Measurement on a network-Studies and Techniques

M. Tech. Seminar Report

Submitted in partial fulfillment of the requirements
for the degree of

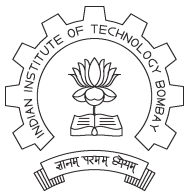
Master of Technology

by

Girija Limaye
Roll No: 07305905

under the guidance of

Prof. Purushottam Kulkarni



Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
Mumbai

Contents

List of Figures

List of Abbreviations

FIFO	First In First Out
HMM	Hidden Markov Model
ICMP	Internet Control Message Protocol
OWD	One-Way Delay
RTO	Re-Transmission Timeout
RTT	Rount-Trip Time
SLoPS	Self Loading Periodic Streams
TTL	Time To Live

Abstract

"Measurements on Network" is a topic related to measuring various network properties, studying effects of some properties on few others. It also covers different techniques used for the same. For example, measuring end-to-end delay (one property of network) and studying its effect on other properties and thus trying to calculate or estimate them, say estimating bandwidth of link(s). The topic mainly focuses on studying these kind of relationships between network properties and on techniques used so far in order to measure them.

Some network properties are important from end-user's perspective (e.g. speed, capacity of network) while others are important for network designers and network administrators (e.g. link bandwidth, queue sizes). There exists a relationship between these two types of properties, like small queue size at server might increase probability of dropping packets which from user's perspective is the reduction in network capacity or reduction in speed. Study of such relationships is one of the objectives of the seminar.

To measure such properties of networks, various techniques can be used. One such simple technique could be to monitor network at some selected points (may be with the help of packet sniffers, like ethereal) and based on some mathematical formalization, get parameter values. In such measurements, where to place such network monitors is also a subject of study. The measurement could also include injecting some additional traffic and then analyzing results and estimating parameters based on them. Studying such techniques and comparing them is another objective of the seminar.

1 Introduction

Network measurements have three aspects: Why?, What? and How?.

1.1 Why Measurements?

Suppose packets at the end host are getting dropped every now and then. Then there can be many reasons to it: Any of the links on the path may be lossy; or one or more of the intermediate routers might be dropping packets, if that is so, the queues at the routers might be overflowing; or it can be because the bits of the packet are getting garbled which results in dropping packet at the end host itself. As we can see, there are various reasons for a phenomenon of "packets getting dropped". So in order to find out the exact cause(s) of this, one needs to measure network properties. Let's consider another situation where a network administrator wants to add more hosts to the existing network, keeping the resources unchanged. In this case, (s)he needs to measure currently supported network performance, like end-to-end delay, packet loss etc. in order to decide whether there is a "room" in existing network to support more hosts.

Above mentioned are the two situations where network measurements are required. Network measurements can have many such objectives. Following could be some of the purposes of network measurements:

- **Capacity planning-** With given amount of resources (which include capacity of links i.e. bandwidths, routers, processing servers etc.) one has to estimate network performance for a certain capacity. This may involve simulating such network with given resources and then estimating the capacity (say, number of hosts it could support). This simulation involves measurements. In another case, when existing network is going smoothly, from various measurements, one might come to know that there is a "room" for new hosts and thus helps in increasing capacity. Which parameters actually indicate performance will be covered in 1.2
- **Improve end-to-end performance-** End-to-end performance includes end-to-end delay, one way delay, delay variation (which results in jitter and important for multimedia applications) and connectivity. Many network properties that indicate end-to-end performance are discussed in 1.2
- **Usage based-** For network providers, measuring bandwidth, databytes downloaded, time periods of user being "ON" are important in order to calculate usage based metrics, for example say billing users for their network usage.
- **Network diagnosis and troubleshooting-** Network administrators not only need to monitor network events periodically but also keep statistics about network activities for long term. This helps them in dignosing and solving some problems. For example, if email delivery is slowed down, from the queue traces collected for a long time, one might get some idea whether excessive queuing is the reason for the delay.

1.2 What to Measure?

One should measure only those properties of network, which will give some insight about any other properties. Example could be measuring a Round Trip Delay to get an estimation about queuing and bandwidth. So in order to decide what to measure, one needs to apply some basics about general routing through network. A **network parameter** is the network property that one measures. While a **metric** is the network property that one estimates or infers from parameters.

Following gives a brief introduction to some of the network parameters:

1. **Round Trip Time (RTT):** This network parameter in addition to distance i.e. number of hops, is also an indication to queuing and congestion. Changes in RTT usually indicate changes in congestion level.
2. **RTT Variance:** Applications like Video and audio are sensitive to the variance in transmit time, where RTT variance comes into picture.
3. **One way delay (OWD):** In addition to RTT, one way delay and its variance also are an indication of link bandwidths and queuing at the routers.
4. **Number of hops:** This parameter might not be directly visible for end user but is important for network administrators.

Following are some of the network metrics:

1. **Bandwidth:** It is the data carrying capacity of the link. A 10 mbps link is said to have a **capacity** of 10 mbps. When it carries data at 7 mbps, it is not fully utilized and its **utilization** is said to be 70 %. The remaining (i.e. capacity - utilized) is called as **available bandwidth**.
2. **Packet Loss Rate:** This is a critical measure giving view of performance, specially in applications like multimedia. Here network parameter that we measure is the number of packets actually lost.
3. **Reachability:** This refers to connectivity. It is a good indication of how often is one edge of the network unreachable from the other.

Once the objective of network measurement and the parameters to be measured are decided, how should one actually carry out measurements? One might run some packet capturers like *ethereal* and get some indication about network activity. One might run a tool like *traceroute* in order to find out number of hops. A tool like *pathchar* gives bandwidth of the links once a network path is known. So from the point of view of the network administrator as well as from end user's point of view, using tools and automated scripts or softwares is an intuitive way of measurement over networks. What lies beneath these tools are some basic measurements techniques and methodologies that we will now look at.

1.3 How to Measure?

Let's see some of the intuitive ways of measurements that one can think of. They are described as follows:

- To measure an RTT, one might think of sending a TCP packet and recording its timestamp. When associated acknowledgement is received, again timestamp is recorded. The difference between these two timestamps gives RTT.
- One might add timestamp value in the packet at the sender and at the receiver. The difference between arrival of packet and its timestamp value gives OWD. For this, clocks at sender and receiver need to be synchronized. So one might consider only the delay variation, which will remove effect of clocks being out-of-synchronization.
- One can use some tools like *traceroute* for estimating number of hops. Also a tool like *pathchar* can be used to estimate network link properties.

Active measurements on network actually transmit packet probes and estimate metrics from the response received, whereas **Passive Measurements** capture system packets (without introducing additional traffic) and estimate metrics.

Thus active measurements provide end-to-end characteristics which improves one's understanding about network traffic dynamics. But they also introduce overhead of additional traffic which may affect measured parameters (and thus estimated metrics).

On the basis of above mentioned intuitive ways, there exist some foundational knowledge about the networks. For instance, the intuituion of measuring RTT comes from the knowledge that there exists a relationship between RTT and other metrics like bandwidth and queuing at the router. In some cases this kind of knowledge application helps while deciding on what to measure, whereas in some other cases, we need some modelling of network in order to estimate metrics. In section 2 and 3, we will see systematic ways of measurements.

