3/8, 1/4, 1/16). This is of compact support (necessary for a wavelet transform), and is point-symmetric. Symmetric wavelets have the advantage of avoiding any phase shifts; the wavelet coefficients do not "drift" relative to the original signal. The B_3 spline filter gives at each resolution level a signal which is much smoother than the one at the previous level without distorting possible periodicities in the data, and preserving the original structure. The B_3 spline filter has been previously used in time series smoothing in [122, 123, 124].

In order to understand how $c_j(t)$ is computed at each time scale j, we schematically present in Figure 6.4 how $c_1(5)$, $c_2(5)$, and $c_3(5)$ are calculated according to Equation (6.3), and the B_3 spline filter. Element $c_1(5)$ is computed based on the values $c_0(t) = x(t)$ at times $(5-2)$, $(5-1)$, 5, $(5+1)$, and $(5+2)$. Then, we can calculate $c_2(5)$, based on $c_1(1)$, $c_1(3)$, $c_1(5)$, $c_1(7)$, and $c_1(9)$. Notice that moving toward coarser levels of resolution we need values from the previous resolution level which are farther apart from each other. For this reason, this wavelet transform is called the à-trous wavelet transform, which means "with holes". One important point we should make is that $c_p(t)$ is defined for each $t = 1, 2, \cdots, n$, where n corresponds to 1.5 hour intervals and is limited by the size N of the original signal. According to Equation (6.3), computing $c_p(n)$ requires values of c_{p-1} until time $n + 2^p$, which iteratively requires values of c_{p-2} until time $n + 2^{p-1}$, etc. As a consequence, the calculation of $c_p(n)$ requires the original time series $x(t)$ to have $n + \sum_{j=1}^{j=p} 2^j$ values. Given that our original signal contains N values, our wavelet coefficients up to the 6th resolution level will contain n values, where $n + \sum_{j=1}^{j=6} 2^j = N$, or $n = N - 126$.

In Figure 6.5 and Figure 6.6 we present the approximation and detail signals for trace 5 at each time scale, when it is analysed up to resolution level $2^6 = 96$ hours. We chose to use the 6th time scale as our coarsest time scale because it provides a sufficiently smooth approximation signal, capturing the evolution of the time series from one week to the next without the effect of the fluctuations at the 12 and 24 hour time scale. Figure 6.5 clearly shows how the wavelet



c_3(1) c_3(2) c_3(3) c_3(4) c_3(5) c_3(6) c_3(7) c_3(8) c_3(9) .....
c_2(−3)
c_2(13)
c_2(1) c_2(2) c_2(3) c_2(4) c_2(5) c_2(6) c_2(7) c_2(8) c_2(9) .....
c_1(1) c_1(2) c_1(3) c_1(4) c_1(5) c_1(6) c_1(7) c_1(8) c_1(9) .....
c_0(1) c_0(2) c_0(3) c_0(4) c_0(5) c_0(6) c_0(7) c_0(8) c_0(9) .....

Figure 6.4: The *à trous* wavelet transform.

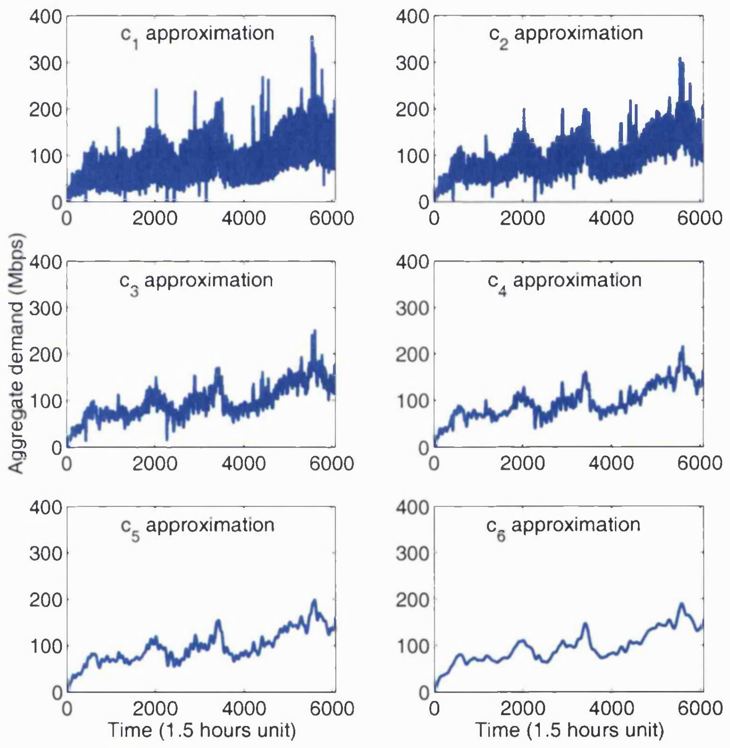


Figure 6.5: The approximation signals for trace 5.

MRA smoothes out the original signal. Visual inspection of the derived detail signals in Figure 6.6 further suggests a difference in the amount of variability that each one contributes.

Given the derived decomposition, we calculate the energy apportioned to the overall trend (c_6) and each one of the detail signals. The energy of a signal $y(t)$, $1 \leq t \leq N$, is defined as $E = \sum_{t=1}^{N} y^2(t)$. Table 6.1 shows that the overall trend c_6 accounts for 95 to 97% of the total energy. Once one subtracts the overall trend from the data, then we notice a substantial difference in the amount of energy distributed among the detail signals. Figure 6.7 shows that across all eight traces in our study, the maximum amount of energy in the details is always located at the 3rd time scale, which corresponds to the fluctuations across 12 hours. Approximating the

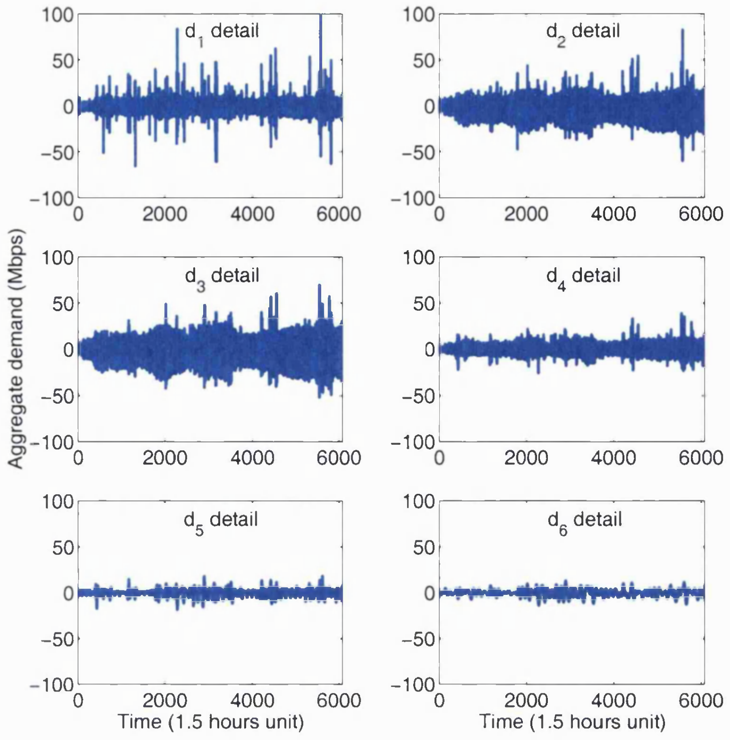


Figure 6.6: The detail signals for trace 5.

original signal as the long term trend, c_6, and the fluctuations at the 12 hour time scale, d_3, is further capable of accounting for 97 to 99% of the total energy (Table 6.1).

Trace ID	1	2	3	4
c_6	96.07%	97.20%	95.57%	96.56%
$c_6 + d_3$	98.10%	98.76%	97.93%	97.91%
Trace ID	5	6	7	8
c_6	95.12%	95.99%	95.84%	97.30%
$c_6 + d_3$	97.54%	97.60%	97.68%	98.45%

Table 6.1: Percentage of total energy in c_6, and $c_6 + d_3$.

In the next section, we look into the properties of the signals derived from the wavelet MRA with respect to the variance they account for in the overall signal.

6.5.3 Analysis of Variance

As explained in Section 6.5.1, the original signal can be completely reconstructed using the approximation signal at the 6th time scale, and the six detail signals at lower time scales. The

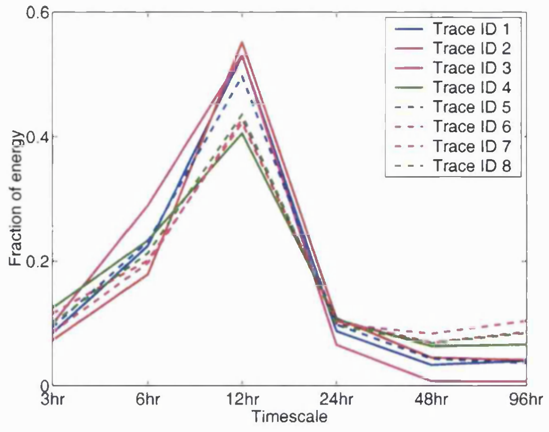


Figure 6.7: Energy distribution for the detail signals.

model defined in Equation (6.5) can also be conceived as a multiple linear regression model, where the original signal $x(t)$ is expressed in terms of its coefficients.

The "Analysis of Variance" (ANOVA) technique is a statistical method used to quantify the amount of variability accounted for by each term in a multiple linear regression model [125]. Moreover, it can be used in the reduction process of a multiple linear regression model, identifying those terms in the original model that explain the most significant amount of variance.

Using the ANOVA methodology we calculate the amount of variance in the original signal explained by the 6th approximation signal and each one of the detail signals. The results indicate that the detail signals d_1, d_2, d_5, and d_6 contribute less than 5% each in the variance of the original signal.

Ideally, we would like to reduce the model of Equation (6.5) to a simple model of two parameters, one corresponding to the overall long term trend, and a second one accounting for the bulk of the variability. Possible candidates for inclusion in the model, except from the overall trend c_6, are the signals d_3 and d_4. We know that the detail signal d_3 carries the majority of the energy among all the detail signals. Thus one possible reduced model is the following:

$$x(t) = c_6(t) + \beta\, d_3(t) + e(t) \tag{6.6}$$

Using least squares, we calculate the value of β for each one of the traces in our disposal. All traces led to a β estimate between 2.1 and 2.3 (Table 6.2). Using ANOVA, we test how representative the model of Equation (6.6) is with respect to the proportion of variance it explains [125].

