Modeling Heterogeneous Traffic Performance with the 802.11 DCF Access Scheme

Tamir Husain and Christopher A. Wood

No Institute Given

Abstract.

1 Introduction

Wireless networks have become pervasive in modern computer and communication systems. The widespread adoption of mobile and personal computing devices, such as smartphones and tablets, continues to drive commercial investment in high performance wireless networking technologies. Concurrently, the type of traffic traversing these networks has also evolved – divergently – an increasing rate. Every kind of information from textual data to video stream segments now traverses wireless networks. Society's dependence on WLAN technologies such as the 802.11 protocol suites have cultivated a massive amount of analytical and experimental research studying their performance.

To date, a major portion of this research has focused on the 802.11 random access control protocol – the distributed control function (DCF). This work was pioneered by Bianchi's original Markov model for the DCF function [?], which stimulated several advancements to his simplistic model [?] and numerous applications in the field [?,?]. Many of these models and empirical studies represent a network of nodes by a single collision parameter. In particular, there is a lack of work combining these models together beyond a single parameter. Furthermore, many of these works study the performance of the DCF function (with respect to individual and average system throughput) for single types of traffic. True multi-node models that study the interaction of different traffic models have not been studied.

The goal of this work is to determine if the random behavior of the DCF protocol is best suited for all types of traffic, or if the DCF should in some way be tailored to the *type* of underlying traffic being served by the DCF access scheme. To this end, we first present a series of extensions to Bianchi's original DCF Markov model to emulate arbitrary types of traffic. In particular, we provide extensions to support arbitrary packet length and variable packet inter-arrival time. We then instantiate individual instances of these Markov models that emulate the behavior of the DCF access scheme by tailoring each of these parameters, e.g., packet length and interarrival time, using realistic values drawn from these types of modern application traffic. Following this, we combine these model instances to study the performance of the DCF function with respect to heterogeneous traffic in a true multi-node system.

Our experimental results seem to provide evidence that the deterministic nature of multimedia (e.g., store-and-forward) traffic streams obtain higher performance than more nondeteministic (elastic) traffic. Furthermore, our results show that large interarrival time between traffic packets benefits the coexistence of heterogeneous traffic streams.

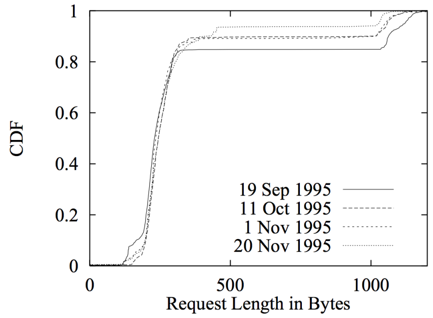
2 Related Work

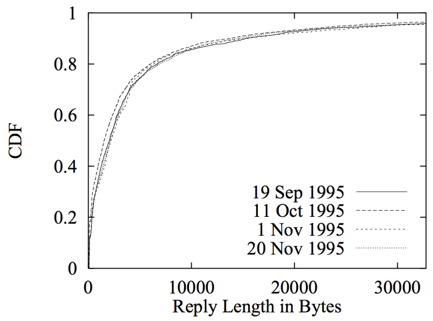
3 Heterogeneous Traffic Models

Today's computer and communication networks are being used to transfer increasingly heterogeneous traffic between parties. In particular, file downloads, standard web browsing, video streaming, and client-server video game traffic are four very common types of traffic that dominate the Internet traffic today. In this section, we describe the characteristics of each of these traffic types. We use a variety of parameters to describe these types of traffic. Specifically, we focus on packet size, interrival time, traffic saturation, and burstiness.

File Download Traffic: File downloads are elastic, meaning that they must be reliable but are also tolerant to random delays. Applications that *generate* file download traffic usually do so with long bursts of packet arrivals and long interarrival times [?]. File downloads are sufficiently random that we do not consider a fixed distribution for any parameters. Rather, we choose these at random by our own free will.

Web Browsing Traffic: Web browsing traffic is quite diverse in packet size, interrarival time, and burstiness. Mah [?] studied various web browsing traces to develop statistics for these characteristics. In particular, it was found that these characteristics, such as content request and reply lengths (packet size) follow a Zipf disribution (see Figures **??** and **??**).For simplicity, we assume that web browsing traffic has random packet sizes sample from a Zipf distribution, deterministic interarrival time, and probabilistic packet arrival in the buffer. This enables us to use a slightly modified version of the nonsaturated DCF model (described in Section 5.2) to study this traffic.





Distribution of web traffic request and reply sizes, respectively [?].

Video Streaming Traffic: Video streaming traffic as described in [?] is composed of two types of packets – I and D packets – each of which are tightly correlated and with different characteristics. Video segments are sent in sets of I and D packets, wherein multiple small D packets carrying video data are dependent on a single I packet with metadata necessary to decode the video stream. The length of each I and D packet is correlated to past packet lengths. For example, let $\lambda_j(t|s)$ denote the probability that a packet of type $j \in \{I, D\}$ is t time slots long given that the previous packet of type j was s packets long. The size of both I and D packets are bounded within a finite range so as to make the analytical model tractable.

In this work we consider a more specific type of multimedia encoding scheme, namely, MPEG. MPEG video traffic streams consist of I, P, and B packets [?]. In this case, a single I frame is transmitted, is an intraframe and is heavily coded (thus, larger) to ensure it can be correctly decoded by all receivers. P frames follow the I frame, and are used to transmit actual multimedia data. These rely on the previous I frame and P frames to be decoded correctly. B frames – bidirectional frames – are transmitted in between I and P frames and naturally serve to relay information in both directions.

To further understand the differences between these types of traffic, we used our simulator to generate system state transition traces in the context of a simulator. The results of these traces are shown in Figure 3. Note that the superposition all traces, shown in the bottom Figure, is the object of study in this work. Namely, we seek to understand the behavior of the DCF as it serves packets generated from sources of heterogeneous traffic.

We will use the characteristics of each of these models to tune our Markov models in the experimental section of this report.

4 The 802.11 Distributed Control Function

The 802.11 DCF [?] is at the core of this work. It is a simplistic random access scheme based on the carrier sense multiple access with collision avoidance (CSMA/CA) protocol. Failed packets are retried according to a

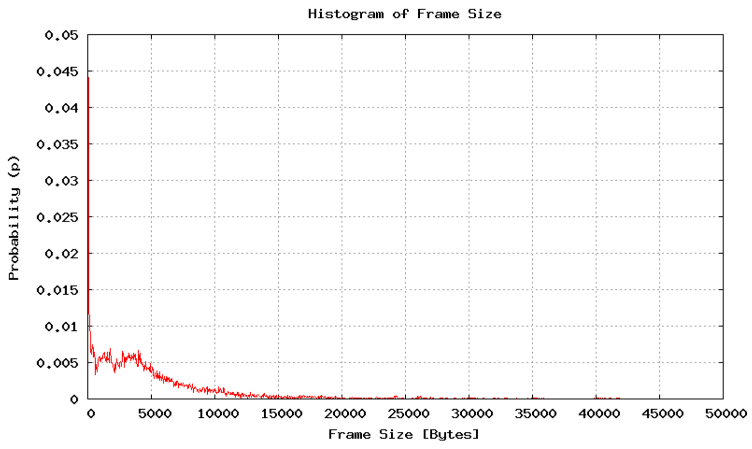


Fig. 1. Distribution of multimedia frame sizes [**?**].