1.4 Scope

While network measurements is a broad area covering several topics, my goal is to cover some of the important techniques and discuss and compare examples of each of them. For this seminar, I have concentrated on wired networks, wherein I try to cover measurements from different aspects. This includes measuring network properties like packet loss, delay and link bandwidths. I also focus on the modelling of a network and certain related issues. This includes statistical modelling and inferences. Wireless domain, specifics like security or application level measurements, like DNS are not defined in my scope.
In section 2, I will briefly introduce the various measurement areas. In section 3, I cover measurement techniques I have read and understood while in section 4, I discuss about the techniques with practical examples and explaination. At the end of this report, I conclude by commenting about the techniques I learnt and giving my ideas on where this study can lead to.

2 Measurements in Action

Measurements on a network are classified in various categories depending on following:

- What is the purpose of measurement?
- Which measurement technique is used?
- At what point in the network, measurement is done?
- Are the measured parameters used for estimation of metrics or as an input for modelling network and thus explaining some observations?

Based on these, following are the measurement categories:

2.1 Network Properties

Packet loss, end-to-end delay characteristics and bandwidth are the important network properties that affect network performance. Following sections give the causes behind these properties, in order to get insight as to what measurement gives what estimate.

2.1.1 Packet Loss and Delay Characteristics

There are various reasons to packet loss as follows:

- **Congestion**: In TCP, packet loss indicates congestion. If network is congested at any point in between, packets will be lost.
- **Bit error**: In case the bits in packet get garbled, packet will be dropped and thus will be lost.
- **Discard for some reasons**: This may include overflow of finite queue at intermediate routers, which makes them drop packets.

Following are a few reasons for delay over a path:

- **Bottleneck Link**: In a network path, the link having lowest bandwidth will affect overall delay. Figure 1 shows this phenomemon. As shown in the figure, assume B1 and B2 are the bandwidths of links shown, such that $B2 < B1$. When a pair of packets P1 and P2 are transmitted over the link with bandwidth B2, due to lower bandwidth, it introduces time spacing between P1 and P2 for all subsequent links. If B2 is the lowest bandwidth over the entire path, the corresponding link is called **Bottleneck Link**.
- **Queuing at the intermediate routers**: Suppose there are R routers where for some router j, there exists a lot of cross traffic[1]. In that case, queuing will take place at router j and thus the delay for packets will increase oonsurther links. As shown in figure 2, packet P1 and P2 are back-to-back from router R1 to R3. But if from router R2, due to cross traffic, another packet $P^{'}$ comes to router R3, and if that gets queued before P2 reaches, this will increase the spacing between P1 and P2 for subsequent links.

This kind of measurement thus helps in inferences about queuing at the routers, link properties like bandwidth.

[1]Cross Traffic refers to the flows other than the one being monitored and measured.

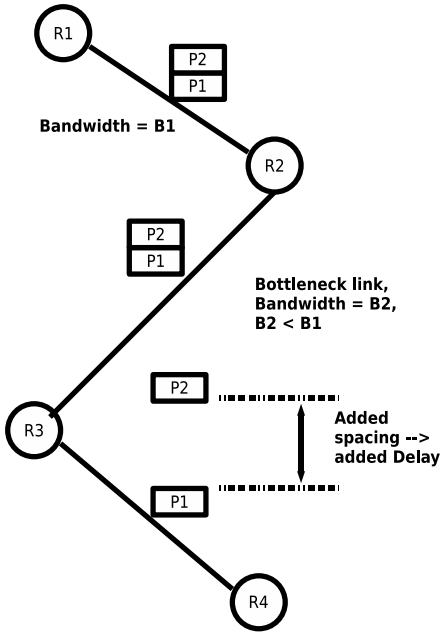


Figure 1: Delay variation due to bottleneck link

2.1.2 Link Bandwidth

Bandwidth refers to the amount of data transfered per unit of time. Bandwidth of the link and of the path (number of links) is usually different. Also one more differentiation is the maximum possible bandwidth (Capacity) and maximum unused bandwidth (Available). If the network administrator has access to the router connected to the link, bandwidth can be directly measured. But if this is not possible, with some techniques at the end hosts, one can estimate bandwidth, referred to as **end-to-end bandwidth estimation**. At layer-2, the links are called segments, where the capacity mainly depends on physical data carrying capacity of the segment. At IP layer, we group such segments into hops and capacity of hop is the maximum rate possible. This will be smaller than that at layer-2 as it involces encapsulation and framing. Considering a path of H hops, end-to-end capacity is

$$C = min\ \{C_i\} \text{ where } i = 1, 2, \ldots, H$$

The link with lowest bandwidth is called **narrow link**. While estimating bandwidth of the entire end-to-end path, this narrow link is called **bottleneck link**.

Bandwidth estimation helps in optimizing end-to-end performance, capacity planning and traffic engineering. It is also important for video-audio streaming applications, end-to-end admission control and intelligent routing systems.

2.2 Network Monitoring

Network monitoring refers to a system that constantly monitors a network and reports the events happening on the network to the network administrator. Such events could be failure of a node, unreachability of some node. This can be considered as a part of network management. The information may be provided to the administrator in the form of charts, summarized tables. Such monitoring and periodic generation of results can help network administrator in troubleshooting a problem in an efficient manner.
Thus the network monitoring is a passive measurement method that places network monitors which could be actual hardware box sampling the flows on that network link, such that they can cover maximum network and generate observations about the same. It is important to

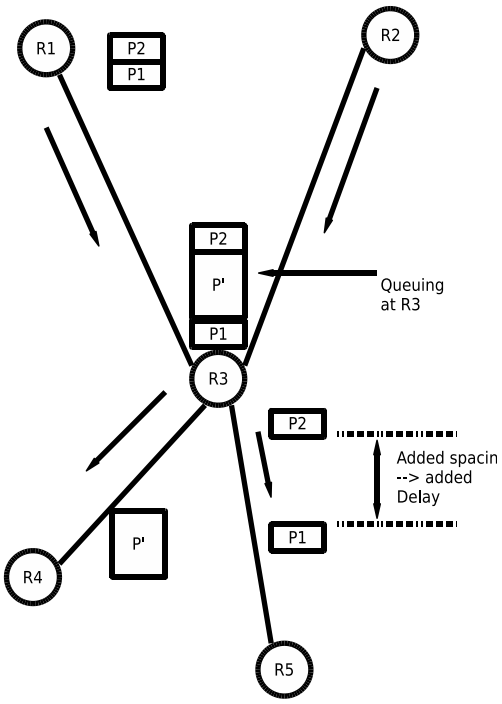


Figure 2: Delay variation due to queuing

design and implement a monitoring system that will reduce complexity and prove efficient in terms of coverage. In section 4.3, we consider an example of placing network monitors efficiently.

2.3 Network Tomography

Network tomography is the study of a network's internal characteristics using information which is measured at the end hosts. So this refers to estimating bandwidth, loss, connectivity and other such network metrics using only end-to-end measurements. There are two forms of network tomography:

- Link level metric estimation based on end-to-end path level parameters
- Sender-Receiver path level traffic intensity based on end-to-end link level parameters.

