Evaluating a TCP Model-Based Network Performance Measurement Method

by

Merry Mou

B.S., Massachusetts Institute of Technology (2016)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2017

Author .....
Department of Electrical Engineering and Computer Science
May 12, 2017

Certified by .....
David Clark
Senior Research Scientist
Thesis Supervisor

Accepted by .....
Christopher Terman
Chairman, Masters of Engineering Thesis Committee

Evaluating a TCP Model-Based Network Performance Measurement Method

by

Merry Mou

Submitted to the Department of Electrical Engineering and Computer Science
on May 12, 2017, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

For the end-to-end measurement of network speed, I implement and evaluate a network performance measurement test framework, which allows for the incorporation of expectations and models specific to the network being tested, as originally proposed in the Model Based Metrics (MBM) IETF Internet-Draft [14]. Specifically, I construct two tests within the test framework that measure a network's bulk transfer capacity: the Basic Data Rate Test and Sustained Burst Test. I analyze the models used by MBM to specify measurements that are independent of their vantage point, to statistically evaluate measurements, and to compose measurements over subpaths. I also empirically evaluate the two implemented tests in a virtual network setup. MBM offers a framework for comprehensive and modular end-to-end measurement of network performance.

Thesis Supervisor: David Clark
Title: Senior Research Scientist

Acknowledgments

I am grateful for Steven Bauer's continual guidance and support on all fronts, without which this thesis would not have been possible.

Matt Mathis and Arthur Berger offered constructive technical feedback and opinions regarding derivations in my thesis.

David Clark, the Advanced Network Architecture Group, and the Internet Policy Research Initiative gave a home and a beacon for this research, rooted in a holistic approach to technology.

During my masters, Nathan Matias and the Center for Civic Media have deeply shaped and encouraged my interests in civic-oriented research, and have played an important role in developing my ability to ask and answer research questions - to be a researcher.

Research is sustained by community. The weekly Advanced Network Architecture student meetings, led by Steven Bauer and Karen Sollins, and the weekly Center for Civic Media meetings, led by Ethan Zuckerman, have offered an incredible camaraderie and education. In addition, countless family, friends, and teachers have inspired me everyday to do what matters to me; in particular I would like to thank Allan Ko, Esther Jang, and my former and current housemates in East Campus and pika.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Network Performance and Measurement

Network measurement is important to a multitude of stakeholders, not only for technical reasons, but also for regulatory and political reasons. As internet access becomes more available globally, it becomes increasingly important to ensure good internet service, not only to fulfill expectations between vendors and customers, but also to honor legal agreements and regulations between vendors, governing bodies, and the public. What qualifies as good internet service is not a simple question to answer; it may be characterized by its reliability, speed, fairness, and other factors. Moreover, it is not very easy to measure and determine whether or not a particular internet service is good.

We focus on the second issue, of determining whether a network is performing as well as it is expected to—in other words, network measurement. In particular, we analyze the existing limitations of some network performance measurement tests today, observe how network measurement will need to adapt to a a future of increasingly higher capacity links, as well as consider how to design network measurement methods for a world that requires ubiquitous network measurement.

How a network's performance is measured depends on what conditions the network is expected to be in. Here, conditions is a non-technical term referring to anything that a network exhibits, and therefore a test should simulate, including: expected

network paths that traffic will travel, amount and nature of concurrent traffic on links, and configurations of protocols being run by the nodes.

Understanding and incorporating such conditions for network measurement is especially important when one believes that measurements should be taken to assess "quality of service," or a user's satisfaction and experience with a service, which in this case, is internet service. "Quality of service" is one way to interpret what "good" network performance measurement is.

In this thesis, within network performance measurement we focus on speed measurements, which are often used to assess a network's quality of service. Even the characteristic of speed has varying definitions and measurement methods [9]; we choose to focus on bulk transfer capacity (BTC), defined as the largest number of bits that a network can be expected to deliver via some congestion-aware protocol from point A to point B over some elapsed period of time, in the long run [5]. This measurement is essentially what several popular network speed tests that are aimed for consumer use, including Internet Health Test and M-Lab's Network Diagnostic Test, seek to measure [9]. Alarmingly, even among different BTC tests, results may vary greatly [10]. An overview of network measurement, BTC, existing BTC tests, and the reasons for their results' variation is in Sections 2.1 and 2.2. One fundamental reason for why these measurements depend so much on the network and testing conditions is the self-regulating nature of congestion-aware algorithms, such as Transmission Connection Protocol (TCP), further described in Section 2.3.

The interdependence between congestion-aware algorithms and the methods used to measure them is complicated, and sometimes, undesirable. In Section 2.4.1, we describe an example of when the self-regulating nature of congestion-aware protocols results in potentially misleading measurements, and we introduce the concept of vantage point independence as a desirable trait of network performance measurement methodologies. Vantage point independence helps decouple the measurement of a network path's throughput from its round trip time in a way that facilitates the measurement of end-to-end performance from piecewise measurements of individual subpaths. This sort of modular measurement enables greater flexibility in the ways

we test network performance.

In designing network measurement methods, we additionally anticipate trends in increasing link capacities in Section 2.4.2, while a typical individual user's expected end-to-end usage will not grow proportionally. Typical traditional BTC measurements are taken by essentially flooding the network to the point of congestion, but this will be increasingly impractical and inappropriate as network capacities increase.

Beyond these considerations, we also aim to design network measurement methods that facilitate ubiquitous—routine and frequent—measurement. Essential to making measurements understandable and meaningful is to allow for easy incorporation of our specific expectations regarding network behavior and performance. This motivation for an easily configurable test framework is described in Sections 2.4.3 and 2.5.

1.2 Model Based Metrics for Bulk Transfer Capacity Measurement

This thesis implements Model Based Metrics (MBM) [14], an IETF proposed network performance measurement test framework. The general test framework allows one to input target performance values (what characteristics we require a network to exhibit) and network models (our understanding of a network's behavior, that may affect our measurement of it) to run suites of performance tests specific to a particular kind of network. The test framework, described further in Sections 3.1 and 3.2, uses the target performance inputs and network models to generate test traffic to send over the path of interest. The path's performance is evaluated by collecting measurements and using the network models to statistically determine whether or not a path is expected to be able to support the given target performance inputs.

We construct and evaluate two tests, the Basic Data Rate Test and Sustained Burst Test, which one might use in a test suite for BTC measurement. The Basic Data Rate Test is used to measure throughput, round trip time and loss rate with a constant bit rate stream. The Sustained Burst Test additionally tests a network's

ability to handle packet bursts. We use and evaluate a particular model of TCP Reno by Mathis [4] for the test stream generation and statistical estimations.

MBM uses network models to define a loss measurement that exhibits vantage point independence, which can be used to determine whether or not a path will be able to support the target performance inputs, described in Section 3.3. We also define the concept of measurement composition, which allows for the estimation of complete path measurements from piece-wise measurements of subsequent subpaths. In Section 3.4, we explain the statistical estimation of this loss measurement for a given link, using the hypothesis test Sequential Probability Ratio Test. We place the statistical test in the context of the overall test evaluation process in Section 3.5. Our implementation of MBM is described in Section 3.6.

To evaluate MBM, we use a virtual network setup, as described in Section 4.1. We empirically evaluate the Basic Data Rate Test on an interconnection link example first introduced in Section 2.4.1 by varying a test's target round trip time and observing the test results in Section 4.2. We then evaluate the Sustained Burst Test by varying the burstiness of the test stream and observing the loss rates and test results in Section 4.3. Finally, we investigate the loss rate measurement composition process that allows for the piece-wise measurement of subsequent subpaths by evaluating the assumption of link loss rate independence in Section 4.4.

1.3 Contributions

The contributions of this thesis are:

1. Implement a network performance measurement test framework that allows for the incorporation of expectations and models specific to the network being tested, as originally outlined in an IETF Internet-Draft [14].

2. Construct two tests within the test framework that evaluate bulk transfer capacity of a link: the Basic Data Rate Test and Sustained Burst Test.

3. Analyze the network models, assumptions, and derivations used in the test

framework, for the construction of vantage point independent measurements, the statistical evaluation of measurements, and the composition of measurements over subpaths.

4. Evaluate the Basic Data Rate Test and Sustained Burst Test in a virtual network setup.

Chapter 2

TCP Performance and Measurement

2.1 Network Measurement

Network performance measurement aims to determine various characteristics of a network, including bandwidth (availability and utilization), loss (average and variance, and patterns over time), and round trip times (average and variance, and patterns over time). Network measurement is of interest to a variety of stakeholders, including network administrators, ISPs, vendors, policy makers, and consumers [12]. The method used in a particular situation depends on the specific stakeholders and goals in mind.

Overall, network measurement methodologies may be categorized as either passive or active measurement. Passive measurement methods do not send their own packets, and instead monitor existing traffic on the network, whether to investigate the nature of existing traffic, to diagnose or troubleshoot problems, or to make inferences about maximum bandwidth or other characteristics of a network. On the other hand, active measurement methods generate and send test packets, designed to probe the network to obtain certain measurements.

Of performance measurement, network speed is often the most popular measure [9]. In fact, what is called broadband internet access technology, which is available almost globally today [1], is defined by its speed: it is high-*speed* internet, typically faster than dial-up. However, speed may refer to several characteristics. Speed can

mean the physical capability of a network link to move data (often called capacity), or it can refer to the average speed at which a particular user can send traffic due to the unused/available bandwidth on the link given other traffic on the link. It can also refer to the throughput that is achieved by some higher-level rate-controlling transport protocol (such as TCP) [9].

In addition, when considering network performance, stakeholders are often interested in measurements that reflect "quality of service," a term to describe a user's satisfaction and experience with a service, which in this case, is internet service. To characterize quality of service, the measurement of end-to-end performance (and if we are focusing on speed, end-to-end speed) is often used to describe the performance of a network from a typical user to a typical destination server (in other words, from end to end). The end-to-end performance of a network will depend on a variety of conditions, including not only the raw link capacities, presence of simultaneous traffic on the link and available link bandwidths, and throughputs achieved by congestion-aware transport protocols, but also the nature of the traffic (number of streams, traffic patterns) that the user wants to send, the routing algorithm's path selection, as well as an end user's and server's machine and connection configurations.

A simple example of what might differentiate a particular link's speed from an end-to-end speed measurement is that even if a particular link's speed (capacity, available bandwidth, or throughput) is quite high, if an end-to-end path includes a bottleneck neck, the bottleneck will dictate end-to-end speed. Another simple and common example of how speed measurements are based on particular contexts is that networks' download and upload speeds will often differ, and meaningful end-to-end speed measurements will need to differentiate between these two use cases.

Many network performance measurement tests today, especially those aimed for typical consumer use, such as the Internet Health Test, DSL Reports Speedtest, M-Lab's Network Diagnostic Test, and Netflix's Fast.com, run active measurement tests to produce speed measurements that intend to reflect a user's network's end-to-end performance. These publicly-available and browser-based measurement tests are widely used, but can give highly variable results, even though they intend to, on a

high level, measure the same characteristic: a given user's end-to-end speed [10].

As one would expect, the factors described above that affect end-to-end performance likewise affect tests that measure end-to-end performance. To be more specific, tests differ in the number of flows they send (simultaneously or sequentially) and selection of and number of destination test servers. In addition, tests differ in what they report: some will report total bytes sent over total time, others will cut off the initial throughput ramp-up and only report the average throughput after the ramp-up, while others will run the test several times sequentially then report an average of each test's total throughput result. These details are further described in Bauer, Lehr, and Mou's work [10].

The high variability of results from such end-to-end performance measurement tests is particularly of concern as network measurement becomes more important in the regulatory space. Some of these tests are being used to inform legal and political decisions, such as the use of the Internet Health Test by the New York Attorney General's office [10]. The increasing importance in network measurement demands more careful consideration of how we design and specify our network performance measurement methods and requirements.

2.2 Bulk Transfer Capacity Measurement

Of active measurement methods for speed, the most commonly used methodology measures a path's bulk transfer capacity. In fact, bulk transfer capacity is what best describes the characteristic that the network performance measurement tests mentioned above intend to measure [9]. A link's bulk transfer capacity (BTC) is, intuitively put, the largest number of bits that it can be expected to deliver via some congestion-aware protocol from point A to point B over some elapsed period of time [5].

In the interest of making end-to-end measurements of speed, we focus on BTC. From here, when we refer to speed we shall mean BTC, and we will focus on the congestion-aware protocol Transmission Connection Protocol (TCP).

The end-to-end speed measurement of BTC is useful for understanding a path's behavior when, for example, transporting a large file or streaming a high-rate video. In fact, most simply put, the most commonly used methodology essentially sends a large file over a network path and sees how long it takes. However, as described above, implementations vary widely in the measurements they report, due to design variations in the number of test streams sent, the selection of and number of test server destinations hit, as well as exactly what measures are calculated and reported.

To explain why these factors can affect the resulting end-to-end speed measurement so much, we take a closer look at TCP and the effects of congestion control on achieved and measured speeds. TCP naturally tries to maximize its sending rate; due to its self-regulating nature, the maximum rate that TCP can achieve depends on how the congestion control algorithm works. In particular, with the case of the TCP Reno congestion control algorithm, the maximum throughput achieved will depend factors including round trip time and the rate of random packet loss [4].

2.3 TCP Congestion Control

The main role of the Transmission Control Protocol (TCP) in the Transport Layer of the Internet protocol suite is to improve the transmission and reliability of flows by adapting and maximizing transmission rates, and identifying and resending dropped packets.

