QUIC-TCP: VALIDATION OF QUIC-TCP THROUGH NETWORK SIMULATIONS

by

Brian Boughen

A Thesis Submitted to the Faculty of

The College of Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

Florida Atlantic University

Boca Raton, Florida

December 2013

QUIC-TCP: VALIDATION OF QUIC-TCP THROUGH NETWORK SIMULATIONS

by

Brian Boughen

This thesis was prepared under the direction of the candidate's thesis advisor, Dr. Xin Wang, Department of Computer & Electrical Engineering and Computer Science, and has been approved by the members of his supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

SUPERVISORY COMMITTEE:

For Xin Wang, Ph.D.
Thesis Advisor

Hanqi Zhuang, Ph.D.

Imad Mahgoub, Ph.D.

Nurgun Erdol, Ph.D.
Chair, College of Electrical Engineering and Computer Science Department

PP Mohammad Ilyas, Ph.D.
Dean, College of Engineering and Computer Science

Barry T. Rosson, Ph.D.
Dean, Graduate College

November 19, 2013
Date

ACKNOWLEDGEMENTS

The author wishes to express his sincere thanks and appreciation to his wife and daughter for their support and patience throughout the writing of this manuscript and studies. The author is grateful to his thesis sponsor whose guidance and understanding made this work possible. The author is extremely appreciative of Zhaoquan Li whose help was invaluable. The author is grateful to the staff and support personnel of FAU who helped him through on his journey.

ABSTRACT

Author:	Brian Boughen
Title:	QUIC-TCP: Validation of QUIC-TCP through network simulations
Institution:	Florida Atlantic University
Thesis Advisor:	Dr. Xin Wang
Degree:	Master of Science
Year:	2013

The scalability of QUIC-TCP was examined by expanding previous developmental 11-node, 4-flow topology to over 30 nodes with 11 flows to validate QUIC-TCP for larger networks. The topology was simulated using ns-2 network simulator with the same ns-2 module of FAST-TCP modified to produce QUIC-TCP agent that the original development used. A symmetrical topology and a random topology were examined. Fairness, aggregate throughput and the object of the utility function were used as validation criteria. It was shown through simulation that QUIC-TCP optimized the utility function and demonstrated a good balance between aggregate throughput and fairness; therefore QUIC-TCP is indeed scalable to larger networks.

QUIC-TCP: VALIDATION OF QUIC-TCP THROUGH NETWORK SIMULATIONS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 Overview

As the world moves away from the tethered life of wired Internet, demand for mobile Internet has increased at an exponential rate. The iPhone is almost 6 years old. The iPad is about to celebrate its 3rd birthday. These 2 devices and their electronic cousins have driven the almost insatiable craving for mobile data. Consider that the global mobile data traffic in 2012 was approximately 12 times greater than the entire Internet traffic in 2000. At some point this year, 2013, the number of mobile devices will exceed the entire worldwide human population. By 2017, just a mere 4 years from now, estimates are that global mobile data usage will increase by another 12 times [1]. As a result of consumer demand and the fact that wireless spectrum is a scarce and regulated resource, technology must change very rapidly in response to consumer's expectations by providing an efficient and fair allocation of resources.

As Internet data usage has shifted from wired networks to mobile wireless networks, the build out of wireless networks has exploded and there is an unrelenting quest to optimize every component of the network. The biggest obstacle for wireless networks is that they are subject to the randomness inherent in wireless technology. Fading, which is essentially interference from the environment, is the largest component

of signal loss with noise and other interference a distant second and third contributors. As the signal propagates through the atmosphere, it is subject to diffraction, reflection and free-space transmission. As a result, the received signal is a combination of multiple paths which can interfere constructively or destructively with each other resulting in a "multipath" signal. Fading occurs when one or both the transmitter and receiver are moving which causes a time varying component to the multipath [38]. Given the increasing desire for wireless data, networks are pushed to their capacity more than ever and will continue to be in the foreseeable future.

In a distributed network where users share resources, some protocol algorithm must govern the efficient and fair use of the resources. Transport Control Protocol, TCP, was specifically designed for this purpose and works efficiently in networks where the number of users and resources vary unpredictably over time [2]. Feedback is an ideal technique for sources to adjust their transmission window size depending on the congestion level; however, message passing siphons available bandwidth. The strength of TCP is its end to end congestion control algorithm which does not use explicit message passing thereby limiting its impact on bandwidth and improving its overall scalability. The current most common version of TCP, Reno, was developed in 1988 and has been modified several times since then [3]. TCP has performed well during that time considering that the Internet has scaled up by 6 orders of magnitude in size, speed, load and connectivity in the past decade [4]. In addition, it is well known and demonstrated that TCP performs poorly in networks with large bandwidth delay products and/or

wireless links which are innately lossy [5] [6] [7]. Given the migration to lossy wireless links and TCP's poor performance with them, TCP must be further developed or it will become a performance bottleneck.

1.2 Background

Much work has been done on both a heuristic and theoretical level. Recently, a duality network fluid model for end to end control was developed to better understand large scale networks [8] [9]. The model proposes that the source algorithm dynamically adjusts its window size in response to congestion from the source to the receiver. In addition, the link algorithm modifies a congestion measure and feeds it back to the source using that link. Thus, congestion control becomes a parallel algorithm, more specifically a distributed algorithm, which attempts to optimize the utility function derived from the combination of TCP scheme and queue management scheme. Source rates act as primal variables and congestion measures take on the role of dual variables to become the Lagrange multiplier that solve the dual problem in equilibrium.

Along with the duality model, tools were developed to model, analyze and design combinations to optimize source rates constrained by link capacities. A paradigm based on the scheme and other network parameters, network utility maximization or NUM, leads to convergence towards an optimum equilibrium constant. [10] [11]. This utility function may be based on network variables such as rate, latency, jitter, etc. Since many of TCP schemes and enhancements were heuristic, the NUM paradigm was

conceptualized to redesign Internet protocols by reverse engineering the existing protocols. NUM's first goal was to demonstrate that TCP congestion control solves a NUM problem [8]. NUM congestion control schemes can be reduced to either a dual controller or a primal-dual controller. Most of the schemes fall under dual controller where two different timescales are used, one for source rates with a static algorithm and one for link queue lengths with a gradient-type algorithm. For primal-dual controllers, source rate and queue lengths are updated at the same time. While dual controllers are the most common and function well in some applications, they rely on message passing and as such are not easily implemented in an end to end congestion control algorithm like TCP. As a result, they are not easily scalable and not easily implemented. They are not good candidates for existing TCP modifications because too much existing infrastructure would need to be changed. Therefore, the NUM paradigm dual controllers have been a good topic for academic research and understanding; however, there is little chance of actual implementation considering the proliferation of TCP based protocols such as Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), or File Transfer Protocol (FTP).

In addition to the duality model and NUM paradigm, engineering heuristics were designed and implemented for TCP, namely TCP-Vegas, HSTCP, STCP, TCP-Westwood, TCP-Veno, TCP-Jersey, etc. [12, 13, 14, 15, 16, 17]. As with most heuristic designs, they were validated with testing and/or simulation but they lack the theoretical groundwork so therefore performance cannot be guaranteed. On the other hand, Mo-

Walrand and FAST-TCP [18] [7] congestion controllers were designed and validated in a more theoretical manner. They both use queueing delays as Lagrange multipliers to optimize their utility functions. Both proved out the stability and/or convergence of their schemes analytically or at least demonstrated it through simulations/experiments. However, all of the analysis was performed for wired networks which have constant and non-coupled link capacities. For wireless networks, link capacities are not constant nor are they independent. Direct implementation of these wired schemes on a wireless network could cause unfairness and a significant performance reduction unless there is some type of optimization for joint link-scheduling [19]. Therefore, a combination of the systematic approach like Mo-Walrand and the use of a primal-dual controller from the NUM paradigm is an area that warrants further investigation.

1.3 Contribution

As we previously indicated, NUM congestion control solutions with primal-dual controllers use queue lengths as Lagrange multipliers. In contrast, we propose that a better choice would be to use queue delay based TCP schemes with queue delay serving as Lagrange multipliers. This is buttressed by the fact that queue delay can be estimated at the source without explicit message passing whereas queue lengths cannot. The overhead for queue length control would be extensive while the estimation of queue delay is contained within the parameters of end to end TCP congestion control and as such allows TCP to be scalable. Therefore, a window-based congestion controller

implemented at the source is proposed where the primal source rates and dual queueing delays become functions of window size. This scheme will adjust window size in response to implicit primal-dual calculations. So, to optimize TCP congestion control, a scheme must be developed that adjusts window size based on implicit primal-dual solvers.

This thesis will validate the existence of a readily deployable, scalable and optimal network scheme for wireless TCP applications where queueing delay is used as the congestion measure. As previously mentioned, the design must be compatible with coupled wireless links so a cross layered design is mandatory (layers 2 and 4 in the Open Systems Interconnection, OSI, model). The congestion control and link scheduling must work in tandem to achieve network optimization. The key is the wireless link scheduler at the access point, AP. Unlike a wired network where link capacities are fixed, in a wireless network the link capacities vary randomly with time and asynchronously with users. As a result the scheduler must realize a multi-user variable channel solution. This is best solved by max weight-type scheduler. [44]

With link scheduling addressed and shifting focus to congestion control design by revisiting the Mo-Walrand scheme, a class of window based QUeueIng-Control (QUIC) algorithms is developed. Here, it will be shown through simulation that these algorithms along with the queue delay based max weight link scheduler function as implicit primal-dual solvers leading to convergence of an optimal solution in equilibrium. The goal of

QUIC-TCP is to take what we have learned from network optimization theory as well as practical Internet designs, and then develop the optimization algorithms that will function within the existing Internet architecture. This new thinking could shift research towards a design-space oriented cross-layer optimization/design of Internet protocols.

1.4 Outline of the Thesis

The remainder of this thesis is organized as follows: Chapter 2 will cover background, TCP, vision and development. Chapter 3 will cover describe our proposed QUIC-TCP and link scheduling schemes. Chapter 4 will cover simulation design and set-up. Chapter 5 will discuss results and Chapter 6 contains concluding remarks.

CHAPTER 2: TCP & LINK SCHEDULING BACKGROUND

2.1 Introduction

This section will give an overview of TCP and link scheduling schemes. That will be followed by a quick examination of the existing shortfalls in current research that our research will examine. In the next chapter, the proposed QUIC-TCP and link scheduling schemes will be discussed as well as analytical results.

2.2 TCP

"TCP is, after all, mainly about efficiency and adaptation: how to deliver data utilizing the maximum available (but fair) share of a dynamically changing network capacity so to reduce the delay." [2]

To function efficiently, TCP was designed to optimize network traffic at the end nodes so that feedback on the network is reduced thereby maximizing bandwidth or channel utilization. For a simple comparison, consider two people engaged in polite conversation. When one talks, the other listens. If there is a pause in the conversation and both parties start talking at the same time, they will both stop talking some random

amount of time and then one of them will restart. An even better analogy would be a pipe 20' or 30' long so the two speakers are separated beyond audible range. A lot of playgrounds built within the last 10 years have such a pipe running 20' or so underground with a funnel on either end. Children are always fascinated that they can see their parent talking but only hear the parent's voice coming out of the funnel. This "toy" is analogous to a communication system similar to a wired network with two users. When one speaks, the other must listen or their voices will collide and the result is unintelligible. Now imagine this simple communication system expanded to include 10 or more users each with his own pipe connected to the main pipe. There would need to be some type of rules or protocols that all users agree to or only noise would come out of the funnels. TCP defines the communication protocol at the transport level. These protocols were developed for wired networks and have some efficiency difficulties in response to the irregularities of wireless networks.

