CHALLENGES FOR TFRC AS A REAL-TIME PROTOCOL OVER MULTIHOP WIRELESS AD HOC NETWORKS

by

Kalumbilo Monica Monde

a thesis submitted in partial fulfillment of
the requirements for the degree of
Master of Science

Department of Computer Science
Faculty of Science
University of Zimbabwe
April 06

To my daughter Maria .... Sacrifice is painful!

ABSTRACT

Multihop wireless networks are emerging as a natural extension of the global Internet for scenarios where wired connections are unfeasible, impossible, or undesired. In these networks, nodes cooperate among themselves by relaying data to each other and generally can move at random. The topology of these networks can change rapidly and unpredictably as the mobile nodes change position or the wireless channel condition fluctuates. Such features require robust, adaptive communication protocols that can handle the unique challenges of these multihop networks smoothly.

TCP Friendly Rate Control (TFRC) is a new congestion control scheme for the Internet, intended for use in video streaming and multimedia applications. It aims to send traffic at the same average rate as a TCP/IP sender, but without the sudden variations. However, TFRC is originally tuned to perform well in wired networks and assumes that any loss is due to congestion. Upon any loss, the congestion avoidance algorithm is used to backoff and increase the retransmission timer such that it will not overload the network. The TFRC response function is not well suited for multihop networks, where most packet losses are due to the features of these networks. Therefore, for TFRC to be successfully deployed over multihop wireless networks it must be adapted to handle the unique challenges of these networks. The novelty of this thesis comes from the realization that TFRC must be adapted in such a way that it faces the challenges in combination and not individually.

This thesis details the key challenges for the TCP Friendly Rate Control (TFRC) in multihop wireless networks and outlines the proposed algorithm to solve the involved problems. The result is an adaptive rate estimation TFRC protocol, ARETFRC.

Acknowledgements

This report is a result of my master thesis project. This master thesis is also the last part of my Master of Science degree in Computer Science at the University of Zimbabwe.

I would like to thank the following: VLIR-UNZA project, for the sponsorship and support throughout my program, Mr. B Nyambo whose guidance, supervision, encouragement, friendship and advice made this work possible, Mr. Chrispin Kabuya for discussions and comments especially regarding Ns2 simulator, my family in a special way for their support and love, God bless you.

"Trust in the lord with all your heart,

And lean not unto your own understanding,

In all your ways, acknowledge him,

And he will direct your path."

Author Unknown

CONTENTS

FIGURES

APPENDICES

Chapter 1

Introduction

Wireless networks are becoming increasingly popular among corporate and home users worldwide. Users are looking forward to new technologies that allow them to communicate anytime, anywhere, and using any communication device. Toward this end, wireless communications are foreseen to play a key role in future communication systems. The primary advantages of wireless networks in comparison with their wired counterparts include flexible mobility management, faster and cheaper deployment, and ultimately easier maintenance and upgrade procedures.

The phenomenal growth of wireless communications today is largely driven by wireless networks based on the IEEE 802.11 standard. The IEEE 802.11 networks are gaining momentum toward the dominant data communication technology. This includes data such as delay-sensitive multimedia applications, including real-time multimedia streaming, conversational services (video-conferencing), surveillance, etc. Their commercial use is already expressive in hot spots such as Internet cafes, airports, hotels, convention centers, etc. At home, more and more users are adopting wireless networks as a simple, flexible, low cost, highly convenient solution for interconnecting their various network devices. These applications generally communicate through a single wireless hop since the distance between communicating nodes or between a node and an access point (medium access coordinator) are relatively short. As a result, the 802.11 infrastructure mode is typically used in such communications. This requires a central entity (base station) coordinating the medium access requests.

In addition to the infrastructure mode, users are also starting to enjoy the ad hoc mode of 802.11 in which multiple wireless hops are used to connect two distant nodes. In ad hoc mode, nodes can communicate directly to each other (without a central coordinator) and should relay data to each other in a self-organizing fashion. This configuration is commonly referred to as multihop wireless ad hoc networks or simply *multihop wireless networks*. Thus, 802.11 are also capable of providing communication for connections spanning several wireless hops. This is a remarkable property of 802.11 that can enable effective communication among a community of nodes vulnerable to topology changes as well as fading channels.

Multihop wireless networks are emerging as a natural extension of the global Internet for scenarios where wired connections are unfeasible, impossible, or undesired. In these networks, nodes cooperate among themselves by relaying data to each other and generally can move at random. The topology of these networks can change rapidly and unpredictably as the mobile nodes change position or the wireless channel condition fluctuates. Such features require robust, adaptive communication protocols that can handle the unique challenges of these multihop networks smoothly.

1.1 Motivation

TCP Friendly Rate Control (TFRC) [1] [2] is a new congestion control scheme for the Internet, intended for use in video streaming and multimedia applications. It aims to send traffic at the same average rate as a TCP/IP sender, but without the sudden variations. For these applications TFRC is the natural choice for streaming over multihop wireless networks. Besides, the use of TFRC facilitates fair interoperation with the commonly deployed protocol, TCP. However, the unique features of 802.11, addressed in detail in this thesis, call for adjustments in the upper layer protocols used in the Internet today. In particular, the end-to-end congestion control provided by TFRC to maintain the stability of the Internet is severely compromised in such networks. To adjust TFRC to these networks is therefore vital, as bandwidth is generally a very scarce resource in wireless networks.

TFRC degradation in multihop networks is mostly caused by the mismatch between TFRC and the MAC protocol. Even though the IEEE 802.11 standard has capability to work on ad hoc mode allowing the setup of a completely infrastructureless network, it is not optimized for scenarios with a large number of hops. In fact, the standard specifies short RTS/CTS control frames to ensure that scenarios relaying on at most three hops are not impacted by the well-known hidden node problem. For more than three hops, contention collisions may arise degrading the channel quality. In general, the overhead of RTS/CTS combined with the lossy nature of the wireless channel as well as mobility can lead a TFRC connection to experience very low performance. The reason is that TFRC was originally designed for wired networks where such constraints do not exist. TFRC's dedication to wireless constraints leads to performance degradation.

1.2 Problem Statement

TFRC is designed to support rate-based streaming multimedia and video streaming applications on the Internet. TFRC is tuned to perform well in wired networks where the packet losses are mainly due to congestion. TFRC protocol faces challenges over multihop wireless ad hoc networks which are characterized by losses due to physical and MAC errors. TFRC interprets these losses as congestion and invokes congestion control mechanisms resulting in degradation of performance.

Previous research on TFRC has focused on picking one challenge and fine tuning TFRC to face that challenge. We believe that the challenges work in combination to degrade TFRC performance. Therefore, TFRC has to be adapted in such a way that it meets the challenges in one state machine.

1.3 Goal

The goal of this thesis is to investigate and detail the challenges for TFRC over multihop wireless ad hoc networks. This investigation will be carried out using the Ns2 simulator [3]. The results of this investigation will be a meaningful characterization of the effects of IEEE 802.11 on TFRC and a proposed algorithm framework to solve the involved problems optimized for the wireless network. We summarize next the key challenges for TFRC over multihop networks to be investigated and mention previous approaches for the involved challenges.

High MAC Contention and Collisions: In order to ensure Internet stability through end-to-end congestion control, TFRC relies on acknowledgment packets from receiver to sender establishing a bidirectional flow of data and ACKs. This is a very costly strategy in multihop wireless networks. First because of the significant MAC overhead associated to an ACK transmission despite the much smaller ACK size relative to a data packet. This happens because of both the RTS/CTS control frames exchanged before any packet transmission and the random backoff procedure that follows any unsuccessful transmission attempt. Yet, data and ACK flowing in opposite directions are highly susceptible to collide inside the network. These collisions are further worsened as TFRC load increases beyond the MAC capacity.

Hence, TFRC should avoid sending redundant ACKs under favorable conditions toward optimal bandwidth utilization and avoid overloading the MAC layer. Previous work

has proposed to keep the transmission rate of TFRC under the MAC threshold. This approach has ignored redundant transmissions into the medium.

High Channel Impairments: Unlike wired environments where a dropped packet is always associated to congestion, wireless networks face loss due to the lossy nature of its medium and may also experience loss caused by link interruption when nodes move. This may cause problems to conventional TFRC because it always reduces its transmission rate when a drop is perceived irrespective of the loss nature.

What is needed here is a mechanism at the sender that can discriminate the actual cause of a packet drop so the sender is able to react properly to each of the factors inducing losses. Past work addressing this problem have serious limitations such as noisy metrics, which justifies further investigations on this issue.

1.4 Reading Guide

In chapter 2 a review of literature relevant to the research problem is presented. This chapter covers simulator basics, main features of TFRC and multihop wireless networks. Simulations are conducted to verify what is reported in the literature. This chapter also discusses the main existing work on TFRC over multihop wireless networks. The drawbacks of each approach are identified.

Chapter 3 describes our approach, the feasibility of this approach and the anticipated risks. In chapter 4, the main concerns for TFRC over multihop wireless networks are addressed. To substantiate discussions, extensive simulations performed in the framework of this thesis are included.

Chapter 5 outlines TFRC response to wireless losses. Chapter 6 presents results of this thesis in form of a state machine and proposed algorithms. The thesis is concluded in chapter 7.

1.5 Abbreviations

Some of the abbreviations used in this thesis are explained below.

AIMD	Additive Increase Multiplicative Decrease
LAN	Local Area Network
WAN	Wide Area Network
Tcl	Tool Command Language
DCF	Distributed Coordination Function
PCF	Point Coordination Function
RE-TFRC	Rate Estimation TCP Friendly Rate Control
ADTFRC	Adaptive TCP Friendly Rate Control
RTT	Round Trip Time

Chapter 2

Literature Review

This chapter presents an overview of the literature relevant to the research problem. Section 2.1 gives an overview of the simulation environment. Section 2.2 details the TFRC protocol. Background on multihop wireless networks is covered in section 2.3 and section 2.4 presents an overview of the IEEE 802.11 standard. Finally sections 2.5 and 2.6 present a summary or multihop routing protocols and previous work respectively.

2.1 The Simulation Environment

The results reported in this thesis are based on simulations using the Ns2 network simulator from Lawrence Berkeley National Laboratory (LBNL) [3]. The Network Simulator is a discreet event simulator targeted at network research. It provides support for simulation of different transport layer protocols, routing, queue management and other applications over wired and wireless networks.

Ns2 is an object-oriented simulator written in C++ with an Object-oriented Tcl (OTcl) interpreter as a front-end. C++ is used for detailed protocol implementation and OTcl for simulation configuration. One drawback of combining two languages is that debugging becomes more complicated than with one language alone.

Ns2 glues C++ and OTcl together by providing support for split objects, that is, objects that appear in both languages, figure 1. When the user creates a new simulator object through the interpreter, the object is first instantiated in the interpreted hierarchy (OTcl), and then a corresponding object in the compiled hierarchy (C++) is created. Attributes of these objects can be bound together; when one is changed, the other is automatically updated. Likewise, compiled methods can be called from OTcl and vice versa. As figure 1 also shows, all objects need not be split objects.

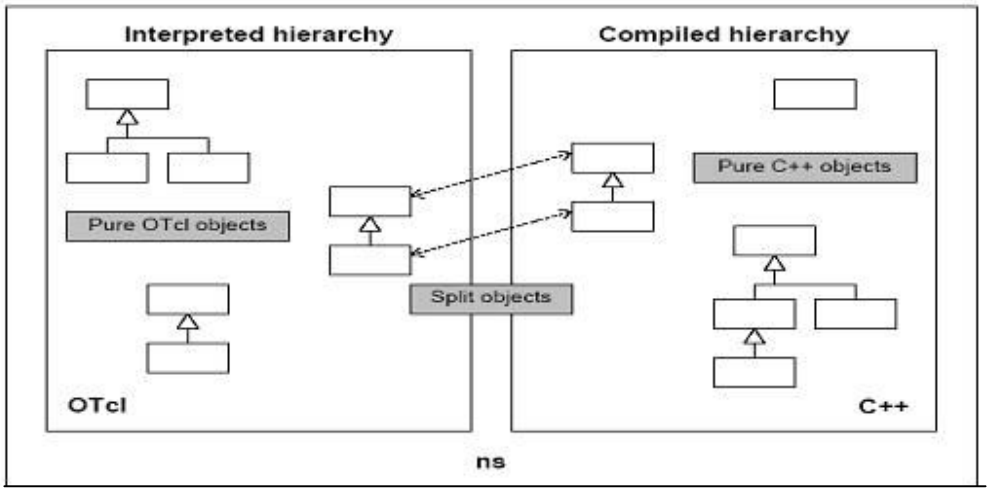


Figure 1: The duality of ns

The basic simulator objects that ns includes are Nodes, links, Agents and Applications. Nodes and Links define the network topology. Agents represent endpoints where network layer packets are constructed or consumed and are commonly used to implement protocols at various levels. Applications are sources that send and receive data. The objects are connected to each other in a layered fashion as seen in figure 2.

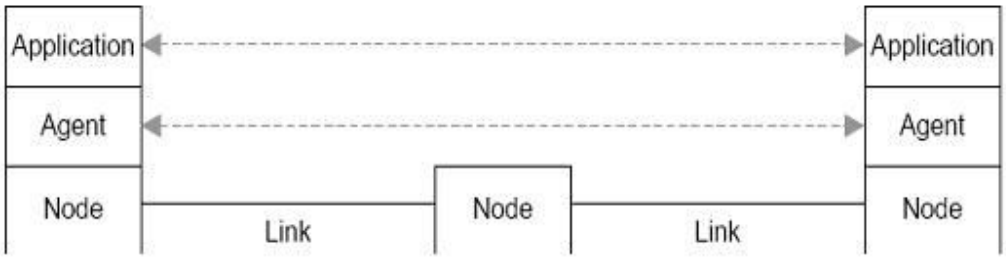


Figure 2: The basic simulator objects in ns and their interconnections

When an agent is attached to a node, it is given a unique address consisting of the node id and a port number within that node. Each node can have multiple agents attached to it. Multiplexing/de-multiplexing between those agents is handled automatically within the Node object. Before the simulation is started, agents must be connected to each other in pairs. This will not trigger a protocol level connection. It will only set the source and destination addresses of the packets to match the two endpoints.

There are two major types of agents in ns: one-way and two-way agents. The one-way agent consists of one sender and one corresponding receiver. Data can only be sent in

one direction and often connection establishment/teardown is not performed. A one-way agent simulates the basic protocol behavior, in most cases how data is transferred, without the added complexity of a full protocol implementation. The result is a simpler implementation with reduced development time. If the service of the one-way agent is not enough for the simulation scenario, two-way agents can be used instead. The two-way agent is both a sender and receiver, supports bi-directional data transfers. And could very well be a real-world counterpart. Currently, Ns2 offers support for various versions of TCP (Tahoe, Reno, New Reno, Sack, Vegas etc.) and for TFRC.

For our experiments we use Ns2 with extensions from the MONARCH project at Carnegie Mellon [3]. The extensions include a set of mobile ad hoc network routing protocols, as well as an 802.11 MAC layer and two radio propagation models. The link layer of the simulator implements the complete IEEE 802.11 standard MAC protocol DCF in order to accurately model the contention of nodes for the wireless medium. All nodes communicate with identical, half duplex, wireless radios that are modeled after the commercially available 802.11-based Wave- LAN wireless radios which have a bandwidth of 2 Mb/s and a nominal transmission radius of 250 m and an interference radius of 550m.

2.2 TFRC: an Equation-Based Congestion Control

TFRC is an equation based congestion control mechanism for unicast traffic where the sender adjusts its sending rate as a function of the loss event rate. This mechanism has been designed to maintain a smoothly changing sending rate, while still being responsive to network congestion over long time periods [1] [2] . TFRC is implemented in the Linux kernel as one of the congestion control options of the Datagram Congestion Control Protocol (DCCP).

2.2.1 The Control Equation: TCP Response Function

For its congestion control mechanism, TFRC directly uses a throughput equation for the allowed sending rate as a function of the loss event rate and round-trip time. In order to compete fairly with TCP, TFRC uses the TCP throughput equation, which roughly describes TCP's sending rate as a function of the loss event rate, round-trip time, and packet size. A loss event is defined as one or more lost or marked packets from a window of data,

where a marked packet refers to a congestion indication from Explicit Congestion Notification (ECN).