In such techniques, the metric could be **loss rate or delay** of the link. Path level delay is composed of delay of every link and router processing delays. Thus measured delay at the end hosts can give an inference about the capacity of links as well as the queuing over the path. Whereas information about packets received successfully at the receiver among the total packets sent at the sender, can give estimate about packet loss.

2.4 Network Modelling and Performance Analysis

Consider an example of TCP. TCP considers the network as being in one of the states *congested* or *non-congested*. If the packet gets dropped or duplicate ACKs are received, the sender will assume network state as *congested* and perform it's congestion avoidance mechanism and exponential backoff. But what if the packets are getting dropped due to some other reason? So it is better if the sender can in some way know what is going on "inside" the network from some measurements and thus can estimate the "real" cause behind the packet loss. Network modelling helps here.

If one models the behaviour of a network in some way, one can use such model to estimate future behaviour. In above example, the network can be modelled as being in one of these (congested or non-congested) states with the pattern of transitions between these states, such a model can be used to estimate performance in future based on current state. In section 4.5, we consider Hidden Markov Modelling [14] of a network state.

2.5 Miscellaneous

Internet Routing and Path Properties

If for some reasons (say congestion), route is changed and the links on the new route have lower bandwidths compared to previous ones, then this results in delay variation. Thus, how this routing changes affect path properties like end-to-end delay is important to study.

Traffic Engineering

Traffic engineering refers to finding patterns or trends in traffic that flows in the network. Such trends can help us in estimating the distribution that the web traffic, file sizes tranferred, request interarrval time follow. This helps in designing network and network services. For example, if we know the distribution of the file requests by the end user, we can predict whether caching will help or not. If the distribution shows that some particular file requests have higher probability of generation from the end user, one can think of caching these files in order to improve performance.
Thus knowing the traffic shape or trend is an important area to study. In section **??**, a brief introduction to web traffic analysis is covered. It also shows the observations about the distributions the web traffic follows and infers reasons behind it.

Topology Identification

For measurements in network tomography, where we try to relate parameters measured at the end hosts with those inside the network, it is important to know or estimate topology. So this area so measurements focus on measuring those parameters of network which will help us understanding topology. This includes finding shared links between many clients and servers in order to identify topology.

Measurements on Wireless Networks

This is an entirely different area of measurements, which I have not covered in my literature survey. This deals with protocol measurements, performance analysis of MAC, routing and Transport layer protocols used in wireless domain, such as estimating maximum throughput. One such technique could be found in [2].

3 Measurement Techniques

Following are the basic measurement techniques. This section gives an overview of how these techniques are useful.

3.1 Packet Based

Single Packet Probes

This is a very simple technique that uses line fitting to estimate link bandwidths. A single packet of varying size is sent and RTT is measured every time. Once we get enough sample points, **linear regression** is applied to fit a straight line through these points, the slope of which is reciprocal of the bandwidth of network path. Refer figure 3 for details. As shown, a graph is plotted for RTT Vs. packet size, and a straight line is fitted through these sample points. The slope m of this line gives reciprocal of bandwidth, as in general

$$RTT = \frac{packetsize}{bandwidth}, \text{ thus } RTT \propto \frac{1}{bandwidth}$$

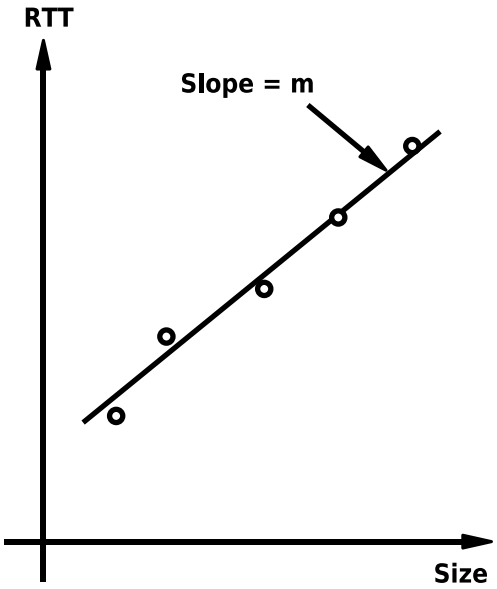


Figure 3: Line fitting for *Pathchar*

Packet Pairs

Packet pairing techniques can be used to estimate bottleneck bandwidth. Let's first discuss the packet pair property of FIFO queuing. Consider there are three links between source S and destination D such that second link has a lower bandwidth. A pair of packets is sent from S at time instance t and $t + \Delta t$ respectively. When it reaches the second link, spacing between the packets will remain Δt but during transmission from second link, due to lower bandwidth, the spacing increases to Δs, which is greater than Δt. Refer figure 1 in section 2.1. As the pair reaches third link, the spacing still remains Δs and though the bandwidth of

third link is more, due to spacing, delay more that Δt, i.e. more than initial sending delay, is observed at D. This difference is used for infering bottleneck link. Refer section 4 for deails.

3.2 Protocol Pased

ICMP Messages

ICMP defines various codes that are used for error indication. ICMP Echo Request and Reply can be used to get an idea about connectivity, as is done in *Ping*. It works as follows: The client generates an ICMP Echo request and sends it to target computer. The target machine builds an ICMP Echo reply and sends it back. When no reply received within the timeout, reachability is assumed to be non-existing. Another use of ICMP packets (ICMP Error messages) is explained below.

TTL Field in IP header

TTL (Time-To-Live) field in an IP header ensures that no packet loops in the network forever. This is because, starting with some TTL value N, it is decremented at every hop and once it reaches zero, the packet is dropped. In addition to dropping the packet, an ICMP error message is also sent. With this return error message, we get an idea about RTT, which we can use, in number of ways. In section 4, we will look at a tool called *Pathchar* implemented using this technique.

Packet Pair with Tailgating

Packet tailgating is an extension to packet pairs technique. In first phase, it tries to estimate characteristics of entire network path using single packet probe technique explained above. While in second phase it estimates characteristics of individual links. In tailgating technique one sends a larger packet, which faces no queuing while a second smaller packet queues after the first packet at the link say L, but at no later link. For this IP TTL field of first packet is set to L. Thus the smaller packet almost always has a lower transmission delay than the larger packet's transmission delay on the next link. This causes the smaller packet (also called the tailgater) to queue continuously after the larger packet (called the tailgated). The TTL for the tailgated (larger) packet will cause it to be dropped at link L, so the tailgater (smaller) can then continue without queuing till the destination. The first phase of technique takes into account non-bandwidth delays for entire network path while second phase then estimates per link bandwidth with the help of tailgating.

3.3 Delay and Delay Variance

SLoPS (Self Loading Periodic Streams)

SLoPS refer to a stream of packets sent from sender S. With such a packet stream, OWD can be infered at the receiver and a trend among these delays can be found out for one such stream. This helps in finding the available bandwidth. This technique, with more specific scenario, is explained in detail in section 4.

The above explained techniques make use of the basics of networking or protocol mechanisms in order to estimate network metrics. There are few other techniques that mathematically model the network and based on probabilistic-statistical methods, estimate network metrics. Following describes some of such probabilistic techniques:

3.4 Statistical Methods and Modelling

Hidden Markov Model

This section explains basics of Markov Models and HMM[14]. Markov Chain is a discrete time stochastic process. It has following parameters:

- State space, which is a sequence of random variables.
- Probability transition matrix giving probabilities with which being in state i, system will make transition to some state j (or itself).
- States obeying Markov Property: Markov Property states that: Given the present state, the future state is independent of past states.