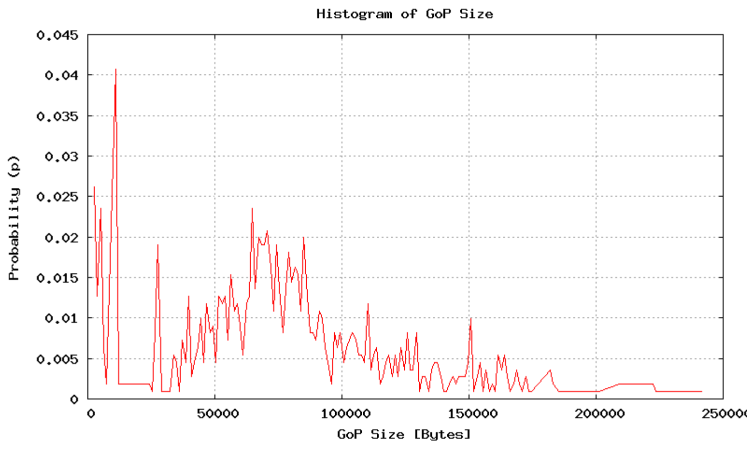


Fig. 2. Distribution of GoP sizes in multimedia traffic [**?**].

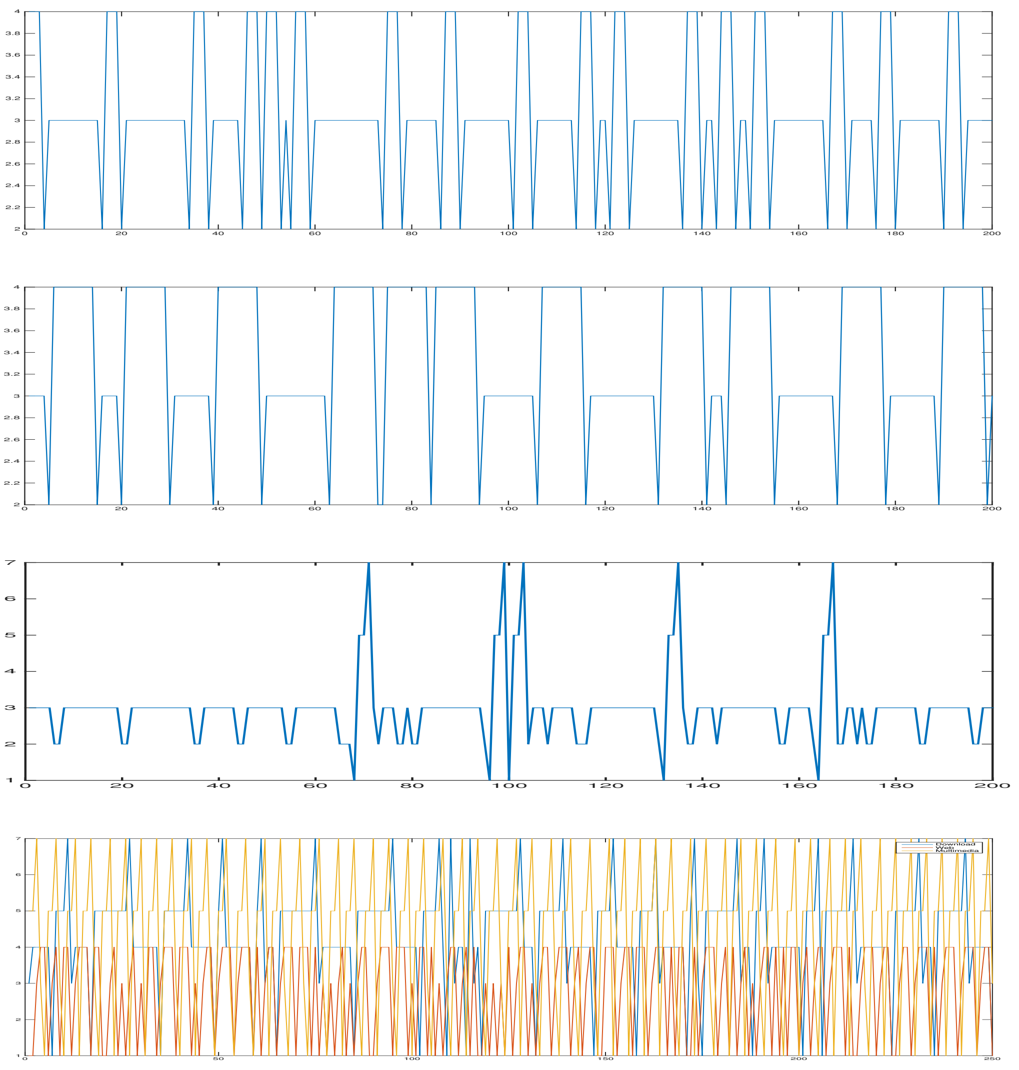


Fig. 3. Individual system transition traces for (1) web browsing, (2) file download, and (3) video streaming traffic. The bottom trace shows the superposition of all the above traces. The x axis is the discrete time in the system, and the y axis is the state. For completeness, state 5 is interarrival, 4 is the packet size calculation, 3 is a waiting state (e.g., backoff, postbackoff), 2 is a failure transmission, and 1 is a successful transmission.

binary exponential backoff rule. At each packet transmission, the backoff is uniformly in the range $(0, w-1)$, where w is the length of the "contention window" and is directly dependent on the number of failed attempts for the packet thus far. The window w is set to W_{min} to begin, and upon every failure, the backoff counter is doubled. At stage i, the backoff timer is $2^i W_{min}$. The maximum backoff is bounded by $W_{max} = 2^m W_{min}$. Selections of W_{min} and W_{max} are dependent upon the physical layer specifications in the 802.11 standard [?,?]. For example, frequency hopping spread spectrum calls for $W_{min} = 16$ and $W_{max} = 1024$ $(m = 6)$.

5 Incremental Markov Models for Heterogeneous Traffic

In this section we describe the necessary evolutions of Bianchi's original Markov model to support parameterized packet length, interarrival time, and buffer saturation.

5.1 Original DCF Model

A simple Markov model for the 802.11 DCF function was proposed in the seminal work by Bianchi [?]. The system state is described by two discrete-time stochastic processes $b(t)$ and $s(t)$: $b(t)$ represents the backoff *time counter* and $s(t)$ represents the backoff *stage* for a given station or node, each of which is advanced at each time step (i.e., from t to $t+1$). Each node is equipped with a *saturated* buffer of packets to transmit; there is never a case where a node has to wait for a packet to arrive in the buffer before attempting a transmission. Also, instead of modeling multiple nodes simultaneously to accurately assess the probability of collision, this model assumes a constant probability of collision p – the conditional collision probability – at each time slot. In the real world, this probability is a function of the network and environmental interferences, e.g., shadowing, fading, etc. However, by making this a constant and therefore independent from other nodes, the model simplifies to capturing the dynamics of the bidimensional process $\{b(t), s(t)\}$ as a discrete-time Markov chain, shown in Figure 4.

The only non-null transition probabilities in this Markov chain are given below:

$$\boxed{\begin{array}{ll} \Pr[i,k|i,k+1] = 1, & k \in [0, W_i - 2],\ i \in [0,m] \\ \Pr[0,k|i,0] = (1-p)/W_0, & k \in [0, W_0 - 1],\ i \in [0,m] \\ \Pr[i,k|i-1,k] = p/W_i, & k \in [0, W_i - 1],\ i \in [1,m] \\ \Pr[m,k|m,0] = p/W_m, & k \in [0, W_m - 1] \end{array}}$$

5.2 Limited and Diverse Traffic

While accurate, the original DCF model is constrained in that it assumes homogeneous traffic and a saturated stream of packets for transmission. Malone et al. [?] presented a modified version of Bianchi's model that captures non-saturated traffic loads. The essential idea behind their variant is that there exists a constant packet arrival probability q at each time slot, much like there exists a constant collision probability p. If a node successfully transmits a packet and the buffer is not-empty, the state of the system proceeds as normal. Conversely, if a packet is not ready for transmission, then the chain enters a stage known as "postbackoff," denoted $(0,k)_e$ for $k \in [0, W_0 - 1]$. A node may remain in this set of states indefinitely until a packet arrives and the channel is idle.