The throughput equation is:

$$X = \frac{s}{r\sqrt{\frac{2bp}{3}} + 3p(t_{rto}(1+32p^2)\sqrt{\frac{3bp}{8}})}$$

Where:

X	transmit rate in bytes/second
s	packet size in bytes
r	round trip time in seconds
p	loss event rate, between 0 and 1.0, of the number of loss events as a fraction of the number of packets transmitted
t_{rto}	TCP retransmission timeout value in seconds
b	number of packets acknowledged by a single TCP acknowledgement

2.2.2 The TFRC Protocol

The primary goal of TFRC is to maintain a relatively steady sending rate while still being responsive to congestion. To accomplish this, TFRC makes the tradeoff of refraining from *aggressively* seeking out available bandwidth in the manner of TCP. Thus, several of the design principles of TFRC can be seen in contrast to the behavior of TCP [1].

- Do not aggressively seek out available bandwidth. That is, increase the sending rate slowly in response to a decrease in the loss event rate.
- Do not halve the sending rate in response to a single loss event. However, do halve the sending rate in response to several successive loss events.

Additional design goals for equation-based congestion control for unicast traffic include:

- The receiver should report feedback to the sender at least once per round-trip time if it has received packets in that interval.
- If the sender has not received feedback after two roundtrip times [2], then the sender should reduce its sending rate by half, and ultimately stop sending altogether.

TFRC's congestion control mechanism works as follows:

- The receiver measures the loss event rate and feeds this information back to the sender.
- The sender also uses these feedback messages to measure the round-trip time (RTT).
- The loss event rate and RTT are then fed into TFRC's throughput equation, giving the acceptable transmit rate.
- The sender then adjusts it's transmit rate to match the calculated rate.

In TFRC, the receiver estimates the loss event rate, the loss event being *one or more* packets lost within a roundtrip time. A loss interval is defined as the number of packets transmitted between two loss events. The loss event rate is measured over the most recent loss intervals, using a method called the Average Loss Interval method. It computes a weighted average of the loss rate over the last *n* loss intervals, with equal weights on each of the most recent *n/2*. The value n for the number of loss intervals used in calculating the loss event rate determines TFRC's speed in responding to changes in the level of congestion. As currently specified, TFRC should not be used for values of n significantly greater than 8, for traffic that might compete in the global Internet with TCP. This method gives the smoothest rate changes.

2.2.3 Smoothness in a steady-state: Comparison of AIMD and TFRC

TFRC is designed to be reasonably fair when competing for bandwidth with TCP flows, where a flow is "reasonably fair" if its sending rate is generally within a factor of two of the sending rate of a TCP flow under the same conditions [4]. Experimental results from [1] have shown that simulations of both TCP and TFRC on a wired network show a fair share of available bandwidth. This work also illustrates that TFRC and TCP co-exist fairly across a wide range of network conditions, and that TCP throughput is similar to what it would be if the competing traffic was TCP instead of TFRC. However, TFRC has a much lower variation of throughput over time compared with TCP, which makes it more suitable for applications such as telephony or streaming media where a relatively smooth sending rate is of importance.

In contrast to [1], the focus of our thesis is a multihop wireless network. We are interested in how TFRC and TCP co-exist on this network. Since this investigation is not a goal of this thesis we present next basic simulations to show the main feature of TFRC,

which is that of smoothness. The following section presents our simulation experiments, results and analysis.

2.2.4 Experimental Setup

The following is the description of our simulation setup. Each node has a queue (called IFQ) for packets awaiting transmission by the network interface that holds up to 50 packets and is managed in a drop tail fashion. DSR routing protocol was used. We consider both static and mobile scenarios.

In the static scenario, we consider a 1500m x 400m network topology with eight nodes (0 through 7). A string topology is adopted; figure 3, the distance between any two neighboring nodes is equal to 200 m, which allows a node to connect only to its neighboring nodes. In other words, only those nodes between which a line exists can directly communicate. The same distances between neighboring nodes ensure that the nodes act equally in the simulation. Only a portion of the nodes in this network are involved in the experiment.

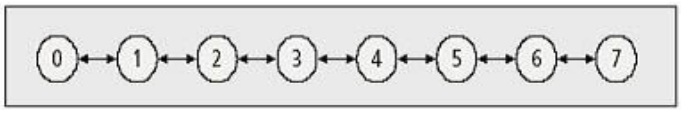


Figure 3: A String Topology

In the mobile scenario 8 wireless nodes roam freely in a 400m x 800m topology following a random waypoint mobility pattern, in which the pause time is zero so that each node is constantly moving.

2.2.5 Results

The aim of these experiments was to measure and plot the throughput of TFRC/TCP over an ad hoc network. This was done to verify that TFRC has a more smoothly changing throughput than TCP over time and hence making it more suitable for multimedia applications on a multihop network.

In simulating TFRC over IEEE 802.11 we faced a challenge in that the wireless extension to Ns2 does not include TFRC packet types. By adding code to Ns2 source files and re-compiling the simulator on windows XP, we were able to simulate TFRC traffic over IEEE 802.11. The added code can be found in Appendix A.

1) Static scenario: In this set of experiments, we set up a single TFRC/TCP connection between a chosen pair of sender and receiver nodes. This was from node 1 to node 5. We also performed the same experiment for TCP. To do this, we provided as input to the simulator scripts we wrote in TCL. The TCL scripts for both experiments can be found in Appendix B. The scripts were used for the simulation and as a result, two trace files were generated as per our specification. The first trace file was used to visualize the simulation run with Network animator. A screen shot of the topology captured from NAM can be found in Appendix C. The second trace file was analyzed for the various parameters that we wanted to measure. We measured the successively received packets over the lifetime of the connection. We wrote a perl script to do this, found in Appendix B. Output from the perl script was used as data for plots with Gnuplot. Gnuplot is a command-driven interactive function plotting program. If files are given as inputs, gnuplot will load each file with the plot command and process the scripts. An example of our Gnuplot usage can be found in Appendix B. Figures 4 and 5 show Gnuplot graphical output of throughput versus time for TFRC and TCP respectively.

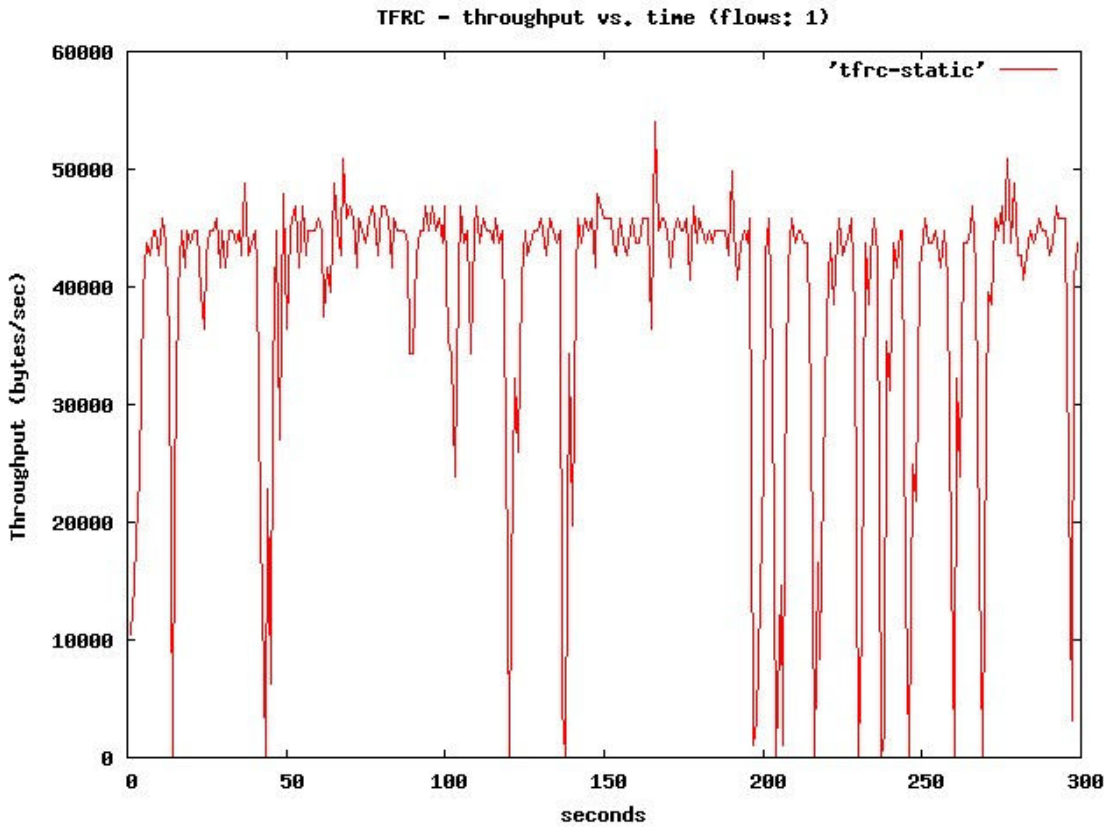


Figure 4: TFRC throughput over static IEEE 802.11

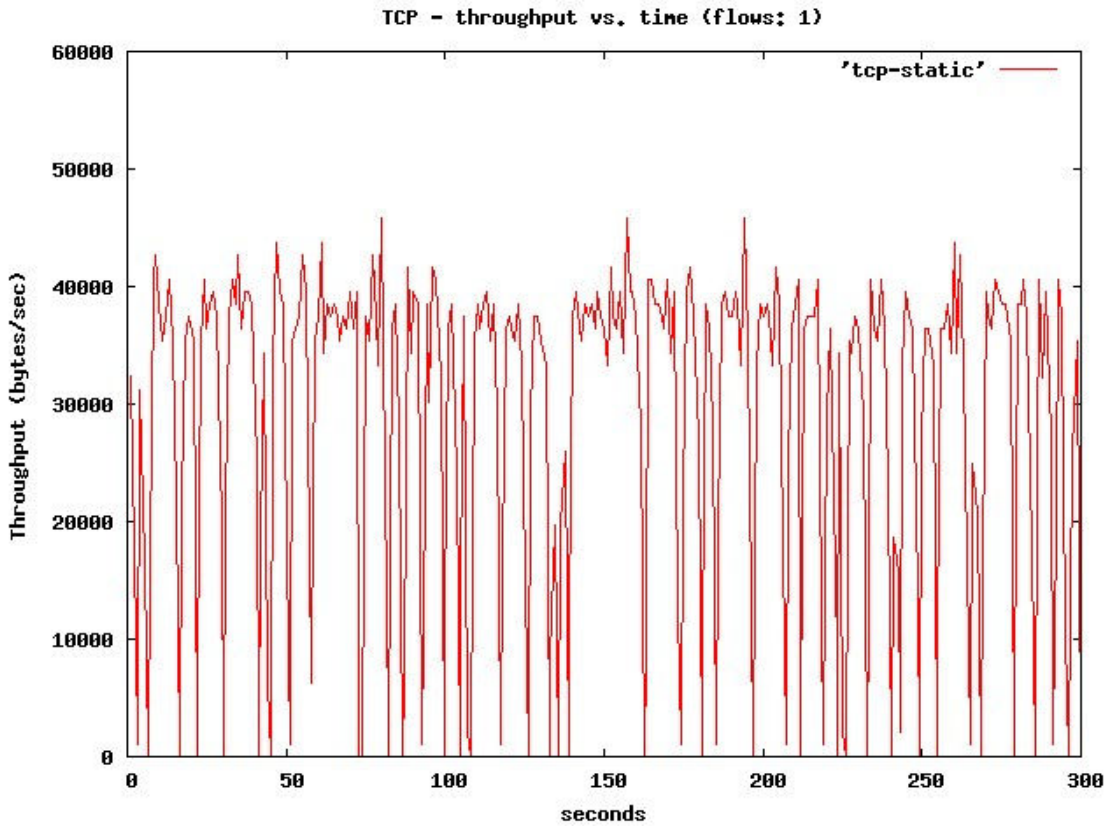


Figure 5: TCP over static IEEE 802.11

2) Mobile Scenario: In a mobile ad hoc network there are a lot of network conditions that can affect transport layer throughput. Conditions such as route changes and mobility induced disconnections could bring throughput to zero. The aim of this experiment is to compare the throughputs of AIMD and TFRC given such a mobile network. We have chosen to use the random waypoint mobility model because it closely represents the way a mobile user would move in a real life situation, which is simply randomly. The speed is set to 1m/sec.

In this set of experiments we set up a single TFRC/TCP connection between node 0 and node 3. Figure 6 and Figure 7 show Gnuplot graphical output of throughput versus time for TFRC and TCP respectively. A screen shot of the topology captured from NAM can be found in Appendix C.

