A COMPARATIVE STUDY OF BROADCASTING ON MULTIPLE ACCESS CHANNELS

by

MUSTAFA A. ALKHAFAJI

B.S., Thi-Qar University, 2011

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Master of Science

Computer Science Program

2017

This thesis for the Master of Science degree by

Mustafa A. Alkhafaji

has been approved for the

Computer Science Program by

Bogdan S. Chlebus, **Chair**

Farnoush Banaei-Kashani

Ellen Gethner

Date: December 16, 2017

Alkhafaji, Mustafa A. (M.S., Computer Science Program)

A Comparative Study of Broadcasting on Multiple Access Channels

Thesis directed by Bogdan S. Chlebus

ABSTRACT

We study broadcasting on multiple access channels by deterministic distributed algorithms. Packet injections are determined by adversarial models. The power of a worst-case adversary is constrained by the rate of injection and a bound on the number of different packets that could be injected in one round. We propose models of packet injections that conform to worst-case adversarial constraints but have stochastic components that can be simulated. We implement and simulate a number of deterministic and randomized broadcast algorithms. The performance of broadcast algorithms is measured by packet latency. We compare various broadcast algorithms by their performance in simulations.

The form and content of this abstract are approved. I recommend its publication.

Approved: Bogdan S. Chlebus

DEDICATION

I dedicate this work to my parents, and to my whole family who have supported me throughout the time of this work and motivated me to challenge the difficulties of life and become the person who I am.

ACKNOWLEDGMENTS

I want to thank the Higher Committee of Education Development in Iraq (HCED) for supporting my studies at the University of Colorado Denver.

Special thanks go to my advisor Bogdan Chlebus for his guidance through this work.

I owe many thanks to my dear friend Mohammed Qasim for his encouragement and advice.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure

CHAPTER I

INTRODUCTION

A multiple access channel is a formal model of of a broadcasting environment. It is an abstraction of the Ethernet [16], which is the most popular implementation of local area networks.

There are a number of nodes (stations) attached to a channel. Multiple soverlapping transmissions result in an interference, which is also called a collision for access to the transmitting medium. The semantics of multiple access channels is defined by two properties:

(1) when one station transmits then the message is heard by all the stations.

(2) multiple overlapping transmissions collide with each other and none is heard on the channel.

We consider the slotted model of a channel in which an execution of a broadcast algorithm is partitioned into rounds, so that it take a round to transmit one message and multiple messages transmitted in the same round result in a collision.

We discuss and compare distributed broadcast algorithms when packets are injected continuously into stations. The numbers of packets injected into the stations are constrained by a leaky-bucket adversarial model [1, 5, 9, 11, 12].

The adversarial model which provides an upper bound on the amount of generated traffic is determined by two parameters: injection rate and burstiness, and is understood as is known as a leaky-bucket adversarial model. Packet injection denotes an upper bound on the average number of packets injected per round, averaged over all possible time intervals. Burstiness is defined as the maximum number of packets that can be injected in a round.

The simulations we design constrain the number of packets generated such that they always conform to an adversary with some injection rate ρ and burstiness b. Packets are initially generated with a suitable Poisson distribution, which has its

parameter λ adjusted to ρ. This together gives three parameters: (ρ, λ, b).

The number of stations that attached to the channel is fixed and their number n is known, in that it can be used in codes of algorithms. Stations are supplied with private queues, in which they can store packets until they are heard on the channel.

The performance of broadcast algorithms is measured by packet latency. It could be understood either in the worst-case, for a suitable adversarial model, or in the average sense, subject to stipulations of packet injection that involves randomization in packet generation.

The traditional algorithmic approach to broadcast on multiple access channels in a distributed manner uses randomization to arbitrate for access to the channel. Such randomized algorithms include back-off protocols, such as the binary exponential back-off used in the Ethernet [16] suite of technologies.

Recently, deterministic broadcast algorithms were proposed and showed to have bounded packet latency in the worst case in adversarial models determined by injection rate and burstiness. We simulate both deterministic algorithms and randomized ones.

The thesis compares various algorithms through simulations. The theoretical worst-case upper bounds on packet latency are compared to bounds obtained in simulations, both average and maximum.

CHAPTER II

RELATED WORK

There is an rich literature on algorithms for multiple access channels, see the surveys [10, 13]. The methodology of adversarial queuing was considered extensively in the literature [7, 8, 9, 11, 12].

Some studies that related with this types of project is considered packet routing when packets are injected continuously into a network. They evolved an adversarial theory of queuing reached at addressing some of the limitations queuing theory based on fixed-time constant generation and origin in probabilistic analysis. They test the stability and reliability of queuing networks and behavior when the arrival data are adversarial [9] .

Anantharamu and Chlebus [1] study broadcast in multiple access channels in dynamically adversary form. There is an unlimited bound provide of stations that don't have names attached to a synchronous channel. There is an adversary that injects packets into stations to be broadcast on the channel. The adversary is constrained by burstiness, injection rate, and by number of passive stations that can be concurrently activated by providing them packets. They consider deterministic distributed broadcast algorithms, who are further classified by their attributes. They show for which injection rates can algorithms investigate bounded packet latency, when adversaries are determined to be able to activate at most one station per round. The rates of algorithms they introduce make the increasing sequence consisting of 1/3, 3/8 and 1/2 give a picture of the additional properties of algorithms. Also, they show that injection rate 3/4 couldn't be handled with bounded packet latency.

Most previous work on broadcasting on multiple access channel has been carried out assuming that randomization has related part to play with. When packets are created subject to stochastic limitations, randomness can affect on the status of protocols either actively, by being a part of the mechanism of a randomized protocol,

or passively. The randomize environment as a part of its specification determined in such a way can be represented as a Markov chain so that stability could be understood as ergodicity. For the recent work on the algorithmic of randomized broadcasting on multiple access channels, see [14, 18].

Andrews and Zhang [6] developed scheduling and routing in wireless networks where each node can be transmited data to at most one neighboring node per time slot. Furthermore, transmission rates and data arrivals are governed by an adversary. They generated scheduling algorithms that ensure network stability for the case in which the adversary specifies the links that the data must transmission through. Bender [7] applied adversarial approach to study the issue of productivity of randomized back-off for multiple-access channels in the queue- free model.

Injection rate is defined as the large average number of messages generated in a category of slots of rounds. Burstiness is the maximum number of packets an adversary could inject in a round. There are two regular models of adversaries in adversarial queuing. When there is no limitation imposed on intervals time to average packets, then this category of adversaries is called leaky- bucket. If we average only over intervals of a constant length, then it is called window size; such adversaries are of window type. In the concept of adversarial queuing, window adversaries were first used by [9] and leaky-bucket adversaries by [6]

Anantharamu et al. [3] proposed how to compare dynamic broadcasting versus adversaries that are unbounded in the sense that they can inject packets into randomly stations with no restrictions on their numbers nor rates of injection. They gave a deterministic algorithm optimal with respect to struggle performance, when measuring either the maximum queue size or the total number of packets in the system. In addition, they showed that the algorithm is stochastically optimal for any expected injection rate smaller than or equal to 1.

Anantharamu et al. [4] studied broadcasting on multiple access channels with

jamming and dynamic packet arrivals. The communication platform is represented by adversarial models which specify restricts on packet arrivals and jamming. They consider deterministic distributed broadcast algorithms and give upper bounds on the (worst-case) packet latency and the number of queued packets that relating to the defining adversaries. Packet arrivals are specified by number of packets that can arrive in one round and the rate of injections. Jamming is limited by the number of consecutive rounds that can be jammed and by the rate with which the adversary can jam rounds.

CHAPTER III

TECHNICAL PRELIMINARIES

A multiple access channel is a broadcast network that allows for each node to transmit its message to all the other nodes in one round. We consider such synchronous networks when executions of communication algorithms are structured as a series of rounds.

It is assumed that a multiple access channel consists the communication medium and some nodes connected to it. Nodes that have packets to transmit are called *active* and otherwise they are *passive*.

