ATP FOR LOSSY WIRELESS LINKS IN AD HOC NETWORKS

Dissertation submitted to the Jawaharlal Nehru University in partial fulfillment of the requirement for the award of the degree of

MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE AND TECHNOLOGY

By
GOLDY

UNDER THE SUPERVISION OF
Dr. D. K. Lobiyal

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067, INDIA
JULY 2007

Dedicated to

My Parent, Supervisor and Sister

जवाहरलाल नेहरू विश्वविद्यालय

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
SCHOOL OF INTERNATIONAL STUDIES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110067(INDIA)

CERTIFICATE

This is to certify that the dissertation entitled "**ATP for Lossy Wireless links in Ad Hoc Networks**" being submitted by Ms. **GOLDY** to **School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi** in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Computer Science and Technology**, is a bonafide work carried out by her in her the School of computer and Systems Sciences. The matter embodied in the dissertation has not been submitted for the award of any degree or diploma.

[illegible]

Dr. D.K.Lobiyal
Associate Professor
School of Computer and Systems Sciences
Jawaharlal Nehru University
New Delhi-110067

[illegible]

Prof Parimala N.
Dean
School of Computer & Systems Sciences
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI-110067

Dean
School of Computer and Systems Sciences
Jawaharlal Nehru University
New Delhi-110067

जवाहरलाल नेहरू विश्वविद्यालय

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
SCHOOL OF INTERNATIONAL STUDIES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110067(INDIA)

DECLARATION

This is to certify that the dissertation titled **"ATP for Lossy Wireless links in Ad Hoc Networks "**, which is being submitted to the **School of Computer & Systems Sciences, Jawaharlal Nehru University, New Delhi,** in partial fulfillment of the requirements for the award of **Master of Technology in Computer Science & Technology** is a bonafide work carried out by me.

The matter embodied in the dissertation has not been submitted for the award of any other degree or diploma.

Goldy.

GOLDY
M.Tech(2005-2007)
SC & SS, JNU,
New Delhi - 110067.

ACKNOWLEDGEMENT

I am very glad to express my sincere gratitude and thanks to my supervisor **Dr. D. K. Lobiyal** for his expert guidance in completing this work. Without his constructive suggestions, critical comments, encouragement, help and kind cooperation, this dissertation would not have taken a shape. I feel it is a great privilege to have had the opportunity to work under his prestigious supervision. His constant encouragement and support helped me a lot. He led me into the exciting field of Ad Hoc Networks and taught me how to approach the problem in rigorous way. The methodology and philosophy that I have learned in my research will definitely benefit my career for life.

I want to express my gratitude to the member of my seminar committee, Prof. K.K. Bhardwaj, Dr. D.P Vidhyarthi & Dr R.K Agrawal for their valuable help, comments and suggestions on the synopsis of this dissertation. Special thanks to Dr. D.K Lobiyal for many helpful discussions.

I thank all the Faculty Members in School of Computer and Systems Sciences who have always been helpful in any of the career perspectives.

I am thankful to **Dean** of the school and administration of JNU for providing a congenial environment for making our work a success. Their academic support has been a real asset in completing this work.

I want to thank the Librarian of software library .

I am also thankful to **Yaser** for his valuable suggestions provided during my dissertation work.

Thanks to my class mates **Yamini**, **Abhishek**, **Sunil**, **Sachin**, **Sonia**, **Arun**, **Ashish**, and all others for their discussions and help at all time during the course of my dissertation work. I also wish them all the best for submitting their dissertations. I also thank all my seniors and juniors by name for their encouragement and support.

Finally, I would like to thank Vikas, my parents and family. I could not have gone through this long journey without their constant love and support. I owe them so much

Goldy.

Goldy

ABSTRACT

TCP a widely used well-known protocol is not suitable for Ad Hoc Networks since it has been primarily designed for wired network. To overcome this problem researchers have provided many solutions that are either variants of TCP or independent of TCP. ATP is one of the protocols designed for ad hoc network independent of TCP. However ATP has limitations due to simple assumptions links are loss-less and epoch time is fixed made in its designing. But in general wireless link are lossy and bandwidth has impact on epoch time since data rate depends on bandwidth. Therefore, the work in the current dissertation is a modest attempt to improve ATP protocol by considering Lossy Wireless Links and Adjusting Epoch Timer according to the different wireless bandwidths.

We have GloMoSim for simulation of our work. In GloMoSim there does not provide user interface at transport layer to embed new protocol unlike other layers. Therefore, we have made modifications in TCP to embed the functionality of ATP so that proposed modifications can be realized and tested.

The work presented in this dissertation is an outcome of simulations conducted using GloMoSim. The results are obtained in terms of three metrics throughput, success rate and energy consumption. Experiments are performed for different network size of 20, 30, 50, and 70 nodes. These results considered as an average of 10 simulations runs for experiment.

From the results of simulation, we observed that the overall performance of ATP is better in terms of success rate and energy consumption. However TCP shows better results in the beginning. But for the throughput the results are not comparable due to unpredictable behavior exhibited by TCP.

CONTENTS

LIST OF FIGURES

ABBREVIATIONS

ABR	:	Associativity Based Routing
Ack	:	Acknowledgement
ACTP	:	Application Controlled Transport Protocol
ADSN	:	Acknowledgement Duplication Sequence Number
AODV	:	Ad Hoc On – demand Distance Vector Routing
API	:	Application Program Interface
ATCP	:	Ad Hoc Transmission Control Protocol
ATP	:	Ad hoc Transport Protocol
BER	:	Bit Error Rate
DSR	:	Dynamic Source Routing
ECN	:	Explicit Congestion Notification
ELFN	:	Explicit Link Failure Notification
ERDN	:	Explicit Route Disconnection Notification
ERSN	:	Explicit Route Successful Notification
CSMA/CA	:	Carrier Sense Multiple Access/ Collision Detection
CDMA	:	Code division Multiple Access
FDMA	:	Frequency Division Multiple Access
ICMP	:	Internet Controlled Message protocol
LACK	:	Local Acknowledgement
LQ	:	Localized Query
MSS	:	Maximum Segment Size
MAC	:	Medium Access Control
MANETs	:	Mobile Ad Hoc Networks
OOO	:	Out-Of-Order event
PN	:	Pivoting Node
RTO	:	Retransmission Time Out
Rtt	:	Round Trip Time
RFN	:	Route Failure Notification
RRC	:	Route Re- Construction
RRN	:	Route Re –establishment Notification
SACK	:	Selective Acknowledgement
ssthresh	:	Slow-Start Threshold
SANETs	:	Static Ad Hoc Network
TPSN	:	TCP Packet Sequence Number
TORA	:	Temporally Ordered Routing Algorithm
TDMA	:	Time Division Multiple Access
TCP	:	Transmission Control Protocol
UDP	:	User Datagram Protocol

Chapter1

CHAPTER 1

Introduction

Mobile Wireless Networks allow users to access information and services regardless of their geographic position. There are two types of mobile wireless networks - Infrastructured networks or cellular networks, and Infrastructureless networks or Ad Hoc Network.

1.1 Background

Mobile Ad Hoc Network (MANET) is a transient network formed dynamically by collection of arbitrarily located wireless mobile nodes that communicate to one other through broadcast radio transmission. However due to radio range limitation, physical broadcasting does not cover all terminals. Therefore each node acts both as host and as a router. Nodes that cannot establish a point-to-point connection because they are too distant relay packets to intermediate nodes that act as routers [12]. Therefore, in MANET, packets travel from source to destination hopping from node to node in multi-hop fashion. [1].

MANETs are characterized by scarce wireless bandwidth, highly dynamic topology due to node mobility, varying propagation characteristics and network scalability. Therefore it gives rise to many issues at the network layer, medium access layer, transport layer and physical layer. At network layer the main problem is of routing packets due to dynamic topology, power constraints and error prone nature of wireless medium.

The choice of Medium Access Scheme is also a challenge for ad hoc networks. Use of TDMA or dynamic assignment of frequency bands is difficult because of the absence of centralized control in ad hoc networks. FDMA is inefficient for dense networks and CDMA is difficult to implement due to node mobility and needs to keep track of frequency- hopping patterns in dynamic manner. However Random Access appear to be practical, but with the drawback of "hidden" and "exposed" terminal problem. [23].

At physical layer power control is the main aim. Transmission power of nodes needs to be regulated so that it is high enough to reach intended receiver while causing minimal interference to interference to other nodes. Iterative power control algorithms have been devised in [22] [27].

At transport layer there are two protocols: TCP and UDP. UDP is simple but being unreliable and connectionless is not suitable for ad hoc networks. TCP a reliable, byte-stream based and connection oriented protocol is tuned to give good performance in wired network but is also not suitable for MANET.

The differences in the characteristics of wired and wireless network pose to be a problem for transport layer protocol. Therefore to develop a framework for a protocol that suits the distinctive nature of ad hoc networks the effect of each of these characteristics on transport layer protocol are given as under

Power Scarcity and limited bandwidth: Mobile hosts are powered by a battery that is always limited and wireless links have narrow frequency band for data transmission in contrast to wireline networks. Additionally due to all wireless nature of ad hoc network, not only the flows in the same vicinity contend with each other, but part of a flow traversing multiple hops can contend with other parts of the same flow in its vicinity [18]. Therefore transport layer protocol has to function in a constrained manner.

Dynamic network topology: As hosts are always mobile in network, therefore, the topology is dynamic which results in frequent path breaks, partitioning and remerging of networks, and high delays in re-establishment of path [4]. Therefore performance of transport layer protocol is drastically affected [4].