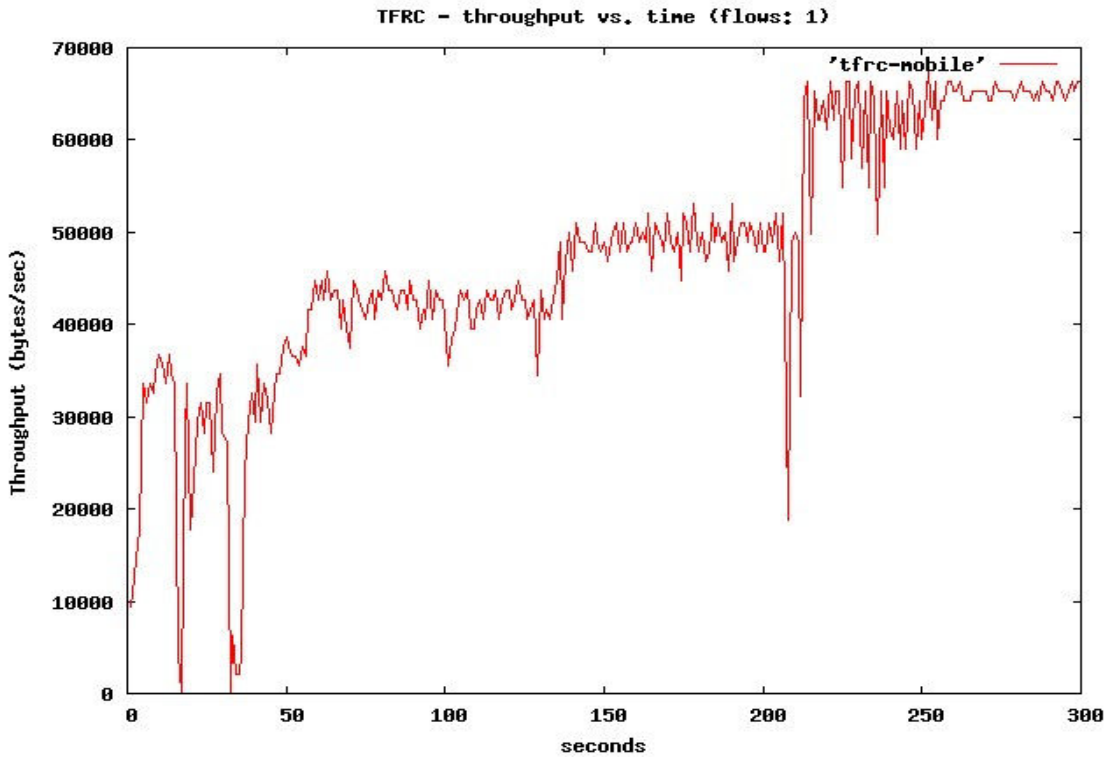


Figure 6: TFRC over mobile IEEE 802.11

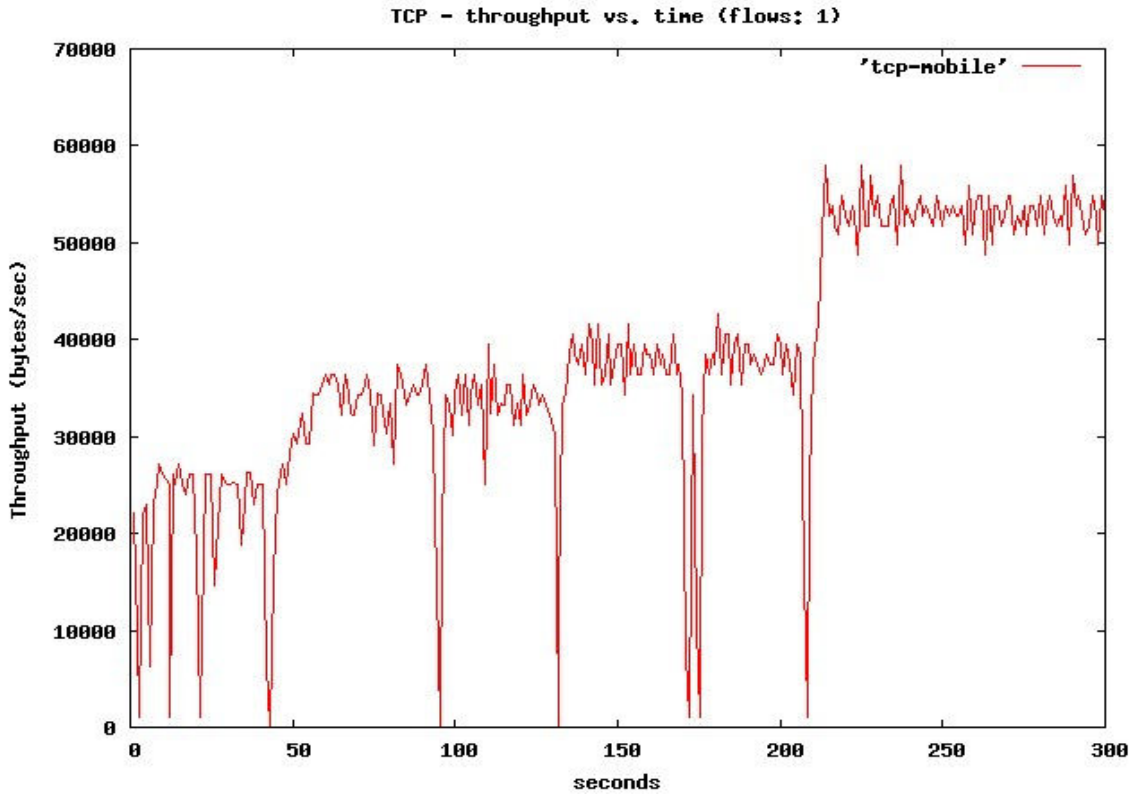


Figure 7: TCP over mobile IEEE 802.11

2.2.6 Analysis

Equation based congestion control (TFRC) provides a better smoothness in a steady state than AIMD congestion control (TCP). The figures 4 to 7 from our experimental evaluations show how TFRC flows are considerably smoother than TCP flows. This makes TFRC highly suitable for multimedia applications. The reason for this behavior can be attributed to how both protocols respond to congestion.

TCP uses the AIMD congestion control mechanism which backs off in response to a single congestion indication, causing abrupt changes in its sending rate. This has an effect of oscillating the throughput as can be seen in figures 5 and 7. TCP's abrupt changes in the sending rate have been a significant impediment to the deployment of TCP's end-to-end congestion control by emerging applications such as streaming multimedia.

TFRC on the other hand uses an equation-based congestion control. Whereas AIMD congestion control backs off in response to a single congestion indication, equation-based congestion control uses a control equation that explicitly gives the maximum acceptable sending rate as a function of the recent *loss event rate*. The sender adapts its sending rate, guided by this control equation, in response to feedback from the receiver. The result is a more smoothly changing throughput as can be seen in figures 4 and 6.

In conclusion, we believe equation-based congestion control is a viable mechanism to provide relatively smooth congestion control for multimedia traffic. For applications that simply need to transfer as much data as possible in as short a time as possible or if reliability is required, using an Additive-Increase, Multiplicative-Decrease (AIMD) congestion control scheme with similar parameters to those used by TCP would be best.

2.3 Multihop Wireless Ad hoc Networks

An ad hoc network consists of a collection of nodes forming a dynamic autonomous network. Nodes communicate with each other over the wireless medium without the intervention of centralized access points or base stations. Each node acts both as a router and as a host. Due to the limited transmission range of wireless network interfaces about 250m, multiple *hops* may be needed to exchange data between nodes in the network, which is why they are termed *multi-hop.*

Figure 8 shows a simple ad hoc network with three nodes. The outermost nodes are not within transmitter range of each other. However the middle node can be used to forward

packets between the outermost nodes. The middle node is acting as a router and the three nodes have formed an ad hoc network.

Figure 8: Example of a simple ad hoc network with three participating nodes

2.3.1 Usage

There is no clear picture of what these kinds of networks will be used for. The suggestions vary from document sharing at conferences to infrastructure enhancements and military applications.

In areas where no infrastructure such as the Internet is available an ad hoc network could be used by a group of wireless hosts. This can be the case in areas where a network infrastructure may be undesirable due to reasons such as cost or convenience. Examples of such situations include disaster recovery personnel or military troops in cases where the normal infrastructure is either unavailable or destroyed.

Other examples include business associates wishing to share files in an airport terminal, or a class of students needing to interact during a lecture. If each host wishing to communicate is equipped with a wireless local area network interface, the group of hosts may form an ad hoc network.

It becomes clear that a multihop wireless ad hoc network can provide a low cost and flexible infrastructure for a variety of traffic. Delay-sensitive multimedia applications, including real-time multimedia streaming, conversational services (video-conferencing), surveillance etc, are one example of such traffic.

2.3.2 Challenges in Multihop wireless ad hoc networks

A central challenge in the design of ad hoc networks is the development of dynamic routing protocols that can efficiently find routes between two communicating nodes. Routing protocols are not the primary target of this thesis. However, we give a brief

discussion of two fundamental routing protocols used in multihop networks, namely Dynamic Source Routing (DSR) and Ad hoc on Demand Distance Vector (AODV). In chapter 3 we show how these routing protocols pose a challenge on multihop networks.

In this thesis we discuss the following key challenges in wireless multihop ad hoc networks: medium access control (MAC) and multihop network conditions that degrade performance of transport protocols. In ad hoc networks conditions such as congestion, channel error, route change and disconnection will cause packet loss. The effect of this on TFRC will be shown in chapter 4.

Since media is a shared and scarce resource in a wireless network, efficiently controlling access to this shared media becomes a complicated task. Further, the medium is assumed to be highly error-prone and the nodes in place may move unpredictably, the medium access control protocol must be not only highly adaptive but also tolerant to transmission failures. The IEEE 802.11 [5] standard was designed to meet some of such requirements, but many problems remain to be addressed, as shown next.

2.4 IEEE 802.11 Standard

The IEEE 802.11 "Distributed Foundation Wireless Medium Access Control" (DFWMAC) [5] is the standard Medium Access Control (MAC) layer protocol adopted for ad hoc networks. The IEEE 802.11 standard specifies both the wireless LAN MAC and physical layer mechanisms for an efficient shared broadcast channel through which the involved mobile nodes can communicate. The standard currently defines a single MAC and three transmission techniques allowed in the physical layer (all of them running at 1 and 2 Mb/s) as follows: frequency hopping spread spectrum in the 2.4 GHz band, direct sequence spread spectrum in the 2.4 GHz band, and infrared. The communicating nodes within a wireless LAN (WLAN) are termed stations, and the IEEE 802.11 standard is mostly termed simply 802.11 or 802.11 MAC protocol. The unit of information used for the MAC messages is frame. These nomenclatures are used throughout this section for compliance with the standard description.

In 802.11, priority may be given to stations but in general all stations receive equal right to access the medium. Collisions are prevented instead of detected after they happen, and multiple hops communication is allowed. The main novelties of 802.11 include: 1) use of acknowledgment for data frames (link layer's ACKs), 2) possibility of using RTS/CTS (request-to-send/clear-to-send) control frames, 3) a virtual carrier sensing mechanism.

These mechanisms aim at mitigating medium collisions and obtaining efficient bandwidth utilization, as explained below.

2.4.1 Distributed Coordination Function (DCF)

The MAC layer supports two modes of operation. The distributed coordination function (DCF), does not use any kind of central control and point coordination function (PCF), uses the base station to control all activity in its cell. Since the PCF cannot be used in an ad hoc network, DCF is thus assumed to be the access method used throughout this thesis.

DCF implements a *carrier sense multiple access with collision avoidance* (CSMA/CA) protocol for controlling the shared medium access among the competing stations. In this mechanism a station wishing to transmit will listen to sense if the medium is idle before transmission. The CSMA/CA distributed algorithm includes a random backoff procedure to minimize the probability of collisions by simultaneous transmission attempts just after the medium has been sensed idle. Actually, the CSMA/CA imposes that a minimal idle interval exists between contiguous frame sequences. Hence, every station must wait for a specific Interframe Space (IFS) after the medium is determined idle, and then delay a random backoff interval before transmitting. This reduces collision probability considerably. In addition to the CSMA/CA protocol, the access method of DCF uses *positive acknowledgments* in that the receiving stations respond to a frame reception with an acknowledgment frame (ACK frame). The transmitting stations are able to retransmit frames that are dropped or have their respective ACK frame dropped. The DCF retransmission mechanism is persistent in the sense that it attempts for several times to recover locally an unsuccessful transmission. The features explained above render DCF a robust protocol for the challenging wireless networks. It mitigates problems such as the classical hidden node as well as typical performance degradation by the high bit error rate inherent in wireless communications. Some of the DCF mechanisms which contribute to the discussion of this thesis are addressed below, where the need of each mechanism will become clear.

2.4.2 Carrier Sensing Mechanism

In this mechanism, both physical channel sensing and virtual channel sensing are used. The CSMA/CA is referred to as a *physical carrier sensing* mechanism since it is

associated to the instantaneous physical medium condition. In other words, CSMA/CA detects whether there is or not an actual transmission going on when a station intends to transmit by analyzing, for instance, the signal strength of other stations. The *virtual carrier sensing* is referred to as the *Network Allocation Vector* (NAV). The principle of NAV is to include information into the frame headers so that a transmitting station may inform the other stations for how long the medium is going to be busy with the current transmission. The other stations overhearing this transmission set their respective NAVs to this announced time value. In this way, all stations will refrain from transmitting for at least NAV interval. The CA is necessary to reduce the probability of having various stations transmitting at the same time, which would lead the medium to experience collision. But there is always a chance of stations simultaneously sensing the medium as free and transmitting at the same time, causing a collision. These collision situations must be identified so the packets can be retransmitted by the MAC layer, rather than by the upper layers. The latter case will cause significant delay. In order to overcome the collision problem, the IEEE 802.11 uses two access methods, where the CA mechanism is coupled with a positive acknowledgement scheme. These are the *Basic* and *RTS/CTS* access methods. We only cover the later since it is commonly deployed in multihop networks.

2.4.3 RTS/CTS Access Method

Since a station in a wireless network may not listen to the medium while transmitting, it has to wait long before detecting that its transmitted frame collided with other station's transmission. A station has to wait for the estimated time for both the data frame and corresponding ACK frame to be transmitted to take action toward recovering the unsuccessful transmission. Hence, the larger the data frame the longer the time to recover from a failed transmission, which incurs in waste of bandwidth.

To address the waste of bandwidth involved with corrupted frames, the 802.11 in DCF access mode can use short control frames to reserve the wireless medium prior to actual data transmission. There are two specific frames for that: Request-To-Send (RTS) and Clear-To-Send (CTS). These frames are exchanged by every peer station involved in a transmission before the data frame may be sent. In case of frame corruption, the waste of bandwidth is much less substantial because both RTS (20 bytes) and CTS (14 bytes) sizes are significantly smaller than the maximum date frame size (2346 bytes). Figure 9 illustrates the RTS/CTS access method. The sending station may transmit upon detection of

idle medium for Distributed Interframe Space (DIFS) interval. The station transmits first an RTS frame and waits for the CTS frame from the receiving station. Immediately after the reception of the RTS frame the receiving station waits only a Short Interframe Space (SIFS) period to transmit the CTS. By receiving this reply, the sending station infers that the medium is reserved for it and so it transmits the data frame and waits for the ACK frame. Note that all the intermediate interframe spaces involved in the transmission of a complete data frame are SIFS. This procedure leads the medium to be reserved for the whole frame exchanges associated to a successful data transmission.

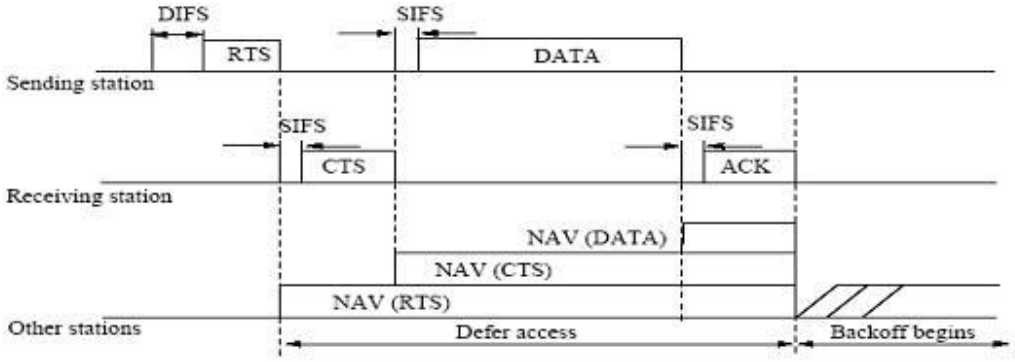


Figure 9: IEEE 802.11 Timing diagram

The technique explained above is called *four-way Handshake* because every data frame transmission requires that a sequence of RTS-CTS-DATA-ACK be exchanged between sending and receiving stations. Figure 9 also shows that the other stations update their waiting time by setting their NAV values in accordance with the duration value announced in the header of the frames exchanged between the two communicating stations. Thus, the other stations remain silent until the end of the *four-way Handshake* plus DIFS period. Afterwards, the backoff procedure is invoked and eventually these stations may transmit.

The use of the short RTS/CTS control frames is also appropriate for mitigating the well-known hidden node problem. This problem occurs when two hidden nodes from each other wish to communicate simultaneously with a third common node, which would result inevitably in collision. As illustrated in Figure 10, the RTS frame silences the stations reached by the sending station's transmission, while the CTS frame does so for the stations hearing the receiving station's transmission.

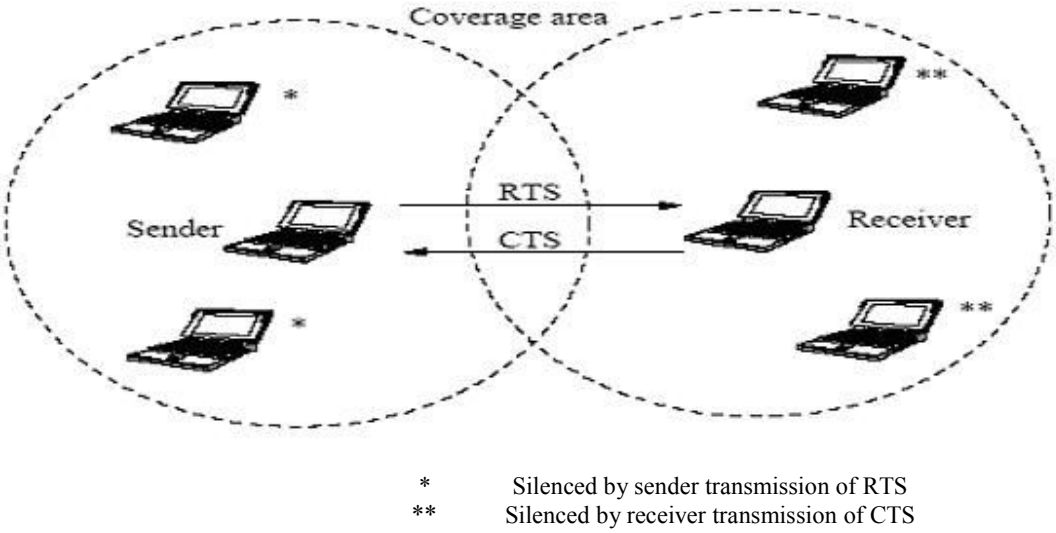


Figure 10: CTS/RTS to prevent hidden node problem

Besides the hidden node problem, wireless packet networks also face the exposed node problem. An exposed node is one that is within the sensing range of the sender but out of the interfering range of the destination. If exposed nodes are not minimized, the available bandwidth is underutilized. However, in the 802.11 MAC layer protocol, there is almost no scheme to deal with this problem.