The congestion control algorithm determines the timing of packet transmissions, with the goal of sustainably sending as many packets as it can. A common definition for congestion is the event of packet drops in an already-filled buffer [8], signaling that the network is or may soon hit resource limits, and will not be able to support the current throughput for long. The concept of congestion can be understood to be the direct result of resource-sharing, due to limits on link available bandwidth and queue sizes, for example.

TCP attempts to identify states of congestion, in order to adjust packet send rate accordingly. Implementations of TCP specify a congestion signal: upon receiving a

congestion signal, TCP will decrease the sending rate. The most common congestion signal utilizes the acknowledgement packets (ACKs) that are sent by the receiver every time it receives a packet from the sender. If the sender receives a number of ACKs that indicate that no more-recently sent packets were successively received by the recipient, it might be due to packet drops induced by filled buffers or due to network connectivity issues—all reasons to suspect that the network congested due to resource availability limits. Additionally, this congestion signal not only affects the sending rate, but also helps in the identification and resending of dropped packets, thus achieving another goal of TCP, which is to improve the reliability of stream transmission.

Another kind of congestion signal used by congestion control algorithms include explicit notifications of queue build-up, aptly named Explicit Congestion Notifications (ECNs), which don't rely on packet drops to convey congestion states. Algorithms also differ in exactly when they send the congestion signal: for example, the congestion signal can be sent when congestion is expected to happen if the sending rate is not decreased, rather than when some buffer is already full, at which point congestion is arguably well underway and adversely affecting network performance.

TCP Reno is one particular implementation of TCP congestion control, and its congestion signal depends on the ACKs that the traffic sender receives from the traffic receiver. How quickly a traffic sender receives these ACKs, which affects how quickly a sender can readjust its window size (and therefore sending rate), depends on the round trip time between the sender and the receiver.

To be more specific, TCP's send rate is determined by the sender's window size, which is the total number of un-ACKed packets that can be outstanding at any given time. Assuming the additive increase/multiplicative decrease (AIMD) scheme, when a congestion signal is received, TCP will multiplicatively decrease its window size. Otherwise, TCP will additively increase the window size, in order to more fully utilize the link. The AIMD scheme for setting the traffic sender's window size (which affects sending rate) causes window size to increase and decrease in a sawtooth pattern, as the sender receives congestion signals caused by packet loss. The window

size increases in increments of one packet until a congestion signal is received, upon which the window size will drop to half its original size, and this process repeats itself.

All of this intuition helps one understand the following formula first proposed by Mathis, Semke, and Mahdavi, which bases upon the window size's sawtooth pattern a derivation of the relationships between throughput t, maximum transmission unit (largest expected packet size) MTU, round trip time rtt, and a constant nonzero random packet loss rate p (a more detailed description of the derivation can be found in the original work [4]).

For a network with periodic loss and a receiver that acknowledges every packet, t can be modeled as:

$$t = \frac{\text{data sent per cycle}}{\text{time per cycle}} = \frac{MTU \times C}{rtt \times \sqrt{p}}, \text{ where } C = \sqrt{\frac{3}{2}} \qquad [4] \qquad (2.1)$$

In other words, the rate at which TCP sends packets decreases as round trip time increases and as packet loss rate increases. (Also, recall that packet losses induce window size decreases; if no packets are lost ($p = 0$) and there are no other mechanisms limiting throughput, then TCP will theoretically continue to increase the window size, and send packets faster and faster.)

2.4 Limitations of Bulk Transfer Capacity Methodology

The speed (essentially BTC) measurement methods used today, some of which were described above in Section 2.2, though all intending to measure end-to-end BTC, vary widely in implementation and results. Through an understanding of TCP, we see that some of these variations are explained by TCP's dependence on round trip time for throughput: depending on the round trip times between the user and the selected test servers, one's throughput measurement will differ. This may seem like a desirable result as long as the round trip times between the user and the test servers

are representative of the expected end-to-end round trip time in a typical use case. However, there are situations where the self-regulating nature of TCP and the effects of round trip time on throughput are not desirable. We present a motivation example in Section 2.4.1, and with it, introduce the concept of vantage point independence as a desirable trait of network performance measurement methodologies that allows for modular measurement of end-to-end performance.

Current BTC measurement methods also face considerable challenges when we consider a near future of higher-capacity (in the gigabit range) links. Not only have methods been shown to give highly variable results in such situations [10], but also one should consider how to sufficiently test such high capacity links without flooding the link when it is not necessary. We believe that there are many typical use cases in which such maximum throughput tests are not necessary. Maximum throughput tests on high capacity links require more time and more test traffic, and may adversely affect other users who send traffic on links undergoing testing. We elaborate and consider alternatives in Section 2.4.2.

In addition to preparing for a future of higher capacity links, we should also prepare to create a future of more prevalent network connectivity that demands ubiquitous network testing, as we discuss in Section 2.4.3. Such network test frameworks should be easily configurable—placing assumptions and expectations in the forefront of network measurement methodology design, test deployment, and results analysis. A world of ubiquitous network testing would have tests designed for routine and modular testing, achieved by vantage point independent measurements and informed choices of test traffic.

2.4.1 Vantage point independence

To better illustrate why a link's round trip time may have an undesired effect on a link's measured throughput, consider a scenario, first described by Mathis [13], where one would like to investigate the performance of a busy interconnection link that is suspected to be a bottleneck (Figure 2-1). The link's round trip time is small, and a lot of traffic utilize the interconnection link as it travels from the sender through one

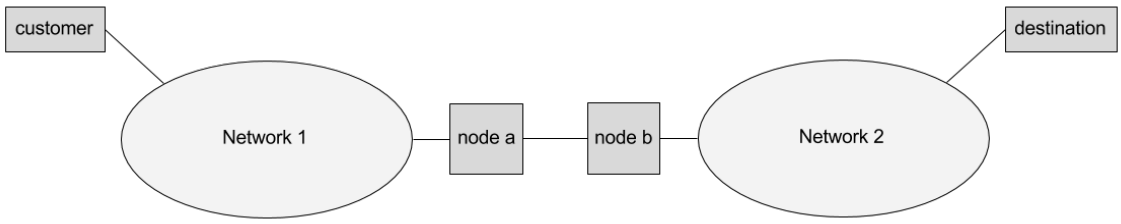

Figure 2-1: Interconnection link example.

ISP to another ISP.

To diagnose that specific link, the naive approach is to measure the throughput of the interconnection link by, as typical BTC tests do, sending test traffic only across the interconnection link (in the figure, from node a to node b). Such a test may find that, due to the link's very short round trip time, the link seems perfectly capable of supporting the required throughput. In other words, the sender is alerted of packet drops (or more generally, the congestion signal) relatively quickly, and is able to adapt its rate more quickly than if congestion signals had to travel the longer, actually expected round trip time between the typical end-user and the typical destination. Instead, in a typical end-to-end use case, with a much longer expected end-to-end round trip time, the test traffic over the interconnection link would need to experience a much *smaller* loss rate if it is to support the required throughput over an end-to-end path's expected round trip time.

Even though the interconnection link is the suspected bottleneck, directly testing on the interconnection link with typical BTC methods will not give accurate results because the endpoints of the interconnection link are not the endpoints of the typical user and destination. In other words, these BTC measurements are not vantage point independent. A measurement of a given link exhibits vantage point independence if the measurement is not affected by from which endpoints it is actually measured, given that that link is included in the test and is the bottleneck.

To better understand the effects of round trip time on throughput, the Mathis theoretical relationship between round trip time, throughput, and loss rate (Equation 2.1) is illustrated in Figure 2-2 with a couple representative loss rates. The log-log

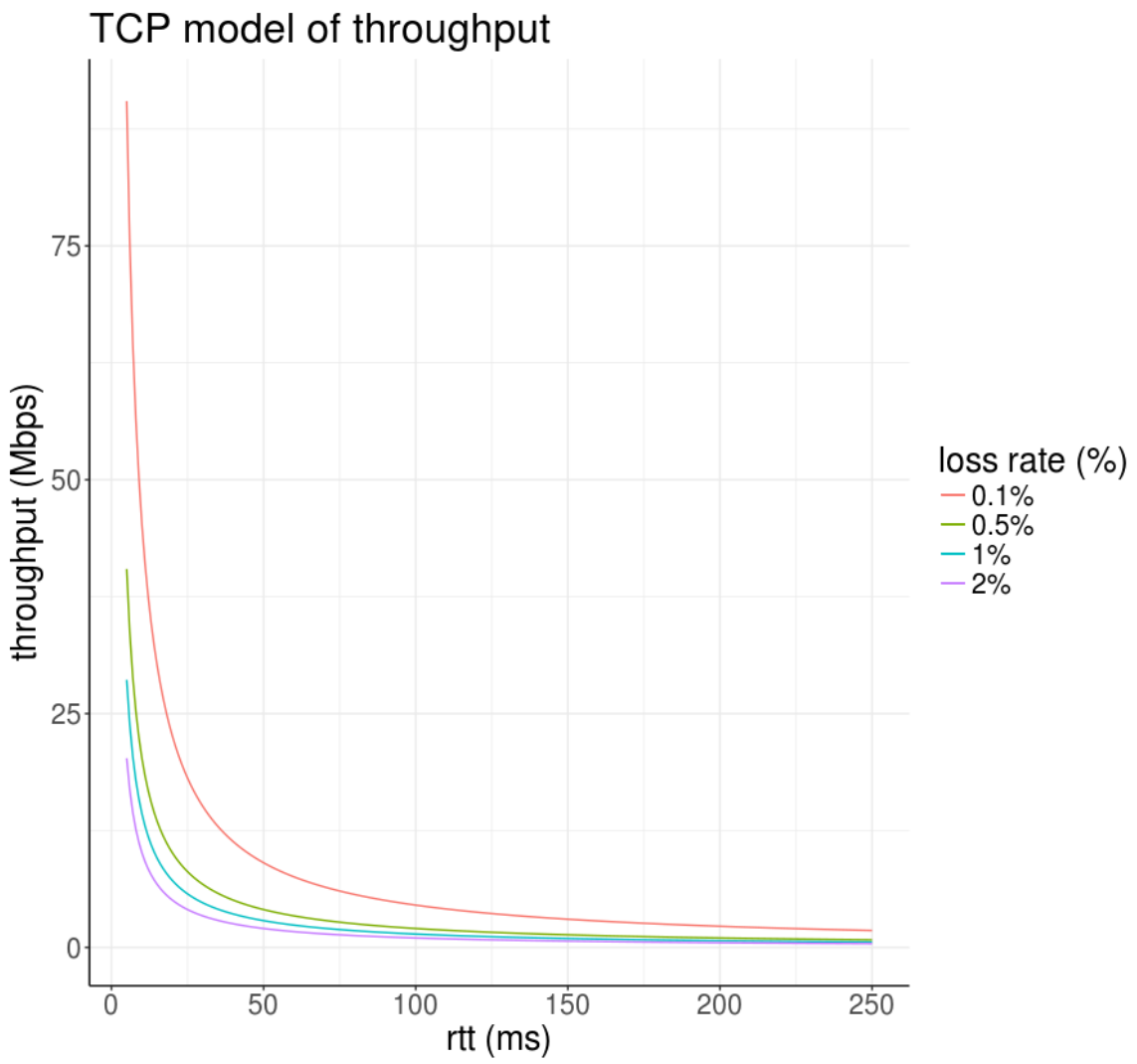

Figure 2-2: Theoretical throughputs given round trip time and loss rate.

plot in Figure 2-3 shows the theoretical results (solid lines) along with the observed throughputs when running iperf, which is one BTC measurement tool. The empirical results follow the theoretical throughputs fairly closely in shape.