If $x(t)$ is the observed response, and $e(t)$ is the error incurred in Equation (6.6), we define $SSX = \sum_{t=1}^{n}(x(t))^2$, $SSE = \sum_{t=1}^{n} e(t)^2$. The total sum of squares (SST) is defined as the

Trace ID	1	2	3	4
β	2.09	2.06	2.11	2.23
R^2	0.87	0.94	0.89	0.87
Trace ID	**5**	**6**	**7**	**8**
β	2.12	2.18	2.13	2.16
R^2	0.92	0.80	0.86	0.91

Table 6.2: ANOVA results for all eight traces.

uncertainty that would be present if one had to predict individual responses without any other information. The best one could do is predict each observation to be equal to the sample mean. Thus we set $SST = \sum_{t=1}^{n}(x(t) - \bar{x})^2$. The ANOVA methodology partitions this variability into two parts. One portion is accounted for by the model. It corresponds to the reduction in uncertainty that occurs when the regression model is used to predict the response. The remaining portion is the uncertainty that remains even after the model is used. We define SSR as the difference between SST and SSE. This difference represents the sum of the squares explained by the regression. The fraction of the variance that is explained by the regression, SSR/SST, determines the goodness of the regression and is called the "Coefficient of Determination", R^2. The model is considered to be statistically significant if it can account for a large fraction of the variability in the response, i.e. yields large values for R^2. In Table 6.2, we present the results obtained for the value of β, and R^2 for all eight traces.

The reduced model is capable of explaining 80% to 94% of the variance in the signal. Moreover, if we decide to include the term d_4 in the model of Equation (6.6), the results on R^2, presented in Table 6.2, are only marginally improved, and increased by 0.01 to 0.04.

6.5.4 Summary of findings from MRA and ANOVA

From the wavelet multiresolution analysis, we draw three main conclusions:

- There is a clear overall long-term trend present in all traces.
- The fluctuations around this long term trend are mostly due to the significant changes in the traffic bandwidth at the time scale of 12 hours.
- The long term trend and the detail signal at the 3rd time scale account for approximately 98% of the total energy in the original signal.

From the Analysis of Variance, we further conclude that:

- The largest amount of variance in the original signal can be explained by its long term trend c_6 and the detail signals d_3, and d_4 at the time scales of 12 and 24 hours respectively.
- The original signal can be sufficiently approximated by the long term trend and its third detail signal. This model explains approximately 90% of the variance in the original signal.

Based on those findings, we derive a generic model for our time series, presented in Equation (6.7). This model is based on Equation (6.6), where we set $\beta = 3$, for a model valid across the entire backbone.

$$x'(t) = c_6(t) + 3\, d_3(t) \tag{6.7}$$

6.5.5 Implications for modelling

For forecasting purposes at the time scale of weeks and months, one may not need to accurately model all the short term fluctuations in the traffic. More specifically, for capacity planning purposes, one only needs to know the traffic baseline in the future along with possible fluctuations of the traffic around this particular baseline.

Component $d_3(t)$ in the model of Equation (6.7) is defined for every 90 minutes interval in the measurements capturing in time the short-term fluctuations at the time scale of 12 hours. Given that the specific behaviour within a day may not be that important for capacity planning purposes, we calculate the standard deviation of d_3 within each day. Furthermore, since our goal is not to forecast the exact amount of traffic on a particular day months in the future, we calculate the weekly standard deviation $dt_3(j)$ as the average of the seven values computed within each week. Such a metric represents the fluctuations of the traffic around the long term trend from day to day within each particular week.

In Figure 6.8 we show the aggregate demand for trace 5, as calculated from the SNMP data. In the same figure we plot the long term trend in the data, along with two curves showing the approximation of the signal as the sum of the long term trend plus/minus three times the average daily standard deviation within a week (Equation (6.7)). We see that approximating the original signal in such a way exposes the fluctuations of the time series around its baseline with sufficient accuracy.

Notice that the new signal dt_3 features one value every week, exposing the average daily standard deviation within the week. Similarly, we can approximate the long term trend with a more compact time series featuring one value for every week. Given that the 6th approximation signal is a very smooth approximation of the original signal, we calculate its average across

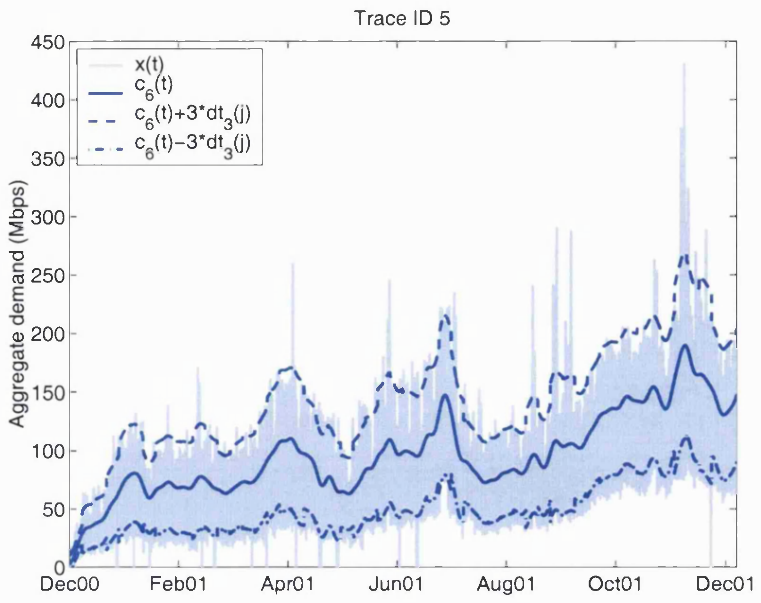


Figure 6.8: Approximation of the signal using $c_6(t)$ and the average daily standard deviation within a week $dt_3(j)$.

each week, and create a new time series $l(j)$ capturing the long term trend from one week to the next. The forecasting process will have to predict the behaviour of

$$\hat{x}(j) = l(j) + 3\, dt_3(j), \tag{6.8}$$

where j denotes the index of each week in our trace. The resulting signal is presented in Figure 6.9. We confirm that approximating the original signal using weekly average values for the overall long term trend, and the daily standard deviation results in a model which accurately captures the desired behaviour.

In the next section, we introduce the linear time series models, and show how they can help derive forecasts for the two identified components. Once we have those forecasts, we compute the forecast for the original time series and compare it with collected measurements.

6.6 Time Series Analysis using the ARIMA model.

6.6.1 Overview of linear time series models

Constructing a time series model implies expressing X_t in terms of previous observations X_{t-j}, and noise terms Z_t which typically correspond to external events. The noise processes Z_t are assumed to be uncorrelated with a zero mean and finite variance. Such processes are the simplest processes, and are said to have "no memory", since their value at time t is uncorrelated with all the past values up to time $t-1$.

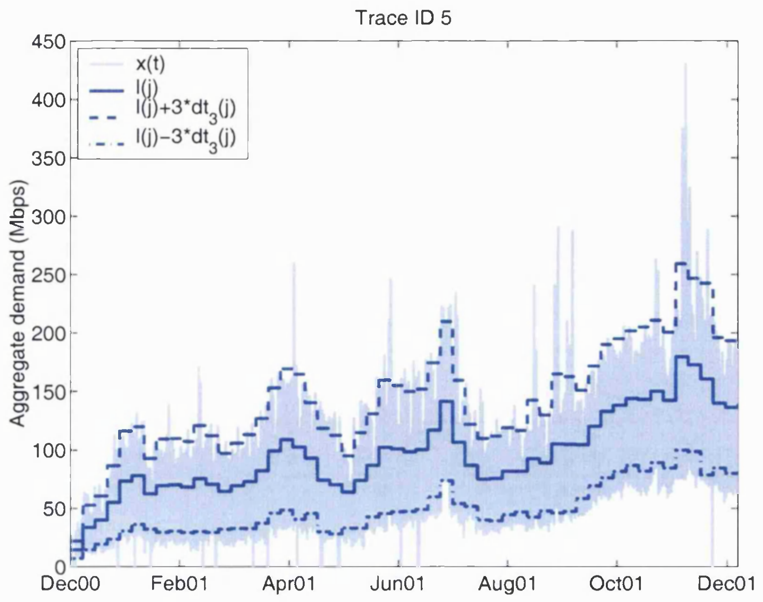


Figure 6.9: Approximation of the signal using the average weekly long term trend $l(j)$ and the average daily standard deviation within a week $dt_3(j)$.

Most forecasting models described in the literature are linear models. From those models, the most well-known are the "Autoregressive" (AR), "Moving Average" (MA), and "Autoregressive Moving Average" (ARMA) models.

A time series X_t is an ARMA(p,q) process if X_t is stationary and if for every t

$$X_t - \phi_1 X_{t-1} - \cdots - \phi_p X_{t-p} = Z_t + \theta_1 Z_{t-1} + \cdots + \theta_q Z_{t-q}$$

where $Z_t \sim WN(0, \sigma^2)$ and the polynomials $(1 - \phi_1 z - \cdots - \phi_p z^p)$ and $(1 + \theta_1 z + \cdots + \theta_q z^q)$ have no common factors [126]. If $p = 0$, then the model reduces to a pure MA process, while if $q = 0$, the process reduces to a pure AR process.

This equation can also be written in a more concise form as:

$$\phi(B)X_t = \theta(B)Z_t \tag{6.9}$$

where $\phi(\cdot)$, and $\theta(\cdot)$ are the p^{th} and q^{th} degree polynomials, and B is the backward shift operator ($B^j X_t = X_{t-j}, B^j Z_t = Z_{t-j}, j = 0, \pm 1, \cdots$).

The ARMA model fitting procedure assumes *the data to be stationary*. If the time series exhibits variations that violate the stationary assumption, then there are specific approaches that could be used to render the time series stationary. The most common one is what is often called the "differencing operation". We define the lag-1 difference operator ∇ by

$$\nabla X_t = X_t - X_{t-1} = (1 - B)X_t, \tag{6.10}$$

where B is the backward shift operator as already introduced. If the non stationary part of a time series is a polynomial function of time, then differencing finitely many times can reduce the time series to an ARMA process.