The main drawback of using RTS/CTS is that the overhead associated to them renders these control frames inefficient in transmission with relatively small packet sizes. In the case of multimedia applications this is an advantage as they have relatively large packet sizes. Furthermore, the RTS/CTS control frames do not fully solve the hidden node problem for scenarios where long chain of nodes are in place as not every node can hear each other's transmission. Chapter 3 addresses them further from a TFRC perspective.

2.5 Multihop Network States

Ad hoc states are the second key challenge we focus on in this thesis. Previous research has indicated that identifying the following network states should be necessary to improve the performance of TCP-friendly multimedia streaming over ad hoc networks [1] [4] [12] [13].

2.5.1 Congestion

Congestion in ad hoc networks is defined as the signal that the offered load exceeds the network capacity. When congestion occurs, the queue size will grow and the network throughput is reduced. To deal with congestion, the transport protocol should reduce the sending rate, similar to the standard TFRC protocol.

2.5.2 Channel Error

When random packet loss occurs, the receiver should not count it as a congestion event. The sender should maintain its current sending rate.

2.5.3 Route Change

The delivery path between the two end hosts can change from time to time, but disconnections are too short to result in a retransmission timeout. Depending on routing protocols, the receiver may experience a short burst of out-of-order packet delivery or packet losses. In both cases, the receiver should not treat it as congestion; and the sender should keep the streaming rate unchanged in the next RTT period, waiting for the receiver to feedback more measurement statistics for the new path.

2.5.4 Disconnection

When the delivery path is disconnected long enough to cause a retransmission timeout, instead of exponentially slowing down and backing off, the sender should freeze the current congestion window and the retransmission timer. It then performs a periodic probing so that the transmission can be resumed promptly once a new path is established. Once it is recovered, the actions of route change should be followed.

2.6 Routing Protocols

Development of routing protocols for ad hoc networks has been one of the hottest topics within this area in recent years. Two proposals have been evaluated extensively in the literature and are under process of standardization in the Internet community. However, we do not address these protocols in detail since our goal is to provide only an overview on routing protocol principles rather than a complete review on them. AODV and DSR routing protocols are described below as an insightful introduction to this broad subject.

2.6.1 AODV

AODV [6] routing protocol is a table-driven algorithm based on the classical Bellman-ford routing mechanism. The key idea of AODV is to reduce the elevated number of required broadcasts typical for keeping up-to-date routing tables. By updating routes on an on-demand basis, AODV provides an optimized routing strategy for large ad hoc networks. In fact, AODV is denoted *a pure on-demand route acquisition system* because it only involves nodes in a selected path to discover and keep the related routing information. The other nodes are completely unaware of such routes existence. The routing procedure in AODV involves two phases: route discovery and route maintenance. Whenever a node needs to find a route to a destination to which it has no table entry, it begins the route discovery algorithm. Once the route has been established, the route maintenance algorithm takes care of the route's state variables until the route is not needed anymore.

2.6.2 DSR

DSR was designed for small scale networks of up to about 200 nodes [7]. It employs the concept of source routing instead of hop-by-hop routing. This means that every packet carries in its header the complete, ordered list of nodes through which the packet must pass. This feature releases the intermediate nodes from the task of keeping route tables to forward packets because the packets themselves already contain such information. Thus, DSR does not need any period message exchanges common in many other routing protocols for maintaining accurate route tables. In DSR, every node contains a cache of source routes it has learned or overheard in order to speed up route discovery when a route breaks. As in AODV, DSR also consists of two phases: route discovery and route maintenance. Nodes initiate route discovery when they need a route that is not found in their respective route caches. Broken routes are detected by proper route maintenance mechanisms.

2.7 Previous Work

Most work done on improving transport layer protocol performance on wireless ad hoc networks, concern TCP. Very little work has been done with respect to TFRC. Related work on TFRC concerning transport layer protocol enhancements is as follows:

2.7.1 MULTFRC

In [8], they explore the necessary and sufficient condition under which using one TFRC connection in wireless streaming applications results in under-utilization of the wireless bandwidth. They propose the use of multiple simultaneous TFRC connections for a given wireless streaming application.

The aim of this research is to send multiple connections over the same link and there by utilize the available bandwidth. The advantages of their approach are as follows: first, it is an end to-end approach, and does not require any modifications to network infrastructure and protocols, except at the application layer. Second, it has the potential to fully utilize the wireless bandwidth provided the number of connections and packet size are selected appropriately. This research shows great potential for streaming multimedia, however, the target network is different from ours. They focus on a wired to wireless link. This type of connection will not experience the rich set of wireless losses such as those found on a multihop wireless ad hoc network.

2.7.2 RE-TFRC

In the problem description we stated that TFRC protocol faces challenges on multihop wireless ad hoc networks which are characterized by losses due to physical and MAC errors. TFRC interprets these losses as congestion and invokes congestion control mechanisms resulting in degradation of performance.

In research done in [9] a study is made of the phenomenon that as the network load increases the MAC reaches a saturation threshold which then results in an increase in MAC errors and round trip time. The related results show that TFRC then records higher congestion levels resulting in degradation of performance. In particular the investigation focused on the problem of the misinteraction between TFRC and the 802.11 MAC layer. The objective was to make TFRC aware of RTS/CTS-induced congestion such that it chooses a near-optimal sending rate that avoids MAC layer saturation. A major contribution of this work was the introduction of a new Rate Estimation (RE) algorithm in TFRC to estimate the saturation capacity of the MAC layer. This involved creating a model for round-trip time during MAC layer saturation and deriving a composite TFRC loss event rate that reflects the current MAC layer congestion level. By limiting the sending rate to a value that is lower than the estimated rate, RE-TFRC avoids MAC layer congestion.

The main aim of this work was to constrain TFRC sending rate to below the MAC saturation level and thereby lessen MAC errors. RE-TFRC manages to reduce errors at the MAC layer. The reduced collisions result in a lower loss event rate and round-trip time and a smoother sending rate for RE-TFRC. RE-TFRC loss event rate is shown to be 8% to 55% less than that of TFRC.

However the recorded merger 5% throughput improvement over TFRC can be attributed to the failure of this research in taking into consideration the other challenges on a multihop ad hoc network that cause packet loss. At the end of the day RE-TFRC may record lesser MAC errors but will still be interpreting losses due to wireless errors as congestion.

2.7.3 ADTFRC

In [10] work is done to adapt TFRC to mobile networks. The key design novelty of this work is to perform multi-metric joint identification for packet and connection behaviors based on end-to-end measurements. The argument is that, it is not enough for TFRC to react to network congestion as the sole cause of packet loss in an ad hoc network. TFRC should take into consideration other events that might cause packet loss individually or in combination. Conditions such as mobility induced disconnection and re-connection, route-change induced out-of-order delivery and error/contention-prone wireless transmissions. In reacting differently to these events, TFRC performance is expected to improve. The aim of this work was to come up with measurements at the end hosts to be used to detect congestion, disconnection, route change, or channel errors, so that the TFRC sender could respond accordingly to achieve better quality of the multimedia streaming over mobile ad hoc networks.

To this end, they propose to use multi-metric joint identification instead of single metrics. Single metrics may be affected by the different network events giving noisy measurements. The results show that ADTFRC is able to significantly reduce the false detection probability, thus greatly improving the transport performance in a TCP friendly way. The advantage of this approach is that the design is easy to implement and deploy and requires only software upgrades at the two end hosts.

In devising these metrics, congestion is rightly identified as the major cause of packet loss and packet loss due to channel error is assumed to be negligible. However [9] [11] make an observation that, unlike wired networks where buffer overflow dominates

packet losses, most packet drops experienced by TCP are due to link layer contention, incurred by hidden terminals. Buffer overflow induced packet loss is rare, and the contention-induced packet loss offers the first sign of network overload. Their analysis and simulations further show that contention drops exhibit a load-sensitive feature: as the offered TCP packets exceed an optimal window size $W*$ and increase further, link drop probability becomes non-negligible and increases accordingly. This observation is applicable to TFRC as a TCP friendly mechanism.

Therefore when it comes to evaluation of the performance improvement analysis of ADTFRC a note is made that, the presence of channel error slightly decreases the performance gap between ADTFRC and TCP+ELFN. The gap comes from the identification inaccuracy of ADTFRC. When both mobility and channel error are present, metric samples become highly noisy; this makes end-to-end network state detection, especially for non-congestion states, more difficult. This scheme has been evaluated in simulation using the ns2 simulator and also via implementation in the Linux kernel. The results are positive but focused on mobility conditions. The interaction between TFRC and the MAC layer were not investigated in detail. In chapter 4 we show detailed investigations to characterize the interaction between TFRC and the MAC layer. In our proposal in chapter 5 we incorporate features that address this interaction.

2.8 Summary

This chapter introduced TFRC. This protocol has been shown to be suited for multimedia streaming over multihop networks.

This chapter also introduced multihop wireless networks. These networks are necessary for establishing long-range wireless communications in ad hoc networks. In particular, ad hoc networks mechanisms were addressed in greater detail because these are the primary target environments of this thesis.

The IEEE 802.11 standard specifies the MAC protocol for ad hoc networks. In 802.11, nodes may share the wireless medium smoothly due to the CSMA/CA mechanism in place. The design of 802.11 also includes functionalities for avoiding typical hidden node problems by the use of the RTS/CTS control frames. The efficiency of 802.11 is, however, to be improved for multihop scenarios relaying on several end-to-end hops. The

problems that arise under such scenarios are discussed in chapter 4 from a TFRC perspective.

A multihop network has features that put the network in different states. TFRC needs to recognize these states in order to improve its performance.

The most known routing protocols for ad hoc networks are AODV and DSR. Both are on-demand protocols, and the former minimizes broadcast messages typical of table driven routing protocols but supports only symmetric paths. The latter works on a hop by-hop basis and does not need conventional route tables because the end-to-end route is carried in every packet header. This may incur prohibitive traffic overhead. DSR also supports non symmetric paths and allows nodes to have several alternative paths to the same destination.

Previous work on the involved problem has focused on adapting TFRC to meet individual challenges.

The next chapter introduces the approach we take in solving the involved problem.

Chapter 3

Approach

This chapter presents the approach we propose to take in solving the research problem, which can be found in section 3.1. Sections 3.2 and 3.3 present the feasibility of our approach and risks anticipated respectively.

3.1 Proposed Approach

Having the concepts in chapter 2 in mind, the following strategy was taken based on the two key challenges on a multihop network, in this thesis. We address the problem of discriminating the nature of dropped packets to enhance TFRC sender reaction to packet loss in combination with improved TFRC performance by mitigating the problems created by the bidirectional flow established in a TFRC connection. This strategy comes from the realization that weaknesses reported in previous work can be attributed to the fact that they adapted TFRC by addressing multihop challenges separately. An analysis of previous work brings us to the conclusion that for TFRC to perform well over ad hoc networks it must be equipped with the following capabilities. It must have the capability to distinguish and respond differently to the various factors that cause packet losses on ad hoc networks. Factors such as congestion, disconnection, route- change and channel errors should trigger different events in the TFRC state machine. By so doing the probability of false congestion detection will be greatly reduced. To this end we further conclude that, TFRC should have awareness of MAC layer RTS/CTS contention effects which have the adverse effect of increasing false congestion detection. Our strategy works as follows:

We propose to enhance the performance of TFRC based on aspects of ADTFRC a TFRC variant designed to perform well over wireless links. ADTFRC uses measurements at the end hosts to detect congestion, disconnection, route change, or channel errors, so that the TFRC sender could respond accordingly to achieve better quality of the multimedia streaming over mobile ad hoc networks. In ADTFRC when MAC errors occur, no action is taken, but to simply maintain connection establishment that is, TFRC is left unconstrained. On a multihop ad hoc network this would not be a wise decision. From the results in [9], when unconstrained, TFRC produces an offered load that is above the rate sustainable by the multihop 802.11 MAC layer. The MAC layer then suffers from multiple frame

retransmissions that increase the round-trip time. Although TFRC eventually receives some packet loss notification because of the frame retransmissions, these packet losses arrive too late for TFRC to curtail its offered load below the saturation point of the MAC layer.

Therefore, we do not ignore packet loss due to MAC errors. Instead we propose to make use of an algorithm proposed in [9], which enables TFRC to adjust its sending rate to below the MAC layer saturation point. Our proposal allows TFRC to respond differently when the network is in a state of MAC error as opposed to the approach taken in ADTFRC.

The methodology we take is as follows: first we conduct extensive simulations to show that the MAC and multihop network states work in combination to degrade TFRC performance. Finally we propose an enhancement to TFRC that, given the network state is able to choose an appropriate action while mitigating the problems created by the bidirectional flow established in a TFRC connection. The result is a protocol we call ARETFRC, adaptive rate estimation TCP friendly rate control protocol.

3.2 Feasibility of Approach

We have good reasons in considering our strategy feasible. In the first place the 802.11 protocol is a reality today, so the concepts introduced in this thesis attempt to get the most out of it rather than propose a new MAC protocol. This implies that currently only short-range multihop networks are feasible, since 802.11 cannot sustain acceptable performance for long networks regarding the number of hops end-to-end.

The second reason refers to the deployment complexity. To change every node in the network is not always a good practice, so end-to-end solutions are appealing for concentrating the changes at the end nodes. Another important reason is related to the possibility of incremental deployment. An enhanced protocol should be able to interoperate with the regular protocols already in place. The fourth reason has to do with energy efficiency. That is, solutions to improve TFRC in multihop networks should not be costly in terms of energy consumption, but should be as energy efficient as possible because the nodes in place are presumably battery powered. Therefore, the contributions of this thesis are built up on the following observations:

- Only short-range multihop wireless networks are feasible today.
- End-to-end solutions minimize implementation complexity.
- Incremental deployment is a clear advantage.
- Energy efficiency is a key issue.

The first three observations have a big influence on our experimental work found in chapter 4. We uphold the fourth observation from the complexity analysis of our proposed algorithm in chapter 6.

3.3 Risks

A major risk we anticipate is that time allocated for this project may not be enough to see us complete the work. To carryout our research, we will need to use a network simulation tool called Ns2. This tool is open source on the internet for anyone wishing to carry out network research. It has been used extensively and has a lot of support on the web. It comes in two versions. The first is a pre-compiled version, targeted at research that only deals with analysis of already implemented protocols. The second, targeted at research involving implementation issues, is component based. This means downloading the necessary components and compiling the simulator in order to install and use it. This brings about several complications. Firstly we have no prior knowledge in using this tool. Gaining that knowledge may take up time from research work. Secondly using this tool means learning a set of new programming languages. These will include TCL, perl and awk. Again this may eat into research time.

Another risk is that limited C++ knowledge may prove limiting on protocol implementation. In order to investigate TFRC over multihop networks, we need knowledge in C++ network programming as TFRC is not fully implemented in ns. What we have is some high level C++ programming to start with. Again this limitation might cost us some time.

All things considered we hope to make good use of our knowledge in design and analysis of algorithms, software engineering and computer networks to address the research problem and come up with a viable algorithm addressing the same.

Chapter 4

TFRC over Multihop Problems & Evaluation

TFRC has been viewed as a viable mechanism to provide smooth congestion control on the Internet for applications such as streaming multimedia. In the near future, more and more TFRC flows are expected to be transmitted over wireless networks as they are an extension of the Internet. The problem is that a regular TFRC implementation cannot properly handle the medium related constraints inherent in multihop wireless networks. TFRC is tuned to work well in wired networks. As shown in section 2.3, wireless networks are susceptible to high bit error rates and rely on very limited bandwidth when compared with their wired counterparts. These features present a unique challenge to transport protocols not found in wired networks. Thus, emerging protocols like TFRC must be adjusted to fit such new environments. Our approach to adapting TFCR is that of discriminating the nature of dropped packets to enhance TFRC sender reaction to packet loss in combination with improved TFRC performance by mitigating the problems created by the bidirectional flow established in a TFRC connection. This chapter introduces TFRC over multihop networks. We begin by detailing problems faced by TFRC and then detailing those created by the bidirectional flow established in a TFRC connection.