The set of nodes may be considered to be either static or dynamic, depending on our preference of the model. In the *static* model, there is a fixed set of some n nodes. In this case, the nodes usually have fixed *names* that identify them, which are unique integers in the range $[0, n-1]$. In the *dynamic* case, nodes may join and leave the system, they do not have fixed names, and only active nodes execute a broadcast algorithm while passive nodes are neutral. There is an unbounded supply of passive nodes in the dynamic model, and a packet may be injected into a passive node, which immediately becomes active.

It is said that a node *hears* a transmission when it receives it successfully. The critical property of multiple access channels is how multiplicity of concurrent transmissions affects hearing messages.In general, there are three possible cases of relevant multiplicities of transmissions in a round: no nodes transmit, exactly one node transmits, and multiple (more than one) nodes transmit.When multiple nodes transmit in a round then this is referred to as a *collision* occurring in this round.

When no node transmits in a round then the channel is *silent*, which is reflected by the corresponding *silence* feedback received from the channel by each node. When only one node transmits then all the other nodes that are actively participating in the execution hear the transmitted message, which is the feedback they receive from

the channel. When multiple nodes transmit in the same round then this creates a collision, to the effect that the messages interfere with each other and none can be heard by any attached node. In this case, the attached nodes hear a *collision signal* as the feedback from the channel.

Multiple access channels come in two variants, which are determined by the feedback obtained from the channel when a message is not heard. The channel *without collision detection* has the property that the silence and collision signals are identical, while in the channel *with collision detection* they are different, so the nodes can distinguish between the two corresponding cases. The feedback obtained from the channel in a round does not depend what a node does in this round; in particular, a transmitting node and a pausing node receive the same feedback, and if a message is heard then the transmitting node hears the message as well.

We consider dynamic broadcasting when packets are injected by adversaries. Various types of adversaries considered in this paper, they are all specializations of the leaky bucket adversarial concept. A leaky-bucket adversary is determined by the following two key numeric parameters. One is *injection rate*, denoted ρ, which is a real number satisfying $0 < \rho \leq 1$. The other is *burst* component, denoted b, which is a positive integer. Together they make the *type* (ρ, b) of the adversary.

In each round, the leaky-bucket adversary of type (ρ, b) injects a number of packets, which can be visualized as occurring in two stages: first the number of new packets to be injected is determined for the round, and then these many packets are injected into some nodes. The type of a leaky-bucket constrains the numbers of packets injected in each round as follows: for each contiguous time interval τ of $|\tau|$ rounds, the number of packets injected into the system during the rounds in τ is at most $\rho \cdot |\tau| + b$.

This captures two postulates for the adversary of type (ρ, b), that it constraints both the average number of injected packets in large intervals and also the bursts of

packets injected in small intervals. Indeed, the injection rate ρ can be interpreted as the average number of packets injected in a round when averaging over large intervals. Also, at most $\lfloor \rho + b \rfloor$ new packets can be generated in one round, because the number of new packets is a nonnegative integer and $\rho \cdot |\tau| + b = \rho + b$ with $|\tau| = 1$.

One may consider variants of adversarial models defined by where the new packets are injected, after their number has been decided on in a round. This naturally depends on whether the adversary is *static*, which applies for the static model of a fixed set of nodes, or it is *dynamic* and so my create new active nodes and add them to the system. Historically, the most studied case was of static adversaries with no restrictions on which nodes obtain the newly generated packets. A lack of restrictions means that the adversary is to model an environment to study worst case performance of algorithms. This adversarial model was considered in [2, 3, 4, 11, 12].

The adversarial model may additionally restrict the number of passive nodes that are activated in a round. At most $\lfloor \rho + b \rfloor$ passive nodes can be activated in a round because each activation requires at least one packet.

An upper bound k on the number of stations that can be activated in a round can be considered an independent parameter of the adversary, which is then called *k-activating*. In particular, Anantharamu and Chlebus [1] considered 1-activating dynamic adversaries. When we refer to an adversary as static or dynamic only, without specifying a bound on number of activated nodes, then this means that no restriction on the number of activations per round applies, except for the bound $\lfloor \rho + b \rfloor$.

In this thesis, it is emphasized on 1-activating static adversaries. The motivation for activating at most one node in a round comes from the real-world applications when multiple access channels model local area networks and is as follows. It expects that nodes that are active usually have multiple packets to broadcast, so new packets are typically injected into nodes that are already active. It also expects that nodes

that are passive stay such through the round, although rarely a passive node gets activated and generates packets to broadcast. This latter interpretation means also that having multiple passive nodes activated in a round is such a rare event then it can be disregarded without distorting the performance of broadcasting as modeled in simulations. The additional advantage of 1-activating adversaries is that it is possible to broadcast in a distributed deterministic manner with bounded packet latency for positive injection rates, against such adversaries, even in dynamic systems, as was shown by Anantharamu and Chlebus [1].

An alternative to adversaries conducive to studying worst case performance, It can build randomness into an adversarial model, which allows to capture the average performance. It refers to such adversaries as *randomized* ones and if randomness is not uses in any manner then the adversary is *worst-case*. A randomized leaky bucket adversary is again determined by its type (ρ, b). Randomness can affect the behavior of randomized leaky-bucket adversaries in two ways: in the process of determining the number of packets injected in a round, and in selecting nodes where the packets are injected. Each randomized adversary can also have an upper bound on the number of stations activated in a round as an additional parameter.

Now It defines *randomized leaky-bucket adversaries* of type (ρ, λ, b), where $0 < \rho \leq 1$ and $0 < \lambda$. First we specify how to determine the number of packets injected in a round. Let D be a variable which we call the *bucket*. Its purpose is to remember the recent injections of packets and it is interpreted as follows: $\lfloor D \rfloor$ is the maximum number of packets that can be injected in the current round.

Initially, D is set to b. The number of packets injected in a given round by the randomized leaky-bucket adversary is determined as follows.

- First bucket is upgraded D by assigning $D \leftarrow \min[D + \rho, b]$. This means that the inequality $D \leq b$ is always satisfied.

- Next, it is generated a non-negative integer X with the Poisson distribution

with parameter λ, which means that

$$\Pr(X = i) = e^{-\lambda} \cdot \frac{\lambda^i}{i!} ,$$

for each integer $i \geq 0$, see [17, 19]

- Then the number j of packets injected in this round is determined as $j = \min\{\lfloor D \rfloor, X\}$.
- Finally, D is updated to $D - j$, which means that the inequality $D \geq 0$ is always satisfied.

This completes the manipulations of D in this round.

Proposition 1 *When a randomized adversary of type (ρ, λ, b), injects packets in a sequence of consecutive rounds then the numbers of packets generated in each round satisfy the requirements of the worst-case adversary of type (ρ, b), for $0 < \rho \leq 1$ and $0 < \lambda$.*

Proof: It needs to show that the randomized adversary injects at most $\rho \cdot |\tau| + b$ packets in each interval τ. Injecting a packet results in decrementing D to $D - 1$. So the number of packets that can be injected is accounted for by all the increments of D and the initial value of D. The bucket D is incremented by at most ρ in the beginning of each round in τ, for the total of at most $\rho \cdot |\tau|$ during all the rounds in τ. Additionally, the bucket satisfies $D \leq b$ before the first round of τ begins. ■

The Poisson distribution with parameter λ has the property that $\mathbb{E}[X] = \lambda$, but the expected number of packets injected in a round is smaller than λ because D is the additional upper bound.

Let Y be the random variable equal to the number of packets injected in a round by a randomized adversary of type (ρ, λ, b). It would be interesting to find $\mathbb{E}[Y]$ beyond the estimate $\mathbb{E}[Y] < \lambda$.

For injection rate $\rho < 1$, one could simply use $\lambda = \rho$ in experiments. An alternative would be to use such $\lambda > \rho$ that $\mathbb{E}[Y] = \rho$. Then the correspondence between the worst case adversary of type (ρ, b) and the randomized adversary of type (ρ, λ, b) would be even closer than what Proposition 1 gives.