High Bit Error rate: Wireless links are prone to errors, which results to loss in data and acknowledgement packets.

Non-congestion delay: Lost packets and long delays in mobile environment are not always due to congestion but due to wireless links being susceptible to error. Therefore misinterpretation of packet loss as congestion results in invocation of congestion control algorithm in the case of not false network congestion [10]. Jacobson [15] assumed that chances of packets being lost due to damage are less as compared to losses due to network congestion. In Jacobson [15] congestion control schemes is not sensitive to damage loss. In ad hoc networks high loss rates degrades TCP throughput by 60% [10]. Furthermore the congestion window shrinking as one of the congestion control scheme exaggerates this problem. Thus conventional methods of identifying congestion in ad hoc network such as packet loss and retransmission time out are not suitable.

Serial Time Outs: When retransmission timer at the sender is doubled with each unsuccessful retransmission attempt, in order to reduce the transmission rate, TCP takes a long time to recover from such a reduction when the mobile is reconnected. Thus frequent disconnections cause a condition called serial timeouts at TCP sender [10].

Packet size variation: Size of packet that is transmitted over the wired links is much larger as compared to wireless links. Thus every packet on the wired network gets fragmented when transmitted over wireless link. Therefore packet size is the key factor in regulating the performance.

Induced Traffic: Ad hoc networks are characterized as multihop network therefore a path from source to destination consists of multihop links and transmission at particular link affects one upstream and one downstream link. This traffic at given link (or path) due to traffic through neighboring links (or paths) is referred to as induced traffic [4].

Induced throughput unfairness: This is the throughput unfairness experienced at the transport layer due to unfairness existing at the lower layers i.e. network and MAC. At the MAC layer nodes contend for the channel and the node that gets access to the channel monopolizes it for more than one transmission even if other nodes are waiting to gain access to channel [2].

Completely decoupled Transport Layer: As we know that ad hoc networks have a dynamic topology, therefore the transport layer has to adapt to changing network environment for this it needs cross- layer interaction with network layer and MAC layer whereas in wired network everything is static therefore there is no need of any cross layer interaction. Thus this is another challenge faced by transport layer in ad hoc networks [4].

1.2 Regular TCP in wired network

TCP is a connection-oriented protocol tuned to perform well in wired network. The major responsibilities of TCP are congestion control, flow control, reliable delivery of packets to destination. It is a ***window-based protocol*** that regulates the number of packets send by inflating and deflating its window. TCP sender starts the session with the window size of 1 MSS (Maximum segment size). It sends one MSS and waits for acknowledgement, once the acknowledgement is received within retransmission time out (RTO) period, window is doubled and two MSS's are sent. This doubling of congestion window with every successful acknowledgement of all segments in current congestion window is called as ***slow-start*** and it continues until the window reaches the slow-start threshold (*ssthresh*). Once it reaches *ssthresh* it grows linearly, adding one segment to congestion window for every received acknowledgement until it reaches the receiver advertised window (***Linear increase***); this phase is called as ***congestion avoidance*** [4].

TCP continually measures how long acknowledgements take to return to determine which packets have reached the receiver and provides reliability by retransmitting lost packets. For this purpose it maintains running average of this delay (round trip delay or RTT) and an estimate of expected deviation from this average (rtt_{dev}). If current delay is longer than the average by more than four times the expected deviation (time out interval or RTO), TCP assumes that the packet was lost, TCP then retransmits the lost packet [10].

RTT for i^{th} packet is [18]

$Rtt_{base} + (i-1) * L/r$

Where, Rtt_{base} is the base round trip delay of underlying path

L is the length of a packet and

r is the available rate

RTO (retransmission time out) calculation is [18]

$rtt_{avg} + 4 * rtt_{dev}$

Where, rtt_{avg} is the exponentially average of rtt samples observed

rtt_{dev} is standard deviation of rtt samples

TCP reacts to packet loss in network by invoking congestion control mechanisms. In this mechanism TCP sender reduces slow- start threshold *ssthresh* half the current congestion window or 2 MSS whichever is larger to decrease the amount of data [4]& [10] and then resets the congestion window to 1 MSS and activates slow-start algorithm (***multiplicative decrease***) therefore restricting the rate at which the window grows to previous levels. It also resets RTO with an exponential back off value which doubles with every subsequent retransmission according to Karns algorithm [20], [4] & [10] and again the same procedure is followed. Moreover if TCP sender receives three duplicate acknowledgements (repeated acknowledgements for the same TCP segment that was successfully received in-order at the receiver) it perceives it as packet loss and invokes *Fast Retransmit Algorithm.* This algorithm assumes that the missing packet starts with the sequence number that is equal to the number acknowledged by the duplicate acknowledgements, and thus retransmits it [4] & [10].

1.3 Regular TCP not suitable for ad hoc networks

When a packet is lost regular TCP assumes that there is congestion. But in wireless Ad hoc networks packet loss can be due to error prone nature of wireless link, increased collision due to hidden terminal, presence of interference, unidirectional links, frequent path breaks due to mobility of nodes & inherent fading properties of wireless channel [book]. Thus it is not always the case that packet loss is due to congestion in wireless network. But regular TCP trigger congestion control that results in decrease in throughput and long delays, hence degrading the performance. Additionally the bit errors on wireless links are bursty and frequent. There are two states: good and Bad. In good state wireless link does not suffer from any losses, while in bad state transmitted packets are corrupted [halla]. Both states are exponentially distributed with mean α_g and α_b respectively. The values of α_g and α_b are subject to following relationship

$$\alpha_b = BER * packet\ size\ in\ bytes * 8 * \alpha_g$$

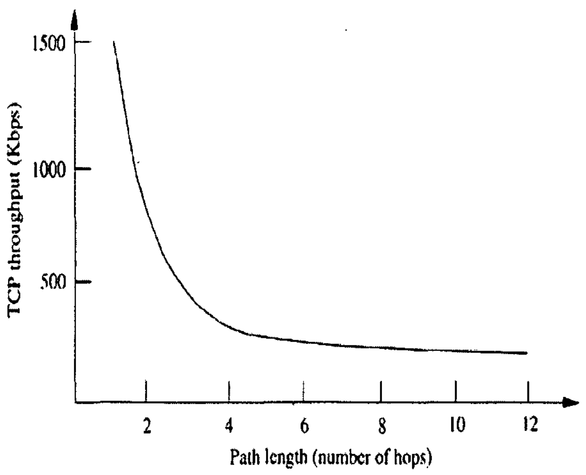
Choice of exponential distribution allows for error bursts. This is attributed to the fact that standard deviation of exponential distribution is equal to its mean [10].

Ad hoc networks have dynamic topology due to mobility of nodes, this leads to frequent path breaks and thus route to destination needs to be recomputed very often. The responsibility of finding the route and reestablishment is given to network layer and if the route reestablishment time is more than RTO the TCP sender assumes congestion and retransmits lost packets and initiates congestion control. This retransmission leads to bandwidth and battery wastage and when a new route is found, TCP throughput continues to be low for some time as it goes to slow start.

Furthermore, the research in [8], [21] & [4] has found that TCP throughput degrades rapidly with increasing path length as shown in the figure 1.1 [4]. Let P_L be the probability of link break and P_B be the probability of path break for path length k can be obtained as

$$P_B = 1-(1-P_L)^k$$

The variation of P_b with path length for P_l=0.1 is shown in figure 1.2



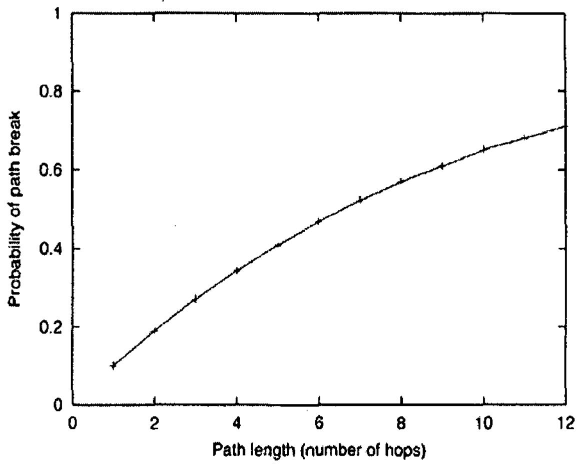


Figure1.1 Variation of TCP throughput with path length **Figure 1.2** Variation of P_b with path length(p_l =0.1)

1.4 Motivation

Thus in MANET packet loss due to broken routes can result in counterproductive invocation of TCP's congestion control mechanisms. Although a number of studies have been conducted and protocol modification suggested for improving TCP performance in MANET. But every improvement has some drawbacks and is suitable to be used only for specific environment. Additionally some protocols have been designed which are specifically tailored towards ad hoc networks. One of these protocols is ATP i.e. Ad hoc Transport Protocol. This protocol also has some limitations which are due to assumptions used in its designing. So our aim is to overcome some of the limitations ATP so that it can to improve its performance.

1.5 Problem Statement

It is inferred that majority of components in TCP are not suitable for ad hoc networks so the new Non TCP protocol ATP(Ad Hoc Transport Protocol) is proposed in [18] that considers simple assumptions links are loss-less and epoch time is fixed. But in general wireless link are lossy and bandwidth has impact on epoch time since data rate depends on bandwidth.

1.6 Objectives

Therefore, in the present work we focus on investigating and evaluating the performance of ATP for the following two conditions:

- Lossy Wireless Links.
- Adjusting Epoch Timer according to the different wireless bandwidths.