HMM is a statistical model in which the system being modeled is assumed to be a Markov Process with unknown (hidden) parameters. The challenge is to determine the hidden parameters from the observable parameters. In a regular Markov Model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In an HMM, the state is not directly visible, but variables influenced by the state are visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM (i.e. observation) gives some information about the sequence of (hidden) states. With this background, we will talk about Hidden Markov Modeling of a network state in section 4.

Baysian Inference

It is based on Bayes' theorem, which is stated as follows:

$$P(A|B) = \frac{P(B|A)*P(A)}{P(B)}$$

where,
$P(A)$ is the **prior probability** of A. It is "prior" in the sense that it does not take into account any information about B.
$P(A|B)$ is the conditional probability of A, given B. It is also called the **posterior probability** because it is derived from or depends upon the specified value of B.
$P(B|A)$ is the **conditional probability** of B given A.
P(B) is the prior probability of B.

With reference to Bayesian Inference, A is called the **Hypothesis** and B as the **Evidence**. Bayesian inference is statistical inference in which the evidence is used to update or to newly infer the probability that the hypothesis may be true. Bayesian inference uses a numerical estimate of the degree of belief in the hypothesis before the evidence has been observed (prior probability) and calculates a numerical estimate of the degree of belief in the hypothesis after the evidence has been observed (posterior probability). A technique using Bayesian inference is covered in the section 4.

4 Discussion and Examples

In this section, we will see some of the techniques mentioned above in detail.

4.1 Packet Loss and Delay Characteristics

4.1.1 Loss Pairs to Detect Network Properties

For discussion on packet pair techniques, refer to section 3.1. Here we extend the idea of packet pair to **loss pair**. A loss pair is a packet pair where both packets follow the same path and are close together in time and at some point, one of the packets is dropped. The reason behind this dropping could be: TTL exceeded or router's buffer (or queue) overflowed and due to some policy router just dropped the incoming packet. This event of loss can give insight into some of the network properties at that instance of time, like packet dropping policy of routers and estimation of buffer sizes at routers. Knowing the packet dropping policy at the router, is helpful for the end-host, specially the sender in designing congestion-control policy (because, packet loss is taken as an indication to congestion). Also as loss of packets result in jitter in some applications like multimedia, streaming, one can predict the amount of disturbance one will get with the help of this dropping policy inference. This method takes passive measurements at the end points and tries to infer properties of routers which are inside the network.
The idea behind loss pairs is as follows: As both the packets are close in time, they face similar network conditions and the one surviving out of them can give insight to the network properties at that time instance. One can snoop the TCP ackowledgements to know about lost packets. Consider two packets P1 and P2 forming a loss pair. Consider a situation in which P1 gets dropped. At the router where P1 gets dropped, the router thus flushes its buffer and make space for new packets wherein P2 fits, which gets delivered preperly. Thus the RTT of P2, which is RTT_{P2} will constitute this time for buffer flushing. One can also get minimum RTT across this path, say RTT_{min}, which refers to only propogation delay, no queuing acticity. Now,

$$RTT_{P2} = RTT_{min} + T \text{ where T is the time to drain the buffer at the router.}$$

With this we can estimate buffer size at the bottleneck router as:

$$B = T * C \text{ where C is bottleneck bandwidth which is assumed to be known.}$$

With these traces, if one plots dropping rate Vs. number of packets in buffer, one can get an idea about which policy the router is using. For further details about this inference, one can refer to [10].

4.1.2 Impact of Routing Changes on Delay

When the routing change occurs this will affect end-to-end delay. It is thus important to know how frequently these routing changes occur on an average. Also there are two aspects to routing: interdomain (meaning across the domains) and intradomain (within a domain). So routing changes can occur both within the domain and across the domains. With passive measurements, i.e. by capturing packet traces, one can notice interdomain routing change. For instance, one can collect traces by (passively) monitoring BGP sessions and infering from headers, which routes are currently being used. This in addition to recording end-to-end delays can give idea about which routes corresponds to higher delays. For intradomain routing, one needs to exploit active measurement techniques. One needs to send probes as soon as BGP session indicates that a "change of route" event has occurred. With *traceroute* such information about changes in intradomain route can be collected. Once these traces are collected, one can plot graphs showing delay values Vs. routing events. The observations

from the paper [5] say that interdomain paths last longer whereas intradomain paths change relatively often.

4.1.3 Alternative techniques

In addition to single packet probe technique and packet pair technique (like in loss pairs), a technique called packet quartets is also useful. In this two pairs of packets (thus a quartet) are used. Each packet pair contains a probe and a pacesetter packet. This technique in addition to varying TTL and delay difference can be used to estimate bandwidth of links. For further details, refer [1].

4.2 Bandwidth Estimation

4.2.1 *Pathchar:* A Tool for Bandwidth Measurement

Pathchar infers link bandwidths along a network path by measuring RTT of packets sent by sender S. The technique used is **by varying TTL**, explained in section 3. Let the network path from S to destination D has H hops. The *Pathchar* works as follows: See figure 4 for reference.

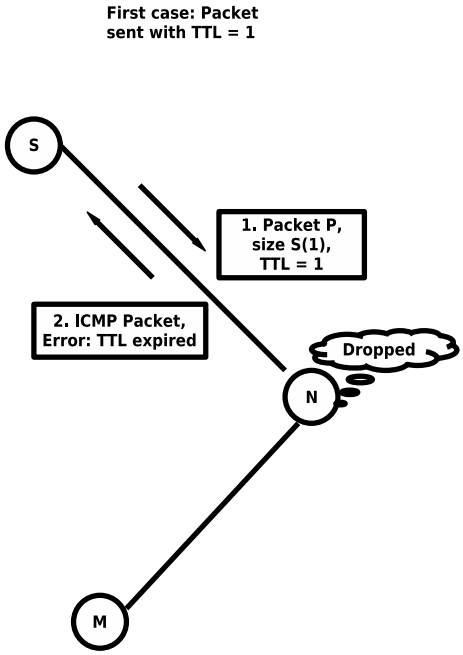


Figure 4: Pathchar setup (TTL = 1)

Initially TTL field is set to 1. Let packet size be S(1). So the hop, which is immediate neighbour of S (say N), will drop the packet and send ICMP, with error code-"TTL Expired" back to S. Let the link between S and N has bandwidth of B(1) and size of ICMP packet be S(ICMP) which is fixed. At S, RTT is calculated, which is actually-

$$\text{RTT}(1) = \text{delay for packet} + \text{delay for ICMP packet}$$

$$\text{So, RTT}(1) = \frac{S(1)}{B(1)} + \frac{S(ICMP)}{B(1)}$$

$$\text{RTT}(1) = \frac{S(1)}{B(1)} + \text{Constant}.$$

By varying packet size S(1), we will get a sample of RTT(1)s. If we observe above equation, it is of the form

$$\text{y} = \text{m x} + \text{C where m} = \frac{1}{B(1)}$$

So with linear regression, we can fit a straight line through these samples, slope of which is reciprocal of the bandwidth B(1). (Refer section 3 for discussion on line fitting.) Thus we infer bandwidth of link 1, which is between S and next hop N.