While there are many components of TCP, we will review only the specifics germane to this thesis. An overview relative to congestion control for loss based schemes will be reviewed followed by a progression to delay based schemes. Before the source initiates transmission, the network is checked to see if the channel is available (no other sources are transmitting). The checking is known as CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance). If the channel is available to transmit, then the source will begin transmitting. If another node happens to transmit at the exact same time, a collision will occur, both sources will back-off a random amount of time, and

then both will check to see if the channel is available before retransmitting. Since the back-off time is a random number, the probability that both sources choose the same random time period is low, so one source will begin transmitting before the other. The second source will determine that the channel is busy and wait until the next available time period to begin sending its data.

How will the sender know that the message has been received? TCP uses acknowledgement, ACK, from the receiver to the sender to confirm that the message has been received. When the source sends a message, it waits for an ACK from the receiver to assure message integrity. Once the receiver has received a sequential message, it sends an ACK back to the sender. The ACK must be received before the Round Trip Time Timer, RTT timer, expires. If the time waiting for an ACK exceeds the RTT, then the sender will retransmit assuming the packet is lost. In addition to the RTT timeout, TCP offers another opportunity to detect packet loss. When the receiver receives an out of sequence packet, it sends a duplicate ACK of the last previously acknowledged in-sequence packet. If the sender receives 3 duplicate ACKs, it assumes the packet has been lost and begins its retransmission sequence [39].

In a dynamic network with sources joining the network at random times, the available buffering between a source and its receiver is constantly changing. TCP uses these enroute buffers to its advantage. Rather than send a burst of data then wait for the ACK which would lead to a lot of dead time thereby reducing the efficiency of

bandwidth utilization, TCP uses the concept of "in flight" data or "in flight" packets. Contained within the ACK is the available buffer size of the receiver which is the maximum amount of unacknowledged packets that the TCP sender can have in the channel at one time (flow control). Since the "network capacity" may be less than the receiver buffer, TCP must consider this value as well. Therefore, the window size, w_s, of in flight packets is the minimum of either the receiver buffer or network capacity as follows:

$$w_s = \min(receiver\ buffer, network\ capacity)$$
$$= max\ \#\ packets\ in\ flight \quad (1)$$

Given the unknown capacity of the network driven by sources accessing the network asynchronously, the sender begins "slow start" where it starts out slowly sending data. TCP will transmit the data in bursts the size of the current w_s. If the ACK is successful, then the sender doubles the size of the burst (up to the receiver buffer size). If no ACK is received before the RTT is exhausted, then TCP assumes that a collision has occurred due to congestion. As a result, it resets to a lower window size, typically ½ the current window size. This increase and decrease is known as Additive Increase Multiplicative Decrease (AIMD). Reno (the most prevalent TCP scheme currently in implementation) and other loss based congestion control schemes use packet loss as described above to adjust their window size and thereby control throughput at the source. The methodology of these schemes is to increase the flow until the point of congestion

and then throttle back. The cycle repeats until there is no more information to send. In a sense, these schemes create congestion, then overcorrect, create congestion, then overcorrect all the while wasting network capacity in the process.

If congestion could be anticipated or predicated, then the flow rate could be marginally backed off so as to not waste excessive network capacity. Vegas, the first of these queueing-delay based schemes, measures the delay between the current RTT and the anticipated RTT time (in practice the minimum RTT observed is used as an approximation of propagation delay without queueing delay present – an uncongested network.) Assuming that the minimum RTT resulted from no congestion on the network, then an increased RTT (and hence increased delay) would show an increase in congestion. "Vegas' strategy is to adjust the source's sending rate (congestion window) in an attempt to keep a small number of packets buffered in the routers along the transmission path [20]." The original developers of Vegas defined the expected throughput as:

$$Expected = WindowSize/BaseRTT \quad (2)[12]$$

where $WindowSize$ is the current window size, $BaseRTT$ is the min RTT measured and $Expected$ is the anticipated throughput [12]. Vegas then compares the actual measured flow calculated from the current RTT, named $Actual$, to the $Expected$ with the following:

$$Diff = Expected - Actual \quad (3)[12]$$

Two other variables are added such that $\alpha < \beta$. If $Diff < \alpha$, then the window size is increased. If $Diff > \beta$, then the window size is decreased. This queue-delay feedback prevents congestion and allows Vegas to use more of the available bandwidth. If we let $D_s(t)$ = the current round trip time and d_s = the round trip propagation delay, then we have the following relationships:

$$Diff = \frac{w_s(t)}{d_s} - \frac{w_s(t)}{D_s(t)}, \; Expected = \frac{w_s(t)}{d_s}, \; Actual = \frac{w_s(t)}{D_s(t)} \quad (4)[12]$$

More mathematically it could be written as the following:

$$w_s(t+1) = \begin{cases} w_s(t) + \frac{1}{D_s(t)} & if \; \frac{w_s(t)}{d_s} - \frac{w_s(t)}{D_s(t)} < \alpha_s \\ w_s(t) - \frac{1}{D_s(t)} & if \; \frac{w_s(t)}{d_s} - \frac{w_s(t)}{D_s(t)} > \alpha_s \\ w_s(t) & else \end{cases} \quad (5)[20]$$

2.3 TCP Enhancements

As the bandwidth delay of networks has increased, it is commonly recognized that loss based schemes are inadequate to maintain stability and efficiency for congestion control [43]. Increasingly, it is believed that TCP congestion control schemes that take advantage of delay will provide a more stable and efficient congestion control than loss

based schemes like Reno. In a wired environment, TCP treats all packet loss as a collision. However, the random nature of wireless and the delays associated with wireless cause problems for TCP [21]. When the wireless signal fades, drops, or the delay is large, TCP treats this condition as a packet collision and starts its collision routine which reduces throughput. As a result of the randomness of wireless, TCP does not optimize the throughput [22].

There have been attempts to enhance TCP to compensate for its poor performance in wireless environments. Schemes like HSTCP and STCP [13] [14] were developed to increase window size more aggressively than linear increase of one packet per RTT and decrease less drastically than multiplicative decrease per loss event. Further, some heuristic schemes were developed to differentiate between packet loss and wireless link errors like ATCP, TCP-Veno, TCP-Westwood and TCP-Jersey [23] [16] [15] [17]. However, given that these were heuristic solutions, there is no theoretical guarantee that they should provide the optimal solution.

In addition to exploiting delay where queueing delays are used as Lagrange multipliers, Mo-Walrand and FAST-TCP were developed analytically and then validated through simulation and testing. This method provided a theoretical foundation that they should guarantee an optimal solution. However, both were developed for wired networks and neither considered a wireless network. Clearly there is a need for a TCP based congestion control scheme that utilizes queueing delay as a control measure. This

scheme needs to be developed systematically/theoretically then validated through simulation or testing. In addition, this scheme must compensate for the variability inherent in wireless networks.

2.4 Link scheduling

Networks are increasingly becoming more complicated with various combinations of hybrid wired backbone and multi-hop access points for mobile communication [48]. For ease of modeling and discussion consider a simple wired backbone with one hop for wireless mobile communication. Again for simplicity, consider downlink transmission only. Recently, several TCP enhancements have been proposed to identify/correct wireless link errors (random packet loss) from link level packet loss [15] [16] [17]. In contrast, we assume that packet loss can be sufficiently masked through link layering schemes. Given the link layering schemes such as error correction coding, fast link adaptation and hybrid ARQ outlined in wireless standards such as IEEE802.11, IEEE802.16, WCDMA HSDPA and CDMA2000 EV-DO [42] [24] [25] [26], packet loss can be sufficiently masked to achieve reliable transmission over wireless links. Our assumption is further bolstered because it was shown that without even employing link layering schemes network coding at the source could mask packet loss [27]. The challenge becomes understanding the link capacities as they are influenced (and more than likely coupled due to rate adaptation) by the broadcast nature

of wireless and channel fading which causes the wireless link capacities to vary with channel conditions and the resource algorithms used.

Our network is modeled with the set of logical data links, L, subject to $L = L_f \cup L_w$ where L_f is the wired link set and L_w is the wireless link set. For the wired links, any link $l \in L_f$ is constrained by the wire and therefore assumed to have an independent and constant link capacity c_l. For the wireless links, r_l is the link capacity and its vector is $\boldsymbol{r} := \{r_l, \forall l \in L_w\}$. Since all the wireless links are subject to random fading, then per fading realization $\boldsymbol{h}$ the largest rate set for the potential coupled links is bounded by a region $\mathcal{R}(\boldsymbol{h})$ which is assumed to be closed and convex. Given that $\boldsymbol{r}$ is a random variable, we have an expectation about the average that $r \in \bar{\mathcal{R}} := \mathbb{E}_h[\mathcal{R}(\boldsymbol{h})]$ where $\mathbb{E}_h$ is the expected value over the fading distribution. Set S comprises the set of sources that share the network. Each end to end flow on the network is represented by their source such that $s \in S$. Each source s sends its data to its receiver via wired or wireless links determined as the lowest cost per the routing table. Source flow s travels from its source to destination via links where $L(s) \subseteq L$ at a transmit rate of x_s with a utility function $U_s(x_s)$. The objective of any scheme is to maximize the aggregate source rates x_s. Further, a solution that can converge to an efficient and fair equilibrium is sought especially when flows have large propagation delay. For each link l, the set of sources that use that link are represented by $S(l) \subseteq S$. Consistent with TCP-Vegas and TCP-FAST [7] a weighted proportionally-fair utility function $U_s(x_s) = p_s log x_s$ with a given weight vector $\boldsymbol{p} := \{p_s, \forall s\}$ is subject to (s. t.) link constraints as follows:

$$\max \quad \sum_{s \in S} \; p_s log x_s \tag{6}$$

$$\text{s. t.} \quad \text{(C1)} \; \sum_{s \in S(l)} \; x_s \leq \; c_l, \; \forall l \in L_f$$

$$\text{(C2)} \; \sum_{s \in S(l)} \; x_s \leq \; r_l, \; \forall l \in L_w$$

$$\text{(C3)} \; \boldsymbol{r} \; \in \; \bar{\mathcal{R}}, \; x_s \; \geq 0, \; \forall s \in S.$$

The problem posed by (6) is to find an optimal solution to congestion control while at the same time optimizing wireless-link scheduling for a last hop wireless network. This problem has been posed before and is readily solvable by queue length based NUM solutions [10] [28] [29] [30]. Similarly, heuristic based TCP enhancements previously discussed can produce results [5] [17] [31] [32] [33]. However, as pointed out earlier, the heuristic based TCP enhancements cannot guarantee an optimal solution and we will show in the next section that NUM based queue lengths solutions have their limitations as well.