After having determined the number of packets j to inject, It needs to determine where the j new packets are injected. Let the adversary be k-activating, where $k \leq \lfloor b + \rho \rfloor$. There are the dynamic and statice cases.

First the dynamic case. In a given round, consider k new *virtually active* nodes. The set of active and virtually active nodes makes the set of nodes *eligible* for this round. Each among the new j packets is assigned an eligible station uniformly at random, with assignments of different packets made independently. If a virtually active station receives a packet then it becomes added to the system as active otherwise it disappears.

Next, the static case. Let ℓ be the number of passive nodes among all the n nodes. The adversary selects $\min\{\ell, k\}$ such stations uniformly at random as *virtually active*. In particular, if $\ell = k$ then all passive nodes become virtually active and if $\ell = 0$ then there is no virtually active node. Again, the set of active and virtually active nodes makes the set of nodes *eligible* for this round, and also each of new i packets is assigned an eligible station uniformly at random, with assignments of different packets made independently.

Each execution of a randomized leaky bucket adversary of a given type satisfies the specialization of the worst-case leaky bucket adversary of this type, in the sense that each pattern of injections of packets satisfies the constrains defining the worst-case adversary. It follows that bounds on performance metrics of deterministic algorithms against the deterministic leaky bucket adversary apply to the randomized version of the adversary of the same type.

The randomized leaky bucket adversary of a given type can be simulated by

generating uniform distributions on finite ranges and a Poisson distribution for a given injection rate interpreted as the expectation of this Poisson distribution.

CHAPTER IIII

BROADCAST ALGORITHMS

It is mostly concerned with deterministic distributed algorithms. Such an algorithm is to perform dynamic broadcasting, in the sense that packets get injected into the nodes and the nodes have these packets heard on the channel by successful transmissions.

An algorithms is *activation based* when a station without a packet to transmit ignores the feedback from the channel and starts participating actively only starting from a round when it gets *activated* by having a packet injected into it and stays such as ling as it has packets to transmit. An algorithm is *full sensing* when all the nodes listen to the channel at all times.

The actions in a round of a node executing a full sensing algorithm or activation based when a station has a packet to transmit are as follows. The node first either transmits a packet or pauses, as determined by its state. Then the nodes obtains the feedback from the channel, the same as all the other stations. Next, the node may have a number of packets injected into it in the round. Finally, the node performs local computation, which can be interpreted as a state transition. This involves the following actions. If packets were injected then they are enqueued in the local queue. Local variables are updated, depending on what occurred in the round up to this moment. The node decides if to transmit in the next round and if so the it builds a message to be transmitted. If the queue includes packets then the message may include a packet.

A message transmitted on the channel may include at most one packet but it may consist of only control bits. An algorithm may not use control bits at all, then the messages transmitted int he course of its execution consist of only packets. An algorithm that has stations transmit messages with control bits is called *adaptive*. The messages in an execution of such an algorithm may consists of only control bits

but for a non-adaptive algorithm if a node does not want to transmit a packet then the node does not transmit at all.

There are three groups of algorithms: deterministic token ones, deterministic ad hoc ones, and randomized.

Algorithms designed specifically for a multiple access channel with a fixed set of nodes attached to the channel each with a unique name may operate by having the nodes exchange a token. The token visiting a node allows the node to transmit, which prevents collisions. Such algorithms could be called *token* ones; see[12, 11, 4]. The other paradigm is for environments without a fixed set of named nodes attached to the channel, and instead nodes are dynamically generated and assigned temporary implicit names. This works for a setting in which at most one new node is added to the system in a round. Such algorithms could be called *ad-hoc* ones; see [1]. Multiple access channels with a fixed set of named nodes attached to them are more effective than channels executing ad hoc algorithms, in that the former can handle injection rate 1 in a stable manner [11] and arbitrary injection rates smaller than 1 with bounded packet latency [3, 4, 2] while ad hoc algorithms cannot handle injection rates that are at least 3/4 with bounded packet latency [1].

What follows are brief specifications of the algorithms that will be compared in experiments. There are two groups of algorithms: deterministic token ones and randomized.

4.1 Distributed Deterministic Algorithms

Let us begin with the considered token algorithms. Algorithm Move-Big-To-Front (MBTF) Algorithm Move-Big-To-Front (MBTF) is an adaptive algorithm for channels without collision detection. Each station maintains a list of all stations in its private memory. A list is initialized to be sorted in the increasing order of names of stations. The lists are manipulated in the same way by all the stations so their order is the same. The algorithm schedules exactly one station to transmit in a round,

so collisions never occur. This is implemented by having a conceptual token assigned to stations, which is initially assigned to the first station on the list. A station with the token broadcasts a packet, if it has any, otherwise the round is silent. A station considers itself big in a round when it has at least n packets; such a station attaches a control bit to all packets it transmits to indicate this status. A big station is moved to the front of the list and it keeps the token for the next round. When a station that is not big transmits, or when it pauses due to a lack of packets while holding the token, the token is passed to the next station in the list ordered in a cyclic fashion. Algorithm MBTF was introduced in [11] and showed to be stable for injection rate 1. It is stable for injection rate 1 and has bounded packet latency for injection rates smaller than 1 .

Algorithm Round-Robin-Withholding (RRW) is a full-sensing (non-adaptive) algorithm for channels without collision detection. It operates in a round-robin fashion, in that the stations gain access to the channel in the cyclic order of their names. Once a station gets access to the channel by transmitting successfully, it withholds the channel to unload all the packets in its queue. A silent round is a signal to the next station, in the cyclic order of names, to take over. Algorithm RRW was introduced in [12] and showed to be universal, that is, stable for injection rates smaller than 1. It has bounded packet latency for injection rates smaller than 1.

Algorithm Search-Round-Robin (SRR) is a full-sensing (non-adaptive) algorithm for chan- nels with collision detection. Its execution proceeds as a systematic sweep across all the stations with the goal to identify these with packets, with such search performed in the cyclic order. When a station with a packet is identified, the station unloads all its packets one by one. A silent round triggers the sweep to be resumed. We apply binary search to identify the next station. The binary search is implemented using collision detection. When we inquire about a segment of stations, then all the stations with packets that are in the segment transmit in the round. A search

is completed by a packet heard. A silence indicates that the segment is empty of stations with packets. A collision indicates that multiple stations are in the segment: this results in having the segment partitioned into two halves, with one segment processed next immediately while the other one is pushed on a stack to wait. A transition to the next segment occurs when the stack gets empty. Algorithm SRR is introduced by [12]. It has bounded packet latency for injection rates smaller than 1.

The algorithms RRW and SRR can be modified by using the approach that could be called "older-go-first," which was introduced by Anantharamu et al.[2]. Each of the algorithms has executions that can be interpreted as having a token that traverses all the nodes in a round robin manner. Call each such a cycle a *phase*. The packets that are injected in a phase are considered "new" during the phase and become "old" when the next phase starts. In the course of a phase, the new packets are ignored and only the old ones are broadcast. The resulting algorithms are called Older-First-Round-Robin-Withholding (OF-RRW) and Older-First-Search-Round-Robin (OF-SRR), respectively.

4.2 Ad-hoc Deterministic Algorithms

Next we present the ad-hoc algorithms; they all were proposed by Anantharamu and Chlebus [1]. Algorithm Counting-Backoff is a non-adaptive activation based algorithm for channels with collision detection.

Active nodes are stored on a virtual stack and an active station remembers its distance from the top of the stack. It was sown that the algorithm has bounded packet latency when injection rate is smaller than 1/3. Finally, there is the algorithm Queue-Backoff which is an adaptive activation-based algorithm for the channel without collision detection that has bounded packet latency for injection rate 1/2.

These two algorithms do not use the names of nodes, even if they available. The bounded packet latency holds even when there is no fixed set of nodes attached to the channel and the adversary has the power to create new stations by injecting packets

into them, but at most one new station in a round.