4.1 Impact of Wireless Transmission Medium on TFRC

Wireless media are characterized by high, variable bit error rates (BERs), ranging typically from much less than 1% to over 20% [14]. Compared with wired networks, wireless networks are susceptible to loss rates that are about two orders of magnitude higher. Wireless induced losses are caused primarily by fading, interferences from other equipments, and diverse environmental obstructions. Any of these factors may induce either single or bursty packet losses.

As opposed to wireless media, wired media have negligible BERs. In wired networks, any packet loss may be safely associated to congestion in the network. For this reason, regular TFRC always handles packet losses as if they were caused by congestion. There are situations, however, in which TFRC should not simply reduce its transmission

rate in the face of a lost packet. Rather, it should determine whether this is indeed the best action to be taken on the basis of the actual reason causing the lost packet [10].

Furthermore, the very scarce bandwidth in wireless channels requires that upper layer protocols like TFRC avoid unnecessary, redundant transmissions into the medium toward high bandwidth utilization. Later in this thesis our proposal takes into consideration unique features of multimedia applications leading to a smart acknowledgment strategy at the receiver to optimize bandwidth utilization in a completely dynamic and adaptive manner.

4.2 Disturbance of Routing Protocol Strategy on TFRC

As mentioned in chapter 2, DSR [7] and AODV [6] have been considered the most prominent routing protocols for ad hoc networks so far. Both protocols work on an on demand basis in order to minimize broadcast messages in the bandwidth constrained medium. The primary difference between both routing protocols is the route cache scheme used only in DSR and the maintenance scheme with time expiration used only in AODV. Yet, the characteristics of DSR and AODV above suggest that the routing efficiency and consequently the TFRC performance depend not only on the network load but also on the mobility pattern in place. AODV is expected to perform better under moderate network load and high mobility since it updates routes periodically. On the other hand, DSR is likely to outperform AODV in scenarios facing high load and low mobility due to its route cache scheme.

We discuss now the performance of AODV and DSR in static scenarios since this may be significant from a TFRC standpoint. As explained in chapter 2, DSR messages carry in the packet header the full route information from source to destination. As a result, the more hops the higher the traffic overhead. This may render DSR inefficient in scenarios where nodes are static and the network load is moderate, because its cache of routes plays no significant role in such scenarios where practically no link interruptions occur. To further clarify this issue, we conducted simulations for comparing AODV and DSR in a static chain topology of figure 3. Figure 11 illustrates the outcome in which throughput against number of hops is presented.

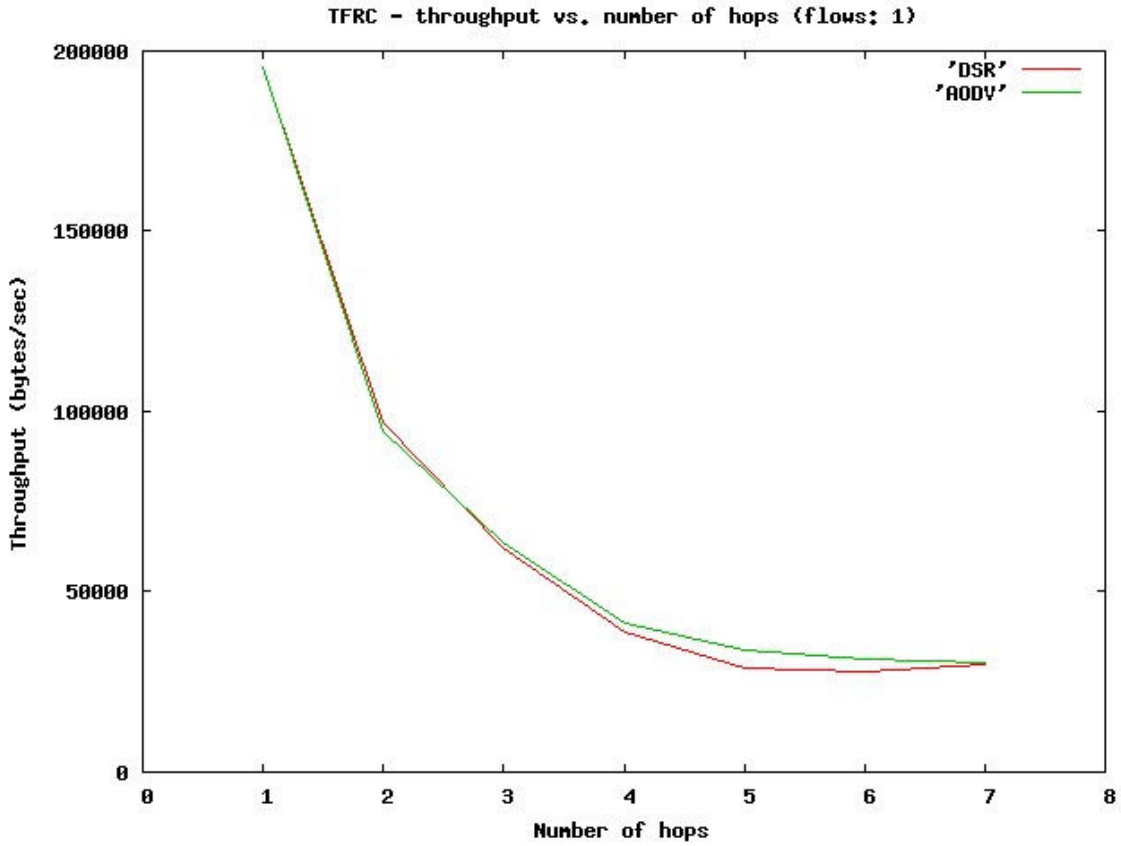


Figure 11: Comparison between AODV and DSR

These simulations were conducted using the ns simulator. We vary the number of nodes from 2 to 8 and measure throughput of TFRC when either DSR or AODV is employed. The results in Figure 11 confirm the predictions above in that DSR overhead may impact TFRC performance in such conditions. In this particular scenario where only one flow is present, AODV performs better than DSR. Hence, from a TFRC perspective, it is indeed difficult to say which protocol is the best. In fact, there are tradeoffs that must be taken into consideration to meet the application needs. Two principles mentioned in chapter 2 and 3 are that DSR was designed for small scale networks of up to about 200 nodes and only short-range multihop wireless networks are feasible today. Since theses are principles this thesis is built on, we therefore use DSR for our simulations throughout this thesis.

4.3 Interaction between TFRC and MAC Protocols

The interaction between TFRC and the IEEE 802.11 medium access control protocol is one of the most crucial problems to be addressed in multihop wireless networks.

The fact is that 802.11 relies on the assumption that every node can reach each other or at least sense any transmission into the medium, which is not always true in a multihop scenario. Consequently, in some conditions the hidden node and exposed node problems can arise, inducing instability and unfairness effects [15] which can impair not only TFRC throughput but also the fairness among simultaneous TFRC connections. We explain next, by means of simulation evaluation, how these problems can take place.

4.3.1 Impact of Hidden and Exposed Node Problems

As described in section 2.4.3, the 802.11 MAC protocol uses the short RTS/CTS control frames to prevent the hidden node problem and consequently the exposed node problem. While this strategy works efficiently for scenarios where a maximum of three hops can be established, it does not scale for larger scenarios. We discuss here how these problems can impair a TFRC connection. We use results from the simulation of one TFRC flow over a static multihop network from section 2.2.4.

In figure 4, the plotted values of the throughput are measured over 1.0 s intervals. We count the successively received TFRC packets in each 1.0s interval and transfer it into the throughput in that interval. Looking at Figure 4, in the 300s lifetime of this connection, they are many times when the throughput reached or neared zero. In those 1.0 s intervals, almost no TFRC packets were successively received, which means that TFRC performance degraded seriously.

We look into the simulation traces of this run. We focus on the packet trace of the period around 267.0s. In particular we follow the trace of a packet with sequence number ***26897*** which was dropped. A part of the MAC layer packet trace is shown in figure 12.

By analyzing the simulation trace, we find it is rooted in the MAC layer. Node 1 cannot reach node 2. After node 1 tries to contact node 2 and fails seven times, the MAC layer reports a link breakage. Note that a limit of seven retries is defined in IEEE 802.11. From the trace, we find that the hidden and the exposed station problem in node 2 prevent node 1 from reaching node 2. Since node 2 can sense node 4, which is sending a large data frame, it has to defer when node 4 is sending. The result is that a collision occurs at node 2 as it tries to receive the RTS from node 1. After failing to receive CTS from node 2 seven times, node 1 quits and reports a link breakage to its upper layer. Then a route failure event occurs. The routing protocol in node 1 then attempts to find a new route to the destination. In the string network topology under study, there is only one route from node 0 to node 7,

so the routing agent will eventually “re-discover” the same route again. The breaking and rediscovery of the path results in the drastic throughput oscillations observed. For a general network with multiple paths from source to destination, the same throughput oscillations will still be expected. This is because the declaration of the link failure is caused by self-interference of traffic of the same flow at adjacent nodes, which by itself delays the forwarding and in the worst case leads the TFRC sender to time out.

```
//source node sends packet number 26897
s 266.739997219 _1_ AGT  --- 26897 tcpFriend 1000 [0 0 0 0] ------- [1:0
5:0 32 0]
s 267.159455270 _4_ MAC --- 0 MAC 44 [125e 5 4 0] //node 4 sends RTS to
node 5
r 267.159631937 _5_ MAC  --- 0 MAC 44 [125e 5 4 0] //node 5 receives RTS
s 267.159641937 _5_ MAC  --- 0 MAC 38 [11bc 4 0 0] //node 5 sends CTS to
node 4
r 267.159794604 _4_ MAC  --- 0 MAC 38 [11bc 4 0 0] //4 receives CTS
//node 4 transmits a large data frame to node 5
s 267.159844604 _4_ MAC  --- 26887 tcpFriend 1092 [a2 5 4 800] -------
[1:0 5:0 32 5]
s 267.160452937 _1_ MAC  --- 0 MAC 44 [125e 2 1 0] //node one sends RTS
to node 2
D 267.160453604 _2_ MAC COL 0 MAC 44 [125e 2 1 0] //2 drops RTS due to
collision
D 267.160800937 _1_ MAC RET 0 MAC 44 [125e 2 1 0] //retry count exceeded
by 1
//
D 267.160800937 _1_ MAC --- 26897 tcpFriend 1040 [a2 2 1 800] -------
[1:0 5:0 32 2]
D 267.641608676 _1_ RTR TOUT 26897 tcpFriend 1020 [a2 2 1 800] -------
[1:0 5:0 32 2]
```

Figure 12: Hidden Node Phenomenon: Extract from tfrc-static.tr

When the hidden and exposed node problems cause the delivery path to be disconnected long enough to cause a retransmission timeout, TFRC will exponentially slow down and back off, which degrades performance.

The problems above get worse as the number of hops increase. As a consequence, TFRC end-to-end throughput decreases significantly as the number of hops grow as depicted in figure 13. This graphic was obtained by our simulation using the ns2 simulator and Gnuplot tool. We kept the parameters similar to those in section 2.2.4 and varied the number of nodes in each run of the experiment. One can see that the throughput decrease is pronounced for short number of hops (up to 4 hops), and then becomes less aggressive. This happens because the first nodes (from source to destination) in the chain interfere among themselves just as explained earlier in this section. As a result, the subsequent nodes

do not get enough packets to substantially contribute to the hidden and exposed node problems.

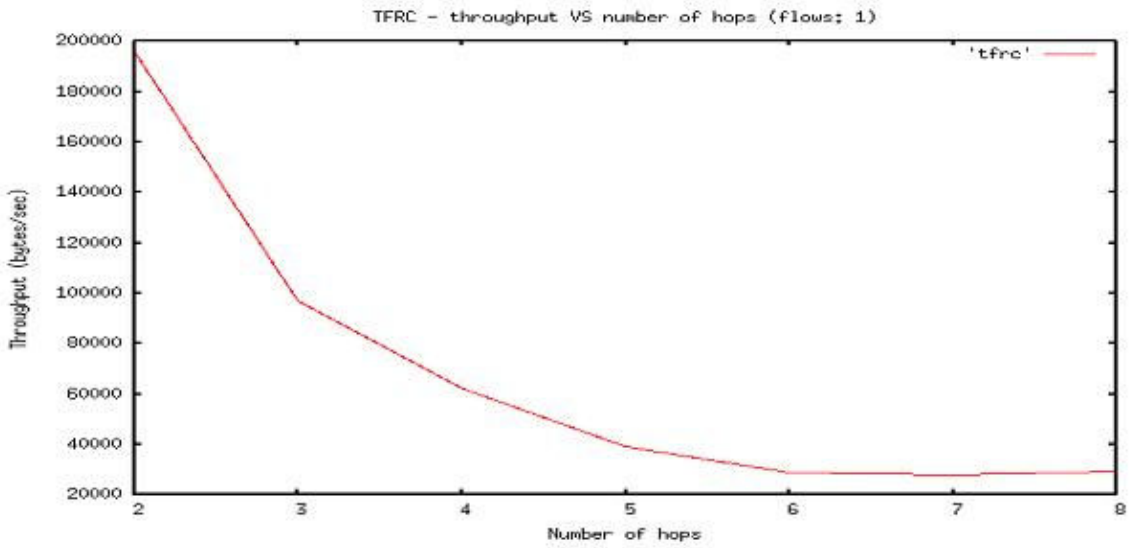


Figure 13: TFRC throughput decreases as the number of hops increase

4.3.2 Capture Effects

The IEEE 802.11 standard has serious problems of unfairness. Its binary exponential backoff mechanism is not efficient for multihop networks because it suffers from the capture effect phenomenon. A capture effect refers to the situation in which a node monopolizes the medium at the cost of the other nodes. This problem may be caused by either hidden node or exposed node problems [15].

In the experiment presented below, we set up two TFRC connections in the network shown in Figure 3. The first starts at 10.0 s, the second 100.0 s later. We will call them *first session* and *second session* in the following parts of this thesis. The whole experiment stops at 300.0 s. Figure 14 shows the throughput for a run of such an experiment calculated from our resulting data file called tfrc-unfair.tr. We used a tool called Tracegraph to plot this graphic. In this experiment, the first session is from 6 to 4, the second from 2 to 3. The first session is a two-hop TFRC. The first session has a throughput of around 100 kb/s after starting from 10.0 s. However, it is completely forced down after the second session starts at 100.0 s. In most of its lifetime after 100.0 s, the throughput of the first session is zero. There is not even a chance for it to restart. The aggregate throughput of these two TFRC connections completely belongs to the second session, around 200 kb/s in the lifetime from

100.0 s to 300.0 s. This is serious unfairness. The loser session is completely shut down even if it starts much earlier.

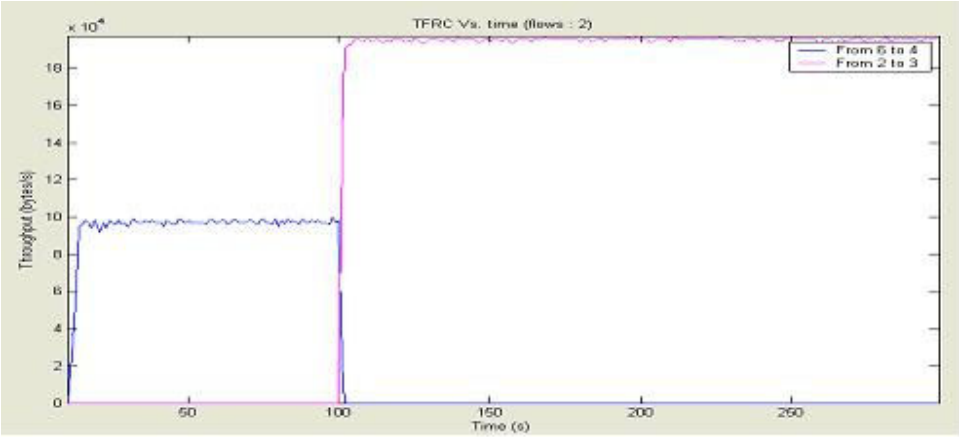


Figure 14: Throughput of two TFRC connections

By analyzing the simulation trace, we find that this problem is rooted in the MAC layer. Node 5 cannot reach node 4. After node 5 tries to contact node 4 and fails seven times, the MAC layer reports a link breakage. A part of the MAC layer packet trace is shown in figure 15.