2.4.2 Test traffic on high-capacity links

The typical BTC measurement methods described essentially flood the network, intentionally reaching the point of congestion, to get a measurement of the maximum throughput the link is able to achieve. Especially as network capacities increase in the future and network testing becomes more important and therefore likely more routine

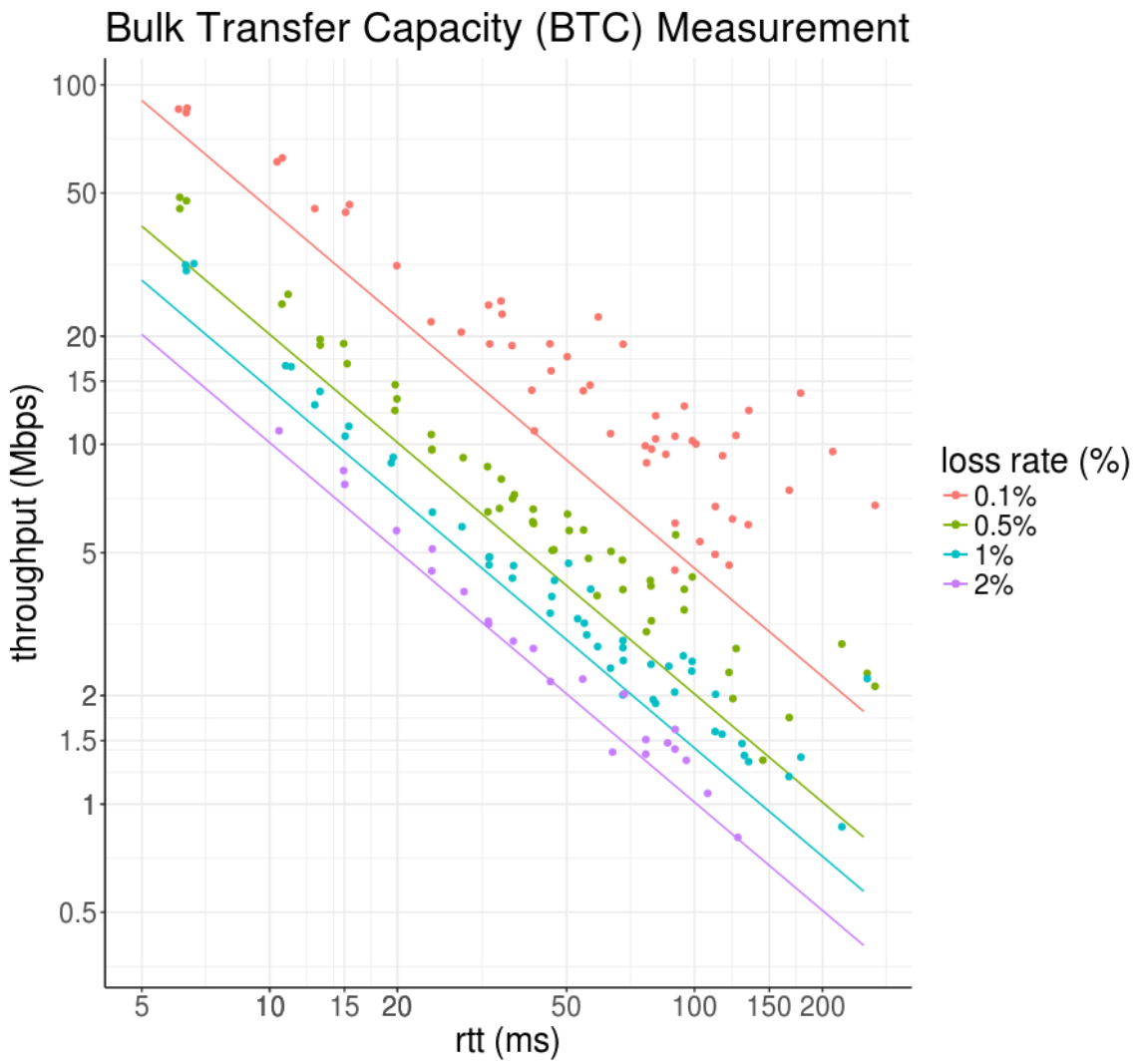

Figure 2-3: Log-log plot of theoretical and empirical throughputs (measured using iperf).

and frequent, measuring the maximum throughput of the link might be unfavorable for several reasons. More frequent flooding of a network and hitting resource limitations adversely affects the experience of other users using that network. Additionally, with higher-capacity links, filling the link and then testing it will take longer.

Perhaps most significantly, such a *maximum* throughput measurement may be unnecessary for evaluating a network for most users' throughput needs. If a typical use case is the streaming of a typical video file, one can more easily test a user's quality of experience with a test stream sent at the throughput of a typical video's bit rate, which is significantly smaller than the size of a gigabit link, for example. Such a test stream would not unnecessarily congest a network, not take a long time (to ramp up, or to show that it can maintain the target throughput), and be better for more routine network testing and monitoring purposes. Maximum throughput tests will still have their purpose in measuring a link's maximum BTC, but we expect that a typical end user will not require this measurement for most daily situations, and therefore, networks should be tested accordingly.

2.4.3 Test configurability

As network connectivity becomes more essential in people's lives, the regulation of network performance, and with that the measurement of network performance, will become all the more important. Designing network measurement for the future requires both specifying measurements that are able to be compared, as well as building test frameworks that allow for configurable and routine testing.

This thesis describes and evaluates end-to-end BTC measurements that, informed by models of TCP, exhibit vantage point independence, and can be compared and analyzed with other measurements taken under similar conditions. Such measurements can be made piecewise, and will not be dependent on the specific topology and round trip times of the test servers.

In addition, this thesis implements a test framework that allows one to plug in the appropriate models, configurations, and goals, for one's particular use case. Many current BTC measurement methods do not allow for the specification of different test

traffic patterns that allow for the measurement of other characteristics of interest to network performance, such as regarding slowstart behavior, onset of queuing, and ability to support packet transmission burstiness. Because end-to-end performance measurement is contextual and is based on specific use cases, the configurability of network testing networks will be essential for producing meaningful results.

2.5 Model Based Metrics IETF Internet-Draft

To improve upon existing BTC measurement, prepare for a near future of higher-capacity links, and design for the goal of ubiquitous and meaningful network measurement, we build upon and evaluate the work of Mathis's Model Based Metrics (MBM), as outlined in the IETF Internet-Draft [14]. We implement the MBM test framework as well as two tests that the framework can run to measure BTC.

The basic idea of the test framework is, given a set of target performance inputs (such as target throughput, target round trip time) and a model of expected network behavior (such as a model of TCP), to create and send test streams across the link(s) of interest, and to evaluate the link's performance based on the observed measurements, target performance inputs, and assumed network models.

In Chapter 3 we describe the system design in more detail, and in particular focus on its implementation for the purposes of BTC measurement, using a model of TCP Reno for our calculations. We cover two possible tests one might have in their measurement suite: the Basic Data Rate Test and the Sustained Burst Test. The Basic Data Rate Test is mainly concerned with the measurements of throughput, round trip time and loss rate. The Sustained Burst Test additionally tests a network's ability to withstand packet bursts.

Chapter 3

Model-Based TCP Throughput Measurement

This thesis implements and evaluates a network performance measurement test framework called Model-Based Metrics (MBM), originally outlined in the IETF Internet-Draft [14], which incorporates the expectations and models of the specific network it is testing. The test framework constructs test streams and evaluates links based on models of the network as well as target performance inputs. The general system design, described in Section 3.1, can be used for a variety of networks and measurement specifications. We focus on the implementation and evaluation of two tests that measure bulk transfer capacity on links configured with TCP Reno: the Basic Data Rate Test and the Sustained Burst Test.

To motivate this design, we explain how incorporating one's expectations of the performance of a network allows us to define measurements of links that are independent of their vantage points in Section 3.2. To build upon these key ideas regarding the decoupling throughput and round trip time, we then describe and analyze Mathis's proposed loss measurement in Section 3.3, in which we also introduce and apply the concept of measurement spatial composition.

In addition to defining a loss measurement, we incorporate a hypothesis test Sequential Probability Ratio Test to statistically evaluate the loss measurement of a link, as well as to further reduce the amount of unnecessary traffic sent during a test,

described in Section 3.4.

3.1 Test Framework Design

The general test framework produces suites of performance tests, each test of which incorporates expectations and models specific to the network being tested. One can imagine that each test suite is designed for and applied to a particular kind of network, given knowledge about its behavior, configuration, expected use cases, and expected performance. For example, depending on the specific network and user needs, one may be interested in designing a variety of goals and tests regarding a network's ability to support certain slow start behavior, certain burstiness, and/or certain loss rates.

A test suite that follows this framework (Figure 3-1) consists of

1. a set of target performance inputs, which are used with
2. the chosen network models for
3. the traffic generator to construct test traffic for each of the tests in the suite, which
4. the delivery evaluator will collect measurements from to determine each test's outcome.

A path passes if each of the tests in its test suite passes.

The two tests that we've implemented for our evaluation are the Basic Data Rate Test and the Sustained Burst Test; specific test design motivations and implementation details are in Section 3.6 and Table 3.2.

3.2 Key Ideas

To achieve measurements that are independent from their vantage point, we send test traffic across a given link at a specific throughput, then determine whether at

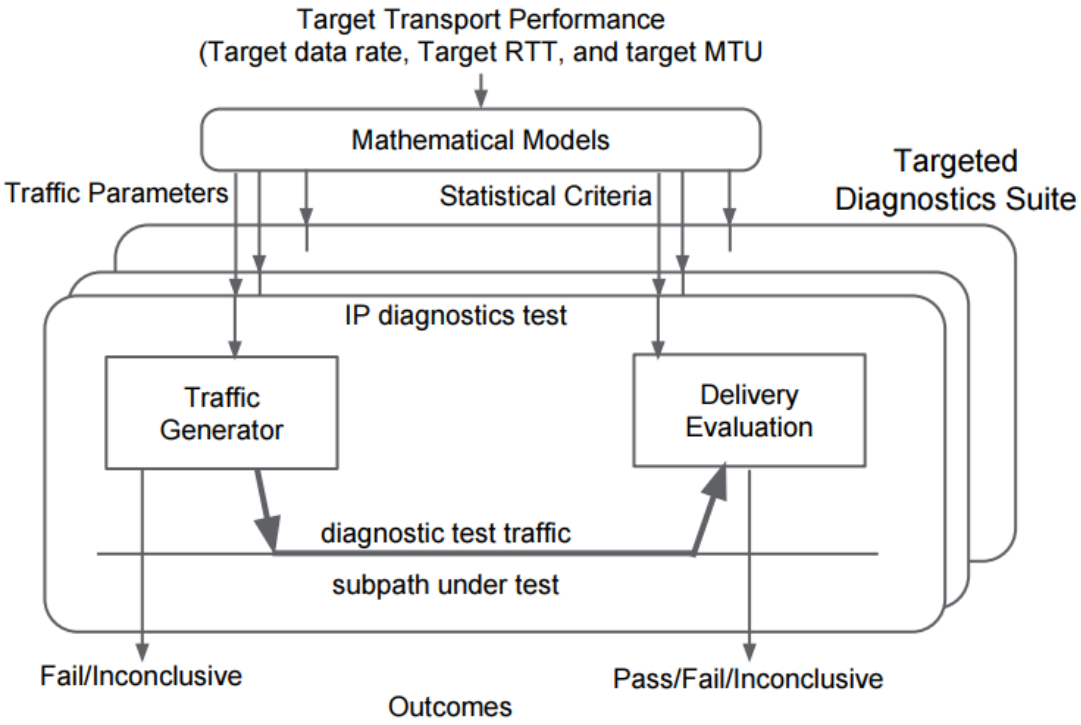

Figure 3-1: MBM test framework design, image from IETF IPPM Working Group [6].

the observed loss rate and at the end-to-end round trip time we want the link to be able to support, the link is able to achieve the specified test traffic throughput. We can consider the link's performance for an arbitrary target end-to-end round trip time, by determining whether the loss rate is low enough.

Recalling the throughput curves for TCP Reno in Figure 2-3, in which the capacity of the link is 100Mbps, we illustrate the link's ability to support a 10Mbps constant bit rate TCP stream under different round trip time and loss rate conditions (Figure 3-2) with data collected in a virtual network. This rate-limited constant bit rate stream does not rely on TCP congestion control and its congestion signals (packet drops) to determine its throughput, as long as the link is able to support the stream. If the link is unable to support the test stream, then TCP congestion control will decrease the send rate until it no longer detects congestion. Loss rate configurations in the virtual network simulate the aggregate random packet loss caused by other concurrent traffic on the link. Then for a given stream's target throughput and measured loss rate, we can determine the range of round trip times (as in, end-to-end round trip times) that we can expect that link to be able to support.

Examining the results in Figure 3-2, at a loss rate of 0.5%, for example, we see that the link is able to support 10Mbps for round trip times to about 12ms, after which congestion overtakes, and we are not able to hit 10Mbps, following the theoretical curve fairly well. For example, if our target end-to-end round trip time is 30ms, and we estimate the link loss rate to be 0.5%, we would expect the link to not be able to support the 10Mbps test stream.

One can draw a horizontal line at any desired target throughput, and by following a particular loss rate's curve, determine the range of round trip times that would be able to support that target throughput. In addition to sending constant bit rate streams that meet certain criteria (such as target throughput), one could also send different test streams with other traffic patterns, and do similar calculations and evaluations.

A closer examination of the results and the relationship between the theoretical and experimental results include two observations: 1) the drop-off in the experimental

bit rates from 10Mbps before theoretical maximum round trip time (marked by the intersection of the constant 10Mbps line and theoretical throughput lines), and 2) the increased deviation from the theoretical line as round trip time increases and throughput goes to zero.

The first observation of the throughput drop-off may be explained by the gradually increasing effect of congestion: as packet drops take longer to transmit to the sender, the sender takes longer to reduce the window size, causing increasing build up on the bottleneck queue. This build up will cause the resulting throughput to decrease *before* the network actually detects congestion and drops packets, which occurs when the buffers actually fill completely.

The second observation of deviation from theoretical results at high round trip times (as well as low round trip times, if one refers back to Figure 2-3 with the non-rate-limited iperf results) may be best explained to be limitations of the model. The model assumes that loss is nonzero, and by the model, throughput approaches infinity as round trip time and/or loss approach zero, and throughput approaches zero as round trip time and/or loss approaches infinity. It is reasonable to expect empirical results to diverge slightly especially at the lower and upper ranges of variable values.

As is in any case that models are used, due to discrepancies between theoretical models and experimental results, the tester should assess the fit of the model before basing performance evaluations upon them. We make these observations about this particular TCP Reno model and use this model in our calculations and implementation of our test framework suite with these considerations in mind.