2.5 NUM Schemes

Examining (6) optimization somewhat closer, we observe that it is a convex optimization [40]. In addition, the primal variables are $\boldsymbol{x} := \{x_s, \forall s\}$ and the wireless link capacity vector previously stated $\boldsymbol{r}$. If we let the Lagrange multipliers for the link capacity constraints be $\lambda := \{\lambda_l, \forall l\}$ following the development in [12], then we can use the Karush-Kuhn-Tucker (KKT) conditions to define $\lambda^s := \sum_{l \in L(s)} \lambda_l$. This implies that the optimal primal $\{\boldsymbol{x}^*, \boldsymbol{r}^*\}$ variables and dual variables $\boldsymbol{\lambda^*}$ satisfy the following:

$$\frac{p_s}{x_s^*} = \lambda^{s*} := \sum_{l \epsilon L(s)} \lambda_l^* \ , \forall s \in S \tag{7}$$

$$\lambda_l^* \left(c_l - \sum_{s \epsilon S(l)} x_s^* \right) = 0, \forall l \in L_f \tag{8}$$

$$\sum_{s \epsilon S(l)} x_s^* \ \leq \ c_l, \forall l \in L_f \tag{9}$$

$$\lambda_l^* \left(r_l^* - \sum_{s \epsilon S(l)} x_s^* \right) = 0, \forall l \in L_w \tag{10}$$

$$\sum_{s \epsilon S(l)} x_s^* \ \leq r_l^*, \forall l \in L_w \tag{11}$$

$$\boldsymbol{r}^* \in \arg max_{r \in \mathcal{R}} \quad \sum_{l \epsilon L_w} \lambda_l^* \, r_l, \quad \boldsymbol{x}^* \geq 0, \quad \boldsymbol{\lambda}^* \geq 0 \tag{12}$$

An optimal value solution of $\{x^*, r^*, \lambda^*\}$ to the above constraints can be found by using NUM schemes such as the ones defined in [34] [35] [29] [36] [28]. For all of these NUM schemes, again queue lengths are used to approximate scaled versions of Lagrange multipliers so that $Q_l \equiv V\lambda_l$ where V is a constant to scale queue lengths to the dual variable. By developing a jointly optimal design of congestion controller and link scheduler for the hybrid network, NUM schemes can realize a solution using concepts such as convex analysis, duality theory and stochastic approximation tools.

2.6 A New Direction

As we saw in the previous section, queue length based NUM schemes employ queue-lengths as scaled Lagrange dual variables. Remember from the intro that the NUM paradigm was created to understand and re-engineer Internet protocols, to essentially look backward and apply numerical analysis as a proof for why the scheme was working. With that understanding, new designs could be created that compensated for any shortfalls in the particular optimization. In its first practical application, TCP congestion control was shown to solve a NUM problem [8]. Even though solutions are realizable using NUM schemes, the implementation is not very feasible because the shortcoming of NUM schemes is that the source rate controllers do not fit well for TCP congestion control. As was shown in [10] [28], the source rate controllers use explicit message passing of aggregate queue-lengths ($\sum_{l \in L(s)} Q_l$) from the network routers. This action creates overhead thereby lowering the available throughput and increasing delay

which may not be tolerated in some quality of service (QoS) applications like Voice Over Internet Protocol, VOIP. In addition, message passing violates the scalability of TCP's end to end congestion control. But more problematically, those proposed rate controllers cannot be easily implemented by a TCP window controller. Specifically, mapping the window size to the source rate can be very difficult. Consider Figure 1 as validation of the difficulty of mapping where x_s is throughput and w_s is window size. Therefore, NUM schemes are not a solution that is easily implemented or desirable.

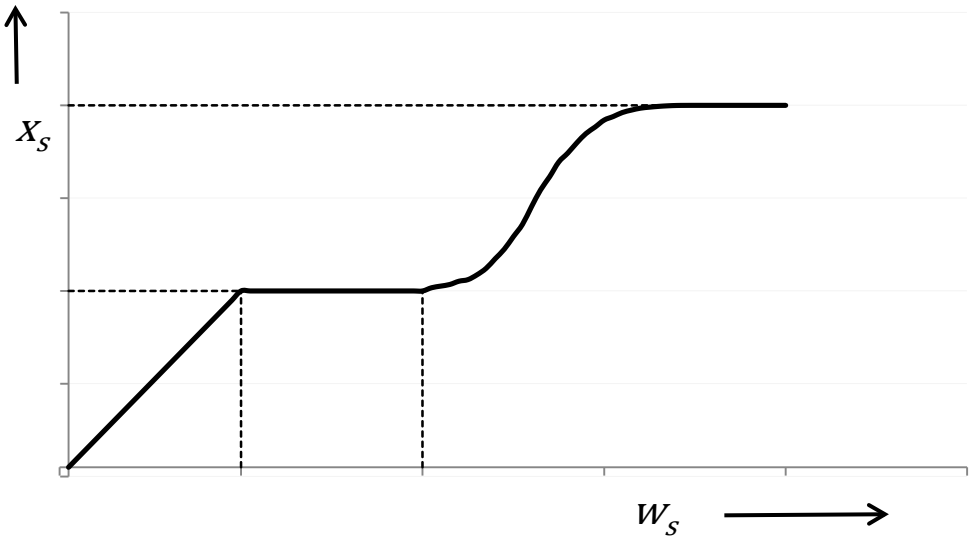


Figure 1 - Difficulty of mapping

Like queue-lengths in NUM based schemes, queue-delays in queue-delay based TCP schemes function as scaled Lagrange multipliers. While both are scaled with respect to link capacity, they are not interchangeable or equivalent. The key difference is that the primal updates in NUM are explicitly performed by the source rate controller

while in queue-delay based TCP schemes they are implicitly determined by the window controllers through estimation of the aggregate queue delay for each flow. If the minimum measured round trip return time, RTT, can be assumed to be the RTT with no delay, then the difference between the average RTT and the minimum RTT is the aggregate queue delay per flow [4] [7] which gives a measure of the congestion of the network. Obviously, the larger the difference the larger the delay on the network but the rate of change also anticipates whether the congestion is increasing or decreasing. Since this congestion measure is determined at the source with no explicit message passing, then the scalability of TCP is preserved. Piggy-backing on the success of Mo-Walrand/FAST-TCP's development of window based congestion controller for wired networks [7] [18], a modification is necessary to accommodate the differences in wireless networks where link capacities are variable and coupled.

2.7 Summary

As we have seen with heuristic TCP schemes, optimal solutions cannot be guaranteed because these schemes were built on engineering "hunches" and as such do not have the systematic mathematical development behind their solutions. On a somewhat related note, a theoretically optimized solution is not robust if the optimization variable, in our case Lagrange multipliers, cannot be mapped to physical entities in a non-unique way. Considering the optimization problem put forth in (6), clearly queue-delay is the better choice for Lagrange multipliers since the solution stays within the

constraints of TCP scalability and facilitates a much easier implementation. The goal of our research then is to combine the strengths of network optimization theory and practical Internet designs to develop non-standard implicit primal-dual solvers that enable end-to end window adjustment schemes for TCP congestion control that will function in a wired and wireless hybrid network. The NUM paradigm and Mo-Walrand/TCP-FAST have provided partial frameworks but neither has presented a complete solution that is readily deployable, scalable and optimally provable as well. Our work builds on their foundation and combines their approaches to develop a TCP window control oriented (cross-layer) network optimization over wireless links.

CHAPTER 3: PROPOSED QUIC & LINK SCHEDULING SCHEMES

3.1 Introduction

In the previous chapters, the need was shown for a joint optimized congestion control and link scheduling scheme. This chapter will detail our proposed schemes and provide analytical results in the network fluid model.

3.2 Congestion Control

The 7 layers of the Open Systems Interconnection (OSI) model have been a useful concept since their development in the mid-1980s. One of the goals of this abstract model was that each layer in a communication system could be independently and asynchronously modified (as long as the protocols remained intact at the layer handoffs) and the system would still function as a whole. Similarly, TCP congestion control at the transport layer can be broken down into 4 components – estimation, window control, data control and burstiness control.

The estimation component provides each source estimation of input parameters (primarily $BaseRTT_s$ and $AvgRTT_s$ where $BaseRTT_s$ is the minimum RTT_s observed so far and $AvgRTT_s$ is the average RTT for each source) to the other 3 decision making components. As explained in the TCP section of this thesis, once data is sent the source waits for an ACK. Since the packet is time stamped, the source can easily determine the current RTT_s. From here, the $AvgRTT_s$ can be calculated and compared to $BaseRTT_s$. If the new $BaseRTT_s$ is lower than the stored value, then the stored value is replaced with the new $BaseRTT_s$. The $AvgRTT_s$ is smoothed using a low pass filter such that:

$$AvgRTT_s \leftarrow \frac{255}{256} \times AvgRTT_s + \frac{1}{256} \times RTT_s$$

Queueing delay is calculated as $q_s = AvgRTT_s - BaseRTT_s$ to smooth any irregular measurements. When 3 duplicate ACKs or a timeout is received, the slow start and/or fast recovery of the standard TCP scheme is initiated as detailed in the Chapter 2 TCP section of this thesis.

While standard TCP like TCP Reno has 2 states of window control depending on whether or not the source is in loss recovery or Additive Increase portion of AIMD [41], QUIC-TCP (like FAST-TCP) has one algorithm to adjust window size. For QUIC-TCP, window control is parameterized by a constant $\rho \in [0, 1]$ and the window size per flow s, w_s, is adjusted as follows:

$$w_s \leftarrow w_s - \kappa \frac{BaseRTTs}{AvgRTTs} w_s^{(-2\rho+1)} \left(w_s - \frac{BaseRTTs}{AvgRTTs} w_s - p_s \right) \qquad (13)$$

where κ is a positive step size. For the remaining two components of congestion control, data control determines which packets to transmit and burstiness control determines when to send them.

3.3 Wireless-link Scheduling

The 7 layer OSI model allows for independent development and asynchronous modifications for each layer. The previous section focused on layer 4, the transport layer. Link scheduling is confined within layer 2. Any changes to layer 2 or layer 4 must operate in harmony with any other configurations of layer 2 or layer 4. However, to achieve a globally optimal solution to the problem of (6) both congestion control from the previous section AND the link scheduling detailed below must operate in harmony.

For a high data rate (HDR) system like CDMA EV-DO 2000 or IEEE 802.16 WiMax, upload transmission is random and asynchronous. Download transmission, on the other hand, is synchronous in small enough time slots (typically 1.67 ms) so that the channel quality can be assumed to be constant over one time slot. Under the right conditions, the channel can even be constant over several time slots.