An ARIMA(p,d,q) model is an ARMA(p,q) model that has been differenced d times. Thus it has the form:

$$\phi(B)(1-B)^d X_t = \theta(B)Z_t, \; Z_t \sim WN(0,\sigma^2) \tag{6.11}$$

If the time series has a non-zero average value through time, then the previous equation also features a constant term μ on its right hand side.

6.6.2 Time series analysis of the long-term trend and deviation

In order to model the obtained components $l(j)$ and $dt_3(j)$ using linear time series models, we have to separate the collected measurements into two parts: 1) one part used for the estimation of the model parameters, and 2) a second part used for the evaluation of the performance of the selected model. Since our intended application is capacity planning, where traffic demand has to be predicted several months ahead in the future, we select the estimation and evaluation period such that the latter contains six months of data.

For each one of the analysed traces, we use the measurements collected up to 15th January 2002 for the modelling phase, and the measurements from 16th January 2002 until 1st July 2002 for the evaluation phase. Given that not all time series are of the same duration, the isolation of the last six months for evaluation purposes may lead to specific traces featuring a small number of measurements for the estimation phase. Indeed, after posing this requirement three out of the eight traces in our analysis (Trace 2, 3, and 7) consist of less than six months of information. Such limited amount of information in the estimation period does not allow for model convergence. As a consequence, we continue our analysis on the five traces remaining.

We use the Box-Jenkins methodology to fit linear time series models [126]. Such a procedure involves the following steps: i) determine the number of differencing operations needed to render the time series stationary, ii) determine the values of p, and q in Equation (6.9), iii) estimate the polynomials ϕ, and θ, and iv) evaluate how well the derived model fits the data. For the model fitting we used both Splus [127] and ITSM [126], and obtained similar results. The estimation of the model parameters is done using Maximum Likelihood Estimation. The best model is chosen as the one that provides the smallest AICC, BIC, and FPE measures [126], while offering the smallest mean square prediction error six months ahead. Details about the metrics used in the quality evaluation of the derived models are presented in Appendix B. One point we should emphasise is that metrics like AICC, and BIC not only evaluate the fit between the values predicted by the model and actual measurements, but also penalise models with large number of parameters. Therefore, the comparison of the derived models against such metrics leads to the most parsimonious models fitting the data.

6.6.3 Models for $l(j)$, and $dt_3(j)$

The computed models for the long term trend $l(j)$ indicate that the first difference of those time series (i.e. the time series of their changes) is consistent with a simple MA model with one or two terms (i.e, $d = 1, q = 1$ *or* $d = 1, q = 2$), plus a constant value μ (Table 6.3). The need for one differencing operation at lag 1, and the existence of term μ across all the models indicate that the long-term trend across all the traces is a simple exponential smoothing with growth. The trajectory for the long-term forecasts will typically be a sloping line, whose slope is equal to μ. For instance, for trace 1 the long-term forecast will correspond to a weekly increase of 0.5633 Mbps. This forecast corresponds to the average aggregate demand of a link in the aggregate. The weekly increase in the total demand between two adjacent PoPs can thus be estimated through the multiplication of this value with the total number of active links in the aggregate. Given the estimates of μ across all models in Table 6.3 we conclude that all traces exhibit upward trends, but grow at different rates.

Applying the Box-Jenkins methodology on the deviation measurements, we see that for some traces the deviation $dt_3(j)$ can be expressed with simple AR models (Trace 4, and 6), while the remaining can be accurately modelled as MA processes after one differencing operation (Table 6.4). Therefore, the deviation for traces 1, 5, and 8 increases with time (at rates one order of magnitude smaller than the increase in their long term trends), while the deviation for traces 4, and 6 can be approximated with a weighted moving average, which indicates slower evolution. These results confirm earlier observations on Figure 6.1 in Section 6.4.2.

From the previous tables we see that one cannot come up with a single network-wide forecasting model for the inter-PoP aggregate demand. Different parts of the network grow at different rates (long-term trend), and experience different types of variation (deviation from the long-term trend). Our methodology extracts those trends from historical measurements and can identify these PoP pairs in the network that exhibit higher growth rates and thus may require additional capacity in the future.

At this point we should note that the Box-Jenkins methodology could also have been applied on the original time series $x(t)$. However, given the existence of three strong periods in the data (which would require a seasonal ARIMA model with three seasons [126]), the variability of the time series at multiple time scales, the existence of outliers, and the size of the original time series, such an approach leads to highly inaccurate forecasts, while being ex-

ID	Order	Model	μ	σ^2
T1	(0,1,2)	$X(t) = X(t-1) + Z(t) - 0.1626Z(t-1) -$	0.5633E+06	0.2794E+15
		$0.4737Z(t-2)$		
T4	(0,1,1)	$X(t) = X(t-1) + Z(t) + 0.4792Z(t-1)$	0.4155E+06	0.1339E+15
T5	(0,1,1)	$X(t) = X(t-1) + Z(t) + 0.1776Z(t-1)$	0.2301E+07	0.1516E+15
T6	(0,1,2)	$X(t) = X(t-1) + Z(t) - 0.3459Z(t-1) -$	0.7680E+06	0.6098E+15
		$0.4578Z(t-2)$		
T8	(0,1,1)	$X(t) = X(t-1) + Z(t) + 0.2834Z(t-1)$	0.2021E+07	0.1404E+16

Table 6.3: ARIMA models for the long term trend.

ID	Order	Model	μ	σ^2
T1	(0,1,1)	$X(t) = X(t-1) + Z(t) - 0.6535Z(t-1)$	0.3782E+05	0.2024E+14
T4	(2,0,0)	$X(t) = 0.8041X(t-1) - 0.3055X(t-2) + Z(t)$	0.1287E+08	0.7295E+13
T5	(0,1,1)	$X(t) = X(t-1) + Z(t) - 0.1493Z(t-1)$	0.3094E+06	0.8919E+13
T6	(3,0,0)	$X(t) = 0.3765X(t-1) - 0.1964X(t-2) -$	0.2575E+08	0.3057E+14
		$0.2953X(t-3) + Z(t)$		
T8	(0,1,1)	$X(t) = X(t-1) + Z(t) - 0.5565Z(t-1)$	0.3924E+05	0.4423E+14

Table 6.4: ARIMA models for the weekly deviation.

tremely computationally intensive. Our technique is capable of isolating the overall long term trend and identifying those components that significantly contribute to its variability. Predictions based on weekly approximations of those components provide accurate estimates with a minimal computational overhead. All our forecasts were obtained in seconds.

In the next section, we use the derived models for the weekly prediction of the aggregate traffic demands. Our forecasts are compared against actual measurements.

6.6.4 Evaluation of forecasts

Using our models we predict a baseline aggregate demand for a particular week in the future, along with possible deviations around it. The overall forecast for the inter-PoP aggregate demand is then calculated based on Equation (6.8). We constrain ourselves to the upper limit alone, since this is the value that would be used for capacity planning purposes.

In Figure 6.10, we present the time series collected until July 1st 2002. On the same figure we present the modelled behaviour in the estimation period, and the forecasts in the evaluation period[4]. From visual inspection of the presented plot, one can conclude that the proposed

[4]In Figures 6.10, 6.12 and 6.13, the vertical dashed line indicates the beginning of the forecasting period.

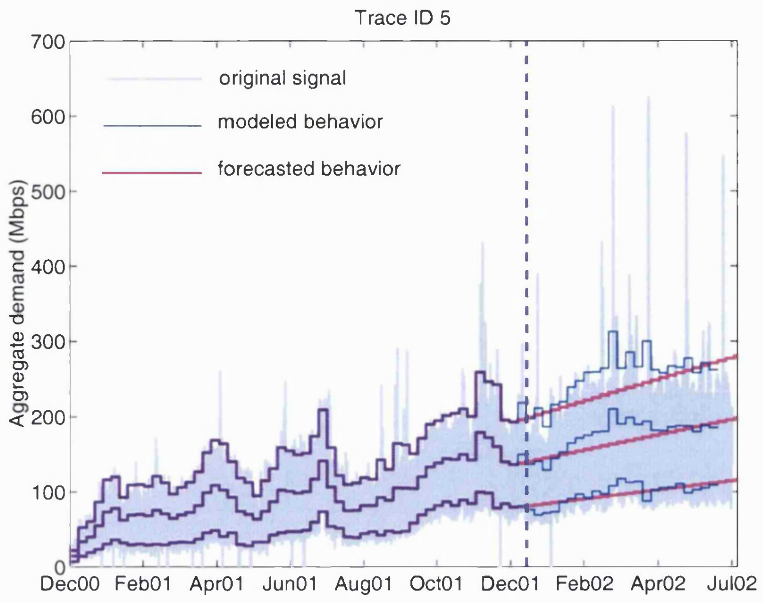


Figure 6.10: Six month forecast for Trace 5.

methodology behaves very well for this particular trace.

In order to be able to quantify the quality of the predictions with respect to the observed behaviour, we proceed as follows:

- We apply the MRA on the measurements in the evaluation period.
- We calculate the long term trend $l(j)$ and weekly deviation $dt_3(j)$ for each week in the same period.
- We compute $\hat{x}(j)$ based on Equation (6.8).
- Lastly, we calculate the error in the derived forecast as the forecasted value minus $\hat{x}(j)$, divided by $\hat{x}(j)$.

In Figure 6.11 we present the relative error between the derived forecast and $\hat{x}(j)$ for each week in the evaluation period. Negative error implies that the actual demand was higher than the one forecasted. As can be seen from the figure, the forecasting error fluctuates with time, but is centred around zero. This means that on average we neither underestimate nor overestimate the aggregate demand. More specifically, we see that 24 weeks in the future our prediction error is 4%. The average prediction error across weeks is -3.6%. Lastly, across all five traces, the average absolute relative prediction error is lower than 15%.