3.3 Calculating Acceptable Loss Rate

As we send our test stream through a link, we need to measure the link's loss rate, which we then compare to a theoretical cutoff loss rate, the goal being, if the measured loss rate is less than the theoretical cutoff loss rate, then we can expect the link to support the test stream under the target conditions (such as target end-to-end round trip time). This theoretical cutoff loss rate is calculated using the target

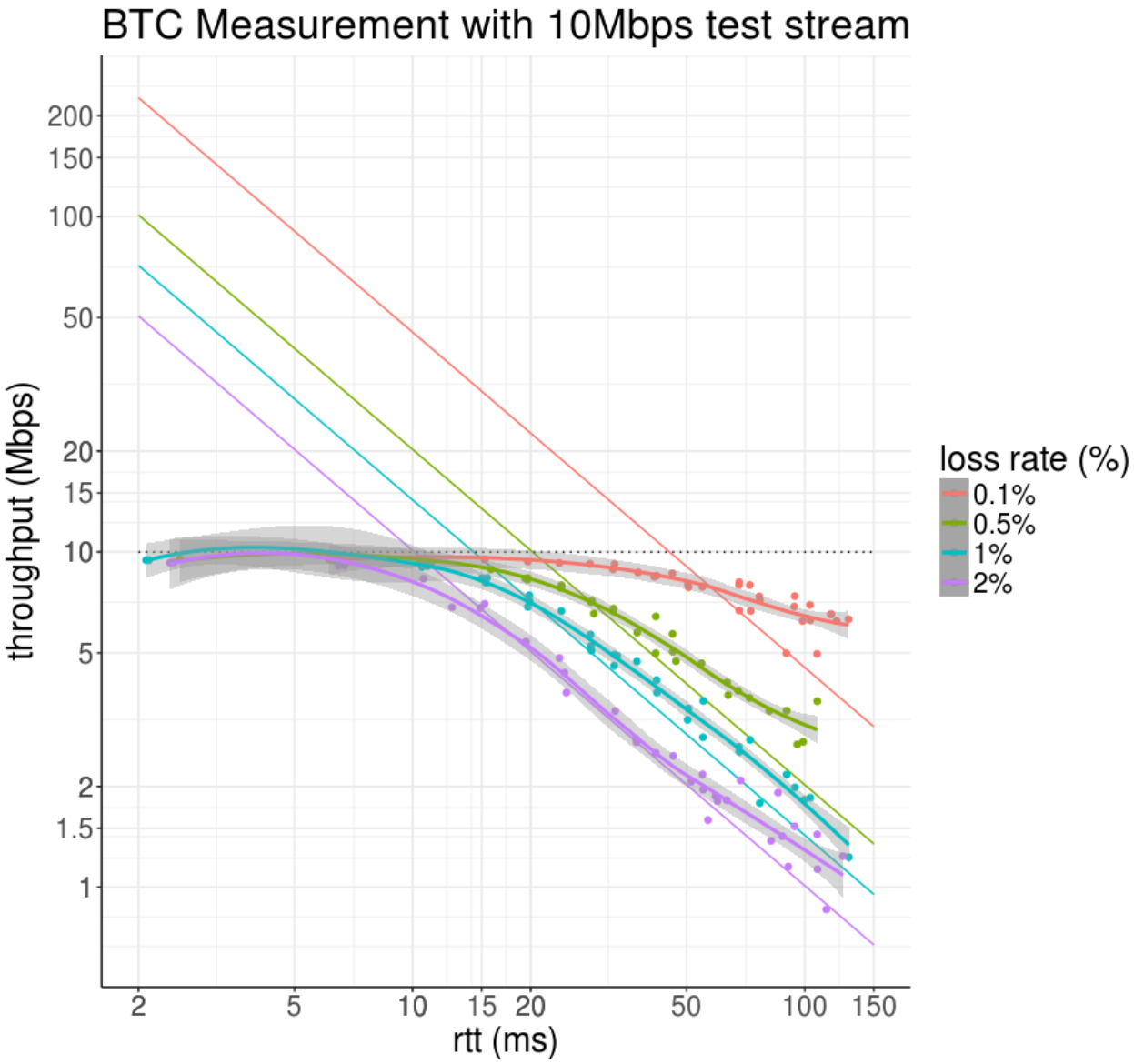

Figure 3-2: The key idea of this model-based measurement method is to send traffic at a determined target performance, for example at a target throughput (in this graph, 10Mbps) that is much lower than the link capacity (100Mbps), then to determine whether at the observed loss rate and at the end-to-end round trip time we want the link to be able to support, the link is able to achieve the specified test traffic throughput. In this graph, the test stream's target throughput is 10Mbps; if the observed loss rate is 0.5%, then the path will be determined to be able to support the target throughput for target expected end-to-end round trip times (rtt) that are less than 20ms).

throughput `target_throughput` (which is also the throughput of the test traffic); the expected end-to-end round trip time `target_rtt`; and the expect MTU (maximum transmission unit, or the expected packet size in bytes)) `target_MTU`.

Mathis's derivation of the maximum acceptable loss rate at which a path will still be able to support the target performance values begins with a calculation of the expected, as in target, window size W based off these target performance values:

$$W = \frac{\texttt{target\_throughput} \times \texttt{target\_rtt}}{\texttt{target\_MTU}} \tag{3.1}$$

From this, Mathis estimates the target run length R, which is the minimum number of packets that must be transmitted between packet losses in order to maintain target performance. Recalling the sawtooth window size behavior of TCP, Mathis reasons that to fill the pipe, "losses must be no closer than when the peak of the AIMD sawtooth reached exactly $2W$ otherwise the multiplicative window reduction triggered by the loss would cause the network to be underfilled" [14]. Calculating the area under the sawtooth, we get:

$$R = (\frac{3}{2}W)(2W) = 3W^2 \tag{3.2}$$

In other words, the measured loss rate p must be at most the cutoff loss rate p_0 defined as:

$$p_0 = \frac{1}{R} \tag{3.3}$$

Defining this loss cutoff allows us to determine whether a path can support a given target performance (throughput, round trip time, etc.), by providing a theoretically-backed cut-off for acceptable throughputs. Mathis explains that this is a conservative estimation for acceptable loss rates, based on some assumptions about queue behavior, such as that any window size greater than the calculated target window size W contributes to a standing queue that affects the delivery time of ACKs (loss signals). We further consider the implications of more or less conservative specifications in 5.3.

In addition, Mathis explains that loss measurements can be defined to be compos-

able. Measurements are composable if one can use measurements from subsequent subpath links to make calculations about the complete path measurement [3]. With link loss measurements specifically, consider a configuration of three neighboring links a, b, and c (as illustrated in Figure 4-6). Given the sufficiently independent loss rates of two subsequent subpaths p_{ab} and p_{bc}, the expected combined loss rate is defined by IETF RFC 6049 to be

$$p_{ac} = 1 - (1 - p_{ab})(1 - p_{bc}) \qquad [3] \tag{3.4}$$

From the composed loss measurement, we can similarly evaluate whether or not we can expect the total path to support a given target performance. How Mathis imagines this to be implemented is that a path's total acceptable loss is specified (possibly by using the target run length derivations), and then each subpath is given a loss budget that is a fraction of the total acceptable loss, which is used as that subpath's cutoff loss rate when statistically evaluating its loss rate (to be described in Section 3.4).

Subpath loss measurements are composable if loss measurements are determined be sufficiently independent. This assumption of independence is empirically tested in Section 4.4 and discussed in Section 5.2. Composable measurements allow for routine, modular, and asynchronous testing of networks, from which we can determine end-to-end performance evaluations.

3.4 Statistical Evaluation of Loss Rate

In the previous section, we explain how to calculate, from target performance inputs, the maximum acceptable loss rate p_0 at which a path will still be able to support the target performance (Equation 3.3). To determine whether or not a path's loss rate p is at most p_0, we can introduce a statistical hypothesis test to make that statistical estimation. In particular, we will use the Sequential Probability Ratio Test (SPRT) [17], which has an additional optimization of ending the collection of new samples once it can accept either the null or alternate hypothesis.

This usage of SPRT, as originally specified in the MBM IETF Internet-Draft [14], tries to determine whether the value of p, the loss probability of a link, is low enough (in which case the link can support the target performance) or too high (in which case the link cannot support the target performance). To contribute to the explanation of the SPRT setup in the MBM IETF Internet-Draft [14], we elaborate on some of the derivation details.

The measured loss rate p of the link is statistically determined to be low enough if $p < p_0$, where p_0 is defined in Equation 3.3. On the other hand, p is determined to be too high if $p > p_1$, where the MBM IETF Internet-Draft defines $p_1 = 4p_0$, "based on analysis of typical values and practical limits on measurement duration" [14]. The underlying intuition is that if p is determined to be too high, then the link would not be able to sustain test traffic sent at the target throughput given the other target performance conditions, because the high frequency of packet losses, and therefore the high frequency of congestion signals, would not allow the sender's window size to remain large enough. This allows us to specify the following pair of hypotheses for SPRT:

$$H_0 : p \leq p_0 = \frac{1}{R}$$
$$H_1 : p \geq p_1 = 4 * p_0 = \frac{1}{4R}$$

The MBM IETF Internet-Draft assumes that each transmitted packet follows the Bernoulli distribution, such that it is lost with probability p, and transmitted successfully with probability $1 - p$. This requires the assumption that packet loss probabilities are independent and identically distributed, which we further discuss in Section 5.2.

SPRT focuses on the count of lost packets m (out of a total of n sent packets). It specifies a lower boundary L and an upper boundary U, such that for every additional packet collected, we continue testing until m crosses either the lower or upper boundary. If $m <= L$, then we accept the null hypothesis; if $n >= U$, then we accept

the alternate hypothesis; else, we continue testing. L and U are based on the specified Type I and II error; we use Type I error = $\alpha = 0.05$, and Type II error = $\beta = 0.05$.

3.5 Test Evaluation

In addition to the process of statistical estimation of loss rate, which helps determine whether or not a path is able to support traffic at a certain throughput and round trip time, MBM test evaluation also includes a traffic delivery evaluation. The test traffic sent over a link must satisfy some traffic requirements before the statistical evaluation of loss rate can be utilized.

For example, if one is attempting to send a stream with a bursty traffic pattern, if the observed test stream does not actually exhibit the desired traffic pattern, then the statistical evaluation of loss rate will not be valid. In another particular case, if one is attempting to send a traffic stream at a throughput that is too high for the link to actually handle, then that traffic stream will never have been successfully sent over the link. In this situation, the statistical evaluation of loss rate is not only invalid, but also unnecessary, since one can already determine that the path is unable to support the target performance.

During the test, we differentiate between a growth period and a test period. During the growth period, the test traffic attempts to achieve some traffic preconditions; only when the test traffic is as expected does the test period begin, at which point statistical evaluations can be run. In the cases of the Basic Data Rate Test and Sustained Burst Test that are implemented in this thesis, their test periods focus on measuring what traffic a link can sustain in the long run, so we allow the sender to "warm up" the sending rate during the growth period before entering the test period. If instead a test aims to evaluate a link's ability to, for example, handle slow start, the criteria for entering the test period should reflect that goal. In summary, before undergoing any statistical estimation, the test stream must be confirmed to meet some test preconditions. Most relevant to our tests is the precondition of the test stream throughput meeting the target throughput.

Table 3.1: Test framework results descriptions

Result	Description
PASS	Traffic preconditions are met and the statistical test accepts H_0 with 95% confidence level, where $H_0 : p \leq p_0 = \frac{1}{R}$. With this observed loss rate p, we believe the link to be able to support the test traffic for the given target performance.
FAIL	Traffic preconditions are met* and the statistical test accepts H_1 with 95% confidence level, where $H_1 : p \geq p_1 = \frac{1}{4R}$. With this observed loss rate p, we believe the link to NOT be able to support the test traffic for the given target performance. *In cases that the traffic preconditions are NOT met because observed traffic throughput $< target\_throughput$, we make the assumption that the reason is because the link cannot support that level of traffic. So, for some particular traffic preconditions, the test returns FAIL if those conditions are not met.
INCONCLUSIVE	Traffic preconditions are met but the test timed out. The statistical test is not able to accept either H_0 or H_1 because not enough samples have been collected for the lost packet count m to cross either the L' or U' boundaries.
ERROR	Traffic preconditions are not met. (Statistical test has no bearing on result in this case.)