4.3 Randomized Algorithms

Finally, it will simulate the behavior of two randomized algorithms. These are the Binary-Exponential-Backoff (BEB) and Quadratic-Backoff (QB). The backoff protocols are considered in their windowed versions. A node tries to broadcast the packets in the order of their injection. When a new packet is available then the node selects a round uniformly at random in the current window then waits for this round to occur and transmits the packet. The first window is of size 1, which means that a new packet is transmitted immediately. When a transmission is heard then the next packet is processed, otherwise when there is a collision then the window is increased and a round in it is selected from it uniformly at random. For Binary-Exponential-Backoff, the ith window size was determined as 2^i and forQuadratic-Backoff, the ith window size was defined to be i^2.where the round of a transmission is drawn from a time interval uniformly at random.

There is an option to consider these algorithms with an upper bound on the window size, as binary exponential backoff is used in the implementation of the Ethernet. For instance, the ith window size for Binary-Exponential-Backoff could be $2^{\min(10,i)}$, which is exactly as in the Ethernet, and for Quadratic-Backoff, the ith window size could be $(\min(i, 32))^2$. Observe that with these choices of constants the maximum size of a window is the same in BEB and QB. The two randomized protocols are proposed in [14, 15, 16].

CHAPTER V

THE METHODOLOGY OF EXPERIMENTS

We consider systems with a fixed number of stations attached to the channel. only the static case The number of nodes n is a parameter of a simulation. This is typically a small positive integer, say, in the range [1, 100]. We will consider only 1-activating adversaries. The simulations will simply mimic randomized leaky-bucket adversaries: but only static 1-activating adversaries.

It needs to specify how to terminate an experiment. For example, we may designate a certain constant L as a parameter. This constant L denotes the number of rounds during which new packets are injected for which is measured packet latency. The precise meaning of L is as follows: packets are injected during the first L rounds of the execution, and It measures packet latency of only these packets, while it is kept injecting packets after the round L until all the packets injected in the first L rounds have been heard on the channel, and then we terminate the simulating execution. This parameter L is a reasonably large positive integer, like $L = 100{,}000$.

We need to relate outcomes of experiments to theoretical bounds on packet latency. Such bounds are in the literature for the cases of static adversaries and for dynamic 1-activating adversaries. Here one may notice that the known three deterministic algorithms for the dynamic 1-activating adversaries can handle injection rates up to 1/2 or less, see [1]. We expect that fixed set of nodes should have a stabilizing effect on executions of a broadcast algorithm.

Conjecture 1 *The three ad-hoc algorithms for the* 1*-activating adversary provide bounded packet latency for each injection rate smaller than* 1 *in the static model, that is, when there is a fixed set of nodes.*

If this conjecture holds true, then the plots of the respective bounds could be used in charts to relate outcomes of experiments to these very "theoretical bounds on packet latency" as the simulations will be for 1-activating static randomized adversary.

5.1 Specific Experiments

There are five deterministic algorithms presented above. (The randomized algorithms interest us only as point of reference in comparisons with deterministic algorithms.) For each of them we should have a theoretical bound on packet latency, say worst-case, then It may measure, say, both worst case or average packet latency in a simulation of the randomized adversary. To obtain a chart, it may plot the corresponding three curves for different injection rates, say for injection rates that are multiples of 1/20. These are the simplest experiments in which one looks at each algorithm independently and compare the theoretical bounds to the bounds obtained through simulations.

More involved experiments will be about comparing different algorithms. For example, deterministic agains randomized, to verify advantages and disadvantages of their designs. Such comparisons may be determined by having parameters of the system vary: injection rate, number of stations, burstiness, etc, while keeping the others constant. A chart will have one curve for an algorithm, say representing worst-case packet latency as measured while simulating the suitable randomized adversary.

5.1.1 Deterministic Distributed Algorithms

Next we we present outcomes of experiments of deterministic algorithms.

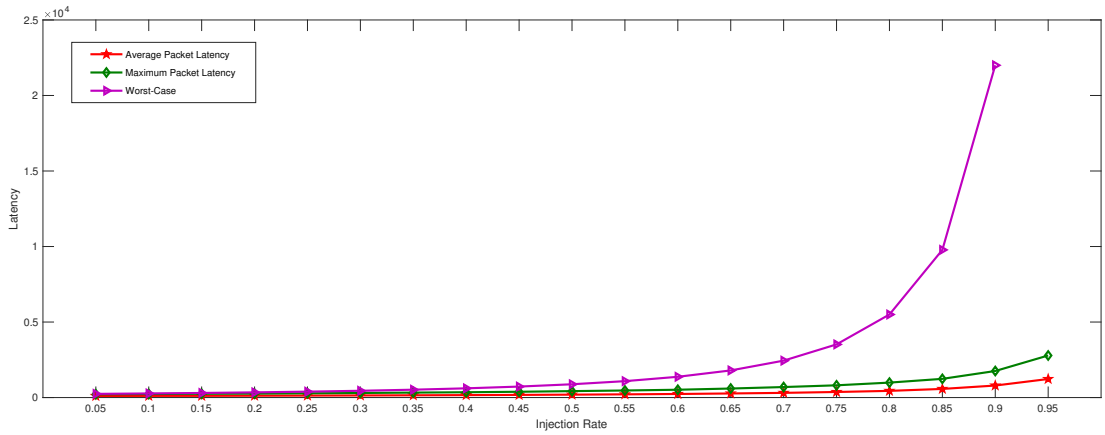

Figure 5.1: Algorithm RRW. Observed values of average, maximum packet latency and worst-case for varying injection rates. Parameter values:$R = 100,000; n = 200; b = 20$.

Figure 5.1 and Table 5.1 on page 21 present outcomes of simulations for the deterministic algorithm Round Robin Withholding (RRW). It is found the average latency, maximum packet latency and upper bounds (wrost-case) with different number of injection rates and parameters: $L = 100,000; n = 200; b = 20$; we could see when injection rate increases, the number of packets that injected per round are increasing also. At the beginning we can notice that when injection rate was 0.05, the values are closed for each other because the number of packets that inject each round are very small. For the middle injection rate like 0.5 the values of latencies are becoming approximately double for each other respectively. At the high injection rate like 0.95 the maximum latency becomes about triple bigger than average value, but the theoretical bound becomes approximately 31 times bigger than maximum latency and about 70 times than average latency.

Table 5.1: Observed values of average, maximum packet latency and worst-case for varying injection rates. Parameter values: $R = 100,000; n = 200; b = 20$.

Average Packet Latency	Maximum Packet Latency	Upper Bounds(Worst-Case)
105	216	244
111	234	272
114	250	304
123	260	344
131	282	391
139	302	449
150	328	521
162	352	611
176	384	727
193	426	880
213	468	1086
237	518	1375
271	598	1796
311	696	2444
369	811	3520
443	990	5500
572	1239	9778
796	1758	22000
1228	2785	84000

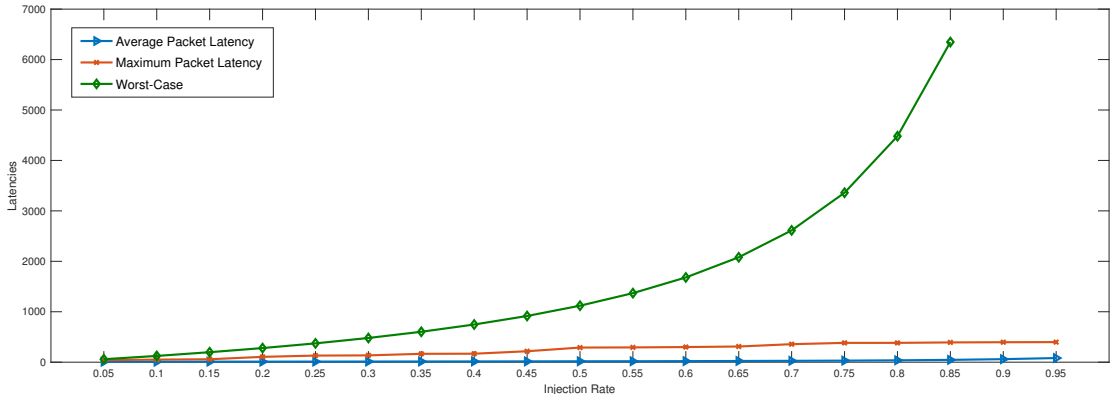

Figure 5.2: Algorithm MBTF. Observed values of average, maximum packet latency and worst-case for varying injection rates. Parameter values:$R = 100,000; n = 20; b = 8$.