The access point performs the wireless-link scheduling for the system by separating the queue for each receiver and functioning as a router to sort each packet to the appropriate queue. First, it must read the queue length, $QueLEN_l[n]$ then calculate the average rate, $AveR_l[n]$. To smooth out variation, a low pass filter is used for each wireless link per slot n. At each time slot a new decision is made based on the channel conditions of each receiver. The goal of the scheduler is to take advantage of channel variations to give higher priority to users that temporarily have a better channel condition. The link-scheduler is constrained by the following:

$$\boldsymbol{r}^*[n] = arg \max_{r \in \mathcal{R}(h[n])} \sum_{l \in L_w} \frac{QueLEN_l[n]}{AveR_l[n]} r_l \tag{14}$$

If $R_l[n]$ is the rate capacity that link l can currently support for $\forall l \in L_w$, then to optimize, the access point selects the link with the highest $\frac{QueLEN_l[n]}{AveR_l[n]} R_l[n]$ per slot n. The goal of a proportionally fair scheduler is to maximize the long term throughputs of the users relative to their average channel conditions. Conversely, our scheduler becomes a modified largest weighted delay first (M-LWDF) algorithm, where channel throughput can be optimized as a function of both current channel conditions and the state of the links queues while ensuring network stability [44].

3.4 Network Fluid Model

In Mo-Walrand's seminal work on end-to-end congestion control [18], they developed a network fluid model. By tweaking their model to fit our conditions, we can build a new model to garner a better understanding of joint congestion control (13) and link scheduling design (14). As with Mo-Walrand, packets are infinitely divisible and small. If we assume that processing delays are negligible, then $\boldsymbol{d} := \{d_s, \forall s\}$ can collect the fixed round-trip propagation delay for all sources (fixed round-trip propagation delay is the minimum delay for a packet from when it leaves the source until the ACK is returned.) The RTT_s for a packet is defined as $RTT_s = d_s + q_s$. If there is no congestion on the network then $q_s = 0$ and hence $RTT_s = d_s$. If there is any amount of congestion then $q > 0$ and $RTT_s = d_s + q_s$. We can let $\boldsymbol{q} := \{q_l, \forall l\}$ collect the round-trip queueing delays for links $l \in L$. We can let $\boldsymbol{w} := \{w_s, \forall s\}$ collect the window sizes for all sources $s \in S$. If we then let $q^s := \sum_{l \in L(s)} q_l$ be the aggregate queueing delays for each flow rate, we can establish the following fluid model identities [18] for x_s(source rates), w_s(window sizes) and q_l (queueing delays):

$$x_s(d_s + q^s) = w_{s,} \quad \forall s \in S \tag{15}$$

$$q_l\left(c_l - \sum_{s \in S(l)} x_s\right) = 0, \qquad \forall l \in L_f \tag{16}$$

$$\sum_{s \in S(l)} x_s \leq c_l, \quad \forall l \in L_f \tag{17}$$

$$q_l \left(r_l - \sum_{s \in S(l)} x_s \right) = 0, \quad \forall l \in L_w \tag{18}$$

$$\sum_{s \in S(l)} x_s \leq r_l, \quad \forall l \in L_w \tag{19}$$

$$\boldsymbol{r} \in \bar{\mathcal{R}}, \quad \boldsymbol{x} \geq 0, \quad \boldsymbol{q} \geq 0. \tag{20}$$

If we examine (15) we can see that window size w_s is equal to the total number of packets in transit $x_s d_s$ plus the total number of packets stored in the buffers from the source to the receiver $x_s q^s$. The link constraints seen earlier in (6) are rewritten as (17 & 19). Since we defined the packets as infinitely divisible and small, equation (16 & 18) show us that when the queue size is zero (and thus the queueing delay is zero) then the capacity of the link (c_l or r_l) must exceed the aggregate rate passing through the link.

Previous NUM schemes mapped Lagrange multipliers to queue lengths. Optimization strategies were developed for queue length based congestion controllers and wireless link schedulers. Essentially, the source rate controller at the transport layer determined the traffic flowing into the network while a Max Weight scheduler at the link level determined the wireless link capacities [10] [28] [30] [35]. Our goal is to develop a

Max Weight link scheduler utilizing queueing delays instead of queue lengths for the weighting of each user. Further, queueing delays should fulfill the role of Lagrange multipliers.

First proposed in the 1950s, Little's law has been beneficial to the development of queueing theory. While seemingly intuitive at first glance, it took almost 10 years until a proof was published. Little's law states that for stable systems the long term average number of customers is equal to the long term arrival rate multiplied by the average time a customer is in the system. In mathematical terms it's written as $L = \lambda w$ or $w = L/\lambda$. For our proposed system, we can identify w = average queueing delay, L = average queue length and λ = average rate.

For each wireless link at the access point, if we can maintain the average rate, $AveR_l[n]$, then when we revisit the link scheduling strategy (14) we note that it is dependent on a "factor" multiplied by the user rate, r_l. From Little's law above, we see that the factor is an "instantaneous" delay, $\frac{QueLEN_l[n]}{AveR_l[n]}$. Further, if we build a deterministic fluid model as in [29], then with the appropriate time scaling, a fluid limit argument can be applied such that the factor in our stochastic system, $\frac{QueLEN_l[n]}{AveR_l[n]}$, can be replaced by the average delay q_l. This implies that the link scheduler is actually a queueing-delay based scheduler which maximizes the total result of queueing delay multiplied by capacity for all links per fading state $\boldsymbol{h}$:

$$\boldsymbol{r}^*(\boldsymbol{h}) \in \ arg \max_{r \in \mathcal{R}(\boldsymbol{h})} \sum_{l \in L_w} q_l \, r_l.$$

Given that the wireless scheduler is a random process, we can consider the ergodic capacities of the wireless links satisfying the following:

$$\boldsymbol{r}^* := \mathbb{E}[\boldsymbol{r}^*(\boldsymbol{h})] \in \ arg \max_{r \in \mathcal{R}} \sum_{l \in L_w} q_l \, r_l. \tag{21}$$

To smooth out any irregularities (measurement error, etc.) we had earlier used a low pass filter to yield $AvgRTT_s$. This value is similar to Brakmo & Peterson's current round trip time $D_s(t)$ so let's use new notation $\bar{d}_s$. Since the $BaseRTT_s$ is the minimum RTT observed so far, it will be very close (if not equal to) the propagation delay d_s. With these two variables we can estimate the aggregate queueing delay q^s, by the following $\bar{d}_s = \ d_s + q^s$. Substituting $\bar{d}_s$ & d_s back into (13) for window adjustment we get:

$$\Delta w_s = \ -\kappa \frac{d_s}{\bar{d}_s} w_s^{-2\rho+1} (w_s - \frac{d_s}{\bar{d}_s} w_s - \ p_s)$$

Recall from (15) that $x_s = \ \frac{w_s}{(d_s + q^s)} = \ \frac{w_s}{\bar{d}_s}$. We can rewrite the equation above as:

$$\Delta w_s = -\kappa \frac{d_s}{\bar{d}_s} w_s^{-2\rho+1} (w_s - x_s d_s - p_s)$$

We can now define a new variable $v_s = w_s - x_s d_s - p_s$ for $\boldsymbol{v} := \{v_s, \forall s\}$. Now when we reconsider the fluid model, we see that the window updates follow the ordinary differential equation:

$$\frac{d}{dt} w_s = -\kappa \frac{d_s}{\bar{d}_s} w_s^{-2\rho+1} v_s, \quad \forall s \tag{22}$$

3.5 Summary

Equation (22) satisfies our goal of finding an optimization equation where queue delays can act as Lagrange multipliers. The Max Weight scheduler is easy to implement for a one hop network given that there is one main access point that can calculate all the weights for the shared links and schedule accordingly. In a multi-hop network without a single central decision making scheduler, finding a Max Weight scheduling policy is NP-complete as demonstrated in [37].

CHAPTER 4: SIMULATION DESIGN

4.1 Introduction

Ns-2 is an open source program used for networking research. It is a discrete event simulator that is in a continuous state of modification driven by on-going research and development. The joint congestion control (13) and link scheduling (14) schemes were simulated using ns-2 and analyzed using Matlab and Excel. For these simulations Linux based ns-2 was running in a virtual box on a Windows based workstation. This chapter will detail the set-up and the next chapter will focus on the results.

4.2 Topology set-up

In the initial publication of QUIC-TCP [47] a 4-flow 11-node system was simulated to validate QUIC-TCP. The goal of this thesis is to validate a larger topology to show that QUIC-TCP is scalable to larger networks. An 11-flow and 32-node topology was chosen (see Figure 2). For the work in [50] 4 nodes were used for their ns-2 simulations and 6 nodes were used for the analysis in [51]. For the improvements demonstrated in [52] 3 flows were used initially but they did expand the simulation to 20

flows for a larger test with 10 flows being simulated the entire time cycle and the other 10 starting halfway through the simulation. F-TCP [53] varied simulations between 10 to 20 node pairs. The analysis of [54] simulated with just 20 nodes. TCP-Jersey [17], TCP-Westwood [15] and the analysis in [55] did simulate 20 flows; however they simulated limited number of nodes. Therefore, the scale-up from 4 flows with 3 UEs (Users Equipment) to 11 flows with 6 UEs seemed like a reasonable increase to validate the scalability of QUIC-TCP.

For simplicity, the network is comprised of fixed nodes which define the wired backbone and a one hop to a wireless network with six users. This topology can now be considered a simulated IEEE 802.16 WiMax network. The network consists of 32 fixed nodes (N_1-N_{32}), two gateway nodes (GW_1, GW_2), a radio network controller (RNC), a base station (BS) and 6 mobile users (UE_1-UE_6). The BS functions as an access point (AP) for the user equipment (UE_1-UE_6).

4.3 Simulation Set-up

To create a QUIC-TCP agent for simulation, we modified an existing ns-2 module of FAST-TCP to satisfy the new congestion control (13). The wireless links are subject to Raleigh fading with a Doppler frequency of 127.8 Hz. The average signal to noise ratio (SNR) for the 6 UEs (UE_1-UE_6) is 5, 9, 13, 5, 9, and 13 respectively. The BS communicates with the UEs via a shared downlink channel with bandwidth of 10M. The

BS employs eight different convolutional-turbo-coded (CTC) quadrature-amplitude modulations (QAMs) as the Adaptive Modulation and Coding (AMC) modes for down link transmission. (Table 1)

Table 1 - QAMS

Mode *n*	Modulation	CTC Coding Rate	Threshhold (dB)
1	QPSK	1/12	1.03
2	QPSK	1/8	3.01
3	QPSK	¼	6.13
4	QPSK	½	10.24
5	16-QAM	½	16.11
6	16-QAM	¾	20.67
7	64-QAM	2/3	25.13
8	64-QAM	¾	27.77

Since SNR is inversely proportional to the distance between the BS and UE, the modulation order and code rate will decrease as the distance increases. Given the estimated average SNR and the thresholds defined in Table 1, the proper AMC mode provides the flexibility to adjust each UE to its average signal strength. These values are fed back to the BS for fast link adaptation and scheduling [47].