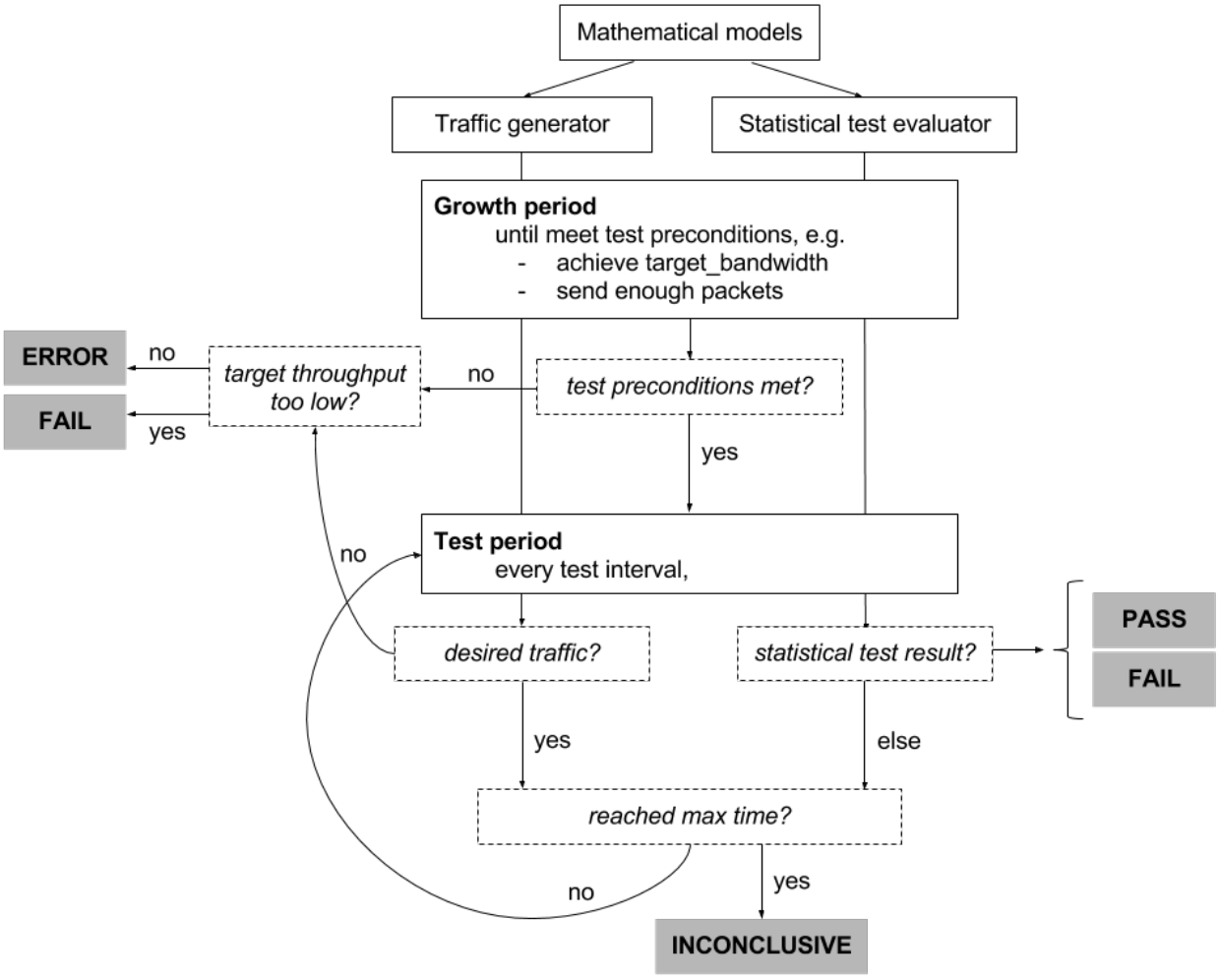

Figure 3-3: Traffic delivery and loss rate evaluations.

The possible results that the test framework can return are described in Table 3.1, with the descriptions of those possible results being specific to our two bulk transfer capacity tests.

To explain the entire test process (Figure 3-3), we begin with a growth period. If the test traffic fails to meet test preconditions for the entire test period, we will either return FAIL or ERROR (refer to Figure 3-3 for details). Else, once test preconditions are met, we enter the test period. For every test interval (which can be as frequent as every packet transmission, according to SPRT, but practically speaking may be less frequent than that), we first evaluate that the test traffic preconditions are met. If they aren't we return FAIL or ERROR (refer to Figure 3-3 for details). Else, we can run the statistical test, which given the number of lost packets m and total packets sent n, will determine whether or not we have crossed either the lower or upper boundaries to accept H_0 (PASS result) or H_1 (FAIL result), respectively. If the test times out before returning a PASS, FAIL, or ERROR result, the test will return INCONCLUSIVE.

If one weren't using SPRT, then the statistical evaluation can also occur after the test stream delivery and data collection are complete, given a pre-calculated minimum amount of data that needs to be collected before the test can reach a statistically significant conclusion.

3.6 Framework Implementation

We implemented the MBM test framework along with the Basic Data Rate and Sustained Burst tests in C++. We used `tcp_info` from the Linux 4.10 Kernel to collect data about the TCP connection. `tcp_info` stores parameters on a network socket's live TCP connections [16]. The framework consists of a server and a client. Once a client connects to a server, the server will start a test connection with the client, on which test traffic is delivered. The server will use data from `tcp_info` to measure and statistically evaluate the loss rate.

In our implementation, we define a lost packet to be a retransmitted packet (which

Table 3.2: Implemented test suite descriptions

Test name	Basic Data Rate Test	Sustained Burst Test
Description/ Goal	Test how well the network can handle the basic test of a constant bit rate stream at some `target_throughput`.	Test how well the network can handle `target_window_size` bursts every `target_rtt`.
Target performance inputs	`target_throughput` (kbps) `target_rtt` (ms) `target_MTU` (B)	`target_throughput` (kbps) `target_rtt` (ms) `target_MTU` (B) `derate`
Test stream generation	Constant bit rate stream at `target_throughput`.	$\frac{\texttt{target\_throughput} \times \texttt{target\_rtt}}{\texttt{target\_MTU}} \times \texttt{derate}$ bursts of packets every $\texttt{target\_rtt} \times \texttt{derate}$.
Test preconditions	Test traffic is maintaining `target_throughput`.	Test traffic is sending bursts as expected, maintaining on average `target_throughput`.
Delivery evaluation	Statistically estimate whether or not path can support the test stream by determining whether loss rate $< \frac{1}{\texttt{target\_run\_length}}$.	Statistically estimate whether or not path can support the test stream by determining whether loss rate $< \frac{1}{\texttt{target\_run\_length}}$.

`tcp_info` keeps count), and a successful packet transmission to be MTU amount of bytes that have been ACKed. Section 5.2 considers the implications of the choice of definition of loss. We set TCP Reno as the congestion control algorithm. For the traffic generator, we create constant bit rate streams using Linux's Fair Queue packet scheduler, and we create bursty test traffic using application-level rate limiting.

The two tests that we've implemented for our evaluation are the Basic Data Rate Test and the Sustained Burst Test (Table 3.2).

3.6.1 Test suite notes

The Basic Data Rate Test sends a constant bit rate stream at `target_throughput`. Especially as links increase in capacity and links require more frequent, routine measurements, one may choose to measure a link with a `target_throughput` based on what the link is expected to support for one user's connection, which especially as

link capacities increase, is expected to be much less than maximum link capacities, as described in Section 2.4.2.

The Sustained Burst Test takes in the same target performance inputs as the Basic Data Rate Test, along with a `derate` parameter, to construct a bursty stream. The worst case that a link is expected to support is a stream with bursts of size $\frac{\texttt{target\_throughput} \times \texttt{target\_rtt}}{\texttt{target\_MTU}}$ (the target window size) every `target_rtt` [14]. Because this test is considered to be a strenuous test, one can construct a less strenuous test stream by sending smaller bursts more frequently using the `derate` parameter, a positive fraction less than or equal to 1.

Chapter 4

Framework Evaluation

In this chapter, we empirically evaluate the tests in a virtual network setup. We determine how the Basic Data Rate Test can estimate the performance of an interconnection link, a motivating example introduced in Section 2.4.1 and Figure 2-1. Second, we also compare the Sustained Burst Test against the Basic Data Rate Test to observe how it evaluates the effect of burst sizes and queue sizes on network performance. Third, we empirically evaluate the process of loss rate measurement composition along subpaths, which would facilitates modular network testing.

4.1 Evaluation Methodology

We ran the framework on Mininet, a software network emulation system that allows for the configuration and repeatable running of hosts, switches, routers, and links on one Linux kernel. Links can be configured with a bandwidth, one-way propagation delay, loss rate, and maximum queue size. The configured loss rate is essentially the aggregate random background loss on the link, to simulate the loss that would be caused by other traffic on the link.

Given the multitude of parameters, we attempt to simplify our evaluations to focus on particular effects, and motivate our parameter value choices with expected real-life scenarios. We set link bandwidths to be 100Mbps, and `target_MTU` to be 1500 B. An interesting future study would be to determine the link and test parameter ranges

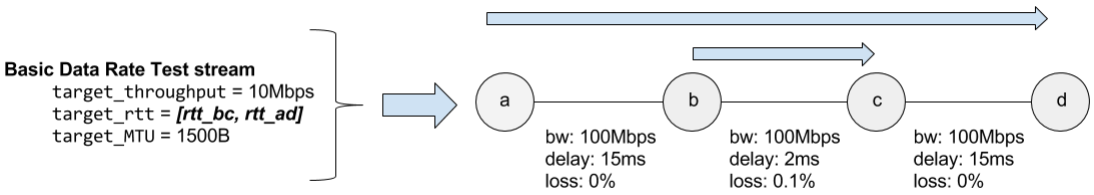

Figure 4-1: Setup for evaluating Basic Data Rate Test on the interconnection link example. We vary the test link's `target_rtt` (bolded and italicized) and observe the test results.

that the link can be tested using MBM to support.

4.2 Testing Interconnection Link Example with Basic Data Rate Test

4.2.1 Test configuration

We evaluate the Basic Data Rate Test on a network that resembles the interconnection link example in Section 2.4.1 and Figure 2-1, which illustrates the effects of round trip time on throughput. We set up four hosts a, b, c, and d, where $link_{bc}$ is the bottleneck interconnection link (Figure 4-1). Hosts a and d are far apart, and represent a typical pair of far-apart nodes that may want to communicate with each other (imagine them approximately a continent apart). We vary the link's `target_rtt` and observe the test results; specifically, we determine whether running the Basic Data Rate Test on the interconnection link $link_{bc}$ with `target_rtt` $= rtt_{ad}$, will accurately reflect the test results for $link_{ad}$ with `target_rtt` $= rtt_{ad}$.

4.2.2 Results

Running the Basic Data Rate Test on $link_{bc}$ with a `target_rtt` $= rtt_{bc}$, on $link_{bc}$ with a `target_rtt` $= rtt_{ad}$, and on $link_{ad}$ with a `target_rtt` $= rtt_{ad}$ give the results PASS, FAIL and FAIL, respectively, very consistently. To better understand how these results come about, we go into the two stages of test evaluation: test traffic

delivery evaluation and loss rate statistical evaluation (as described in Figure 3-3).

With a `target_throughput` of 10Mbps, we determine that the test traffic does not meet the required preconditions if the `target_throughput` of the test traffic does not achieve at least $0.9 \times$ `target_throughput` = 9Mbps. This is a simple cutoff that can be more refined in the future. We see that the test traffic on $link_{bc}$ achieves the cutoff throughput of 9Mbps, but the test traffic on $link_{ad}$ does not (Figure 4-2). This is why the test on $link_{ad}$ fails.

Moving on to the observed loss rates, we observe that all three test cases exhibit about the same loss rate. It should be noted though that the observed loss rate on $link_{ad}$ was not used for the test evaluation (Figure 4-3): because the test traffic never satisfied the throughput preconditions, the test never finished the growth period to enter the test period, during which statistical evaluation of loss rate occurs.

Focusing on the tests on $link_{bc}$, though both tests are on the same link and the observed loss rates are about the same, the target loss rate cutoffs (as defined in Equation 3.3 to be the inverse of the target run length, and marked as p_0 in Figure 4-3) used in the two tests' null hypotheses differ. So the test statistically estimates the loss rate of $link_{bc}$ to be significantly less than the p_0 calculated with `target_rtt` = rtt_{bc}, but not significantly less than the p_0 calculated with `target_rtt` = rtt_{ad}. In other words, the test on $link_{bc}$ does not pass if one is interested such a `target_rtt` that is equal to rtt_{ad}; this relatively larger target end-to-end round trip trip (when compared to rtt_{bc}) is expected to require a much smaller loss rate if it is to be able to support the target performance values.

4.2.3 Discussion

This test setup highlights how the target performance inputs are utilized in the test evaluation. Specifically, for example, regardless of the actual round trip time of a link, the target round trip time can be set to evaluate the performance of the link according to the target performance inputs, thus achieving vantage point independence.

Both the tests on $link_{ad}$ and $link_{bc}$ with `target_rtt` = rtt_{ad} give the result FAIL. The test fails on $link_{ad}$ because it is not able to achieve the target throughput, due

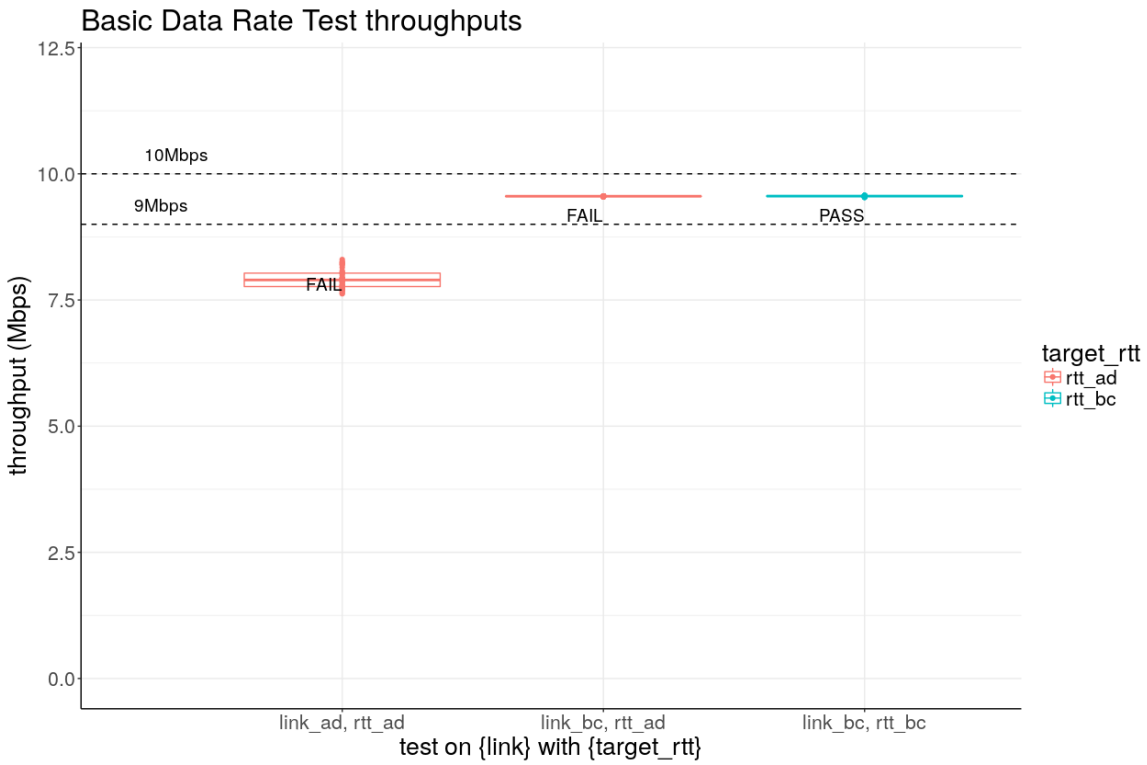

Figure 4-2: Throughput results for evaluation of Basic Data Rate Test. The test traffic throughputs meet the target throughput of 10Mbps for the two test cases on $link_{bc}$.