Figure 5.2 represents outcomes of experiments of the deterministic algorithm Move-Big-To-Front (MBTF). We conclude that the average latency, maximum latency and theoretical bound for each injection rate with parameters: $L = 100,000; n = 20; b = 8$. We can observe that when injection rate is very low like 0.05, the average and maximum latency could see them in close values then they they still closed together while the upper bound gradually increasing until 0.5. After that upper bounds (worst-case) goes high because number of packets that injected are high and this algorithm needs time to seek for station that has packets no less than n (number of stations), it is effective with high injection rates. Another observation is which average packet latency is no more than 100 rounds that is because the burstiness and number of station are small. Indeed, this deterministic algorithm is slower than RRW.

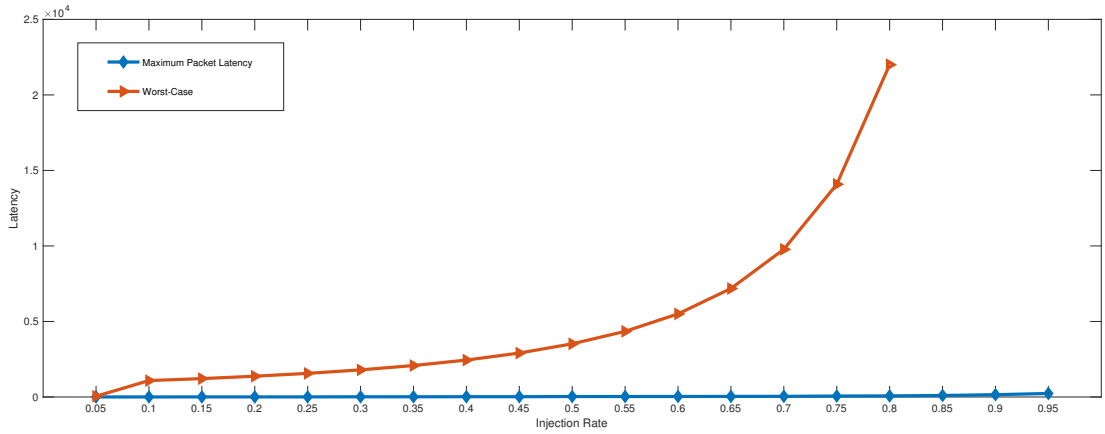

Figure 5.3: Algorithm SRR. Observed values of average, maximum packet latency and worst-case for varying injection rates. Parameter values:$R = 100,000; n = 200$.

Figure 5.3 is about algorithm SSR. It gives the maximum packet latency and worst-case upper bound without average latency because it is very small and close to 0. Here the parameters that are used: $L = 100,000; n = 200; b = 20$ with varying injection rates; I see that the maximum latency is increasing slowly; however, the theoretical bound started higher than the maximum packet latency then it goes very big. Additionally, when the injection rate close to 1 the maximum latency with this algorithm is around the number of stations that maybe because the binary search that is applied in this algorithm. We can see that this algorithm has smaller packet latency than RRW and MBTF.

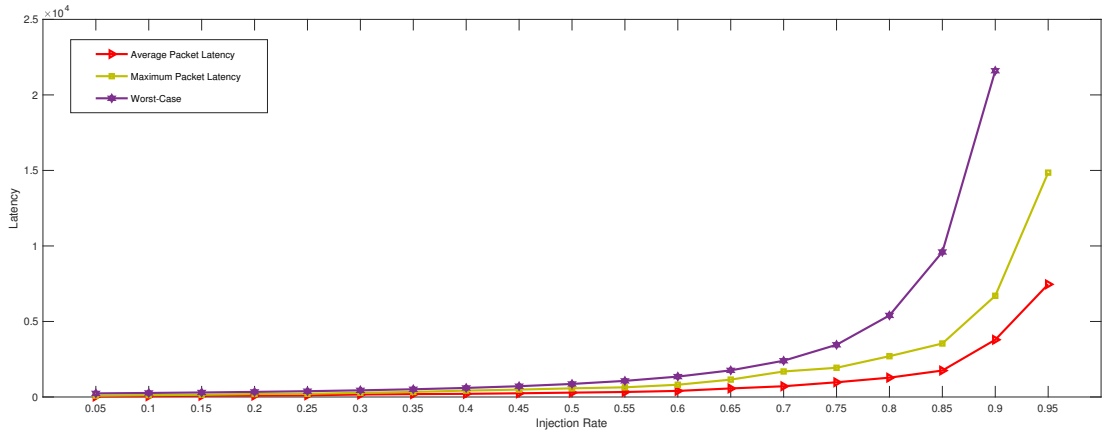

Figure 5.4: Algorithm OFRRW. Observed values of average, maximum packet latency and worst-case for varying injection rates. Parameter values:$R = 100,000; n = 50; b = 8$.

Figure 5.4 is about a deterministic algorithm OFRRW. It gives the average latency, maximum packet latency and upper bounds (worst-case) with different number of injection rates with parameters: $L = 100,000; n = 50; b = 8$; here we reduced the number of stations rather than I used with RRW.

We can see that when injection rate increases, the number of packets that injected per round are increasing too. At the beginning we can notice that when injection rate is 0.05, the maximum latency is three times bigger than the average ,and upper bound is six times bigger the the average latency and so on where each step that the injection rate increased the different among them is triple times. We can conclude that this algorithm is slower than RWW and SRR, but it is faster than MBTF in the settings mentioned. If we play with number of stations, we could get different observation; we will see that.

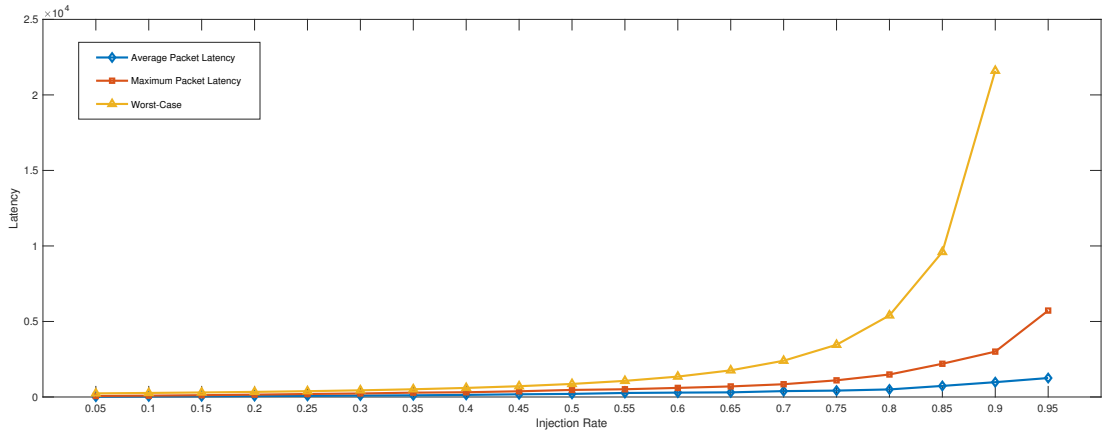

Figure 5.5: Algorithm OFSRR. Observed values of average, maximum packet latency and worst-case for varying injection rates. Parameter values:$R = 100,000; n = 50; b = 8$.