It is clear that the major cause of node 5 failing to reach 4 is the collision. Since node 4 can sense node 2, a RTS from node 5 causes a collision at node 4. The result is that node 4 cannot send back CTS.

However, the TFRC connection from node 2 to 3 is only one hop. After node 2 receives a data packet (here it is a TFRC ACK) from 3, it sends out a RTS to request the channel, preparing to send out another TFRC packet. Once node 3 receives this RTS and replies with CTS, node 2 starts sending the TFRC packet. Normally, the size of this data packet is much larger than the control packets. If node 5 sends out an RTS for the channel to node 4, this control packet will experience a collision at node 4. So the only chance for node 5 to access the channel to node 4 is by sending out an RTS before node 2 sends out an RTS. Note that this must be after node 3 finishes sending back the data packet (TFRC ACK). The time window opening for node 5 to access the channel is very small. Also, because the binary exponential backoff scheme in the MAC layer always favors the last succeeding station (node 2 in this case), node 5 hardly wins the contention. After seven

failures, it will quit and report a link breakage to its upper layer. Then a route failure event occurs.

Our findings are in line with what is called *one-hop unfairness*. Since one-hop connection is the most popular case in a wireless ad hoc LAN, it is really an important problem that needs to be solved.

```
r 101.345074726 _2_ MAC  --- 9134 tcpFriendCtl 68 [a2 2 3 800] -------
[3:0 2:0 32 2]
r 101.345099726 _2_ AGT  --- 9134 tcpFriendCtl 68 [a2 2 3 800] -------
[3:0 2:0 32 2]
s 101.345406726 _2_ MAC  --- 0 MAC 44 [122e 3 2 0]
r 101.345583393 _3_ MAC  --- 0 MAC 44 [122e 3 2 0]
s 101.345593393 _3_ MAC  --- 0 MAC 38 [118c 2 0 0]
r 101.345746059 _2_ MAC  --- 0 MAC 38 [118c 2 0 0]
s 101.356181393 _2_ MAC  --- 9123 tcpFriend 1080 [a2 3 2 800] -------
[2:0 3:0 32 3]
s 101.356971059 _5_ MAC  --- 0 MAC 44 [123e 4 5 0]
D 101.356971726 _4_ MAC  COL 0 MAC 44 [123e 4 5 0]
s 101.357025594 _2_ AGT  --- 9138 tcpFriend 1000 [0 0 0 0] ------- [2:0
3:0 32 0]
s 101.357749059 _5_ MAC  --- 0 MAC 44 [123e 4 5 0]
D 101.357749726 _4_ MAC  COL 0 MAC 44 [123e 4 5 0]
D 101.358097059 _5_ MAC  RET 0 MAC 44 [123e 4 5 0]
D 101.358097059 _5_ RTR NRTE 8694 tcpFriend 1032 [a2 4 5 800] -------
[6:0 4:0 32 4]
D 101.358097059 _5_ RTR NRTE 8880 tcpFriend 1032 [a2 4 5 800] -------
[6:0 4:0 32 4]
D 101.358097059 _5_ RTR NRTE 8876 tcpFriend 1032 [a2 4 5 800] -------
[6:0 4:0 32 4]
```

Figure 15: One-hop Unfairness- Extract from tfrc-unfair.tr

4.3.2 Link Capacity

The interference range in a multihop network is typically slightly higher than twice the transmission range, 550m to be specific. This means that a given node can only communicate with nodes placed at a maximum distance defined by its transmission range but can interfere with other nodes located as distant as its interference range. As a consequence, the IEEE 802.11 standard establishes a spatial reuse property defining the maximum end-to-end capacity that can be achieved in multihop networks. Taking the spatial reuse property into consideration, the design of upper layer protocols can be optimized to mitigate the effects of the hidden node problem for scenarios with more than three hops end-to-end. This subject has been investigated in detail in [9].

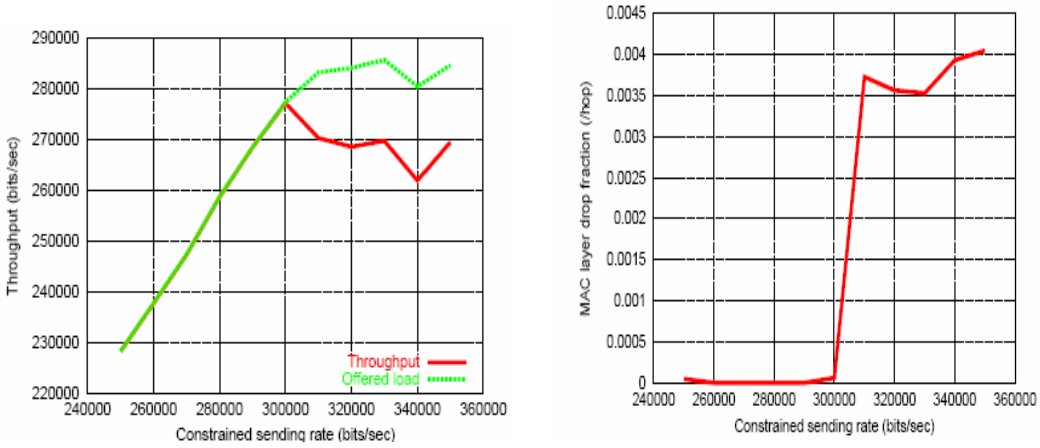
Research in [16] establishes the maximum throughput for an ad hoc network to be approximately $\frac{1}{5}$ to $\frac{1}{7}$ of the link capacity. Our simulations show the maximum achievable throughput for TFRC over a multihop wireless network to be significantly lower than the line capacity. Figures 4 and in particular figure 13 show that even for an ideal 3 node topology the maximum throughput is lower than the link capacity. Since Bianchi [17] showed that 802.11 MAC layer throughput decreases when offered load exceeds the saturation threshold, this lower-than expected throughput can be attributed to the RTS/CTS congestion [18], that occurs when the MAC layer becomes saturated. We scanned our output file from the script in Appendix 9.2.2 to determine the cause of packet loss. Figure 16 shows an extract of the drop statistics.

```
D 94.815068380 _2_ MAC COL 0
D 94.815415713 _1_ MAC RET 0
D 94.815415713 _1_ MAC --- 10857
D 94.819184439 _2_ MAC --- 10876
D 94.861543780 _1_ MAC --- 10899
D 94.921974447 _2_ MAC COL 0
D 94.923912447 _2_ MAC COL 0
D 94.961128447 _2_ MAC COL 0
D 186.463294087 _4_ MAC RET 0
D 186.463294087 _4_ MAC --- 21139
D 186.494888241 _2_ MAC COL 0
D 186.523617969 _2_ MAC COL 0
D 186.580489636 _2_ MAC COL 0
D 186.580836969 _1_ MAC RET 0
D 186.580836969 _1_ MAC --- 21149
D 186.583424201 _2_ MAC --- 21154
D 186.633183997 _1_ MAC --- 21162
D 186.707415663 _2_ MAC COL 0
D 186.732421928 _4_ MAC --- 21171
```

Figure 16: Extract of MAC layer Statistics

By making the interface queue large enough not to overflow, we ensure that the network does not become congested. As expected, all transport layer packet loss in our simulations is caused by MAC layer contention and frame drops when no transport layer congestion is induced. TFRC is supposed to react to network congestion which results in packet drops due to router buffer overflows; it is not tuned to respond to MAC layer congestion resulting in frame drops and hence does not reduce its sending rate appropriately.

To clarify the behavior of TFRC in overloading the 802.11 MAC layer in a multihop network figure 17(a) shows the TFRC offered load and throughput as the constrained sending rate is varied for a seven hop network . As the constrained rate increases, offered load and throughput increase linearly until a divergence occurs at approximately 300 Kbps. Beyond this point, increasing the constrained TFRC rate yields reduced throughput. The observed gap between offered load and throughput at high TFRC rates is due to lost packets. Figure 17(b) shows a sharp increase in MAC layer losses starting at about 300 Kbps, as the constrained sending rate increases.



a) Offered load and throughput vs. Constrained sending rate

b) MAC layer drop fraction vs. Constrained sending rate

Figure 17: TFRC sending rate vs. MAC layer [9]

From the results above, when unconstrained, TFRC produces an offered load that is above the rate sustainable by the multi-hop 802.11 MAC layer. The MAC layer then suffers from multiple frame retransmissions. Although TFRC eventually receives some packet loss notification because of the frame retransmissions, these packet losses arrive too late for TFRC to curtail its offered load below the saturation point of the MAC layer.

4.4 Cross-Layer Interaction in 802.11 Networks

The detailed problems in multihop 802.11 networks in this section have two implications. First, the behavior of the end user (i.e., transport) directly affects the degree of the extended hidden terminal problem because 802.11 cannot perfectly avoid signal interference by itself. In some sense, this is a violation of the layered architecture of IP

networks. Second, congestion occurs when transport layer attempts to push more packets beyond the optimal spatial channel utilization, and the outcome is link loss instead of queue overflow. Each of these points makes a fundamental difference in the system design over multihop 802.11 networks. There exists a cross-layer interaction among TFRC, routing and MAC in 802.11 networking environments. The interaction turns out to be very inefficient, affecting the overall system performance of 802.11 networks. Here is the multi-loop interaction model describing the interaction among TFRC, routing and MAC in 802.11 networks.

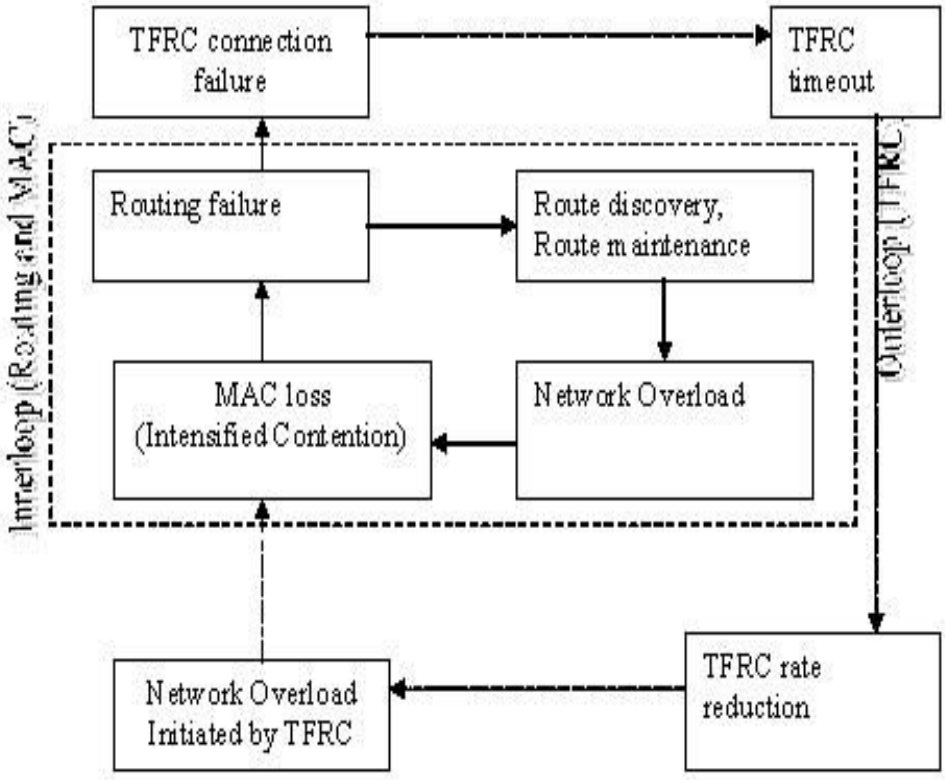


Figure 18: Connection cycle in chain topologies of 802.11 multi-hop networks

4.5 Summary

This chapter detailed the problems faced by TFRC over multihop networks. Wireless transmission medium constraints such as diverse environmental obstructions causing packet loss will mislead TFRC's response as a regular TFRC always handles packet losses as if they were caused by congestion.

This chapter also detailed extensive simulation evaluations of the problems created by the bidirectional flow established in a TFRC connection over a multihop network. The

root of these problems has been shown to be a result of the hidden and exposed node phenomenon. This phenomenon is a direct result of IEEE 802.11's use of RTS/CTS mechanism. While this mechanism works efficiently for scenarios where a maximum of three hops can be established, it does not scale for larger scenarios.

Further the RTS/CTS mechanism coupled with the transmission and interference ranges in the IEEE 802.11 standard establishes a spatial reuse property defining the maximum end-to-end capacity that can be achieved in multihop networks. Taking the spatial reuse property into consideration, the design of upper layer protocols can be optimized to mitigate the effects of the hidden node problem for scenarios with more than three hops end-to-end.

TFRC has been shown to provide load beyond the MAC capacity. In order for TFRC to mitigate the problems created by its bidirectional flow over multihop networks it needs to keep its sending rate below MAC threshold. To be able to adjust its sending rate to below the MAC layer saturation point, TFRC needs to determine the loss event rate that corresponds to the MAC layer congestion point. Research in [9] addresses this issue. Our work here addresses only the problems for TFRC over multihop networks and what should be done to address these problems. How TFRC should estimate the MAC saturation threshold is left for future work. In chapter 6 we include an algorithm developed in [9] for completeness only.

Chapter 5

TFRC Dedicated Response to Wireless Constraints

TFRC mechanisms are fine tuned not to cause collapse in the network over which it is running [1] [2]. As discussed in chapter 2, TFRC was first designed to work in wired networks, where packet loss may be safely associated to congestion. This chapter also detailed the main mechanisms of TFRC aimed at avoiding aggressiveness towards the network. These mechanisms lead a TFRC sender to slow down in the event of packet loss and then to increase the transmission rate gradually. The drawback with such an approach arises when TFRC is working in a wireless network. In such networks, packet losses are not only caused by congestion but also by the lossy nature of the wireless medium, as discussed in chapter 4.

As a result, TFRC performance may be impaired in scenarios where packet drops induced by the medium occur often. By too conservatively slowing down in the event of packet drops, TFRC may waste bandwidth since the communication channel is not really facing congestion. This problem can be avoided if the TFRC sender is informed about the nature of dropped packets. In other words, a TFRC sender should be able to discriminate between congestion and wireless medium induced losses. Having this information allows a sender to take proper decisions when reacting to losses.

This subject has been investigated by researchers in [10] as outlined in section 2.7.3. However we noted that in this research the interaction between TFRC and the MAC layer was not investigated in detail. Further, previous research has indicated that identifying the network states in section 2.5 should be necessary to improve the performance of TCP-friendly multimedia streaming over ad hoc networks [1] [2] [12] [13]. Given the results of our detailed investigations of the interaction between TFRC and the MAC layer in chapter 4, we now proceed to outline multihop network states that result from congestion and wireless medium induced losses. We include our recommended TFRC response function optimized for multihop networks in each event leading to the development of a state machine for our proposed ARETFRC algorithm.

5.1 Congestion Induced Losses

Congestion in ad hoc networks is defined as the signal that the offered load exceeds the network capacity. When congestion occurs, the queue size at router buffers will grow. When the capacity of these buffers is exceeded, packets will be dropped and the network throughput is reduced.

To deal with this kind of congestion, TFRC has been tuned to reduce its sending rate so as to maintain Internet stability. This TFRC response action has been optimized for wired networks were buffer overflows are a result of the interaction between transport protocols and routing devices at the network layer. However, given our results from chapter 4, this response action is not necessarily optimal for multihop networks. The dynamics of figure 18 produce an interaction among transport, network and MAC layers over a multihop network.

It might happen that MAC layer saturation results in route change, which further forces multiple flows to go through one routing node causing buffer overflow. To justify this assumption, figure 19 shows statistics of multihop events of the scene we developed for our mobile scenarios in section 2.2.4. Route changes are the most common events.

```
Destination Unreachables: 45

 Route Changes: 122

 Link Changes: 31

 Node | Route Changes | Link Changes
    0 |            42 |            7
    1 |            35 |           10
    2 |            33 |            9
    3 |            42 |           12
    4 |            28 |            9
    5 |            33 |            6
    6 |            31 |            9
```

Figure 19: Summary of Events in a Mobile Network

Therefore, a response action optimized for multihop networks would be, in the event of congestion reduce sending rate while ensuring that MAC capacity has not been exceeded.