Notice that our forecasting models can be used to predict demand for more than six months in the future, and identify when the forecasted demand will exceed the operational thresholds

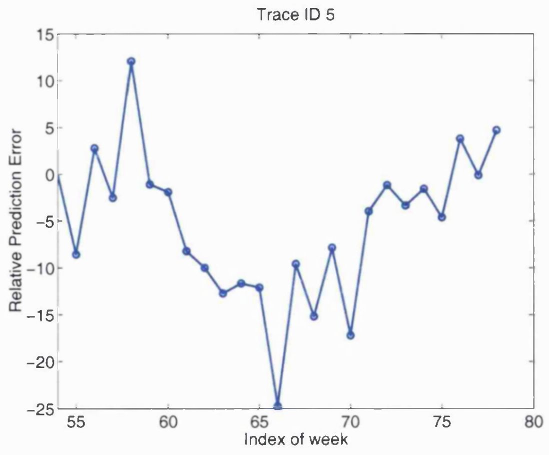


Figure 6.11: Weekly relative prediction error for Trace 5.

that will trigger link upgrades (as explained in section 6.3). In that case though forecasts should be used with caution. As is the case with any forecasting methodology, the farther ahead in the future one attempts to predict, the larger the error margin that should be allowed.

6.7 Discussion and Future Work

In the previous sections, we modelled the average aggregate demand between any two adjacent PoPs based on its long term trend and possible short-term fluctuations around it. The long term trend in the data is sensitive to long-lasting routing changes, topological changes, and market conditions that dictate the behaviour of the users in the Internet.

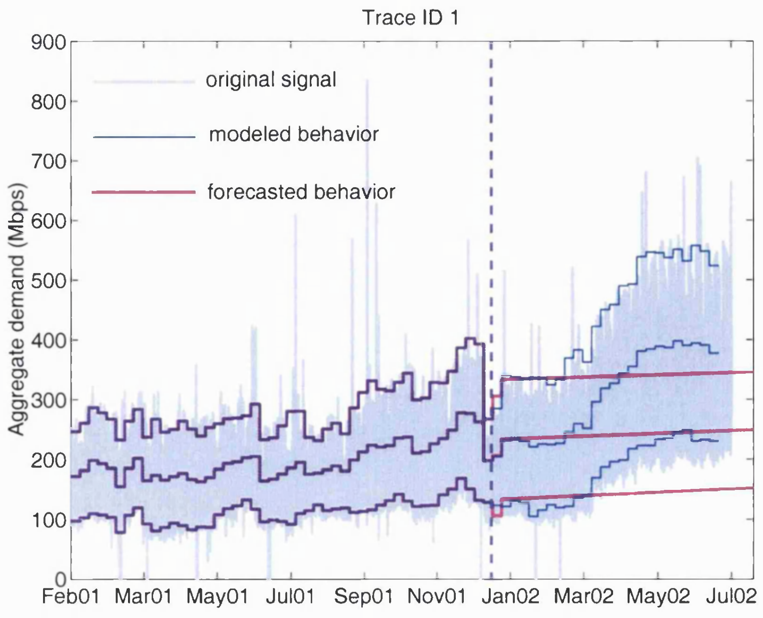


Figure 6.12: Weekly prediction for Trace 1.

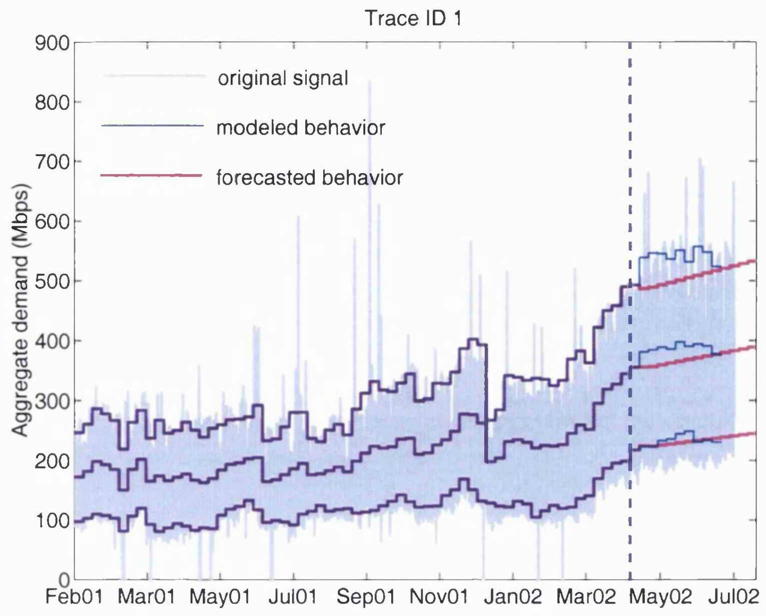


Figure 6.13: Adjusted weekly prediction for Trace 1.

Consequently, there will be cases when the fitted ARIMA models will have to be re-estimated in order to adjust to changes in the environment. One example is trace 1. This particular trace is the worst-performing trace in our dataset exhibiting a dramatic change in its overall long term trend right after April 2002. Our analysis uses data until January 15, 2002 to estimate the parameters of the ARIMA models, missing the change in trend that occurs in April. As a result our forecasts are highly accurate until April 2002 (with an average relative prediction error of 1%) but substantially smaller than the ones observed between April and July 2002 (Figure 6.12).

Nonetheless, if re-estimation of the model parameters is triggered at the end of April 2002, then the forecasts are much closer to the measured behaviour, and within 7% (Figure 6.13). Due to lack of SNMP information beyond July 2002, we cannot evaluate the accuracy of the new predictions in the six months horizon.

Future work will address the issue of detecting changes in the overall trend. Once such a change is detected, then our models can be applied as already shown. In fact, such a detection should not necessarily be automated. Network operators themselves could trigger the re-estimation of the derived ARIMA models a few months after significant topological and/or routing configuration changes have taken place. Within the topological changes, we also include the activation of new high-capacity customer links which may have a significant effect on the load in the core. Such changes would usually be known ahead of time. Consequently, in future work we also intend to incorporate marketing information in our model and evaluate its effect on our forecasts.

6.8 Conclusions

In this chapter, we presented a methodology for predicting *when and where* link upgrades/additions have to take place in the core of an IP network. We measured aggregate demand between any two neighbouring PoPs in the core of a major tier-1 IP network, and analysed its evolution at time scales larger than one hour.

We showed that the derived time series exhibit strong periodicities at the cycle of 12, and 24 hours, as well as one week. Moreover, they experience variability at multiple time scales, and feature distinct overall long-term trends.

Using wavelet MRA, we isolated the *overall long term trend*, and analysed variability at multiple time scales. We showed that the largest amount of variability in the signal comes from its *fluctuations at the 12 hour time scale*. Our analysis indicates that a parsimonious model consisting of those two identified components is capable of capturing 98% of the total energy in the original signal, while explaining 90% of its variance. The resulting model is capable of revealing the behaviour of the network traffic through time, filtering short-lived events that may cause traffic perturbations beyond the overall trend.

We showed that the weekly approximations of the two components in our model can be accurately modelled with low-order ARIMA processes. Our results indicate that different parts in the network grow at different rates, and may also experience increasing deviations from their overall trend, as time progresses. We further showed that calculating future demand based on the forecasted values for the two components in our traffic model yields highly accurate estimates. Our average relative forecasting error is less than 15% for at least six months in the future.

Due to the properties of the collected time series direct application of traditional time series analysis techniques proves cumbersome, computationally intensive and prone to error. Our methodology is simple to implement, and can be fully automated. Moreover, it provides accurate forecasts for at least six months in the future with a minimal computational overhead. In this chapter, we demonstrated its performance within the context of capacity planning. However, multiresolution analysis of the original signal and modelling of selected approximation and detail signals using ARIMA models could possibly provide accurate forecasts for the behaviour of the traffic at other time scales, such as from one day to the next or at a particular hour on a given day in the future. These forecasts could be useful for other network engineering tasks like scheduling of maintenance windows or large database network backups.

The work presented in this chapter is the result of collaboration with Nina Taft, Zhi-Li Zhang, and Christophe Diot. Zhi-Li Zhang suggested the use of wavelets for the detrending of the collected timeseries, and helped with the theoretical aspects of the wavelet decomposition. Nina Taft supervised this project and supplied help throughout the duration of this work. All three collaborators commented on the writing. However, the methodological approach presented, all the code, and the interpretation of the results have been carried out by me.

Chapter 7

Contributions and Future Work

Current IP backbone network design and provisioning practices are heavily dependent on the intuition of the human operators. Traffic variability, scalability issues, lack of monitoring information, and complex interactions between inter- and intra-domain routing protocols result in network management techniques which usually rely on trial and error. The central goal of this thesis was to demonstrate the benefits of using different types of monitoring information in the formalisation of different network provisioning tasks, and provide a methodological framework for their analysis. Our main contribution can be summarised as follows:

Current IP backbone provisioning practices can be greatly enhanced based on measurements, but require a diverse set of monitoring information collected or processed at multiple granularities of time and traffic aggregation.

We collect four main sources of monitoring information from the Sprint operational IP backbone network, and analyse them offline. These four sources correspond to (i) *GPS-synchronised packet traces* listing the first 44 bytes of every packet traversing a monitored link, (ii) *BGP routing table dumps* collected from a BGP router inside each one of the four PoPs where IPMON systems have been installed, (iii) *network-wide SNMP* information on every operational link of the Sprint IP backbone, and (iv) *topological information* listing all the active interfaces of each router in the network, along with their capacity, their IP addresses, and the name of the router and interface terminating the link at the other end. In what follows we summarise our contributions, discuss possible limitations and alternative approaches to addressing the problems at hand. Finally, we suggest possible directions for future research.

7.1 Contributions

7.1.1 Single-hop delay analysis

In Chapter 4, we focused on the evaluation of the queueing delay experienced by packets through a single backbone router. Using the collected topological information, we identified the subset of the links monitored by IPMON systems that are attached to the same router inside the backbone. We then *matched* the packet traces collected on links connected to the same router in order to identify their common packets. The outcome of this process is a trace containing all the packets present in both traces, along with accurate timestamps for their arrival and departure at the respective router. Subtracting the arrival timestamp from the departure timestamp allows us to compute the total amount of time each packet spends inside the router, which we call "single-hop delay".

We presented our single-hop delay measurements and proposed a methodology for the identification of its contributing factors. We showed that in addition to packet processing, transmission at the output link, and queueing at the output queue, packets may face large delays that cannot be justified within the context of a pure FIFO output queue. Using a single output queue model, we isolated those delays and provided possible explanations for the events that may have led to such extreme delay values through a single node.