To capture the mechanics of the DCF function in this condition, the postbackoff set of stages are nearly indentical to backoff stage $i = 0$. If a packet arrives at any time when the system is in a postbackoff state, it immediately transitions into the backoff stage $i = 0$, with a decremented backoff timer. However, if the postbackoff timer reaches 0, where the postbackoff is said to be complete, the node will stay in this state until a packet arrives with probability q. Once a packet arrives in state $(0,0)_e$ with probability q, there are three possible outcomes: (1) the packet is transmitted successfully, a collision occurs with probability p, or the medium (channel) is sensed as busy with probability $1 - P_{idle} = p$.

With the addition of the postbackoff states S', the Markov chain itself is now a multidimensional process $\{b(t), s(t), S'\}$. The new state transition probabilities needed to capture the transitions between these multiple dimensions are below.

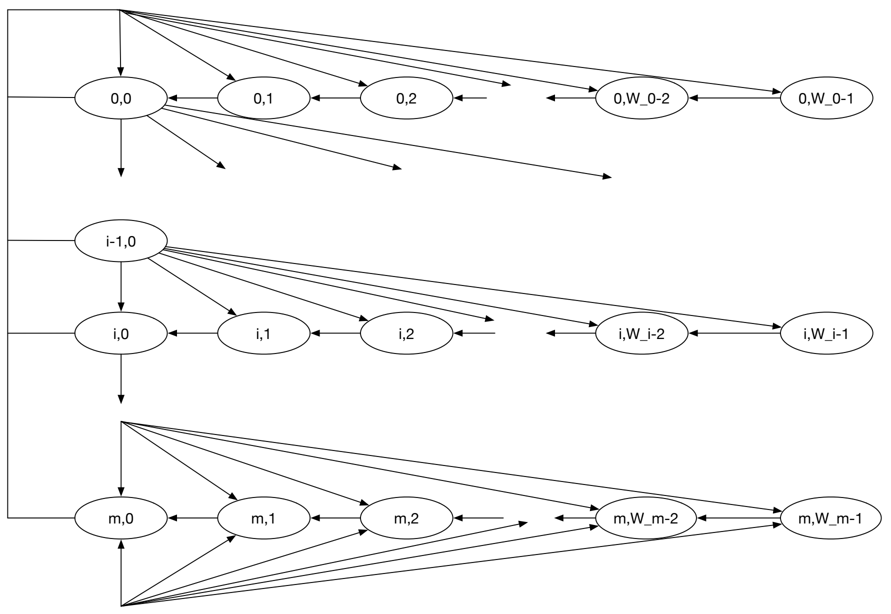


Fig. 4. The original saturated DCF Markov model [**?**].

$\Pr[(i,k-1)\mid(i,k)] = 1$	$k \in [1, W_i - 1]$	$i \in [1, m]$
$\Pr[(0,k-1)_e\mid(0,k)_e] = 1 - q$	$k \in [1, W_i - 1]$	$i \in [1, m]$
$\Pr[(0,k-1)\mid(0,k)_e] = q$	$k \in [1, W_i - 1]$	$i \in [1, m]$
$\Pr[(0,k)_e\mid(i,0)] = \frac{(1-p)(1-q)}{W_0}$	$k \geq 0$	$i \in [0, m]$
$\Pr[(0,k)\mid(i,0)] = \frac{(1-p)q}{W_0}$	$k \geq 0$	$i \in [0, m]$
$\Pr[(\max\{(i+1,m\},k)\mid(i,0)] = \frac{p}{W_{\max\{i+1,m\}}}$	$k \geq 0$	$i \in [0, m]$
$\Pr[(0,0)_e\mid(0,0)_e] = 1 - q + \frac{qP_{idle}(1-p)}{W_0}$		
$\Pr[(0,k)_e\mid(0,0)_e] = \frac{qP_{idle}(1-p)}{W_0}$	$k > 0$	
$\Pr[(1,k)\mid(0,0)_e] = \frac{qP_{idle}p}{W_1}$	$k \geq 0$	
$\Pr[(0,k)\mid(0,0)_e] = \frac{q(1-P_{idle})}{W_0}$	$k \geq 0$	

Malone et al. [**?**] exploited the inclusion of this postbackoff state and the fixed arrival probability constant to study the performance of the DCF while transferring packets from unsaturated heterogeneous traffic, e.g., file downloads, web traffic, etc. However, their analysis was limited in scope, since heterogeneous traffic types were parameterized only by p and q.

5.3 Adding Variable Length Frame Payloads

Despite the flexibilty in the previous model, it is still limited with respect to packet size. In particular, the model assumes that each packet has an equally sized payload. This is not true, especially for video traffic [**?**], which typically consists of two types of packets of distinctly different sizes: d packets are small video frame/segment packets that carry information needed to render a piece of video data, and i packets are large "metadata" packets that contain information necessary to decode d packets. Multiple d packets are often tied to a single i packet in such a way that if the i packet is lost, none of its children d packets can be decoded correctly. Consequently, we need to be able to model packets of different sizes.

To do this, we consider a range of possible packet sizes, e.g., $[1, l]$, where l is the maximum packet payload size. In this context, the packet size corresponds to *how many* discrete time slots are needed to transfer the content over the channel. In the previously two discussed models, the packet size was assumed to be 1 since

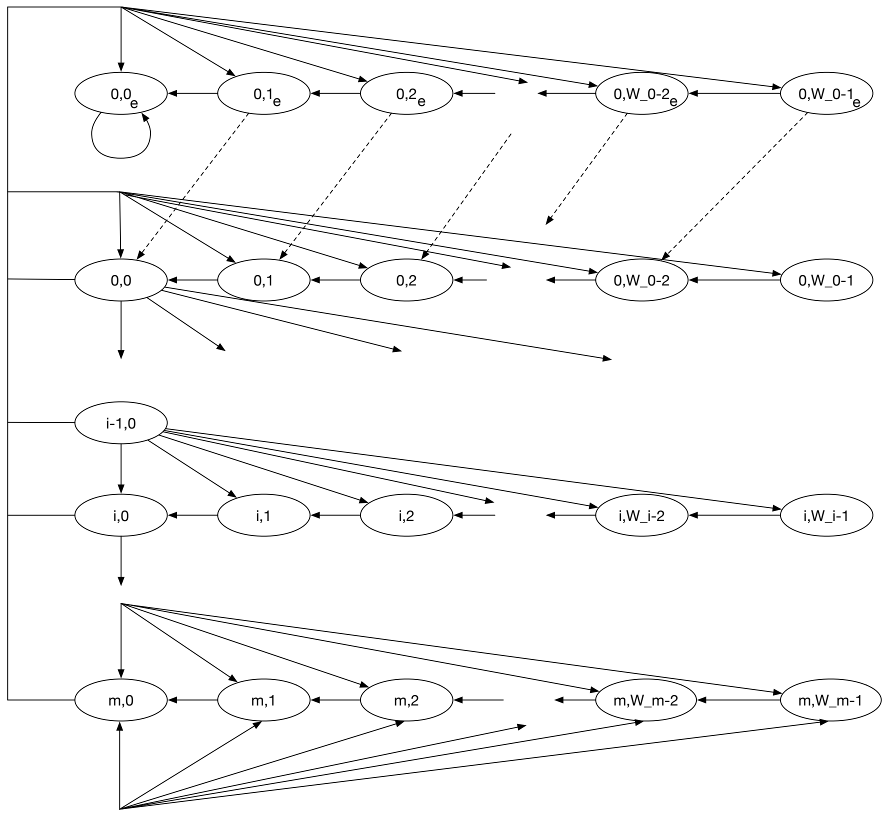


Fig. 5. The modified DCF Markov model that captures non-saturated traffic [**?**].

they were assumed to be transferred in a single time step. Since we may want to support both deterministic and arbitrary packet sizes, drawn from a specific distribution, we use what we call size chain blackboxes to capture this type of variability. Figure 6 shows a sample size chain blackbox for the range of values $[1, 4]$. There are l ($l = 4$) transitions into the blackbox, and each transition l' is to a chain of length l'.