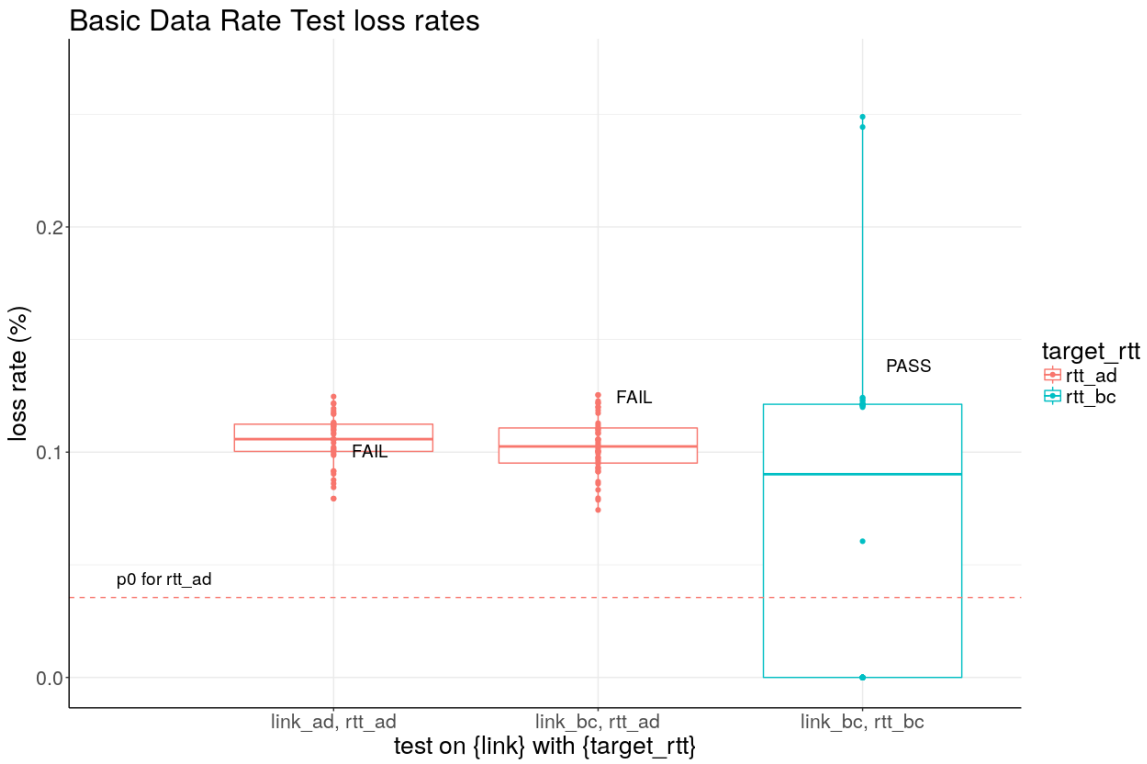

Figure 4-3: Loss results for evaluation of Basic Data Rate Test. The loss rate on $link_bc$ is significantly lower than the target loss p_0 that is calculated with `target_rtt` $= rtt_{bc}$. The loss rates for the other two test cases are not significantly lower than the target loss p_0 calculated with `target_rtt` $= rtt_{ad}$.

to the expected effects of the longer round trip time on the sender's detection of loss, as described in Mathis's model [4]. The test fails on $link_{bc}$ because the observed loss rate is estimated to be too high to support the test traffic at `target_throughput` on paths of length `target_rtt`. If the test traffic on $link_{ad}$ had been generated as expected (in other words, met the target throughput), then the observed loss rate would be compared to the same target loss rate as that for the test on $link_{bc}$.

It is interesting to note the high variance in loss rates for the test over $link_{bc}$ with `target_rtt` $= rtt_{bc}$, even though the test streams sent over $link_{bc}$ are the same regardless of what the `target_rtt` is set to. This is explained by the fact that the SPRT test, when `target_rtt` $= rtt_{bc}$, can end sooner, since determining that the loss rate is significantly less than or equal to a relatively high target loss requires fewer packets. When the number of packets observed is smaller, it is reasonable to expect that observed loss rates will exhibit higher variance; in our case, on average almost 100 times fewer packets were sent before the test returns PASS when `target_rtt` $= rtt_{bc}$, than before the test returns FAIL when `target_rtt` $= rtt_{ad}$.

4.3 Testing Queue Sizes with Sustained Burst Test

4.3.1 Test configuration

We set up two hosts a and b (Figure 4-4), through which we run the Sustained Burst Test and Basic Data Rate Test with the same target performance inputs, to observe the effect of test traffic burst size on loss rate. We vary the test's `derate`, which varies the burst size and time between bursts, and observe the resulting loss rates and test results. We compare the Sustained Burst Test results with results from the Basic Data Rate Test.

4.3.2 Results

Running the Basic Data Rate Test on this link gives an average loss rate of 0%, which gives the result PASS. This is a basis of comparison for the Sustained Burst Test,

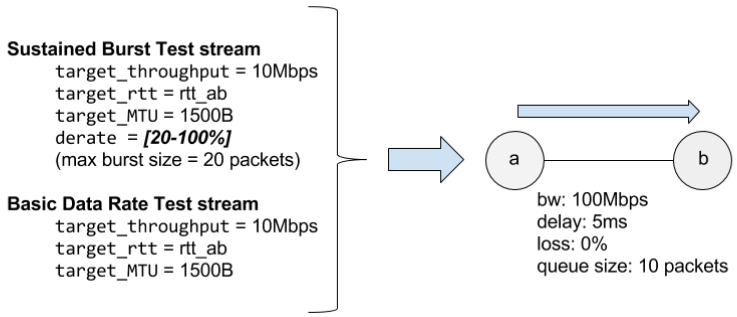

Figure 4-4: Setup for evaluating Sustained Burst Test with varying `derate` inputs (bolded and italicized). We observe the resulting loss rates and test results, and also compare the Sustained Burst Test results with results from the Basic Data Rate Test.

which we run multiple times while varying the `derate` variable from 0.4 to 1.0. Recall that the Sustained Burst Test constructs a test stream with $\frac{\texttt{target\_throughput} \times \texttt{target\_rtt}}{\texttt{target\_MTU}} \times$ `derate` -sized packet bursts every `target_rtt` $\times$ `derate` seconds.

Figure 4-5 displays the loss rates for the Basic Data Rate Test and all of the Sustained Burst Tests with varying `derate` inputs. The plotted p_0 and p_1 are the loss rate cutoffs as defined in the statistical test hypotheses, given the target performance inputs (specified in Figure 4-4). When `derate` $= 1.0$, the Sustained Burst Test generates the most strenuous test that a path is expected to be able to support. We see high variation of observed loss rate for when `derate` $= 1.0$. Loss rates decrease to almost 0.0% by `derate` $= 0.6$, and the test passes.

4.3.3 Discussion

The test setup seeks to determine the effects of derating in the Sustained Burst Test, which decreases the size of bursts and increases the frequency of bursts. Note that the Sustained Burst Test always generates a test stream that has an long-term average throughput of `target_throughput`, and theoretically, as `derate` approaches 0, the test stream will approximate a constant bit rate stream, like one sent by the Basic Data Rate Test.

That the Basic Data Rate test passes with loss rate 0%, while the Sustained Burst Test at highest expected intensity (`derate` $= 1.0$) mostly fails demonstrates

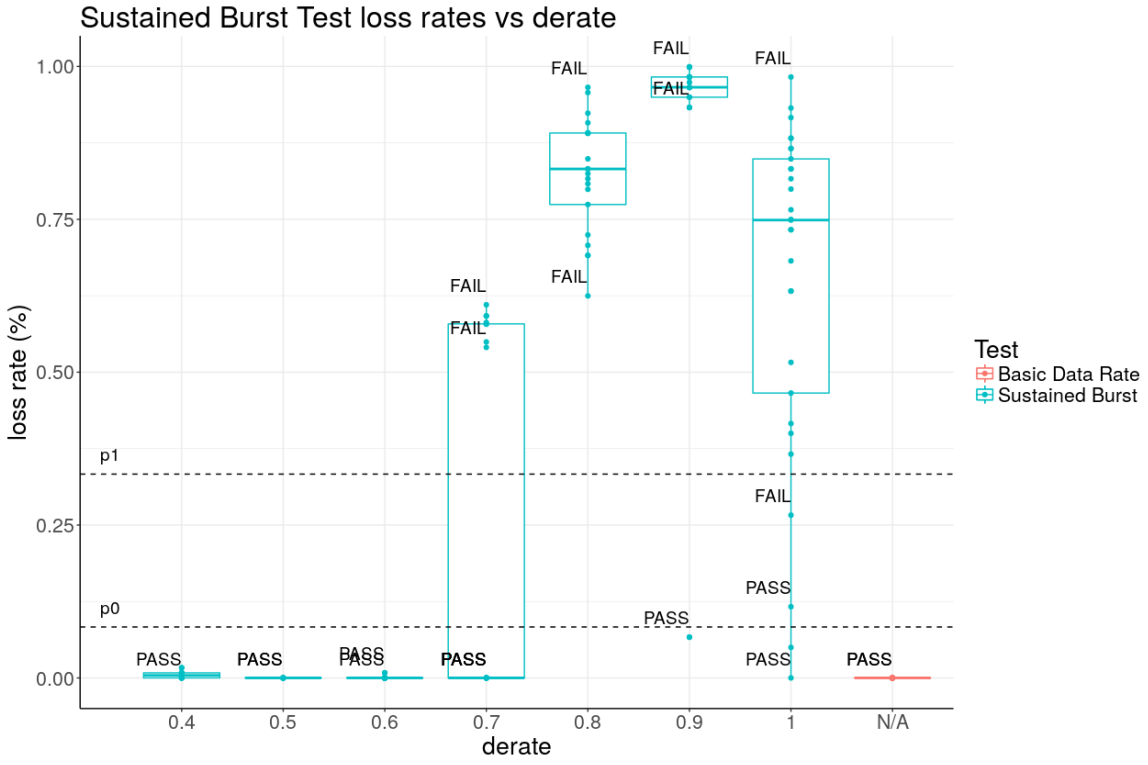

Figure 4-5: Loss results for evaluation of Sustained Burst Test. For the Sustained Burst Test, loss rates are too high (at least p_1) for high `derate` inputs (> 0.7), where p_0 and p_1 are loss rate cutoffs as defined in the statistical test hypotheses. In comparison, the Basic Data Rate Test passes on this link with average 0% loss (right-most boxplot).

the importance of testing link bulk transfer capacity with streams other than constant bit rate streams. Buffer sizes affect network performance, so network performance tests should test the buffer sizes. A constant bit rate stream will not be able to sufficiently determine how well a path's buffers will be able to handle bursts of traffic while maintaining the target throughput.

One may remark that the queue size that the link was configured to (10 packets, as is specified in Figure 4-4) seems smaller than a realistic queue size. This queue size was chosen in order to demonstrate the interesting effects of derating on link loss rate and test results, given the configuration of the link. The minimum required queue size that would support the target performance is expected to increase as link round trip time increases, link loss rate increases, target throughput increases, and/or number of concurrent traffic increases. We chose to test a link configured with a small round trip time and 0% random loss, at a relatively small target throughput, with only one stream (the test stream) being sent through the link at a time, all factors that may explain the relatively small queue size configuration. This setup is meant to better focus on the effects of queue size, though actual testing situations will most likely require a larger queue size.

4.4 Testing by Spatially Composing Loss Measurements

4.4.1 Test configuration

One feature of the test framework is that measurements from subsequent subpaths can be composed to estimate measurements for complete paths, as described at the end of Section 3.3. The testing process would begin with determining the target run length, and therefore the acceptable loss rate, for the complete path. Then, to allow one to test each of the subpaths separately and make an evaluation about the complete path, one would allocate budgets of loss for each subpath, based on informed expectations about each subpath's performance. One would determine that

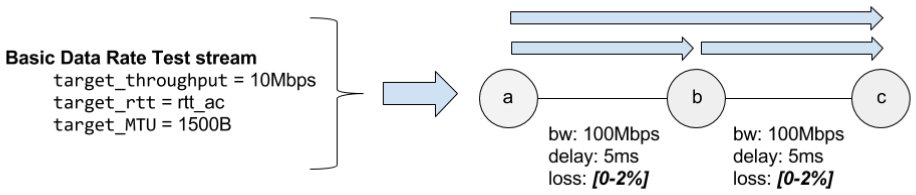

Figure 4-6: Setup for evaluating the spatial composition of loss rate measurements. We vary the configured loss rates of the two links (between 0% and 2%) (bolded and italicized), and observe the loss rates for the three paths $link_{ab}$, $link_{bc}$, and $link_{ac}$. We calculate the composed loss rate $loss'_{ac}$ (according to Equation 3.4) from the average measured $loss_{ab}$ and $loss_{bc}$, which we compare to the average actual measured loss rate $loss_{ac}$.

a complete path is able to support the specified target performance if each subpath's loss rate is estimated to be less than its loss allocation.