1.7 Organization of dissertation

Chapter 1 gives a brief introduction about ad hoc networks, the basic principles of TCP are described and need to develop another protocol at transport layer of ad hoc networks is discussed. This is followed by the problem statement and objectives to be met in this work. Various improvements of TCP are suggested and studied. Major Advantages and disadvantages of each one of them are pointed out in chapter 2

Chapter 3 illustrates the fundamentals of Ad Hoc Transport Protocol (ATP), and discusses the issues to be addressed in it and modifications needed in its algorithm to meet the stated objective.

Chapter 4 presents the simulator details and implementation details of ad hoc transport protocol in GloMoSim. Simulations and results of experiments are discussed in this chapter.

Chapter 5 Concludes the work presented in the dissertation and the scope of further extension of this work.

Chapter2

CHAPTER 2

Related Work

In the first chapter we discussed the unsuitability of Traditional TCP over Ad Hoc networks, it seems mandatory to describe various solutions that researcher have designed for Ad Hoc networks based on TCP or independent of TCP. However, this chapter presents only solutions based on TCP.

2.1 Classification of various TCP solutions

TCP has four major problems as describe below:

- TCP cannot distinguish between losses caused by route failure and congestion.
- TCP suffers from frequent path breaks.
- Contention on wireless channel
- TCP unfairness

First two problems are found in MANETS whereas contention and unfairness mainly affects SANETS. So here in the text we will only deal with first two problems that severely affect MANET and study the solutions and protocols suggested. For details on solution to problems affecting SANETS refer [1].

So the first classification is based on these problems and figure 2.3 shows the classification.

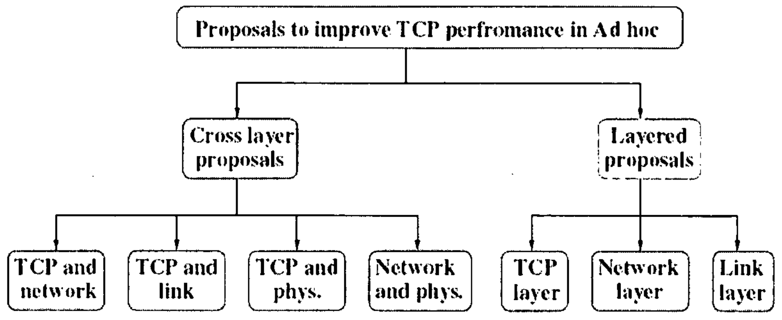


Figure 2.1 Classification 1 for transport layer protocols [1]

Cross layer proposals [1] as the name suggests relies on the interaction between any two layers of the OSI stack. The motivation for this comes from the fact that "providing lower layer information to the upper layer should help the upper layer perform better". And the further classification into TCP and network, TCP and link, TCP and physical and Network and physical depend on which two layers are interacting with each other.

Layered solutions rely on adapting a layer of OSI stack in isolation that is independent of any other layer [1]. Therefore three such layers have been found which on some changes give improved performance in ad hoc networks these are TCP layer, network layer, Link layer.

In the second classification [4] transport layer solutions are divided into two categories namely *TCP over ad hoc networks* and *Non TCP protocols*. TCP over ad hoc network solutions make improvements in existing TCP for achieving a better throughput when used in ad hoc networks where as other transport layer solutions come up with a idea of developing entirely new protocol specific to the needs of ad hoc networks .

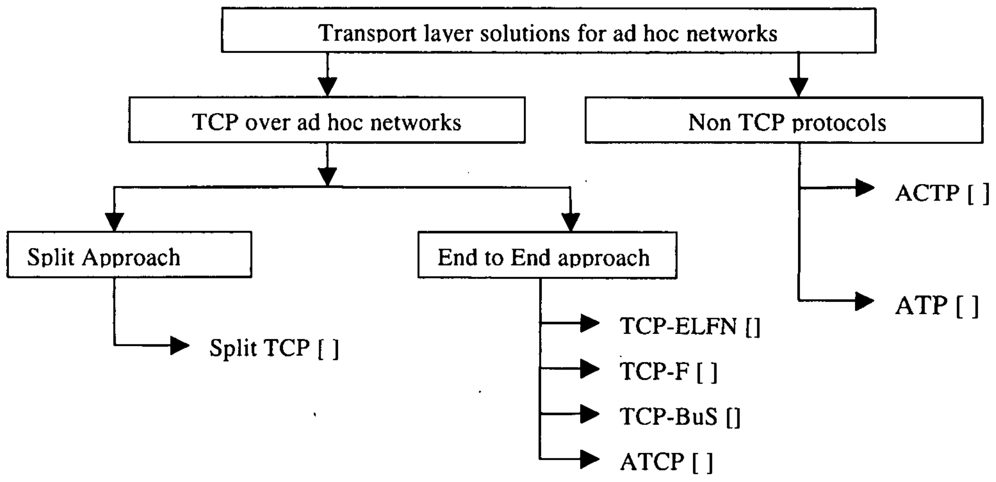


Figure 2.2 Classification 2 for transport layer protocols [4]

2.2 Proposals to improve TCP in respect of its inability to distinguish between losses due to route failure and congestion

These solutions fall in two categories TCP and Network cross layer proposals, and TCP layer proposals [1].

- **TCP and Network Cross Layer Proposals**

TCP-F [17]

It is TCP with feedback, to handle the route failure in ad hoc networks. In this when a routing agent of the node detects the route failure; it explicitly sends a Route Failure Notification (RFN) to source. Source on receiving RFN goes to the *snooze state*, the TCP sender stops sending packets and will freeze all its variables such as timers and congestion window size. TCP sender remains in snooze state until it gets a Route Re-establishment Notification (RRN), on receiving RRN TCP sender resumes transmission based on previous sender window and timeout values. To avoid blocking scenario in the snooze state, TCP sender upon receiving the RFN, triggers a route failure timer. When this timer expires the congestion control algorithm is invoked normally.

Advantages [4]: it provides a simple feedback based mechanism to minimize the problem of frequent route failure in ad hoc networks. Additionally it permits TCP congestion control mechanism to respond to congestion in the network.

Disadvantages [4]: TCP-F uses the congestion window of the old broken route for its transmissions on the new route, therefore may not reflect the achievable transmission rate acceptable to the network and the TCP-F receiver.

TCP-ELFN [8]

TCP-ELFN is based on explicit link failure notification technique as TCP-F but this is based on a real interaction between TCP and the routing protocol. This interaction aims to inform the TCP agent about the route failure when they occur. It uses the ELFN message that is piggybacked onto the route failure message sent by the routing protocol to the sender. The ELFN message is like a "host unreachable" ICMP message, which contains the sender receiver addresses and ports, as well as the TCP packets sequence number. Source on receiving the ELFN enters "*stand by*" state where it disables all the retransmission timers. While TCP sender is in stand by mode it probes the network to check if the route is restored. If the acknowledgement of the probe packet is received, TCP sender leaves the stand by state resumes its retransmission timers and continues the normal operations.

Advantages [4]: TCP- ELFN improves the TCP performance by decoupling the path break information from the congestion information by the use of ELFN.

Disadvantages [4]: In the case of network partition, the duration of path failure may be long and this result in wastage of bandwidth and power due to the generation of periodic probe packet. Moreover TCP-ELFN uses the congestion window of the old broken route for its transmissions on the new route, therefore may not reflect the achievable transmission rate acceptable to the network and the TCP receiver

ATCP [13]

Ad hoc TCP also utilizes network layer feedback, in addition to dealing with the problem of route failures it also deals with the problem of high Bit Error Rate (BER) in wireless networks. The TCP sender can be put into persist state, congestion control state or retransmit state. A layer called ATCP is inserted between the TCP and IP layer of TCP source nodes. ATCP listens to network state information provided by ECN (Explicit Congestion Notification) and by ICMP "Destination unreachable" messages; then ATCP puts TCP agent into the appropriate state. On receiving a "destination unreachable" message, the TCP enters persist state. During this state the TCP agent is frozen and no packets are send until a new route is found by probing the network. The ECN is used as a mechanism to explicitly notify the sender about the network congestion along the route being used. Upon reception of ECN, TCP congestion control is invoked normally. To detect the packet losses due to channel errors, ATCP monitors the received ACKs. When ATCP sees three duplicate ACKs have been received, it does not forward the third duplicate ACK but puts TCP in persist state and quicky retransmits the lost packet from TCP's buffer. After receiving the next ACK, ATCP will resume TCP to normal state. ATCP allows interoperability with TCP sources or destinations that do not implement ATCP.

Advantages [4]: it maintains end to end semantics of TCP and is compatible with traditional TCP therefore allow to seamlessly work with internet while increasing the throughput in ad hoc networks

Disadvantages [4]: it needs information from network protocol about network partition and route changes, which all routing protocols cannot implement. Beside this addition of an ATCP layer in TCP/IP protocol stack requires changes to interface function currently in use.