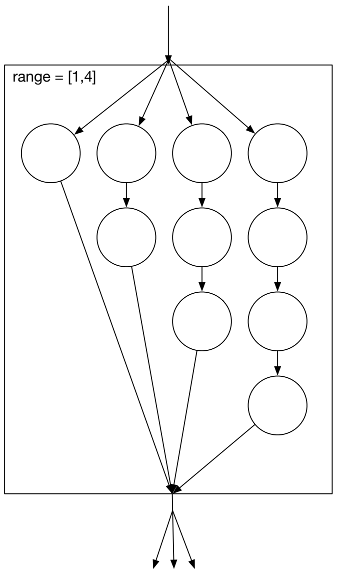


Fig. 6. A blackbox Markov chain that captures bounded variabilities in a particular parameter, such as packet length or packet interarrival time.

To show how this chain would be used, assume that the size chain blackbox represents the size of a particulr packet. Further, assume that the packet size is a discrete random variable ranging from 1 to 4 with uniform distribution. The probability of transferring to any chain within the blackbox from the entry state is $(1/4)$. If the transition was to the last chain of length $l = 4$, then the state would appear to “loop” in place for 4 time steps before exiting the blackbox. Conversely, if the transition was to the first chain of length $l = 1$, then the state would only loop once before exiting. Although these states are costly from the persepctive of space, this type of construction enables us to model such discrete random variables with any distribution within our Markov chain.

The first application of these size chain blackboxes are to extend the previously discusssed nonsaturated model to support variable packet sizes. Specifically, the number of time steps needed to transfer a packet and detect collision is now a bounded discrete random variable. This means that once a packet has begin transmitting it enters a size chain blackbox before either (a) detecting collision or (b) completing successfully. It is important to note that for packets of length $l > 1$, the probability of a collision is no longer q. Rather, a collision occurs if there is a collision in *any* time slot during which the packet was being transmitted. This means that the probability of collision for a packet of length l is $1 - (1 - p)^l$.

The new states required to model the size chain blackboxes are tuples of the form (i, l, l'), where i is the backoff stage and l is the packet length, and l' is the *state* within the size chain of length l. For example, if the state transitions into a size chain of length $l = 4$ from state $(i, 0)$, then the following series of transitions would occur:

$$\begin{aligned}(i, 4, 4) &\rightarrow (i, 4, 3)\\ &\rightarrow (i, 4, 2)\\ &\rightarrow (i, 4, 1)\end{aligned}$$

To model this behavior, which is visually shown in Figure 7, the following new state transition probabilities are included into the model:

$$\boxed{\begin{array}{ll}
\Pr[(i,l,l'-1)|(i,l,l')] = 1 & l' \le l\ l' \ge 0 \\
\Pr[(i+1,k)|(i,l,0)] = \frac{1-(1-p)^l}{W_{i+1}} & k \in (0, W_{i+1}-1) \\
\Pr[(0,k)|(i,l,0)] = \frac{(1-q)(1-p)^l}{W_0} & k \in (0, W_0-1) \\
\Pr[(0,k)_e|(i,l,0)] = \frac{q(1-p)^l}{W_0} & k \in (0, W_0-1)
\end{array}}$$

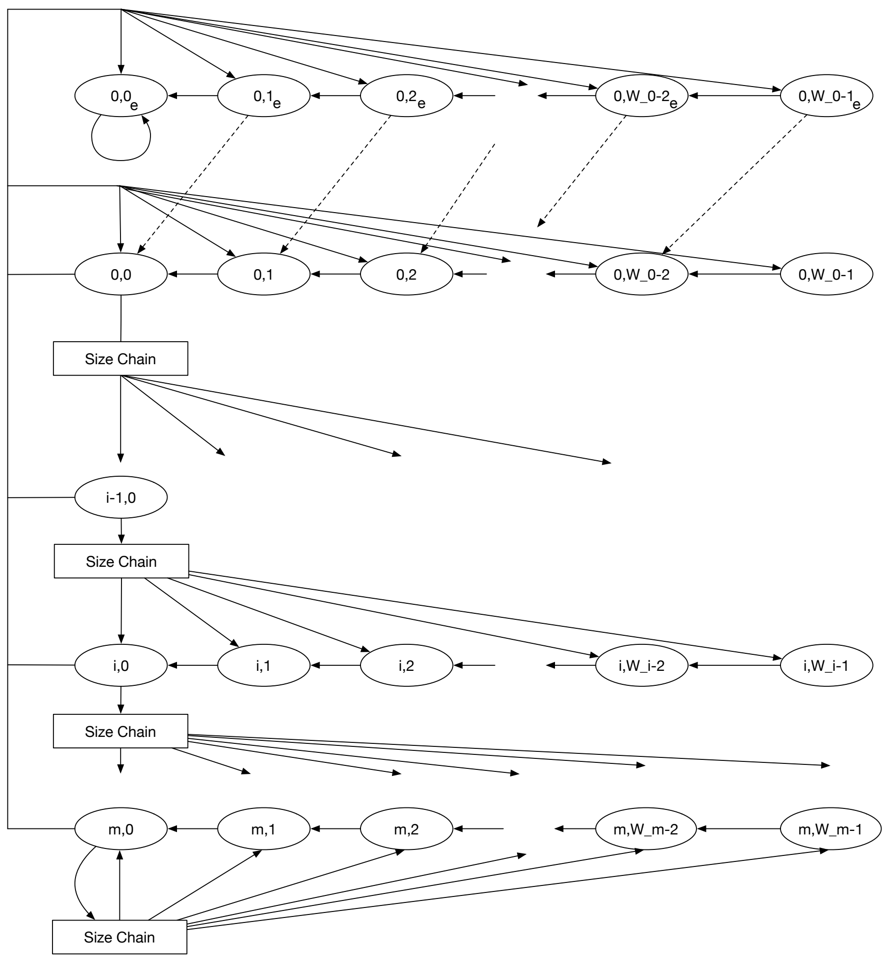


Fig. 7. The modified unsaturated DCF Markov model that captures variable-length packet payloads.

5.4 Adding Arbitrary Inter-Arrival Times

The final extension of our Markov model addresses the interrarival packets. The nonstaurted model proposed by Malone et al. [?] partly solves this problem by introducing a fixed probability q such that, at every time step, a new packet will arrive in the buffer to be transmitted. While useful, this does not allow us to capture more sophisticated interarrival times. For example, the interrarival time may be a random variable with a Zipf distribution. This mode would accurately capture a user quickly browsing though websites like Pinterest or Reddit.

To capture these dynamics, we re-use the size chain blackboxes introduced in the previous section. In particular, after a successful transmission, the state of the system can enters an interarrival time blackbox before either (a) receiving the next available packet or (b) entering the postbackoff state because a packet has not yet arrived. This extension is shown in Figure 8. If need be, we can enforce that $q = 1.0$ so that the postbackoff states are replaced by the interarrival time blackboxes.