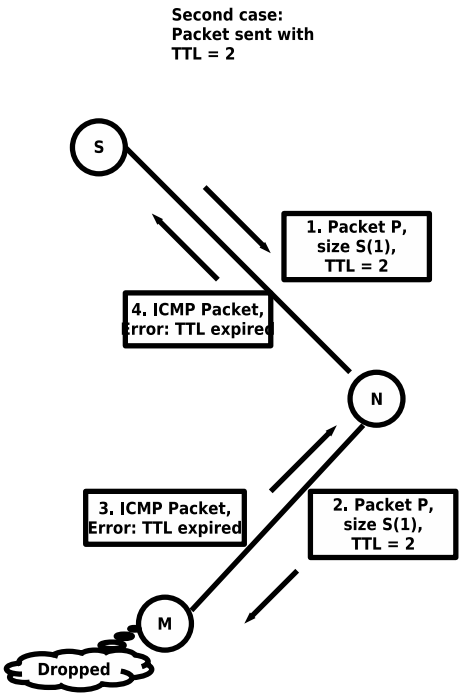


Figure 5: Pathchar setup (TTL = 2)

As shown in figure 5, now we set TTL to 2 and packet reaches to a hop M, which is an intermediate neighbor of N, where it gets dropped and ICMP error message is returned. Let the link between N and M has bandwidth of B(2). With the RTT measurement at S, we have

$$\text{RTT}(2) = \text{S}(1) * \left(\frac{1}{B(1)} + \frac{1}{B(2)} \right) + \text{Constant}$$

By varying packet sizes again, we get many sample values for RTT, which we can now plot. With similar analogy of

$$\text{y} = \text{m x} + \text{C, where m} = \frac{1}{B(1)} + \frac{1}{B(2)}$$

After we substitute B(1) inferred above, we get bandwidth B(2). Continuing this way, we can infer bandwidths of all links. For further references, see [4].

4.2.2 SLoPS to Infer Available Bandwidth

This technique makes use of Self Loading Periodic Streams. Consider a network path with H links between sender S and destination R. Refer figure 6 for setup. Let capacity of link i is denoted as C_i. A link in network path may not be fully utilized always. If utilization of link is U (<= 1), then utilized bandwidth is C_i * U. So in case of $U < 1$, C_i * (1 - U) is unused bandwidth. It is called available bandwidth because given an ongoing traffic on the link, this unused bandwidth is available for other applications. For a network path with H hops, each link has different available bandwidth A_i. The effective available bandwidth of the network path is the lowest available bandwidth.

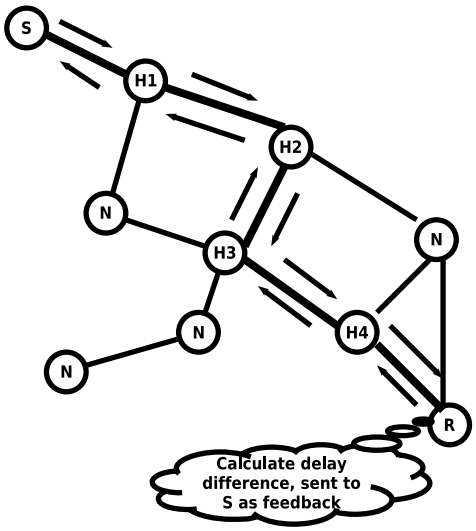


Figure 6: SloPS setup

Thus,

$$\text{A} = \min\ (A_i)$$

Such a lowest available bandwidth link is called **tight link**. SLoPS consist of K packets each of size L sent at constant rate R. OWD for a packet k is D^k. It is measured at destination D. So D^k is given as:

$D^k = \sum_{i=1}^{H}(\frac{L}{C_i} + \frac{q_i^k}{C_i})$ where q_i^k is the queue size at the i^{th} hop faced by the k^{th} packet.

Let $d_i^k = \frac{q_i^k}{C_i}$ where d_i^k is queuing delay at hop i for k^{th} packet. So OWD difference between two successive packets reaching receiver is:

$$\Delta D^k = D^{k+1} - D^k = \sum_{i=1}^{H} \frac{\Delta q_i^k}{C_i} = \sum_{i=1}^{H} \Delta d_i^k$$

With a preposition 1 in [6], we know that if $R > A$, OWDs will be different and when $R <= A$, OWDs will be same. Then an iterative algorithm given below can estimate A. Sender starts with a rate R0. From R_n, and feedback from receiver, it will tune its new rate to R_{n+1} as follows:

$$\text{If } R_n > A, Rmax = R_n$$
$$\text{If } R_n <= A, R_{min} = R_n$$
$$R_{n+1} = (R_{max} + R_{min})/2$$

This procedure is repeated till $R_{max} - R_{min} <= \omega$, where ω is some user specified resolution.

The problem with above method is the assumption that available bandwidth remains same throughout the measurement process, which is not true in real time. So we refine the above process as follows. Instead of applying preposition 1 in [6] for a pair of packets, we should look for increasing trend between delay variance in entire stream. Also there might be a case where there is no strict ordering between R and A. With these modifications, they have implemented this technique in a tool called *Pathload*. Below are the modifications:

Detecting Increasing OWD Trend

Let for a particular stream, OWDs are D^1 to D^k. We partition them in the $\lambda = \sqrt{K}$ groups of λ consecutive OWDs and compute median for each group. Let $\bar{D}_i$ be the medians. Following two new measures are defined:

- Pairwise Comparison Test

  $$SPCT = \frac{\sum_{k=2}^{\lambda} I(\bar{D}^k > \bar{D}^{k+1})}{\lambda - 1} \text{ where I(X) is 1 if X holds, 0 otherwise.}$$

  This measures fraction of OWD pairs that are increasing. If there is strong increasing trend, SPCT approaches one.

- Pairwise Difference Test

  $$SPDT = \frac{\bar{D}^{\lambda} - \bar{D}^1}{\sum_{k=2}^{\lambda} |\bar{D}^k - \bar{D}^{k-1}|}$$

  This quantifies how strong start-to-end OWD variation relative to OWD absolute variations during the stream. If there is strong increasing trend, SPDT approaches one.

If either one of the above metrics shows increasing trend, *Pathload* characterizes stream as type I (increasing) else N (non-increasing).

Fleet of Streams

As opposed to previous case, increasing trend is not determined based on single packet stream but on N packet streams having same sending rate R. If a large fraction f of N streams is of type I, increasing trend is assumed and thus $R > A$ is inferred. In other case, when fraction does not show an increasing trend, $R <= A$ is inferred. It may be possible that less than $f * N$ are of I type and less than $f * N$ are of N type, which means there is no strict ordering between R and A i.e. $R >< A$. So we need to revise above-mentioned iterative algorithm as follows:

Revised Iterative Algorithm: In addition to R_{max} and R_{min}, G_{max} and G_{min} are kept. In third case when $R >< A$, $G_{max} = R_n$ if $G_{max} < R_n < R_{max}$ and $G_{min} = R_{min}$ if $G_{min} > R_n > R_{min}$. In case at some point we detect $R >< A, R_{n+1}$ is halfway between G_{max} and R_{max} if $R_n = G_{max}$ and R_{n+1} is halfway between G_{min} and R_{min} if $R_n = G_{min}$. If we have not yet detected $R >< A$, we can apply previous algorithm itself.