TCP-BuS [6]

It also uses the network layer feedback to detect the route failure events and take appropriate action in response. In this protocol the main advancement is the introduction of Buffering Capability in the mobile nodes. This protocol specifically selects the "Associativity based routing protocol (ABR)" [5]. Two control messages are used to inform source about the route failure. These are ERDN i.e. Explicit Route Disconnection Notification and ERSN i.e. Explicit Route Successful Notification. The node that detects the route failure is known as pivoting Node (PN), PN sends the ERDN to source and source on receiving it stops sending. Likewise after route re-establishment by the PN using Localized Query (LQ), PN will transmit ERSN to source. On receiving ERSN source resumes its data transmission. During route reconstruction (RRC) phase, packets along the path from source to PN are buffered and to avoid time out event in this phase, retransmission timer for buffered packets is doubled. As the retransmission timer is doubled, lost packets along the path from source to PN are not retransmitted until adjusted retransmission timer expires. To overcome this indication is made to source so that it can retransmit these lost packets selectively. Now when route is restored the destination notifies source about lost packets along the path from PN to destination. On receiving this notification, source retransmits these lost packets. However packets buffered along the path from source to PN may arrive at destination earlier than the retransmitted packets. In this case a destination sends duplicate ACKs. These unnecessary request packets for fast retransmission are avoided. Moreover the delivery of control messages ERDN and ERSN is reliable.

Advantages [4]: This protocol outperforms TCP-F and TCP-ELFN

Disadvantages [4] and [1]: The functioning of protocol increasingly depends on underlying routing protocol and it needs buffering capability at intermediate nodes. Protocol did not take into account that the PN may fail to establish a new partial route to destination.

- **TCP Layer proposals**

Fixed RTO [1]

This is a sender-based technique that relies on the author's heuristics to distinguish between route failures and congestion. When two timeouts expire in sequence, which corresponds to the situation in which the missing ACK is not received before he second RTO expires, the sender concludes that a route failure. The unacknowledged packet is retransmitted but RTO is not doubled a second time. The RTO is fixed until the route is re-established and retransmitted packet is acknowledged.

Advantages: it reports significant enhancement in performance when used with on demand routing protocols.

Disadvantages: it is restricted to work only in wireless networks.

TCP-DOOR [7] & [1]

This is TCP Detection of Out-Of – Order and Response. It is an end – to –end approach that does not require cooperation from of intermediate nodes and is based on out-of-order (OOO) delivery events. OOO events happen when a packet sent earlier arrives later than a subsequent packet, this is due to the fact that packet later in the flow may have changed the route and this new route might be faster than the earlier route followed by packet sent earlier to it. Therefore OOO signifies route change or failure of the past route. OOO can be detected by sender based or receiver based mechanism. The sender based mechanism uses non-decreasing property of ACK sequence number to detect the OOO event. In the case of duplicate ACK packets, which bear the same sequence number, has one byte option added ACK Duplication Sequence Number (ADSN). The ADSN is incremented and transmitted with each duplicate ACK. However receiver based mechanism use two byte TCP option called TCP Packet Sequence Number (TPSN), to detect OOO events. The TPSN is incremented and transmitted with each TCP packet, including retransmitted packets. If the receiver detects a OOO event it should notify sender by setting a specific option bit, called OOO bit, in the ACK packet header. Once TCP sender knows about a OOO event has happened, it takes two actions: it temporarily disables congestion control algorithm for specific time period (T_1), if the congestion control algorithm was invoked during past time period (T_2), TCP sender recover immediately to state before congestion control. T_1 and T_2 are function of RTT.

Advantages [1]: TCP-DOOR improves TCP performance by 50% and does not require any feedback from network layer nor do require cooperation from intermediate nodes

Disadvantages [1]: The supposition OOO event exclusively result in route failure requires much more analysis and Multipath routing protocol such as TORA [] may produce OOO events that are not related to route failures.

2.3 Proposals to reduce route failures

These solutions fall in three categories: TCP and network cross layer proposals; network and physical cross layer proposals; and network layer proposals.

- **TCP and Network Cross Layer Proposals**

Split TCP [26]

The known fact about TCP connections is that as the number of hops in a connection increase then the probability of route failure increase so to resolve this problem Split TCP splits long TCP connection into short localized segments. The

interfacing node between two localized segments is called a proxy. Proxy buffers TCP packets and acknowledges their receipt to source (or previous proxy) by sending local acknowledgements (LACK). Proxy is also responsible for delivering packets at an appropriate rate to next local segment. Upon receipt of LACK (from next proxy or final destination), a proxy will purge the packet from its buffer. To ensure source to destination reliability, an ACK is sent from destination to source.
Advantages [4] and [1]: it provides improved throughput i.e. up to 30%, improved throughput fairness with reduction in the effect of mobility.
Disadvantages [4]: it needs large buffers, network overhead and complex behavior of the proxy nodes. Moreover, in case of proxy node failure it can lead to reduction in throughput.

- **Network and physical Cross Layer Proposals**

Preemptive Routing in Ad Hoc networks [28]
This proposal addresses the problem of frequent route failures and help in reducing route reconstruction latency. The technique is coupled with the on – demand routing protocols AODV and DSR. The mechanism to detect failure is power based i.e when the intermediate node detects that the signal power of the received packet from upstream nodes drops below a given threshold, called preemptive threshold, it detects route failure. On detecting this event intermediate node will notify the source. In response source will proactively looks up a new route. When a new route is available, the routing agent switches to this new route. The value of this preemptive threshold is critical, so repeated short message probing is used to verify the correctness of route failure message.
Advantages [1]: This scheme results in reduction of route failures and decreases latency by 30%.
Disadvantages [1]: This scheme is "packet receipt event driven" and that failures cannot be detected if no packets are transmitted.

- **Network Layer Proposals**

Back up path routing [9]

In this several paths from source to destination are maintained, but only one is used at a time. When current path breaks, it can quickly switch to an alternate path. For every connection one primary path and one alternate path are kept. In first scheme shortest-hop path is a primary path and shortest delay path is an alternate path where as in second scheme shortest delay path is primary path and maximally disjoint path is an alternate path. First scheme outperforms second scheme.

Advantages [1]: it reports improvement in TCP throughput of up to 30% with a reduction in routing overheads.

Disadvantages [1]: Further evaluation is needed, for route selection criteria.

2.4 Non TCP protocol

ACTP [14]

It is a light weight Application controlled transport protocol that is not an extension of TCP. The key fact in its designing is that provisioning of reliability is left with the application layer and provides simple feedback information about the delivery status of packets to application layer. ACTP support priority of packets to be delivered, but it is the responsibility of lower layers to actually provide a differentiated service based on priority. It is implemented as a layer between Application and network layer and application layer uses API functions to interact with ACTP Layer.

Advantages [4]: it is scalable for large networks and gives the freedom to choose required reliability level to application layer. Throughput is not affected by path failures.

Disadvantages [4]: ACTP is not compatible with TCP. Moreover when it is used in large ad hoc networks it can lead to heavy congestion, as it does not have any congestion control mechanism.

TH-14558

Chapter 3

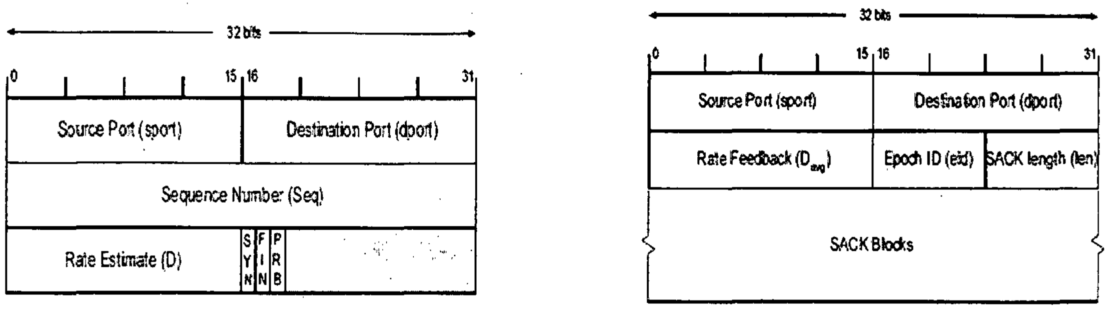
CHAPTER 3

ATP Details and Algorithm Modification

ATP is Ad Hoc Transport Protocol proposed by authors of [18]. It is specifically designed for ad hoc networks and is not a variant of TCP. It relies on coordination among layers, rate based transmissions, decoupled congestion control and reliability.

3.1 ATP Design

The header format of ATP is shown below in figure 3.1



(a) Data packet header

(b) Feedback packet header

Figure 3.1 ATP Header format [18]

The **Data Packet Header** consists of following fields:

Sport: source identifier to uniquely identify the source of the connection

Dport: destination identifier to uniquely identify the destination of the connection

Seq: it is the sequence number at transport connection level.

D: is the rate estimate field (delay in ms)

Syn: flag for connection initiation.

Fin: flag for connection termination

Prb: probe flag

The **Feedback Packet Header** consists of following fields:

Sport: source identifier to uniquely identify the source of the connection

Dport: destination identifier to uniquely identify the destination of the connection

D_{avg}: is the rate feedback field (ms)

eid: Epoch identifier

len: indicate number of SACK blocks contained in feedback packet

SACK blocks: selective ACK blocks send by receiver for reliability

The key design elements of ATP are

- **Layered coordination**

ATP use lower layer information and explicit feedback information from network nodes to assist transport layer mechanism. It relies on feedback from both MAC and routing layer.