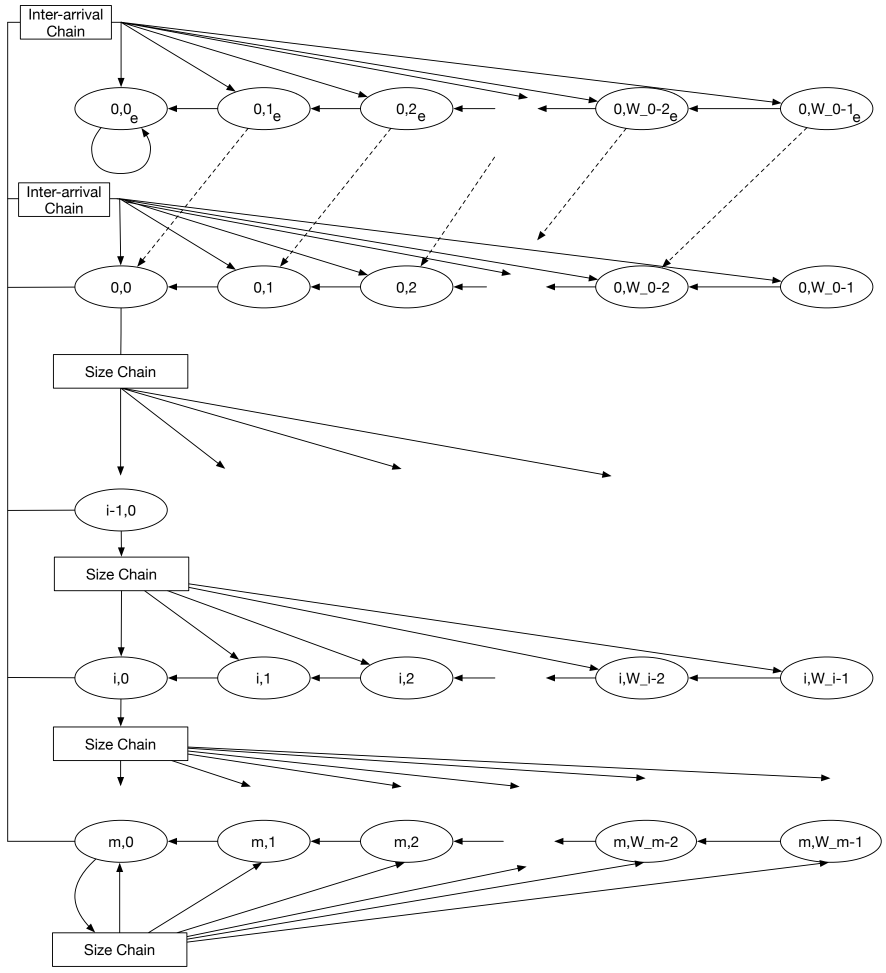


Fig. 8. The modified unsaturated DCF Markov model that captures variable-length packet payloads.

6 Simulator Design and Implementation

A major part of this research was developing a simulator that could manage the growing complexity of the multi-dimensional Markov state. To this end, in this section we describe the relevant design and implementation details that were used to realize the analytical models just described. These details will be of importance to those seeking to extend our tunable analytical models.

6.1 Managing Markov Model Dimensional Complexity

By adding more tunable parameters to Bianchi's 2-dimensional DCF, we needed a way to manage the growing complexity of the model. We accomplished in two ways:

1. Creating small Markov models for each "state" of the system, i.e., the packet size calculation, treating them as black boxes.
2. Use "compressible" states as logical and instantaneous bridges between Markov model black boxes.

An example of a compressible state is shown in Figure 9. Observe that the $x1$ and $x2$ states are the "bridges" between the states $S1$ and $S2, S3, S4$. When "compressed", the transition probabilities between $S1$ and $S2$, for example, becomes the product of the transition probabilities into and out of the compressible state.

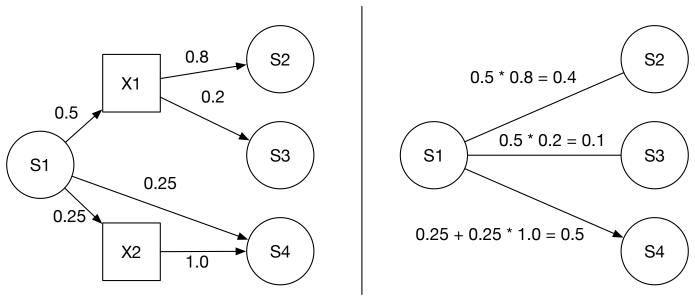


Fig. 9. A sample usage of "compressible" states that bridge the transition gap between separate Markov chains.

To illustrate the efficacy of these states, consider the compressible variant of the DCF model shown in Figure 10. It is easy to see its equivalence to the original DCF model after the compressible (green) states are compressed. Our implementation uses these compressible states to bridge between separate Markov chains. In this case, we treat each backoff timer stage as a separate chain. Moving between these chains, either through transmission success or failure, happens through compressible states.

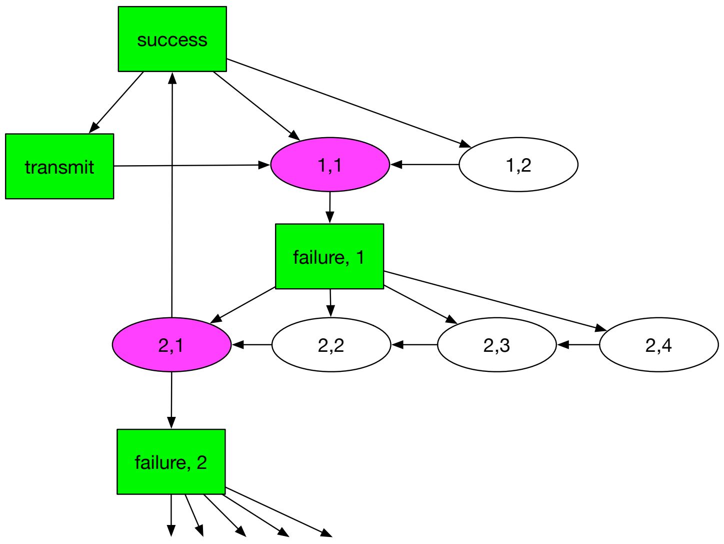


Fig. 10. Bianchi's DCF model represented using compressible states. Green states are collapsible.

With this representation, adding support for the postbackoff, interarrival, and packetsize Markov chains is simple. To this end, consider the extended Markov model shown in Figure 11. After a packet successfully transmits, it enters the transmit state, indicating that it is ready to transfer another packet. If there is a non-zero probability of an interarrival time greater than one (1), the transmit state will enter the inter-arrival state. Then, depending on the distribution of the interarrival times, the state will transition into the interarrival chain at the appropriate index. Notice that the index into this chain determines the interarrival time.

As an example, let the interarrival time be a discrete random variable with uniform distribution sampled from the range [1,3]. If the interarrival time is determined to be 3, which will happen with probability exactly 1/3, then the chain will enter the index at state 3. Since the probabilities between the interarrival chain states are deterministic with probability 1.0, this means that there *will always* be 3 epochs before the state transitions out of the interarrival chain and into the "ready-to-transmit" state.