This way *Pathload* infers available bandwidth. Refer [6] for further details.

4.3 Network Monitoring

4.3.1 Placing Network Monitors in an Efficient Way

In passive measurements, where ongoing network flows are sampled, it becomes important to place network monitors at the locations where it proves to be both efficient and cost-effective. The aim should be to come out with such a placement of network monitors that will use optimal number of network monitors still covering much of the network flows. With this purpose in mind, one can formulate this problem as follows:

Let's start with the components of cost of placing monitors. Placing a network monitor incurrs deployment cost which involves monitor's hardware/software cost, space cost and a maintenance cost. So for a cost-effective way, number of monitors should be smaller. Also one has to decide whether a placed monitor will capture all the flows or only some of the flows passing through the link. This could be controlled by configuring the monitors. The cost of per packet capture involves link speed. Also the collected information has to be transferred to a centralized location which also adds up to cost. Thus, the problem of efficiently placing network monitors is an optimization problem which in one way is depicted from figure 7. This optimization problem is the simplest one in which total cost is restricted and aim is to maximize the monitoring reward (i.e. to get maximum coverage). In this way, many problems can be formulated based on parameters given in figure 7.

Few of the important parameters for such a formulation are:

- f_i is the deployment cost of monitor being placed at link i.
- y_i indicates (0 / 1) whether a link i is being monitored or not.
- c_i is the cost of operating [2] monitor on link i.
- m_{ij} is the fraction of flow j sampled at link i.
- ρ_j is the traffic demand of flow j.
- v_j is the benefit gained from sampling flow j.

Thus following costs can be calculated, where L and D are all feasible links and set of all flows respectively.

- Deployment cost:

$$C_D = \sum_{i \in L} f_i y_i$$

- Operating cost:

$$C_O = \sum_{i \in L} y_i c_i \sum_{j \in D} \rho_i m_{ij}$$

- Monitoring reward:

$$C_M = \sum_{j \in D} v_j$$

Costs involved:	
Deployment Cost = sum of costs of deploying all monitors	Formulation as Greedy Problem Maximize Monitoring coverage
Operating Cost = total amount of traffic from all flows j for link i	Subject to Deployment cost

Figure 7: Network monitoring problem

For further details of these problems and their formulation, refer [9].

[2] Cost of Operating captures the cost of sampling flows

4.4 Network Tomography

4.4.1 Bayesian Inference to Identify Lossy Links

We try to estimate bottleneck link by passive measurements, i.e. observing end-to-end network traffic. Refer figure 8. For Bayesian inference, we need an observed parameter, which is evidence and a calculated metric, which is hypothesis. In this technique, we refer to evidence as D, which is packets transmitted and lost, as evidence and θ, which is loss rates of links, as hypothesis. Thus, θ is a vector here, say $l_1, l_2, \ldots, l_L$ and D is number of packets transmitted successfully and number of packets lost, s_j and f_j respectively for client c_j.

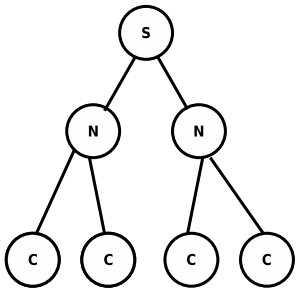


Figure 8: Tree topology

Thus θs will affect D. And thus, form the law of total probability [15],

$$P(D) = \int_\theta P(\theta)P(D|\theta)d\theta$$

With Bayes' theorem,

$$P(A|B) = \frac{P(B|A)*P(A)}{P(B)}$$

Let's define a term called P_j denoting loss rate observed at client c_j. Let the path from server to client has T_j links. If we consider any link i in this group of T_j links, having loss rate l_i, (as per above notation), then with probability $(1 - l_i)$, a packet will be successfully delivered over that link. Similarly, taking into consideration entire path of T_j links, with $\Pi_{i \in T_j}(1 - l_i)$ is the probability that packet is transmitted successfully over all the links in T_j. Thus, P_j is given by the probability of packet being lost on any of the T_j links. Thus,

$$P_j = 1 - \Pi_{i \in T_j}(1 - l_i)$$

Thus with binomial theorem, the distribution of s_j and f_j for client c_j is given as $(1-P_j)^{s_j} * P_j^{f_j}$.

Now $P(D|\theta)$ can be written as the probability that we observe this loss for all clients j and thus,

$$P(D|\theta) = \Pi_{j \in clients}(1 - P_j)^{s_j} * P_j^{f_j}$$

With these, the method of estimating bottleneck link is as follows: Initially, we set all l_i to some arbitrary value depending on nature of links. But we can also start setting all to a uniform value of 1. So now in the above equation, we have to find $P(\theta|D)$ which will give idea about lossyness of all the links and help us in estimating bottleneck link. Here we sample each link; say we consider a link i among them. Thus,

$$P(l_i|D) = \frac{P(D|l_L)}{\int_{l_i} P(D|l_L)dl_i} \text{ given } l_i\text{s are set to 1, initially.}$$

In this way, we sample all the links one by one and get new estimated values for their loss rates.

4.4.2 Alternatives to Bayesian Inference

One can also use Randon Sampling instead of Bayesian Inference to estimate lossy link. In Randon Sampling, each link is sampled one by one, some loss rate assigned to it and with the residual loss rate distributed among remaining links. For details, refer [11]

4.5 Network Modelling and performance Analysis

4.5.1 HMM for Internet Communication Channel

For basics about HMM, refer section 3.4. Consider a network channel whose state is hidden whereas what we see is an observation of loss process say

$$X = (X_i)_{i=1}^{T}$$

where $X_i = 0$ or 1 when packet is received or lost respectively.

Let's assume that the channel switches between K states, such as

$$\text{spate space is } S = (1, 2, \ldots, K)$$

We model this state space as Markov Chain of $Y = Y(t)$ The transition probability matrix for this HMM is give by:

$$\Lambda(i, j) = Prob(Y_{t+1} = j | Y_t = i)$$

Once this modelling is done, with the Expectation-Maximization algorithm (refer [12]) is used to re-estimate network parameters. For this to work, we need data traces,as this involves training the model initially. Figure 9 shows such modelling.

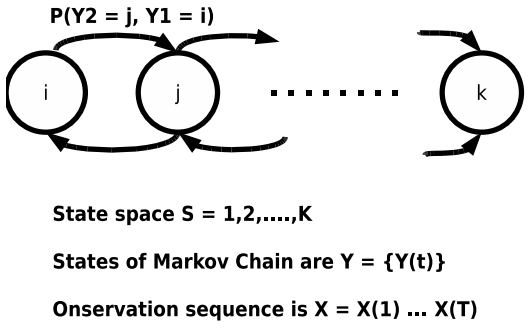


Figure 9: Hidden Markov Modelling

For details of this modelling, refer [8].

4.5.2 More on this

Here we have considered packet loss as the only observable measure in order to esimate hidden state of the network. But one can also use delay to model network. The problem with delay is, it is a continuous parameter and thus difficult to model than packets lost which

is a dicrete parameter. Also for correct mapping between observation and inference of states, clocks between sender and receiver need to be synchronized. Taken care of this, one can also model network based on delay observations.
Non-HMM modelling can also be considered which increases number of states. HMM uses lesser number of states with added complexity.