Functioning of MAC at intermediate node

The MAC layer at each intermediate node calculates queuing delay (Q_{sample}) and transmission delay (T_{sample}). Queuing delay is the time spent by a packet right from when it is inserted into queue till it reaches the head of the queue to be dequeued by the MAC Layer and Transmission delay is measured as time spent by the packet in the MAC layer right from when it is dequeued by the MAC layer till it gets transmitted successfully on the channel. These values of Q_{sample} and T_{sample} are processed by intermediate node; to provide rate feedback information (D) i.e the field shown in data packet header in fig 3.1. Each intermediate node maintain a sum of Q_t and T_t. The values are maintained using exponentially averaging as follows.

$$Q_t = \alpha * Q_t + (1 - \alpha) * Q_{sample}$$
$$T_t = \alpha * T_t + (1 - \alpha) * T_{sample}$$

The value of 'α' used in simulation is 0.85. When the packet is dequeued for transmission, the intermediate node checks that if the value of 'D' field is smaller than the $Q_t + T_t$ value at that node then it updates the value of 'D' to its $Q_t + T_t$ value. Therefore the rate feedback field 'D' is the maximum of $Q_t + T_t$ value at the upstream nodes that the packet has traversed. There is a slight change in the evaluation of 'D' during connection start up when probe packet is send by setting ' prb' field to one in data packet header. This is because there is no traffic present around the intermediate node during connection start up, therefore node views a idle channel, and it uses η*($Q_t + T_t$) as delay instead of $Q_t + T_t$. The use of η* (Qt + Tt) is justified at connection start up because when the channel is idle the (Qt + T_t) value is determined by the actual queuing and transmission delay experienced by the probe packet. However when the real data flow begins packets belonging to the flow will contend with packets belonging to the same flow at both upstream and downstream nodes. Thus intermediate node projects the real behavior of transmission by appropriately setting the value of η. For CSMA/CA the typical value of η is 3.

Functioning of MAC layer at receiver

Receiver act as a collator of the rate feedback information received from the sender in the form of 'D' field in the data packet header. For this it runs an *epoch period* 'E' to periodically send the exponentially averaged rate feedback information (D_{avg}) with epoch id (eid) in the feedback packet header. Thus epoch period determines rate at which feedback is send to the sender. The exponentially averaged rate feedback is calculated as:

$$D_{avg}= \beta * D_{avg} + (1-\beta) * D$$

Moreover receiver is also responsible for the **flow control**. ATP monitors the rate R_{app} at which the application process the in sequence data from the receive buffer. When receiver sends the feedback rate to sender, if application read rate (R_{app}) is smaller than the rate feedback, the rate feedback information (D_{avg}) is replaced with the $R_{app.}$ Thus receiver sends both rate feedback and reliability information in periodic feedback message (once every epoch)

Functioning of routing layer

When a intermediate node experiences link failure (as indicated by MAC layer), routing layer sends a path failure notification message to the source of the packet. The routing layer in response to this informs ATP of the path failure. ATP then switches to rate estimation mode, where it probes for the available bandwidth along the mew route to the destination. When the routing layer establish a new route, ATP sender automatically obtains rate feedback from receiver indicating available bandwidth along the new route and hence switch to rate adaptation mode.

- **Rate based Transmissions**

ATP use Rate based transmissions instead of window based transmissions as in TCP. Therefore the need of self-clocking is eliminated; this avoids the drawbacks due to burstiness and helps in decoupling of congestion control and reliability. As TCP use slow- start mechanism during connection initiation or when recovering from a timeout, this takes a few round- trip times before it converge on the available bandwidth for a flow. Therefore this slow- start mechanism is not a good choice for ad hoc networks as they are prone to path failures and hence spend considerable portion of lifetime in the slow- start phase and, hence degrades network utilization. ATP sender uses a mechanism known as **Quick Start** to probe to the available network bandwidth in one round trip time. Figure 3.2 shows the pseudo-code for quick start algorithm.

ATP use TCP like SYN-SYN+ACK exchange between sender and receiver. The intermediate nodes stamp Delay information (Q_t+T_t), while forwarding SYN packet. Receiver in response sends the feedback packet piggybacked with the Q_t+T_t value as seen on incoming SYN packet. Sender, upon receiving the feedback packet, starts using the rate value obtained based on the feedback. ATP performs Quick-Start both during connection initiation and when it recovers from path failure. The probe packets are send, identified by setting 'prb' field in the packet. Thus performing bandwidth estimation after path failure using quick-start allows a connection to operate at its true bandwidth and does not underutilize or over utilize the network resource along the new path

Initial Rate Estimation:

```
Sender
1    send probe packet

Intermediate node
2    Compute $Q_t + T_t$ for packet
3    if($Avg(Q_t) + Avg(T_t) > \epsilon$)
4        $Avg(Q_t) = \alpha * Avg(Q_t) + (1 - \alpha) * Current(Q_t)$
5        $Avg(T_t) = \alpha * Avg(T_t) + (1 - \alpha) * Current(T_t)$
6        if ($Avg(Q_t) + Avg(T_t) > stamped\ D$)
7        $stamped\ D = Avg(Q_t) + Avg(T_t)$
8    else
9        $D_{projected} = i * (Current(Q_t) + Current(T_t))$
10       if($D_{projected} > stamped\ D$)
11           $stamped\ D = D_{projected}$

Receiver
12   Set $Avg(D) = Current(D)$
13   send $packet_{feedback}$ to sender with $Avg(D)$

Sender
14   $packet_{feedback}$ received with $Avg(D)$
15   compute rate $R = \frac{1}{Avg(D)}$
16   $send\_rate\ S = R$
17   $start\_epoch\_timer()$
18   $send\_packet()$
```

Figure 3.2 Pseudo- code for quick-start [18]

Congestion control in TCP consist of two phases linearly increasing phase and multiplicatively decreasing phase, while ATP's congestion control consist of three phases increase phase decrease phase and maintain phase. TCP does not use the feedback from the intermediate nodes and hence does not know the true extent of congestion therefore it conservatively performs multiplicative decrease in congestion window size on the other hand ATP relies of feedback from the intermediate nodes therefore its decrease phase can be less conservative than that of TCP, and it operates in maintain phase when network conditions do not change.

Increase phase: When the feedback rate from the receiver is greater than the current rate S by a threshold Φ * S, the sender enters the increase phase (lines 12-13 in Figure 3.3), where Φ is a small constant used to prevent fluctuations (empirically set to 0.1). ATP sender performs a linear increase of rate similar to TCP.

Decrease phase: When the feedback rate is smaller than the current rate sender performs decrease phase (lines 14-15 of Figure 3.3), by adjusting current rate to feedback rate i.e. it proportionately decrease the rate. Thus ATP uses LIPD (Linear Increase Proportional Decrease). LIPD paradigm is studied for its fairness and efficiency [17], and obtains steady state throughput that is proportional to$1/p$ instead$1/\sqrt{p}$in LIMD ('p' is packet loss probability). Thus ATI obtains better utilization and convergence properties by using feedback from intermediate node.

Maintain phase: If the available rate R lies between (S, S+Φ*S), the sender maintains its rate. This is a state of equilibrium for ATP when network conditions remain stable.

Moreover it is inferred from the above discussion that the sender can only take the decision for congestion control, if it receives feedback from receiver, but there is always a possibility of the reverse path failure, due to which the feedback from receiver can be lost. In this condition ATP respond by performing multiplicative decrease of sending rate for at most two epochs in which it does not receive the feedback from the receiver. At the end of third epoch, again if it does not receive the feedback from the receiver, it goes to connection initiation phase, sending one probe every epoch till it hears back from the sender.

Normal Operation:

```
Intermediate node
1     Compute $Q_t + T_t$ for packet
2     if($Avg(Q_t) + Avg(T_t) > \epsilon$)
3         $Avg(Q_t) = \alpha * Avg(Q_t) + (1-\alpha) * Current(Q_t)$
4         $Avg(T_t) = \alpha * Avg(T_t) + (1-\alpha) * Current(T_t)$
5         if ($Avg(Q_t) + Avg(T_t) > stamped\ D$)
6             $stamped\ D = Avg(Q_t) + Avg(T_t)$

Receiver
7         $Avg(D) = \beta * Avg(D) + (1-\beta) * Current(D)$
      On epoch_timer expiry
8         stamp $Avg(D)$ on $packet_{feedback}$
9         send $packet_{feedback}$ to sender

Sender
10    $packet_{feedback}$ received with $Avg(D)$
11    Compute new rate $R = \frac{1}{Avg(D)}$
      Rate Adjustment:
12    if send_rate $S < R - \phi * S$
13        $S = S + 1$
14    else if $S > R$
15        $S = R$
16        else maintainS
17    start_epoch_timer()
18    send_packet()
```

Figure 3.3 Pseudo-code for normal operation [18]

- **Decoupling of congestion control and reliability**

ATP use collated rate feedback provided by the receiver for congestion control and selective ACKS to report back the sender any new holes in the data stream. Moreover receiver maintains the epoch period for periodically sending the feedback to the sender therefore reliability feedback is not provided for every incoming data packet but on periodic basis therefore ATP maintains a larger number of SACK block as compared to TCP-SACK. ATP uses a maximum of 20 SACK blocks and the 'len' field in the feedback packet header represents the actual number of SACK blocks sent by the receiver. The pseudo code for reliability operation is shown in figure 3.3

Reliability Operation:

```
Receiver
1   On epoch_timer expiry or receipt of probe pkt
2       identify all holes encountered
3       stamp number of SACK blocks (len) on $packet_{feedback}$
4       stamp SACK blocks starting from first hole on $packet_{feedback}$
5       send $packet_{feedback}$ to sender

Sender
6   On $packet_{feedback}$ receipt with SACK information
7       update scoreboard and identify pkts for retransmission
8       On send_timer expiry
9           pkts marked for retransmission are sent preferentially
```

Figure 3.4 Pseudo-code for reliability operation [18]

3.2 ATP Advantages and Disadvantages

The core advantages [4] of ATP are improved performance, decoupling of congestion control and reliability mechanisms, and avoidance of congestion window fluctuations as congestion information is gathered directly form nodes that experience it.