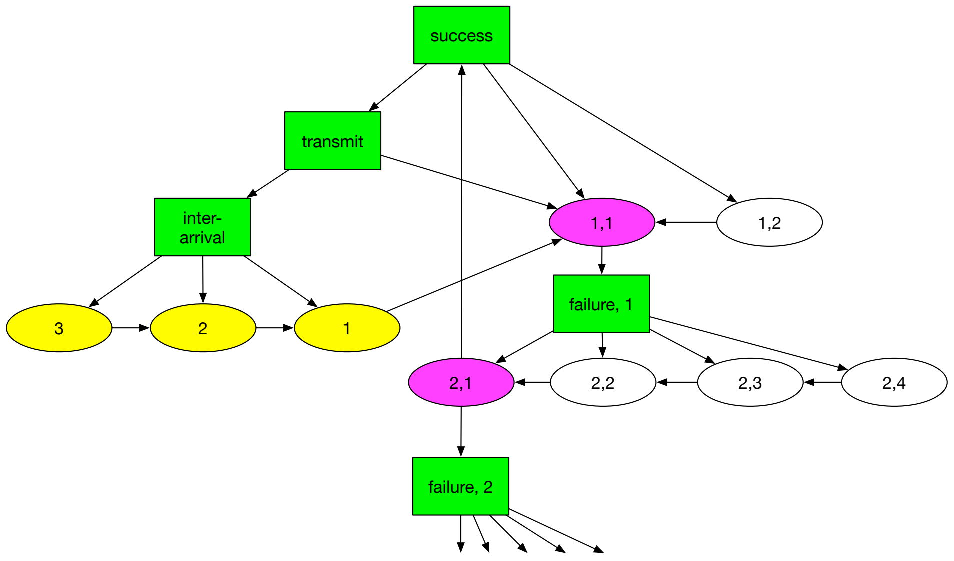


Fig. 11. The extended compressible DCF model with support for parameterized interarrival times.

To support different packet lengths, we follow the approach outlined in the previous section and use packet size chains, which are similar to interarrival time chains, before attempting to transmit a packet. We emphasize here that the implementation does not strictly adhere to the model previously described. The astute reader will have observed that the model should compute the size of a packet *once*, and then use that same packet size for every transmission attempt. The extension we presented in the previous section recomputes the packet size at every transmission attempt. We presented the model this way for clarity only.

In the actual implementation, a packet size range of length n is modeled by *duplicating* the extended DCF model n times, where each copy has $i = 1, \ldots, n$ has a packet size chain of length i at the "beginning" of the model. This is illustrated in Figure 14, where, after a packet arrives, the state of the system transitions into the copy with the appropriate packet size chain. For example, let the packet size be a discrete random variable sampled from [1,3] with a uniform distribution. With probability 1/3 the state of the chain will transition into the the DCF copy (black box) with the packet size of length 1, where it will remain until the packet is transmitted successfully.

6.2 Simulator Overview

Having described the architecture of the Markov chain, we now describe how we implement the multi-node simulator using individual instances of these chains. Let n be the number of nodes in a system, each with a unique set of traffic characteristics and, therefore, a unique Markov model, state space, and transition probability matrix. The simulator maintains a collection of the

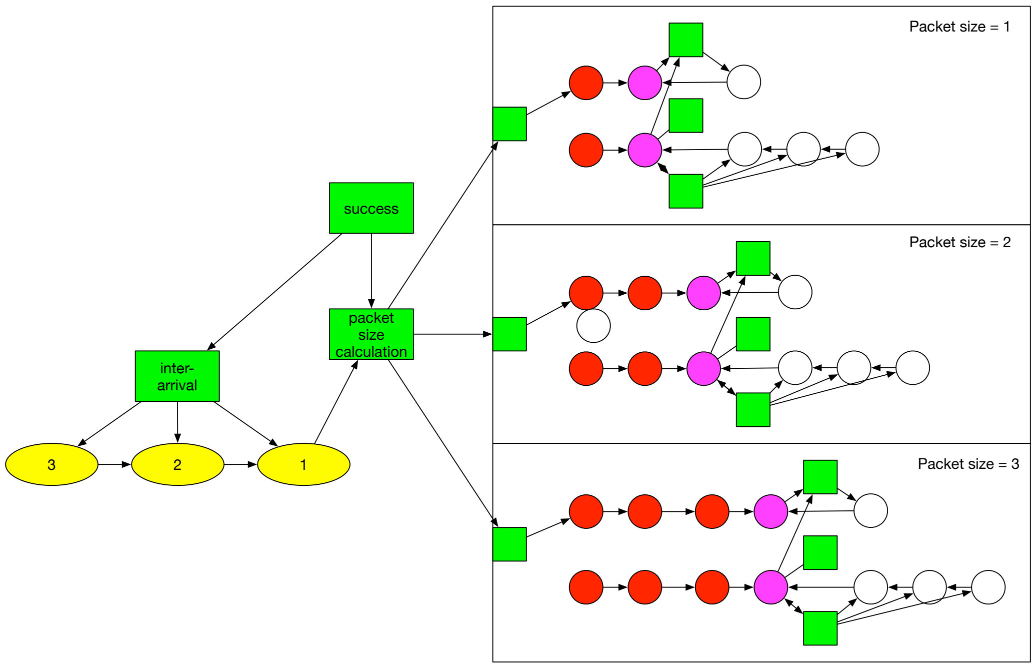


Fig. 12. The extended compressible DCF model with support for parameterized packet lengths.

```
Require: Some long text here. Unfortunately this text is a mess. Spaces and line breaks are missing and
    the text gets weird block layout when setting line breaks manually.
  for all $i = 1 \ldots | L_{items} |$ do
    i miss spaces here, too
  end for
```

6.3 Metrics Computations

In our work, we mainly consider the following metrics: throughput, packet transmission success probability, and packet failure probability. Let n be the total number of steps in the simulation, s be the number of successful transmissions, and f be the number of failed transmissions. Throughput S is defined as the ratio of successful transmissions out of *all* possible time steps in the system, i.e., $S = p/n$. The success probability P_s is defined as the number of success transitions compared to the number of transmission attempt transitions, i.e., $P_s = p/(p + f)$. Finally, the packet failure probability P_f is defined as the number of failied transitions compared to the number of transmission attempts, i.e., $P_f = f/(p + f)$.

7 Experiment Setup

8 Simulation Results and Analysis

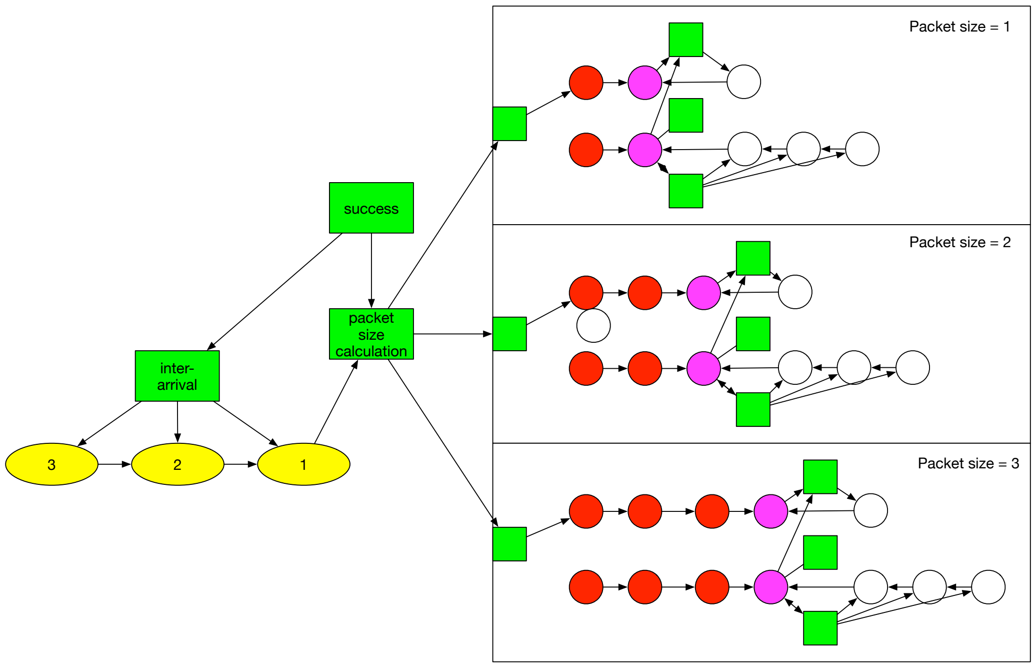


Fig. 13. TODO.