4.6 Internet Routing and Path Properties

4.6.1 Measuring and Classifying Out-Of-Sequence Packets

Out-of-sequence packets is an indication of network performance. There are various causes that result in Out-of-sequence packets:

- Packets may get lost and thus some sequence numbers get skipped in between.
- Due to looping, a packet which is already seen may be received again, which is out-of-sequence.
- Packets may get duplicated due to broadcasting which may be repeatedly received.

Measuring the fraction of such out-of-sequence packets and classifying them according to their causes helps in finding out root cause that affects performance. Suppose, a packet is getting braodcasted, then the goodput[3] of the system might be low. But the reaon here is not lossy links because of which packets get lost but broadcasting. So finding the causes that affect network performance is important.

This can easily be done with Passive Measurements. Refer section 1.3 for Passive measurements. We can capture first few bytes of IP and TCP packets,which have fields like sequence number, packet ID etc., passing across a link. Considering data packets in both directions, we will get two traces. Once traces are collected, one can identify an Out-of-sequence packet as follows: If sequence number of current packet is less than or equal to that of previous packet on that link, current packet is out-of-sequence. Once this identification is done, few protocol mechanisms are applied in order to classify these packets according to their causes. Few of these protocol mechanisms are:

- Number of duplicate packets seen
- Time lag between the packets
- Whether an ACK for this (data) packet is seen or not
- (From the IP header,) whether IP packet ID is different than the one in previously seen packet or not

The intuition behind such tests lies in the way the protocol headers are designed and the protocol mechanisms are defined. One such simple test is given in figure 10.

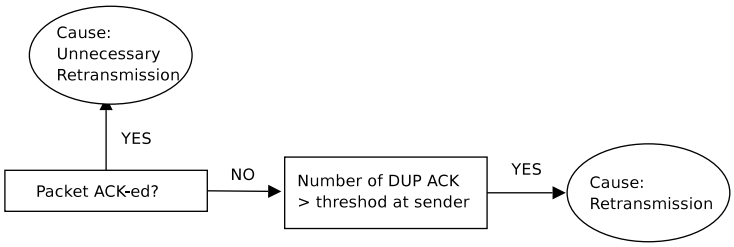


Figure 10: Out-of-Sequence packets: Cause classification

[3]Goodput: Here referred as useful packets received per unit time

4.6.2 RTT Estimation

This technique that exploit the protocol mechanism like RTT, RTO of TCP needs to know (or estimate) the value of RTT, RTO. This estimation can also be done using similar method described above. For instance for estimation of RTT, one can consider following. Refer figure 11 for details. The RTT at sender can be considered to be equal to the round trip time from measurement point to the receiver (measured through ACK, as d1 in figure) and round trip time from sender to measurement point (measured through Data, as d2 in figure).
According to me, this assumption that sender will transmit data at the moment it receives ACK may not be possible in all cases, which might result in wrong estimation.

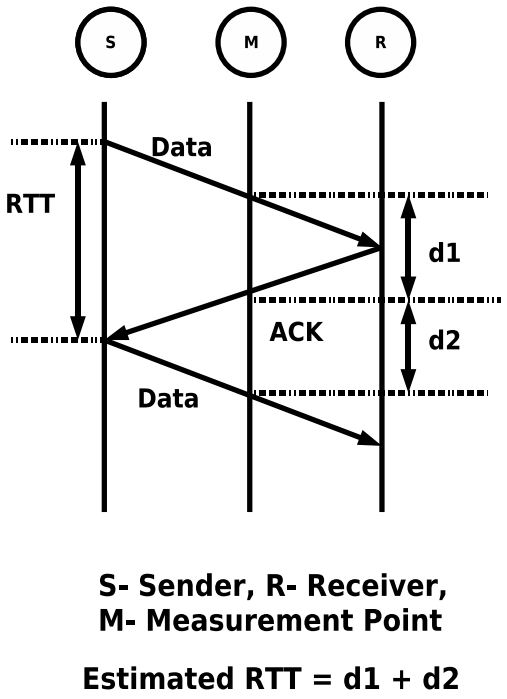


Figure 11: RTT Estimation using single point observations

4.7 Traffic Engineering

4.7.1 Web traffic analysis

Let's look at few important concepts:

- Heavy-tailed distribution: The term *heavy-tailed* means that the probability of value of a random variable begin higher is considerably high. Thus the tail of such distributions does not decay fast, in fact it is heavier than that of exponential distribution. For reference, see [13]. Example is Paerto Distribution, whose definition is as follows:

  $$Prob(X > x) = \left(\frac{x}{x_m}\right)^{-k}$$

  So any distribution of the form

  $$Prob(X > x) \sim x^{-\alpha}$$

  is considered as a heavy-tailed distribution. Refer figure 12 for details. (This figure is taken from [7])

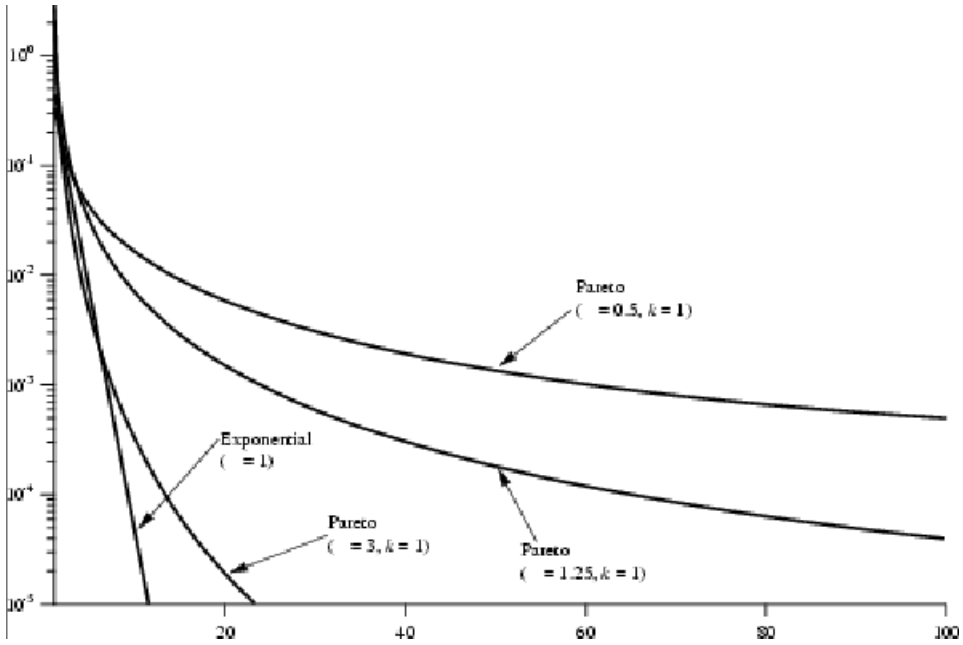


Figure 12: Pareto and Exponential Probability Density Functions

- Self-Similarity: **Self-Similarity** is a property that we associate with an object like a fractal, whose appearance remains same irrespective of the scale at which we view it. This concept when applied to stochastic processes, it means that such stochastic process exhibits bursts, which are extended periods above mean, over a wide range of timescale. Refer [16] for mathematical notations and explaination about self-similarity. Given a timeseries data (stochastic process data), there are few tests for checking if that data follows self-similarity. Refer [3] for details on the tests.