The chief disadvantage [4] of ATP is lack of interoperability with TCP and for large ad hoc network; the fine-grained per-flow timer used at the ATP sender may become the scalability bottleneck in resource constrained mobile nodes.

3.3 Issues uncovered

- The design of ATP has not considered very lossy links. It is possible in wireless environments that the bit error rate on the link can result in packet drop rates as high as 10%. The corresponding impact on ATP is that estimation of Q_t and T_t parameters would be affected and this would in turn affect rate adaptation.

- The epoch timer used is fixed, but it needs to be adapted as network wide parameter with respect to different wireless bandwidths.

3.4 ATP Algorithm Modification

To resolve the issues uncovered we will make slight modification in the pseudo-code for quick start and normal operation shown in figure 3.2 and 3.3 respectively. The function ***start_epoch_timer*** called in the algorithms is a fixed timer of 1 second used in all the simulations of ATP. We will modify this timer with respect to the parameter '***bandwidth***' of the physical link and then explore the results.

Chapter 4

CHAPTER 4

Implementation details and simulation results

4.1 Simulator used

Simulator is the tool used to analyze a network protocol using a computer tailored numerical solution. This tool helps us to evaluate the performance of a given system. It takes a set of assumptions to model a system. So the very first thing while modeling a protocol or a system is to have a goal and decide the various assumptions and relationship between them and that determine mathematical model. Thus simulation is essential tool in the sense that it can often help to improve or validate protocols. All simulators provide a complete toolkit to developers that enable metrics collection and various wireless network diagnostics. Moreover in MANET there is no coordination or configuration before the set up of network, due to which it suffers from several challenges. The leading solution to these challenges is simulation. There are various simulators available for MANET for e.g. ns 2, DIANEmu, GloMoSim, GTNet, J Sim, OMNET++, OPNET, QualNet, SWANS. We will use GloMoSim for implementing our transport layer protocol ATP.

4.2 GloMoSim

GloMoSim is Global Mobile System Simulator. It is library based sequential and parallel simulator for wireless networks [29]. The library of GloMoSim is developed using PARSEC (Parallel Simulation Environment for Complex Systems) a C based Parallel simulation language. GloMoSim has been designed extensible and composable, as it is built using layered approach similar to OSI layered architecture. The GloMoSim network architecture can be seen below Figure 4.1:

LAYER	**PROTOCOLS**
Application	CBR, FTP ,HTTP, TELNET
Transport	TCP and UDP
Network	IP with AODV, Bellman-ford, DSR,Fisheye, LAR Scheme 1, ODMRP, WRP
Data link(MAC)	CSMA, IEEE 802.11 and MACA
Packet reception Models	SNR Bounded, BER based with BPSK/QPSK Modulation
Radio Model	Noise Accumulating
Radio Propagation Model	Two Ray and Free space
Mobility	Random way point, random drunken, Trace based

Figure 4.1 GloMoSim Architecture

Models of protocol at one layer interact with those at lower are higher layer via API's. Due to the modular design of GloMoSim it is capable of being scaled up to thousands of heterogeneous network nodes, with a reasonable simulation time; this is done using the concept of node aggregation. As each entity is required to examine packets received only from the nodes located in the region it is simulating, many partitions are used to reduce total search space. If a packet sent by a node located in Partition (0, 0) cannot reach the border of the partition, no message needs to be sent to the other partitions. Therefore, the other partitions do not have to examine the reception of the packet.

For this project GloMoSim is installed on Windows XP. The basic structure of GloMosim [11] is as follows:
/doc – contains the documentation
/scenarios – contains directories of various sample configuration topologies
/main – contains the basic framework for GloMosim
/bin – contains the executables and the input/output files
/include – contains common include files
/application – contains code for the application layer i.e. files for traffic generation
/transport – contains the code for the transport layer
/network – contains the code for the network layer
/mac – contains the code for the Mac layer, including 802.11b
/radio – contains the code for the physical layer

All of these files need to be compiled from the /main directory. To do this in Windows, first set the Visual Basic Environment and then run the batch file "makent" to create the executable "glomosim.exe".

To perform simulations in GloMosim, the input file needs to be configured. Within the /bin directory the basic input file "**config.in**" can be found. This file contains the general simulation parameters for GloMosim. It is structured according to each layer and allows a protocol to be chosen for each layer. In the file, anything following"#," is treated as a comment. Therefore in order to select a protocol, the "#" is deleted. Figure 4.2 is a snapshot of the config.in file, nodes.input file& app.conf file

```
[config.in]

NUMBER-OF-NODES

NODE-PLACEMENT

NODE-PLACEMENT FILE  /nodes.input

#NODE-PLACEMENT    uniform

MAC-PROTOCOL              IP

ROUTING PROTOCOL   BELLMAN-FORD

APP-CONFIG FILE       ./app.conf

RADIO-TYPE       RADIO-ACCONOISE
```

```
[nodes.input]
0 0 (20.2,0.9,0.11)
1 0 (80.4,90.8,0.17)
2 0 (60.7,30.4,0.10)
```

```
[app. conf]
#CBR<src_node>,<dest_node>
#<items><items_size><Interval_
time
#,start_time.><end_time>
CBR 0 1 10 512 1S 0S 0S
```

Figure 4.2 Snapshot of Config.in file

To run the input file, from the /bin directory, type "GloMosim config.in". The simulation will run and produce an output file "**GLOMO.stat**". This file contains the results for each layer including the throughput, average end-to-end delay, hop distance, data packets transmitted and received etc.

From the above file the simulation will have 3 nodes, positioned according to the file "nodes.input" above. The structure of "nodes.input" is <node address>, <0>, <(x, y,z)>. The MAC protocol chosen is 802.11, the Network protocol is IP and the application chosen for this particular simulation is Constant Bit-Rate (CBR) from the "app.conf" fileabove. The structure of "app.conf" is <source node address>, <destination nodeaddress>, <no. of items to send>, <item size>, <interval time>, <start time>, <endtime>. A simulation time is also added at the beginning of the file along with terrain dimensions.

The **visualization tool** [16] is not dependent on the platform as it is coded in Java. Therefore in order to run the tool, Java the Java Development Kit (JDK) version 1.2 must be installed. Once JDK is installed the Java VT files must be compiled. From the /java_gui directory, type "javac *. java". To start the VT type "java

GlomoMain&". The visualization tool was not used in the simulations, as it was quite unstable and unreliable.

4.3 Embedding New Protocol ATP in GloMoSim

We have implemented ATP and added it to the transport layer of GloMoSim. To implement the functionality of ATP at transport layer we have made modifications in existing TCP in GloMoSim to realize the functioning ATP.

Structure of GloMoSim code

Each directory of the GloMoSim library has three important files, which must be altered or created to change or add new protocols. These are as follows:

- **Header files (.h)** – Basic definitions of functions and data objects
- **.pc files** – Actual functions
- **Layer.pc** – Layer Interface

Different steps have to be considered when adding protocol to a layer in order to maintain consistency and proper functionality in GloMoSim.

Three main functions have to be elaborated to add a new model:

1. **Initialization Function**: It must allocate and initialize the model specific data.
2. **Finalization Function**: It generates the output statistics from the simulation run for this model.
3. **Simulation Event Handling Function**: It performs actions when simulator triggers a scheduled with an event.

Scheduling of events takes place as follows:

- Creates the GloMoSim Message:

Message* msg = GLOMO_MsgAlloc(node, MyID, MyLayer, MyProtocol, MSG_LAYER_PROTO_MY_EVENT);

- Set the Event Specific Information in the message:

MyEventInfoType* MyEventInfo;

GLOMO_MsgInfoAlloc(node, msg, sizeof(MyEventInfoType));

MyEventInfo-> MyFirstParameter =1;

• Schedule the Event in the message:

GLOMO_MsgSend(node, message, MyChosenDelay);

The three functions required for adding a new protocol are described below:

Initialization function for ATP

Initialization of ATP is controlled by initialization function of transmission control protocol i.e.

```
TransportTcpInit (GlomoNode *node, const GlomoNodeInput
*nodeInput)
```

PURPOSE Initialization function for ATP.

PARAMETERS:

node: node being initialized.

nodeInput: structure containing contents of input file

Simulation Event Handling function for ATP

```
void TransportTcpLayer(GlomoNode *node, Message *msg)
```

PURPOSE Models the behaviour of ATP on receiving the message encapsulated in msg

PARAMETERS:

node: node, which received the message

msg: message received by the layer

Finalization function for ATP

```
void TransportTcpFinalize(GlomoNode *node)
```

PURPOSE: Called at the end of simulation to collect the results of the simulation of the ATP protocol of Transport Layer.

PARAMETERS:

node: node for which results are to be collected.

The files which are being modified to create ATP and used for collection of information for ATP layered coordination are given below:

Message.h

This file is found in include directory in GloMoSim folder. In file a message a structure, *message_str* is defined. It is further type defined as *message* in *main .h* file. In this structure we will add our own variables as given below:

`double TimeQueueInsert`

This variable is used for calculation of time packet that is inserted in the queue.

`double TotalTimeInQueue`

This variable is used for calculation of queuing delay.

api.pc

In this file the structure for **glomonode** is defined. We have added some variables in this structure. These are as follows:

`float AvgQueuingDelay`

PURPOSE: this variable is used by intermediate node to hold the value for average queuing delay. This value is used in calculation of delay at intermediate nodes.