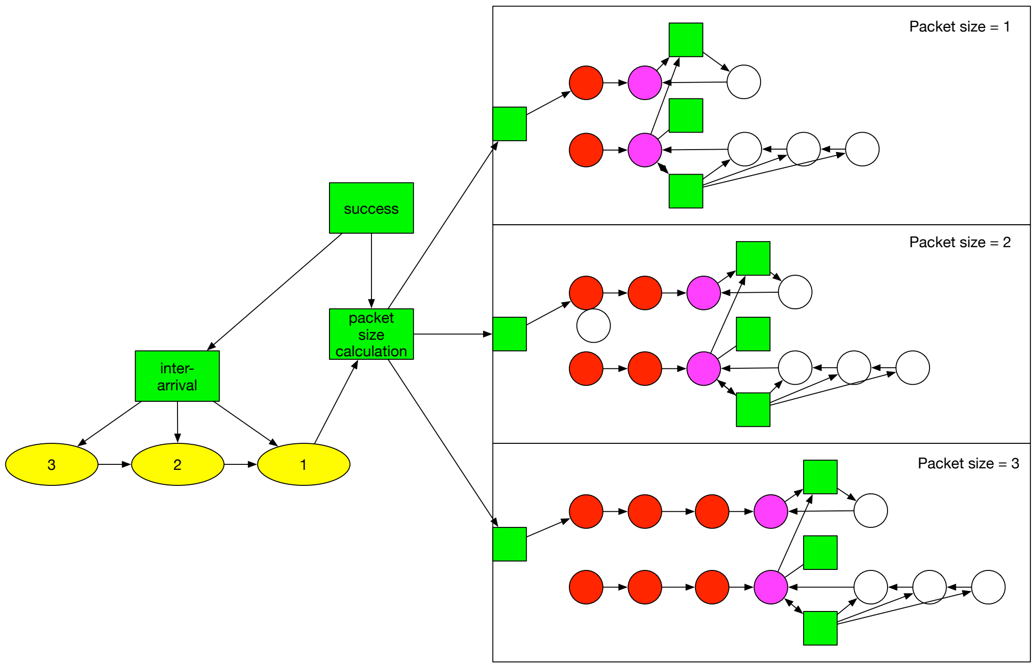
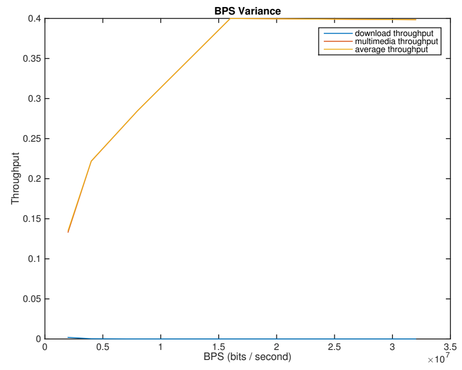
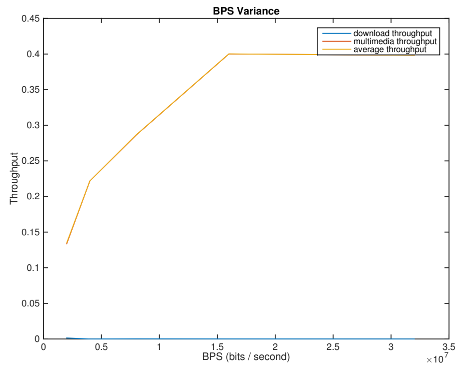
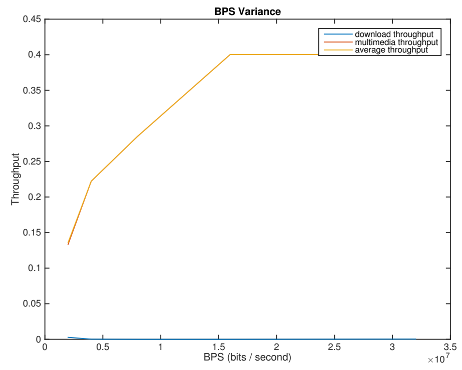
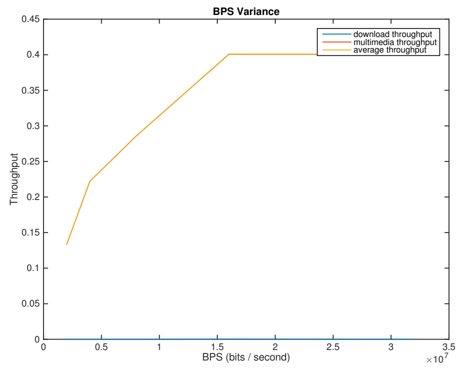


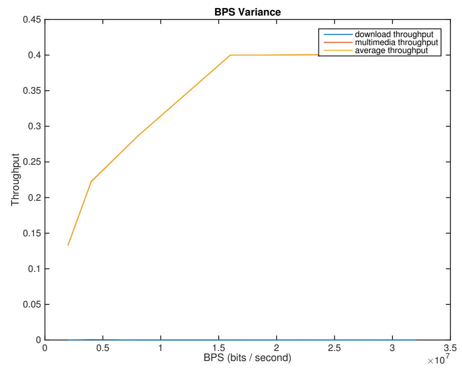
Fig. 14. TODO.