Figure 5.5 is about a deterministic algorithm OF-SRR. Here It is applied the same setting for OF-RRW. It followed the same behavior of OF-RRW where the average packet latency, maximum latency started very small while the worst-case is started high because OF-SRR and OF-RRW have the same theoretical bounds. Finally, observe that this algorithm has smaller packet latency than SRR and RRW while it similar in this respect to OF-RRW, but it is faster than MBTF.

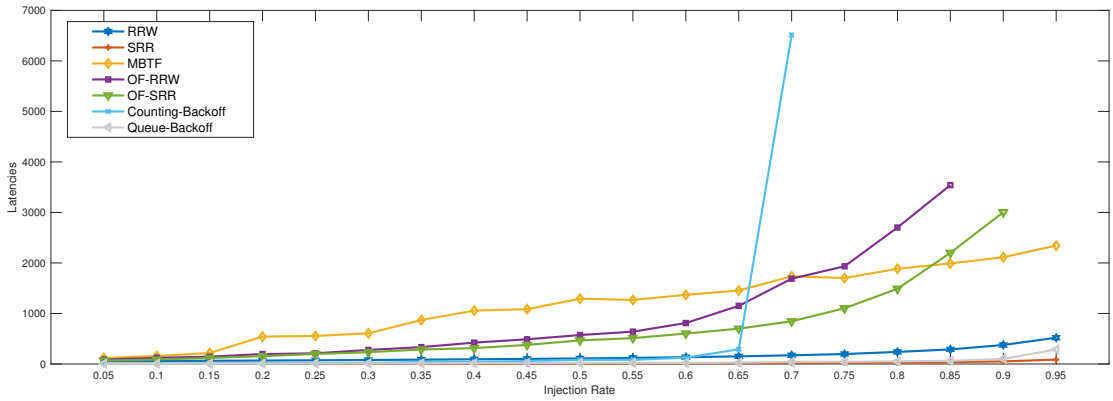

Figure 5.6: Seven deterministic algorithms. Observed values of maximum packet latency for varying injection rates. Parameter values:$R = 100,000; n = 20; b = 20$.

Figure 5.6 compared seven deterministic distributed algorithms two backoff ones and other five to see which algorithm is better and which one is the worst among all deterministic protocols. We can observe that Search-Round-Robin algorithms is the faster among all deterministic algorithms while Counting-Backoff is the slower protocol.

5.1.2 Randomized Versus Deterministic Distributed Algorithms

We tested the randomized algorithms Binary-Exponential-Back-off and Quadratic-Backoff in simulations similarly as deterministic distributed algorithms, with varying injection rates with parameters: $R = 100,000; n = 20; b = 20$.

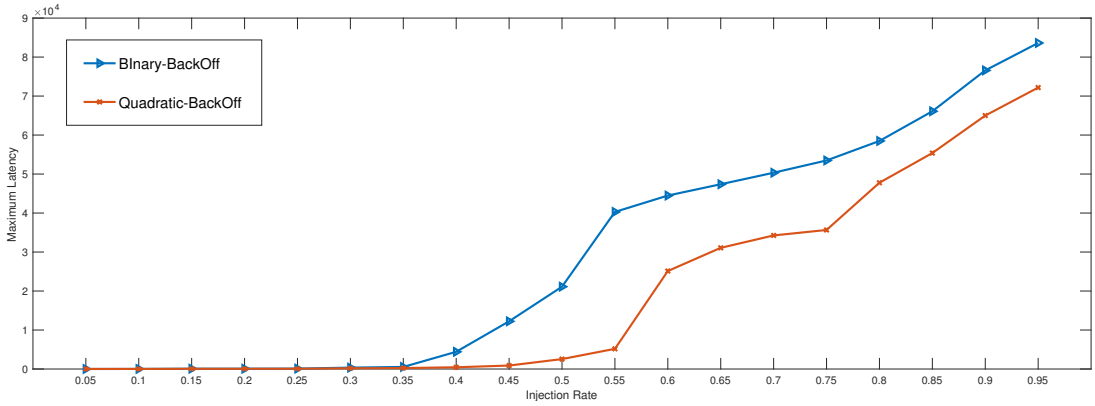

Figure 5.7: Randomized algorithms. Observed values maximum packet latency for varying injection rates. Parameter values:$R = 100,000; n = 20; b = 20$.

Figure 5.7 presents outcomes of experiments for th=wo kinds of backoff algorithms: exponential and quadratic. We can notice that the maximum latency are very closed together when the injection rate less than 0.35 while after the injection rate increased the maximum latency for algorithm Binary-Exponential-BackOff goes higher than Quadratic-BackOff protocol. We can notice that algorithm Quadratic-BackOff is the winner when we compare as randomized protocols; however, Search-Round-Robin algorithm is the winner and Binary-Exponential-BackOff is the worst among all algorithms that considered in this thesis with varying injection rates.

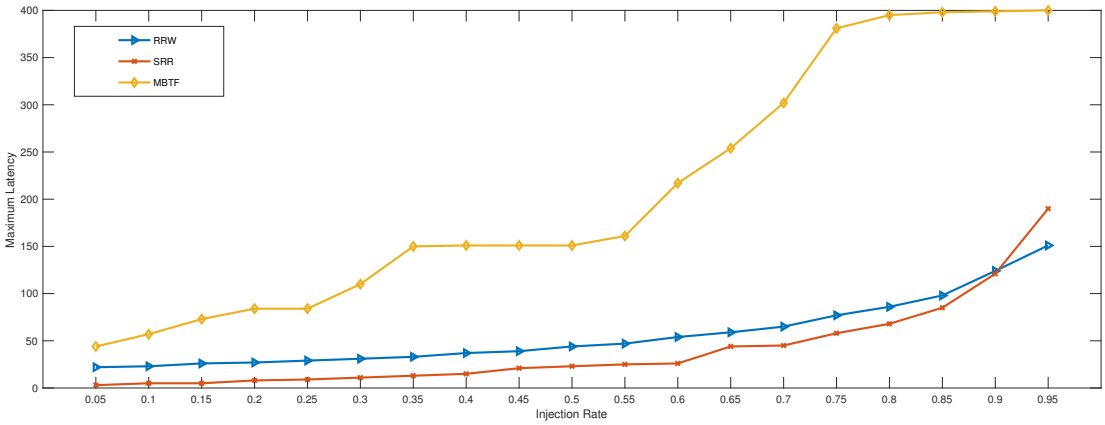

Figure 5.8: Three deterministic algorithms. Observed values of maximum packet latency for varying injection rates. Parameter values:$R = 100,000; n = 20; b = 20$.

We discovered that the maximum packet latency with the same setting that are used in randomize algorithms in Figure 5.8 to see the differentness among all non-randomize and randomize algorithm by using the parameter maximum latency.

We conclude that MBTF is the slower one than all others protocol and Binary-Exponential-Backoff and Quadratic-Backoff should very fast algorithms rather than with others; however, in general, the Quadratic-Backoff is the faster one.

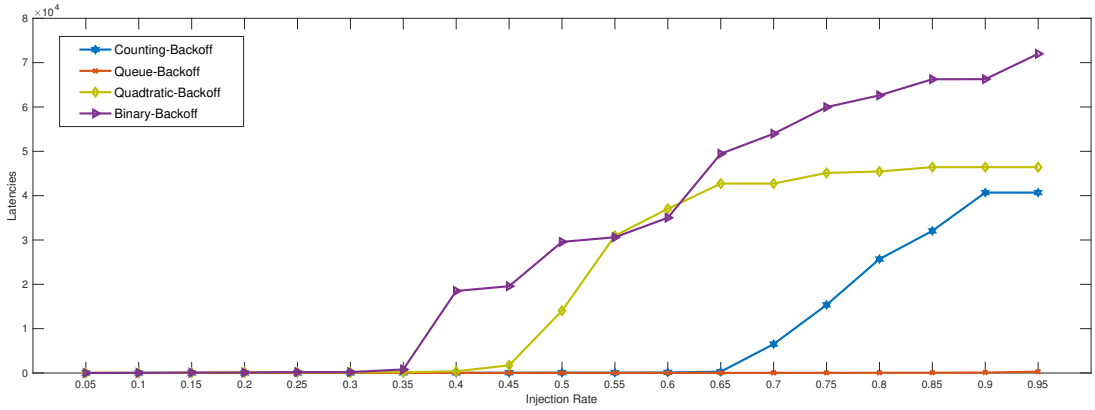

Figure 5.9: Four Backoff algorithms. Observed values of maximum packet latency for varying injection rates. Parameter values:$R = 100,000; n = 80; b = 20$.