Our contributions have been:

- We measured and presented the first actual single-hop delay measurements from an operational backbone network.
- We introduced a methodology able to identify the different factors contributing to single-hop delay. The modelling of the different factors in single-hop delay can be used in simulation environments providing a more realistic model for backbone routers.
- We showed that packets experience minimal queueing in the backbone. A well-expected result, given the amount of overprovisioning present in current backbone networks.
- We showed that router idiosyncrasies, however, may increase single-hop delays to the order of milliseconds. This behaviour was shown to resemble a "coffee break" inside the router.
- We analysed the tail behaviour of the queueing delay distribution and showed that it can be modelled with a Weibull distribution, as shown in previous analytical studies.
- The measured average queueing delay is greater than predicted by M/M/1 and FBM models when the link utilisation is below 70%, but its absolute value is quite small.

We conclude that delays faced by packets while transiting the Internet core are dominated by the propagation delays. As long as ISPs maintain their links moderately utilised, end-to-end delays will mainly depend upon the physical paths taken throughout the network. However, link and equipment failures are events in the lifetime of a backbone network, that are rather frequent and lead to unpredictable resulting network state. Such phenomena are usually accompanied by transient overload on the links selected by routing protocols to take over the traffic previously accommodated by the failed links. Under such circumstances, network operators usually interfere manually, increasing the IS-IS link weight of the overloaded link, so as to divert traffic to other, less utilised parts of the network. However, given that the network operator does not have knowledge of the traffic matrix of his network, and therefore cannot simulate the effect of his action, such a change in the link weight is not unlikely to cause overload on other paths throughout the network.

7.1.2 Identifying persistent elephant flows

In Chapter 5, we looked into a controlled way of offloading heavily utilised links by re-routing the high-volume destination network-prefix flows traversing the congested link. In order to be able to perform such an operation, the network operator needs to be able to identify those destination network-prefix flows that account for the majority of the traffic on the link. Moreover, in order for such an operation to have the targeted effect, the selected flows need to possess sufficient bandwidth persistence in time. In Chapter 5, we proposed a methodology to identify these high-volume flows on a link, that are more likely to contribute significant amounts of load persistently over time.

In our analysis we used BGP routing table dumps collected from two PoPs in the Sprint IP network, and packet traces collected from OC-12 links in the respective PoPs. Similar analysis has also been performed on OC-3 and OC-48 packet traces. We defined flows at the granularity of a destination network prefix and measured their bandwidth at fixed intervals. We defined as elephants a small number of flows that contributes high volumes of traffic persistently over time. We proposed two "single-instant" classification schemes for the computation of a threshold rate that separates flows into two classes according to their properties during a single time interval. We showed that classifying flows based on their "instantaneous" bandwidth leads to high volatility in the resulting classification. We concluded that successful identification of the persistent elephants needs to account for their temporal behaviour. We defined a new metric, called "latent heat" to capture the temporal behaviour of a flow. We showed that classification schemes accounting for the elephants' latent heat are more successful at identifying elephants with the desirable properties.

Our contributions can be summarised as follows:

- We analysed packet traces in conjunction to BGP routing tables, and showed that destination network-prefix flows exhibit the behaviour usually referred to as the "elephant and mice" phenomenon. A small number of flows is responsible for the majority of the load.

- We identified bandwidth persistence in time as an important property of elephant flows used in a traffic engineering context.

- We proposed two "single-instant" classification schemes for the separation of the flows into two classes according to their bandwidth during a single time interval. These techniques make no assumptions about how high or low the separation threshold should be, but capitalise on the properties of the collected measurements for its derivation.

- We proposed four performance metrics for the evaluation of the proposed schemes. Two of these metrics are specifically designed so that they evaluate the persistence of the resulting classification in time.

- We showed that classifying flows according to their "instantaneous" behaviour leads to highly volatile classes of traffic. This volatility was illustrated in three ways: (i) we found that flows remain in the elephant state for surprisingly short periods of time; (ii) the number of elephants varies considerably over different time intervals; (iii) if elephants are not reclassified frequently, the total load in the elephant class can change significantly.

- We proposed a new classification scheme, which allows flows to experience short-lived bursts or drops across the separation threshold. We defined a new per-flow metric, called *latent heat*, which tracks the evolution of the difference between the flow's bandwidth and the threshold computed by the single-instant classification schemes. We demonstrated that classification of flows according to their latent heat leads to significant improvement in the volatility of the elephant class. Under our latent heat classification scheme, the number of elephants identified exhibits smaller fluctuations, the captured load is reasonably consistent in time, while the average holding time in the elephant state has increased from 30 minutes to more than 1.5 hours.

Re-routing elephant destination network prefixes can be deployed as a solution to overcome congestion only in cases when the observed congestion is a transient phenomenon. Persistent congestion within the network will need to be addressed with further provisioning, in the shape of link additions or link upgrades. However, given that circuit provisioning cycles can last six months, network operators need to be able to identify the parts of the network that may need additional provisioning as early as possible.

7.1.3 Forecasting Backbone Network Traffic

In Chapter 6, we analysed three years worth of SNMP information collected from the entire network. This SNMP information reports on the utilisation of every operational link in the backbone at 5 minute intervals. We combined the SNMP information with topological information and computed the average aggregate traffic demand of a link between any two adjacent PoPs. We proposed a methodology that can indicate *when* and *where* further provisioning will be needed in the core of an IP backbone network. With minimal computational overhead our technique is capable of producing predictions within 15% of the actual measurements for at least six months in the future.

Our main contributions can be summarised as follows:

- We presented the first analysis of Internet backbone traffic at time scales larger than one hour.
- We showed that Internet backbone traffic can be modelled for capacity planning purposes using a parsimonious model with only two parameters. These parameters are the *long term trend* present in the evolution of inter-PoP average aggregate demand, along with its shorter-term *fluctuations at the 12 hour time scale.* The proposed model is able to account for 98% of the total energy in the original signal, while explaining 90% of its variance.
- We demonstrated that the weekly approximations of the two identified components can be accurately modelled with low-order AutoRegressive Integrated Moving Average (ARIMA) models.
- The derived models show that different parts of the network grow at different rates and may experience increasing deviations from their overall trend as time progresses. Therefore capacity planning decisions should not only be based on current traffic volume between two PoPs, but also on the rate of growth that the corresponding PoP pair exhibits.
- Using the derived models, we showed that Internet backbone traffic can be accurately predicted for at least six months in the future with an average prediction error of less than 15%.
- Direct application of linear time series modelling techniques on the collected data is computationally prohibitive and prone to error. Our technique is capable of focusing on the factors which are important to capacity planning, and yields accurate estimates with a minimal computational overhead.

7.2 Discussion and Future Work

The area of large-scale IP network measurements is a fairly new research area. Only until recently did network providers and research facilities deploy dedicated monitoring equipment in operational networks to study their behaviour. Therefore, in this thesis, we have only addressed three critical problems out of the numerous problems that could be addressed within a measurement context.

Our work presented in Chapters 4 and 5 relies on the existence of packet traces collected from the operational IP backbone network. However, our current belief is that unless passive measurement capabilities are integrated with commercial router equipment, some of the analyses included in this thesis will be rendered infeasible. It has already been noted in the research community that monitoring equipment will face severe challenges as link speeds increase. Therefore, in future work we intend to address network provisioning issues detached from explicit passive measurement information.

We are interested in devising techniques that could make use of ever-existing information, such as SNMP, in order to formally define optimal network provisioning rules. In addition, we look into lightweight ways in which additional information could be provided by the routers themselves. Our analysis presented in Chapter 5 involved the correlation of BGP routing table dumps and packet traces collected on specific links of the network. The volume of traffic destined towards specific destination network prefixes could as well be provided by the router itself; a router needs to map each packet with the appropriate destination network prefix in its routing table to decide on the appropriate output interface the packet needs to be forwarded towards. Therefore, it would be an interesting avenue to investigate how feasible it would be for a router to maintain counters on the activity of specific destination network prefixes. Our analysis has shown that only a small number of destination network prefixes are active at any 5-minute interval. Moreover, only a few hundreds out of those active prefixes receive traffic at sufficiently high rates. Therefore, exporting traffic counts only for the "elephant" network prefixes may be one way to obtain more detailed information from the router, that also incorporates specific knowledge about the routing configuration itself.

Another interesting future avenue is to extend our delay and forecasting work to identify what "acceptable utilisation levels" mean in an operational environment. The current best practice dictates moderate network-wide link utilisation, that should rarely exceed 50%. This percentage is usually selected so that any link can take over the load of any other neighbouring link in case of failure. However, the utilisation of a link can only be observed using SNMP and is reported as a 5-minute average value. As link speeds increase a transient burst lasting for

1 ms will correspond to Megabytes of traffic (at OC-192 rates (10 Gbps) 1ms corresponds to 10 Megabits of traffic). The analysis of link behaviour at time scales smaller than 5 minutes is likely to reveal short-lived congestion events even when the reported SNMP link utilisation is below 50%. Identifying the error margins that should be accounted for in network capacity planning based on SNMP information is an important task that would greatly enhance current practices. Moreover, definition of alternate link utilisation metrics that could prove more meaningful for link dimensioning tasks and included in proprietary MIBs is of great interest.

Knowledge of the appropriate link utilisation levels can further be incorporated in the forecasting methodology proposed in Chapter 6, so that it leads to the exact number of links needed between any two adjacent PoPs and their respective capacity. We would further like to enhance the proposed forecasting methodology to incorporate configuration and marketing information. As shown, the derived predictions are accurate in cases when modifications in the network topology and configuration have not led to dramatic changes in the traffic behaviour. Integration of our approach with specific configuration information, and possible marketing predictions is an interesting problem that we intend to address in future work.

Appendix A

Moving Average and Exponential Smoothing Models

A.1 Simple Moving Average

In time series analysis and forecasting, we use a *Moving Average* model to estimate the current value of the mean based on past measurements. The moving average is often called a "smoothed" version of the original series, since short-term averaging has the effect of smoothing out the bumps in the original series.

At any instant k, the average of the latest n samples of a data sequence x_i is given by

$$x_k = \frac{1}{k} \sum_{i=k-n}^{k-1} x_i.$$

Hence, the forecast equals the simple average of the last n observations. This average is "centred" at period $i - (n+1)/2$ which implies that the estimate of the local mean will tend to lag behind the true value of the local mean by about $(n+1)/2$ periods. Thus, we say that the *average age* of the data in the simple moving average is $(n+1)/2$ relative to the period for which the forecast is computed.