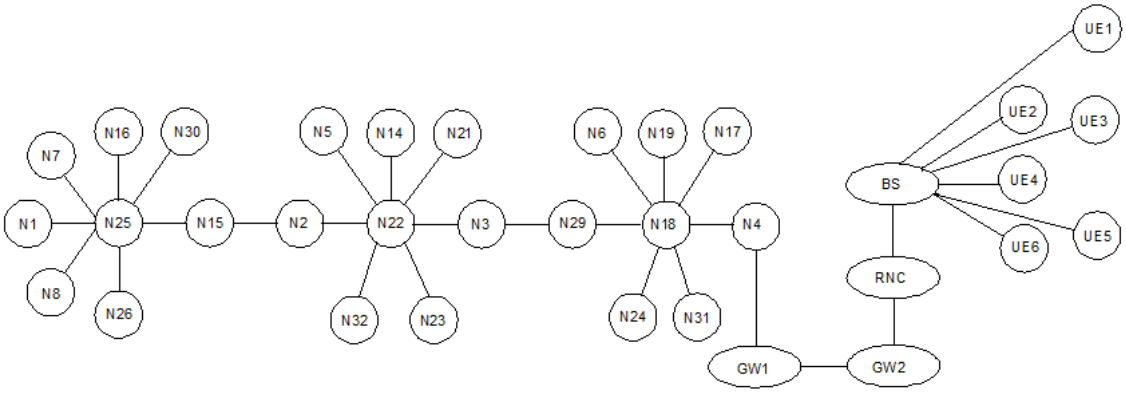
Eleven TCP connections are established in the network to carry the 11 FTP flows. All FTP flows start at the same time. Four different window control schemes TCP-Reno, TCP-Vegas, FAST-TCP and QUIC-TCP are examined in the presence of feedback delays where $\rho = 0.5$ unless otherwise noted. Step size is adjusted so that QUIC-TCP has a reasonable response and is not too sluggish. The default value of $\kappa = 0.5$ is used for both FAST-TCP and QUIC-TCP unless otherwise noted. The link schedulers evaluated are Round-Robin (RR), Maximum Signal to Noise Ratio (MSNR) which favors the link with the highest SNR, Queue Length (Qlen) from prior NUM solutions [28] [30], and Queue Delay (Qdelay) based scheduler developed for QUIC-TCP. While the uplink of a wireless UE is asynchronous, the downlink from the BS is governed by Time Division Multiple Access (TDMA) with slots of 10 ms.

Due to the large sizes of these networks, simulation time of 500 seconds was chosen so that the actual simulation run time would not be excessive. Packet size is 100 bytes. The queue limit of the fixed nodes was set to 2000 but the BS had to be increased to 3000 due to oscillations during simulation. The target queue size (p_s) of each flow is set to 200 packets.

Since our objective is to optimize the utility function in (6), we expect that QUIC-TCP with queue delay link scheduling scheme satisfying (13) and (14) to perform optimally. QUIC-TCP is evaluated relative to the other schemes and schedulers by comparing the following:

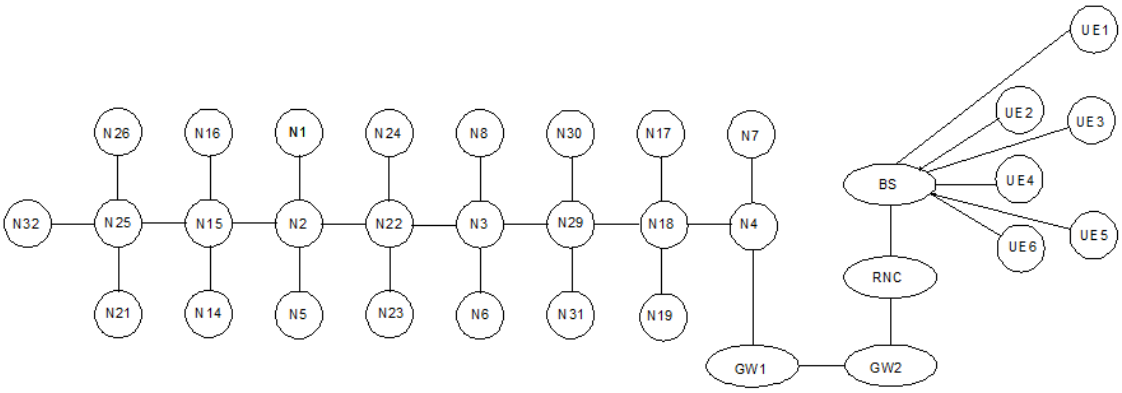
- Jain's index to measure fairness [49] where $F = \frac{(\sum_s x_s)^2}{|S| \sum_s (x_s)^2}$ and $|S|$ is the number of users
- Aggregate throughput
- The object of (6), $\sum_s p_s \log x_s$

A symmetric topology (named Fair) was designed and evaluated, see Figure 2. In contrast, another 32-node network (named Random) was designed where the nodes were added randomly to see the influence on the criteria above, Figure 3.



ftp0: $N_1 \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_1$
ftp1: $N_8 \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_2$
ftp2: $N_7 \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_3$
ftp3: $N_{16} \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_4$
ftp4: $N_{26} \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_5$
ftp5: $N_{30} \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_6$
ftp6: $N_5 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_6$
ftp7: $N_{14} \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_{19}$
ftp8: $N_{21} \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_{17}$
ftp9: $N_{32} \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_{24}$
ftp10: $N_{23} \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_{31}$

Figure 2 – 32-Node symmetric topology - Fair



ftp0: $N_1 \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_1$
ftp1: $N_8 \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_2$
ftp2: $N_7 \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_3$
ftp3: $N_{16} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_4$
ftp4: $N_{26} \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_5$
ftp5: $N_{30} \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_4 \rightarrow GW1 \rightarrow GW2 \rightarrow RNC \rightarrow BS \rightarrow UE_6$
ftp6: $N_5 \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_6$
ftp7: $N_{14} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_{19}$
ftp8: $N_{21} \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{18} \rightarrow N_{17}$
ftp9: $N_{32} \rightarrow N_{25} \rightarrow N_{15} \rightarrow N_2 \rightarrow N_{22} \rightarrow N_{24}$
ftp10: $N_{23} \rightarrow N_{22} \rightarrow N_3 \rightarrow N_{29} \rightarrow N_{31}$

Figure 3 – 32-Node randomly added topology - Random

4.4 Summary

We have defined a new congestion control algorithm QUIC-TCP that bridges the gap between NUM schemes and heuristic design for hybrid wired/wireless networks. QUIC-TCP satisfies the weighted proportionally fair utility function defined in (6). Unlike AIMD, QUIC-TCP is an equation based algorithm which allows the network to move quickly towards equilibrium when it is far away and slow down as it converges towards equilibrium. Chapter 5 will simulate the performance of this scheme and queue delay based scheduler.

CHAPTER 5: SIMULATION

5.1 Introduction

Chapter 4 defined the parameters of the ns-2 simulations. Chapter 5 will show the results of the simulation. Initially, the Fair topology was simulated for RR and the different TCP schemes were compared to see the overall responsiveness of each scheme and to see which system stabilized better at equilibrium. That analysis is reviewed followed by a review of both topologies.

5.2 Overall Responsiveness and Equilibrium Stability

RR link scheduling was chosen and the 4 TCP schemes were evaluated for responsiveness and stability at equilibrium, Figures 4 to 7. The wired portion of the topology looks reasonable for each scheme, so my analysis will focus on wireless. Reno moves towards equilibrium using AIMD but a loss event at about 315 seconds seriously impacts the throughput. This is the beginning of "sawtooth" like performance due to AIMD and would continue to occur if the simulation were extended longer than 500 seconds. Vegas, on the other hand, has a somewhat sluggish response. This is mainly

driven by the fact that the lower bounds is set to 200 packets and the upper bounds set to 203 packets. Vegas then ramps up linearly regardless of how far away the current value is from equilibrium. Hence, Vegas has a slow response. Both FAST-TCP and QUIC-TCP reach equilibrium quickly with no overshoot. You can see that they accelerate quickly when far from equilibrium and slow down as they approach equilibrium as expected from the analysis earlier in this thesis. In addition, we observe that FAST-TCP and QUIC-TCP produce similar results which we expect when $\rho = 0.5$ for QUIC-TCP. We expect FAST-TCP and QUIC-TCP to produce similar results throughout the analysis. With these observations, we move into a further analysis.