5.2 Wireless Medium Induced Losses

Since the primary goal of TFRC is to be TCP friendly, congestion is given the highest priority in detection as a cause of packet loss. In multihop networks it will happen that packet loss occurs in the event of non-congestion, these losses can be attributed to the wireless medium as shown in section 4.3.2. Figure 19 shows these events and we summarize next how these non-congestion events cause packet loss.

5.2.1 Non-Congestion Channel Error

Channel error in ad hoc networks can be defined as any MAC layer transmission failures leading to frames being dropped. Results from our investigations in chapter 4, reveal that these transmission failures are a result of collisions due to the RTS/CTS mechanism of 802.11 coupled with the hidden/exposed node phenomenon. Section 4.3.2 also shows that theses packet losses occur in non-congestion conditions. In the event of channel error the receiver will experience random packet loss in most cases. The receiver should not count this as a congestion event at the network layer. Previous research [10] recommends that the sender should maintain its current sending rate.

Again given our results from chapter 4, this response action is not necessarily optimal for multihop networks. Therefore, the sender should maintain its current sending rate provided it is below MAC saturation level.

5.2.2 Non-Congestion Route Change

The delivery path between the two end hosts can change from time to time, but disconnections are too short to result in a retransmission timeout. Depending on routing protocols, the receiver may experience a short burst of out-of-order packet delivery or packet losses. Referring to figure 18, in both cases, the receiver should not treat it as congestion; and the sender should keep the streaming rate unchanged in the next RTT period granted it is below the MAC saturation level, waiting for the receiver to feedback more measurement statistics for the new path.

5.2.3 Non-Congestion Disconnection

Disconnection happens when packet delivery is interrupted for non-network congestion reasons long enough to trigger a retransmission timeout at the sender. Multiple

network conditions can trigger such a timeout at the sender including frequent route changes, heavy channel error, and mobility induced network partition.

When the delivery path is disconnected long enough to cause a retransmission timeout, according to TFRC specification the sender should cut the sending rate in half. Once the sender has received feedback from the receiver, it should modify its cached copy of the receive X_recv (receiving rate at receiver). Because the sending rate is limited to at most twice X_recv, modifying X_recv limits the current sending rate, but allows the sender to slow-start, doubling its sending rate each RTT. This is a specification fine tuned for wired networks.

Instead of exponentially slowing down over multihop networks, the sender should freeze the current sending rate and the retransmission timer. It then performs a periodic probing so that the transmission can be resumed promptly once a new path is established. This probing technique should be designed to avoid congesting the already bandwidth constrained medium of 802.11. Some probing techniques proposed for this purpose in previous research can be found in [19] [20] [21]. The sender leaves the probing state when a new feedback is received or the probing is timed out. The connection is closed after multiple probing attempts fail.

5.3 Summary

The states above take an action oriented classification. The response actions are not necessarily optimal. They represent the necessary states that TFRC uses to improve its performance. It might happen that MAC layer saturation results in route change, which further forces multiple flows to go through one routing node causing buffer overflow or bursty channel error might cause repeated link layer failures, so that route change or disconnections eventually take place. If TFRC were required to respond optimally to the different network states, such a response would result in a conflicting outcome. However, since our primary goal is to be TCP friendly, congestion is given the highest priority in detection. The other states are considered only if the network is not congested.

Metrics are needed to detect the identified states outlined in this chapter so far. Our work here addresses only how to identify the states not how to detect them. How these states should be detected is left for future work. The aim of our work was to identify the nature of packet loss as a result of cross-layer interaction on a multihop network. By identifying the nature of packet loss TFRC has the information it needs not to

conservatively slow down in the event of packet drops. Since TFRC derives TCP friendly sending rate based on end-to-end loss frequency measurement at the receiver side, we recommend adopting ad hoc network congestion detection at receiver side as it fits naturally to the existing TRFC protocol [2].

As an insightful introduction to the broad subject of network state detection we provide a brief discussion on related work in this field. Some previous work uses delay-related metrics to measure the congestion level of the network. For example, [21] and [22] use inter packet arrival delay, and [23] uses RTT to estimate the expected throughput. A challenge in ad hoc networks is that packet delay is not only influenced by network queue length, but also susceptible to other conditions such as random packet loss, routing path oscillations, MAC layer contention, etc. These conditions make such measurements highly noisy. Therefore, any single metric may not be reliable. To this end research in [10] proposes the use of multiple metrics to detect the congestion level of the network. This multiple metric scheme proved inefficient in the presence of channel error and mobility.

From our findings in chapter 4 and this chapter we note that the receiver will experience unique drop patterns. These are continuous, random and out-of-order packet loss for states congestion, channel error and route change respectively. By applying metrics that apply some form of either logic or pattern recognition we believe metrics can be devised that are optimized for multihop network conditions. This will be our future work.

Chapter 6

Results

The goal of this thesis was to investigate and detail the challenges for TFRC over multihop wireless ad hoc networks. The results of this investigation were projected to be a meaningful characterization of the effects of IEEE 802.11 on TFRC and a proposed algorithm to solve the involved problems optimized for the wireless networks. We present in this chapter these results.

6.1 ARETFRC State Machine

In our problem statement of section 1.2 we stated that research on TFRC has focused on picking one challenge and fine tuning TFRC to face that challenge. This was shown to be true in our account of related work, section 2.7. Our hypothesis that the challenges work in combination to degrade TFRC performance was shown to hold given results from chapter 4 and sections 5.1-2. Given these results TFRC has to be adapted in such a way that it meets the challenges in one state machine. Figure 20 is our proposal for such a state machine, ARETFRC. It shows a modified TFRC state machine that given the network state is able to choose an appropriate action while mitigating the problems created by the bidirectional flow established in a TFRC connection.

Adaptive rate estimation TFRC (ARETFRC) uses identical connection establishment and connection teardown processes similar to that of standard TFRC. It estimates the RTT and derives the sending rate identical to TFRC, by using the TCP throughput equation. To improve the performance of TFRC in ad hoc networks, ARETFRC makes several extensions at both the sender side and the receiver side.

6.1.1 Receiver Side

Upon each packet arrival at the receiver, besides the normal operations network states should be estimated. Depending on the scheme used the receiver should detect the state the network is experiencing. The receiver then passes this state estimate, i.e., congestion or channel error/route change, to the sender in every feedback packet. Besides regular feedbacks each RTT, the receiver should generate *Urgent* state update packets as soon as a congestion event is detected and send feedback to the sender immediately. This

way, information about persistent network conditions will likely be relayed to the sender in multiple feedback packets, providing robustness against possible losses of state report.

6.1.2 Sender Side

The sender maintains the most current state report received, and proceeds with normal TFRC operations until either of the following two events happens: the reception of a feedback packet, or the re-transmission timeout. A modified TFRC state diagram is shown in Figure 20 for the sender.

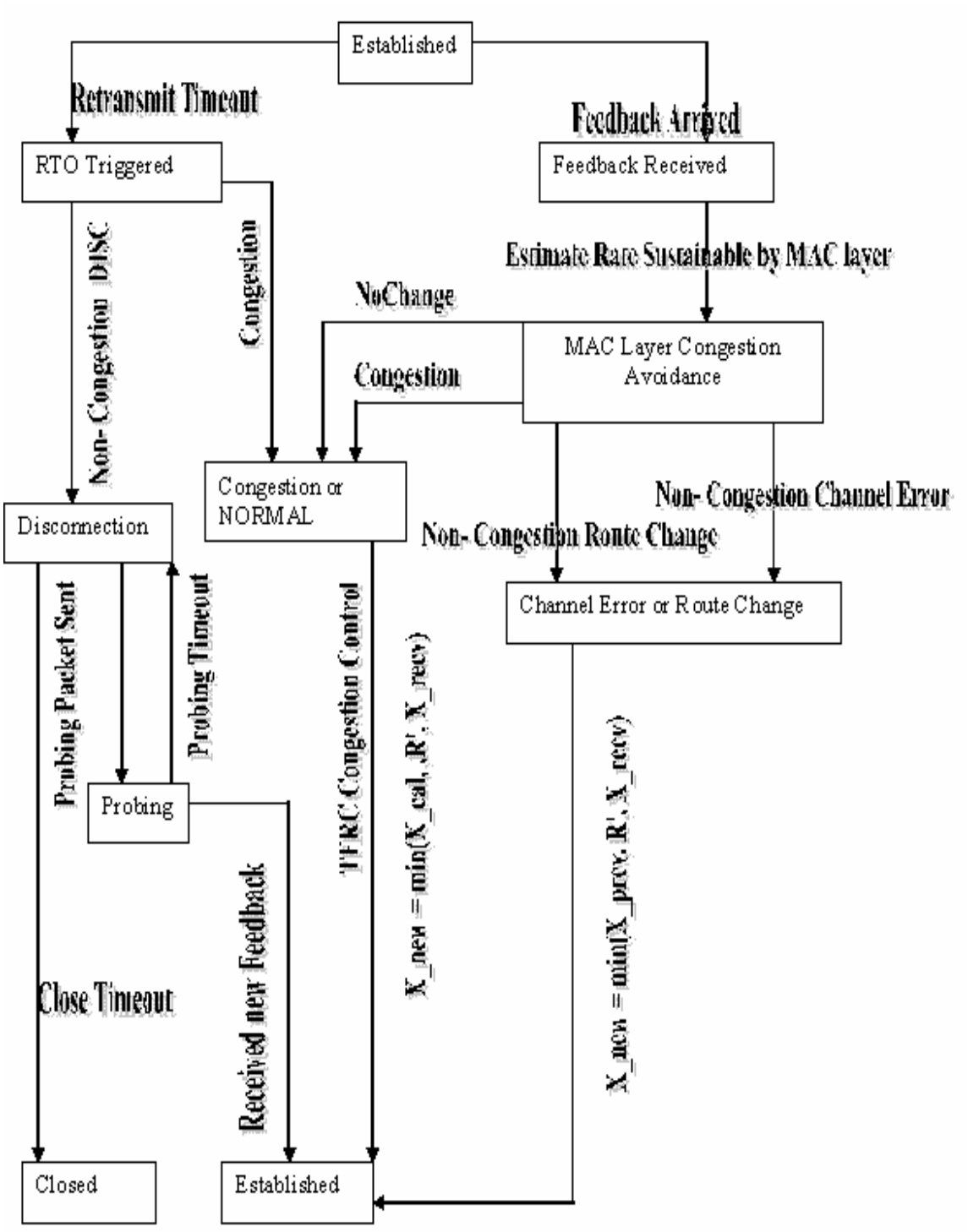


Figure 20: ARETFRC state diagram for sender

A feedback report or retransmission timeout triggers ARETFRC to take different control actions according to the current network state estimation. In particular, a MAC layer congestion avoidance state is introduced to deal with RTS/CTS induced congestion in ad hoc networks. The highlights of this state are for the sender to estimate a sending rate using an optimal round-trip time based on the network topology and equivalent loss event rate. The optimal round-trip time is estimated by modeling multihop contention delay and service time and the equivalent loss event rate is estimated using the inverse TCP Friendly rate equation with the optimal round-trip time. The basic idea is to infer the lower-layer MAC layer jamming in the upper layer TFRC to make it aware of lower layer congestion and reduce the jamming effects [9].

6.2 ARETFRC Pseudo Code

In this section we present the pseudo-code to illustrate the actions taken at both the sender and receiver. In our design, receiver-side identification is treated as an enhancement to TFRC in ad hoc networks that helps the sender to take more appropriate control and improve transport performance significantly. Without these enhancements, ARETFRC would behave exactly as TFRC.

6.2.1 Algorithm 1 Receiver Side: Upon Packet Arrival

1. apply scheme to detect network states
2. identify network state
3. ***if*** packet lost AND CONG ***then***
   Urgent = 1
   end if
4. update congestion frequency ***flost***
5. measure X_recv //receiving rate
6. ***if*** Urgent ***OR*** now_last feedback > feedback interval ***OR*** probing packet ***then***
   fill in ***flost*** and set state bits in option field
   transmit feedback
   end if

6.2.2 Algorithm 2 Sender Side: Upon Receiving Feedback

On receiving ack

1. ***if*** (not slowstart)
2. // non-congestion retransmission timeout

    if *probing state* ***then***

        probing state = 0

        probing count = 0

        resume next packet timer

    end if
3. measure RTT
4. //choose modeled RTT or smallest measured RTT

    r_{opt} = *max(r (N), min ([sliding window]))*
5. //Compute new loss event rate given RTT

    $p' = \overline{f}\ (r_{opt,}\ R)$
6. //compute sustainable TCP-Friendly rate

    $R' = f\ (r_{cur},\ p')$
7. //check the network state from the latest feedback;

    if *CONGESTION* ***OR*** *NORMAL* ***then***

    X_{calc} = *f* ($r_{cur,}$ *ack.p)* //compute original TCP-Friendly rate

    //if there is a rate change do so incrementally

        if ($rate_{cur}$ $> R'$)

            decrease_rate ()

            $X_{\_new} = min\ (X_{calc},\ R')$

        else

            $X_{\_new} = X_{calc}$

        end if

    else if *CHANNEL ERROR* ***OR*** *ROUTE CHANGE* ***then***

        $X_{new} = min\ (X_{\_prev},\ R')$ *//maintain current rate*

    end if

end if

Chapter 7

Conclusion

This thesis has identified key TFRC problems in multihop networks and proposed solutions. This chapter summarizes the problems addressed in the thesis, revises the proposed solutions, draws conclusions, and gives directions for future work.

7.1 Challenges and Solutions

Data transfer over multihop wireless networks is one of the most difficult tasks to be accomplished. Emerging transport protocols like TFRC face severe performance degradation over multihop networks given the noisy nature of wireless media as well as unstable connectivity conditions in place. The research community has been seeking ways to improve TFRC over such networks largely because of the emerging real-time applications that have been developed for streaming over such networks.

As discussed in chapter 5, TFRC was initially designed to work in wired networks. These networks rely on communication channels that experience generally very low bit error rates, typically much less than 1%. The communicating nodes in a wired network are normally fixed, i.e., these nodes do not change location often. Moreover, these traditional networks count on reasonable bandwidth resource to deliver data. In contrast, wireless networks do not encompass any of these characteristics fully. As a consequence, TFRC experiences substantial performance degradations in multihop wireless networks.

This thesis proposes solutions to two key problems faced by TFRC in such networks: mitigation of problems over multihop networks and association of losses to congestion. The RTS/CTS mechanism coupled with the transmission and interference ranges in the IEEE 802.11 standard establishes a spatial reuse property defining the maximum end-to-end capacity that can be achieved in multihop networks. TFRC has been shown to provide load beyond the MAC capacity. Taking the spatial reuse property into consideration, the design of upper layer protocols can be optimized to mitigate the effects of the hidden node problem for scenarios with more than three hops end-to-end. In order for TFRC to mitigate the problems created by its bidirectional flow over multihop networks this thesis proposes keeping its sending rate below MAC threshold, chapter 4.

TFRC associates dropped packets to network congestion. As a result, whenever a lost packet is perceived TFRC reduces its transmission rate to alleviate the presumed congestion inside the network. Congestion is not the only reason for packet loss in multihop networks. Rather, these wireless networks are prone to much higher bit error rates due to the medium nature. Hence, a conventional TFRC may waste precious bandwidth by reducing its transmission rate when reacting to a single drop caused by a random noise rather than by congestion.

This problem calls for a mechanism at the sending node to determine the actual reason of a dropped packet. Being aware of the nature of the loss allows the sender to react properly. This thesis has proposed a mechanism to perform packet loss discrimination at the sender, which is addressed in chapter 6.

7.2 Lessons Learned

Simulation reliability is an important subject that has been discussed in the research community. Some researchers do not trust simulation evaluations. From our experience, we believe that simulation can be as useful as real testbeds. An important point to be noticed here is that the simulators parameters have to be set realistically. This ensures that simulation results represent real life systems' behaviors very closely.