A.2 Simple Exponential Smoothing or Exponentially Weighted Moving Average

The simple moving average model has the undesirable property that it treats the last n observations equally and completely ignores all preceding observations. Intuitively, past data should be discounted in a more gradual fashion. The *simple exponential smoothing (SES)* model accomplishes this. If α denotes the "smoothing constant", with $0 < \alpha \leq 1$, and s_t denotes the value of the smoothed series at period t, the following formula can be used recursively to update the smoothed series as new observations are recorded:

$$s_t = \alpha x_{t-1} + (1-\alpha)s_{t-1},\ 0 < \alpha \leq 1,\ and\, t \geq 3 \qquad \text{(A.1)}$$

Thus, the current smoothed value is an interpolation between the previous smoothed value and the previous observation, where α controls the closeness of the interpolated value to the most recent observation. If we expand Equation (A.1) by substituting for s_{i-1} we have:

$$s_t = \alpha x_{t-1} + (1-\alpha)\left[\alpha x_{t-2} + (1-\alpha)s_{t-2}\right] = \alpha x_{t-1} + \alpha(1-\alpha)x_{t-2} + (1-\alpha)^2 s_{t-2}$$

By recursively substituting for s_{t-2}, then for s_{t-3} and so forth until we reach s_2 which is equal to x_1, it can be shown that the expanding equation can be written as:

$$s_t = \alpha \sum_{i=1}^{t-2}(1-\alpha)^{i-1}x_{t-i} + (1-\alpha)^{t-2}s_2,\ t \geq 3\ \mathit{and}\, 0 < \alpha \leq 1$$

The above equation shows the exponential behaviour. We further can see from the summation term that the contribution of each value $x_i,\ i < t$ to the smoothed value s_t becomes less at each consecutive time interval. The speed at which old responses are dampened is a function of the value α. When α is close to 1, dampening is quick and when α is close to 0, dampening is slow.

The weights $\alpha(1-\alpha)^t$ decrease geometrically. Given that the mean of the geometric distribution of the weights is equal to $1/\alpha$, the "average age" of the data in the simple exponential smoothing forecast is $1/\alpha$ relative to the period for which the forecast is computed. In the previous section, we saw that the "average age" of the data in a simple moving average estimate for a period of n intervals is $(n+1)/2$. Consequently, the rule of thumb for the setting of the "smoothing constant" α in order to exhibit an average age of n periods can be described as:

$$\frac{1}{\alpha} \approx \frac{n+1}{2},\ \mathit{or}\ \alpha \approx \frac{2}{n+1} \tag{A.2}$$

Appendix B

ARIMA model order selection criteria

From a forecasting point of view, it is not advantageous to choose p and q arbitrarily large. Fitting a very high order model will generally result in a small estimated white noise variance, but when the fitted model is used for forecasting, the mean squared error of the forecasts will depend not only on the white noise variance of the fitted model but also on errors arising from the estimation of the parameters of the model. These will be larger for higher order models. For this reason there is a need to introduce a "penalty factor" to discourage the fitting of models with too many parameters.

Many criteria based on such penalty factors have been proposed in the literature since the problem of model selection arises frequently in statistics, particularly in regression analysis. For the evaluation of our models we restrict our attention to the FPE, AIC, and BIC criteria of Akaike and a bias-corrected version of the AIC known as the AICC.

Before proceeding with the definition of the above criteria, we introduce the concepts of *causality* and *invertibility* of an ARMA(p,q) process. These properties will be required in the definition of the selection order criteria provided in the following sections.

Causality: An ARMA(p,q) process X_t is **causal**, or a **causal function** of Z_t, if there exist constants ψ_j such that $\sum_{j=0}^{\infty} \mid \psi_j \mid < \infty$ and $X_t = \sum_{j=0}^{\infty} \psi_j Z_{t-j}$ for all t. Causality is equivalent to the condition

$$\phi(z) = 1 - \phi_1 z - \cdots - \phi_p z^p \neq 0 \; for \; all \; \mid z \mid \leq 1.$$

Invertibility: An ARMA(p,q) process X_t is **invertible** if there exist constant π_j such that $\sum_{j=0}^{\infty} \mid \pi_j \mid < \infty$ and $Z_t = \sum_{j=0}^{\infty} \pi_j X_{t-j}$ for all t. Invertibility is equivalent to the condition

$$\theta(z) = 1 + \theta_1 z + \cdots + \theta_q z^q \neq 0 \; for \; all \; \mid z \mid \leq 1.$$

B.1 Forward Prediction Error (FPE) Criterion

The FPE criterion was developed by Akaike to select the appropriate order of an AR process to fit a time series $X_1, \cdots, X_n$. Instead of trying to choose the order p to make the estimated white noise variance as small as possible, the idea is to choose the model for X_t in such a way as to minimise the one-step ahead mean squared error when the model fitted to X_t is used to predict an independent realization Y_t of the same process that generated $\{X_t\}$.

If $\hat{\sigma}^2$ is the maximum likelihood estimator of σ^2, i.e. the white noise variance of the AR(p) model, then the FPE for an AR process of order p can be estimated according to the following equation:

$$FPE_p = \hat{\sigma}^2 \frac{n+p}{n-p} \tag{B.1}$$

To apply the FPE criterion for autoregressive order selection we therefore choose the value of p that minimises FPE_p as defined in Equation (B.1).

B.2 The AICC Criterion

A more generally applicable criterion for model selection than the FPE is the Akaike Information Criterion, known as the AIC. This was designed to be an approximately unbiased estimate of the Kullback-Leibler index of the fitted model relative to the true model (defined below). The AICC is a bias-corrected version of the AIC, proposed by Hurvich and Tsai [128].

If $\mathbf{X}$ is an n-dimensional random vector whose probability density belongs to the family $\{f(\cdot;\psi), \psi \in \Psi\}$, the Kullack-Leibler discrepancy between $f(\cdot,\psi)$ and $f(\cdot;\theta)$ is defined as

$$d(\psi|\theta) = \Delta(\psi|\theta) - \Delta(\theta|\theta),$$

where

$$\Delta(\psi|\theta) = E_\theta(-2lnf(\mathbf{X};\psi)) = \int_{\Re^n} -2ln(f(\mathbf{x};\psi))f(\mathbf{x};\theta)d\mathbf{x},$$

is the Kullback-Leibler index of $f(\cdot;\psi)$ relative to $f(\cdot;\theta)$.

Given observations $X_1, \cdots, X_n$ of an ARMA process with unknown parameters $\theta = (\beta, \sigma^2)$, the true model could be identified if it were possible to compute the Kullback-Leibler discrepancy between candidate models and the true model. Since this is not possible, we *estimate* the Kullback-Leibler discrepancies and choose the model whose estimated discrepancy (or index) is minimum. In order to do this, we assume that the true model and the alternatives are all Gaussian. Then for any given $\theta = (\beta, \sigma^2)$, $f(\cdot;\theta)$ is the probability density of $(Y_1, \cdots, Y_n)'$, where $\{Y_t\}$ is a Gaussian ARMA(p,q) process with coefficient vector β and white noise variance σ^2.

We select the values p and q for our fitted model to be those that minimise AICC($\hat{\beta}$) where

$$AICC(\beta) := -2lnL_X(\beta, S_X(\beta)/n) + 2(p+q+1)n/(n-p-q-2). \tag{B.2}$$

The AIC statistic, defined as

$$AIC(\beta) := -2nL_X(\beta, S_X(\beta)/n) + 2(p+q+1), \tag{B.3}$$

can be used in the same way. Both $AICC(\beta, \sigma^2)$ and $AIC(\beta, \sigma^2)$ can be defined for arbitrary σ^2 by replacing $S_X(\beta)/n$ in the preceding definitions by σ^2. The value $S_X(\beta)/n$ is used in Equation (B.2) since $AICC(\beta, \sigma^2)$ (like $AIC(\beta, \sigma^2)$) is minimised for any given β by setting $\sigma^2 = S_X(\beta)/n$.

B.3 The BIC Criterion

For fitting autoregressive models, Monte Carlo studies suggest that the AIC has a tendency to overestimate p [129, 130]. The penalty factors $2(p+q+1)n/(n-p-q-2)$ and $2(p+q+1)$, for the AICC and AIC statistics are asymptotically equivalent as $n \to \infty$. The AICC statistic, however, has a more extreme penalty for large-order models, which counteracts the overfitting tendency of the AIC. The BIC is another criterion that attempts to correct the overfitting nature of the AIC. For a zero-mean causal invertible ARMA(p,q) process, it is defined to be [131]

$$\begin{aligned} BIC &= (n-p-q)\, ln[n\hat{\sigma}^2/(n-p-q)] + n\,(1 + ln\sqrt{2\pi}) + \\ &\quad (p+q)\, ln\left[(\sum_{t=1}^{n} X_t^2 - n\hat{\sigma}^2)/(p+q)\right], \end{aligned} \tag{B.4}$$

where $\hat{\sigma}^2$ is the maximum likelihood estimate of the white noise variance.

The BIC is a consistent order selection criterion in the sense that if the data $\{X_1, \cdots, X_n\}$ *are* in fact observations of an ARMA(p,q) process, and if $\hat{p}$ and $\hat{q}$ are the estimated orders found by minimising the BIC, then $\hat{p} \to p$ and $\hat{q} \to q$ with probability 1 as $n \to \infty$ [132]. This property is not shared by the AICC or AIC. On the other hand, order selection by minimisation of the AICC, AIC, or FPE is asymptotically efficient for autoregressive processes while order selection by BIC minimisation is not.

In the modelling of real data there is rarely such a thing as the "true order". For the process $X_t = \sum_{j=0}^{\infty} \psi_j Z_{t-j}$ there may be many polynomials $\theta(z)$, $\phi(z)$ such that the coefficients of z^j in $\theta(z)/\phi(z)$ closely approximate ψ_j for moderately small values of j. Correspondingly there may be many ARMA processes with properties similar to $\{X_t\}$. This problem of identifiability becomes much more serious for multivariate processes. The AICC criterion does, however, provide us with a rational criterion for choosing between competing models. It has been suggested that models with AIC values within c of the minimum value should be considered competitive (with $c = 2$ as a typical value) [133]. Selection from among the competitive models can then be based on such factors as whiteness of the residuals and model simplicity.

References

[1] J. Licklider, "On-Line Man-Computer Communication," in *Spring Joint Computer Conference*, National Press, Palo Alto, California, May 1962, vol. 21, pp. 113–128.