`float AvgTxDelay`

PURPOSE: this variable is used by intermediate nodes to hold the value for average transmission delay. This value is used in calculation of rate estimation at intermediate node.

nwip.pc

In this file the value of variable ***TimeQueueInsert,*** declared in structure message of message.h file is calculated inside the following function:

`VoidQueueUpIpFragmentForMacLayer(GlomoNode*,node, Message*msg,InterfaceIdType initInterfaceId,NODE_ADDR nextHop)`

PURPOSE: Sends an IP packet to the Mac layer specifying the next node address "nextHop" for packet to go.

This function is found in file *nwip.pc* where the value of variable ***TimeQueueInsert*** is calculated. This gives the time when packet is inserted in the queue. We have defined a function **intermediate_node_calculation** in this file. It is called to compute the delay for every packet traversing node on the path between sender and receiver and modifying the rate estimate(D) found in the header of the incoming packet according to the algorithm in fig 3.2 and fig 3.3 .

`Void NetworkIpReceivePacketFromMacLayer( GlomoNode* node, Message* msg, NODE_ADDR lastHopAddress)`

found in nwip.pc. It uses the queuing delay calculated in variable ***TotalTimeInQueue***. The value returned by this function is set in the RateEstimate field of tcp_hdr.

Mac 802.11.pc

The value of variable ***TotalTimeInQueue*** is calculated in the function.

`void Compute_Queuing_Delay(Message *msg)`

This function is defined in MAC_802.11.pc, it takes the current simulation time using the function simclock() and TimeQueueInsert variable defined earlier. It gives the time spent by the packet in the queue before it gets dequeued by the MAC layer for transmission on the channel. This queuing delay is used in function **`intermediate_node_calculation`** in nwip.pc file.

Radio_accnoise

This is the file found in radio directory. In this file there is a variable **`txDuration`** that gives the transmission time of a packet. The value of this variable is used is used in function **`intermediate_node_calculation`**.

Tcp_hdr.h

In this file the fields specific to ATP are declared in the structure tcp_hdr. This structure has both the fields for data packet header and feedback packet header as tcp header use a single header for data as well as ACK. The new fields added are as follows:

`unsigned int prb`

PURPOSE: used for identifying probe packets.

`float RateEstimate`

PURPOSE: It is used by sender to calculate the rate for sending packets according to the feedback rate obtained from the receiver.

`float RateFb`

PURPOSE: used by the receiver to send the averaged delay in the feedback packet.

`BOOL FBK`

PURPOSE: This field is used to distinguish between the ACK packets and ACK packets with piggybacked feedback information.

Tcp_input.pc

This file has a function to calculate rate feedback.

`float receiver_calculation(unsigned int prb, float recv_rate_estimate)`

PURPOSE: This function is used for the exponential averaging of the rate and hence calculates of rate estimate which is send to ATP sender in the feed back packet. It is called in tcp_input.pc file.

PARAMETERS:

`Prb`: If this field is set then Quick start Algorithmic receiver calculation is implemented otherwise Initial Rate receiver calculation is implemented.

`recv_rate_estimate` : This is the delay calculated and updated at each intermediate node according to intermediate_node_calculation_function called at each intermediate node in nwip.pc file.

Tcp_output.pc

`float sender_calculation(float Rate_Feedback)`

PURPOSE: This function is used for calculating rate at which a sender transmits packets in accordance with the feedback rate from the receiver and thus helping in congestion control. It is called in tcp_output.pc. It calls a another function given below:

`float decrese_increase(float OldRateEstimate, float NewRateEstimate)`

This function takes the old rate estimate and rate feedback returned by sender_calculation function to implement congestion control by increasing its rate linearly or decreasing its rate or maintaining its old rate according to ATP algorithm. This rate is used for manipulating the window to maintain the send rate.

`clocktype EpochTimer( GlomoNode* node)`

PURPOSE: Approach is to make epoch time of less duration for large bandwidth so that we can conserve the scarce resource of adhoc networks.This function is used to run an epoch timer that is a function of bandwidth. It is called before transport layer sends packet to network layer in tcp_output.pc.

4.4 Simulation and Results

Simulation environment

We have used GloMoSim as simulation tool to test the modifications proposed in ATP. In our simulation, nodes are spread in a terrain of 1000*1000. We have considered network of size 20, 30, 50 and 70 nodes with each node having equal transmission range and placed randomly within the area. Dynamic source routing (DSR), and IEEE 802.11 are used as routing and MAC protocol, respectively. Results of experiments are obtained by taking the average of 10 simulation runs for different seed values. Mobility is introduced by using random waypoint model. Results of the simulation are expressed using following three metrics:

- Throughput
- Success rate
- Energy consumed

Throughput is a measure of bytes (packets) received per second by a receiver application. We have considered the packets received at FTP client.

Success rate is defined as the ratio of total packet received to the total number of packets sent. In our simulation, it is a ratio of total number of bytes received by a FTP client to the total number of bytes sent by the FTP server.

Energy consumption is expressed in terms of the total energy consumed by all the nodes in transmission as well as receiving the packets.

Results and Analysis

The results obtained in terms of the three metrics throughput, success rate and energy consumed are represented in the form of graphs, compared and justified. Fig. 4.3 shows the results of success rate of ATP and TCP

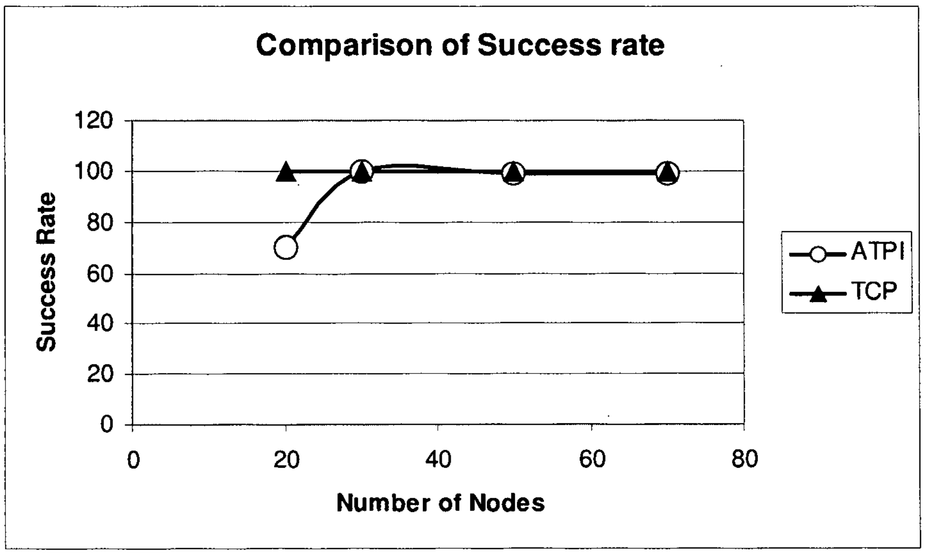


Figure4.3: Success Rate of Network

Success rate of ATP is found to be equal to that of TCP for large number of nodes except for network size of 20 to 30 nodes. The reason behind this behavior is lower connectivity of the network. ATP starts with higher packet transmission rate where as TCP uses slow start mechanism. The lower connectivity results in less number of successful deliveries of packets in the beginning.

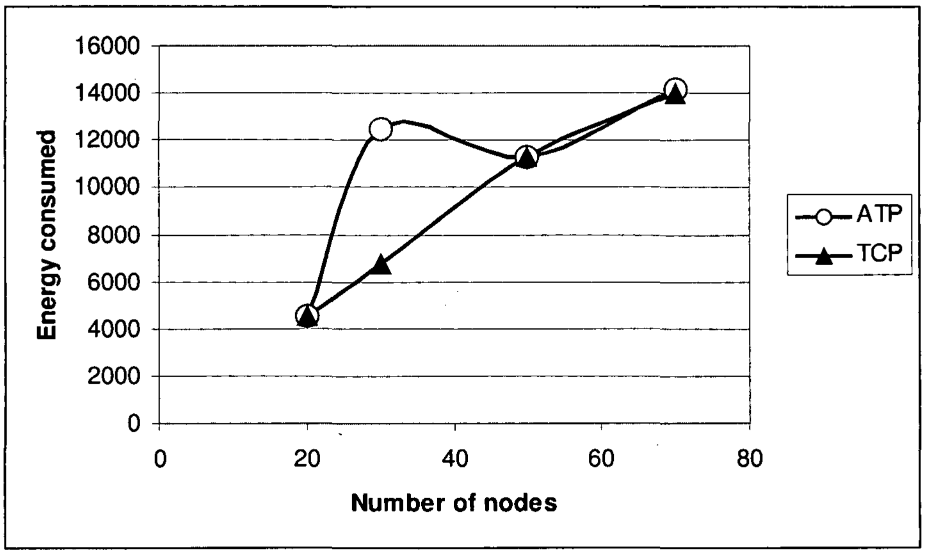


Figure: 4.4 Energy Consumption of the Network

Fig. 4.4 shows energy consumption of the network as computed below:

Total Energy consumption (tp) = $\sum_{i=1}^{n} p_i$

Where p_i is energy consumed by a i_{th} node.

Energy consumption of ATP is less than that of TCP for very large number of nodes. Here again a large number of packets are being transmitted at the start and nodes attempt to forwards them consume energy though could not succeed in delivering the packets for smaller network size.