FAIR

Reno RR

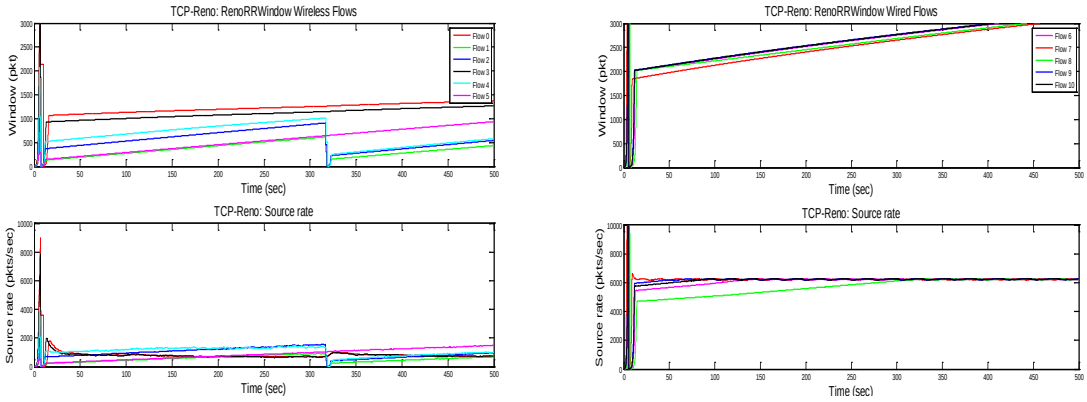


Figure 4 – Reno RR, Fair topology

Vegas RR

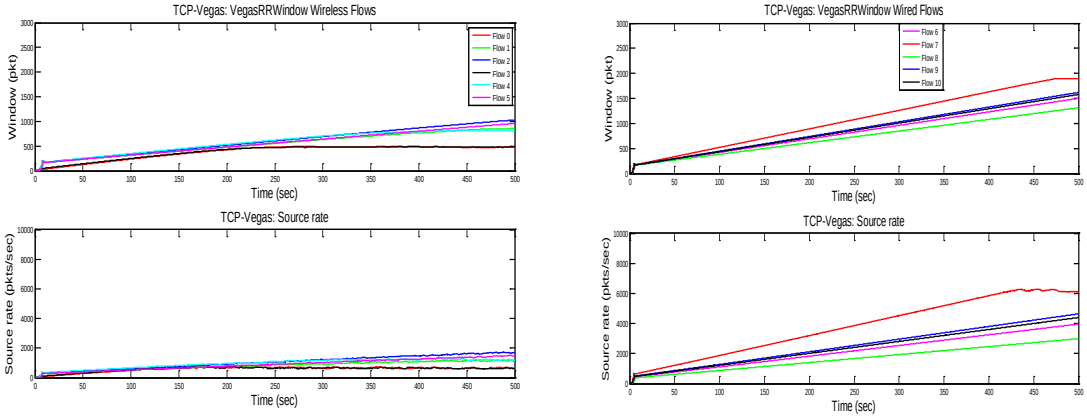


Figure 5 – Vegas RR, Fair topology

FAST RR

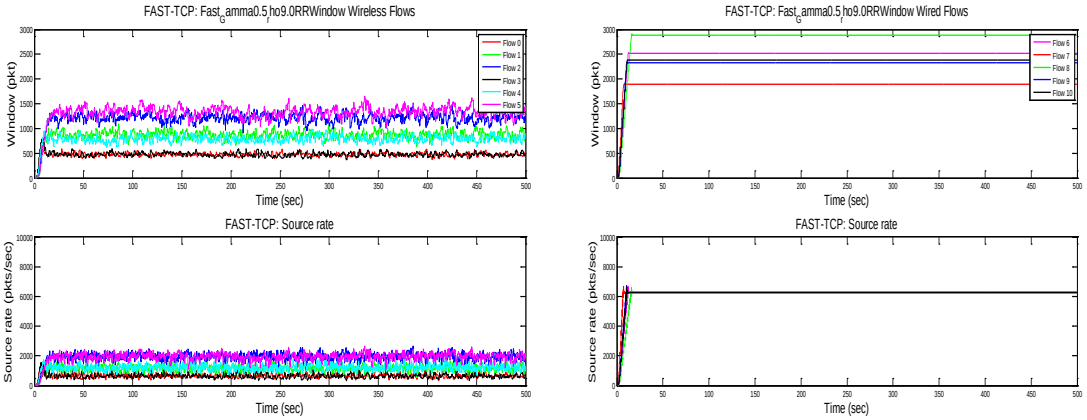


Figure 6 – FAST-TCP RR, Fair topology

QUIC RR

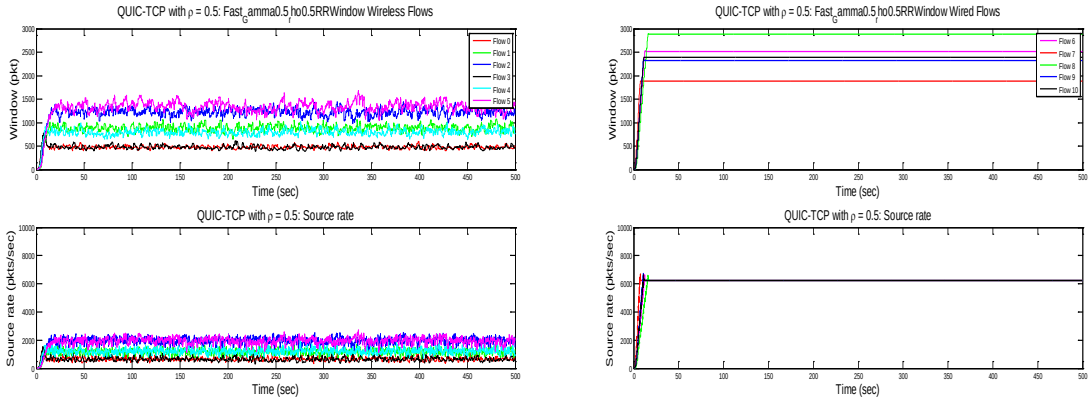


Figure 7 – QUIC-TCP, Fair topology

5.3 Analysis of Fair Topology

With the parameters established in Chapter 4, first the symmetrical topology named Fair was simulated and evaluated with the 4 TCP schemes and 4 link schedulers. Figure 8 shows the results of Jain's Index.

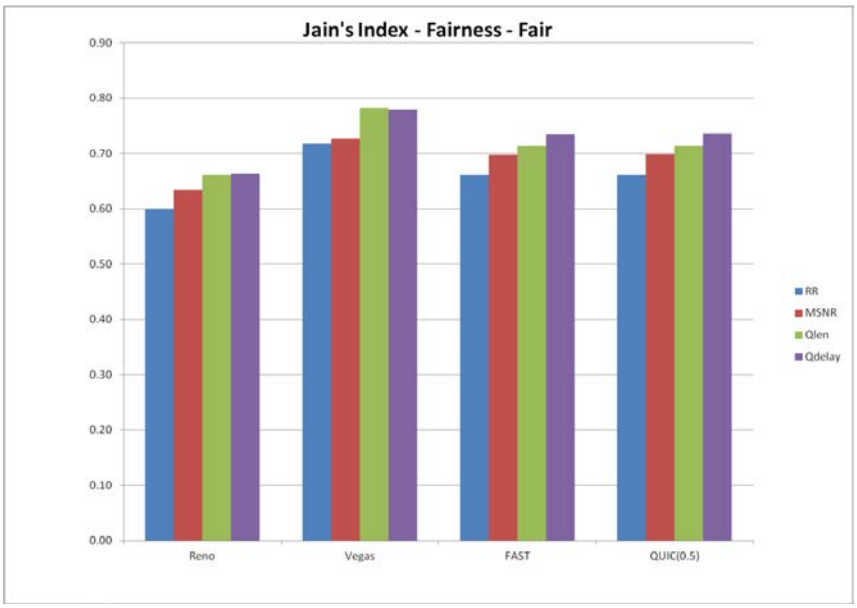


Figure 8 – Jain's index by scheme by scheduler

Qdelay produces the fairest results for 3 out of the 4 schemes. In addition for QUIC-TCP, we observe the following results: $F_{RR} = 0.66$, $F_{MSNR} = 0.70$, $F_{Qlen} = 0.71$, $F_{Qdelay} = 0.74$. Surprisingly, Vegas is slightly more fair than the other schemes. Reno is about 0.10 less fair than Vegas regardless of which schedulers are compared. If we compare the link scheduling Qdelay, Vegas has Jain's index $F = 0.78$ with FAST-TCP and QUIC-TCP have $F = 0.74$ and Reno lagging behind at $F = 0.64$.

Observing Aggregate throughput next in Figure 9, we see that Vegas sacrifices throughput for fairness. This is driven by the sluggish response of Vegas observed in Chapter 4.1. Vegas throughput falls between 15.0 and 20.0 packets/msec while the other 3 schemes are around 20 packets/msec more. For QUIC-TCP, Qdelay is 41.5 packets/msec compared to RR at 38.1 (8% improvement). Qdelay also approaches MSNR (42.8 packets/msec) and Qlen (43.4 packets/msec).

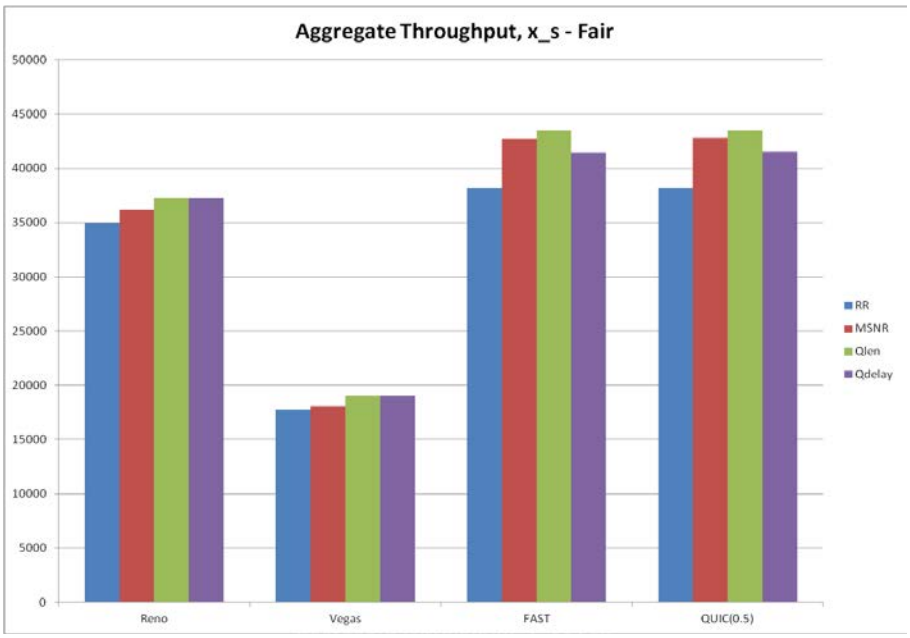


Figure 9 - Aggregate throughput by scheme by scheduler

Finally, we take a look at our last criteria the object of the optimization utility function from (6) $\sum_s p_s \log x_s$ which can be seen in figure 10. As expected, QUIC-TCP performs better than Reno or Vegas with QUIC-TCP queue delay having the best results. Vegas performs the worst due to its sluggish throughput, it measures around 6800 to 7000. For QUIC-TCP, MSNR has the lowest value compared to Qdelay 7637 (4% improvement). As expected, QUIC-TCP provides the most optimal results of (6). In addition, it demonstrates a nice balance between fairness and aggregate throughput for the Fair topology.

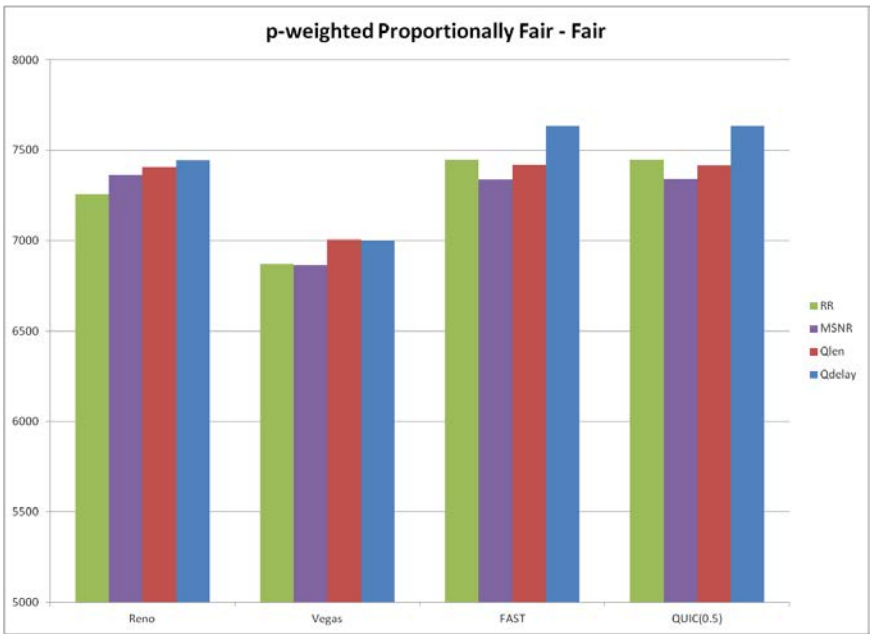


Figure 10 – p-weighted proportionally fair by scheme by scheduler

5.4 Analysis of Random Topology

While simulation of a symmetrical topology is interesting, it is not a very real world example. Therefore, the Random topology (Figure 3) was developed then simulated with the same 4 schemes and schedulers as Fair was simulated in Chapter 5.3. The same criteria are used to evaluate the performance. Figure 11 shows the results of Jain's Index. In the Random topology the performance of FAST-TPC and QUIC-TCP are better than Reno or Vegas and show a significant improvement when compared to Fair. In addition, Qdelay outperforms the other schedulers. Qdelay has $F = 0.74$ compared to RR at $F = 0.66$ (10% improvement), MSNR at $F = 0.70$ (5.4% improvement) and Qlen at $F = 0.71$ (4% improvement).