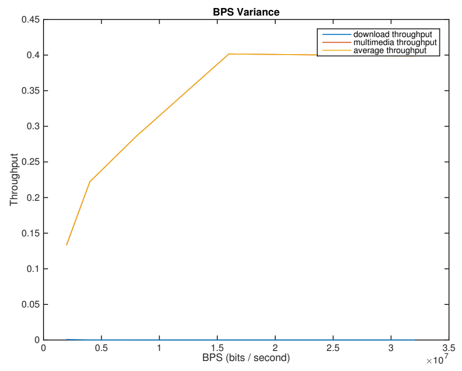

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

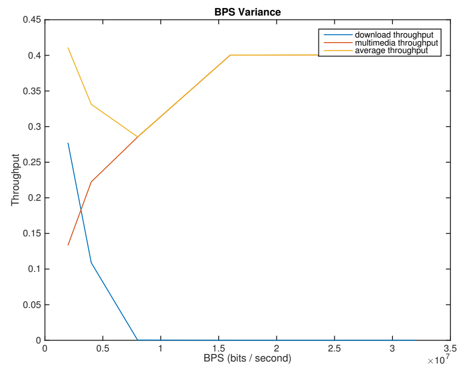

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

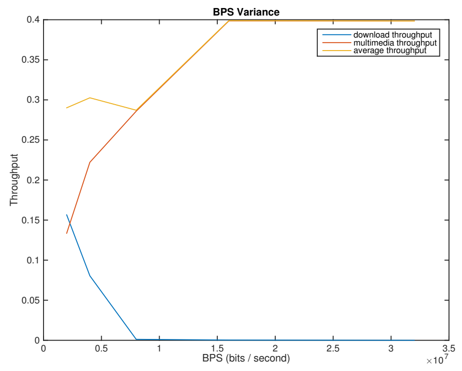

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

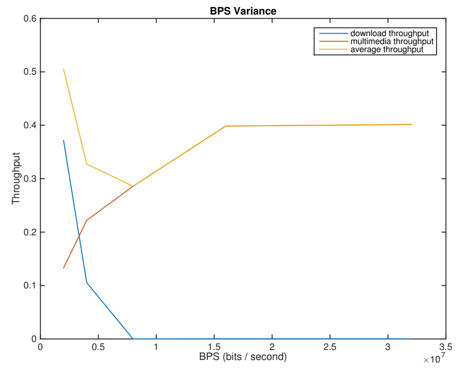

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

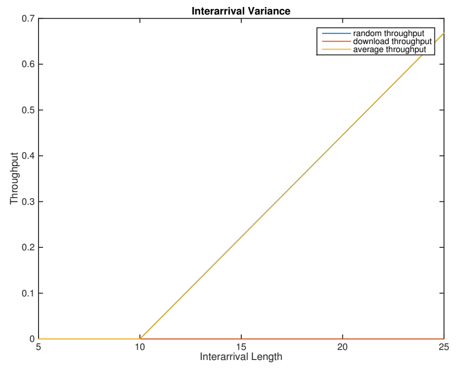

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

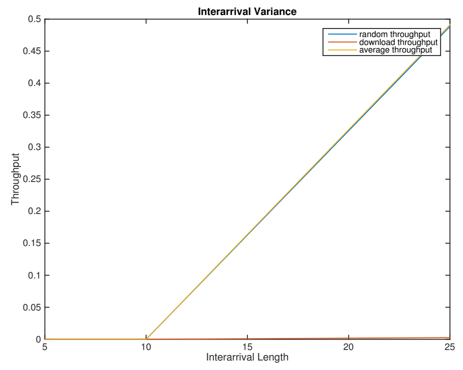

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

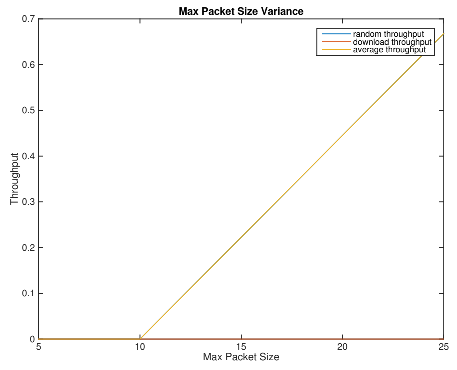

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

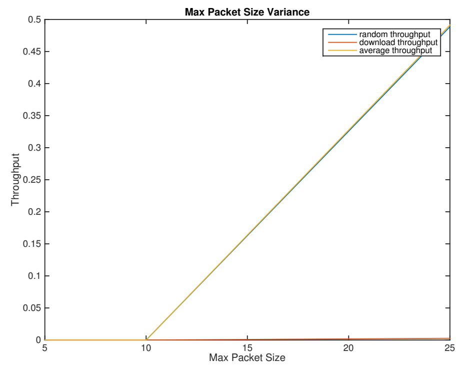

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
$\times 10^7$

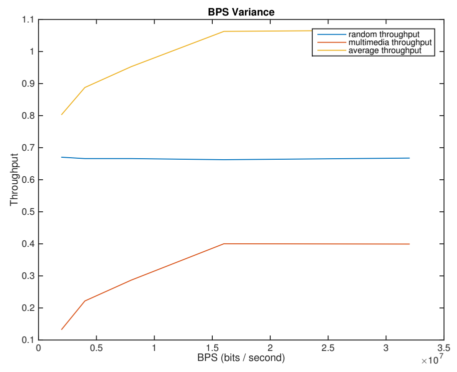

BPS Variance
download throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
×10^7

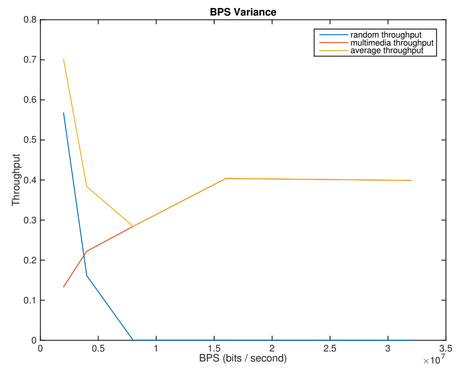

Interarrival Variance
random throughput
download throughput
average throughput
Throughput
Interarrival Length

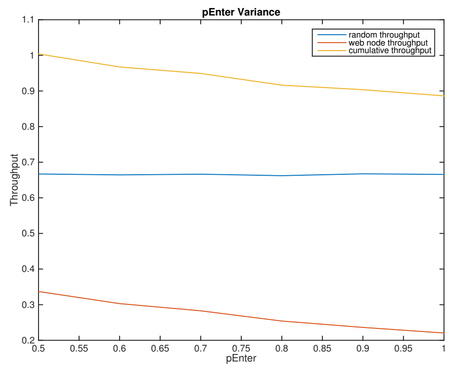

Interarrival Variance
random throughput
download throughput
average throughput
Throughput
Interarrival Length

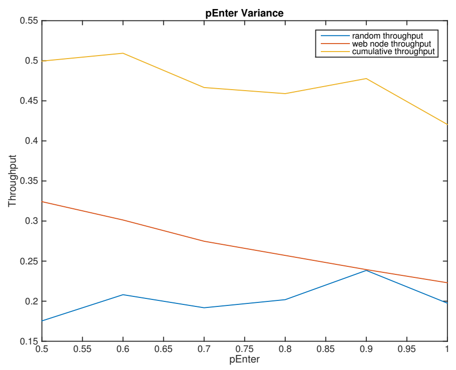

Max Packet Size Variance
random throughput
download throughput
average throughput
Throughput
Max Packet Size

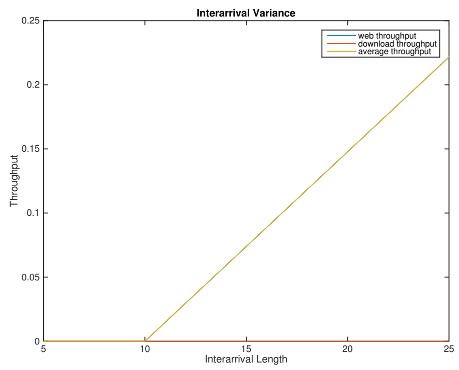

Max Packet Size Variance
random throughput
download throughput
average throughput
Throughput
Max Packet Size

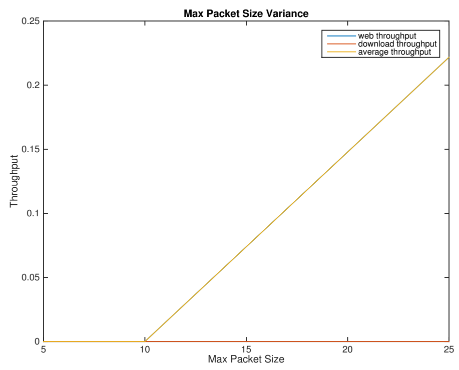

BPS Variance
random throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
×10^7

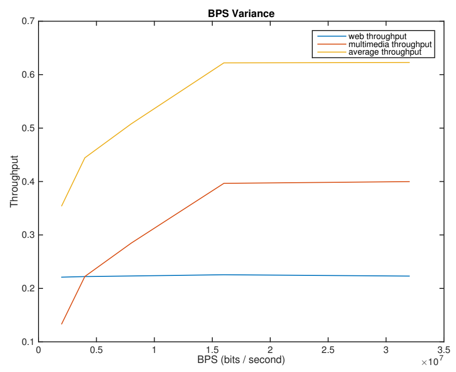

BPS Variance
random throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
×10^7


pEnter Variance
random throughput
web node throughput
cumulative throughput
Throughput
pEnter


pEnter Variance
random throughput
web node throughput
cumulative throughput
Throughput
pEnter


Interarrival Variance
web throughput
download throughput
average throughput
Throughput
Interarrival Length


Max Packet Size Variance
web throughput
download throughput
average throughput
Throughput
Max Packet Size


BPS Variance
web throughput
multimedia throughput
average throughput
Throughput
BPS (bits / second)
×10^7