Figure 5.9 depicts measured maximum packet latencies for four distributed Backoff protocols with varying injection rates. As we can see that counting protocol is so faster one among them and still Binary-Exponential-Backoff is the slowest one while Quadratic-Backof and Counting-Backoff are in between then Queue-Backoff is the winner in this respect. As mentioned before when the injection rate is large, the the packet latency for Binary-Exponential-Backoff algorithm is too large.

5.1.3 Varying Burstiness

We consider another scenario of varying burstiness with fixed number of stations, rounds and injection rate.

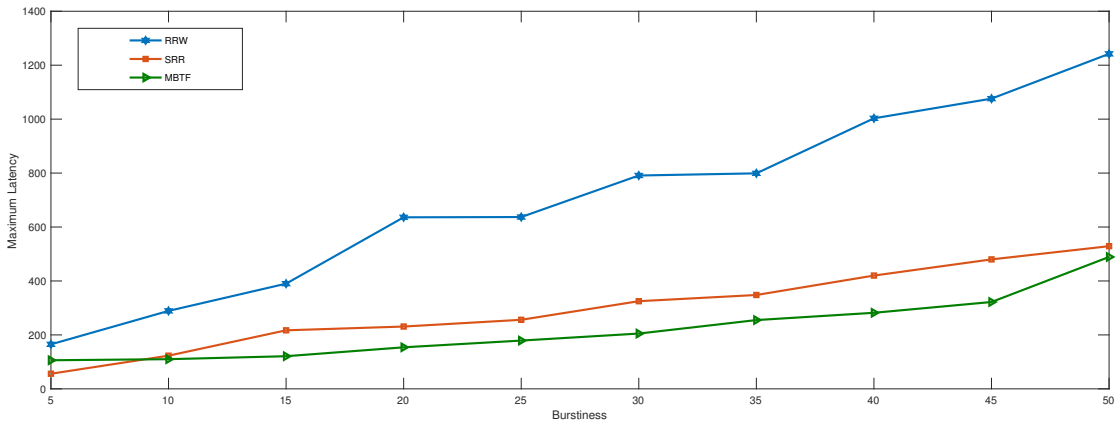

Figure 5.10: Three deterministic algorithms. Observed values maximum packet latency for varying Burstiness. Parameter values:$R = 100,000; n = 10$; injection rate is 1.

Figure 5.10 considers a small number of stations. Then MBTF is beaten while RRW is winner. That means MBTF is more sensitive by number of stations, but RRW is more sensitive by burstiness. SRR appears to combine sensitively to the number of stations and burstiness.

5.1.4 Varying Numbers of Stations

This section is illustrated different number of stations with fixed burstiness, rounds and injection rate=0.75 to figure out the maximum packet latency for some deterministic and randomized algorithms.

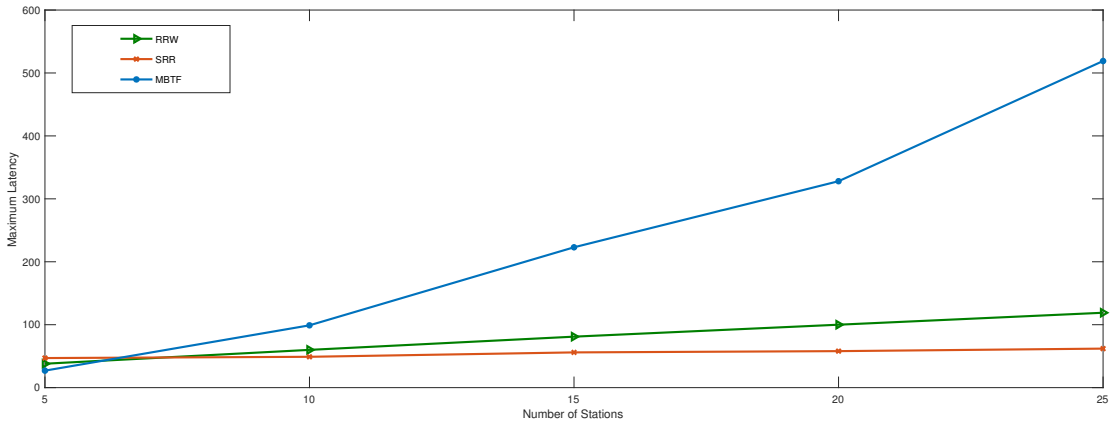

Figure 5.11: Observed values maximum packet latency for varying number of stations. Parameter values:$R = 100,000; b = 20$; injection rate $= 0.75$.

Figure 5.11 demonstrated how MBTF is beaten because this protocol is more effective with number of stations while RRW affects by number of stations, burstiness and injection rate. We can see that SRR here becomes the winner because I fixed number of burstiness, rounds, and injections. It is more sensitive by how many packets that injected in each stations; also it is effective by number of stations, but more sensitive with injection rate and maximum packets that could inject per round.

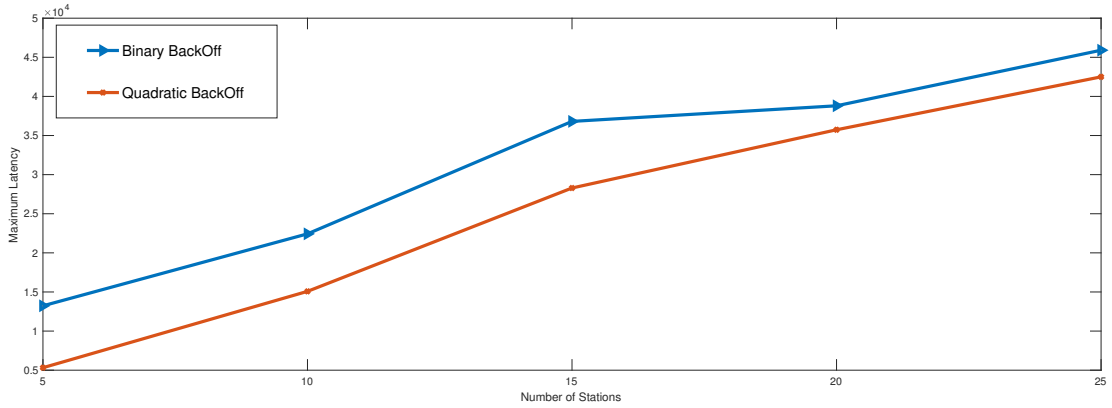

Figure 5.12: Observed values maximum packet latency for varying number of stations. Parameter values:$R = 100,000; b = 20$; injection rate $= 0.75$.

The randomized protocols are also effective by different number of stations as we can see in the Figure 5.12 with above settings setup that maximum latency for Binary-Exponential-BackOff and Quadratic-Exponential-BackOff algorithms are increased high concurrently the number of stations are increased, but, in general the deterministic protocols are winner in order to compare them with randomized when applied to this scenario.

5.2 Throughput

In this section is found the number of packets that are injected and number of packets that are sent; it is another way to check the performance of each algorithm. The settings that used to figure out these numbers are $n = 200; b = 20, L = 100,000; \lambda = \rho/0.3$.