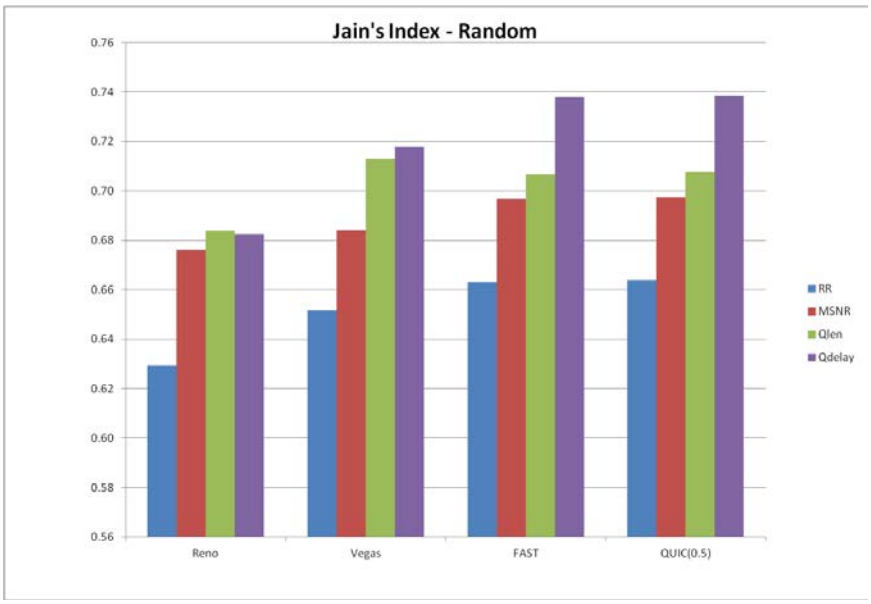


Figure 11 – Jain's index by scheme by scheduler

In Figure 12, aggregate throughput is compared and again QUIC-TCP outperforms Reno or Vegas. Vegas' performance again disappointments for the same reason as Fair. Specifically for QUIC-TCP, we see the same pattern from Fair where MSNR and Qlen are slightly higher than Qdelay.

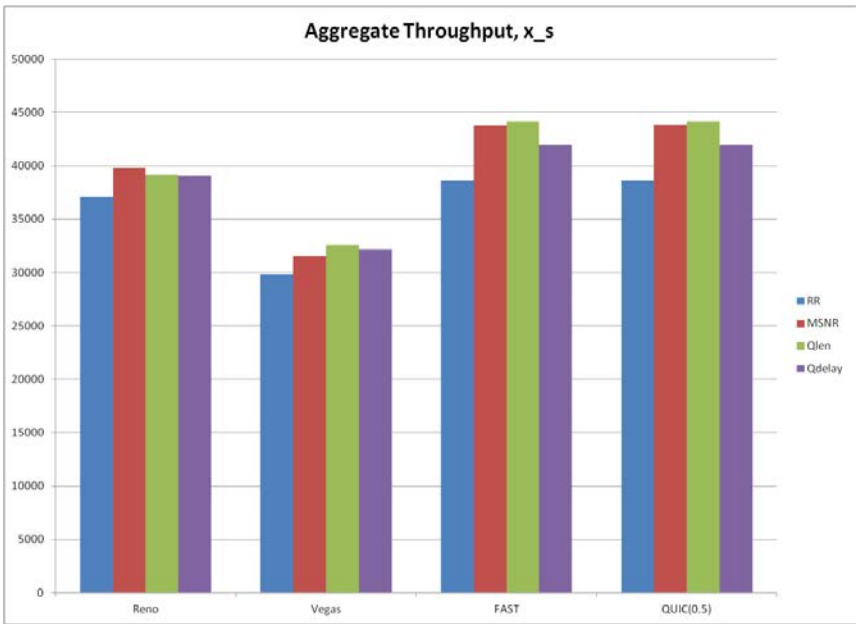


Figure 12 - Aggregate throughput by scheme by scheduler

Finally, the optimization object $\sum_s p_s \, log \, x_s$ is compared in Figure 13. QUIC-TCP shows an improvement compared to the other schemes, especially queue delay, however the results are somewhat disappointing. This may be caused by the fact that the throughput of the wired flows is significantly larger than the throughput of the wireless flows sometimes by a factor as large as 30:1.

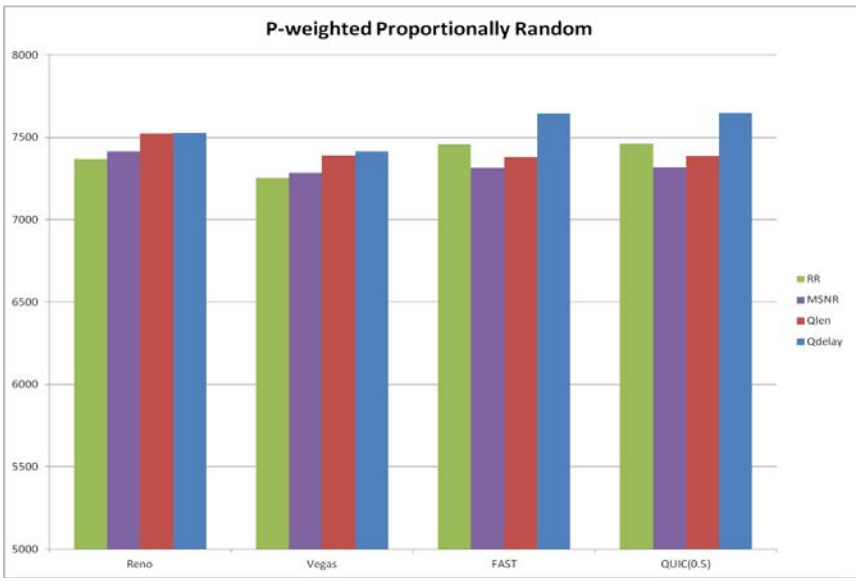


Figure 13 – p-weighted proportionally fair by scheme by scheduler

This disparity between wired and wireless distorts the calculations muddying the results. If we separate the wired flows from the wireless flows and recalculate the 3 criteria, the picture changes. The wired results within each scheme are very similar – Jain's index is typically 1 – so let's not consider them at this point. However, the wireless flows produce the following:

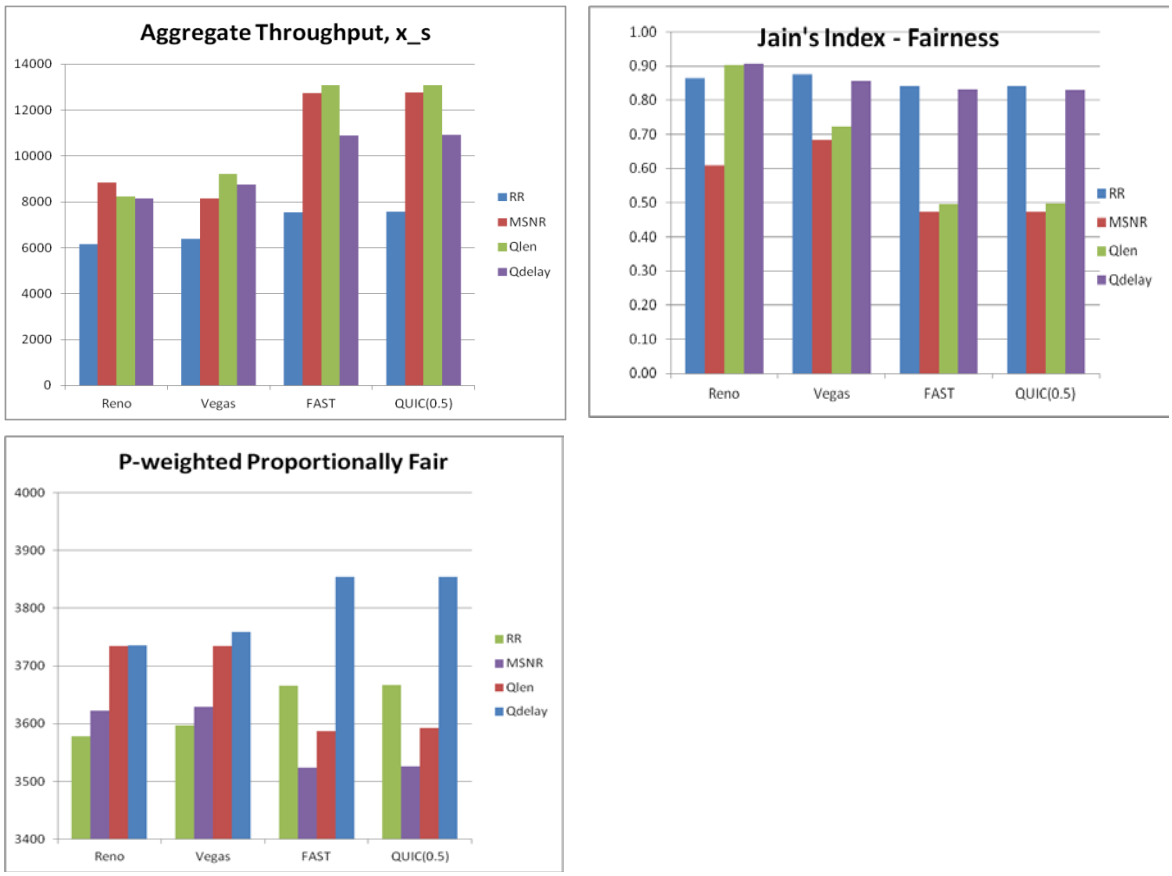


Figure 14 – Random topology wireless flows

If we examine the Aggregate throughput, we see that FAST-TCP and QUIC-TCP perform much better than Reno and Vegas as expected. Curiously, Qlen and MSNR perform better than Qdelay. However, inspection of Jain's Index shows that Qlen and MSNR are not very fair. For MSNR, this is driven by the scheduler preferring the flow with the highest SNR. As a result of choosing SNR of 5, 9, 13, 5, 9, 13, most of the flow goes through the 2 flows with the highest SNR (the data for the QUIC-TCP flows is 57,394; 496,939; 2,607,584; 67,977; 499,094; and 2,648,659). Qlen shows a similar pattern. If we examine the 3rd criteria, the object of the optimization problem posed in (6), we see that QUIC-TCP with the scheduler MSNR and Qlen does not solve (6) well.

We also see that QUIC-TCP queue delay is a significant improvement over the other schemes and schedulers. QUIC-TCP's value is 3854 compared to 3667 for RR (4.8% improvement), 3592 for Qlen (6.8% improvement) and 3526 for MSNR (8.5% improvement). Given the balance between throughput and fairness as well as the performance of the solution to (6), we can see that QUIC-TCP is a combination we would choose.

5.5 Summary

Two larger topologies, Fair and Random, were simulated. QUIC-TCP queue delay demonstrated a nice balance between aggregate throughput and fairness compared to the other schemes and link schedulers. In addition, QUIC-TCP queue delay showed the optimal results of the log utility function from (6) as expected. Further, since the aggregate flow of the wired flows is so much larger than the wireless flows, when the wireless flows are separated out, the results are more prominent in favor of QUIC-TCP queue delay.