Spatial composition of loss rates is based on the assumption that link loss rates are sufficiently independent. This test evaluates this assumption by setting up three hosts a, b, and c (Figure 4-6), and running the Basic Data Rate Test over each link $link_{ab}$ and $link_{bc}$, and over the complete path $link_{ac}$, in order to measure the loss rates.

We vary the configured loss rates of the two links (between 0% and 2%), and observe the loss rates for the three paths. We calculate the composed loss rate $loss'_{ac}$ (according to Equation 3.4) from the average measured $loss_{ab}$ and $loss_{bc}$, which we compare to the average actual measured loss rate $loss_{ac}$. We expect the measured loss rates on $link_{ac}$ to be similar to the composed loss rates if and only if loss rates are sufficiently independent.

4.4.2 Results

Average measured $loss_{ab}$ and $loss_{bc}$, which are used to calculate the composed loss rate $loss'_{ac}$, were found to be very close the configured loss rates for those links, and are not shown here. Next, to determine whether the actual measured loss rates $loss_{ac}$ are similar to the composed loss rates $loss'_{ac}$, we determine whether or not their distributions are similar using a Q-Q plot (Figure 4-7). Using actual loss rates

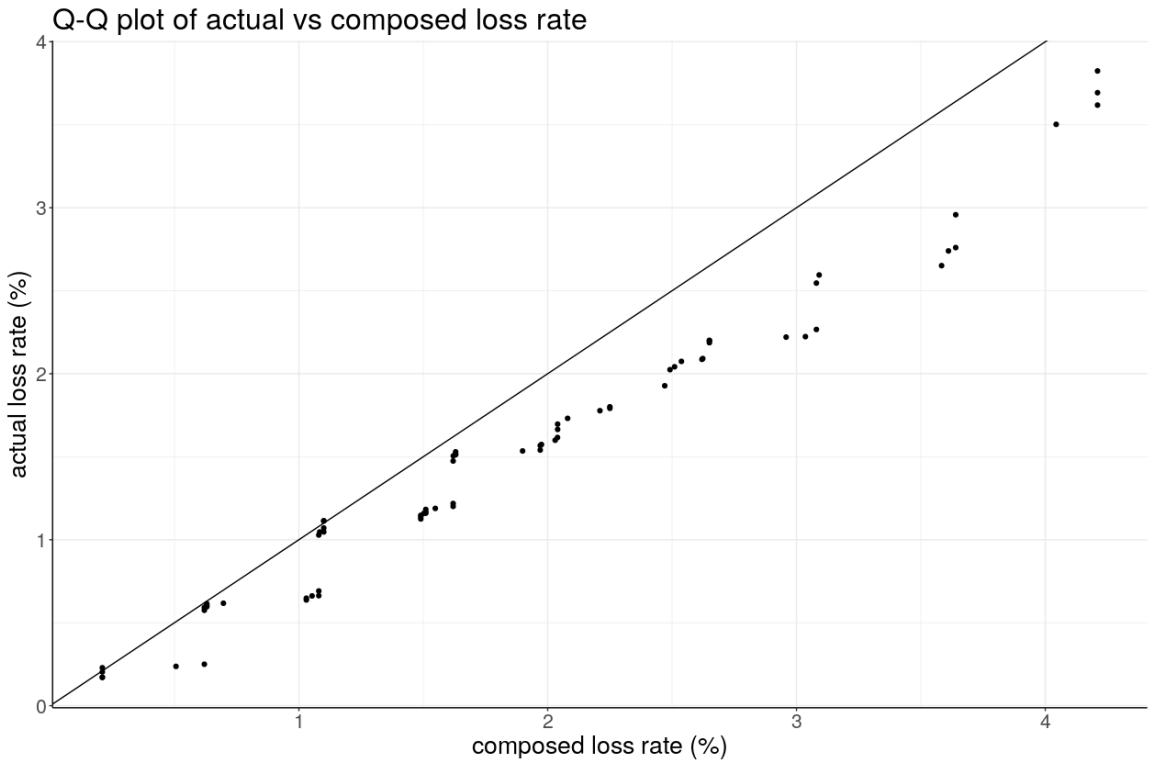

Figure 4-7: Results for evaluation of loss rate composition. A Q-Q plot shows that the composed loss rates and actual loss rates for $link_{ac}$ have similar distributions.

$loss_{ac}$ for test runs that collected more than 1000 samples (why this subset was taken is discussed in the following section), we see that the distributions of actual and composed loss rates are similar, so we can expect loss rates to be sufficiently independent.

4.4.3 Discussion

By comparing the distributions of the composed (expected) loss rates and the measured (actual) loss rates, we find that the measured loss rates are fairly similar to the composed loss rates, implying that link loss rates are independent. Note that these results are for a target throughput (10Mbps) that is much smaller than the link capacity (100Mbps), an evaluation setup that reflects how we imagine MBM being used, but assumptions of independence may not hold in other situations.

An interesting consequence of using the statistical test SPRT for loss rate evaluation is that the measured loss rates for $link_{ac}$ had very high variation, especially

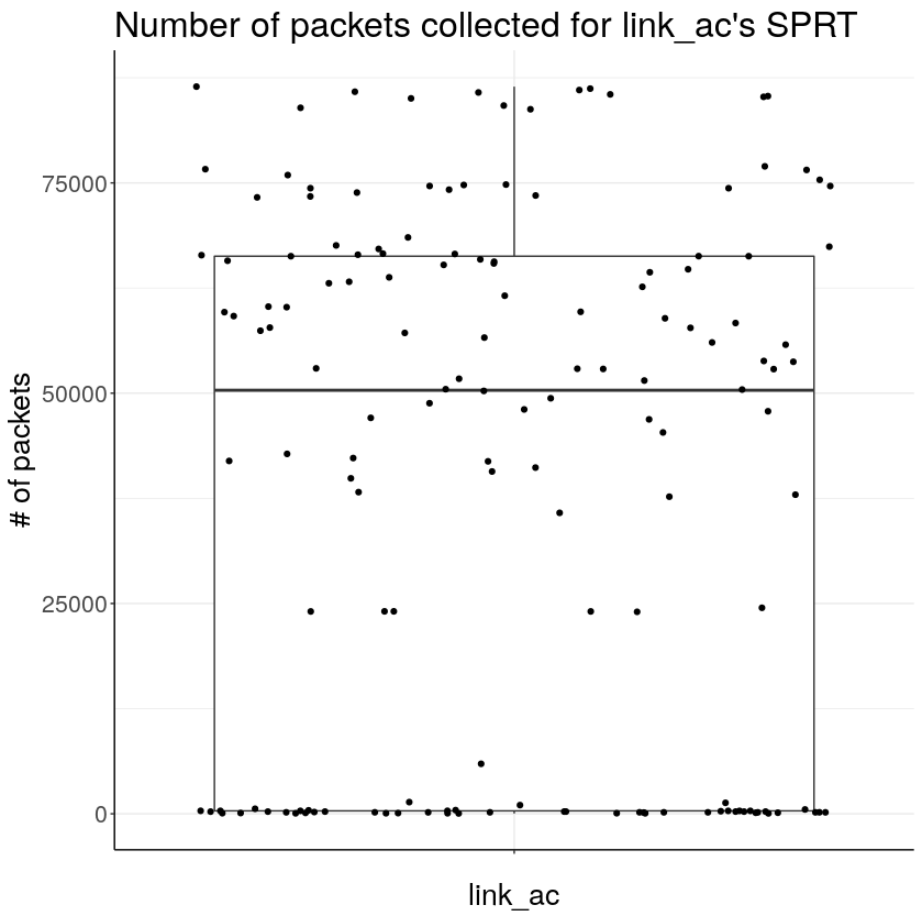

Figure 4-8: Distribution of packets collected for $link_{ac}$'s Sequential Probability Ratio Test before accepting or rejecting the null hypothesis, ranging from about 40 to 86440 packets.

for higher composed losses. Recall that the goal of the SPRT setup is to determine whether the measured loss rate is low enough (in which case the test passes) or too high (in which case the test fails), and SPRT returns upon being able to make a statistically significant claim. SPRT can return a result for a wide range of number of observations. In the case of the Basic Data Rate Tests run on $link_{ac}$, we see that the number of packets SPRT collected before accepting or rejecting the null hypothesis varied widely, ranging from about 40 to 86440 packets (Figure 4-8).

Tests that collected relatively fewer samples tended to report either much lower or much higher loss rates than expected. This is due to the nature of SPRT: if it observes early on a very low or very high loss rate, then it will be able to return sooner, and accept that the loss rate is more likely to be from the null distribution (if

it observes a very low loss rate) or alternate distribution (if it observes a very high loss rate).

The potential disadvantage of SPRT's method is that given a few abnormal early observations, SPRT is likely to return an inaccurate result. This effect can be observed if we also include the measured loss rates for test runs that observed less than 1000 samples in Figure 4-9. 1000 packets was roughly chosen as a sufficient cutoff to differentiate between the first quartile and median number of packets; the higher the cutoff, the more that we approach a hypothesis test that aims to estimate the link's true loss rate with a preset required sample size. Especially as configured loss rates increase, the measured loss rates $link_{ac}$ for test runs that collected less than 1000 packets is more likely to be too high, as the early-on high loss rate caused SPRT returned so quickly.

In our network simulation setup, we can expect that the link's performance is the same over time, and that test runs that collected more samples are more likely to have more accurate loss rate measurements. Even if SPRT does not aim to explicitly estimate the loss rate of the link (instead, its goal is to estimate whether the loss rate is too high or too low), the high rate of overestimations of loss rate, especially when link loss rates are high (in our setup, when composed loss rate $loss'_{ac} > 2.5\%$)), can be concerning.

We discuss the results for SPRT runs that collected relatively few samples here as a side observation, in order to further illustrate the behavior of SPRT. The conclusion here is that for links that are known to sometimes have high loss rates, SPRT may not be an appropriate test, given the observed high rate of false FAILS for high loss rates.

However, the question of SPRT's statistical power is not central to the main purpose of this evaluation, which is to investigate the independence assumption of loss. The conclusion here is that for this setup, in which the target throughput is much smaller than the link capacity, link loss rates can be considered sufficiently independent.

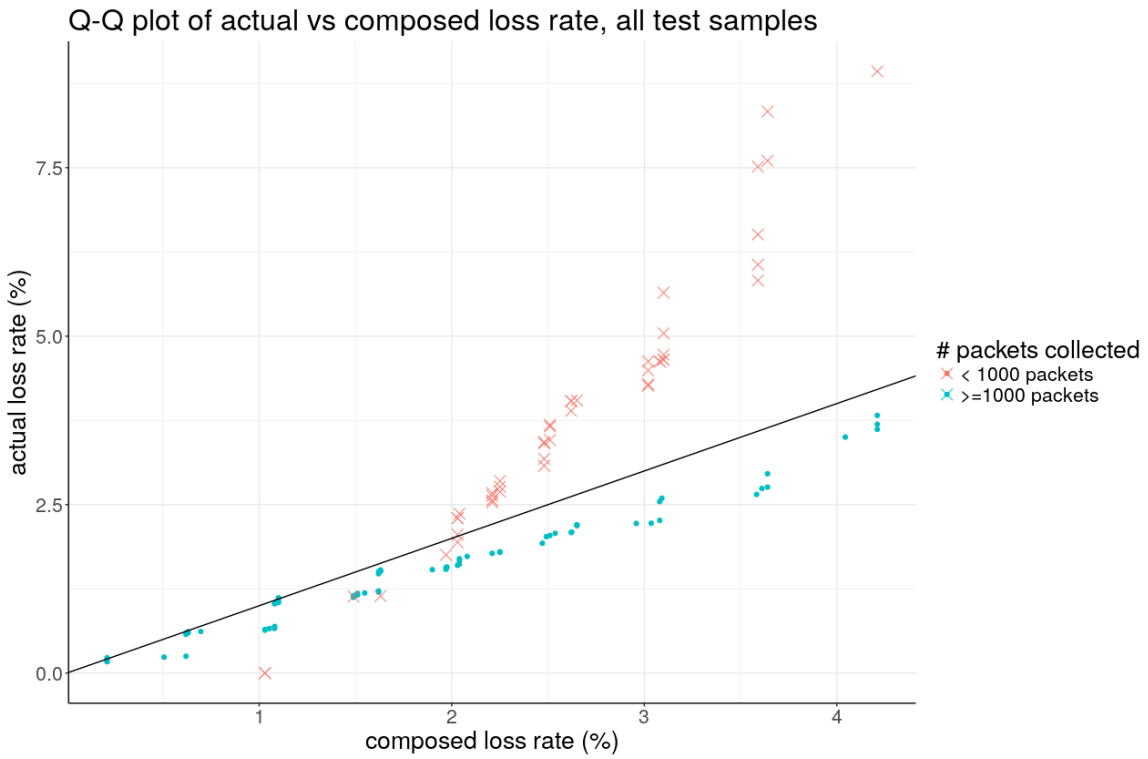

Figure 4-9: We overlay on our Q-Q plot from Figure 4-7 the measured loss rates for all test runs, including those that collected fewer than 1000 packets (red x's). Test runs that collected fewer than 1000 packets are more likely to under or overestimate link loss rate, due to goals and behavior of the Sequential Probability Ratio Test.