7.3 Limitations and Future Work

The proposal in this thesis serves as a general framework on which TFRC performance should be enhanced. As a framework with time complexity of $O(1)$ it serves as a good starting place for future work. Some obvious extensions of the present work include; how TFRC should estimate the MAC saturation threshold and devising schemes that can detect the identified ad hoc network states in this thesis.

References

[1] S. Floyd, M. Handley, J. Padhye, J. Widmer. Equation-Based Congestion Control for Unicast Applications. In *Proceedings of ACM SIGCOMM Conference*, pages 43-56. Stockholm, Sweden, August 2000.

[2] M.Handley, S.Floyd, J.Padhye J.Widmer. TCP Friendly Rate Control (TFRC) RFC 3448, IETF Network Working Group, January 2003.

[3] University of California Berkeley. The Network Simulator - ns-2. [Online] http://www.isi.edu/nsnam/ns/. Accessed, September 2005.

[4] S. Floyd, K. Fall. Promoting the Use of End-to-end Congestion Control in the Internet. *In proceedings of* IEEE/ACM Transactions on Networking. August 1999.

[5] IEEE. Standard 802.11-1999, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. In *The Institute of Electrical and Electronics Engineers*. 1999.

[6] C. Perkins, E. Belding-Royer, S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, IETF Network Working Group, December 2003.

[7] D. B. Johnson, D. A. Maltz, and Y. Hu. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR). Internet-draft, IETF MANET Working Group, July 2004. Draft-ietf-manet-dsr-10.txt.

[8] M. Chen. A. Zakhor. Rate Control for Streaming Video over Wireless. *Special issue on* Advances in Wireless Video, IEEE Wireleee Communications, vol. 12, issue 4, August 2005.

[9] M. Li, C. Lee, E. Agu, M. Claypool, R. Kinicki. Performance Enhancement of TFRC in Wireless Ad Hoc Networks. *In Proceedings of the 10th International Conference on Distributed Multimedia Systems (DMS),* Hotel Sofitel, San Francisco, California, USA September 8 - 10, 2004.

[10] Z. Fu, X. Meng and S. Lu.A Transport Protocol for Supporting Multimedia Streaming in Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications (JSAC)*, 21(10), December 2003.

[11] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, M. Gerla. The Impact of Multihop Wireless Channel on TCP Throughput and Loss. *In Proceedings of IEEE Infocom*, Mar. 2003.

[12] D.Bansal, H.Balakrishnan. Binomial Congestion Control Algorithms. In *Proceedings of IEEE Infocom, 2001.*

[13] I.Rhee, V.Ozdemir, Y.Yi. TEAR: TCP Emulation at Receivers-Flow Control for Multimedia Streaming. *Technical Report, NCSU, April 2000.*

[14] J.Liu. Transport layer protocols for wireless networks. Ph.D. Thesis. University of South Carolina, 1999, 117 pages. Discusses schemes for improving TCP performance for mobile ad hoc networks.

[15] S. Xu, T. Saadawi. Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks? IEEE Communication. June 2001.

[16] J. Li, C. Blake, D. D. Couto, H. I. Lee, R. Morris. Capacity of Ad Hoc Wireless Networks. In *Proceedings of ACM/IEEE MobiCom*, 2001, pp. 61–69.

[17] G. Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE Journal on Selected Areas in Communications*, Mar. 2000.

[18] S. Ray, J. Carruthers, D. Starobinski. RTS/CTS-induced Congestion in Ad-Hoc Wireless LANs. In *Proceedings of IEEE WCNC*, March 2003, pp. 1516–1521.

[19] G. Holland, N. Vaidya. Analysis of TCP performance over mobile ad hoc networks. *MOBICOM'99*.

[20] V. Tsaoussidis, H. Badr. TCP-Probing: Towards an Error Control Schema with Energy and Throughput Performance Gains. The 8th IEEE Conference on Network Protocols, November 2000.

[21] P. Sinha, N. Venkitaraman, R. Sivakumar and V. Bharghavan, "WTCP: A reliable transport protocol for wireless wide-area networks," *MOBICOM' 99*.

[22] S. Biaz, N.H. Vaidya. Distinguishing congestion losses from wireless transmission losses. *IEEE 7th International Conference on Computer Communications and Networks*, October 1998.

[23] L. Brakmo, S. O'Malley, L. Peterson. TCP Vegas: New techniques for congestion detection and avoidance. *ACM SIGCOMM 1994.*

Appendices

Appendix 1: TFRC Packet Type Extension in Ns2

1. // add to cmu-trace.h

```
void    format_tcpFriend (Packet *p, int offset);
void    format_tcpFriendCt1(Packet *p, int offset);
```

2. //add to cmu-trace.cc

```
#include <tfrc.h>
void    CMUTrace::format_tcpFriend(Packet *, int){
}
void    CMUTrace::format_tcpFriendCt1 (Packet *, int){
}
```

3. //add to method, void CMUTrace::format(Packet* p, const char *why)

```
case PT_TFRC:
        format_tcpFriend(p, offset);
        break;
case PT_TFRC_ACK:
        format_tcpFriendCt1(p, offset);
        break;
```

Appendix 2: Gnuplot Script Example: commands for figure 11

```
Terminal type set to 'x11'
gnuplot> set title 'TFRC - throughput vs. number of hops  (flows: 1)'
gnuplot> set xlabel 'Number of hops'
gnuplot> set ylabel 'Throughput (bytes/sec)'
gnuplot> set yrange [0.0:200000.0]
gnuplot> set xrange [0.0:300.0]
gnuplot> plot 'DSR' w lines 1, 'AODV' w lines
```

Appendix 3: *TFRC-static.tcl*

```
# tfrc-static.tcl
# ========================================================
# Define options
# ========================================================
set val(chan)          Channel/WirelessChannel          ;# channel type
set val(prop)          Propagation/TwoRayGround         ;# radio-propagation model
set val(netif)         Phy/WirelessPhy                  ;# network interface type
set val(mac)           Mac/802_11                       ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue          ;# interface queue type
set val(ll)            LL                               ;# link layer type
set val(ant)           Antenna/OmniAntenna              ;# antenna model
set val(ifqlen)        50                       ;# max packet in ifq
set val(nn)            8                        ;# number of mobilenodes
set val(rp)            DSR                       ;# routing protocol
set val(x)         1500
set val(y)         400
set val(stop)      300              ;# time of simulation end
# ========================================================
# Main Program
# ========================================================
# Initialize Global Variables
set ns_                [new Simulator]
set tracefd        [open tfrc-static.tr w]
set namtrace               [open tfrc-static.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

# set up topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)

# Create God
create-god $val(nn)
```

```
# configure node
        $ns_ node-config -adhocRouting $val(rp) \
                         -llType $val(ll) \
                         -macType $val(mac) \
                         -ifqType $val(ifq) \
                         -ifqLen $val(ifqlen) \
                         -antType $val(ant) \
                         -propType $val(prop) \
                         -phyType $val(netif) \
                         -topoInstance $topo \
                        -channelType $val(chan) \
                         -agentTrace ON \
                         -routerTrace OFF \
                         -macTrace ON \
                         -movementTrace OFF

        for {set i 0} {$i < $val(nn) } {incr i} {
                set node_($i) [$ns_ node]
        }
# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenodes
$node_(0) set X_ 10.0
$node_(0) set Y_ 200.0
$node_(0) set Z_ 0.0
$node_(1) set X_ 210.0
$node_(1) set Y_ 200.0
$node_(1) set Z_ 0.0
$node_(2) set X_ 410.0
$node_(2) set Y_ 200.0
$node_(2) set Z_ 0.0
$node_(3) set X_ 610.0
$node_(3) set Y_ 200.0
$node_(3) set Z_ 0.0
```

```
$node_(4) set X_ 810.0
$node_(4) set Y_ 200.0
$node_(4) set Z_ 0.0
$node_(5) set X_ 1010.0
$node_(5) set Y_ 200.0
$node_(5) set Z_ 0.0
$node_(6) set X_ 1210.0
$node_(6) set Y_ 200.0
$node_(6) set Z_ 0.0
$node_(7) set X_ 1410.0
$node_(7) set Y_ 200.0
$node_(7) set Z_ 0.0
# TFRC connections between node_(1) and node_(5)
set tfrc [new Agent/TFRC]
set tfrcsink [new Agent/TFRCSink]
$tfrc set class_ 2
$tfrcsink set class- 2
$ns_ attach-agent $node_(1) $tfrc
$ns_ attach-agent $node_(5) $tfrcsink
$ns_ connect $tfrc $tfrcsink
$tfrc set packetSize_ 1000
$ns_ at 0.1 "$tfrc start"
$ns_ at 300.0 "$tfrc stop"
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns_ initial_node_pos $node_($i) 30
}
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop) "$node_($i) reset";
}
$ns_ at $val(stop) "$ns_ nam-end-wireless $val(stop)"
$ns_ at $val(stop) "stop"
```

```
$ns_ at 300.01 "puts \"end simulation...\" ; $ns_ halt"
proc stop {} {
   global ns_ tracefd namtrace
   $ns_flush-trace
   close $tracefd
   close $namtrace
}
puts "Starting Simulation..."
$ns_ run
```

Appendix 4: *TCP-static.tcl*

```
# tcp-static.tcl
==========================================================
# Define options
// SAME AS FOR TFRC-Static.tcl
==========================================================
# Main Program
==========================================================
# Initialize Global Variables
set ns_                 [new Simulator]
set tracefd       [open tcp-static.tr w]
set namtrace            [open tcp-static.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
# set up topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
# Create God
create-god $val(nn)
# configure node
```

```
//SAME AS FOR TFRC-Static.tcl
```

```
        for {set i 0} {$i < $val(nn) } {incr i} {
```

```
            set node_($i) [$ns_ node]
        }
# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenodes
```

//SAME AS FOR TFRC-STATIC.TCL

```
# Setup traffic flow between nodes
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(1) $tcp
$ns_ attach-agent $node_(5) $sink
$ns_ connect $tcp $sink
$tcp set packetSize_ 1000
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 0.1 "$ftp start"
$ns_ at 300.0 "$ftp stop"
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns_ initial_node_pos $node_($i) 30
}
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop) "$node_($i) reset";
}
$ns_ at $val(stop) "$ns_ nam-end-wireless $val(stop)"
$ns_ at $val(stop) "stop"
$ns_ at 300.01 "puts \"end simulation...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd namtrace
    $ns_flush-trace
    close $tracefd
    close $namtrace
```

```
}
puts "Starting Simulation..."
$ns_ run
```

Appendix 5: Throughput.pl

```
$infile=$ARGV[0];
$tonode=$ARGV[1];
$Granularity=$ARGV[2];
$sum=0;
$clock=0;
        open (DATA,"<$infile")
        || die "Can't open $infile $!";
        while (<DATA>) {
                @x = split(' ');
if ($x[1]-$clock <= $Granularity)
{if ($x[0] eq 'r')
        {if ($x[2] eq $tonode)
                {if ($x[3] eq 'AGT')
                        {if ($x[6] eq 'tcpFriend')
                                {$sum=$sum+$x[7];
                                }}}}}
else
{
        $throughput=$sum;
        print "$x[1] $throughput\n";
        $clock=$clock+$Granularity;
        $sum=0;
        }}
                close DATA;
                 exit(0);
```

Appendix 6: TFRC-mobile.tcl

```
# tfrc-mobile.tcl
==================================================
```

```
# Define options
==========================================================
set val(chan)          Channel/WirelessChannel    ;# channel type
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy            ;# network interface type
set val(mac)           Mac/802_11                 ;# MAC type
set val(ifq)           Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)            LL                         ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                          ;# max packet in ifq
set val(nn)            8                          ;# number of mobilenodes
set val(rp)            DSR                        ;# routing protocol
set val(sc)                "scene-400-800"
set val(x)                 400
set val(y)                 800
set val(stop)    300                                ;# time of simulation end
==========================================================
# Main Program
==========================================================
# Initialize Global Variables
set ns_                [new Simulator]
set tracefd       [open tfrc-mobile.tr w]
set namtrace           [open tfrc-mobile.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
# set up topography object
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
# Create God
set god_ [create-god $val(nn)]
# configure node
        $ns_ node-config -adhocRouting $val(rp) \
                         -llType $val(ll) \
```

```
                -macType $val(mac) \
                -ifqType $val(ifq) \
                -ifqLen $val(ifqlen) \
                -antType $val(ant) \
                -propType $val(prop) \
                -phyType $val(netif) \
                -topoInstance $topo \
                -channelType $val(chan) \
                -agentTrace ON \
                -routerTrace OFF \
                -macTrace OFF \
                -movementTrace OFF

        for {set i 0} {$i < $val(nn) } {incr i} {
                set node_($i) [$ns_ node]
        }
# Define traffic model
puts "Loading scenario file..."
source $val(sc)
# TFRC connections between node_(0) and node_(3)
set tfrc [new Agent/TFRC]
set tfrcsink [new Agent/TFRCSink]
$tfrc set class_ 2
$tfrcsink set class- 2
$ns_ attach-agent $node_(0) $tfrc
$ns_ attach-agent $node_(3) $tfrcsink
$ns_ connect $tfrc $tfrcsink
$tfrc set packetSize_ 1000
$ns_ at 0.1 "$tfrc start"
$ns_ at 300.0 "$tfrc stop"
#Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
```

```
$ns_ initial_node_pos $node_($i) 30
}
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop) "$node_($i) reset";
}
$ns_ at $val(stop) "$ns_ nam-end-wireless $val(stop)"
$ns_ at $val(stop) "stop"
$ns_ at 300.01 "puts \"end simulation...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd namtrace
    $ns_ flush-trace
    close $tracefd
    close $namtrace
}
puts "Starting Simulation..."
$ns_ run
```

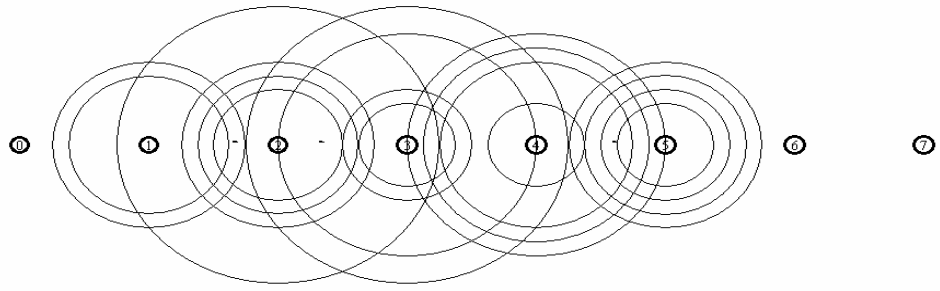
Appendix 7: *TCP-mobile.tcl*

```
# tcp-mobile.tcl
=========================================================
# Define options
//SAME AS TFRC-Mobile.tcl
=========================================================
# Main Program
=========================================================
# Initialize Global Variables
set ns_                 [new Simulator]
set tracefd     [open tcp-mobile.tr w]
set namtrace            [open tcp-mobile.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
# set up topography object
```

```
set topo       [new Topography]
$topo load_flatgrid $val(x) $val(y)
# Create God
set god_ [create-god $val(nn)]
# configure node
//SAME AS TFRC-Mobile.tcl
                    for {set i 0} {$i < $val(nn) } {incr i} {
                     set node_($i) [$ns_ node]
          }
# Define traffic model
puts "Loading scenario file..."
source $val(sc)
# TCP connections between node_(0) and node_(3)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(3) $sink
$ns_ connect $tcp $sink
$tcp set packetSize_ 1000
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 0.1 "$ftp start"
#Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 30 defines the node size for nam
$ns_ initial_node_pos $node_($i) 30
}
# Tell nodes when the simulation ends
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop) "$node_($i) reset";
}
$ns_ at $val(stop) "$ns_ nam-end-wireless $val(stop)"
```

```
$ns_ at $val(stop) "stop"
$ns_ at 300.01 "puts \"end simulation...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd namtrace
    $ns_flush-trace
    close $tracefd
    close $namtrace
}
puts "Starting Simulation..."
$ns_ run
```

Appendix 8: String Multihop topology with 8 nodes



Appendix 9: Random Multihop topology with 8 nodes