CHAPTER 6: CONCLUSION

We have described a readily deployable joint TCP congestion control and wireless link scheduling for a one hop mobile wireless network. A class of QUIC-TCP algorithms was developed to optimize the log utility function. QUIC-TCP utilizes window-based congestion control schemes based on primal-dual solvers without explicit feedback from the routers in a coupled wireless link environment. Queueing delay and packet loss are used as a congestion signal for this equation based scheme which means QUIC-TCP moves quickly towards equilibrium when it is far away and slows down when it is close to equilibrium.

A theoretical analysis was shown where feedback delays were assumed to be zero in the fluid model. In contrast, for the simulation feedback delays were present and the network demonstrated stability and convergence. To simplify both the analysis and the simulation, a last hop wireless network was used; however, both the analysis and simulation could apply to a multi-hop network if an AP coordinates the network and the wireless links maintains a convex capacity region. This thesis builds on the work of [47] and demonstrates that QUIC-TCP applies to larger networks as well. Future work will be to develop an experimental model to test QUIC-TCP in the physical world.

REFERENCES

1. "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2012–2017," *Cisco White Paper*, Feb. 6, 2013, Feb. 26, 2013 http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.html
2. I. Marsic, *Computer Networks Performance and Quality of Service*, Newark, NJ, Rutgers University, 2010 <http:/www.ece.rutgers.edu/~marsic/books/QoS/>
3. O. Ait-Hellal and E. Altman, "Analysis of TCP Vegas and TCP Reno," *Communications, 1997. IEEE ICC '97*, Montreal, Que. June 1997, pp 495-499, vol. 1
4. C. Jin, D. Wei, S. Low, G. Buhrmaster J. Bunn, Hyojeong D. Choe, R. Cottrell, J. Doyle, W. Feng, O. Martin, H. Newman, S. Ravot, and S. Singh, **"**FAST TCP:From Theory to Experiment,*" IEEE Newt.*, vol. 19, no. 1, pp 411, Jan-Feb 2005.
5. H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Trans. Netw.*, vol. 5. no. 6, pp. 756–769, Dec. 1997.
6. Y. Tian, K. Xu, and N. Ansari, "TCP in wireless environment: Problems and solutions," *IEEE Commun. Mag.*, vol. 43, pp. 27–32, Mar. 2005.
7. C. Jen, D. Wei, and S. Low, "FAST TCP Motivation, Architecture, Algorithms, Performance," *IEEE INFOCOM 2004, Twenty-third Annual Conference of the IEEE Computer and Communications Societies*, Vol. 4, 2004, pp 2490-2501
8. S. Low, "A duality model of TCP and queue management algorithms," *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 525–536, Aug. 2003.
9. R. Srikant, *The Mathematics of Internet Congestion Control*, Cambridge, MA: Birkhauser, 2004.
10. M. Chiang, S. Low, A. Calderbank, and J. Doyle, "Layering as optimization decomposition," *Proc. IEEE*, vol. 95, no. 1, pp. 255–312, Jan. 2007.
11. Y. Yi and M. Chiang, "Stochastic network utility maximization: A tribute to Kelly's paper published in this journal a decade ago," *Eur. Trans. Telecommun.*, vol. 19, no. 4, pp. 421–442, June 2008.
12. L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE Journal on Selected Areas in Communication*, Vol. 13, No 8, October 1995, pp. 1465-1480, Oct. 1995
13. S. Floyd, "High speed TCP for large congestion windows," RFC 3649, Dec. 2003.

14. T. Kelly, “Scalable TCP: Improving performance in highspeed wide area networks,” *ACM SIGCOMM Comput. Commun. Review*, vol. 33, no. 2, Apr. 2003.
15. S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, and R. Wang, “TCP Westwood: Bandwidth estimation for enhanced transport over wireless links,” in *Proc. ACM Mobicom*, pp. 287–297, July 2001.
16. C. Fu and S. Liew, “TCP Veno: TCP enhancement for transmission over wireless access networks,” *Proc. EUROCON'2001, Int. Conf. Trends in Communications, pp. 202-209, 2001*
17. K. Xu, Y. Tian, and N. Ansari, “TCP-Jersey for wireless IP communications,” *IEEE J. Sel. Areas Commun.*, vol. 22, no. 4, pp. 747–756, May 2004.
18. J. Mo and J. Walrand, “Fair end-to-end window-based congestion control,” *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.
19. V. Gambiroza, B. Sadeghi, and E. Knightly, “End-to-end performance and fairness in multihop wireless backhaul networks,” in *Proc. ACM Mobicom*, Philadelphia, PA, Sept. 2004.
20. S. Low, L. Peterson, and L. Wang, “Understanding TCP Vegas: A Duality Model,” *Journal of the ACM*, Vol. 49, No. 2, pp. 207–235, March 2002.
21. M. Leconte, J. Ni, and R. Srikant, “Improved Bounds on the Throughput Efficiency of Greedy Maximal Scheduling in Wireless Networks,” *Proc. ACM MobiHoc*, pp165-174, May 2009
22. F. LeFevre and G. Vivier, **“**Understanding TCP's behavior over wireless links,” *Proc. Commun. and Vehic. Tech.*, 2000 SCVT-200, 2000, pp. 123–30
23. J. Liu and S. Singh, “ATCP: TCP for mobile ad hoc networks,” *IEEE J. Sel. Areas Commun.*, vol. 19, no. 7, pp. 1300–1315, July 2001.
24. *IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface*, IEEE Std 802.16-2001, June 1998.
25. *3rd Generation Partnership Project, Technical Specification Group Radio Access Network; Physical layer aspects for evolved Universal Terrestrial Radio Access (UTRA)*, 3GPP Std. TR 25.814 v. 7.0.0, 2006.
26. P. Bender, P. Black, M. Grob, R. Padovani, N. Sindhushyana, and A. Viterbi, “CDMA/HDR: A bandwidth efficient high speed wireless data service for nomadic users,” *IEEE Commun. Mag.*, vol. 38, no. 7, pp. 70–77, July 2000.
27. J. Sundararajan, D. Shah, M. Medard, M. Mitzenmacher, and J. Barros, “Network coding meets TCP,” in *Proceedings of the IEEE*, Vol. 99, No. 3, March 2011, pp. 490–512, 2011.
28. Y. Yu and G. Giannakis, “Joint congestion control and OFDMA scheduling for hybrid wireline-wireless networks,”. in *Proc. IEEE INFOCOM*, pp. 973–981, Anchorage, AK, May 6–12, 2007.
29. A. Stolyar, “Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm,” *Queueing Syst.*, vol. 50, pp. 401–457, 2005.
30. X. Lin and N. Shroff, “The impact of imperfect scheduling on crosslayer rate control in wireless networks,” *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, Apr. 2006.

31. L. Andrew, S. Hanly, and R. Mukhtar, “Active queue management for fair resource allocation in wireless networks,” *IEEE Trans. Mobile Comput.*, vol. 7, no. 2, pp. 231–246, Feb. 2008.
32. M. Chan and R. Ramjee, “TCP/IP performance over 3G wireless links with rate and delay variation,” in *MOBICOM’02*, pp. 1-12, Atlanta, Georgia, Sept. 2002.
33. T. Goff, J. Moronski, D. Phatak, and V. Gupta, “Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments,” in *Proc. IEEE INFOCOM*, vol. 3, pp. 1537–1545, Mar. 2000.
34. F. Kelly and A. Maulloo, and D. Tan, “Rate control in communication networks: Shadow prices, proportional fairness and stability,” *J. Opl. Res. Soc.*, vol. 49, no. 3, pp. 237–252, Mar. 1998.
35. M. Neely, E. Modiano, and C. Li, “Fairness and optimal stochastic control for heterogenous networks,” *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 396–409, Apr. 2008.
36. X. Wang, G. B. Giannakis, and A. G. Marques, “A unified approach to QoS-guaranteed scheduling for channel-adaptive wireless networks”, *Proc. IEEE*, vol. 95, no. 12, pp. 2410–2431, Dec. 2007.
37. L. Tassiulas and A. Ephremides, “Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks,” *IEEE Trans. Autom. Control*, vol. 36, no. 12, pp. 1936–1948, Dec. 1992.
38. S. Haykin and M. Moher, *Modern Wireless Communication*, Upper Saddle River, NJ Pearson Prentice-Hall, 2005 pg 12
39. M. Allman, V. Paxson, and W. Stevens, “TCP congestion control,” RFC 5681, Sep. 2009.
40. S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge Uiversity Press, 2004.
41. W. Stevens, “TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms,” RFC 2001, Jan. 1997.
42. *IEEE Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std 802.11™-2012 (Revision of IEEE Std 802.11-2007), Feb. 2012
43. C. Jin, D. Wei, S. Low, “The Case for Delay-based Congestion Control,” *IEEE*, September 2003.
44. M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, P. Whiting, and R. Vijayakumar, “Providing quality of service over a shared wireless link,” *IEEE Commun. Mag.*, pp. 150–154, Feb. 2001.
45. M. Gast, *802.11® Wireless Networks: The Definitive Guide* O’Reilly, April 2002
46. M. Greis, “The ns Manual (formerly ns Notes and Documentation),” July 13, 2005 Sept 5, 2011 http://www.isi.edu/nsnam/ns/tutorial/
47. X. Wang, Z. Li, and Gao, Na, “Joint TCP Congestion Control and Link Scheduling for Internet with Last-Hop Wireless Links,” *Global Telecommunications Conference (GLOBECOM 2011)*, 2011 IEEE, pp 1-6, 2011

48. L. Liyanage, *TCP Behavior Over Wireless Links: Possible Improvements with Scheduling and Networking Coding,* Master's Thesis, University of Nice Sophia Antipolis, Sophia Antipolis, France, August 2011, retrieved from http://www.cwc.oulu.fi/~lliyanag/Other/Madhusanka_Thesis_Ubinet.pdf
49. R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling*, New York: Wiley, 1991.
50. S. Waghmare, A. Parab, P. Nikose, and S. Bhosale, "Comparative Analysis of different TCP variants in a wireless environment", *2011 3rd International Conference on Electronics Computer Technology (ICECT),* pp 158-162, 2011
51. G. Abed, M. Ismail, and K. Jumari, "A Comparison and Analysis of Congestion Window for HS-TCP, Full-TCP, and TCP-Linux in Long Term Evolution System Model", *2011 IEEE Conference on Open Systems (ICOS),* pp 358-362, 2011
52. N. Parvez and E. Hossain, "Improving TCP Performance in Wired-Wireless Networks By Using a Novel Adaptive Bandwidth Estimation Mechanism", *Telecommunications Conference (GLOBECOM 2004) Vol. 5,* pp 2760-2764, 2004 IEEE
53. J. Sansa-Otim, I. Rai, and J. van der Hulst, "F-TCP: a Delay-based Protocol with Fair Co-existence", *2010 Ninth International conference on Networks (ICN),* pp 34-41, 2010
54. N. Sengottaiyan, R.Somasundaram, S. Arumugam, "A Modified approach for measuring TCP Performance in Wireless Adhoc Network", *2010 International Conference on Computing (ARTCom),* pp 267-270, 2010
55. S. Henna, "A Throughput Analysis of TCP Variants in Mobile Wireless Networks", *2009 Third International Conference on Next Generation Mobile Applications,* Services and Technologies, pp 279-284, 2009