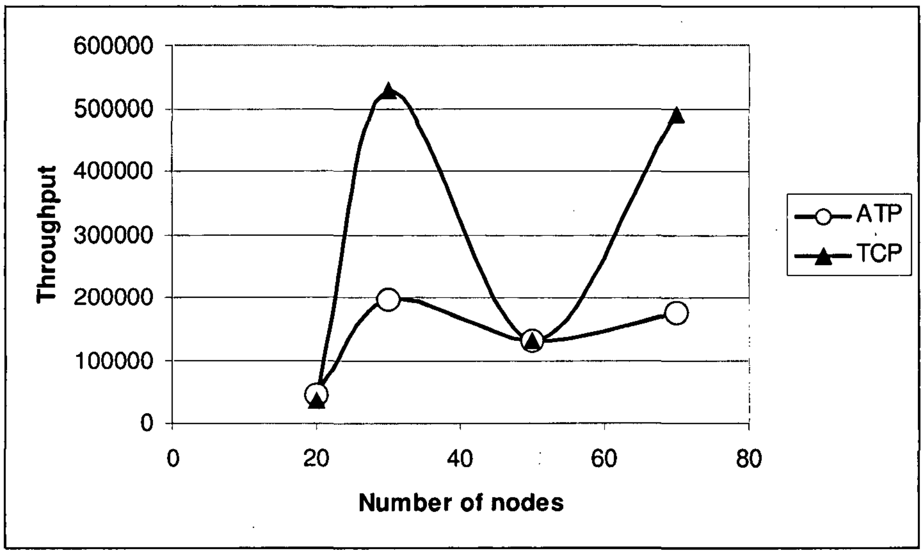


Figure 4.5 Throughput of the Network

The Throughput of TCP is found to be random and unpredictable as the network size grows. It is evident from the figure that for a size of 20 nodes it is low and it becomes very high for the network size of 30. But again for the network size of 50 it falls down. We find it difficult to explain this behavior of TCP. However the throughput of ATP for small size network is low due to partial connectivity of the network. But the throughput becomes steady as the network size grows. Therefore, it is difficult to compare the throughput of the both. However, we assume that if ATP is implemented a fresh rather than realizing it through modifications in TCP, it may provide increased throughput than as exhibited in the current work.

Chapter 5

CHAPTER 5

Conclusion and Discussion

5.1 Conclusion

It is known that TCP is not suitable for ad hoc networks. Therefore, we have attempted to improve and implement ATP designed for ad hoc network. The proposed improvements are implements by making modification to existing TCP. From the results of simulation, we observed that the overall performance of ATP is better in terms of success rate and energy consumption. But the third metric i.e. throughput is not comparable due to unpredictable and unstable behavior of TCP.

5.2 Discussion

In GloMoSim there does not provide user interface at transport layer to embed new protocol unlike other layers. Our initial attempt to implement ATP with proposed modifications from scratch could not succeed. However, this could have been complete have there been sufficient time. Therefore, we decided to modify

TCP to embed the functionality of ATP so that proposed modifications can be tested within the specified period.

5.3 Contribution

The work carried out in this dissertation contributes to the knowledge both in terms of theory and implementation. The contributions of the work are summarized as follows:

- Modifications suggested in ATP for lossy wireless environment.
- Estimation of packet transmission rate based on bandwidth in terms of epoch timer.
- Modification made in TCP of GloMoSim to realize the behavior of ATP.
- Simulation and Analysis of ATP realized through TCP.

5.4 Future work

- Modification of proposed work may consider highly lossy wireless links causing packet drop rate as high as 10%.
- ATP considers delay caused by erroneous packets in computation of average delay at intermediate nodes. This may be true for wireless links with the MAC layer that could not recover from the errors. However, in very lossy wireless links with MAC layer able to recover from the errors, the delay caused by erroneous packets should not account for total average delay experienced by a node.
- Epoch timer is a function of bandwidth. However, increase in network size may result in longer delay for which epoch timer will be smaller then round trip time.'Therefore, epoch timer may be modified to scale up and down in accordance with the network size

References

[1] Ahmad Al Hanbali, Eitan Altman, and Philippe Nain, Inria Sophia Antipolis France "A Survey of TCP over Ad Hoc Networks", *IEEE Communications Surveys & tutorials*, Third Quarter 2005.

[2] C. E. Koksal and H. Balakrishnan, "An Analysis of Short-term Fairness in Wireless Media Access Protocols (poster)," in *Proceedings of ACM SIGMETRICS*, Measurement and Modeling of Computer Systems, Santa Clara, CA, 2000, pp. 118–119.

[3] C.Parsa and J.J Garcia-luna-Aceves, "Improving TCP Performance over WirelessNetworks at the Link layer", *Mobile Networks and Applications*, vol5, no.1, pp.57-71, 2000

[4] C.Siva Ram Murthy and B.S Manoj "Ad hoc wireless Networks, Architecture and protocols" 2^{ed} Pearson education,2005

[5] C. Toh, "Associativity-based routing for ad hoc mobile networks," Journal of Wireless Personal Communications, vol. 4, no. 2, pp. 103–139, Mar. 1997.

[6] D. Kim, C. Toh, and Y. Choi, "TCP-BuS: Improving TCP performance in wireless ad hoc networks," *Journal of Communications and Networks*, vol. 3, no. 2, pp. 175–186, Jun. 2001.

[7] F. Wang and Y. Zhang, "Improving TCP performance over mobile ad hoc networks with out-of-order detection and response," in *Proc. of ACM MOBIHOC*, Lausanne, Switzerland, Jun. 2002, pp. 217–225.

[8] G. Holland and N. Vaidya, "Analysis of TCP Performance over Mobile Ad Hoc Networks," *Proceedings of ACM MOBICOM 1999*, pp. 219-230, August 1999.

[9] H. Lim, K. Xu, and M. Gerla, "TCP performance over multipath routing in mobile ad hoc networks," in *Proc. of IEEE ICC, Anchorage, Alaska, May 2003.*

[10] HALA ELAARAG "Improving TCP Performance over Mobile Networks", *ACM Computing Surveys*,Vol.34, No.3, September 2002, pp357-374.

[11] http://pcl.cs.ucla.edu/projects/glomosim/GloMoSimManual.html

[12] Imrich Chlamtac, Marco Conti, Jennifer J.-N. Liu, "Mobile ad hoc networking: imperatives and challenges", *ELSEVIER Ad Hoc Networks* 1(2003) 13-64.

[13] J. Liu and S. Singh, "ATCP: TCP for mobile ad hoc networks," IEEE JSAC, vol. 19, no. 7, pp. 1300–1315, Jul. 2001.

[14] J.Liu and S.Singh, "ACTP: Application Controlled Transport Protocol for Mobile Ad Hoc Networks," *Proceedings of IEEE WCMC 1999*, vol.3, pp. 1318-1322, September 1999.

[15] Jacobson, V.1988. "Congestion Avoidance and Control". *SIGCOMM Symposium on Communication Architectures and Protocols*, pp.314-329.

[16] Jorge Nuevo, INRS – Université du Québec "A Comprehensible GloMoSim Tutorial", March 26 2003.

[17] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A Feedback based Scheme for Improving TCP Performance in Ad-Hoc wireless networks," *in Proc. of the International Conference on Distributed Computing Systems (ICDCS'98)*, Amsterdam, Netherlands, May 1998

[18] K.Sundaresan, V.Anantharaman, H-Y. Hsieh and R. Sivakumar, "ATP:Reliable Transport Protocol for Ad-Hoc Networks", *Proceedings of 4th ACM International Symposium on Mobihoc*, May 2003

[19] Kaixin Xu, Sang Bae, Sungwook Lee, Mario Gerla, "TCP Behavior across Multihop Wireless Networks and the Wired Internet," Computer Science Department, UCLA, WoWMoM'02, September 28'2002, Atlanta, Georgia,USA.

[20] KARN, P. and Partridge, C. Nov. 1991. "Improving round trip time estimates in reliable transport protocols. *ACM Transactions on Computer Systems* 9, 4, 364-373.

[21] M.Gerla, K.Tang, and R.Bagrodia,"TCP Performance in Wireless Multi-Hop Networks", *Proceedings of IEEE WMCSA 1999*, pp. 41-50, February 1999.

[22] N. Bambos, S. Chen, and G. Pottie, "Radio link admission algorithms for wireless networks with power control and active link quality protection," *in Proc. IEEE INFOCOM,* Boston, MA, 1996.

[23] Piyush Gupta, P.R. Kumar, "The Capacity of Wireless Networks" *IEEE Transactions on Information Theory*, vol 46, No.2, MARCH 2000.

[24] Rajive Bagrodia, Richard Meyer, Yu-an Chen, Xiang Zeng, Jay Martin, Ha Yoon Song "Parsec: A Parallel Simulation Environment for Complex Systems", UCLA October 1998.

[25] Richard A. Meyer and Rajive Bagrodia "PARSEC User Manual", Release 1.1, UCLA Parallel Computing Laboratory, August 1998.

[26] S. Kopparty, S. Krishnamurthy, M. Faloutous, and S. Tripathi, "Split TCP for Mobile Ad Hoc Networks," in *Proc. of IEEE GLOBECOM,* Taipei, Taiwan, Nov. 2002.

[27] S. Ulukus and R. Yates, "Stochastic power control for cellular radio systems", Preprint, 1996.

[28] T. Goff, N. Abu-Ghazaleh, D. Phatak, and R. Kahvecioglu, "Preemptive routing in ad hoc networks," *in Proc. of ACM MOBICOM, Rome, Italy, 2001*, pp. 43–52.

[29] Xiang Zeng, Rajive Bagrodia, Mario Gerla, "Glomosim: A Library for Parallel Simulation of Large –Scale Wireless Networks" PADS, 1998