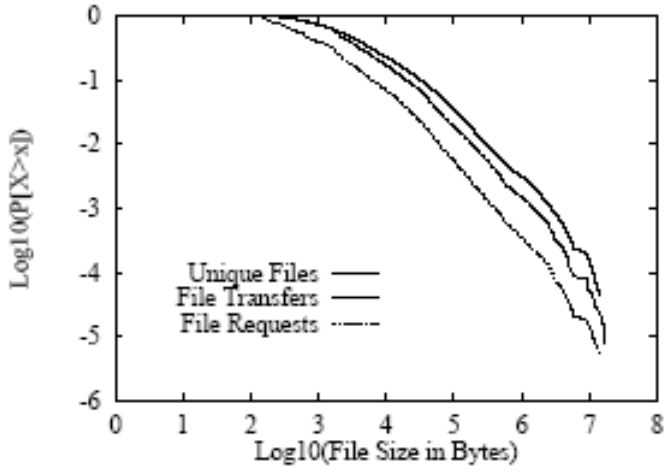


Figure 13: Plot of various file distributions for web traffic analysis

Normally it is assumed that the web requests arrive with Poisson distribution. But if one collects traces from web traffic, one understands that the distribution of request arrivals, file sizes transferred on internet etc. are heavy tailed. For these traces, one needs some

timestamped events happening at the end user as well as web server. Also from such traces, one needs to show that the traffic follows this distribution. Observations show that self-similarity is present in web traffic, which indicates self similar nature of distributions of many events happening on the web. Thsese events include interarrival requests from the user, which indicate user's think time[4], size of files transferred over the web etc.
For instance, see figure 13. It shows LLCD plot of file transfers, file requests and unique files. This is taken from [3]. In this case, finite caches are present which are shared by multiple users. If caches are performing well, set of files transferred would be identical to set of unique files. This results in one miss per file. The cache will not take into consideration the size of the files but the access pattern. So when the cache is performing effectively, available files are the key determiner of the distribution which is heavy tailed. [3] gives many such observations and reasons behind them.
Such web traffic analysis throws light on some aspects of human information processing, availability of web cashes and thus indicate that protocol changes won't help remove this self-similarity as this is not a machine-induced effect.

In this section, we have seen few of the examples and discussions on the techniques that we expalined in section 3. There are many such techniques, which have many applications, details of few of them can be found in references.

[4]Think time refers to the time between a user gets response and a next request is generated by that user.

5 Conclusion

Network measurements are very important in order to monitor network performance, utilization as well as to diagnose problems occuring in the network. The objective of network measurements should be to measure useful parametes of network in order to estimate useful metrics with minimum overhead of additional traffic.
To summarize, there are mainly two categories of measurements, **active** and **passive**. Active measurements have a risk of introducing additional traffic and thus adding overhead which might affect measured and thus estimated network properties. Passive measurements only monitor network without introducing extra traffic and infer from the information received.
Some techniques use single packet probes, packet pairs or a packet quartet (a pair of packet pair) to measure delay and delay variance, which mainly indicate queuing delay and propogation delay. Such techniques give an indication about the queues at the routers and inference about the bandwidths of the links. Some other techniques use a parameter-number of packets lost over a connection, which is an indication of the loss rates of the links.
Some traces can be collected based on passive measurements, which are used to model the network or explaining various phenomenons of the network, like Out-of-sequence packets, packet loss. For problems like expanding existing network capacity in terms of additional users (and thus additional traffic), the notion of utilized and available bandwidth are important. Most of these bandwidth estimation techniques use active measurements, which introduce additional traffic, which can affect the bandwidth estimated to some extent. Also available bandwidth changes over time whereas the bandwidth estimation tools are run only at a particular instance of time, which is unlikely to give a bigger picture.
Some techniques use sniffing of protocol headers like capturing IP and TCP protocol headers for classifying Out-Of-Sequence packets. If the headers are encrypted for security reasons [5], exploiting this method is not possible.
Some analysis of web activity at the end-user (like user requests, files downloaded, size of such files, etc.), provides important inferences like traffic shape or trend in network events and help in designing and improving network performance, using only passive measurements.
To conclude, measurements on the network is a vast area covering studies about relationships between various network properties and techniques to measure and estimate them.

5.1 Future Work

Some of the techniques can be implemented practically, like already implemented tools like *Pathchar*, *Pathload*. Also, one can focus on measuring properties in pasive manner for which currently active measurement techniques exist. This might lead to a network measurement framework, wherein for a small campus or organization, a complete network monitoring, measurement and troubleshooting system can be designed and implemented; which will help in many day-to-day troublesome situations like server dropping connections frequently or mail delievery getting delayed. One can also understand a network with better statistical and probabilistic modelling.

[5] This is applied mainly for wireless domain, where protocol headers might be encrypted for preventing eavesdropping, so for wired medium, it is a less likely phenomenon.

Acknowledgements

I would like to thank Prof. Purushottam Kulkarni for helping and guiding me in selecting appropriate reading material, clearing my doubts and for encouraging discussion on the topics read.

References

[1] Darryl Veitch Attila Pasztor. Active probing using packet quartets, IMW 2002.

[2] Giuseppe Bianchi. Performance analysis of the ieee 802.11 distributed coordination function, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 18, NO. 3, MARCH 2000.

[3] Mark E. Crovella and Azer Bestavros. Self-similarity in world wide web traffic: Evidence and possible causes, IEEE/ACM Transactions on Networking, 1997.

[4] Allen B. Downey. Using pathchar to estimate internet link characteristics, SIGMETRICS 1999.

[5] Z. Morley Mao Himabindu Pucha, Ying Zhang and Y. Charlie Hu. Understanding network delay changes caused by routing events, SIGMETRICS 2007.

[6] Manish Jain and Constantinos Dovrolis. End-to-end available bandwidth: Measurement methodology, dynamics, and relation with tcp throughput, SIGCOMM 2002.

[7] Hei Xiao Jun. Heavy-tailed distributions. `http://www.ee.ust.hk/~heixj/publication/comp660f/node4.html`.

[8] Sandrine Vaton Kave Salamatian. Hidden markov model for internet communication channel, SIGMETRICS 2001.

[9] Jim Kurose Don Towsley Kyoungwon Suh, Yang Guoy. Locating network monitors: Complexity, heuristics, and coverage, INFOCOMM 2005.

[10] Jun Liu and Mark Crovella. Using loss pairs to discover network properties, ACM SIGCOMM Internet Measurement Workshop 2001.

[11] Helen J. Wang Venkata N. Padmanabhan, Lili Qiu. Server-based inference of internet link lossiness, IEEE INFOCOM 2003.

[12] Wikipedia. Expectation-maximization algorithm.

[13] Wikipedia. Heavy tailed distibutions.

[14] Wikipedia. Hidden markov model.

[15] Wikipedia. Law of total probability.

[16] Wikipedia. Self similarity.