Table 5.2: Throughput

Algorithm	Packets Injected (ρ=0.05)	Packets Sent (ρ=0.05)	Packets Injected (ρ=0.995)	Packets Sent (ρ=0.995)
RRW	5019	5013	99519	95456
SRR	10037	5019	199037	99518
MBTF	10036	5012	199037	92648
OF-RRW	5018	5018	99518	96085
OF-SRR	5019	4590	99619	48666
BEB	4915	4915	96764	56668
QEB	4928	4928	97218	35396

Table 5.2 summarizes the outcomes of experiments which reflect how each protocol works under low and high injection rates. We can notice that algorithm Round-Robin-Withholding and Old-First-Round-Robin-Withholding perform comparatively well. The efficiency is approximated about 99% for RRW and OF-RRW. Search-Round-Robin and Move-Big-To-Front efficiency are about 50% because it sent about half packets that are injected in both high and low injection rate. The throughput of algorithm Old-First-Round-Robin-Withholding is about 98% when the injection rate is low while the throughput is about 51% when the injection rate is high. Finally, the randomized protocols are doing perfect throughput when set low injection rate where

the efficiency for them is 100%; on the other hand, when we applied high injection rate algorithm Binary-Exponential-BackOff throughput is approximately 58% while algorithm Quadratic-Exponential-BackOff is about 36%.

5.3 Future Work

We considered $\rho = \lambda$ when generating packets per round because. This is only one of possible approaches to create a simulation environment related to worst-case adversarial models.

For future work to improve this project or to get different outcomes by assuming λ is a injection rate and ρ is closed to 1 then see what will get from these changes. Another way to generate packets each round assumes that λ is very close to ρ, but it is more than ρ. It assumes that because we need to keep bucket nonempty and non-full like something in between to guarantee that the simulation could injection number of packets each round without idle or busy. In addition, it considered that the randomized adversarial injections model can activate at most one station per round. There is another way we could modify this approach that the model of adversary could activate more than one station per round. This change should give us different results by measuring packet latency or other factors. Furthermore, it can consider other deterministic and distributed protocols and simulate them and compare their outcomes. This type of study accepts many assumptions and suggestions to generate packets or the way that the adversarial injection model simulates or techniques to activate station

CHAPTER VI

CONCLUSION

We proposed an adversarial model, called randomized leaky bucket adversary, that is amenable to simulations. It is determined by three parameters: injection rate, burstiness, and a parameter of a Poisson distribution. In simulations, it was always a 1-activating static adversary, which means at most one station was activate per round.

We simulated five distributed deterministic protocols Round-Robin-Withholding (RRW), Older-First-Round-Robin-Withholding (OF-RRW), Search-Round-Robin (SRR), Older-First-Search-Round-Robin (OF-SRR), and Move-Big-To-Front (MBTF). The maximum packet latency and average latency was compared with the upper bounds (worst- case) for each algorithm. This allows to see the behavior of each algorithm when the parameters of the environment vary, like injection rates or the numbers of stations.

The simulations included two randomized algorithms Binary-Exponential-Backoff (BEB) and Quadratic-Backoff (QB). The backoff protocols are considered in their windowed versions. They operate under the principle that a station tries to transmit the packets in the order of their injection, and when a new packet is available then the node selects a round uniformly at random in the current window and then waits for this round to occur and transmits the packet. The goal was to investigate by simulating these protocol is to compare them with five deterministic protocols mentioned above.

The result that got are reflecting the behavior of each algorithm. When the number of stations is a big as 200 and the burstiness gets fixed, with 19 incremented injection rates, MBTF is the beaten one while SRR is the winner among them because MBTF protocol is more sensitive with number of stations.

We carried out a comparative study among randomized and deterministic algo-

rithms to see which is faster and which is slower by using same settings for all of them and measuring the maximum packet latency for each. We observed that when the set 10 stations and fixed the burstiness with 19 varying injection rates, BEB is the beaten and SRR is the winner. Here MTTF becomes the winner compared with other deterministic protocol because of the small number of stations we can say that a big observation.

Furthermore, we considered the maximum latency for randomized and deterministic protocols for different numbers of stations and fixed the injection rate to be 0.75 and burstiness to be 20. It turned out that Counting-Back performed worst among all considered deterministic algorithms when the injection rate increased. For randomized protocols, they are also sensitive with number of stations where the maximum latency for algorithm Binary-Exponential-Backoff (BEB) increased higher than the maximum packet latency for algorithm Quadratic-Backoff (QB). Indeed, with different number of station the BEB is the slower protocol among all algorithms that are considered in this project while algorithm Search-Round-Robin is the faster one among them.

REFERENCES

[1] Lakshmi Anantharamu and Bogdan S. Chlebus. Broadcasting in ad hoc multiple access channels. *Theoretical Computer Science*, 584:155–176, 2015.

[2] Lakshmi Anantharamu, Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Deterministic broadcast on multiple access channels. In *Proceedings of the* 29*th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–5, 2010.

[3] Lakshmi Anantharamu, Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Medium access control for adversarial channels with jamming. In *Proceedings of the* 18*th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 6796 of *Lecture Notes in Computer Science*, pages 89–100. Springer, 2011.

[4] Lakshmi Anantharamu, Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Packet latency of deterministic broadcasting in adversarial multiple access channels. *CoRR*, abs/1701.00186, 2017.

[5] Matthew Andrews, Baruch Awerbuch, Antonio Fernández, Frank Thomson Leighton, Zhiyong Liu, and Jon M. Kleinberg. Universal-stability results and performance bounds for greedy contention-resolution protocols. *Journal of the ACM*, 48(1):39–69, 2001.

[6] Matthew Andrews and Lisa Zhang. Routing and scheduling in multihop wireless networks with time-varying channels. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1031–1040, 2004.

[7] Michael A. Bender, Martin Farach-Colton, Simai He, Bradley C. Kuszmaul, and Charles E. Leiserson. Adversarial contention resolution for simple channels. In *Proceedings of the* 17*th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 325–332, 2005.

[8] Daniel S. Berger, Martin Karsten, and Jens Schmitt. On the relevance of adversarial queueing theory in practice. *SIGMETRICS Perform. Eval. Rev.*, 42(1):343–354, June 2014.

[9] Allan Borodin, Jon Kleinberg, Prabhakar Raghavan, Madhu Sudan, and David P. Williamson. Adversarial queuing theory. *Journal of the ACM*, 48(1):13–38, January 2001.

[10] Bogdan S. Chlebus. Randomized communication in radio networks. In Panos M. Pardalos, Sanguthevar Rajasekaran, John H. Reif, and Jose D. P. Rolim, editors, *Handbook of Randomized Computing*, volume I, pages 401–456. Kluwer Academic Publishers, 2001.

[11] Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Maximum throughput of multiple access channels in adversarial environments. *Distributed Computing*, 22(2):93–116, 2009.

[12] Bogdan S. Chlebus, Dariusz R. Kowalski, and Mariusz A. Rokicki. Adversarial queuing on the multiple access channel. *ACM Transactions on Algorithms*, 8(1):5:1–5:31, 2012.

[13] Robert G. Gallager. A perspective on multiaccess channels. *IEEE Transactions on Information Theory*, 31(2):124–142, 1985.

[14] Leslie Ann Goldberg, Mark Jerrum, Sampath Kannan, and Mike Paterson. A bound on the capacity of backoff and acknowledgment-based protocols. *SIAM Journal on Computing*, 33(2):313–331, 2004.

[15] Johan Håstad, Frank Thompson Leighton, and Brian Rogoff. Analysis of backoff protocols for multiple access channels. *SIAM Journal on Computing*, 25(4):740–774, 1996.

[16] Robert M. Metcalfe and David R. Boggs. Ethernet: Distributed packet switching for local computer networks. *Communications of the ACM*, 19(7):395–404, 1976.

[17] Michael Mitzenmacher and Eli Upfal. *Probability and Computing*. Cambridge University Press, Second edition, 2017.

[18] Prabhakar Raghavan and Eli Upfal. Stochastic contention resolution with short delays. *SIAM Journal on Computing*, 28(2):709–719, 1998.

[19] Sheldon M. Ross. *Simulation*. Elsevier, Fifth edition, 2013.