Chapter 5

Conclusion

5.1 Further Discussion

In this thesis, we have implemented two tests that measure a network's bulk transfer capacity within a network performance measurement test framework that uses expectations and models of network performance to evaluate a network. We have empirically evaluated the Basic Data Rate Test to be useful in incorporating expectations of round trip time in evaluating a path, and to allow for vantage point independent evaluations of links of interest. We have also compared measured loss rates with expected loss rates calculated from the composition of subpaths' loss rates, which assumes link loss rate independence.

Both vantage point independence and measurement composition work together to allow for the modular testing of networks. While testing individual links, one can evaluate each link according to target performance inputs and/or pre-allocated loss allowances that reflect end-to-end performance expectations. This allows for the modular measurement and evaluation of single links, that are contextualized within specified end-to-end behavior expectations.

This test framework is most advantageous in situations where the individual links of interest are already known or identified, and the tester has access to the endpoints in order to run the test. This is the case in the bottleneck interconnection link example (first described in Section 2.4.1), where likely a administrator is responsible

for intermediate link performance. The test framework is also useful as an end-user-friendly tool, that is executed end-to-end from a user to a test server, in a similar way to existing consumer-friendly tools. MBM offers the ability to configure specific expectations of target performance; in particular, we've shown the usefulness of specifying a target round trip time.

Any network measurement scenario stands to benefit from the test framework's suite of tests that comprehensively determine whether or not a path will be able to maintain the target performance. The need for more comprehensive test suites—in particular with regard to bulk transfer capacity, beyond the Basic Data Rate Test and its constant bit rate stream, which most bulk transfer capacity tests use—is demonstrated in this thesis by the Sustained Burst Test, which is found to be useful in testing a path's ability to support bursty traffic. A path's ability to support burst traffic is evaluated with a test stream that exhibits periodic bursts of traffic, the size and frequency of which are determined by the target performance inputs. Testing this aspect of networks is important especially because queue sizes do affect network performance, and networks are known to have queue-specific traffic controlling mechanisms, such as policers, that will intentionally limit a path's ability to handle traffic with large bursts [11].

5.2 Analysis of Models of Loss

Packet loss plays a fundamental role in TCP congestion control and in how the MBM test framework evaluates path performance. Packet loss is one way to detect congestion on the link. To be more specific about the event we are interested in, we are in TCP's multiplicatively decrease of window size (triggered by a packet loss congestion signal), which causes the throughput of test stream to decrease. Because the tests aim to evaluate whether or not a path can support a given target throughput, throughput measurement is affected by the events that cause changes, in particular decreases, in the test stream throughput. Therefore, when we say "loss rate," more accurately, we mean the count of "decrease window size events" over the count of all received

packets.

Our implementation approximates a count of "decrease window size events" with a count of retransmitted packets, as that is a count that is available in the struct `tcp_info` is the number of retransmitted packets. This is not a perfect estimation, because for example, packets are retransmitted when they are determined to be lost, but not all retransmitted packets are actually lost.

In addition to defining loss, the models used by the framework as specified by Mathis in the IETF Internet-Draft [14] makes assumptions about loss rate independence. First, it assumes that a link's loss rate (count of "lost" packets over count of total packets sent) is equal to each packet's loss probability, which assumes that loss probabilities are independent and identically distributed. Second, for the statistical evaluation of loss rate using SPRT, by modeling a packet transmission with a Bernoulli process, it also assumes that packet loss probabilities are independent and identically distributed. Third, for the composition of loss rates across subsequent subpaths, it also assumes that link loss rates are independent. These assumptions must be more closely evaluated, as loss rates on the internet have been extensively studied and have been found to be typically not well modeled as independent [2, 15, 7].

Regarding the link loss rate independence assumption for the purposes of loss rate composition, our empirical tests (Section 4.4) demonstrate that loss rates can be considered sufficiently independent in our simple network simulation setup, in which our test stream's target throughput is much lower than the link's capacity. We used a test stream with a relatively smaller target throughput reflect an intended typical use case for MBM. This result is consistent with past analyses, that have found packet losses to be random when test traffic throughput is a small fraction of the link capacity [2], but should be further investigated to consider other factors that might affect loss rate independence.

Loss rate independence is also an assumption used in the statistical estimation of loss rate. In general, loss rates are more likely independent if 1) test traffic throughput is relatively small compared to the link capacity, and 2) if the target loss rates (acceptable loss rate cutoffs) are conservatively defined in the statistical test. Regarding

the first point, test traffic that doesn't hit link capacity is less likely to experience congestion-related (in other words, related to filled buffers) losses, which are generally correlated. Regarding the second point, conservative target loss rate definitions will more likely avoid the "progressive onset of loss" (in a similar way that active queue management tail-drops aim to avoid the onset of congestion), which will lead to highly correlative loss. We went ahead with our simple statistical models and our assumptions of independence with this reasoning in mind, though future studies should further understand the implications of these assumptions, and possibly explore alternate models of packet transmission.

5.3 Analysis of Target Run Length Derivations

The test framework relies on reasonable estimations of target run length (in our specification, defined in Equation 3.2), which then inform the loss rate cutoffs p_0 (defined in Equation 3.3) and p_1 used in the statistical test for estimating the loss rate. Mathis intentionally proposes calculations that are conservative [14]. Since how target run length is calculated will affect whether or not the test passes or not, the conservativeness of the test will affect its rates of false PASSes and false FAILs. If MBM is to become a trusted means of network measurement, the conservativeness and rates of false results will need to be continually and transparently evaluated and disclosed.

Though all models are approximate, a rough evaluation of the Mathis's calculations used in the IETF Internet-Draft and this implementation gives a sense of their degree of conservativeness. Recall the following definitions used in the IETF Internet-Draft, described in Equations 3.1 (for window size W), 3.2 (for target run length R), 3.3 (for maximum acceptable loss rate cutoff p_0). Then p_0 can be written as

$$
\begin{aligned}
p_0 &= \frac{1}{R} \\
&= \frac{1}{3 \times W^2} \\
&= \frac{\texttt{target\_MTU}^2}{3 \times \texttt{target\_rtt}^2 \times \texttt{target\_throughput}^2}
\end{aligned}
$$

Now, if we would like to model the throughput t_0 that would result from the loss rate cutoff p_0, we can use Mathis's TCP throughput equation (Equation 2.1), which was found to be fairly accurate in shape when compared to empirical results (Figure 2-3). We set $p = p_0$, and assume $\texttt{target\_MTU} = MTU$ and $\texttt{target\_rtt} = rtt$, to calculate the expected throughput t_0:

$$
\begin{aligned}
t_0 &= \frac{MTU \times C}{rtt \times \sqrt{p}} \\
&= \frac{MTU \times C}{rtt} \times \sqrt{\frac{3 \times \texttt{target\_rtt}^2 \times \texttt{target\_throughput}^2}{\texttt{target\_MTU}^2}} \\
&= \sqrt{3} \times C \times \frac{MTU}{rtt} \times \frac{\texttt{target\_throughput} \times \texttt{target\_rtt}}{\texttt{target\_MTU}} \\
&= \sqrt{3} \times C \times \texttt{target\_throughput}
\end{aligned}
$$

Likewise, given that the statistical test fails (accepts the alternate hypothesis) if the link's loss rate is estimated to be greater than $p_1 = \frac{4}{\texttt{target\_run\_length}} = 4p_1$, using the same process we derive the throughput t_1 corresponding to a loss rate of $p = p_1$:

$$
t_1 = \frac{\sqrt{3}}{2} \times C \times \texttt{target\_throughput}
$$

In other words, given a test's target performance inputs, in which the test stream is at `target_throughput`, the test will fail if the link's loss rate is found to be greater than the upper cutoff p_1. When $C > \frac{2}{\sqrt{3}} \approx 1.154$, $t_1 > \texttt{target\_throughput}$. This im-

plies that there are cases where the test will fail for a given `target_throughput`, even when the throughput t_1 derived from such a loss rate $> p_1$ is *more* than `target_throughput`.

Having such a value for C is not at all unreasonable, as we used $C = \sqrt{\frac{3}{2}} > 1.154$, derived for situations of periodic loss and acknowledgements for every segment [4] for our illustrations (Figures 2-2 and 2-3). This value for C was even suggested to be a bit low, according to our empirical BTC throughput results (Figure 2-3).

When we use this value for C in calculating t_0, we find that for a test with a traffic stream at `target_throughput`, the test passes only if the link's loss rate is found to be less than p_0, which is theoretically expected to be able to support a throughput t_0 that is more than 2 times the `target_throughput`.

Given that the TCP throughput model used is sufficiently accurate, these calculations offer a greater sense that the specifications of target run length and loss rate cutoffs p_0 and p_1 are very conservative, and are likely to cause undesirably high levels of false FAILs. A closer examination of the assumptions and derivations for p_0 and p_1 is advisable moving forward.

5.4 Future Work

To build upon the work in this thesis, though the virtual network setups enabled repeatable and configurable testing, which were invaluable for our test development and evaluation, future work includes deploying and evaluating the two tests in a real network, such as on M-Lab's testing infrastructure. Real-world testing would allow us to continue investigating what are appropriate network models and target performance inputs for MBM to use. It will also allow us to further evaluate the conservativeness of the derivations, and the implications of various target loss specifications. Though the parameters used in this thesis are all considered fairly conservative, the parameters used in different testing situations will have to tolerate different levels of false FAILs and false PASSes, and the conservativeness of the models will need to reflect this. Also, in general, it would be interesting to determine the dynamic ranges of these

tests, given varying link configurations, target performance inputs, network models, statistical tests, and other factors.

A most compelling study would find and analyze cases of bottleneck links that exhibit the characteristics in our motivating example of the interconnection link, to further explore the benefits of vantage point independent measurements that use target round trip times and other target performance inputs to incorporate end-to-end expectations of a network into its measurement.

Loss rate composition and the allocation of loss rate budgets per subpath will be another characteristic to better understand, given its role in MBM's piecewise measurement capabilities. MBM must better understand the scenarios under which composition is appropriate (when loss rates are sufficiently independent), as well as the conservativeness and flexibility (or lack thereof) of the budgeting of subpath loss rate.

In addition, the suitability of SPRT as the hypothesis test for loss rate estimation must be better determined in varying network conditions. Though SPRT's optimization to send as few test packets as needed is nice to incorporate into a routine network testing framework, it does affect the test's accuracy. The high rates of false FAILs and PASSes that SPRT introduces under some network conditions may make it inappropriate.

As network performance measurement becomes a more central topic in the maintenance and regulation of networks, and network link capacities increase to the point of making typical network-flooding BTC measurement techniques inappropriate, these studies will be essential for the further development of vantage point independent and modular end-to-end measurement of network performance.

Bibliography

[1] D. Belson. State of the internet q4 2016 report. Report, Akamai, March 2017.

[2] J. C. Bolot. End-to-end packet delay and loss in the internet. *Journal of High Speed Networks*, 2(3), December 1993.

[3] A. Morton et al. Spatial composition of metrics. IETF RFC 3148, January 2011.

[4] M. Mathis et al. The macroscopic behavior of the tcp congestion avoidance algorithm. *Computer Communication Review*, 27(3), July 1997.

[5] M. Mathis et al. A framework for defining empirical bulk transfer capacity metrics. IETF Network Working Group RFC 3148, July 2001.

[6] M. Mathis et al. Model based metrics. IETF 93 IPPM Working Group Presentation Slides, July 2015. Available at `https://www.ietf.org/proceedings/93/slides/slides-93-ippm-9.pdf`.

[7] M. Yajnik et al. Measurement and modelling of the temporal dependence in packet loss. *INFOCOM 1999*, 1:345–352 vol.1, Mar 1999.

[8] S. Bauer et al. The evolution of internet congestion. *TPRC 2009*, August 2009. Available at SSRN: `https://ssrn.com/abstract=1999830`.

[9] S. Bauer et al. Understanding broadband speed measurements. *TPRC 2010*, August 2010. Available at SSRN: `https://ssrn.com/abstract=1988332`.

[10] S. Bauer et al. Improving the measurement and analysis of gigabit broadband networks. *TPRC 2016*, March 2016. Available at SSRN: `https://ssrn.com/abstract=2757050` or `http://dx.doi.org/10.2139/ssrn.2757050`.

[11] T Flach et al. An internet-wide analysis of traffic policing. *SIGCOMM 2016*, August 2016.

[12] V. Mohan et al. Active and passive network measurements: A survey. *International Journal of Computer Science and Information Technologies*, 2(4), 2011.

[13] M. Mathis. Model based metrics. IETF 90 IPPM Working Group Presentation Slides, July 2014. Available at `https://www.ietf.org/proceedings/90/slides/slides-90-ippm-4.pdf`.

[14] M. Mathis. Model based metrics for bulk transport capacity. IETF IP Performance Working Group Internet-Draft, February 2017.

[15] V. Paxson. End-to-end internet packet dynamics. *SIGCOMM Comput. Commun. Rev.*, 27(4):139–152, October 1997.

[16] R. Pfeiffer. Measuring tcp congestion windows. *Linux Gazette*, March 2007. Available at `http://linuxgazette.net/136/pfeiffer.html`.

[17] A. Wald. *Sequential Analysis*. John Wiley & Sons, Inc", New York, 1947.