[2] B. Leiner, V. Cerf, D. Clark, R. Khan, L. Kleinrock, D. Lynch, J. Postel, L. Roberts, and S. Wolff, "The Past and Future History of the Internet," *Communications of the ACM*, vol. 40, no. 2, pp. 102–108, Feb. 1997.

[3] L. Roberts, "Multiple Computer Networks and Intercomputer Communication," in *ACM Gatlinburg Conference*, Oct. 1967.

[4] G. Huston, *ISP Survival Guide: Strategies for Running a Competitive ISP*, John Wiley & Sons Inc., 1999.

[5] J. Postel, "DoD Standard Internet Protocol," RFC 760, Jan. 1980.

[6] V. Fuller, T. Li, J. Yu, and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy," RFC 1519, Sept. 1993.

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

[8] D. Allen, "The Impact of Peering on ISP Performance: What's Best for You?," *Network Magazine*, 2001.

[9] Cisco Systems, "Cisco Very Short Reach OC-192/STM-64 Interface: Optimizing for Network Intra-POP Interconnections," White Paper.

[10] J. Moy, "OSPF Version 2," RFC 2328, Apr. 1998.

[11] D. Oran, "OSI IS-IS Intra-domain Routing Protocol," RFC 1142, Feb. 1990.

[12] C. Huitema, *Routing in the Internet*, Second Edition, Prentice Hall, 2000.

[13] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," RFC 1771, Mar. 1995.

[14] S. Halabi and D. McPherson, *Internet Routing Architectures*, Cisco Press, 2000.

[15] D. Clarke, "The Design Philosophy of the DARPA Internet Protocols," in *ACM SIGCOMM*, Aug. 1988.

[16] J. Case, J. Davin, M. Fedor, and M. Schoffstall, "Simple Gateway Monitoring Protocol," RFC 1028, Nov. 1987.

[17] M. Schoffstall, M. Fedor, J. Davin, and J. Case, "A Simple Network Management Protocol (SNMP)," RFC 1157, May 1990.

[18] J. Case, K. McCloghrie, M. Rose, and S. Waldbusser, "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)," RFC 1902, Jan. 1996.

[19] J. Case, R. Mundy, D. Partain, and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework," RFC 2570, Apr. 1999.

[20] K. McCloghrie and M. Rose, "Management Information Base for Network Management of TCP/IP-based Internets: MIB-II," RFC 1213, Mar. 1991.

[21] S. Willis, J. Burruss, and J. Chu, "Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIv2," RFC 1657, July 1994.

[22] F. Baker and R. Coltun, "OSPF Version 2 Management Information Base," RFC 1850, Nov. 1995.

[23] S. Waldbusser, "Remote Network Monitoring Management Information Base," RFC 1757, Feb. 1995.

[24] Cisco Systems, "NetFlow services and applications," http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm, White Paper.

[25] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, "NetScope: Traffic Engineering for IP networks," *IEEE Network Magazine*, 2000, special issue on Internet traffic engineering.

[26] R. S. Cahn, *Wide Area Network Design*, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1998.

[27] M. Schwartz, *Computer Communication Network Design and Analysis*, Prentice Hall, Upper Saddle River, NJ, 1977.

[28] H. Leijon, "Basic Forecasting Theories: A Brief Introduction," Tech. Rep., ITU, Nov. 1998.

[29] A. Medina, C. Fraleigh, N. Taft, S. Bhattacharyya, and C. Diot, "A Taxonomy of IP Traffic Matrices," in *SPIE ITCOM: Scalability and Traffic Control in IP Networks II*, Boston, Aug. 2002, vol. 4868.

[30] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and Diot C., "Traffic Matrix Estimation: Existing Techniques and New Directions," in *ACM SIGCOMM*, Pittsburgh, USA, Aug. 2002.

[31] C. Boutremans, G. Iannaccone, and C. Diot, "Impact of link failures on VoIP performance," in *12th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, Miami, May 2002, pp. 63–71, ACM Press.

[32] A. Markopoulou, F. Tobagi, and M. Karam, "Assessment of VoIP quality over Internet backbones," in *IEEE INFOCOM*, New York, June 2002.

[33] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Transactions on Networking*, 1994.

[34] "The Quality of Internet Service: AT&T's IP Measurements," http://ipnetwork.bgtmo.ip.att.net/paper.pdf, AT&T white paper.

[35] J. McQuilan, G. Falk, and I. Richer, "A review of the development and performance of the ARPANET routing algorithm," *IEEE Transactions on Communications*, vol. 26, pp. 1802–1811, Dec. 1978.

[36] A. Downey, "Using Pathchar to Estimate Internet Link Characteristics," in *ACM SIGCOMM*, Sept. 1999.

[37] B. Mah, "pchar: a Tool for Measuring Internet Path Characteristics," http://www.employees.org/ bmah/Software/pchar/, Feb. 1999.

[38] K. Lai and M. Baker, "Measuring Link Bandwidths Using a Deterministic Model of Packet Delay," in *ACM SIGCOMM*, Sept. 2000.

[39] R. Carter and M. Crovella, "Measuring Bottleneck Link Speed in Packet-Switched Networks," *Performance Evaluation*, vol. 27 & 28, pp. 297–318, 1996.

[40] K. Lai and M. Baker, "Measuring Bandwidth," in *IEEE INFOCOM*, Mar. 1999.

[41] C. Dovrolis, P. Ramanathan, and D. Moore, "What do Packet Dispersion Techniques Measure?," in *IEEE INFOCOM*, Apr. 2001, pp. 905–914.

[42] V. Paxson, "End-to-end Internet packet dynamics," *IEEE/ACM Transactions on Networking*, vol. 7, no. 3, pp. 277–292, 1999.

[43] V. Ribeiro, M. Coates, R. Riedi, S. Savrotham, B. Hendricks, and R. Baraniuk, "Multifractal Cross-Traffic Estimation," in *ITC Specialist Seminar on IP Traffic Measurement, Modeling, and Management*, Sept. 2000.

[44] M. Jain and C. Dovrolis, "End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput," in *ACM SIGCOMM*, Aug. 2002.

[45] V. Paxson, "End-to-end routing behavior in the Internet," *IEEE/ACM Transactions on Networking*, vol. 5, no. 5, pp. 601–615, 1997.

[46] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis, "An architecture for large-scale Internet measurement," *IEEE Communications*, vol. 36, no. 8, pp. 48–54, Aug. 1998.

[47] S. Kalidindi and M. Zekauskas, "Surveyor: An infrastructure for Internet performance measurements," in *INET*, San Jose, CA, June 1999.

[48] W. Matthews and L. Cottrel, "The PingER project: Active Internet performance monitoring for the HENP community," *IEEE Communications*, vol. 38, no. 5, pp. 130–136, May 2000.

[49] H. Uijterwall and D Karrenberg, "Internet delay measurements using test traffic: First results," in *SANE'98*, Nov. 1998.

[50] R. Cáceres, N. Duffield, D. Towsley, and J. Horowitz, "Multicast-based inference of network-internal loss characteristics," *IEEE Transactions on Information Theory*, vol. 45, no. 7, pp. 2462–2480, November 1999.

[51] T. McGregor, H.-W. Braun, and J. Brown, "The NLANR Network Analysis Infrastructure," *IEEE Communications*, vol. 38, no. 5, May 2000.

[52] "The Interaction of Web Content and Internet Backbone Performance," http://www-.keynote.com/services/html/wp_compdata.html, Keynote white paper.

[53] "MIQ Ratings Methodology," http://ratings.miq.net/method.html.

[54] J. Mahdavi and V. Paxson, "IPPM Metrics for Measuring Connectivity," RFC 2678, Sept. 1999.

[55] G. Almes, S. Kalidindi, and M. Zekauskas, "A One-way Delay Metric for IPPM," RFC 2679, Sept. 1999.

[56] G. Almes, S. Kalidindi, and M. Zekauskas, "A One-way Packet Loss Metric for IPPM," RFC 2680, Sept. 1999.

[57] G. Almes, S. Kalidindi, and M. Zekauskas, "A Round-trip Delay Metric for IPPM," RFC 2681, Sept. 1999.

[58] M. Mathis and M. Allman, "A Framework for Defining Empirical Bulk Transfer Capacity Metrics," RFC 3148, July 2001.

[59] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving Traffic Demands for Operational IP Networks: Methodology and Experience," *IEEE/ACM Transactions on Networking*, , no. 3, pp. 265–280, June 2001.

[60] N. Brownlee, C. Mills, and G. Ruth, "Traffic flow measurement: Architecture," RFC 2722, Oct. 1999.

[61] S. Handelman, S. Stibler, N. Brownlee, and G. Ruth, "RTFM: New attributes for traffic flow measurement," RFC 2724, Oct. 1999.

[62] J. Quittek, T. Zseby, B. Claise, S. Zander, G. Carle, and K. Norseth, "Requirements for IP Flow Information Export," Internet draft, Aug. 2002.

[63] J. Apsidorf, K.C. Claffy, K. Thompson, and R. Wilder, "OC3MON: Flexible, Affordable, High Performance Statistics Collection," in *INET*, June 1997.

[64] K. Thompson, G. J. Miller, and R. Wilder, "Wide-Area Internet Traffic Patterns and Characteristics," *IEEE Network*, pp. 10–23, November 1997.

[65] K.C. Claffy, "The Nature of the Beast: Recent Traffic Measurements from an Internet Backbone," in *INET*, 1998.

[66] G. Iannaccone, C. Diot, I. Graham, and N. McKeown, "Monitoring very high speed links," in *ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, Nov. 2001.

[67] "Dag 3.2 SONET network interface," http://dag.cs.waikato.ac.nz/dag/dag4-arch.html.

[68] "Dag 4 SONET network interface," http://dag.cs.waikato.ac.nz/dag/dag4-arch.html.

[69] K. Thompson, G. Miller, and R. Wilder, "Wide Area Internet Traffic Patterns and Characteristics," *IEEE Network*, Nov. 1997.

[70] "Dag synchronization and timestamping," http://dag.cs.waikato.ac.nz/dag/docs-/dagduck_v2.1.pdf.

[71] F. Drake, "Documenting Python Release 2.2.1," http://www.python.org/doc/current/download.html, Apr. 2002.

[72] G. Huston, "Analyzing the Internet's BGP Routing Table," *The Internet Protocol Journal*, vol. 4, no. 1, Mar. 2001.

[73] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, K. Papagiannaki, and F. Tobagi, "Design and Deployment of a Passive Monitoring Infrastructure," in *Passive and Active Measurement Workshop*, Amsterdam, April 2001.

[74] R. Cảceres, N. Duffield, A. Feldmann, J. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. Kalmanek, B. Krishnamurthy, D. Lavelle, P. Mishra, J. Rexford, K. Ramakrishnan, F. True, and J. Van der Merwe, "Measurement and analysis of IP network usage and behavior," *IEEE Communications*, vol. 38, no. 5, pp. 144–151, May 2000.

[75] J. Cao, D. Davis, S. Vander Weil, and B. Yu, "Time-Varying Network Tomography," Journal of the American Statistical Association, 2000.

[76] Y. Vardi, "Estimating Source-Destination Traffic Intensities from Link Data," Journal of the American Statistical Association, 1996.

[77] C. Tebaldi and M. West, "Bayesian Inference of Network Traffic Using Link Count Data," Journal of the American Statistical Association, 1998.

[78] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," *IEEE Communication Magazine*, Oct. 2002.

[79] B. Fortz and M. Thorup, "Internet Traffic Engineering by Optimizing OSPF Weights," in *IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000.

[80] M. Thorup, "Fortifying OSPF/IS-IS against link failure," manuscript, Sept. 2001.

[81] A. Nucci, B. Schroeder, S. Bhattacharyya, N. Taft, and C. Diot, "IS-IS Link Weight Assignment for Transient Link Failures," Sprint ATL Technical Report TR02-ATL-071000, Sprint Labs, July 2002.

[82] C. Labovitz, A. Ahuja, and F. Jahanian, "Experimental Study of Internet Stability and Wide-Area Network Failures," in *International Symposium on Fault-Tolerant Computing*, June 1999.

[83] C. Labovitz, A. Ahuja, A. Bose, and Jahanian F., "Delayed Internet Routing Convergence," in *ACM SIGCOMM*, Stockholm, Sweden, Aug. 2000.

[84] C. Labovitz, G. Malan, and F. Jahanian, "Internet Routing Instability," *IEEE/ACM Transactions on Networking*, Aug. 1997.

[85] C. Labovitz, G. Malan, and F. Jahanian, "Origins of Pathological Internet Routing Instability," in *IEEE INFOCOM*, Mar. 1999.

[86] G. Iannaccone, C-N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures over an IP backbone," in *ACM SIGCOMM Internet Measurement Workshop*, Marseilles, France, Nov. 2002.

[87] I. Norros, "On the use of fractional Brownian motion in the theory of connectionless networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 6, pp. 953–962, 1995.

[88] A. Erramilli, O. Narayan, and W. Willinger, "Experimental queuing analysis with long-range dependent packet traffic," *IEEE/ACM Transactions on Networking*, 1996.

[89] A. Feldmann, A.C. Gilbert, and W. Willinger, "Data networks as cascades: Investigating the multifractal nature of Internet WAN traffic," in *ACM SIGCOMM*, 1998.

[90] A. Erramilli, O. Narayan, A. Neidhardt, and I. Saniee, "Performance Impacts of Multi-Scaling in Wide Area TCP/IP Traffic," in *IEEE INFOCOM*, 2000.

[91] N. McKeown, "A Fast Switched Backplane for a Gigabit Switched Router," *Business Communications Review*, Dec. 1997.

[92] N. McKeown, "iSLIP: A Scheduling Algorithm for Input-Queued Switches," *IEEE Transactions on Networking*, vol. 7, no. 2, Apr. 1999.

[93] D. E. Knuth, *The Art of Computer Programming, Volume I: Fundamental Algorithms*, Second Edition, Addison-Wesley Publishing Company, Reading, 1973.

[94] V. Paxson, *Measurements and Analysis of End-to-End Internet Dynamics*, Ph.D. thesis, University of California Berkeley, April 1997.

[95] R. W. Wolff, "Poisson Arrivals See Time Average," *Operations Research*, vol. 30, pp. 223–231, 1982.

[96] S. Keshav and R. Sharma, "Issues and Trends in Router Design," *IEEE Communications Magazine*, vol. 36, no. 5, pp. 144–151, May 1998.

[97] D. R. Cox, "Long-range dependence: a review," in *Statistics: An Appraisal*, H. A. David and H. T. David, Eds., pp. 55–74. Iowa State University Press, Ames, IA, 1984.

[98] M. E. Crovella and M. S. Taqqu, "Estimating the Heavy Tail Index from Scaling Properties," *Methodology and Computing in Applied Probability*, vol. 1, no. 1, 1999.

[99] S. Bhattacharyya, C. Diot, J. Jetcheva, and N Taft, "POP-level and Access-Link-Level Traffic Dynamics in a Tier-1 POP," *ACM SIGCOMM Internet Measurement Workshop*, November 2001.

[100] W. Fang and L. Peterson, "Inter-AS Traffic Patterns and Their Implications," *IEEE Globecom*, December 1999, Brazil.

[101] A. Shaikh, J. Rexford, and K. Shin, "Load-Sensitive Routing of Long-Lived IP Flows," *ACM SIGCOMM*, September 1999.

[102] S. Uhlig and O. Bonaventure, "On the Cost of Using MPLS for Interdomain Traffic," *Quality of Future Internet Services*, September 2000, Berlin.

[103] S. Floyd, "Simulation is Crucial," *IEEE Spectrum*, January 2001.

[104] M. Crovella, "Performance Evaluation with Heavy Tailed Distributions," in *Lecture Notes in Computer Science 1786*, March 2000.

[105] C. Estan and G. Varghese, "New Directions in Traffic Measurement and Accounting," *ACM SIGCOMM Internet Measurement Workshop*, Aug. 2001.

[106] X. Su and G. de Veciana, "Dynamic multi-path routing: asymptotic approximation and simulations," *ACM SIGMETRICS Performance Evaluation Review*, vol. 29, no. 1, pp. 25–36, June 2001.

[107] H. Abrahamsson, B. Ahlgren, J. Alonso, A. Andersson, and P. Kreuger, "A Multi Path Routing Algorithm for IP Networks Based on Flow Optimisation," in *International Workshop on Quality of future Internet Services (QofIS)*, Zurich, Switzerland, Oct. 2002.

[108] M. Crovella and M. Taqqu, "Estimating the Heavy Tail Index from Scaling Properties," *Methodology and Computing in Applied Probability*, 1999.

[109] J. W. Steward, *BGP4: Inter-Domain Routing in the Internet*, Addison Wesley, 1999.

[110] N. K. Groschwitz and G. C. Polyzos, "A Time Series Model of Long-Term NSFNET Backbone Traffic," in *IEEE ICC'94*, 1994.

[111] S. Basu and A. Mukherjee, "Time Series Models for Internet Traffic," in *24th Conf. on Local Computer Networks*, Oct. 1999, pp. 164–171.

[112] J. Bolot and P. Hoschka, "Performance Engineering of the World Wide Web: Application to Dimensioning and Cache Design," in *5th International World Wide Web Conference*, May 1996.

[113] K. Chandra, C. You, G. Olowoyeye, and C. Thompson, "Non-Linear Time-Series Models of Ethernet Traffic," Tech. Rep., CACT, June 1998.

[114] R. A. Golding, "End-to-end performance prediction for the Internet," Tech. Rep. UCSC-CRL-92-96, CISB, University of California, Santa Cruz, June 1992.

[115] A. Sang and S. Li, "A Predictability Analysis of Network Traffic," in *INFOCOM*, Tel Aviv, Israel, Mar. 2000.

[116] R. Wolski, "Dynamically Forecasting Network Performance Using the Network Weather Service," in *Journal of Cluster Computing*, 1999.

[117] I. Daubechies, "Ten Lectures on Wavelets," in *Cbms-Nsf Regional Conference Series in Applied Mathematics*, 1992, vol. 61.

[118] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1989, vol. 11, pp. 674–693.

[119] J. Walker, *A primer on wavelets and their scientific applications*, Chapman & Hall, 1999.

[120] G. Nason and B. Silverman, "The Stationary Wavelet Transform and some Statistical Applications," in *Lecture Notes in Statistics: Wavelets and Statistics*, 1995, pp. 281–300.

[121] M. Shensa, "The Discrete Wavelet Transform: Wedding the À Trous and Mallat Algorithms," in *IEEE Transactions on Signal Processing*, 1992, vol. 40, pp. 2464–2482.

[122] J.-L. Starck and F. Murtagh, "Image restoration with noise suppression using the wavelet transform," *Astronomy and Astrophysics*, vol. 288, pp. 342–348, 1994.

[123] A. Aussem and F. Murtagh, "Web traffic demand forecasting using wavelet-based multiscale decomposition," in *International Journal of Intelligent Systems*, 2001, vol. 16, pp. 215–236.

[124] P. Yu, A. Goldberg, and Z. Bi, "Time Series Forecasting using Wavelets with Predictor-Corrector Boundary Treatment," in *7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 2001.

[125] R. Jain, *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*, John Wiley, New York, 1991.

[126] P. Brockwell and R. Davis, *Introduction to Time Series and Forecasting*, Springer, 1996.

[127] W. N. Venables and B. D. Ripley, *Modern Applied Statistics with S-PLUS*, Springer, 1999.

[128] C.M. Hurvich and C.L. Tsai, "Regression and time series model selection in small samples," *Biometrika*, vol. 76, no. 2, pp. 297–307, 1989.

[129] R.H. Jones, "Fitting autoregressions," *Journal of the American Statistical Association*, vol. 70, pp. 590–592, 1975.

[130] R. Shibata, "Selection of the order of an autoregressive model by Akaike's information criterion," *Biometrika*, vol. 63, no. 1, pp. 117–126, 1976.

[131] H. Akaike, "Time series analysis and control through parametric models," in *Applied Time Series Analysis*, D.F. Findley, Ed. Academic Press, New York, 1978.

[132] E.J. Hannan, "The estimation of the order of an ARMA process," *Annals of Statistics*, vol. 8, no. 5, pp. 1071–1081, 1980.

[133] Q.P. Duong, "On the choice of the order of autoregressive models: a ranking and selection approach," *Journal of Time Series Analysis*, vol. 5, pp. 145–157, 1984.